IFAC Conference on Advances in PID Control
PID'12
Brescia (Italy), March 28-30, 2012 ThB2.5

Dead-band self-triggered PI control for processes with dead-time

U. Tiberi *, **C.F. Lindberg** **, **A.J. Isaksson** **

* *ACCESS Linnaeus Center, KTH Royal Institute of Technology, Stockholm, Sweden (e-mail:ubaldot@ee.kth.se)*
** *ABB Corporate Research, Västerås, Sweden (e-mail:{carl-fredrik.lindberg,alf.isaksson}@se.abb.com)*

Abstract: Current implementations of digital controllers assume that sensing, control and actuation are performed in a periodic fashion. In classic control schemes, where sensors and controllers are directly connected, periodicity does not provide particular drawbacks, but, in the case of wireless sensor networks, such a choice may be questionable. One of the driving constraints in the design of wireless sensor networks is represented by its energy efficiency, and it has been shown that the main cause of energy consumption is due to the radio activities of the sensor nodes. By using periodic implementations, the sensor nodes are enforced to keep on transmitting measurements to the controller even if it is not really needed, thus wasting energy. To cope with these problems, self-triggered control was recently introduced. This technique aims at reducing the conservativeness of periodic implementations providing an adaption of the inter-sampling intervals based on the current output of the system. Existing work on self-triggered control considers linear systems controlled by state feedback controllers under the assumption of small time-delays. In this paper the problem of designing a self-triggered control scheme that applies to first-order processes with large dead-times controlled by PI controllers is addressed. Moreover, the proposed self-triggered scheme is robust with respect to set-point changes and external disturbances, which are typical in process industry. The results are validated by simulations.

Keywords: KLT Process, Self-triggered Control, PI Controllers, Wireless Sensor Networks

1. INTRODUCTION

The emergence of wireless sensor networks (WSNs) is enabling the development of new applications in various engineering fields, including environmental monitoring, smart grids, process industry, health care applications etc. Willig [2008], Ploplys et al. [2004]. The main benefits offered by such networks rely on their configuration flexibility, low cost and on their facility of deployment and maintenance. The introduction of such technology in process industry is promising substantial improvements of the existing applications, in addition to a strong reduction of both installation and maintenance costs. Nevertheless, despite the benefits provided by WSNs, new problems related to reliability, time delays and energy efficiency arise, Park [2011]. While reliability and time delay problems are present in any networked control system, Hespanha et al. [2007], in the case of WSN there is the additional problem of energy consumption, being the nodes usually battery powered. Since the main cause of energy consumption is due to the radio activities of the nodes, Texas-Instruments [2007], an energy efficient control strategy should reduce as much as possible the amount of communication among the nodes, while achieving a desired behavior of the closed-loop system.

To cope with these problems, in the last years event-triggered control has been introduced Årzen [1999], Tabuada [2007], Dimarogonas and Johansson [2009], Åström and Bernhardsson [2002], Velasco et al. [2003], Mazo et al. [2010], Anta and Tabuada [2010], Wang and Lemmon [2009]. In event-triggered control schemes, the sensor nodes continuously measure the process output and they send a new measurement to the controller only when a function of the output crossed a certain threshold. Hence, event-based control is a *reactive* control paradigm, since the controller is updated after an event is detected. Alternatively to event-triggered control, in the last years, self-triggered control has been also introduced. Instead of reacting to the detection of an event, self-triggered control *predicts* the occurrence of such an event based on the current measurement and on a model of the process, it is then a *predictive* control paradigm.

Both the methods present benefits and drawbacks, and the utilization of event or self-triggered control schemes in a WSN context shows relevant differences. For instance, event-based control provides a natural robustness with respect to external disturbances and set-point changes, since the system output is constantly monitored. Moreover, event-based schemes, in general, do not require the knowledge of the process model, but are based only on the output measurements. However, if there are packet dropouts the controller is not able to detect it. On the other side, self-triggered control provides a deadline by which the next measurement should be transmitted and received at every transmission time. Thus packet dropouts can be easily identified. Nevertheless, since self-triggered strategies may allow the system to run in open loop for long time, if there is a set-point change or if a disturbance suddenly enters the system, such problems can be handled only in the next transmission

[1] The work of the authors was supported by the VINNOVA project WiComPI, the Knut and Alice Wallenberg Foundation and the FeedNetBack and Hycon2 EU projects.

time. In addition, since the next transmission time is predicted, self-triggered schemes require a model of the controlled process as well. Finally, while event-based control implicitly suggests the utilization of contention based communication protocols, another advantage with self-triggered control is that more deterministic time-slotted protocols may be applied.

