Feedback scheduling for real-time control of systems with communication delays

O. Sename*†, D. Simon*, D. Robert*
*INRIA Rhone-Alpes, 655 avenue de l'Europe, Montbonnot, 38 334 Saint-Ismier Cedex, FRANCE
Fax :+33 4 76 61 54 77
Email :{Olivier.Sename, Daniel.Simon}@inrialpes.fr
†Laboratoire d'Automatique de Grenoble, ENSIEG-BP 46, 38402 Saint Martin d'Heres Cedex, FRANCE
Email: Olivier.Sename@inpg.fr

***Abstract*— This work is devoted to integrated control-scheduling in the case of real-time control of systems with communications delays, i.e. network induced delay and input output latency. Feedback control of computing resources is tackled, to deal with CPU resource variations and unpredicitble workload. Thus two new control schemes are provided for feedack scheduling of single and multi control task systems. Moreover the system control design takes into account the unknown delays due to the temporal uncertainties that are unavoidable in real-time control. A new method of state feedback control design for discrete time-delay systems is presented in a LMI formulation. Two illustrative examples show the interest of the proposed approach.**

I. INTRODUCTION

Feedback control systems over a real-time communication network are now widely used in industry. For instance, new generation of cars involve communication networks where data (coming from sensors, actuators, and subsystems) are exchanged for on-line diagnosis and control purposes. GPS, ESP, powertain control, chassis control, airbags ... are today parts of the standard embedded real-time systems in automotive applications. The use of communication networks makes it necessary to deal with network-induced delays. These delays may be unknown and time-varying, and may deteriorate the closed-loop performances and even destabilize the closed-loop system. On the other hand, in robotic applications, the performances of a robot can be improved using a multi-rate controller [1], where several control laws run in parallel with different sampling rates inside a hierarchy of more or less tightly coordinated layers. In such systems, scheduling of control tasks is of great importance to keep production costs low and to avoid unstability or poor control performances, in particular in resource insufficient environments.

Digital control systems are often implemented as a set of tasks running on top of an off-the-shelf real-time operating system (RTOS) using fixed-priority and preemption [2]. The performances and stability of the control scheme strongly relies on the respect of the specified sampling rates and computing delays (latencies) [3]. However, if control systems are mostly considered as examples of "hard real-time systems" where deadline violations are forbidden, experiments show that a closed-loop control with good stability and robustness (w.r.t. parameter uncertainties) properties may also provide robustness w.r.t. timing uncertainties: these may not lead to unstability, but only act as disturbances. A "weakly hard" assumption could then be used instead, where absolute deadlines are replaced by statistical ones, e.g. the allowable output jitter compliant with the desired control performance. Even if computing such statistics is out of the scope of current control theory, this intrinsic robustness of closed-loop controllers gives an additional degree of freedom which can serve QoS computation and flexible scheduling design.

The real-time community has usually considered that control tasks have fixed periods, hard deadlines and worst-case execution times. This assumption has served the separation of control and scheduling designs, but has led to underutilization of CPU resources. However current real-time design methods and associated analysis tools do not provide a model flexible enough to fit well with control systems engineering requirements. Taking into account the unsuitability of current real-time design to capture feedback control systems requirements naturally leads to use control/scheduling co-design. This closer interaction is particularly needed for control applications requiring high degrees of flexibility or when computing resources are limited. This integrated approach which takes care of both control performance and timing requirements has been studied in [4], [5], [6], [7]. However while off-line methods can handle control requirements they cannot easily handle timing uncertainties and dynamic reconfigurations.

On the other hand, our objective is to keep stability and good performances for the closed-loop system, even if the utilization resources vary, and in the presence of timing uncertainties as communication delays. Thus it can be useful to consider more *dynamic solutions*, i.e. to adapt the execution of the control task (period and value) according to the availability of the resources. This is called feedback scheduling [8], the purpose of which is to deal with on-line trade-offs between control performance and computing resources (CPU time and communication bandwidth) utilization.

This paper deals with feedback scheduling in the presence of communication delays, mainly the input output latency. As in [9], we aim to adjust on-line the sampling period

of the controller in order to meet the computing resource requirements. The contribution of the paper is two-fold: first a new method for controlling the resource utilization is given (in the cases of single control task and multi control tasks systems), then the system control design takes into account delays that are unavoidable in real-time control. We here provide, for the first time, a state feedback control law for discrete systems with unknown control delay, the parameters of which depend on the sampling period of the controller.
The outline of the paper is the following. In section 2 the problem is described and some background on real-time control and related works are presented. In section 3, two feedback sheduling control schemes are provided for single and multi control task systems respectively. Section 4 presents the control design for discrete systems with input delays. Section 5 illustrated the proposed scheme for a real-time control of an inverted pendulum, and section 6 is devoted to a multi-task control system of a robot. The paper ends with some concluding remarks.

II. Problem statement and state of the art

In most of computer-controlled systems the computer must do in each period: sampling of the process output, executing of the control algorithm, sending the new control signal to the process. This implies that the control task is supposed to have a fixed period and that the input-output latency (i.e the control delay) is small and without jitter. If not, this should be considered in the control desing.
We are interested here in real-time control of systems with communications delays (see fig 1). Such delays may be of two-fold: network-induced delays in the case of networked control systems (communication between the sensor and the controller, and between the controller and the actuator), and the computational delay corresponding to the control computation cost (see [10] for more details on networked control systems). The latter is generally less than a sampling period. However when the control task is preempted by higher priority tasks, this may lead to delays larger than a sampling period.

