STABILITY OF ON-LINE COMPENSATED REAL-TIME SCHEDULED CONTROL TASKS

Pau Martí*, Ricard Villà*, Josep M. Fuertes*, and Gerhard Fohler**

**Dept. of Automatic Control*
Technical University of Catalonia
Pau Gargallo 5, 08028 Barcelona SPAIN
Ph.: 34-3-4011679, Fax: 34-3-4017045, Email: {pmarti, villa, pepf}@esaii.upc.es

*** Dept. of Computer Engineering*
Mälardalen University
PO Box 883, SE-721 23 Västerås, SWEDEN
Email: gerhard.fohler@mdh.se

Abstract: Real-time scheduling methods introduce various types of jitter in task instance execution. For real-time computer-controlled systems, the introduced sampling jitter and sampling-actuation delays may degrade the system performance and even lead to instability in the system. The degradation of the system performance can be compensated on-line by updating the controller parameters at each controller task instance execution, in what we call the compensation approach. In this paper we present stability analysis for controller tasks that perform the compensation approach. *Copyright © 2001 IFAC*

Keywords: Real-time computer systems, Real-time tasks, Scheduling algorithms, Discrete-time systems, Stability analysis

1. INTRODUCTION

Traditionally, real-time computer-controlled systems have been implemented using cyclic executives. A cyclic executive runs a sequence of non-preemptive tasks, invoking each task in a fixed order throughout the execution history. For an evaluation of cyclic executive models for controller tasks see Baker and Shaw [1988]. Although the advantages of using cyclic executives for computer-controlled systems have been proven, some real limitations have also been identified. Locke [1992] discusses such difficulties arising when trying to fit complex systems into the cyclic executive model.

A solution to overcome this problem is the use of more general real-time scheduling methods for computer control. Since the early work on real-time scheduling presented by Liu and Layland (1973), real-time tasks can be scheduled using a wide variety of general purpose scheduling algorithms. One common feature of almost all these algorithms is that they introduce different forms of jitter in task instance execution. Martí *et al.* (2001a) have shown that these jitters may result in a degradation of the system control performance, and even lead to instability in the system. However, in Cervin (2000) and Årzen *et al.* (2000) it is suggested and in Martí *et al.* (2001b) further developed that this degradation can be compensated by adjusting the controller parameters according to these jitters at each controller task instance execution, in the so-called compensation approach.

In this paper we present stability analysis of the on-line compensation approach. In order to be general, all types of jitter introduced by real-time scheduling methods are treated and the stability analysis is finally performed.

This paper is structured as follows: in section 2, we revise the compensation approach under study. In section 3, the problem formalization is given. In section 4, we characterize which kind of jitter patterns real-time scheduling algorithms produce. In section 5, the stability analysis is presented, and finally, section 6 outlines some conclusions and future work.

2. COMPENSATION APPROACH

2.1. Overview

The compensation approach compensates for the degradation that sampling jitter and sampling-actuation delays may introduce in the control system performance. The basic idea is that at each controller task instance execution, the controller task parameters are adjusted according to both the actual sampling interval and the actual sampling-actuation delay. For an extensive revision of the compensation approach as well as for a jitter characterization, see Martí *et al.* (2001b).

2.2. Problem description

The problem can be stated as follows. Let's suppose that all instances of a controller task that is running in isolation on a CPU are executing an appropriate control law (designed for a time-invariant system) according to a constant sampling period h (without taking into account any time delay). If the same controller task is scheduled with other tasks by a real-time scheduling algorithm on a single CPU, the actual sampling interval (h_k) and the actual sampling-actuation delay (τ_k) may vary at each controller task k-instance execution. Consequently, in the later case, the previous control law may not be useful anymore.

However, under the compensation approach and with the adequate control law, the controller parameters will be adjusted at run-time (on-line), at each controller task instance execution. Although the control system is no longer time-invariant, the time variability that appears under the compensation approach can be analysed by studying all types of jitter that are introduced by real-time scheduling methods. Simulations have shown that the compensation approach seems to work very well. However, a proper analysis is needed in order to be able to assess when, from a stability viewpoint, this approach can be applied.