However, in both cases of event and self-triggered control, it has been shown that it is possible to save over 90% of communication between the controller node and the sensor node still ensuring acceptable performance of the closed loop system. See Norgren and Styrud [2011], where a first version of our self-triggered scheme was tested at a Swedish paper mill.

Focusing on a process industry perspective, most industrial processes are well represented with first-order models with dead-time (KLT processes) and typically the controllers are PI, Bialowskii [1992]. The study of event-based PI controller provided fertile ground of research in the recent years, Årzen [1999], Song et al. [2006], but to the best of our knowledge, self-triggered strategies that target the specific case of first-order processes and PI controllers is not yet addressed.

In this paper we address the specific case of designing a self-triggered strategy based on the dead-band sampling, Otanez et al. [2002], that applies for first-order processes (with and without dead-time) controlled by PI controllers. The proposed self-triggered strategy has the task of scheduling the transmissions from the sensor to the controller, and it must be capable of handling set-point changes as well as unmeasured external disturbances, typical in process industry. We design a self-triggered sampler by assuming that the controller updates are performed only at the transmissions instants. The results presented in this paper are inspired by Tiberi et al. [2010]. However Tiberi et al. [2010] considered only static controllers, assumed time-delays were shorter than the inter-sampling times and that the external disturbance was constant. Here PI controllers are considered with time-delays that are possibly larger than the inter-sampling times and the sampling scheme is improved by a disturbance observer.

The paper is organized as follows: the next section introduces some preliminaries and the problem is formulated. In Section 3 the considered system architecture is described. In Section 4 a self-triggered sampler for KLT processes is presented, and integrating processes are treated as a special case. In Section 6 the proposed self-triggered strategy is validated by simulation, and, finally, a discussion in Section 7 concludes the paper.

2. PRELIMINARIES AND PROBLEM STATEMENT

Nowadays control architectures usually consider continuous-time processes controlled by PI controllers implemented on digital devices. Between two consecutive controller update, the control signal is generally kept constant. In process industry a common mathematical model used to represent several different processes is the KLT model given by

$$P(s) = \frac{K}{1+Ts}e^{-Ls}, \tag{1}$$

where $K \in \mathbb{R}$ is the static gain, $T \in \mathbb{R}$ is the time constant and $L \in \mathbb{R}$ is the process time delay.

Another common process is the integrating processes described by

$$P(s) = \frac{K_v}{s}e^{-Ls}, \tag{2}$$

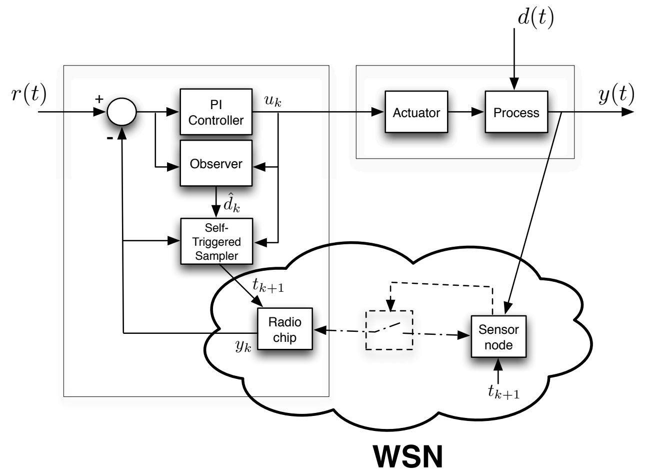


Fig. 1. The proposed system architecture presents a WSN, disturbance observer and a self-triggered sampler. The next transmission time is decided by the controller and it is communicated to the sensor, which between two consecutive transmission times can turn the radio off to save energy.

where $K_v \in \mathbb{R}$ is the integration speed.

Rewriting (1) into state-space form and adding an external disturbance gives

$$\begin{aligned} \dot{x}_p(t) &= ax_p(t) + bu(t-L) + d(t)\,, \\ y(t) &= x_p(t)\,, \end{aligned} \tag{3}$$

where $x_p \in \mathbb{R}$ is the state of the process, $u(t) \in \mathbb{R}$ is the process input, $y(t) \in \mathbb{R}$ is the output, $d(t) \in \mathbb{R}$ is a non-measurable bounded external disturbance, $a = -1/T$, and $b = K/T$. A state-space representation of the integrating process (2) is obtained by setting $a = 0$ and $b = K_v$ in (3).

