Continuous-time Parameter Identification Using PI Controllers

Pasi Airikka

Metso Corporation, Tampere, P.O Box 237, FIN-33101
FINLAND (Tel: +358-40-5872730; e-mail: pasi.airikka@metso.com).

Abstract: For several decades, PID (Proportional-Integral-Derivative) control has succesfully been applied to numerous industrial processes for regulating them. The PID controller has indisputably become a backbone of every automation system being available for controlling a large scale of industrial processes. However, it has also other capabilities than just that for regulating processes by manipulating actuators. A PI controller can be designed to identify continuous-time parameters for a given process model using recorded real process data. This paper proposes a PI controller -based identification method for identifying parameters of simple continuous-time linear and time-invariant single-input single-output process models. The method is presented and simulation examples are shown to address its applicability for identification problems.

Keywords: PID control, identification, continuous-time.

1. INTRODUCTION

For numerical applicability and simplicity, most of the available system and parameter identification methods known in literature are given in discrete-time domain. The discrete parametric models can be characterised in different ways depending on the model structures and chosen inputs and outputs. There are several rather thorough references on system identification such as the books by Söderström and Stoica (1989) and Ljung (1999).

Later, there has been research on developing identification methods for continuous-time systems e.g by Young (2002) and a bit earlier by Garnier and Mensler (2000) which was later updated by Garnier et. all (2006). A couple of years later, the methods and numerical routines were collected in the book by Garnier and Hugues (2008). However, several other transient-based or numerically simple methods had been developed and reported earlier e.g by Chen (1989) and Åström and Hägglund (1995).

A PID controller has a fundamental position in process control. It has spread basically everywhere as a relatively simple numerical routine to be implemented. Consequently, it is available practically in every automation system and it is applied and adopted to various and sometimes rather different control problems. It is a bread-and-butter tool and a true workhorse for every automation engineer.

Both Åström (2000) and Visioli (2012) have published articles where they have presented their views on the future trends of PID control research. It is quite interesting to see that in these publications, a PID controller is solely considered as a processs controller without considering any other functions that PID controllers might have. Vilanova and Visioli (2012) has neither expanded this control-oriented view in their latest, rather elaborate, book on PID control.

In addition to its control usage, a PI controller can be used as a workhorse for identifying continous-time system parameters. The basic idea on how to use a PI controller for model parameter estimation has been given in Friman and Airikka (2012). In their publication, the PI controller was streched to work as a numerical routine for providing model parameters for a dynamic simulator the models of which are being updated by using PI controllers executed parallel to a DCS system.

The PI controller has an interesting resemblance with a common recursive parameter estimation formula where a previous parameter estimate is updated using an estimation error amplified by an estimator gain. This observation gives rise to a belief that perhaps a PI controller can be used for system identification. To accomplish that, the idea is simply to feed the identification cost function subject to minimisation to a PI controller that provides a new parameter estimate for the given model type at each execution cycle. Then, the process model is updated by the parameter estimate to generate a new process model output for the next execution cycle. Finally, the parameter estimate converges if certain assumptions hold.

Figure 1 illustrates the concept of using a PI controller for parameter identification. The process measurements (input u and output y) are taken at each identification execution cycle to a system model block. The continous-time system model block generates an estimated process output $\hat{y}$ using the given updated process data and the latest model parameter estimate θ. The modelling error (residual) $e = y - \hat{y}$ is then taken to a cost function block that calculates the identification cost function J based on the residual e. Given a zero setpoint $r = 0$ and the cost function value J as a measurement $y_{PI} = J$, the PI controller yields an output which is the new model parameter estimate. The updated estimate is taken to the system model to produce a new model output for the next execution cycle.

