Optimal Online Sampling Period Assignment: Theory and Experiments

Anton Cervin, Manel Velasco, Pau Martí, and Antonio Camacho

***Abstract*—In embedded systems, the computing resources are often scarce and several control tasks may have to share the same computer. In this brief, we assume that a set of feedback controllers should be implemented on a single-CPU platform. We study the problem of optimal sampling period assignment, where the goal is to assign sampling rates to the controllers so that the overall control performance is maximized. We derive expressions relating the expected cost over a finite horizon to the sampling period, the computational delay, and the amount of noise acting on the plant. Based on this, we develop a feedback scheduler that periodically assigns new sampling periods based on estimates of the current plant states and noise intensities. Extensive experiments show that online sampling period assignment can deliver significantly better control performance than the state-of-the-art, static period assignment.**

***Index Terms*—Cost function, double integrator, electrical circuits, embedded systems, feedback scheduling, optimal control.**

I. Introduction

TODAY, the vast majority of all controllers are implemented using computers, relying on periodic sampling and control. The sampling rate is normally fixed and is selected in relation to the speed of the system under control. Typical rules of thumb [1] suggest that the sampling rate should be chosen to be 10 to 30 times the bandwidth of the closed-loop system. Faster sampling implies better disturbance rejection and smaller sensitivity to modelling errors. In embedded systems with very limited computational resources, it may not be possible to let the controller run at the desired rate. This can be true especially if there are many tasks competing for the same CPU. Hence, design of embedded systems always involves tradeoffs between the various tasks in the system.

The tradeoff between different feedback controllers in an embedded system was first formulated as an offline optimization problem in a seminal paper by Seto *et al.* [2]. The performance of each controller was captured by a cost function that described the relationship between the sampling rate and the quality of control. Assuming that the cost versus rate for each controller can be approximated by an exponentially decreasing function, the paper gave an optimization algorithm that assigns optimal sampling rates to the controllers, subject to a CPU utilization constraint.

Later work has refined the model of Seto *et al.* and also moved the optimization algorithm from offline use to online use. Eker *et al.* [3] considered linear quadratic (LQ) state feedback controllers and derived expressions relating the LQ-cost to the sampling interval. They also proposed an online optimization algorithm for sampling period assignment, that iteratively adjusted the sampling rates based on the CPU load. An extension to general linear dynamic controllers is found in [4].

Martí *et al.* [5] introduced feedback from the actual control performance by incorporating the current plant states into a heuristic cost function. The same authors presented an extension [6] where the relation between its cost function and standard quadratic cost functions was established. Henriksson *et al.* [7] further formalized this approach for linear quadratic controllers by incorporating the current plant states into a finite-horizon quadratic cost function, which also took the expected future plant noise into account. The extension to the general case of linear controllers was given in [8].

Palopoli *et al.* [9] developed optimal period assignment for a set of state feedback controllers using the stability radius as a performance criterion. No online optimization was performed, however.

In a parallel line of research, Ben Gaid *et al.* [10], [11] presented a complementary approach for the optimal integrated control and real-time scheduling of control tasks. It combined non-preemptive optimal cyclic schedules according to the H_2 performance criterion, with an efficient online scheduling heuristic in order to improve responsiveness to disturbances.

In this brief, we present an online sampling period assignment approach that extends the model of Seto *et al.* to capture several important properties of real controllers in the optimization problem. The performance of each controller is captured in a finite horizon cost function, taking into account: 1) the sampling period; 2) the computational delay; and 3) the amount of noise acting on the plant. In addition, the cost function is developed for general linear dynamic controllers (and not only for state feedback controllers as in much of previous work).

Since finding an analytical solution to the optimization problem is not possible in the general case even for simpler cost functions [7], we present an algorithm that allows solving the problem at run-time. By evaluating the cost function, it is noted that most of the terms can be computed offline. Then, for convex functions on the sampling period, an online procedure that approximates the optimal solution is developed that allows identifying the set of optimal sampling periods.

Noting that the evaluation of the cost function strongly depends on the noise intensities, we complement the feedback scheduler with a noise estimator. Two different standard noise estimators are suggested, and their pros and cons discussed and evaluated. We show that the noise estimators are effective when

Manuscript received January 04, 2010; revised May 05, 2010; accepted June 01, 2010. Manuscript received in final form June 10, 2010. Date of publication July 12, 2010; date of current version June 17, 2011. Recommended by Associate Editor P. J. Mosterman. This work was supported in part by the Swedish Research Council, C3DE Spanish CICYT DPI2007-61527, and by ArtistDesign NoE IST-2008-214373.

A. Cervin is with Lund University, Automatic Control LTH, SE-22100 Lund, Sweden (e-mail: anton@control.lth.se).

M. Velasco, P. Martí, and A. Camacho are with Technical University of Catalonia, Automatic Control Department, 08028 Barcelona, Spain.

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2010.2053205

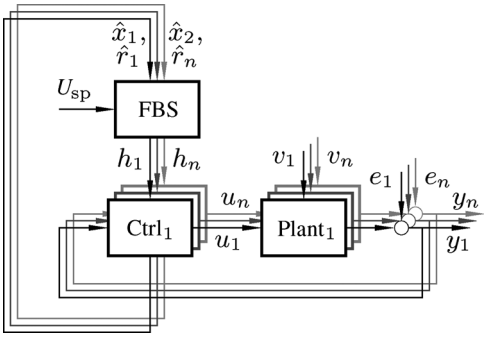

Fig. 1. FBS assigns sampling periods $h_1 \dots h_n$, based on state estimates $\hat{x}_1 \dots \hat{x}_n$ and noise intensity estimates $\hat{r}_1 \dots \hat{r}_n$, to meet the utilization set-point $U_{\rm sp}$ and to optimize the performance of the control loops.

the noise intensity changes are slower than the rate of execution of the period assignment algorithm.