PI-controllers are usually implemented on digital devices, here the following implementation is used

$$\begin{aligned} x_c(t_k) =& x_c(t_{k-1}) + T_s(r(t_k) - y(t_k)), \\ u(t_k) =& K_p\Big((r(t_k) - y(t_k)) + \frac{1}{T_i}x_c(t_k)\Big), \end{aligned} \tag{4}$$

where $x_c(t_k) \in \mathbb{R}$ is the integrator state, $T_s = t_k - t_{k-1}$ is the sampling period, $r \in \mathbb{R}$ is the set-point signal, $K_p \in \mathbb{R}$ is the proportional gain and $T_i \in \mathbb{R}$ is the integral time. With a slight abuse of notation, in the rest of the paper we also indicate with z_k the sampled value of a signal $z(t)$ at time t_k, i.e. $z_k = z(t_k)$.

A natural way to employ a WSN in the feedback channel in the control architecture described so far, is to perform periodic transmissions of the output measurements to the controller every T_s units of time. As previously discussed, such an implementation may require a large amount of communication between the sensor and the controller if T_s is small, resulting in a large energy consumption of the nodes of the WSN. Clearly, the number of transmissions can be reduced by using larger inter-transmission periods. However, the utilization of large inter-transmission periods could lead to instability, to bad performance if external disturbances suddenly enter the system, or long delays in case there are set-point changes.

In this paper we address the problem of designing a self-triggered sampler that applies to KLT and integrating processes controlled by PI controllers. The self-triggered sampler implements the deadband sampling, namely, given a measurement of the output y_k at time $t = t_k$, the self-triggered sampler should predict the time by which the triggering condition

$$|e(t)| := |y_k - y(t)| \geq \delta \tag{5}$$

is fulfilled, where $\delta > 0$ is a design parameter. Moreover, the self-triggered scheme should be capable to accommodate set-point changes and non-measurable external disturbances.

3. PROPOSED SYSTEM ARCHITECTURE

The proposed system architecture is depicted in Fig 1. Such architecture comprises a process that can be either KLT or integrating process, a PI controller and a WSN used for the exchange of information between the controller and the sensor node. The wireless communication is performed in an aperiodic fashion, and the control signal is updated using (4) when a new measurement is available with T_s as the actually elapsed inter-sampling time. On the controller is further implemented a self-triggered sampler and a disturbance observer. We assume that all the computations to determine the next transmission time are implemented at the controller, but it is possible to distribute the computations between the sensor node and the controller as will be discussed later on in the paper. At every transmission instant $t = t_k$, the sensor node sends a fresh measurement y_k to the controller, and after the reception of y_k, the controller performs the following actions

(1) it computes a new control input u_k based on the received measurement y_k;
(2) it estimates the disturbance $\hat{d}_k$;
(3) it computes the next transmission time t_{k+1};
(4) it sends back the value of t_{k+1} to the sensor node;

Once the sensor node has received the value t_{k+1}, it goes to sleep to save energy until the time $t = t_{k+1}$. That is, both the controller and the sensor know the time t_{k+1} at which the next transmission will be performed.

4. SELF-TRIGGERED TRANSMISSIONS POLICY

In this section we describe how to design a self-triggered sampler that implements the triggering rule implicitly defined by (5). We consider the cases of KT, KLT process and integrating processes as a special case. For the sake of clarity, we first assume that the controller updates and the transmissions are synchronized, namely the controller is updated only and exactly at the time when receiving a new measurement from the sensor node. However, in practice the communication network and controller typically are not synchronized. To compensate for the lack of synchronization the controller code is then executed using a fixed (relatively short) period but with updates of the control signal only if a new measurement is available, while the transmissions are still performed according to the self-triggered sampler.