2.3. Example

An inverted pendulum mounted on a motor driven car will be used. A sketch of this system can be found in Martí *et al.* (2001b). The goal of the control is to maintain the desired vertically oriented position at all times. A linear model of the plant is:

$$\begin{bmatrix}\dot\theta\\ \dot\omega\\ \dot x\\ \dot v\end{bmatrix}=\begin{bmatrix}0 & 1 & 0 & 0\\ \frac{(M+m)\cdot g}{M\cdot l} & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ -\frac{m\cdot g}{M} & 0 & 0 & 0\end{bmatrix}\cdot\begin{bmatrix}\theta\\ \omega\\ x\\ v\end{bmatrix}+\begin{bmatrix}0\\ -\frac{1}{M\cdot l}\\ 0\\ \frac{1}{M}\end{bmatrix}u(t)$$

$$y(t)=\begin{bmatrix}\theta\\ x\end{bmatrix}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\cdot\begin{bmatrix}\theta\\ \omega\\ x\\ v\end{bmatrix}$$

In all that follows, for the sake of simplicity, we will focus on the output angle (θ). It can be easily justified that the open-loop system is unstable.

With the appropriate sampling period h, and using a pole placement observer design approach that meets the specified closed-loop requirements, a state feedback control law for the time-invariant system is obtained. All the following simulations have been done using the simulator presented by Eker and Cervin (1999). The step transient response of a controller task implementing the state feedback control law executing in isolation on a single CPU can be partially seen in Figure 1.

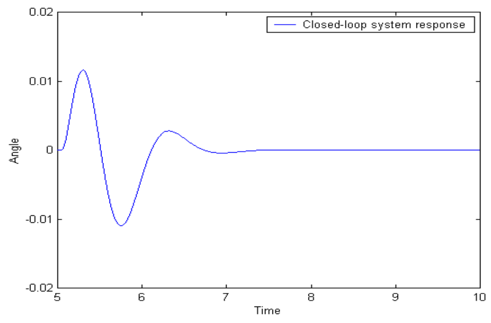

Fig. 1. Closed-loop system response

If we schedule the same controller task with other tasks by Rate Monotonic Scheduling (Liu and Layland, 1973) on a single CPU, the system becomes unstable (Figure 2) due to the jitters introduced by the scheduling method.

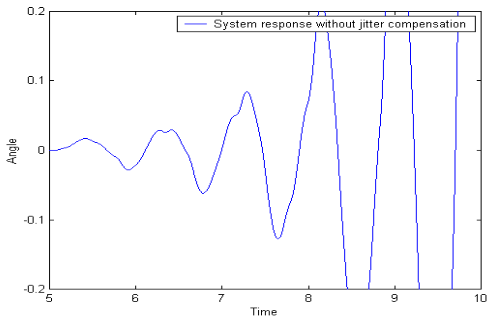

Fig. 2. Degradation of the closed-loop system response due to jitters introduced by RMS

However, if the same controller task in the same situation as before implements the compensation approach, the closed-loop system response meets again the specified requirements. It can be seen that the compensated task (Figure 3) gives almost as good response as the task that is executing in isolation on a CPU (Figure 1).

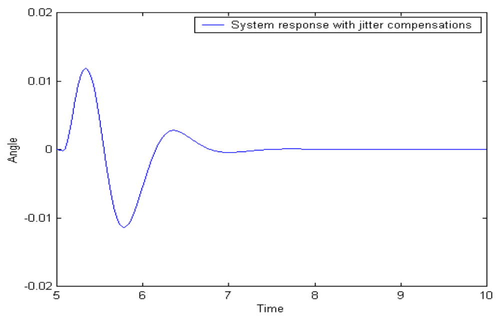

Fig 3. Closed-loop system response of a task implementing the compensation approach

3. PROBLEM FORMALIZATION

In order to formalize the control problem for the posterior stability analysis, we do a simplification that we remove later on. In a first approximation, we suppose that the scheduling algorithm only introduces sampling jitter. Once this case is properly formulated in what we call *irregular sampling discrete time system model*, we analyse the case with sampling-actuation delays (which can also be introduced by the scheduling algorithm) in what we call *discrete time system model with varying time-delays*. Finally, both models are combined in what we call *irregular sampling discrete time system model with varying time-delays*.

