MMAR 2009 14th International Conference on **Methods and Models in Automation and Robotics**
19 - 21 August 2009, Miedzyzdroje, Poland

Codesign of Distributed And Real-Time Control Systems[*]

Wojciech Grega

Department of Automatics, AGH University of Science and Technology
Al. Mickiewicza 30, 30-059 Kraków (e-mail: wgr@ia.agh.edu.pl)

Abstract: The performance of a digital control system, besides the sampling period, depends on many variables, such as the control loop execution time, jitter and communication parameters. Traditionally, digital control algorithms are designed without the consideration of their real-time implementation details and the constraints of the communication links. For distributed control systems variable queuing delays, transmission delays and lost of data, leads to the deterioration of the quality of control. The timing requirements derived from the assumptions of discrete-time control theory are not realistic for computer implementations. In the paper some control techniques improving the temporal robustness are proposed. The general conclusion is that, in the computer implementation of distributed control systems, real-time algorithms, data transmission models and digital control methods can not be developed separately.

1. INTRODUCTION

Digital control theory normally assumes evenly spaced sampling intervals and constant control delay between sampling and actuation. However, this can seldom be practically achieved in a real resource-constrained system. The relations between control task timing and control systems properties were described by several authors (Pilat *et al*., 2007, Arzen *et al*., 2000, Sågfors, 1998). It was stressed, that care must be taken when real-time execution of control algorithms generates sampling – actuation jitters or other kinds of run time violation of the closed-loop timing assumptions.

The introduction of data transmission networks into the feedback loop in many cases violates conventional control theories assumptions such as non-delayed or evenly spaced sampling sensing and actuation. The computer network is characterized by its maximal throughput. This parameter limits the amount of data that can be sent within a time unit. Network-induced delays may vary depending on the network load and medium access protocol. Generally, networked control often introduces some additional temporal non-determinism, For distributed control systems variable queuing delays, transmission delays, transport layer ACK delays and the lost of data, leads to the deterioration of the quality of control (Zhang *et al*., 2001).

Control theory specialists and control engineers do not care very much about real-time or distributed control implementations of control algorithms. In many cases they do not understand control timing constraints. The typical solutions proposed are: “buy a faster computer” or “install a more efficient data transmission network”. Control theory does not advise them on how to design controllers to take that limitation into account. In some cases they try to separate the real-time aspects and the dynamics of the control system. They develop controllers that guarantee all task deadlines under the worst case of the controller load.

When taking a closer look at the timing behaviour behind each control real time control implementation, we conclude that timing requirements derived from the assumptions of discrete-time control theory are not realistic for computer implementations. The reason is, that choices made in the real-time computer system design affect the control design and vice versa. Especially, one observes the following effects:

- variable computation-induced delays,
- variable network induced delays,
- violation of the assumption that sampling/actuation intervals are evenly spaced,
- violation of the causality principle.

The use of safety critical, distributed control systems has caused an increasing need for the simultaneous consideration of the control system and its real time implementation. It has been stated in a previous work (Grega, 2008), that integrated approaches combining two disciplines: real-time computation systems and control systems, results in better quality for digital control systems. This problem is analysed in this paper. We address the questions about how to improve the temporal robustness of distributed control systems and to how avoid violations of the causality principle.

2. DISTRIBUTED ARCHITECTURE

The basic block diagram of the distributed control system is shown in Fig.1. The process outputs are measured and control signals are applied through the distance I/O devices. The I/O devices are integrated with A/D and D/A converters.

[*] This work was supported by the European Regional Development Fund Grant no. UDA-POIG.01.03.01-12-171/08/00

The communication to and from the controller node is made over a network. From a digital control point of view, it is natural to sample the process with an equal period T_0 and to keep the control delay as short as possible. This suggests that the sensor and actuator (A/D and D/A) converters are time-triggered (sampling period T_0), while the controller is event-triggered, which means that they are triggered by the arrival of the new data.

The operation of the controller can be split into three main tasks: the sampling of sensors, control algorithm computation and actuation. The tasks are associated with the events (i.e. timer events), termination of a data frame transmission, signals that data are ready to be read from the input devices, fault detection, etc. The tasks usually share the same processor and exchange data with each other. Although a great variety of scheduling policies is available, in this work periodic task scheduling method is assumed.