4.1 Processes without time-delay

As preamble consider the state-space representation (3) with $L = 0$, the dynamics of $e(t)$ satisfy $\dot{e}(t) = -\dot{y}(t)$ and at the sampling instants $t = t_k$ it holds $e(t_k) = 0$. Hence, to design the self-triggered sampler we consider the dynamics, for $t \in [t_k, t_{k+1})$

$$\dot{e}(t) = -ay(t) - bu_k - \hat{d}_k = -ay_k + ae(t) - bu_k - \hat{d}_k \,, \tag{6}$$

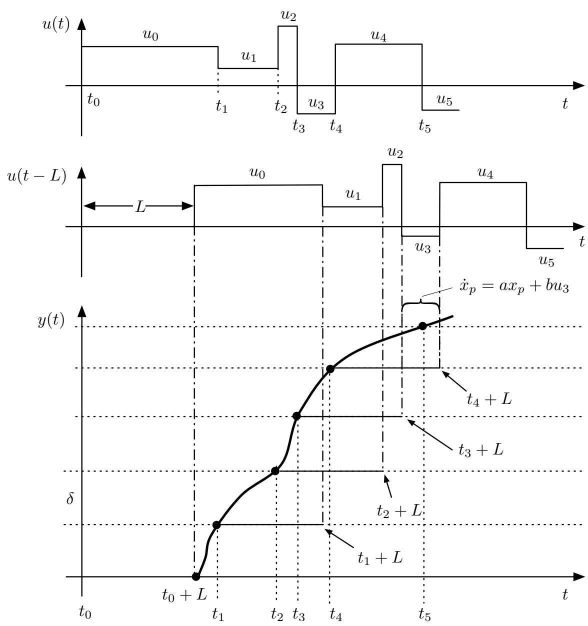


Fig. 2. Example of output response of process with dead-time. Between two consecutive inter-transmission times, there could be input signal changes. For example, the sequence of input acting for $t \in [t_4, t_5)$ are u_1, u_2, u_3.

where $\hat{d}_k$ is the estimate of the disturbance. Such estimate can be computed as described in Section 5. It follows

$$e(t) = -\frac{(ay_k + bu_k + \hat{d}_k)}{a}(e^{a(t-t_k)} - 1)\,, \tag{7}$$

for all $t \in [t_k, t_{k+1})$ from which, by setting $e(t_{k+1}) = \pm\delta$, one derives for stable systems i.e. $a < 0$

$$t_{k+1} = t_k + \frac{1}{a}\ln\left(1 + \frac{a\delta}{|ay_k + bu_k + \hat{d}_k|}\right)\,, \tag{8}$$

that looks similar to the self-triggered sampler proposed in Tiberi et al. [2010]. However, in Tiberi et al. [2010] the self-triggered sampler is designed by considering only proportional state feedback, while within this new scheme, we explicitly use the control input u_k that may come from any dynamic controller, and we use the output y_k instead of the state x_k. In addition, here we do not consider the worst case of the external disturbance, but we use an estimate $\hat{d}_k$ of it to provide a certain degree of robustness to model changes.

When the process is stable, it may happen that the logarithm in (8) does not give a real valued result, which corresponds to $\lim_{t\to\infty}|e(t)| < \delta$. Then a maximum sampling interval $T_{\max}$ has to be introduced. It should be chosen such that for a given tuning K_p, K_i the closed-loop system is still stable for a periodic $T_s = T_{\max}$. Note that asymptotic stability of the discretized closed-loop system prevents steady-state oscillatory behavior of the output around the set-point.

4.2 KLT Processes with time delay

Because of process time delays and the aperiodic sampling policy, the effective input acting on the process after a certain time instant $t = t_k$ is not constant, but is piecewise constant, see Fig. 2. This is due because of the process dynamics, in which there is present a dead-time, and because of the adopted sampling scheme, i.e. the dead-band sampling. To predict the next time by which the triggering condition (5) is violated, it is then useful to consider the control changes during the time interval $[t_k, t_{k+1})$. For instance, the sequence of inputs that will take effect for $t \in [t_k, t_{k+1})$ are $u_{m(k)}, u_{m(k)+1}, \dots u_{n(k)}$, where

$$m(k) := \max_{0\le i\le k} \{i : t_i + L \le t_k\}, \tag{9}$$

$$n(k) := \max_{m(k)\le i\le k} \{i : t_i + L \le t_{k+1}\}. \tag{10}$$

For the sake of clarity, in the sequel we simply indicate $m = m(k)$ and $n = n(k)$. The design of the self-triggered sampler is performed by splitting the process dynamics as

$$\begin{cases} \dot{y}(t) = ay(t) + bu_m + \hat{d}_k & \text{for } t_k \le t < t_{m+1} + L\,, \\ \dot{y}(t) = ay(t) + bu_{m+1} + \hat{d}_k & \text{for } t_{m+1} + L \le t < t_{m+2} + L\,, \\ \quad\vdots & \quad\vdots \\ \dot{y}(t) = ay(t) + bu_n + \hat{d}_k & \text{for } t_n + L \le t < t_{k+1}\,. \end{cases} \tag{11}$$