Finally, we provide a proof-of-concept implementation, where the feedback scheduling approach is put to test in a real system. We study different key parameters of the proposed approach with respect to control performance and resource utilization. In addition, we provide a comparative performance analysis of our approach with respect to previous related work.

This brief extends the preliminary results presented in [8]. An extended version of this brief can be found in [12]. The rest of this brief is outlined as follows. In Section II, the problem formulation is given. In Section III, formulas for the quadratic cost of a general linear controller are derived. The algorithm for the online optimization is given in Section IV. The experimental setup and results are described in Section V. Finally, Section VI concludes this brief.

II. Problem Formulation

A. Plant and Controller Models

We assume a system in which n physical plants should be controlled by a computer with limited computational resources. The control is achieved by n concurrent control tasks executing in the computer, each task being responsible for the sampling, control computation, and actuation in one loop. There exists an additional task in the system, the feedback scheduler (FBS), that may change the sampling rates of the tasks depending on the current state of the system. The overall situation is depicted in Fig. 1.

More formally, each plant $i = 1 \dots n$ is described by a continuous-time linear stochastic system

$$\begin{aligned} \dot{x}_i(t) &= A_i x_i(t) + B_i u_i(t - \tau_i) + v_{ci}(t) \\ y_i(t) &= C_i x_i(t) + e_i(t) \end{aligned} \tag{1}$$

where x_i is the plant state, u_i is the control signal, y_i is the measured output, A_i, B_i, and C_i are matrices of appropriate sizes, v_{ci} is a continuous-time white noise process with intensity R_{1ci}, e_i is a discrete-time Gaussian white noise process with variance R_{2i}, and $\tau_i = \alpha_i h_i$, $0 < \alpha_i < 1$, is a constant time delay representing the input-output delay in the implementation (see Section II-B). To capture different operating conditions, we allow the process noise intensity to be time varying

$$R_{1c_i}(t) = r_i(t)\bar{R_{1c_i}} \tag{2}$$

where $r_i(t)$ is a positive scalar and $\bar{R_{1ci}}$ is the nominal noise intensity. The controller can be constant, designed assuming a constant noise level, or time-varying, assuming different noise intensities given by different $r_i(t)$ values.

Sampling the plant (1) with the interval h_i and control delay $\tau_i = \alpha_i h_i$ yields the discrete-time system (see [1])

$$\begin{aligned} z_i[k+1] &= \Phi_i(h_i) z_i[k] + \Gamma_i(h_i) u_i[k] + v_i[k] \\ y_i[k] &= \Theta_i z[k] + e_i[k] \end{aligned} \tag{3}$$

where

$$\begin{aligned} z_i[k] &= \begin{bmatrix} x_i[k] \\ u_i[k-1] \end{bmatrix} \\ \Phi_i(h_i) &= \begin{bmatrix} \phi_i(h_i) & \phi_i(h_i - \alpha_i h_i)\gamma_i(\alpha_i h_i) \\ 0 & 0 \end{bmatrix} \\ \Gamma_i(h_i) &= \begin{bmatrix} \gamma_i(h_i - \alpha_i h_i) \\ I \end{bmatrix} \\ \Theta_i &= [C_i \quad 0] \end{aligned}$$

and

$$\phi_i(t) = e^{A_i t}, \quad \gamma_i(t) = \int_0^t \phi_i(s) B_i ds.$$

Assuming the noise level to be constant over the sampling interval, the covariance of the discrete-time white noise process v_i is given by

$$R_{1i}(h_i) = \begin{bmatrix} \int_0^{h_i} \phi_i(s) R_{1ci} \phi_i^T(s) ds & 0 \\ 0 & 0 \end{bmatrix}. \tag{4}$$

Each controller $i = 1 \dots n$ is described by a discrete-time linear system with adjustable sampling interval h_i

$$\hat{x}_i[k+1] = F_i(h_i)\hat{x}_i[k] + G_i(h_i) y_i[k] \tag{5}$$

$$u_i[k] = H_i(h_i)\hat{x}_i[k] + K_i(h_i) y_i[k] \tag{6}$$

where $\hat{x}_i$ is the controller state (representing the plant state estimate) and F_i, G_i, H_i, and K_i are matrices of appropriate sizes.

The performance of each control loop $i = 1 \dots n$ is measured by a finite-horizon continuous-time quadratic cost function

$$J_i^{T_{\rm fbs}} = \mathrm{E} \int_0^{T_{\rm fbs}} \left\| \begin{matrix} x_i(t) \\ u_i(t) \end{matrix} \right\|_{Q_{c_i}}^2 dt \tag{7}$$

where

$$Q_{c_i} = \begin{bmatrix} Q_{1c_i} & Q_{12c_i} \\ Q_{12c_i}^T & Q_{2c_i} \end{bmatrix}$$

is a positive semidefinite matrix, and $T_{\rm fbs}$ is the period of the feedback scheduler. For this kind of cost function, a linear-quadratic Gaussian (LQG) controller [1] is optimal. Note that our theoretical framework does not require the controllers to be

optimal. This means that PID controllers or controllers designed using (non-optimal) pole placement may be included, as long as their performance is evaluated using a quadratic cost function of the type (7).

B. Real-Time Scheduling Model

In contrast to previous work on optimal sampling period assignment [2], [3], [5], [7], [10], in this work we explicitly take the real-time scheduling and the computational delay into account in the optimization. It is assumed that the worst-case execution time of each controller i is described by

$$E_i = E_i^{\text{Calculate}} + E_i^{\text{Update}} \tag{8}$$

where $E_i^{\text{Calculate}}$ is the worst-case execution time for calculating the control signal [essentially (6) above], and E_i^{Update} is the worst-case execution time for updating the controller state [(6) above].