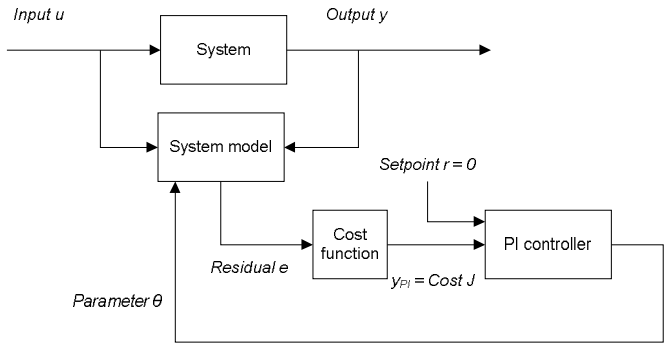

Figure 1. Principle of using a PI controller for SISO (Single-Input-Single-Output) system model parameter identification.

In PID control design, one PI controller can basically manipulate only one variable at a time. The same applies to identification as illustrated in figure 1. A single PI controller allows to identify only one model parameter at a time. However, there are a few ways to bypass this restriction for allowing simultaneous identification of more than one parameter. These methods are discussed in the next chapter 2.

There are many things to be considered when designing PI controllers for system identification. Most of them are of general nature and, therefore, apply to any system identification method. For example, these common design issues are selection of model type to be fitted into process data, selection of model order, process data filtering and process excitation to allow sufficiently rich process data for identification.

This paper proposes some simple cost function types to be used for system identification. At its simpliest, the cost function block can be treated as a bypass gate with $y_{pi} = e$. In that case, the PI controller receives a model residual as a measurement for producing the model parameter estimate of θ. Selection of appropriate parameter for identification is discussed with limitations on model types. The PI controller tuning is dealt with some general practicalities on e.g controller saturation and control signal scaling.

The paper introduces the proposed method making comparison to a common recursive parameter estimation method. Limitations of the proposed method are addressed and some design guidelines are given. The method is enlightened through a few simulation examples.

2. PI CONTROLLER AND PARAMETER ESTIMATION

A PI controller in parallel form can be expressed as

$$\theta(t) = k_p e(t) + k_i \int_0^t e(\tau) d\tau \tag{1}$$

where $\theta(t)$ is controller output, $e(t) = r(t) - y(t)$ is control error between setpoint $r(t)$ and measured output $y(t)$ and proportional gain k_p and integral gain k_i are the PI controller parameters.

After derivating both sides of the PI controller (1) with respect to time *t*, the following is obtained

$$\frac{d\theta(t)}{dt} = k_p \frac{de(t)}{dt} + k_i e(t) \tag{2}$$

Now, by approximating $d\theta(t) \approx \Delta\theta(t)$, $de(t) \approx \Delta e(t)$ and $dt \approx \Delta t > 0$, the PI controller can be expressed in its incremental form

$$\begin{aligned} &\frac{\Delta\theta(t)}{\Delta t} = k_p \frac{\Delta e(t)}{\Delta t} + k_i e(t) \Leftrightarrow \\ &\Delta\theta(t) = k_p \Delta e(t) + k_i \Delta t e(t) \end{aligned} \tag{3}$$

In (3), $\Delta\theta(t)$ represents the PI controller output change between two consequtive execution periods $\Delta\theta(t) = \theta(t_k) - \theta(t_{k-1})$ for a constant control execution period $\Delta t = t_k - t_{k-1}$. Now, if the incremental PI controller (3) is reduced to an integral controller by setting k_p = 0, the following equation is obtained

$$\begin{aligned} &\Delta\theta(t) = k_i \Delta t e(t) \Leftrightarrow \\ &\theta(t_k) = \theta(t_{k-1}) + k e(t_k) = \theta(t_{k-1}) + k\big(r(t_k) - y(t_k)\big) \end{aligned} \tag{4}$$

with $k = k_i \Delta t$.

The equation (4) is actually a common update equation used in recursive parameter estimation. By applying (4), the previous estimate $\theta(t_{k-1})$ is updated based on an estimation error $e(t)$ between the targeted process output $r(t)$ and its estimated output $y(t)$. The update equation also has a gain k which is a tuning parameter allowing a trade-off between convergence speed and robustness to measurement noise.