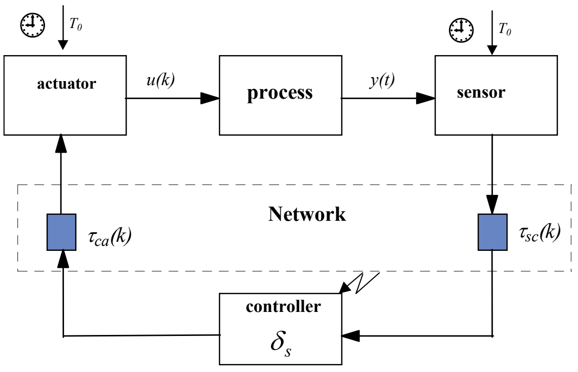


Fig.1. Basic model of distributed control system

3. SELECTION OF THE CONTROL TASK PERIOD

We will assume, that the control algorithm design is based on correctly identified models of the process and the disturbances (referred to as "nominal models"). We assume that it is possible for the nominal models to estimate a maximal, admissible sampling period, which would guarantee acceptable control performance. For the design purpose, the following timing assumptions have been made about the three main tasks of a digital control loop.

The control algorithm is designed assuming equidistant time instants at sensor nodes given by T_0. For the controller/ actuator nodes one can set MT_0, For $M<1$ we observe oversampling and some data selection procedure must be applied at the controller node. For $M>1$ we have data deficiency, and some prediction algorithm at the controller or actuator nodes must be applied.

The control task period for discrete-time control designs, beyond confirming to the Shannon theorem, can be selected following one of various so called "engineering rules" (Aström *et al.*, 1997), depending on the desired performance of the closed loop control system and the estimation of the dynamics of the process to be controlled. One rule is to choose this period according to the dominant pole positions of the continuous time process model. One accepted rule is that the control task frequency should be a $(a>1, a\in N)$ times smaller than the period of the cut-off frequency, approximated in some reasonable way for the process model.

$T_0^u = \dfrac{T^{max}}{a}$, where $T^{\max} = \max\{T_1, T_2..T_n\}$ for $A(s) = (T_1 s+1)(T_2 s+1)...(T_n s+1)$ - denominator of the process linear model. The extended rule applies for the complex dominant poles $T_0^u = \dfrac{2\pi}{\omega a\sqrt{1-\xi^2}}$. There is some flexibility in a selection:

- $a=10$, giving T_0^u as the control task period for an "ideal" (not disturbed) control system, where modelling and identification errors, as well as time delays and variations of sampling periods are negligible,
- $a>10$, giving T_0^l - the period guaranteeing the robust operation of the control system, if the system is under the influence of external and internal disturbances.

If we assume that the performance of the closed-loop control system is a strictly monotonic function of T_0 then any sampling period $T_0 < T_0^u$ improves the control performance. For $T_0 < T_0^l$ improvement is not observed.

Finally, the control task period can be estimated as $T_0 \in [T_0^l, T_0^u]$. The applied control platform (processor, peripherals hardware and operating systems) are characterized by a minimal (a shortest accessible) closed - loop execution time, estimated as $\delta_s = [\delta_s^l, \delta_s^u]$, where δ_s^l - is the lower bound of the execution time for simple control algorithms, δ_s^u - is the execution time for complex control algorithms.

The control algorithm is classified as "simple", if the pseudocode of the controller task includes no more than 5-10 operations (loops are excluded). The examples of "simple" algorithms are: incremental PID or state feedback controller. If the pseudocode of the controller includes more than 10 operations or loops are included then the algorithm is classified as "complex". The examples of "complex" control algorithms are: time-optimal or model-reference controllers.

4. SAMPLING AND ACTUATION TASKS

The presence of networks introduces communication delays and limits the amount of data that can be transferred between nodes. The communication constraints become tighter if there are several control systems sharing a common bus. In some cases not all samples from sensor or to actuator (produced with period T_0) can be sent, because the network requires intervals longer than T_0 between the transfers of two consecutive packets.

Therefore, a constraint T_o^{data} on the process data availability, introduced by the communication channel is defined (Fig.2).