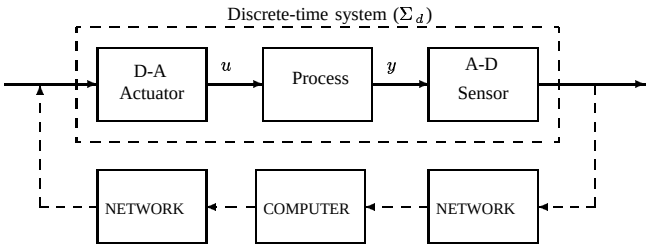

Fig. 1. Overview of a computer-controlled system with network-induced delay

As seen on fig 1, the different delays can be lumped together, for analysis purpose, in one single control delay h.
In this work devoted to computer-controlled systems, in particular to control implementation, discrete time-delay systems are considered as:

$$\Sigma_d : \quad x(k+1) \quad = \quad Ax(k) + Bu(k-d) \qquad (1)$$

where $x(k) \in \mathbb{R}^n$ is the state vector assumed to be measured, $u(k) \in \mathbb{R}^r$ is the control input vector, d is the positive *unknown* delay value, A and B are real matrices of appropriate dimensions. Let us recall that in this formulation, d contains the computational delay, and the network-induced delays. It may be larger than a sampling period and is unknown.

Remark 1: Note that control theory for linear systems sampled at fixed rates, including fixed and known delays, has been much studied for ten years. Let us cite [3] where an augmentation approach is used for sampling a system with time-delay and obtain an equivalent free delay representation. Since the delay is here assumed to be unknown, such an approach cannot be applied. On the other hand specific methods to time-delay systems could be used to derive a discrete-time representation with delays [11] from a continuous-time one.

For systems (1), the following family of state feedback control laws is considered:

$$u(k) = Kx(k) \qquad (2)$$

In the sequel, some background on real-time control and feedback scheduling is presented.

A. Background in real-time control systems

Any task is characterized by a period, dealine and worst case execution time. Well known scheduling policies, such as Rate Monotonic for fixed priorities and EDF for dynamic priorities, assign priorities according to timing parameters resp. sampling periods and deadlines. These methods tend to maximise the computing resource usage or the number of tasks which meet their deadline. However it has been shown, e.g. [12], that a blind use of such purely computing parameters based scheduling policy can lead to an inefficient controller implementation while a scheduling policy based on application's requirements gives better results.
Moreover, such design and off-line schedulability analysis relies on a right estimation of the tasks worst case execution time. However there exist uncertainties in the control algorithm cost due for instance to a badly known calculation cost, or to dynamic environment (e.g. to various control modes). Thus real-time control design based on worst case execution and strict deadlines inevitably leads to a low average usage of the computing resource.
This emphasizes the interest of using *dynamic solutions*, i.e. to modify on-line the execution of the control task (period and value) according to the availability of the resources. This needs to take into account the temporal uncertainties as unknown delays in the control application.

B. Feedback scheduling

This new approach has been initiated both from the real-time computing side [13] and from the control side [14],

[8], [9]. The idea consists in adding to the process controller an outer sampled feedback loop ("scheduling regulator") to control the scheduling parameters as a function of a QoC (Quality of Control) measure. The QoC criterion captures the control performance requirements and the problem can be stated as QoC optimisation under constraint of available computing resources. During each experiment, an estimate of the current requested utilization may be computed as:

$$U_{req} = \sum_{i=1}^{n} \frac{C_i}{h_i}$$

where C_i is the estimated execution time of the task i (filtered from job execution-time measurements), and h_i the sampling period assigned to task i.
Preliminary studies [8] suggest that a direct synthesis of the scheduling regulator as an optimal control problem leads, when it is tractable, to a solution too costly to be implemented in real-time. Practical solutions will be found in the available control toolbox or in enhancements and adaptation of current control theory. For instance the calculation of the new task periods can be done by the rescaling [14]:

$$h_i^{new} = h_{i_{nominal}} \frac{U_{req}}{U_{sp}}$$

where U_{sp} is the utilization set-point.
The feedback scheduler then controls the processor utilization by assigning task periods that optimize the overall control performance.

III. Further solutions to feedback scheduling

In order to adjust, on-line, the period of the control tasks, a control-scheme should be established for the scheduler, as done in [9]. The feedback scheduler is here implemented as an application task that runs in parallel with the control task, with a higher priority. It executes as a periodic task, with a period h_S, larger than the sampling periods of the control tasks, in order to change of sampling period only when resource availability changes have been observed. Feedback scheduling is a dynamic approach allowing to better use the computing resources, in particular when the workload changes e.g. due to the activation of an admitted new task. We propose in figure 2 a hierarchical control structure. The feedback scheduler controls the CPU activity according to the computing resource availability (measured through some computing load metric) by adjusting the periods of the tasks used in the process controller(s).

On the other hand the internal process controller is here designed to take into account timing uncertainties, e.g. due to preemptions which are unavoidable in real-time control and difficult to accurately predict in a dynamic environment. Indeed unknown input-output latencies can deteriorate the process performances and stability; thus the considered Quality of Control measure is composed of the usual tracking error and the robustness w.r.t. control delays.

Two different methodologies are described in the sequel, depending on the number of system control tasks.