In parameter estimation (fig. 1), the setpoint given to the PI controller is set $r = 0$. The feedback signal y_{pi} to the PI controller is the cost function value J that is subject to minimisation. Consequently, the PI controller output for parameter estimation can be given as

$$\theta(t_k) = \theta(t_{k-1}) - k y(t_k) = \theta(t_{k-1}) - k J(t_k) \tag{5}$$

It has been now shown that the I controller (or a PI controller with k_p = 0) has a dual representation in recursive parameter estimation. This encourages to continue with the idea of using a PI controller for parameter estimation. However, there are several questions that rise as a result.

2.1 Cost function

In (5), the cost function J taken to the controller could be the model residual $e = y - \hat{y}$. However, as illustrated in figure 1, the cost function could be anything that describes the model mismatch in that particular case. However, for avoiding complexity, the cost function is suggested to be of form

$$y(t) = J(t) = sign\{e(t)\} \cdot |e(t)|^p,\ p = 0.5, 1, 2 \qquad (6)$$

where $e(t) = y(t) - \hat{y}(t)$ is the residual between the process output $y(t)$ and the model output $\hat{y}(t)$, $sign(e)$ is the sign of the residual e. Selection of p = 0.5 results in a square root of the residual, p = 1 to the residual and p = 2 in a squared residual. Other cost functions, such as MSE (Mean of Squared Error) or standard deviation of the error can be equally used but they require a buffer of recorded values for calculating them.

2.2 Proportional control in parameter estimation

It was shown that an integral controller formula has a lot in common with a recursive parameter estimation update equation being actually its dual representation. However, the duality proven in (4) does not include a proportional control. Now, the question rises if the proportional control could be used if it is even needed to be used at all for parameter identification.

The idea of applying the proportional control for estimation is exactly the same than that of process control. By introducing the proportional control and by tuning it more aggressive, the closed loop system can be speeded up. Similarly, identification and parameter convergence is expected to be speeded up by introducing the proportional gain. However, there is a clear drawback of doing that as it is shown in an example given later in this paper.

2.3 Controller tuning for parameter estimation

The PI controlled process in parameter estimation is the system model and its cost function calculation as illustrated in figure 2. Having setpoint r = 0, the PI controller receives the cost function J as a measurement for calculating the parameter estimate θ for the system model. Then, having the parameter value θ as a generated control signal, the system model provides a residual e and, eventually, a cost function value J.

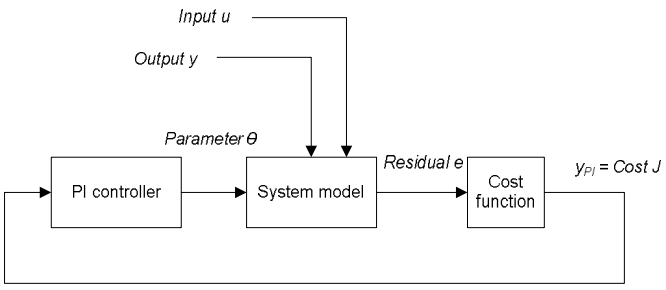

Figure 2. PI controller oriented approach to parameter estimation.

The controllable process (system model + cost function) can be linear or non-linear and from simple to complex. Luckily, the controllable process does not include dead times as the system output and the residual-based cost function J can be calculated within the same execution period for the generated PI controller output θ. Therefore, no compensation for dead time is required. Yet, the PI controller is at its best for linear processes which then encourages to use the PI controller for estimating parameters only for simple process models with only a few parameters to be identified.

Real process variables, that is, process input u and process output y, act as measured load disturbancies when using a PI controller for parameter estimation (figure 2). In real process control, load disturbances are quite often undesired guests that need to be compensated. In this context, they are warmly invited guests for carrying sufficiently rich information for parameter estimation.

Closed loop stability and robustness of the closed loop system (fig. 2) for parameter estimation need to be considered. The system model and the cost function together determine the dynamics that must be carefully addressed in PI controller design. The PI controller tuning is not treated in this paper but, instead, it is advised to apply an auto-tuning technique to tune a PI controller as proposed by Friman and Airikka (2012).

2.4 Selection of estimation parameter

The process model subject to parameter estimation can be basically anything: e.g static or dynamic, time-based or frequency-based, linear or non-linear. Despite the model type, the parameter for estimation should always be selected in such a way that it has an impact on the process output. General identification principles and guidelines equally hold in this context as well.