The example in Fig.2a presents the situation $\delta_0^u < T_0^l$ i.e. when the interval of controller execution times does not cover the interval of control task periods. We can move T_0 below T_0^l - in some cases it is possible to set $T_0 = \delta_0^l$. This opens a possibility to use a simplified controller design technique, known as "emulation design". Using this method, a model for the controller is first developed in the s-domain and then transformed into the z- domain, using some numerical approximation methods (Tustin approximation is the most popular). In this case the controller execution frequency must be 30 times the bandwidth of the process model in order to receive good results.

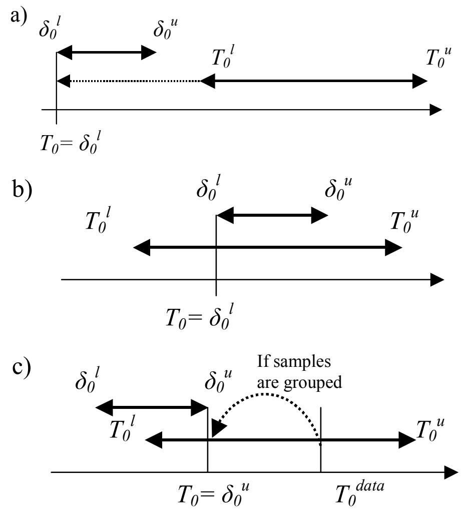


Fig 2. Controller task period and controller execution period

Example 2b presents the situation $\delta_0^l > T_0^l$ i.e. when the interval of controller execution times is inside the interval of the control task periods. We can not move T_0 below T_0^l, it is only possible to set $T_0 = \delta_0 = \delta_0^l$.

If there are several control systems sharing a common data transmission bus in some cases not all samples from the sensor or to the actuator (produced with period T_0) can be sent, because the network requires intervals longer than T_0 between the transfers of two consecutive packets. In this case, we have: $T_0 = T_0^{data}$, where T_0^{data} is a constraint on the process data availability introduced by communication channel. However, it is possible to increases network utilization by some modification of the transmission pattern – for example by sample grouping.

Let us suppose that one sample from sensor is two-byte long and that the remaining data in single a datagram occupy 48 bytes (including the network overhead). If we send four samples in four separate packets then $4 \cdot (2+48)=200$ bytes are used. With sample grouping algorithm we utilize only $48+4 \cdot 2=65$ bytes – over three times less.

In this case, sampling and actuation "observed" by the process and controller will be again: $T_0 = \delta_0 = \delta_0^u$. An appropriate adaptation of the control algorithm is needed, and the execution time of the algorithm will increase (Fig. 2c).

5. ADAPTING THE CONTROL ALGORITHM

If a network is not capable of transferring all data packets carrying signals' samples (due to timing constraints) the following approach can be applied (Grega *et al.*, 2007).

All samples from sensor **S** are transferred through network **N$_S$**, but before that they are lumped together (grouped) into a *M*-element package. This case is illustrated in Figure 3.

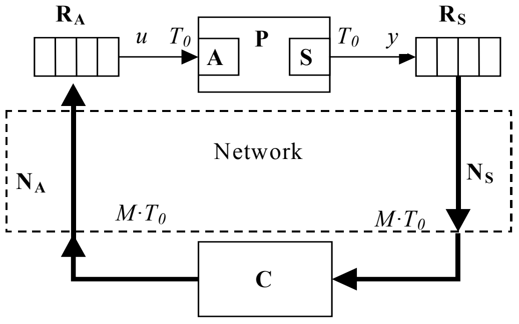


Fig. 3. Grouping of samples: control loop configuration

A new package is sent, after gathering up *M* elements (*M>1*). The collecting task is done by shift register **R$_S$**. After receiving such a package compensator **C** calculates a new package of *M* consecutive control values and sends it to register **R$_A$**, which passes them on one by one in the proper order and with the correct timing to actuator **A** (non delayed data transfer is assumed for this case). In this case effective sampling periods of particular nodes are diverse. Sensor and actuator are triggered with frequency *M* times greater than controller. On the other hand, the controller obtains complete (although not punctual) information from the sensor and the actuator receives different (in general) control values for each sampling period T_0.