In the implementation, the input (I) operation (i.e., the analog-to-digital (A/D) conversion of $y[k]$), the Calculate portion of the code, the output (O) operation (i.e., the digital-to-analog (D/A) conversion of $u[k]$), and the Update portion of the code are assumed to be scheduled as separate subtasks. The I/O operations are scheduled at interrupt priority and are assumed to take negligible time to execute. The Calculate and Update subtasks are scheduled using the earliest-deadline-first (EDF) algorithm [13], [14]. The Calculate subtask is released at the start of each period, with a relative deadline $\alpha_i h_i$, where

$$\alpha_i = \frac{E_i^{\text{Calculate}}}{E_i^{\text{Calculate}} + E_i^{\text{Update}}}. \tag{9}$$

The Update subtask is released with the offset $\alpha_i h_i$, with a relative deadline $(1-\alpha_i)h_i$. This implies that the input-output delay of controller i is constant and given by $\tau_i = \alpha_i h_i$. Moreover, the organization of the code ensures that controller i has a constant processor demand [14] of $U_i = E_i/h_i$. Schedulability of all subtasks can hence be guaranteed if and only if $\sum_{i=1}^{n} U_i \leq 1$ which is the schedulability condition for EDF.

C. Optimization Problem

The feedback scheduler is assumed to have knowledge of all plant and controller parameters, except the noise intensities, which are to be estimated online. The scheduler runs as a high-priority task with a fixed period $T_{\text{fbs}} \gg h_i$ and execution time E_{fbs}. Upon invocation at time t_0, the scheduler is informed about the current controller states $\hat{x}_1(t_0) \dots \hat{x}_n(t_0)$ (representing, for instance, estimates of the current plant states). The scheduler should then assign new sampling intervals $h_1 \dots h_n$ such that the total expected cost over the next T_{fbs} units of time is minimized. This is formulated as the following optimization problem to be solved online:

$$\begin{aligned} &\min_{h_1 \dots h_n} \sum_{i=1}^{n} J_i^{T_{\text{fbs}}} \\ &\text{subj. to} \sum_{i=1}^{n} \frac{E_i}{h_i} \leq U_{\text{sp}} - U_{\text{fbs}}. \end{aligned} \tag{10}$$

Here, U_{sp} is the CPU utilization set-point, which must be smaller than 1, and $U_{\text{fbs}} = E_{\text{fbs}}/T_{\text{fbs}}$ is the processor demand of the feedback scheduler. The optimization problem is convex if the cost functions are convex functions of $1/h_i$ (see [2]). Efficient online solution of (10) is discussed in Section IV.

III. Evaluation of the Cost Functions

To solve the optimization problem, we must be able to evaluate the cost function (7) for each control loop online. In this section, we therefore derive expressions for the stationary and transient cost of a linear discrete-time controller regulating a linear plant with a fixed time delay. Throughout this section, for clarity, we drop the plant/controller index i.

A. Sampling the Cost Function

Assuming that T_{fbs} can be evenly divided into h, the cost can be written as

$$J^{T_{\text{fbs}}} = \sum_{k=0}^{T_{\text{fbs}}/h-1} J[k] \tag{11}$$

where $J[k]$ is the cost in the kth sampling interval. By sampling (7), this cost can be expressed as (12), shown at the bottom of the next page, where

$$\begin{aligned} Q_1(h) &= \begin{bmatrix} Q_{1a}(h) & Q_{1b}(h) \end{bmatrix} \\ Q_{1a}(h) &= \begin{bmatrix} q_1(\alpha h) + \phi^T(\alpha h) q_1(h-\alpha h)\phi(\alpha h) \\ q_{12}^T(\alpha h) + \phi(\alpha h) \end{bmatrix} \\ Q_{1b}(h) &= \begin{bmatrix} q_{12}(\alpha h) + \phi^T(\alpha h) \\ q_2(\alpha h) + \gamma^T(\alpha h) q_1(h-\alpha h)\gamma(\alpha h) \end{bmatrix} \\ Q_{12}(h) &= \begin{bmatrix} \phi^T(\alpha h) q_{12}(h-\alpha h) \\ \gamma^T(\alpha h) q_{12}(h-\alpha h) \end{bmatrix} \\ Q_2(h) &= q_2(h-\alpha h) \\ J_{\text{const}}(R_1, h) &= \operatorname{tr} Q_{1c} \int_0^h \int_0^s \phi(s) R_{1c} \phi^T(s) ds\, dh \end{aligned}$$

and

$$q_1(t) = \int_0^t \phi^T(s) Q_{1c} \phi(s) ds$$
$$q_{12}(t) = \int_0^t \phi^T(s) \left(Q_{1c}\gamma(s) + Q_{12c} \right) ds$$
$$q_2(t) = \int_0^t \left(\gamma^T(s) Q_{1c} \gamma(s) + 2\gamma^T(s) Q_{12c} + Q_{2c} \right) ds.$$

The term $J_{\text{const}}(R_1, h)$ is due to the inter-sample noise [15].

B. Evaluation of the Cost

Combining (3) with (5) and (6), the closed-loop system can be written as

$$\begin{bmatrix} z[k+1] \\ \hat{x}[k+1] \end{bmatrix} = \Phi_{cl}(h) \begin{bmatrix} z[k] \\ \hat{x}[k] \end{bmatrix} + \Gamma_{cl}(h) \begin{bmatrix} v[k] \\ e[k] \end{bmatrix} \tag{13}$$

where

$$\Phi_{cl}(h) = \begin{bmatrix} \Phi(h) + \Gamma(h)K(h)\Theta & \Gamma(h)H(h) \\ G(h)\Theta & F(h) \end{bmatrix}$$
$$\Gamma_{cl}(h) = \begin{bmatrix} I & \Gamma(h)K(h) \\ 0 & G(h) \end{bmatrix}.$$