In process controls, a single PI controller can have one manipulated variable for regulating one controlled variable. Fundamentally, the same principle holds for a PI controller in parameter estimation as well. By manipulating one model parameter, the system model output can be regulated to match the real process output. Luckily, there are some ways to bypass loosen this restriction.

Consider a following static process model of two unknown parameters a and b to be fitted in input-output data $\{y(t),u(t)\}$

$$y(t) = au(t) + b \qquad (7)$$

Selecting the coefficient a to be manipulated by a PI controller, the coefficient $\hat{a}(t_k) = \hat{\theta}(t_k)$ is updated at each control cycle allowing also estimation of the constant b

$$\hat{b}(t_k) = y(t_k) - \hat{a}(t_k)u(t_k) = y(t_k) - \hat{\theta}(t_k)u(t_k) \qquad (8)$$

As a result, two model parameters can be estimated for a single-input single-output model using only one PI controller. A number of identifiable parameters using only one PI controller can also be increased by sharing the PI controller output for different model parameters

$$\theta(t) = a\theta(t) + (1-a)\theta(t) = \theta_1(t) + \theta_2(t), \quad \text{for } 0 < a < 1 \qquad (9)$$

The idea of sharing the same PI controller output for two parameters is illustrated in figure 3. The proposed structure, however, assumes that the parameters have rather equal

parameter value ranges like two time constants of a process model might have.

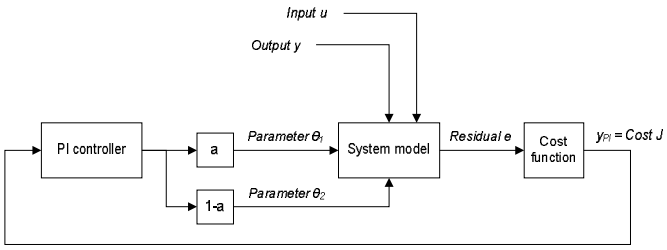

Figure 3. Several parameters can be estimated using one PI controller if the controller output is divided for different parameters.

2.5 Multivariable models

The proposed method allows parameter estimation for a SISO model. This paper does not suggest detailed methods for dealing multi-input multi-output (MIMO) or not even multi-input single-output (MISO) models with several process inputs. Yet, by having several cost functions for parameter estimation, such as separate residuals for each process output, a number of the PI controllers can be increased to cover more model parameters for identification.

2.6 Some practicalities

Signal filtering and detrending of data is of essence for parameter estimation. Unwanted frequencies and offsets should be filtered out before sampling process data. Noise filtering is equally important for parameter estimation using a PI controller than that for process control. For succesful signal filtering, appropriate measures are recommended for dealing with noise effects in parameter estimation.

Parameter saturation, that is, control signal saturation must be considered when applying a PI controller containing integration. First of all, for not allowing goofy parameter values, the admissible parameter range $\theta_{\min}..\theta_{\max}$ should be used for limiting the PI controller output. Also, appropriate measures for avoiding integrator anti-windup should be taken not to risk parameter estimation convergence speed due to integral windup.

For practical reasons, it is convenient to deal with a limited PI controller output range, e.g $-1 \le u_\theta(t) \le 1$. Then, a parameter estimate θ is obtained by using the allowable scale $\Delta\theta = \theta_{\max} - \theta_{\min}$ and its minimum value $\theta_{\min}$ as follows

$$\theta(t) = \theta_{\min} + (1 + u_\theta(t))\frac{\Delta\theta}{2} \tag{10}$$

A good initial parameter estimate $\theta_0 = \theta(0)$ reduces estimation effort also in PI controller -based parameter estimation. Process impact direction is essential to know a priori for a PI controller also in parameter estimation. An increasing change in a PI controller output θ may have either an increasing or a decreasing change in the cost function *J*. The PI controller is given this information for enabling correct control actions for succesful parameter estimation.