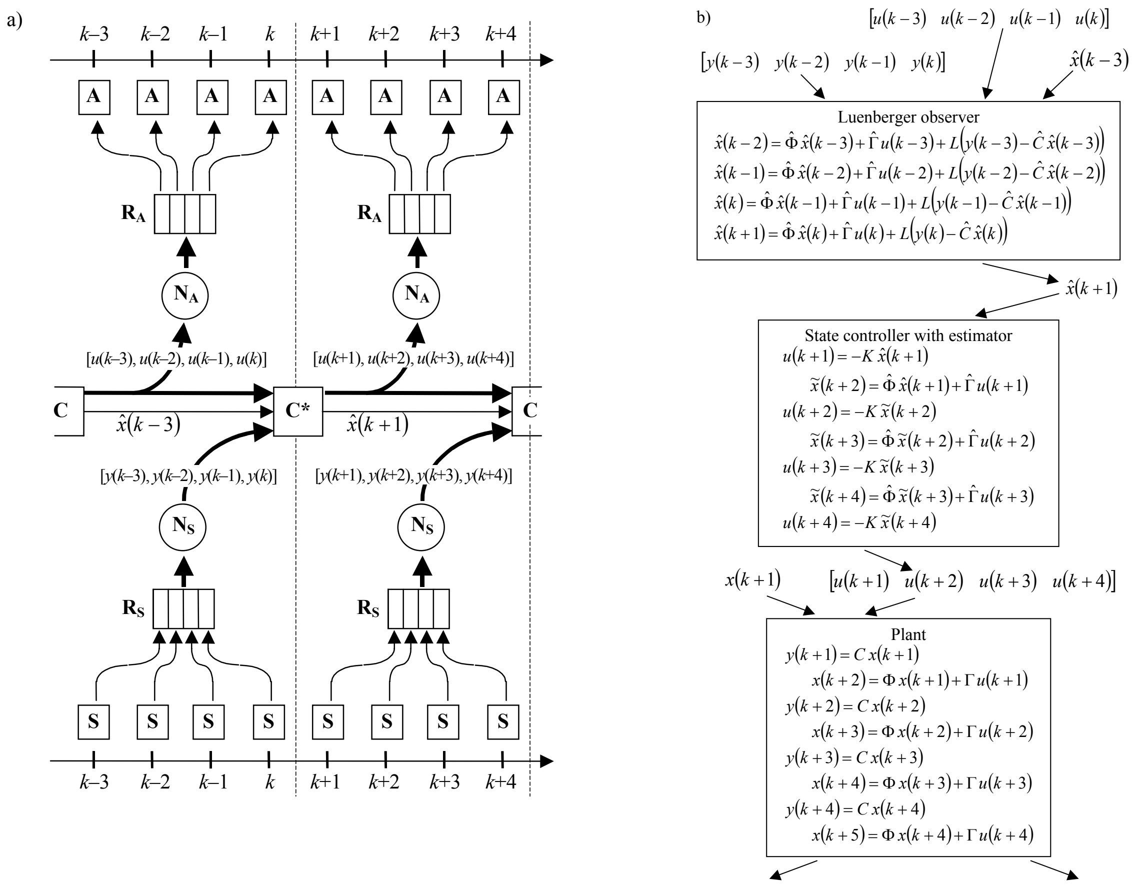
Sample grouping effects can be compensated by an approximate model of the process ("observer") at the controller side, for some range of the sampling period and modeling errors.

Let **P** be a linear time invariant and discrete process described by linear state space equation

$$\begin{cases} x(k+1) = \Phi x(k) + \Gamma(u(k) + v(k)) \\ y(k) = C x(k) + w(k) \end{cases} \qquad (1)$$

where *v* and *w* are stochastic (random) disturbances. Let the system **P** be observable and controllable. Compensator **C** for such a system can be implemented as a conjunction of Luenberger state observer and proportional state controller.