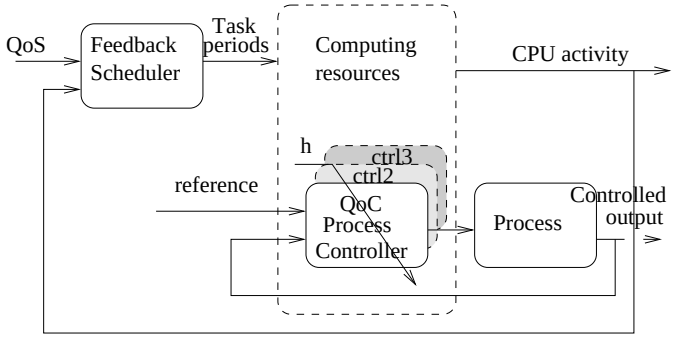

Fig. 2. Hierarchical control structure

A. Case of single task control systems

Step 1: estimation of resources utilization
In this case of single task control system, we choose to estimate the CPU utilization, for each period of the scheduler, from filtered job response-time measurements of the control task, as:

$$\begin{aligned} \hat{R}(kh_S) &= \lambda \hat{R}(kh_S - 1) + (1-\lambda) r(kh_S) \\ \hat{U}(kh_S) &= \hat{R}(kh_S)/h(kh_S) \end{aligned} \quad (3)$$

where $h(kh_S)$ is the sampling period currently assigned to the control task, λ is a forgetting factor and $r(kh_S)$ is here the mean of the measured response-times (i.e. execution-times + preemptions) of the control task executions on each period of the scheduler. Samples are taken here at the period of the feedback scheduler, which is usual for identification purpose.

Remark 2: Note that more precised models could be used as done in by [13] for miss ration and utilization control. However (3) is interesting for control purpose as it takes into account the preemptions the control task is sujected to (as its priority is lower than the scheduler's one). This also indicates the freedom degrees in the resource utilization usable by the control task. Finally note that, in this case of single control task, we have $R \simeq C$.
We have here chosen $\lambda = 0.3$ which ensures a fast estimation.

Step 2: H_∞ Control design
For control design, we consider a simple control model of the system whose input is the period of the control task, and output is the corresponding estimated resource utilization. The modelling approach is based on a temporal test, i.e. a step input on the sampling period, in order to deduce a transfer fucntion $G(z-1)$ (see section 5 for illustration).
We may apply any advanced control methodology to design a controller for such a system. We have chosen the H_∞ control theory which can lead to robust control against modelling errors (see [15] for details on H_∞ control). Such a control method uses some weighting functions that have to be chosen to satisfy the performance specifications i.e. mainly a closed-loop system with a rise time of $1s$ and a module margin higher than 0.5 for robustness. Using the classical methods available in control design softwares, the H_∞ control problem is solved according to these specifications.

B. Case of multi tasks control systems

Step 1: estimation of resources utilization

In a similar way as in [9], for each period of the scheduler h_S, the CPU utilisation is estimated from job execution-time measurements of the control tasks, as:

$$\begin{aligned} \overline{U}(kh_S) &= \sum_{i=1}^{n} \frac{\overline{c}_i(kh_S)}{h_i((k-1)h_S)} \\ \hat{U}(kh_S) &= \lambda\hat{U}((k-1)h_S) + (1-\lambda)\overline{U}(kh_S) \end{aligned} \quad (4)$$

where, for each period of the scheduler, $h_i(kh_S)$ is the sampling period currently assigned to the control task i, and $\overline{c}_i(kh_S)$ is here the mean of the measured execution-times of the control task i during each period of the scheduler. Samples for the measured output (i.e. CPU utilisation) are taken here at the period of the scheduler to be controlled, which is usual for identification purpose.

Remark 3: Note that, using the response-time of the tasks to estimate the CPU resource in a multi-task control system it not satisfactory as it will lead to an upper estimation of the resource availability.

In (4)λ is a forgetting factor, chosen as $\lambda = 0.3$, which ensures a fast estimation.

Step 2: H_∞ Control design

Here a new control scheme is provided for the feedback scheduler. First one should note that, if the execution times are constant, then the relation, $U = \sum_{i=1}^{n} C_i f_i$ (where $f_i = 1/h_i$ is the frequency of the task) is a linear function (while it would not be as a function of the task periods). A linear model can therefore be considered for the scheduling controller design. Now, using (4), the estimated requested CPU load is:

$$\hat{U}(kh_S) = \frac{(1-\lambda)q^{-1}}{1-\lambda q^{-1}} \sum_{i=1}^{n} \overline{c}_i(kh_S) f_i(kh_S) \quad (5)$$

where q stands for the shift operator. For the control design, we have chosen to consider a "normalised" control model (i.e independent on the execution time) as $H(z^{-1}) = \frac{(1-\lambda)z^{-1}}{1-\lambda z^{-1}}$. As illustration, in the two control tasks system presented in the following section, the control scheme is therefore as in figure 3 where the estimated execution-times are used on-line to adapt the gain of the controller for the original CPU system (5).