To predict the next time by which the triggering condition (5) is violated, we exploit the dynamics of $e(t)$. Such dynamics, for the time interval $t_k \le t \le t_{m+1} + L$, are given by

$$\dot{e}(t) = -ay(t) - bu_m - \hat{d}_k = -ay_k + ae(t) - bu_m - \hat{d}_k\,.$$

By taking into account that by definition at the sampling times $e(t_k) = 0$, we have, for $t \in [t_k, t_{m+1} + L)$

$$e(t) = -\frac{(ay_k + bu_m + \hat{d}_k)}{a}(e^{a(t-t_k)} - 1)\,. \tag{12}$$

For a generic time interval $t_{m+i} + L \le t \le t_{m+i+1} + L, i > 0$, the dynamics of the $e(t)$ error satisfy

$$\dot{e}(t) = -ay(t) - bu_{m+i} - \hat{d}_k = -ay_k + ae(t) - bu_{m+i} - \hat{d}_k\,,$$

and $e(t)$ is given by

$$\begin{aligned} e(t) =& e^{a(t-t_{m+i}-L)} \cdot e(t_{m+i} + L) \\ & - \frac{(ay_k + bu_{m+i} + \hat{d}_k)}{a}(e^{a(t-t_{m+i}-L)} - 1)\,, \end{aligned} \tag{13}$$

where $e(t_{m+i} + L)$ is the value of $e(t)$ at time $t = t_{m+i} + L$. Given the equation (13), we define the discrete version of $e(t)$, as

$$\begin{cases} e_0 = 0\,, \\ e_{j+1} = \alpha(j)e_j + \beta(j), \end{cases} \tag{14}$$

where

$$\alpha(j) = \begin{cases} e^{a(t_{m+1}+L-t_k)} & \text{for } j = 0\,, \\ e^{a(t_{m+j+1}-t_{m+j})} & \text{for } 0 < j < n - m\,, \\ e^{a(t_{k+1}-t_{m+j}-L)} & \text{for } j = n - m\,, \end{cases} \tag{15}$$

$$\beta(j) = \begin{cases} -\dfrac{(ay_k + bu_m + \hat{d}_k)}{a}(\alpha(j) - 1) & \text{for } j = 0\,, \\ -\dfrac{(ay_k + bu_{m+j} + \hat{d}_k)}{a}(\alpha(j) - 1) & \text{for } 0 < j \le n - m\,, \end{cases} \tag{16}$$

and where $j \in \{0, 1, \dots, n-m\}$ is a relative index that is reset to zero at every $t = t_k$. To derive the next sampling time t_{k+1}, we consider both the continuous-time dynamics (13), and the discrete-time dynamics (14) of e. By doing that, it is easy to see that the time $t_{k+1} - t_k$ needed to e to go from 0 to δ is given by

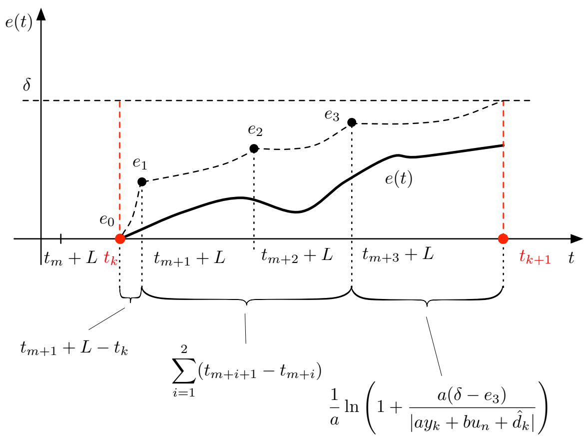


Fig. 3. Example the proposed self-triggered sampler. To determine the next sampling time we use both the discrete and the continuous dynamic of $e(t)$. In this example we have $n - m = 3$.

$$\begin{aligned} t_{k+1} - t_k =& (t_{m+1} + L - t_k) + \sum_{i=1}^{n-m-1}(t_{m+i+1} - t_{m+i}) \\ & + \frac{1}{a}\ln\left(1 + \frac{a(\delta - e_{n-m})}{|ay_k + bu_k + \hat{d}_k|}\right)\,. \end{aligned} \tag{17}$$