The time diagram of the control algorithm for this method and for the package length *M*=4 is shown in Figure 4a and the corresponding flow diagram is given in Figure 4b. Squares and circles in Figure 4a indicate time moments of actions of particular nodes of the system and lines show signals transmitted between nodes. In the equations shown on the flow diagram, the matrixes Φ, Γ and *C* were replaced by $\hat{\Phi}$, $\hat{\Gamma}$ and $\hat{C}$. This reflects the fact that, in practice, the exact model of the plant **P** is unknown and instead, uncertain estimates must be used. This influence of this uncertainty was also the subject of analysis (Montestruque *et al.*, 2003).



a)
k–3 k–2 k–1 k k+1 k+2 k+3 k+4
A
R_A
N_A
[u(k–3), u(k–2), u(k–1), u(k)]
[u(k+1), u(k+2), u(k+3), u(k+4)]
C
C*
$\hat{x}(k-3)$
$\hat{x}(k+1)$
[y(k–3), y(k–2), y(k–1), y(k)]
[y(k+1), y(k+2), y(k+3), y(k+4)]
N_S
R_S
S
b)
$[u(k-3)\ \ u(k-2)\ \ u(k-1)\ \ u(k)]$
$[y(k-3)\ \ y(k-2)\ \ y(k-1)\ \ y(k)]$
$\hat{x}(k-3)$
Luenberger observer
$\hat{x}(k-2)=\hat{\Phi}\hat{x}(k-3)+\hat{\Gamma}u(k-3)+L(y(k-3)-\hat{C}\hat{x}(k-3))$
$\hat{x}(k-1)=\hat{\Phi}\hat{x}(k-2)+\hat{\Gamma}u(k-2)+L(y(k-2)-\hat{C}\hat{x}(k-2))$
$\hat{x}(k)=\hat{\Phi}\hat{x}(k-1)+\hat{\Gamma}u(k-1)+L(y(k-1)-\hat{C}\hat{x}(k-1))$
$\hat{x}(k+1)=\hat{\Phi}\hat{x}(k)+\hat{\Gamma}u(k)+L(y(k)-\hat{C}\hat{x}(k))$
$\hat{x}(k+1)$
State controller with estimator
$u(k+1)=-K\hat{x}(k+1)$
$\tilde{x}(k+2)=\hat{\Phi}\hat{x}(k+1)+\hat{\Gamma}u(k+1)$
$u(k+2)=-K\tilde{x}(k+2)$
$\tilde{x}(k+3)=\hat{\Phi}\tilde{x}(k+2)+\hat{\Gamma}u(k+2)$
$u(k+3)=-K\tilde{x}(k+3)$
$\tilde{x}(k+4)=\hat{\Phi}\tilde{x}(k+3)+\hat{\Gamma}u(k+3)$
$u(k+4)=-K\tilde{x}(k+4)$
$x(k+1)$
$[u(k+1)\ \ u(k+2)\ \ u(k+3)\ \ u(k+4)]$
Plant
$y(k+1)=Cx(k+1)$
$x(k+2)=\Phi x(k+1)+\Gamma u(k+1)$
$y(k+2)=Cx(k+2)$
$x(k+3)=\Phi x(k+2)+\Gamma u(k+2)$
$y(k+3)=Cx(k+3)$
$x(k+4)=\Phi x(k+3)+\Gamma u(k+3)$
$y(k+4)=Cx(k+4)$
$x(k+5)=\Phi x(k+4)+\Gamma u(k+4)$

Fig. 4. Control algorithm with the grouping of samples (example for *M=4*): a) time diagram, b) flow diagram

6. ACHIEVING CAUSALITY

The assumption made at the design stage (constant sampling period and/or constant time delay) can be easily violated in the distributed system. The presence of networks introduces communication variable delays. This can degrade the closed-loop performance or even destabilize the system. From the control theory point of view, to analyse the stability of the model (1) one must consider an infinite product of closed-loop matrices that follow a periodic diagram, or (most complicated case) an infinite product of closed-loop matrices taken randomly from a finite set [3].

The communication delays between the sensor and the controller are denoted as $\tau^{sc}(k)$, while $\tau^{ca}(k)$ is the communication delay between the controller and the actuator, $\delta_s(k)$ is the computational delay in the controller, *k* – is the number of the control step. The total delay observed by the controller is: $\tau_s(k)=\tau_{sc}(k)+\delta_s$, while the total delay (except the process model delay) in the control loop is:

$$\tau(k)=\tau_{sc}(k)+\delta_s+\tau_{ca}(k).$$

Fig. 5 illustrates the typical timing models one can use for regularly sampled process but variable delays. For Case a) sampling and actuation are performed practically at the same sampling time (time delays can be neglected). For this model the *causality principle* is fulfilled. The following definition applies in this case.