3. SIMULATION DERIVED RESULTS

3.1 Simple static process with gain

Consider a simple static process model

$$y(t) = ku(t) + \varepsilon(t) \tag{11}$$

where gain $\theta = k = 0.5$ is to be estimated from input-output data of $\{y(t),u(t)|t \ge 0\}$ with white noise $\varepsilon(t)$. The PI controller is designed with $k_p = 0$ and $k_i = 0.002$ and the cost function is the pure residual $e(t) = y(t) - \hat{y}(t) = y(t) - \hat{k}u(t)$.

Figure 4 illustrates input-output data, cost function *J*, estimated gain $\hat{k}$ and PI controller output u_θ with respect to time. The parameter scale is $\Delta k = 2$ with $k_{\min} = 0$ and the initial value is $k_0 = 1$. As a result, the estimated gain converges to 0.5 in less than two minutes.

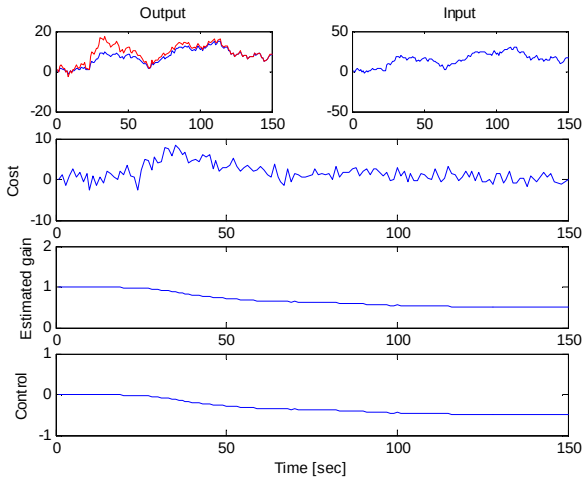

Figure 4. Convergence of static gain of process (11). *Top*: process data: output (*left*), input (*right*). *Second top*: cost function. *Second bottom*: estimated gain. *Bottom*: control signal.

3.2 Controller tuning and cost functions

Figure 5 compares PI controller tunings (k_p = 0) for different integral gains k_i = 0.002, 0.004, 0.008 and 0.016. Every time the tuning is tightened by doubling the integral gain, the estimated parameter converges faster. However, with k_i = 0.016, the estimate starts introducing noise.

Figure 6 shows the impact of having an unchanged PI tuning but different cost functions (5) with *p* = 0.5 (square root), 1 and 2 (squared). It can be seen in fig. 6 that by increasing the value of *p*, the convergence gets faster but with increasing noise.

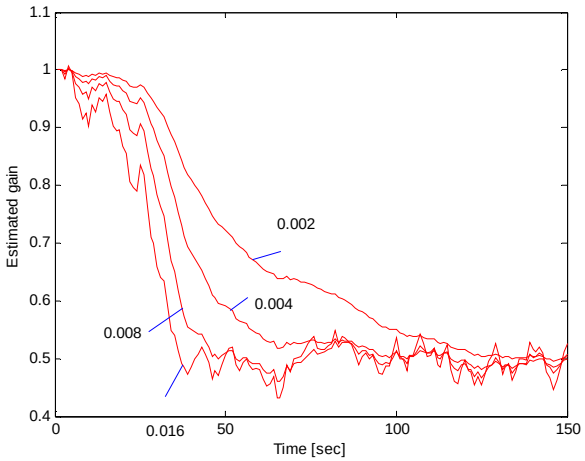

Figure 5. Convergence of estimated parameter for different integral controller gains k_i = 0.002, 0.004, 0.008, 0.016.

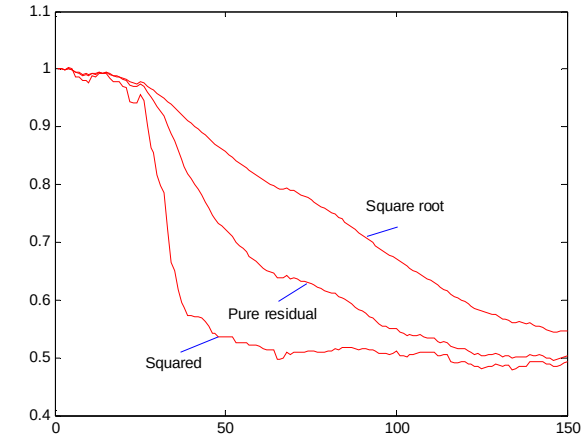

Figure 6. Convergence of estimated parameter for various cost function types (squ root p = 0.5, p = 1, squared p = 2).