3.1. Irregular sampling discrete time system model

In the state space form a continuous linear time-invariant process is modelled by equations (1) and (2).

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t) \tag{1}$$

$$y(t) = Cx(t) + Du(t) \tag{2}$$

In (1) and (2) *A* is the system matrix, *x(t)* is the state vector, *B* is the input matrix, *u(t)* is the control input, *y(t)* is the system output, *C* is the output matrix and *D* is the direct matrix, all matrices of suitable dimension. Equation (2) is the output equation. For periodic sampling with constant period *h*, the discrete-time system can be described by (3) and (4), where Φ and Γ are obtained from (1) and (2) as detailed in (5) and (6).

$$x(kh+h) = \Phi(h)x(kh) + \Gamma(h)u(kh) \tag{3}$$

$$y(kh) = Cx(kh) + Du(kh) \tag{4}$$

$$\Phi(h) = e^{Ah} \tag{5}$$

$$\Gamma(h) = \int_0^h e^{As} ds B \tag{6}$$

To meet the closed-loop requirements, the system specified by (3) and (4) is controlled using state feedback, where matrix *L* (gain matrix) can be obtained by a design method such as pole placement or optimization approach. Equation (7) gives the state feedback control law.

$$u(kh) = -L(h)x(kh) \tag{7}$$

Equation (3) can be rewritten in terms of (7) as in (8).

$$x(kh+h) = (\Phi - \Gamma L)x(kh) \tag{8}$$

At the end, the closed-loop time-invariant system is characterized (at least) by equations (3), (4), and (7). To study the system stability, we have to examine the closed-loop matrix (9), where Φ, *L*, and Γ are constant matrices in terms of a constant sampling period h.

$$\Phi_{cl} = \Phi - \Gamma L \tag{9}$$

The system is stable iff the spectral radius of the closed-loop matrix Φ_{cl} is less than one.

$$\text{Stable} \Leftrightarrow \rho(\Phi_{cl}) < 1$$

The spectral radius ρ of a matrix A is defined as follows:

$$\rho(A) = \max\{|\lambda| \mid \lambda \text{ is an eigenvalue of A}\}$$

However, if the on-line compensation approach is used and at each controller task instance execution the controller parameters are adjusted according to the actual sampling interval, the discrete-time system is no longer time-invariant. Therefore, the stability of the system will not only depend on a fixed closed-loop matrix Φ_{cl}, but also on the sampling jitter.

If we denote the actual sampling interval of each task instance execution by h_k, systems described by equations (1), (2) and (7) but with irregular sampling can be modeled by equations (10), (11) and (12).

$$x(\bar{h}_{k+1}) = \Phi(h_k)x(\bar{h}_k) + \Gamma(h_k)u(\bar{h}_k) \tag{10}$$

$$y(\bar{h}_k) = Cx(\bar{h}_k) + Du(\bar{h}_k) \tag{11}$$

$$u(\bar{h}_k) = -L(h_k)x(\bar{h}_k) \tag{12}$$

where $\bar{h}_k = \sum_0^k h_k$ and matrices $\Phi(h_k)$, $\Gamma(h_k)$ and $L(h_k)$ are obtained from (5) and (6), using the same control design approach, all with $h=h_k$ at each controller task instance execution. In this approach, the stability of the system controlled by a task τ_i will depend on a product-sequence of matrices Φ_{clk}, each one depending on the task instance sampling interval h_k.