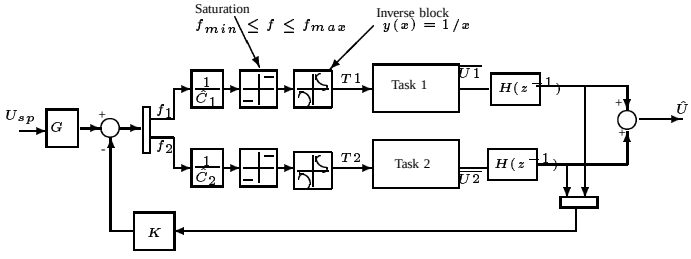

Fig. 3. Control scheme for CPU resources

In Fig. 3, G is a matrix gain that ensures a unit static gain in closed-loop.

According to this control scheme where the resource utilization of each task is measured, we have chosen the LQG control theory that allows to weigth differently the two control tasks, according to the resources they have to be allowed.

IV. Stabilization of discrete time-delay systems

Few results have been obtained for discrete time-delay systems, mainly due to the fact that, when the delay is known, the augmentation approach can be used to convert the time-delay system into a free delay one, but of higher dimension. Nevertheless, if the delay vary the dimension of the non-delayed system will vary accordingly, which is not acceptable. Moreover, when the delay is not exactly known, this is not applicable. This motivates the need in real-time control, to consider discrete-time representations with time-delay.

Using the control law (2), the closed-loop system is then:

$$x(k+1) = Ax(k) + BKx(k-d)$$

Since the closed-loop system is a state-delayed system, specific method to study the stability of such systems must be used in order to design the feedback gain K. Two kinds of stability results can be obtained: either delay-independent, or delay-dependent ones. Many results have been obtained in this framework for ten years (see for instance [16], [17] for continuous-time system mainly. For discrete-time systems let us cite the papers of [18], [19] as they consider systems with unknown delay, with uncertainties, and eventually a disturbance.

In the considered framework of real-time control with communication delays, experiments may allow delay measurements (see for instance [20]). Even if such delays are unknown, it can be possible to estimate a bound on the control delay, i.e. a maximal delay. We then focus here on delay dependent methods for discrete time-delay systems, that ensure stability for a maximum allowable delay.

In this framework, the following results gives a stability condition for systems with unknown delay.

Proposition 1: [18] If there exist P, Q, positive definite, and X, Y, Z such that

$$\begin{bmatrix} (1,1) & -Y & A_0^T P & (1,4) \\ -Y^T & -Q & A_1^T P & \bar{d}A_1^T Z \\ PA_0 & PA_1 & -P & 0 \\ (1,4)^T & \bar{d}ZA_1 & 0 & -\bar{d}Z \end{bmatrix} < 0, \quad (6)$$

$$\begin{bmatrix} X & Y \\ Y^T & Z \end{bmatrix} \geq 0 \quad (7)$$

with $(1,1) := -P + \bar{d}X + Y + Y^T + Q$, $(1,4) := \bar{d}(A_0 - I)^T Z$, then the system $x(k+1) = A_0 x(k) + A_1 x(k-d)$ is asymptotically stable for any time-delay d satisfying $0 \leq d \leq \bar{d}$.

The main theoretical result of the paper is the design of the discrete state feedback control law for system (1), using the above proposition.

Proposition 2: If there exist P, Q, positive definite and X, Y, S such that

$$\begin{bmatrix} (1,1) & -Y & PA^T & (1,4) \\ -Y^T & -Q & S^TB^T & \bar{d}S^TB^T \\ AP & BS & -P & 0 \\ (1,4)^T & \bar{d}BS & 0 & -\bar{d}P \end{bmatrix} < 0, \quad (8)$$

$$\begin{bmatrix} X & Y \\ Y^T & P \end{bmatrix} \geq 0 \quad (9)$$

with $(1,1) := -P+\bar{d}X+Y+Y^T+Q$, $(1,4) := \bar{d}P(A-I)^T$, then the system (1) with the control law $u(k) = Kx(k)$, $K = SP$, is asymptotically stable for any time-delay d satisfying $0 \leq d \leq \bar{d}$.

Proof: Let us consider the system (1) with the state feedback $u(k) = Kx(k)$. Assume that the LMIs (8-9) are satisfied for some matrices P, Q, X, Y and S.
Now, let $P_0 = P^{-1}$, $Q_0 = P_0QP_0$, $X_0 = P_0XP_0$ and $Y_0 = P_0YP_0$. Multiplying on both sides, the LMI (8) (resp(9)) by the positive symmetric matrix $T = diag[P_0, P_0, P_0, P_0]$ (resp.$T = diag[P_0, P_0]$), one obtains the following equivalent LMIs:

$$\begin{bmatrix} (1,1) & -Y_0 & A^TP_0 & (1,4) \\ -Y_0^T & -Q_0 & P_0S^TB^TP_0 & \bar{d}P_0S^TB^TP_0 \\ P_0A & P_0BSP_0 & -P_0 & 0 \\ (1,4)^T & \bar{d}P_0BSP_0 & 0 & -\bar{d}Z_0 \end{bmatrix} < 0, \quad (10)$$

$$\begin{bmatrix} X_0 & Y_0 \\ Y_0^T & P_0 \end{bmatrix} \geq 0 \quad (11)$$

where $(1,1) := -P_0 + \bar{d}X_0 + Y_0 + Y_0^T + Q_0$, $(1,4) := \bar{d}(A_0 - I)^TP_0$.
Choosing $K = SP_0$ and $Z_0 = P_0$, the LMIs (10-11) correspond to the LMIs (6-7), with $A_0 = A$ and $A_1 = BK$. By proposition 1 the closed-loop system (1-2) is asymptotically stable for $0 \leq d \leq \bar{d}$. This ends the proof. ■