The control algorithm is causal with the *causality margin* α, if it can be described as:

$$u(kT_0) == f(x((k-\alpha)T_0), x((k-\alpha-1)T_0),...$$
$$..,u((k-\beta)T_0), x((k-\beta-1)T_0),..),$$

for: $m \geq 0,\quad \alpha \geq 0,\quad \beta > 0,\quad m,\alpha,\beta \in N$.

If $\alpha > 0$ the algorithm will be classified as *strictly casual.* For algorithms described by linear discrete-time models:

$G(z^{-1})=\dfrac{U(z^{-1})}{X(z^{-1})}=\dfrac{A(z^{-1})}{B(z^{-1})}$ we have a simple rule: $G(z^{-1})$ is causal if and only if it is proper.

Model 5.b assumes that the control task takes some period of time, network-induced time delays are variable, but is less than the sampling period. The causality principle can be fulfilled, if the actuation is performed at the next sampling instant, i.e. $\alpha = 1$ is assumed. For model c) time delays are longer than the sampling period. One must assume $\alpha > 1$ to fulfil the causality principle.

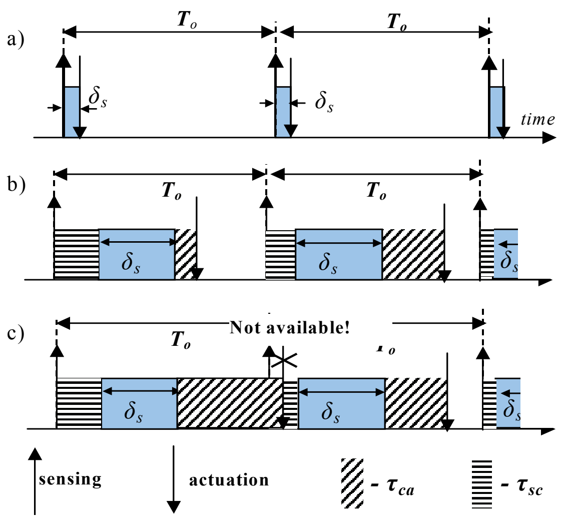


Fig.5. Timing models that can be used for a regularly sampled process

7. ADAPTING THE CONTROL ALGORITHM

The proposed solution assumes that the exact sensor data are delivered by the network during a single sampling period, but are randomly delayed. The delays are compensated by a one-step buffer at the controller side (Fig.6). In this case, the delays observed by the controller are fixed. Actuation is performed at the next sampling instant, i.e. the causality margin $\alpha = 1$ is assumed. This approach minimizes the jitter of delays, but in some cases gives a longer delay than necessary.

A number of control methods for fixed time delay compensation have been developed (Marshall *et al.*,1992). Between them are: Smith predictor and augment of the process model.

For example process, the feedback: $u(kT_0) = Kx((k-1)T_0)$, is applied and the closed-loop model of process is in the form:

$$\begin{bmatrix} x((k+1)T_0) \\ z((k+1)T_0) \end{bmatrix} = \begin{bmatrix} \Phi & \Gamma K \\ I & 0 \end{bmatrix} \begin{bmatrix} x(kT_0) \\ z(kT_0) \end{bmatrix} \qquad (2)$$

where: $z(kT_0) = x((k-1)T_0)$.

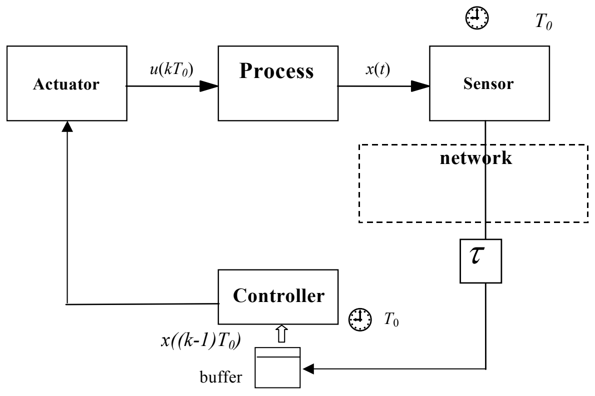


Fig. 6. Buffering the network data to fulfill the causality principle

Due to the addition of a one step delay in the control law, the closed loop model has additional eigenvalues affecting the dynamics of the closed-loop system.