One difficulty here is that $u[k]$ appears in the cost function, but not in the formulation of the closed-loop system. However, noting that

$$u[k] = H(h)\hat{x}[k] + K(h)\Theta z[k] + K(h)e[k] \tag{14}$$

the cost (12) can be rewritten as

$$J[k] = \mathrm{E} \left\| \begin{matrix} z[k] \\ \hat{x}[k] \end{matrix} \right\|^2_{Q_{cl}(h)} + K^T(h)Q_2(h)K(h)R_2 + J_{\text{const}}(R_1, h) \tag{15}$$

where

$$Q_{cl}(h) = \begin{bmatrix} Q_{1cl}(h) & Q_{12cl}(h) \\ Q_{12cl}^T(h) & Q_{2cl}(h) \end{bmatrix}$$
$$Q_{1cl}(h) = Q_1(h) + Q_{12}(h)K(h)\Theta + \Theta^T K^T(h) Q_{12}^T(h) + \Theta^T K^T(h) Q_2(h) K(h) \Theta$$
$$Q_{12cl}(h) = Q_{12}(h)H(h) + \Theta^T K^T(h) Q_2(h) H(h)$$
$$Q_{2cl}(h) = H^T(h) Q_2(h) H(h).$$

Finally, assume that the initial state is $\left[\begin{smallmatrix} z[0] \\ \hat{x}[0] \end{smallmatrix}\right]$. The cost can then be evaluated as

$$\begin{aligned} J^{T_{\text{fbs}}} = & \begin{bmatrix} z[0] \\ \hat{x}[0] \end{bmatrix}^T S[0] \begin{bmatrix} z[0] \\ \hat{x}[0] \end{bmatrix} \\ & + \sum_{k=1}^{T_{\text{fbs}}/h} \operatorname{tr} S[k] \Gamma_{cl}(h) \begin{bmatrix} R_1 & 0 \\ 0 & R_2 \end{bmatrix} \Gamma_{cl}^T(h) \\ & + \frac{T_{\text{fbs}}}{h} K^T(h) Q_2(h) K(h) R_2 \\ & + \frac{T_{\text{fbs}}}{h} J_{\text{const}}(R_1, h) \end{aligned} \tag{16}$$

where

$$S[k] = \sum_{m=0}^{T_{\text{fbs}}/h - k} \left(\Phi_{cl}^T(h) \right)^m Q_{cl}(h) \left(\Phi_{cl}(h) \right)^m.$$

In summary, we have shown that the cost can be written as a sum of four terms, where the two first terms represent the cost due to the initial state, while the last two terms represent the expected cost due to future noise.

IV. Efficient Online Optimization

Computing (16) online would be expensive. However, it can be noted that almost everything in (16) can be precomputed offline and stored in a lookup table. Online, we must only account for the current initial state, noise intensity, and sampling interval.

$$\begin{aligned} J[k] &= \mathrm{E} \left\{ \int_{kh}^{kh+\alpha h} \left\| \begin{matrix} x(t) \\ u(t) \end{matrix} \right\|^2_{Q_c} dt + \int_{kh+\alpha h}^{kh+h} \left\| \begin{matrix} x(t) \\ u(t) \end{matrix} \right\|^2_{Q_c} dt \right\} \\ &= \mathrm{E} \left\{ \int_0^{\alpha h} \left\| \begin{matrix} \phi(t)x[k] + \gamma(t)u[k-1] \\ u[k-1] \end{matrix} \right\|^2_{Q_c} dt + \int_0^{h-\alpha h} \left\| \begin{matrix} \phi(t)\left(\phi(\alpha h)x[k] + \gamma(\alpha h)u[k-1]\right) + \gamma(t)u[k] \\ u[k] \end{matrix} \right\|^2_{Q_c} dt \right\} + J_{\text{const}}(R_1, h) \\ &= \mathrm{E} \begin{bmatrix} z[k] \\ u[k] \end{bmatrix}^T \begin{bmatrix} Q_1(h) & Q_{12}(h) \\ Q_{12}^T(h) & Q_2(h) \end{bmatrix} \begin{bmatrix} z[k] \\ u[k] \end{bmatrix} + J_{\text{const}}(R_1, h) \end{aligned} \tag{12}$$

TABLE I
ORDERED TABLE WITH $\Pi_{i,k}$ VALUES FOR EACH PLANT AND SAMPLING PERIODS

	h_1	h_2	...	h_{j-1}	h_j
Plant 1	$\Pi_{1,1}$	$\Pi_{1,2}$	...	$\Pi_{1,(j-1)}$	$\Pi_{1,j}$
Plant 2	$\Pi_{2,1}$	$\Pi_{2,2}$	...	$\Pi_{2,(j-2)}$	$\Pi_{2,j}$
...	...	...	...	...	...
Plant n	$\Pi_{n,1}$	$\Pi_{n,2}$	...	$\Pi_{n,(j-1)}$	$\Pi_{n,j}$

A. Approximate Evaluation of the Cost

In (16), the initial state m_0 is somewhat problematic, since the plant state $z[0]$ is not known by the feedback scheduler. However, if we assume that the controller contains a state observer and that $\hat{x}[0]$ represents the current state estimate, we may use

$$m_0 \approx \begin{bmatrix} \hat{x}[0] \\ \hat{x}[0] \end{bmatrix}. \tag{17}$$

Finally, under the assumption that the noise level will be constant during the scheduling period and equal to $R_1 = r_1 \bar{R}_1$, the expected cost can be written as

$$J^{T_{\text{fbs}}} \approx \hat{x}^T[0] S_x(h) \hat{x}[0] + r_1 J_1(h) + J_2(h) \tag{18}$$

where

$$\begin{aligned} S_x(h) &= S_{11}[0] + S_{12}[0] + S_{21}[0] + S_{22}[0] \\ J_1(h) &= \sum_{k=1}^{T_{\text{fbs}}/h} \operatorname{tr} S[k] \Gamma_{cl}(h) \begin{bmatrix} \bar{R}_1 & 0 \\ 0 & 0 \end{bmatrix} \Gamma_{cl}^T(h) \\ &\quad + \frac{T_{\text{fbs}}}{h} J_{\text{const}}(\bar{R}_1, h) \\ J_2(h) &= \sum_{k=1}^{T_{\text{fbs}}/h} \operatorname{tr} S[k] \Gamma_{cl}(h) \begin{bmatrix} 0 & 0 \\ 0 & R_2 \end{bmatrix} \Gamma_{cl}^T(h) \\ &\quad + \frac{T_{\text{fbs}}}{h} K^T(h) Q_2(h) K(h) R_2. \end{aligned}$$

Note that $S_x(h)$ is a matrix function, while $J_1(h)$ and $J_2(h)$ are scalar-valued functions. They can be computed offline for each plant i and sampling period h. Note that they do not depend on the current state estimate nor on the current noise level. Therefore, they can be stored in a table for run-time lookup. The online computations involves looking up the relevant values from the table and applying (18).