V. Integrated control-scheduling for a single task control systems

In this part, we describe our methodology for the feedback-scheduling of a single control task in the case of the pendulum in downward position, presented in [8], i.e.

$$\begin{cases} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{cases} \quad (12)$$

where $A = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\xi\omega_0 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ -\omega_0/g \end{bmatrix}$ and $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$, with $\omega_0 = 3.77$, $\xi = 0.2$, $g = 9.81$. y is the controlled output and x is here assumed to be measured.
According to the choice of a sampling period h, the following discrete model is obtained:

$$x(k+1) = Ax(k) + Bu(k) \quad (13)$$

In this case, the nominal sampling period is $0.1sec$, and is assumed to be changed between $0.02s$ and $0.5s$ by steps of $0.02s$. To this model, the control delay must be added to lead to the considered system representations (1).

Simulations have been performed using Truetime, a Matlab/Simulink toolbox for real-time control [21].

A. State feedback control design

To illustrate the interest of the provided results in control design, an open-loop scheduling is considered first. First a state feedback control law is designed without taking into account the control delay, using a pole placement method which leads to $K = [33.9332 \;\; 19.1858]$.
Then using the result presented in section 4, a control law is obtained for any unknown time-delay satisfying $0 \leq d \leq 1.95s$. The LMIs (8-9) are solved and we get:

$$P = 10^3 * \begin{bmatrix} 0.43 & -0.53 \\ -0.53 & 5.39 \end{bmatrix}, \; Q = \begin{bmatrix} 37.66 & -119.97 \\ -119.97 & 555.98 \end{bmatrix}$$

$$X = \begin{bmatrix} 20.24 & -80.14 \\ -80.14 & 326.18 \end{bmatrix}, \; Y = 10^3 * \begin{bmatrix} -0.07 & 0.27 \\ 0.27 & -1.17 \end{bmatrix}$$

$$S = 10^4 * \begin{bmatrix} 0.61 & -2.41 \end{bmatrix}, K = \begin{bmatrix} -9.85 & 3.50 \end{bmatrix}$$

A control input delay of $0.5sec$ is applied, i.e. 5 times the nominal sampling period. Results are presented in figures 4 and 5.

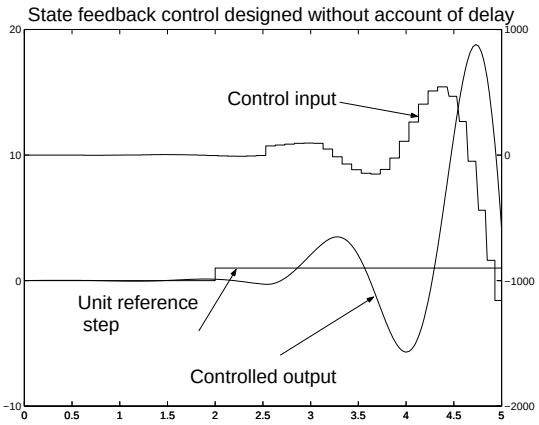

Fig. 4. Classical state feedback control - control delay =0.5 sec

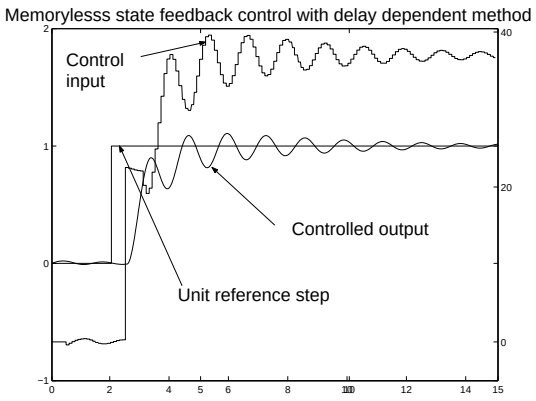

Fig. 5. Memoryless delay dependent control - control delay =0.5 sec

Clearly the classical design method leads to an unstable

closed-loop system, in the presence of delays, while the proposed delay dependent stabilization method keeps the closed-loop stability. This is a real advantage if either communication delays are present, or if the control task is preempted by a higher priority task during 5 sampling periods.
Note that this control law has been calculated for a range of sampling periods, i.e. $[0.02 - 0.5]$ with a step of $0.02s$, and stored in a table.

B. Feedback scheduling design with H_∞ control

We have chosen here $h_S = 0.5s$ for the period of the scheduler. First the modelling approach consists in a temporal test, with a step input on the sampling period, the CPU utilization being estimated, for each period of the scheduler, from filtered job response-time measurements of the control task. The corresponding result is presented in fig 6.

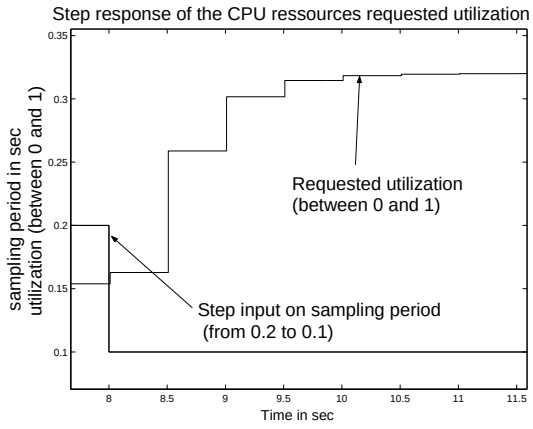

Fig. 6. Response of the estimated CPU resource to a change of sampling period in the control task

Form Fig. 6 the scheduler is therefore modelled as the first order system $G(z^{-1}) = \frac{k}{1-\lambda z^{-1}}$ with $k = 1.16$.