The working principle of the self-triggered sampler (17) is depicted in Fig. 3. Notice how the next transmission time t_{k+1} depends on the previous $n - m$ computed input and transmission times. This means that the self-triggered implementation requires a memory to store the values of the previous transmission times performed during the time interval $(t_k - L, t_k]$. Notice also how the next transmission time t_{k+1} can be computed at time $t = t_k$, according to the predictive nature of the self-triggered control. Even here it can happen that $\lim_{j\to+\infty} e_j < \delta$, but such a drawback can be easily avoided by using a $T_{\max}$ as described in the previous section.

4.3 Integrating Processes

For integrating processes we can proceed as in the case of KLT process. In this case (14), becomes

$$\begin{cases} e_0 = 0\,, \\ e_{j+1} = e_j + \beta(j)\,, \end{cases} \tag{18}$$

where $\beta(r)$ is defined as

$$\beta(j) = \begin{cases} -(bu_m + \hat{d}_k)(t_m + L - t_k) & \text{for } j = 0 \\ -(bu_{m+j} + \hat{d}_k)(t_{m+j+1} - t_{m+j}) & \text{for } 0 < j \le n - m \end{cases} \tag{19}$$

The self-triggered sampler (17) becomes

$$\begin{aligned} t_{k+1} - t_k =& (t_{m+1} + L - t_k) + \sum_{i=1}^{n-m-1}(t_{m+i+1} - t_{m+i}) \\ & + \frac{\delta - e_{n-m}}{|bu_n + \hat{d}_k|}\,. \end{aligned} \tag{20}$$

The computations needed to determine the next transmission times are described next. Based on the dynamics (14), the controller computes the value of e_{n-m}, it picks the last stored $n - m$ inter-transmission times and, based on the output measurement y_k and the disturbance estimate $\hat{d}_k$, computes the next transmission time. The value of e_{n-m} is computed by using the discrete time dynamics (14), using an estimate of the disturbance discussed in the next section.

Remark 4.1. Despite we assumed that all the computation are performed by the controller, we may actually distribute such computations between the sensor node and the controller. For example, a more accurate disturbance observer may be implemented at the sensor, which at every transmission time will send to the controller the values of y_k and $\hat{d}_k$, and the controller will reply with the next transmission time t_{k+1}. ◁

Remark 4.2. If the model used by the self-triggered sampler is different from the real process, the methodology still holds. If the sampling rule (5) is implemented, but the model is not exact, what happens in reality is that we are sampling when a triggering condition $|\Delta y(t)| \le \delta + \mu$, is violated, where μ takes into account model mismatch. Clearly, if the model is exact, then $\mu = 0$. Hence, model mismatch is included in the real threshold $\delta + \mu$, but the value of the threshold modifies only the trade-off between the number of transmissions and the closed-loop performance. However, the optimal selection of δ is an open problem, currently under investigation. ◁

5. DISTURBANCE OBSERVER

5.1 A Dead-beat Observer

As noticed above the derived self-triggered sampler assumes that the process disturbance is estimated. To do so, we have to introduce some assumption about the disturbance. Simplest is of course to assume a constant disturbance, and the simplest observer just solves for $\hat{d}$ using two consecutive measurements (which corresponds to a deadbeat observer)

$$\begin{aligned}\hat{d}_k =& \frac{1}{g(t_k, t_{k-1})}\bigg(y_k - f(t_k, t_{k-1})y_{k-1} \\ &- b\sum_{i=0}^{n-m-1} g(t_{m+i+1}, t_{m+i})u_{m+i} \\ &- bg(t_{m+1}+L, t_{k-1})u_m - bg(t_k, t_n + L)u_n\bigg). \end{aligned} \quad (21)$$