B. Structure

For the implementation, we construct a table with entries $\Pi(i,k) = \{S_x(i,k), J_1(i,k), J_2(i,k)\}$ (see Table I), where i is the plant and k is the position of the sampling period inside an ordered vector with h_1 being the smallest sampling period and h_j the largest. Note that the number of sampling periods h^* for each control task within the specified ranges $[h_i^{min}, h_i^{max}]$ depends on the granularity H, defined as the difference between two consecutive periods, $H = h_{k+1} - h_k$ (and $T_{fbs} \bmod H = 0$).

C. Algorithm

Given estimates of the current noise levels ($\hat{r}_i$) and plant states ($\hat{x}_i$) the algorithm that we propose searches for the minimum of the cost function. The basic idea is to start with a non-schedulable solution where all task periods at their minimal values. Then the task periods are incremented successively in such a way that the cost function increment $\Delta J(i)$ in each step is always minimal, until the task set is feasible. The algorithm is shown in Fig. 2.

```
h = [h_1^min, h_2^min, ..., h_n^min]
C = [C_1, C_2, ..., C_n]
x = [x_1, x_2, ..., x_n]
r = [r_1, r_2, ..., r_n]
while sum_{k=1}^{n} C_k/h_k > U_sp - U_fbs do
   lower_cost_task = 1
   for i = 1 to n do
      if h(i) + H <= h_i^max then
         ΔJ(i) = J(h(i) + H, x(i), r(i)) - J(h(i), x(i), r(i))
         if ΔJ(i) < ΔJ(lower_cost_task) then
            lower_cost_task = i
         end if
      end if
   end for
   h(lower_cost_task) = h(lower_cost_task) + H
end while
return h
```

Fig. 2. Algorithm for the implementation of the period assignment.

By doing so, the algorithm assumes that each task should run with its shortest period in order to achieve the minimum cost, i.e., the cost is a convex, increasing function on the sampling period. However, this is not a strict requirement. If the monotonicity assumption is removed, i.e., the cost is only convex, the stopping condition for the algorithm is to reach a set of feasible periods (as before) and $\forall i$, $\Delta J(i) > 0$. The second condition forces to enlarge tasks periods as long as they imply a decrease in the cost.

Note that it is possible to simplify the complexity of the optimization algorithm down to $O(kn)$, where n is the number of plants and k is the number of possible sampling periods. In addition, the linear search applied to the table could be replaced by more effective search algorithms.

D. Note on Stability

It is well known that fast switching between stabilizing controllers may lead to an unstable closed-loop system [16]. In our case, however, the switching is assumed to be infrequent in comparison with the sampling intervals of the controllers, $h_i \ll T_{\text{fbs}}$, implying that stability will not be an issue in practice. Still, if the control designer would like to assert stability for the family of controllers with different sampling intervals, it can be done using well-known techniques such as finding a common Lyapunov function for the controllers [16].

E. Disturbance Estimators

The cost function (18) depends heavily on the variance of the process noise and the measurement noise. Consequently, feedback scheduling schemes assuming stationary noise processes are not ideally suited to handle situations when the level of noise changes dynamically. In the following, we describe two different online estimators for the intensity of the process noise, R_{1c}, given measurements of the plant output, y. The estimators

are based on the residuals, $\tilde{y} = y - \hat{y}$, for which the variance is given by

$$\sigma^2 = \mathrm{E}(\tilde{y}\tilde{y}^T) = C\tilde{P}C^T \tag{19}$$

where

$$\tilde{P} = \mathrm{E}(xx^T) - 2\mathrm{E}(x\hat{x}^T) + \mathrm{E}(\hat{x}\hat{x}^T) \tag{20}$$

depends on the plant, the noise, and the controller design.

The first method, residual variance estimator (RVE), uses the observations of $\tilde{y}$ collected during the last feedback scheduler period to compute the variance estimate

$$s^2 = \frac{\sum_1^N \tilde{y}^2}{N} \tag{21}$$

where $N = T_{\mathrm{fbs}}/h$ is the number of recorded observations. The value of s^2 is then used in a lookup table of precomputed values of r_1 to obtain the corresponding estimate $\hat{r_1}$.

The second method, maximum-likelihood estimator (MLE), also sums the squares of the residuals, $\tilde{y}$, in the feedback scheduler period. Using the fact that, if $\tilde{y}_i \in N(0, \sigma)$, then

$$\frac{1}{\sigma^2}\sum_1^N \tilde{y}_i^2 \in \chi^2(N) \tag{22}$$

the MLE can be used to estimate which of several values of r_1 that is most likely given the observations.

For both estimators, $\tilde{y}^2$ can be computed by each control task in the controller update part, while the estimator is invoked by the feedback scheduler. The implementation of the MLE typically requires less resources because high accuracy of the RVE demands a large number of possible noise levels L in the controller.

V. Experiments

As a prototype and performance demonstrator, a proof-of-concept implementation of the online sampling period assignment approach is presented. The setup consists of three plants in the form of double integrator electronic circuits that are controlled by three control tasks concurrently executing in the Erika real-time kernel [17] and scheduled under the EDF scheduling algorithm. The hardware platform is a Full Flex board [18] equipped with a dsPIC micro-controller. A fourth task, acting as feedback scheduler, periodically assigns new control task periods considering the current states and noise intensities of the plants. To debug and extract information from the micro-controller and plants, an RS232 link to a standard computer has been established. This link is managed from the computer side using MATLAB. See further details in [12].