Using the classical methods available in control design softwares, the solution of the H_∞ control problem with the previous performance specifications, gives the controller:

$$K(z) = \frac{-0.6937z^2 - 0.4856z + 0.2081}{z^2 + 0.0004z - 0.9994}$$

C. Simulation results

This part presents the simulation results of feedback scheduling, and mainly the corresponding closed-loop performances of the pendulum, for two cases of control delay, i.e. $0.1s$ and $0.5s$. The performance of the classical state feedback control law, designed without the account of delay, and of the memoryless state feedback which ensures delay dependent closed-loop stability, are compared for these two cases.
Benchmark: the scenario used for the feedback control of the resource utilization is the following. At $t = 0$ the control task begins. At $t = 2$, the scheduling manager is switched on, and the feedback control of the computing resources is realized around 10% of utilization, leading to changes in the control sampling period. At $t = 7$, a reference step input is sent, representing an increase of 20% of availability, leading to a decrease of the control sampling period. At $t = 15$, the set-point of resources availability is set to 20%, and at $t = 25$ a new task execution is added (a workload) which corresponds to a disturbance for the feedback scheduling. At $t = 30$ the set-point of resources availability comes back to 30%.
Performance analysis: Figure 7 shows that the feedback

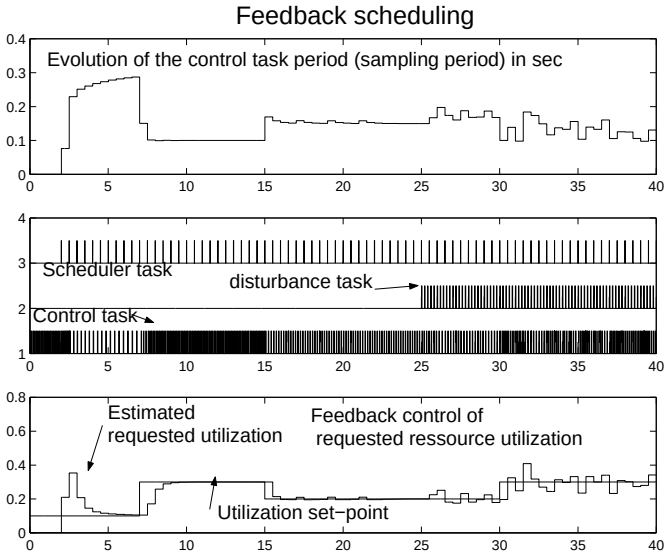

Fig. 7. Feedback scheduling

control of computing resources is done with a rise time of about $1s$, as chosen in the design step. Of course the response of the scheduler (in the way it is measured) is deteriorated by the disturbance (at $t = 25s$) but this does not much deteriorate the closed-loop system performances, as seen on figures 9 and 11.
On figures 8 and 10 we can see that the state feedback control, designed without account for delay, only tolerates a small delay of one sampling period. Larger delay, as 0.5s, leads to unstability. On the contrary, the memoryless delay dependent control law ensures robust stability of the system, in the presence of control delay as seen on figures 9 and 11. Moreover the closed-loop performance is still quite good, even better since the sampling period changes according to the feedback scheduling.

VI. Integrated control-scheduling for multi tasks control systems

We consider here a four-link planar robot (i.e. with 4 degrees of freedom) that has been considered in the TELEDIMOS project [22]. The problem under consideration is to track a desired trajectory for the position of the end-effector. Using the Lagrange formalism, and neglecting the Coriolis terms, the following model can be obtained:

$$\Gamma = M(q)\ddot{q} + G(q) + diag(\dot{q})F_v \qquad (14)$$

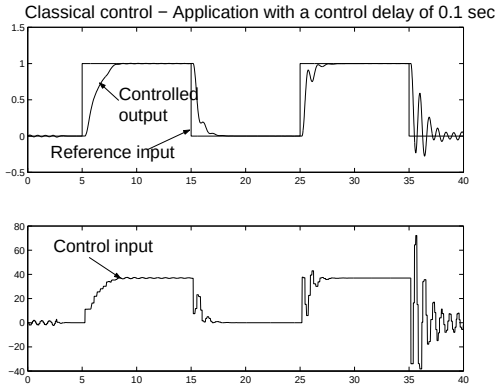

Fig. 8. Classical state feedback control - control delay=0.1 sec

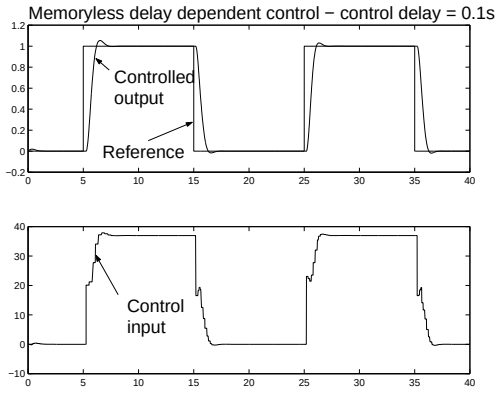

Fig. 9. Memoryless delay dependent control - control delay=0.1 sec

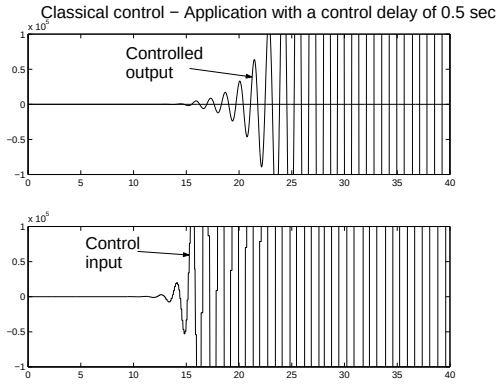

Fig. 10. Classical state feedback control - control delay=0.5 sec

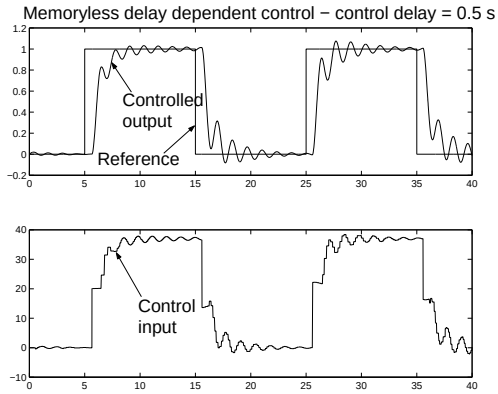

Fig. 11. Memoryless delay dependent control - control delay=0.5 sec

where q stands for the positions of the articulations, M the inertia matrix, G the gravity, and F_v the viscous friction.
The structure of the controller includes a compensation of the gravity and a PD controller for the tracking problem, as:

$$\Gamma = G(q) + K_p(q_d - q) + K_d(\dot{q_d} - \dot{q}), \qquad (15)$$

for which a control delay will be added.
This controller is divived in two tasks, i.e. a specific task is considered for the PD control and for the gravity compensation, in order to use a multi-rate controller.
In this application, the period of the feedback scheduler has been fixed to $2s$ to be much larger than the robot control tasks (which may vary here from $0.03s$ to $0.5s$. As previously TrueTime [21] is here used for simulation.
Integrated control-scheduling design: For this structure, a feedback scheduling LQG controller, that minimizes $\sum_0^\infty (x^T(k)Qx(k) + u^T(k)Ru(k))$, is designed choosing the following weighting matrices:

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 30 \end{bmatrix} \quad R = \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix} \qquad (16)$$

This allows preference over the execution of the PD controller task. Concerning the process control, our aim is to compare, for the tracking problem, two control designs, a classical PD one and a delay dependent state feedback one. The delay dependent state feedback consists in obtaining K_p and K_d from a state space representation of the robot, using the method described in section 4, for the range of sampling periods from $0.03s$ to $0.5s$, and to implement them as the PD control (15).
Benchmark: The trajectory to be tracked consists in a point to point motion, coming from the horizontal position of the robot (initial conditions $(4, 0)$ for the end-effector), to place the end-effector at a position $(2, 1)$.
For this control problem the communication delay is fixed to $0.1sec$.
Concerning feedback scheduling, the nominal set point of resource utilization is 50%. At $t = 15s$, the resource availability is set to 80%, leading to a decrease of control task periods. At $t = 45s$ a workload appears leading to a resource availability of 20%.
Results: As seen on Fig. 12 and as expected, more CPU resources are allowed to the PD control and less to the compensation of gravity. This is quite interesting as the most important is to keep the dynamic control period as quick as possible for the motion control problem. In this multi-rate control problem, and thanks to the LQG controller, the preference is given to the PD control. A larger sampling period is then given to the compensation of gravity.
From Fig. 13-14, it can be seen that the communication delay

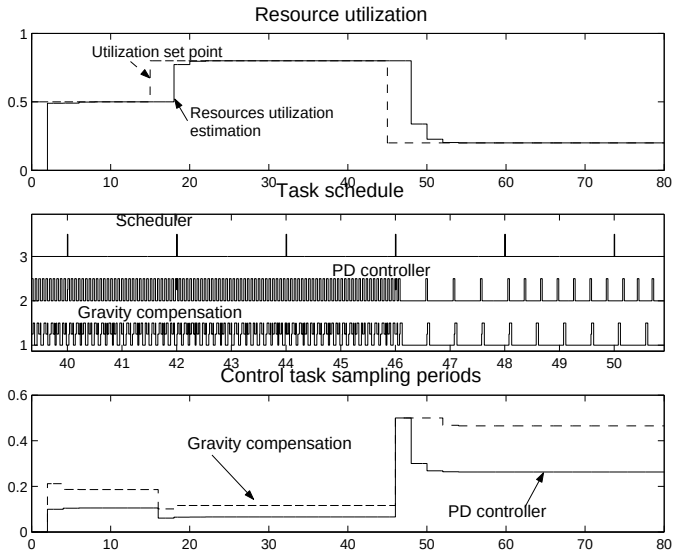

Fig. 12. Scheduling

degrades the performance of both control laws. The PD delay dependent control keeps stability and tracks the trajectory while the classical PD control leads to a non acceptable behavior.

This points out the interest of the proposed methodology for feedback control in the presence of resource utilization variations and with communication delays.