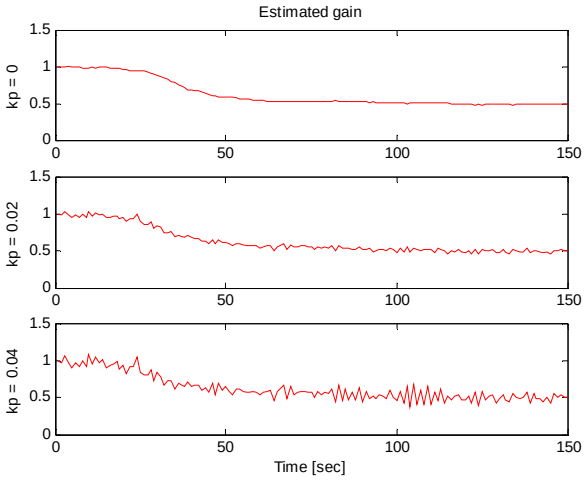

Figure 7. Convergence of estimated parameter for different proportional gains. *Top*: k_p = 0. *Middle*: k_p = 0.02. *Bottom*: k_p = 0.04.

Figure 7 illustrates the impact of having a proportional control involved. When having no proportional control (k_p = 0 and k_i = 0.002), the yielded estimate is rather smooth. By introducing the proportional control (k_p = 0.02), the estimate does not converge much faster but introduces some noise. By doubling the proportional gain (k_p = 0.04), noisiness increases. The proportional part clearly amplifies the modelling error but does not contribute greatly to the parameter estimate itself.

3.3 Dynamic process with time constant and gain

Next, consider a dynamic first order system with static gain k and time constant T

$$y(s) = \frac{k}{Ts+1}u(s) + \varepsilon(s) \tag{12}$$

The time constant T is selected to be estimated by a PI controller. Then, using (12), the static gain k can be solved as

$$Tsy(s) + y(s) = ku(s) + \varepsilon(s) \Rightarrow k = \frac{Tsy(s) + y(s)}{u(s)} \tag{13}$$

Consequently, the PI controller generates the time constant estimate $T(t)$ at each execution cycle, after which, the gain estimate can be updated by $k(t) = \left(T(t)\dot{y}(t) + y(t)\right)/u(t)$ where $\dot{y}(t)$ is time derivative of $y(t)$. Assume the process (12) with k = 0.5 and T = 20 sec. The parameter scale for T is $\Delta T = 50$ with $T_{\min} = 0$ and the parameter is initialised as $T_0 = 100$. The PI controller is designed with $k_p = 0$ and $k_i = 0.5$. Figure 8 shows results with cost function J and estimated parameters T and k with respect to time. It takes approximately two minutes for the PI controller output to settle and produce a correct time constant estimate T = 20 but less time for the static gain to reach its correct value k = 0.5.

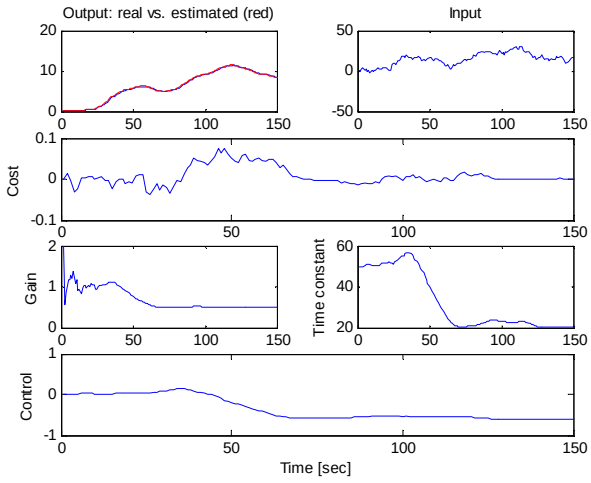

Figure 8. Convergence of model (12) parameters. *Top*: process data (left: output, right: input). *Second top*: cost function. *Second bottom*: estimated gain (*left*) and estimated time constant (*right*). *Bottom*: control signal.