Each plant $i \in \{1, 2, 3\}$ is modeled as

$$\begin{aligned}\dot{x}_i(t) &= \begin{bmatrix} 0 & -21.28 \\ 0 & 0 \end{bmatrix} x_i(t) + \begin{bmatrix} 0 \\ -21.28 \end{bmatrix} u_i(t - \tau_i) + v_{c_i}(t) \\ y_i(t) &= [1 \quad 0] x_i(t) + e_i(t).\end{aligned} \tag{23}$$

The goal of the controller is to make the circuit output voltage y_i track a reference signal by giving the appropriate voltage input (control signal) u_i.

In addition to the implementation of the optimal sampling period selection approach, a complementary implementation has been also coded into the Erika kernel. It corresponds to the traditional static, periodic case: the three control tasks always execute with the same sampling period, irrespective of noise changes. In some of the evaluated scenarios, small modifications have been also applied to the periodic case in order to provide more illustrative performance numbers.

A. Performance Evaluation: Base Scenario

For the results presented in this section, the following parameters characterizing the base scenario have been applied. The utilization set-point is $U_{\mathrm{sp}} = 0.0135$ and the feedback scheduler task period is $T_{\mathrm{fbs}} = 1000$ ms. Sampling period choices for each controller are defined as follows. Considering that the feedback scheduler execution time is $E_{\mathrm{fbs}} = 1.6$ ms, the available CPU utilization for the three control tasks is $U_{\mathrm{sp}} - U_{\mathrm{fbs}} = 0.0135 - (1.6/1000) = 0.0119$. Considering that the execution time of each controller is $E_{1,2,3} = 0.180$ ms (0.083 ms for the calculate part and 0.097 ms for the update part), and the minimum period is specified to be $h_{1,2,3}^{\min} = 20$ ms, the longest period is $h_{1,2,3}^{\max} = 90$ ms. For the baseline periodic implementation, sampling periods for control tasks are always 40 ms, which gives the same utilization factor U_{sp}. In addition, the granularity of the optimization algorithm is set to $H = 10$ ms. Hence, each task can run at $h^* = 8$ different sampling periods.

Two noise levels named low and high are injected to the plant, $r_i(t) = 1100$. The artificial tasks injecting noise are configured in such a way that during an interval of 3000 ms, one plant is injected a high intensity noise, i.e., $r(t) = 100$, while the others two are injected a low intensity noise, i.e., $r(t) = 1$. Hence $R_{1c,low} = 1 \cdot R_{1,base}BB^T = [0\ 0; 0\ 0.1]$ and $R_{1c,high} = 100 \cdot R_{1,base}BB^T = [0\ 0; 0\ 10]$, where $R_{1,base} = 0.0002209$. The plant affected with the high intensity noise varies cyclically every 3000 ms. We refer to this noise pattern as a noise with a period of $h_{\mathrm{noise}} = 3000$ ms. In addition, changes in noise intensities happen at the beginning of the executions of the feedback scheduler (noise and feedback scheduler are fully synchronized). Also, an ideal noise estimator is considered.

Each control task implements an LQG controller considering the plant model (23), the cost function (7) with

$$Q_{1c} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad Q_{12c} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad Q_{2c} = 1$$

the noise level as described above with $R_2 = 5 \cdot 10^{-5}$ and parameterized according to the sampling period that applies among the possible choices. For the case of the controllers for the periodic baseline strategy, the parameters are the same but the noise level is specified to be the highest level that applies.

The evaluation of the optimal online sampling period assignment and the periodic case has been performed for each plant according to the cost function

$$J = \sum_{k=0}^{200} \int_{0.005k}^{0.005(k+1)} \left\| \begin{matrix} x_i(t) \\ u_i(t) \end{matrix} \right\|_{Q_{c_i}}^2 dt. \tag{24}$$

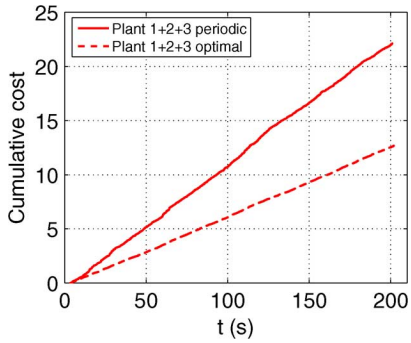

Fig. 3. Cumulative cost of the optimal sampling period assignment approach versus the standard periodic approach.

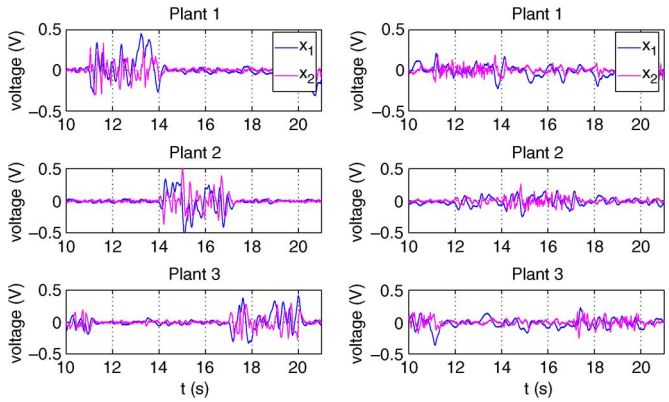

Fig. 4. Plant dynamics for the three plants. (a) Periodic strategy. (b) Feedback scheduling strategy.

Hence, each execution run lasts 200 s and plant states and control signals are recorded every 5 ms. Summing the three costs for the three plants, we obtain the accumulated cost.

The result for the base scenario is shown in Fig. 3. It shows the accumulated cost for the three controllers using the online sampling period assignment, named "Optimal", compared to the standard approach, named "Periodic": the lower the curve, the lower the cost, i.e., the better the approach. As it can be seen, after 200 s, the cost improvement of the optimal with respect to the static is around 40%.