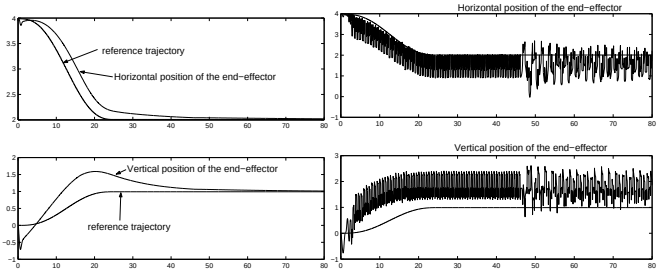

Fig. 13. Trajectory track for PD delay dependent control

Fig. 14. Trajectory track for classical PD control

VII. Conclusion

In this paper, we have provided a new design method for feedback scheduling of systems with communication delays. The control synthesis of the feedback scheduler has been provided either using the H_∞ control theory or with a LQG control. An integrated control-scheduling approach is proposed, where the state feedback control law has been designed according to a range of sampling periods. Moreover, the proposed control law ensures closed-loop stability in the presence of communication control delays, which had not been done yet. Simulation results have been given in the case of an inverted pendulum (for a single-task control system) and of a four-link planar robot (for a multi-task control system). This emphasizes the interest of this approach.

Further works may concern the improvement of the feedback scheduling scheme, particularly concerning the estimation of the computing resource utilization which is a very important issue. This should improve the modelling step of the resources utilization in the framework of control engineering.

References

[1] D. Simon, E. Castillo, and P. Freedman, "Design and analysis of synchronization for real-time closed-loop control in robotics," *IEEE Trans. on Control Systems Technology*, vol. 6, no. 4, pp. 445–461, july 1998.

[2] C. Liu and J. Layland, "Scheduling algorithms for multiprogramming in hard real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 40–61, 1973.

[3] K. Astrom and B. Wittenmark, *Computer-Controlled Systems*, 3rd ed., ser. Information and systems sciences series. New Jersey: Prentice Hall, 1997.

[4] A. Cervin, "Towards the integration of control and real-time scheduling design," Department of Automatic Control, Lund Institute of Technology, Sweden, Tech. Rep. Licentiate thesis, May 2000.

[5] K.-E. Årzén, A. Cervin, J. Eker, and L. Sha, "An introduction to control and scheduling co-design," in *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, Dec. 2000.

[6] D. Simon and F. Benattar, "Design of real-time periodic control systems through synchronisation and fixed priorities," INRIA, Tech. Rep. RR4677, december 2002.

[7] M. Ryu, S. Hong, and M. Saksena, "Streamlining real-time controller design - from performance specifications to end- to-end timing constraints," in *IEEE Real-Time Technology and Applications Symposium*, Montreal, june 1997.

[8] J. Eker, P. Hagander, and K.-E. Årzén, "A feedback scheduler for real-time control tasks," *Control Engineering Practice*, vol. 8, no. 12, pp. 1369–1378, 2000.

[9] A. Cervin, J. Eker, B. Bernhardsson, and K.-E. Årzén, "Feedback-feedforward scheduling of control tasks," *Real-Time Systems*, vol. 23, no. 1, 2002.

[10] B. Zhang, M. Branicky, and M. Phillips, "Stability of networked control systems," *IEEE Control Systems Magazine*, pp. 84–99, Feb. 2001.

[11] A. Fattouh, O. Sename, and J.-M. Dion, "Pulse controller design for linear time-delay systems," in *IFAC Symposium on System Structure and Control*, Prague, 2001.

[12] J. Eker and A. Cervin, "A matlab toolbox for real-time and control systems co-design," in *6th Int. Conf. on Real-time Computing Systems and Applications*, Hong Kong, december 1999, pp. 320–327.

[13] C. Lu, J. Stankovic, G. Tao, and S. Son, "Feedback control real-time scheduling: Framework, modeling and algorithms," *Real-Time Systems*, vol. 23, no. 1, 2002.

[14] A. Cervin and J. Eker, "Feedback scheduling of contol tasks," in *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, Dec. 2000.

[15] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: analysis and design*. John Wiley and Sons, 1996.

[16] L. Dugard and E. I. Verriest, *(Eds) Stability and control of time-delay systems*, ser. LNCIS. Springer Verlag, 1998, vol. 228.

[17] O. Sename, "New trends in design of observers for time-delay systems," *Kybernetika*, vol. 37, no. 4, pp. 427–458, 2001.

[18] Y. S. Lee and W. H. Kwon, "Delay-dependent robust stabilization of uncertain discrete-time state-delayed systems," in *IFAC 15th World Congress*, Barcelone, Spain, 2002.

[19] M. S. Mahmoud, "Robust h_∞ control of discrete systems with uncertain parameters and unknown delays," *Automatica*, vol. 36, pp. 627–635, 2000.

[20] J. Nilsson, "Real-time control systems with delays," Ph.D. dissertation, Department of Automatic Control, Lund Institute of Technology, Sweden, Jan. 1998.

[21] D. Henriksson, A. Cervin, and K.-E. Årzén, "Truetime: Simulation of control loops under shared computer resources," in *Proceedings of the 15th IFAC World Congress on Automatic Control*, Barcelona, Spain, July 2002.

[22] D. Simon, *TELEDIMOS, Real-Time Simulator: Updated Functional Specification Document*, INRIA Rhne-Alpes, September 2001.