where the functions $f(\cdot,\cdot)$ and $g(\cdot,\cdot)$ are defined as

$$f(p,q) := e^{a(p-q)}, \quad g(p,q) := \frac{1}{a}(e^{a(p-q)} - 1)$$

In the case of a process without delay, i.e. $L = 0$, this observer becomes

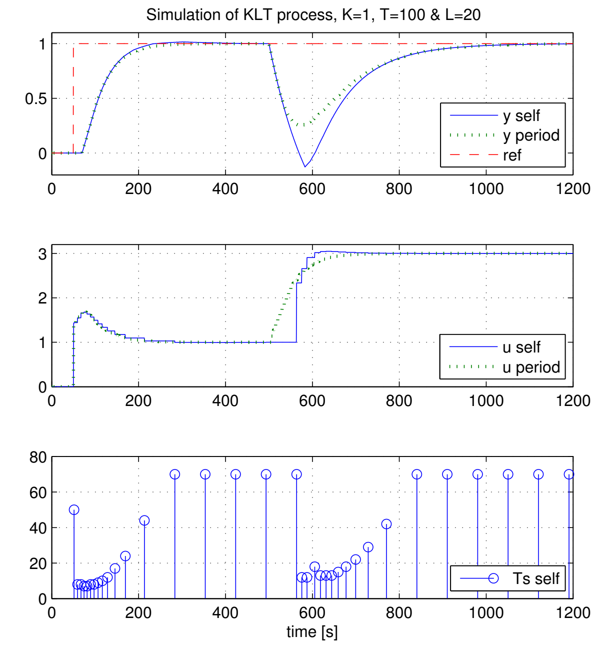
$$\hat{d}_k = \frac{y_k - f(t_k, t_{k-1})y_{k-1} - b\, g(t_k, t_{k-1})u_{k-1}}{g(t_k, t_{k-1})}. \quad (22)$$

This way, we are able to accommodate variations in the external disturbances.

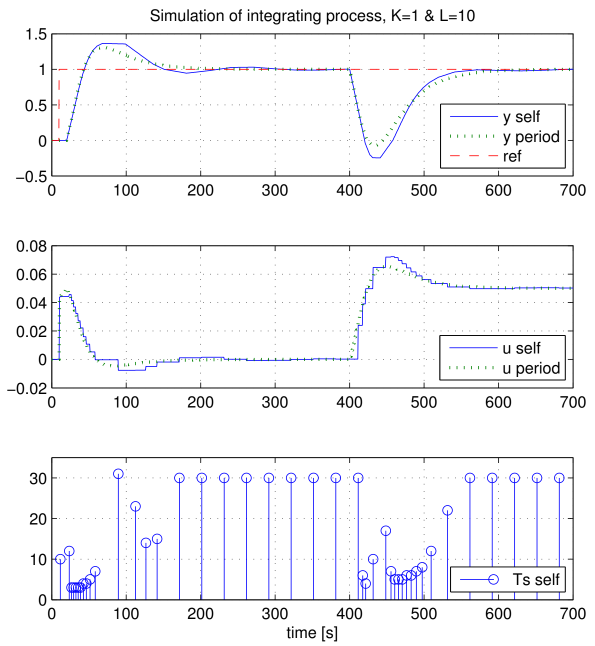
The disturbance observer for an integrating process is the same as in (21) but the functions $f(p,q)$ and $g(p,q)$ are replaced with $f(p,q) = 1$ and $g(p,q) = (p-q)$.

5.2 A Kalman filter based Observer

For noisy situations it may be better to deploy a Kalman filter with d as an extra state with zero derivative. Performing the time update of the Kalman filter becomes almost trivial (at least when T_{max} is not activated), since by construction the predicted next value of the output is $y_k \pm \delta$. A measurement update is then carried out at the next sampling instant using the above defined $f(t_{k+1}, t_k)$ and $g(t_{k+1}, t_k)$ for the update of the Riccati equation.



(a) Simulation of a KLT process for a step in set-point and disturbance.



(b) Simulation of an integrating process for a step in set-point and disturbance.

Fig. 4. Simulation results with self-triggered PI-control.

6. SIMULATIONS

Some simulation examples have been done for a two different processes. In the first example a KLT process with dead-time has been simulated and controlled.

$$G(s) = \frac{1}{100s+1}e^{-20s}$$

There is a change in the set-point after 50s and the disturbance d changes from zero at $t = 500$s. The process is controlled by a self-triggered PI-controller with gain $K_c = 1.43$ and integral time $T_i = 100$. The disturbance has been estimated by a dead-beat observer in the self-triggered algorithm. The result of the simulation is presented in Fig. 4(a). Maximum sampling time has been set to 70 s and minimum to 1 s. The simulation summed up to 36 samples, and most samples are as expected taken when y is changing. Compared to the fast periodic controller sampled by 5s, this strategy saves 85% of the number of samples. The fast periodic controller has however a better disturbance rejection for load disturbances.