8. EXAMPLE: BUFFERING DISTRIBUTED CONTROL OF TANK SYSTEM

The problem of the distributed control of a tank system is considered (Grega, 2006). The tank system consists of two water tanks placed above each other (Fig.7). Water is pumped into the most upper tank from a supply tank by a pump driven by a DC motor. The water flows out from the tanks only due to gravity. The orifices C_1 and C_2 act as flow resistors. The general objective of control is to reach and stabilize the level in the lower tank by adjustment of the pump operation.

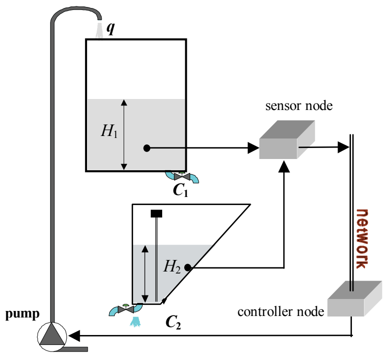


Fig. 7. Two-tanks distributed control system

If levels in the tanks are introduced as the states variables, the nonlinear model of the tank system could be linearized at equilibrium point. For an assumed sampling period T_0 the equivalent discrete-time model is:

$$\Phi = \begin{bmatrix} e^{a_1 T_0} & 0 \\ \dfrac{a_3}{a_1 - a_4}(e^{a_1 T_0} - e^{a_4 T_0}) & e^{a_4 T_0} \end{bmatrix},$$

$$\Gamma = \begin{bmatrix} \dfrac{b_1}{a_1}(e^{a_1 T_0} - 1) \\ \dfrac{b_1 a_3}{a_1 a_4}\left[\dfrac{a_1(e^{a_1 T_0} - e^{a_4 T_0})}{a_1 - a_4} - (e^{a_1 T_0} - 1)\right] \end{bmatrix}$$

The buffering algorithm was applied for this model. For simulation experiments the *True-Time* toolbox was applied (Ohlin, 2009). *TrueTime* is a *Matlab/Simulink*-based simulator for real-time co-simulation of controller task execution in real-time kernels, network transmissions and continuous process dynamics.

The sampling period was selected as $T_0 = 20s$. Linear controller parameters (based on the prescribed location of eigenvalues) were calculated. In Figure 8 the response of the system (levels in the tanks and control signal) with zero delays are given. Next, the sequence of random distributed time delays was generated (Fig. 9) Using *TrueTime Network* Block the Ethernet data transmission link was simulated with the uniform distribution of delays. Example results of the process output are given in Fig. 10. It may be concluded that the introduction of one-step buffer compensates variable delays, introduced the causality margin and only slightly decreases the control quality.

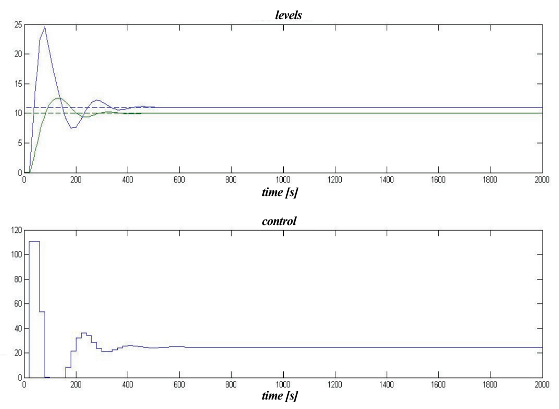


Fig.8. Control of the tank system: no network delay (T_0= *20s*)