3.2. Discrete time system model with varying time-delays

In a state space form a continuous linear time-invariant process with a constant time-delay τ is modelled by (13) and (14).

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t-\tau) \tag{13}$$

$$y(t) = Cx(t) + Du(t) \tag{14}$$

In (13) τ is assumed to be less than or equal to the sampling period h (for further details, see Åström and Wittenmark, 1997 and Wittenmark *et al.*, 1995). For periodic sampling with constant period *h*, the discrete-time system can be described by (15) and (16).

$$x(kh+h) = \Phi(h)x(kh) + \Gamma_0(h,\tau)u(kh) + \Gamma_1(h,\tau)u(kh-h) \quad (15)$$

$$y(kh) = Cx(kh) + Du(kh) \quad (16)$$

Matrices Φ, Γ_0 and Γ_1 are obtained from (13) and (14) as detailed in equations (17), (18) and (19).

$$\Phi(h) = e^{Ah} \quad (17)$$

$$\Gamma_0(h,\tau) = \int_0^{h-\tau} e^{As}ds B \quad (18)$$

$$\Gamma_1(h,\tau) = e^{A(h-\tau)}\int_0^{\tau} e^{As}ds B \quad (19)$$

As in the previous section, to meet the closed-loop systems requirements, the system specified by (15) and (16) is controlled using state feedback (20) where the gain matrix *L* can be obtained using the same methods.

$$u(kh) = -L(h,\tau)x(kh) \quad (20)$$

Equation (15) can be rewritten in terms of (20) as in (21).

$$\begin{bmatrix} x(kh+h) \\ x(kh) \end{bmatrix} = \quad (21)$$
$$\begin{bmatrix} \Phi(h) - L(h,\tau)\cdot\Gamma_0(h,\tau) & -L(h,\tau)\cdot\Gamma_1(h,\tau) \\ I & 0 \end{bmatrix} \cdot \begin{bmatrix} x(kh) \\ x(kh-h) \end{bmatrix}$$

At the end, the closed-loop time-invariant system is characterized by equations (15), (16), and (20). In this case, to study the system stability, we have to examine the closed-loop matrix (22), where Φ, L, Γ_0 and Γ_1 are constant matrices in terms of a constant sampling period h and a constant time-delay τ.

$$\Phi_{cl} = \begin{bmatrix} \Phi(h) - L(h,\tau)\cdot\Gamma_0(h,\tau) & -L(h,\tau)\cdot\Gamma_1(h,\tau) \\ I & 0 \end{bmatrix} \quad (22)$$

As before, the system is stable iff the spectral radius of the closed-loop matrix Φ_{cl} is less than one.

$$\text{Stable} \Leftrightarrow \rho(\Phi_{cl}) < 1$$

As before, if the on-line compensation approach is used and at each controller task instance execution the controller parameters are now adjusted according to the actual sampling-actuation delay, the discrete-time system is no longer time-invariant. Therefore, the stability of the system will not only depend on a fixed closed-loop matrix Φ_{cl}, but also on the sampling-actuation delay.

If we denote the actual sampling-actuation delay of each task instance execution by τ_k, systems described by equations (13) and (14) with discrete constant sampling and varying time-delays can be modeled by equations (23), (24) and (25).

$$x(kh+h) = \Phi(h)x(kh) + \Gamma_0(h,\tau_k)u(kh) + \Gamma_1(h,\tau_k)u(kh-h) \quad (23)$$

$$y(kh) = Cx(kh) + Du(kh) \quad (24)$$

$$u(kh) = -L(h,\tau_k)x(kh) \quad (25)$$

Matrices Φ(h), Γ_0(h,τ_k), Γ_1(h,τ_k) and L(h,τ_k) are obtained from (17), (18) and (19), using the same control design approach as in (20), all with $\tau=\tau_k$ at each controller task instance execution. In this approach, the stability of the system controlled by a task will depend on a product-sequence of matrices Φ_{clk}, each one depending on the task instance sampling-actuation delay τ_k and the constant sampling period h.