Fig. 4 shows details of the control performance of the standard periodic policy [see Fig. 4(a)] and the online period assignment policy [see Fig. 4(b)]. Each sub-figure shows the dynamics of the three plants, each one affected by a high noise level during an interval of 3000 ms. The optimal policy has the ability of mitigating the effects of the noise due to the online sampling period adjustment.

B. Performance Evaluation: Controller Design Strategy

Fig. 5 shows the combined analysis of control performance and memory demands for the periodic and for the feedback scheduling approach following different controller design strategies. The first two "cost/memory" bars on the left show the performance of the periodic approach for two cases, where both controllers always execute at 40 ms. The cost/memory bars labeled "Periodic 1C' are the case when only one controller gain applies, which corresponds to the upper curve in

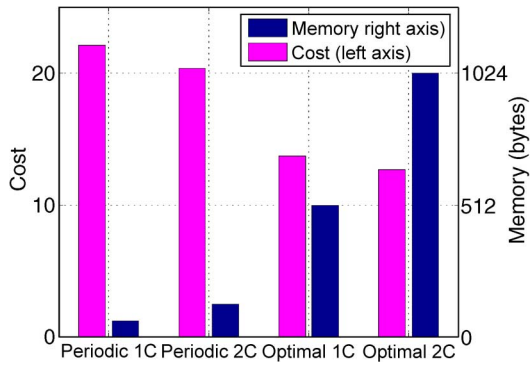

Fig. 5. Cumulative cost and memory demand analysis.

Fig. 3. The memory demand in this case is for storing one controller. The cost/memory bars labeled "Periodic 2C" are also the standard periodic case with the difference that two sets of control parameters apply depending on the noise level. The cost is a little bit better while the memory demand doubles.

The two "cost/memory" bars on the right of Fig. 5 focus on the optimal approach for two cases. The cost/memory bars labeled "Optimal 1C" are the case when the controller gain is designed for one noise level for each sampling period. The cost is much better than the previous two periodic strategies. The memory demand however is for storing $h^* = 8$ controllers. The cost/memory bars labeled "Optimal 2C" are the case when the control parameters that apply also depend on the noise level. The cost in this case is the best, which is the one also shown by the lower curve of Fig. 3. However, the memory demand is the highest, for storing h^*L controllers.

Note that in the base experiment and in the experiments in the following subsections, the controller design strategy that applies for the periodic controller is "Periodic 1C" while for the feedback scheduling approach it is "Optimal 2C." The other two bars shown in Fig. 5 illustrate alternative strategies with different tradeoffs between cost and memory.

C. Performance Evaluation: Granularity H

Fig. 6(a) shows the performance of the feedback scheduling strategy for different values of H. When the granularity is small, say $H = 1$ ms (or $H = 2$ ms), a number of $((90-20)/1)+1 = 71$ (or 36) entries appears in the search table of the optimization algorithm. As a consequence the execution time of the feedback scheduler increases, and the additional overhead that this represents leaves no room for the control tasks to speed up their rate of execution to improve control performance, as shown by the top curve (or third highest curve). When the granularity is high, say $H = 30$ ms, the number of period choices is very limited, and therefore, although the overhead of the feedback scheduler is very small, control tasks can not take advantage of the available CPU as efficiently, as shown by the second performance curve from the top. Hence, from a control point of view, the granularity H has to be chosen carefully, avoiding introducing significant overhead in the feedback scheduler but still giving enough period choices for each task. Good values in this experiment are $H = 5$, 10, or 20, which basically provides the same performance (the lower three curves). Fig. 6(b) complements

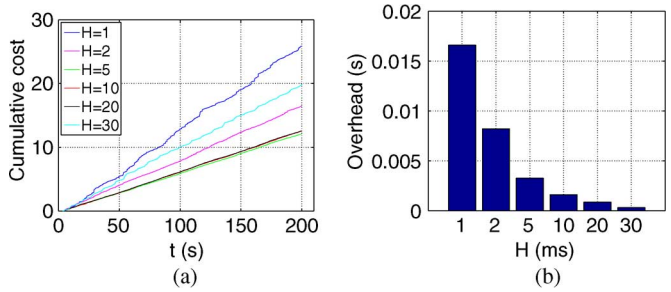

Fig. 6. Analysis of the algorithm granularity H. (a) Control performance. (b) Computational overhead.

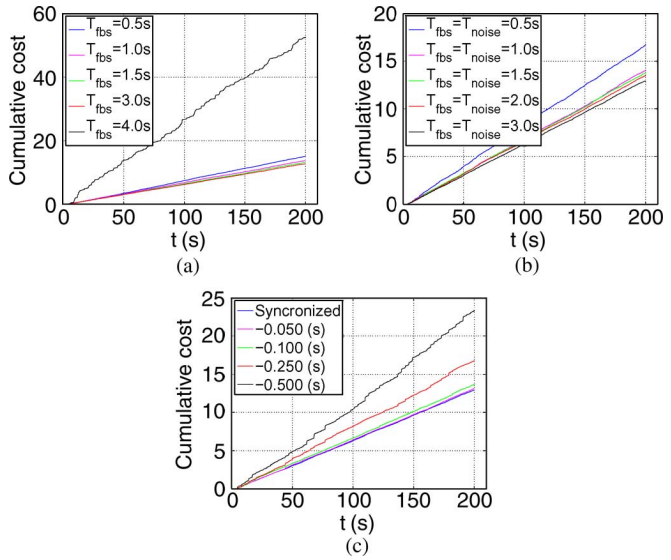

Fig. 7. Cumulative cost of the optimal sampling period assignment approach for different patterns of the feedback scheduler periodicity and noise periodicity. (a) Different FBS periods. (b) Different FBS/noise. (c) Unsynchronized noise/FBS.

the previous analysis by showing the CPU time consumed by the feedback scheduler for each H choice. It can be seen that as H decreases, the overhead increases exponentially. It has to be pointed out that better search algorithms in the optimization procedure of the feedback scheduler than the linear search that we have applied could result in lower computational overhead.