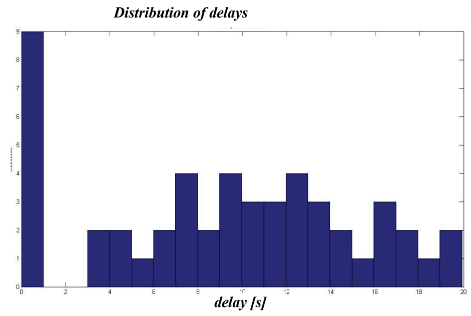


Fig. 9. Distribution of delays

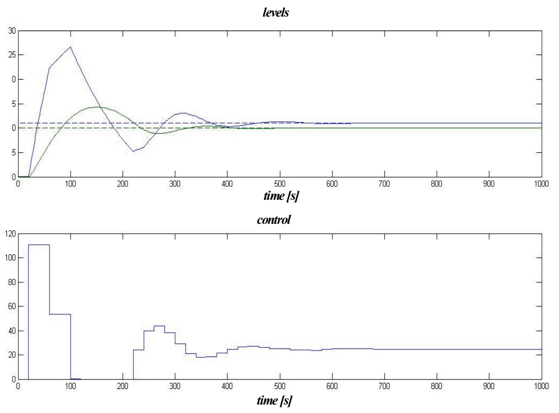


Fig. 10. Control of the tank system: variable network-induced delays and buffering of samples

9. CONCLUSIONS

The introduction of networks, limited throughput of data transmission channels, combined with non-optimized hardware and software components introduce non-determinism in the real-time control system. Classical digital control theory assumes equidistant sampling intervals and a negligible or constant control delay in the feedback loop. If we have to first design the controller, assuming a constant sampling period, and then implement the control law neglecting the sampling period jitter, we may observe that jitter effects will damage the quality of the controlled system response. This paper addresses the following questions: Does the temporal non-determinism violate the causality principle? Can we use control techniques in order to improve the temporal robustness?

Liu and Goldsmith (Liu *et al.,* 2003) claim that three communication parameters need particular attention from the distributed control perspective: *Data rate* – a high value means high temporal granularity, *Latency* – a low value implies a faster response, *Packet loss* – a small probability is associated with a reliable communication link. In this paper we claim that two other, more control oriented parameters also require attention: *Jitter* and *Causality margin*.

Two algorithms improving the temporal robustness of the distributed control system were proposed in this paper. The general conclusion is that in the computer implementation of distributed control systems, real-time algorithms, data transmission models and digital control theory methods can not be developed separately because an unexpected control system performance may occur.

REFERENCES

Arzen K-E., A. Cervin, J. Eker and L. Sha (2000). An Introduction to Control and Scheduling Co-design, In: *Proc. of 39th IEEE CDC*, Australia, Sydney.

Aström K.J. and B. Wittenmark (1997). *Computer Controlled Systems.* Prentice Hall, London.

Górecki, A. Korytowski, and K. Walton (1992). *Time-Delay Systems: Stability and Performance Criteria With Applications.* Ellis Horwood, New York.

Grega W. (2006). Compensation of Communication Constraints for Distributed Control System. In: *Proceedings of the 11th IEEE international conference on Methods and Models in Automation and Robotics,* (S. Domek, R. Kaszyński Ed), 30–31. Szczecin.

Grega W. and A. Tutaj A (2007). Network traffic reduction by sample grouping for distributed control systems. In: *Proceedings of 3rd Int. Workshop on Networked Control Systems, (*Réseau de Formation des Chercheurs en Automatique et Productique Ed), 1-8. Nancy University, Nancy, France.

Grega W. (2008). Design of Distributed Control Systems: From Theoretical to Applied Timing. In: *Recent advances in control and automation,* (K. Malinowski Ed), 343–352. Publishing House EXIT, Warszawa.

Liu X. and A. Goldsmith (2003). Wireless communication tradeoffs in distributed control In: *IEEE Decision and Control Conference*, 688-694.

Montestruque L.A. and P.J. Antsaklis (2003). On the model-based control of networked systems", *Automatica*, **39**, 1837-1843.

Ohlin M., D. Henriksson and A. Cervin (2007). *TrueTime 1.5—Reference Manual.* Department of Automatic Control, Lund University, Sweden.

Piłat A. and W. Grega (2007). Hardware and software architectures for reconfigurable time-critical control tasks, *Computer Science*, **8**, 33- 39.

Sågfors M. (1998). *Optimal Sampled-Data and Multirate Control,* Ph.D. Dissertation, Process Control Laboratory, Faculty of Chemical Engineering, Åbo Akademi University.

Zhang W., M. Branicky and S. Philips (2001). Stability of Networked Control Systems, *IEEE Control System Magazine*, **21**, 84-99.