3.3. *Irregular sampling discrete time system model with varying time-delays*

Combining appropriately the two models described in the two previous sections, that is, compensating both sampling jitter and sampling-actuation delays, systems described by equations (13) and (14) with irregular discrete sampling and varying time-delays can be modeled by equations (26), (27), (28) and (29).

$$x(\bar{h}_{k+1}) = \Phi(h_k)x(\bar{h}_k) + \Gamma_0(h_k,\tau_k)u(\bar{h}_k) + \Gamma_1(h_k,\tau_k)u(\bar{h}_{k-1}) \quad (26)$$

$$y(\bar{h}_k) = Cx(\bar{h}_k) + Du(\bar{h}_k) \quad (27)$$

$$u(\bar{h}_k) = -L(h_k,\tau_k)x(\bar{h}_k) \quad (28)$$

$$\bar{h}_k = \sum_0^k h_k \quad (29)$$

Matrices Φ(h_k), Γ_0(h_k,τ_k), Γ_0(h_k,τ_k) and L(h_k,τ_k) are obtained from (17), (18) and (19), using the same control design approach as in (20), all with $h=h_k$ and $\tau=\tau_k$ at each controller task instance execution. In this case, the stability of the system controlled by a task will depend on a product-sequence of matrices Φ_{clk} (as in (22)), but each one depending on the task instance sampling interval h_k and sampling-actuation delay τ_k.

3.4. *Stability condition*

In the three previous subsections, we concluded that the stability of the system depends on a product-sequence of specific closed-loop matrices Φ_{clk}. Therefore the system stability depends on the expression described by (30).

$$\left\{\prod_{k=0}^{\infty} \Phi_{clk}\right\} \quad (30)$$

As a consequence, an overview of what kind of product-sequences will apply, due to real-time scheduling methods, will be given. Afterwards, according to these types of product-sequences of matrices, we will present a stability analysis for the last case we formalized in subsection 3.3, which is the generalization of all the other cases.

4. JITTER PATTERNS

Real-time scheduling methods can be divided into two major categories according to the time when jitters are generated and analyzable:

- *Offline*: the whole schedule is constructed before run time (e.g., see Fohler, 1994 or Ramamritham, 1990): over some hyper-period (usually LCM of all periods of all scheduled tasks), it is known at which time each task instance will execute. Therefore, all the jitters are known or could be specified beforehand, and follow a periodic pattern over the hyper-period. Depending on how the offline schedule is constructed, two kinds of jitter periodic pattern apply: a known sampling interval constant (h) with constant sampling-actuation delays (τ) through all the execution life of a controller task or a finite sequence of known sampling intervals ($h_1, h_2, ..., h_k, ..., h_n$) with a finite sequence of known sampling-actuation delays ($\tau_1, \tau_2, ..., \tau_k, ..., \tau_n$) that repeats periodically through all the execution life of a controller task if the runtime execution preserves the off-line scheduled controller instances starting times.

- *On-line*: task instances are dispatched at run time according to the scheduling algorithm that is being used. Therefore, there is not a priori knowledge of the schedule that will apply at run time nor whether the runtime dispatcher will preserve the execution pattern of controller task instances that have been previously scheduled offline. In these cases, the only available knowledge is that during run time, an infinite sequence of variable but bounded sampling intervals will apply with an infinite sequence of variable but bounded sampling-actuation delays through all the execution life of a controller task. If the system is schedulable, the only available knowledge is that each sampling interval h_k and each sampling-actuation delay τ_k for all instances of a given controller task will be bounded:

$$\forall k, \qquad \min(C^k_i) \le h_k \le \max(st^{k+1}_i - st^k_i)$$
$$0 < \tau_k \le \max(h_k) - \min(C^k_i)$$

  where C^k_i is the computation time and st^k_i is the starting time of a k-instance of a i-task.