In next example an integrating process with no dead-time has been controlled.

$$G(s) = \frac{1}{s}e^{-10s}$$

The set-point is changed at $t = 10$ s and the disturbance d changes from zero at $t = 400$s. The process is controlled by a self-triggered PI-controller with gain $K_c = 0.0437$, and integral time $T_i = 70$. The result of the simulation is presented in Fig. 4(b). The maximum sampling time has been set to 30 s and minimum to 1 s. The simulation summed up in total 44 samples. Compared to the fast periodic controller that is sampled by 1 s, this strategy saves more than 94% of the samples. As before, the fast periodic controller has a better disturbance rejection for load disturbances.

7. CONCLUSIONS

In this paper we presented a self-triggered implementation of the dead-band sampling that applies to KLT and integrating processes controlled by PI controller, extending existing works that deals with static controllers and small time-delays. The presented self-triggered scheme further allows us to handle set-point changes and varying external disturbances. Although in the paper we considered only PI controllers, the proposed self-triggered scheme can be adapted to any type of controller, since it requires only the value of the control signal and hence does not depend on the particular controller structure.

Future work includes the joint utilization of self and event-triggered scheme to further improve the robustness of the self-triggered scheme.

REFERENCES

A. Anta and P. Tabuada. To sample or not to sample: Self-triggered control for nonlinear systems. *IEEE Transactions on Automatic Control*, 55(9):2030 –2042, 2010.

K. E. Årzen. A simple event-based PID controller. In *IFAC World Congress*, pages 423–428, Beijing, China, 1999.

K. J. Åström and B. Bernhardsson. Comparison of Riemann and Lebesgue sampling for first order stochastic systems. In *Proc. of the* 41^{st} *CDC*, volume 2, pages 2011–2016, Las Vegas, Nevada, USA, December 2002.

W. L. Bialowskii. A view from both sides of the gap. *Control Systems 92: Dream vs. Reality*, pages 283–298, 1992.

M. V. Dimarogonas and K. H. Johansson. Event-triggered control for multi-agent systems. In *Proceedings of the* 48^{st} *Conference on Decision and Control*, Shanghai, China, 2009.

J. P. Hespanha, P. Naghshtabrizi, and Y. Xu. A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1), January 2007.

M. Mazo, A. Anta, and P. Tabuada. An ISS self-triggered implementation of linear controllers. *Automatica*, 46(8): 1310 – 1314, 2010.

T. Norgren and J. Styrud. Non-periodic sampling schemes for control applications. Master's thesis, ABB Corporate Research, Våsteräs, Sweden, 2011.

P. G. Otanez, J. R. Moyne, and D. M. Tilbury. Using deadbands to reduce communication in networked control systems. In *ACC*, volume 4, pages 3015–3020, 2002.

P. Park. *Modeling, Analysis, and Design of Wireless Sensor Network Protocols,*. PhD thesis, KTH, 2011.

N. J. Ploplys, P. A. Kawka, and A. G. Alleyne. Closed-loop control over wireless networks. *IEEE Control Systems Magazine*, 24(3):58–71, 2004.

J. Song, A. K. Mok, D. Chen, M. Nixon, T. Blevins, and W. Wojsznis. Improving pid control with unreliable communications. In *ISA EXPO Technical Conference*, 2006.

P. Tabuada. Event-triggered real-time scheduling of stabilizing control tasks. *IEEE Transactions on Automatic Control*, 52, 2007.

Texas-Instruments. *A True System-on-Chip solution for 2.4 GHz IEEE 802.15.4 / ZigBee(TM) (Rev. F).* Texas-Instruments, 2007.

U. Tiberi, C. Fischione, K. H. Johansson, and M. D. Di Benedetto. Adaptive self triggered control over 802.15.4 networks. In 49^{th} *CDC*, Atlanta, Georgia - USA, Dec. 2010.

M. Velasco, P. Marti, and J. Fuertes. The self triggered task model for real-time control systems. In *Work-in-Progress Session of the* 24^{th} *IEEE RTSS03*, 2003.

X. Wang and M.D. Lemmon. Self-triggered feedback control systems with finite-gain $\mathcal{L}_2$ stability. *IEEE Transactions on Automatic Control*, 54(3):452–467, March 2009.

A. Willig. Recent and emerging topics in wireless industrial communication. *IEEE Transactions on Industrial Informatics*, 4(2):102–124, 2008.