4. CONCLUSION

This paper proposed a continuous-time parameter estimation method using an industrial PI controller. The method seems to be applicable to simple SISO system models with one or two model parameters to be identified. The proposed method can be explained by comparing it to a general recursive parameter estimation method which is shown to be a dual representation of the proposed method.

From a practical point of view, the method can be justified by its reliance on the industrial PI controller which is the only numerical routine required for identification. PI controllers are basically everywhere and easily available for system identification purposes. A control engineer is familiar with a PI controller making it rather easy to adopt the proposed method for system identification.

The proposed method is limited to simple process models and, similarly, simple model residual –based cost function formulations were suggested to avoid complexity of the dynamics of identifiable parameters in the PI controlled closed identification loop. For PI controller tuning, auto-tuning methods are proposed to make the control design less troublesome.

It was shown that using only an integrating controller (I) can be adequate for parameter estimation. The proportional controller (P) may speed up parameter convergence but, unfortunately, at the cost of noise amplification. And to recover from cases where the zero value of the cost function may not be reached at all, the PI controller must have an integrator anti-windup feature implemented to avoid excessive controller output saturation.

A good initial parameter estimate is useful along with an appropriate parameter selection for estimation. The parameter should be selected in a way to avoid strongly non-linear dependence between the estimated parameter and its impact on the selected cost function. After all, the process to be controlled by the PI controller for parameter estimation consists of the model structure and the cost function. Obviously, this justifies the selection of a simple model structure and a simple cost function.

The suggested method is equally applicable to on-line and off-line system identification. An interesting application of the proposed method is to apply it for updating critical process models of an operator training simulator (OTS) using existing real process data. By retrieving recorded process data from a process data archive or reading it on-line, selected system models of the simulator can be updated without interfering the real process. This can be done using a method given in this paper totally in parallel with a real process control system but without affecting the real control system or real processes as the communication takes place only in one way: from the real process to the OTS system having PI controllers for identifying critical process parameters.

REFERENCES

Chen, C.L (1989). *A simple method for on-line identification and controller tuning,* AIChE Journal, 35 (12), pp. 2037-2039.

Friman M., P. Airikka (2012), Tracking Simulation Based on PI Controllers and Autotuning, *IFAC Conference on Advances in PID Control (PID'12)*, Brescia, Italy.

Garnier H., M. Mensler (2000). The CONTSID toolbox: a Matlab toolbox for CONtinuous-Time System Identification, *12th IFAC Symposium on System Identifcation*, Santa Barbara, USA.

Garnier H., Gilson M., Cervellin O. (2006), Latest developments for the Matlab toolbox CONTSID, *the 14th IFAC Symposium on System Identification*, SYSID 2006, Newcastle, Australia, pp. 714-719.

Garnier, Hugues et. al (2008), *Identification of Continuostime Models from Sampled Data*, Springer, 1st edition, 413 p.

Ljung L. (1999), *System identification*. Theory for the user, 2nd edition, Prentice Hall.

Söderström, T., P. Stoica (1989), *System Identification*, Prentice-Hall, UK, 612 p.

Vilanova, R., A. Visioli (2012), *PID Control in the Third Millennium: Lessons Learned and New Approaches*, Springer-Verlag, 880 p.

Visioli, A. (2012), Research Trends for PID Controllers, *Acta Polytechnica*, Vol. 52 No. 5.

Young P.C (2002), Optimal IV identification and estimation of continous time TF models, *15th IFAC World Congress*, Barcelona, Spain.

Åström K.J (2002), The Future of PID Control, *IFAC Congress*, Barcelona, Spain.

Åström K.J, T. Hägglund (1995), *PID Controllers: Theory, Design and Tuning*, 2nd ed., Instrument Society of America, USA.