Therefore, we distinguish three types of jitter patterns:

Case 1: Constant jitters.
Case 2: Known repeating sequence of known jitters.
Case 3: Unknown infinite sequence of bounded jitters.

5. STABILITY ANALYSIS

Given the system specified by equation (3), (4), and (7), the system stability depends on the closed-loop matrix in (9). Taking into account that the state feedback law will be computer implemented and a real-time scheduling algorithm will schedule the corresponding controller task, the following two statements will hold:

1. The sampling period h belongs to an interval $h \in [h_{min}, h_{max}]$ with $h_{min} > C$ and $h_{max} < T_s$, where T_s is the Shannon's sampling theorem limit and C shall include the minimum computation time of a controller task instance.

$$h_{min} < h < h_{max}$$

Similarly, the sampling-actuation delay τ belongs to an interval $\tau \in [\tau_{min}, \tau_{max}]$ with $\tau_{min} > 0$ and $\tau_{max} < h_{max}$.

$$\tau_{min} < \tau < \tau_{max}$$

2. The sampling period and the sampling-actuation delay are integers multiple of the clock ticksize.

$$h = m * \text{ticksize},\ \tau = n * \text{ticksize}, \quad n,m \in N$$

Therefore, both intervals for the sampling period and for the sampling-actuation delay have a finite number of values, which we can characterize by the following sets:

$$H = \{h \mid h = m * \text{ticksize},\ m \in N \text{ and } h_{min} < h < h_{max}\}$$
$$T = \{\tau \mid \tau = n * \text{ticksize},\ n \in N \text{ and } \tau_{min} < \tau < \tau_{max}\}$$

The stability analysis for the jitters characterized in the previous section is depicted in the following. If jitters are constant, the following Case 1 applies.

Case 1: there is one and only one closed-loop matrix, Φ_{cl}. In this case, the stability test is the following, where ρ is the spectral radius:

$$\text{Stable} \quad \Leftrightarrow \quad \rho(\Phi_{cl}) < 1$$

If jitters are not constant, we can distinguish two possibilities (note that the process is no longer time-invariant and equations (26), (27), (28) and (29) can describe the new system), depicted in Case 2 and Case 3.

Case 2: A variable sampling period ($h_k \in H$) and a variable sampling-actuation delay ($\tau_k \in T$) that repeats periodically:

$$h_1, h_2, ..., h_k, ..., h_n, h_1, h_2, ..., h_k, ..., h_n, ...$$
$$\tau_1, \tau_2, ..., \tau_k, ..., \tau_n, \tau_1, \tau_2, ..., \tau_k, ..., \tau_n, ...$$

In this case, the closed-loop system will be characterized by a known finite set of matrices that will follow a known periodic pattern. Therefore, a known repeating sequence of known matrices will apply. In this case, the stability test can be performed by checking the stability of the product of the repeating sequence of matrices (Dogruel and Özgüner, 1995):

$$\text{Stable} \Leftrightarrow \rho(\Phi_{cl1} \cdot \Phi_{cl2} \cdot ... \cdot \Phi_{cln}) < 1$$

Case 3: A variable sampling period ($h_k \in H$) and a variable sampling-actuation delay ($\tau_k \in T$), but with unknown execution pattern:

$$h_1, h_2, \ldots, h_k, h_{k+1}, \ldots$$
$$\tau_1, \tau_2, \ldots, \tau_k, \tau_{k+1}, \ldots$$

In this case, the closed-loop system will be characterized by a product of an infinite number of matrices taken randomly from a finite set of matrices. In this case, the stability test applied above cannot be used. However, another test can be applied using known results of linear algebra (see for example Strang, 1980):

$$\forall\ \Phi_{clk} : \mathrm{eig}(\Phi^T_{clk} \cdot \Phi_{clk} - I) < 0 \ \Rightarrow\ \text{Stable}$$

If each matrix Φ_{clk} satisfies that all eigenvalues of $\Phi^T_{clk} \cdot \Phi_{clk} - I$ are less than zero (each $\Phi^T_{clk} \cdot \Phi_{clk} - I$ is negative definite), each matrix Φ_{clk} will guarantee immediate decay. In this case, any product of an infinite number of such matrices Φ_{clk} will guarantee stability in the system.

The presented stability analysis has covered all possible types of jitter that real-time scheduling algorithms may introduce. Therefore, the on-line compensation approach will be applicable when the closed-loop system matrix or matrices fall into one of the three cases.

6. CONCLUSIONS

A review of the stability of on-line compensated real-time scheduled control tasks has been done showing that three cases can be distinguished. Two of them were already known. The third case has been categorized and a sufficient stability test presented. Using this analysis, given a system to be controlled and a scheduling algorithm, it will be easy to decide if the system will behave correctly from a stability viewpoint when using the compensation approach.

Even though extensive simulations of the on-line compensation approach show good performance, more study besides the presented stability analysis is needed in order to achieve a deeper characterization of the controlled system under this approach. Therefore, on the one hand, further work will focus on study how the compensation approach behaves in terms of system responsiveness. On the other hand, we will focus on what are the implications of using the on-line compensation approach from a schedulability viewpoint.

ACKNOWLEDGEMENTS

The presented research has received support from Spanish CICYT project ref. DPI2000-1760-C03-01.

REFERENCES

Årzen, K.-E., A. Cervin, J. Eker and L. Sha. (2000). An Introduction to Control and Scheduling Co-Design. In *39th IEEE Conference on Decision and Control, Sydney*, Australia, December

Åström, K. J and B. Wittenmark (1997). *Computer-Controlled Systems. Third edition*. Prentice Hall.

Baker, T.P. and A. Shaw (1988). The Cyclic Executive Model and Ada. In *Proceedings of IEEE Real-Time Systems Symposium*, pages 120-129, Madrid, Spain.

Cervin, A. (2000). Towards the Integration of Control and Real-Time Scheduling Design. Licentiate Thesis. ISRN LUTFD2/TFRT—3226—SE, Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden, May

Dogruel, M and U. Özgüner (1995). Stability of a Set of Matrices: A Control Theoretic Approach. In *Proc. of the 34th Conference on Decision and Control*, New Orleans, USA, September

Eker J. and A. Cervin (1999). A Matlab Toolbox for Real-Time and Control Systems Co-Design. *Proc. 6th Int. Conf. on Real-Time Computing Systems and Applications*, Hong Kong, China, December

Fohler, G. (1994). Flexibility in Statically Scheduled Real-Time Systems. Phd Thesis. Technisch-Naturwissenschaftliche Fakultaet, Technische Universitaet Wien, Austria, April

Liu, C. and J. Layland (1973). Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. *J.ACM*, 20, 46-61.

Locke, C.D. (1992). Software Architecture for Hard Real-Time Applications: Cyclic Executives vs. Fixed Priority Executives. In *J. of Real-Time Systems*, 4, 37-53, Kluwer Academic Publishers.

Martí P., R. Villà, J.M. Fuertes and G. Fohler (2001a). On Real-Time Control Tasks Schedulability. In *IFAC European Control Conference,* September, Porto, Portugal

Martí P, G. Fohler, K. Ramamritham and J.M. Fuertes, (2001b). Jitter Compensation in Real-Time Control Systems. *Accepted for publication to 22th IEEE Real-Time Systems Symposium*, December, London, UK.

Ramamritham, K. (1990). Allocation and Scheduling of Complex Periodic Tasks. *In 10th Int. Conf. on Distributed Computing Systems*, pages 108--115.

Strang, G. (1980) *Linear Algebra and its Applications. Second Edition.* MIT. Academic Press, Inc. USA

Wittenmark B., J. Nilsson and M. Törngren (1995). Timing Problems in Real-time Control Systems. In *Proc. American Control Conference*, USA