D. Performance Evaluation: FBS and Noise Periods

Fig. 7(a) shows the behavior of the online period assignment approach with respect to different values of the period of the feedback scheduler while keeping the same noise intensity rate of change to $h_{\text{noise}} = 3000$ ms. As it can be seen in the figure, as long as T_{fbs} is smaller than h_{noise}, the presented approach delivers good numbers. However, when this relation does not hold, as for the case of $T_{\text{fbs}} = 4$ s, then, the performance drastically decreases. Hence, the presented approach requires having $T_{\text{fbs}} \leq h_{\text{noise}}$ in order to provide good control performance results.

Fig. 7(b) complements the previous figure by showing the control performance of the presented approach for different T_{fbs} values while keeping the relation $h_{\text{noise}} = T_{\text{fbs}}$. As it can be seen, the performance is satisfactory except when the period is small, due to the overhead introduced by many executions of the feedback scheduler.

Fig. 7(c) shows the control performance of the presented approach for $T_{\text{fbs}} = 1000$ ms while not keeping the relation $h_{\text{noise}} = T_{\text{fbs}}$. In this scenario, the synchronization is lost for different amounts of time. As it can be seen, as the synchronization decreases, the cost increases. Hence, a noise estimator is required.

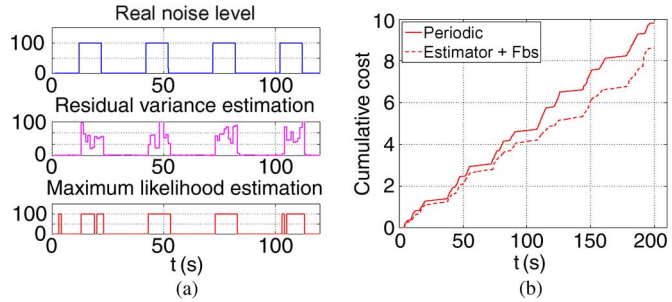

Fig. 8. Adding noise estimation in the FBS. (a) Estimators performance. (b) FBS with estimator.

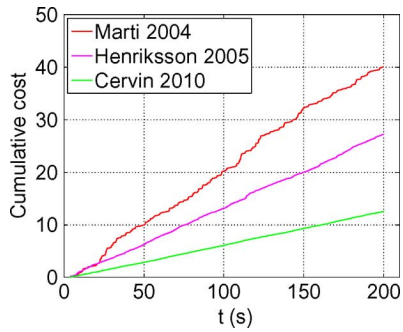

Fig. 9. Cumulative cost of the optimal sampling period assignment approach with respect to previous work.

E. Performance Evaluation: Noise Estimator

This section analyzes the performance of the presented approach with a more realistic setting where noise intensities are estimated for each plant and their rate of change is randomly generated. In particular, we forced $h_{\text{noise}} \gg T_{\text{fbs}}$.

To evaluate the estimators introduced in Section IV-E, Fig. 8(a) shows the noise intensities estimates $\hat{r_1}$ for one of the plants when the intensity of the noise changes between certain levels. Although both estimators are able to detect the changes in the noise intensity, the MLE provides better estimates because the noise levels are *a priori* known. However, if the noise changes are not that abrupt and not known *a priori*, the RVE performs better (see further details in [12]).

In Fig. 8(b), we show the performance of optimal sampling period assignment approach when the noise intensity estimation was performed by the MLE executed by the feedback scheduler task. In terms of overhead, the MLE only adds 0.015 ms to each feedback scheduler execution and requires storing two thresholds for computing the correct estimates, one for each noise level. As it can be seen in the figure, the performance improvement with respect the static case is still significant (around 15%). It must be noted that all the previous case studies also apply to

this "real" scenario with the appropriated scaling. However, in order to be able to characterize which benefit is a consequence of a particular parameter, the ideal estimator was more suited for the detailed performance analysis.

F. Performance Evaluation: Policies Comparison

The proposed approach is compared to two previous relevant approaches, Martí *et al.* [5] and Henriksson *et al.* [7]. Using the same control settings for all the approaches as explained in the base experiment, the performance numbers are shown in Fig. 9. As expected, Martí *et al.* [5] provide the lowest performance because only the current state is accounted for in the optimization and the noisy states highly jeopardize the operation of the approach. Then, Henriksson *et al.* [7] provides a medium performance curve because varying noise is not accounted for, and then the noise states predominate in the finite horizon optimization solution. Finally, the presented approach, labeled "Cervin 2010," provides the best performance curve.

VI. Conclusion

In this brief, we have demonstrated how feedback from the control applications can be used to online optimize the performance. The feedback scheduler evaluates the expected cost over a finite time horizon, taking into account the noise intensity, the sampling interval, and the computational delay for each control loop. Based on this information, optimal sampling intervals are assigned to the control tasks. Extensive experiments on a set of real plants have shown that the real-time implementation of the feedback scheduling approach improve the control performance quite substantially compared to a state-of-the-art periodic implementation.

References

[1] K. J. Åström and B. Wittenmark, *Computer-Controlled Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[2] D. Seto, J. P. Lehoczky, L. Sha, and K. G. Shin, "On task schedulability in real-time control systems," in *Proc. 17th IEEE Real-Time Syst. Symp.*, 1996, pp. 13–21.

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

[4] A. Cervin, J. Eker, B. Bernhardsson, and K.-E. Årzén, "Feedback-feedforward scheduling of control tasks," *Real-Time Systems*, vol. 23, no. 1–2, pp. 25–53, Jul. 2002.

[5] P. Martí, C. Lin, S. A. Brandt, M. Velasco, and J. M. Fuertes, "Optimal state feedback based resource allocation for resource-constrained control tasks," in *Proc. 23rd IEEE Real-Time Syst. Symp.*, 2004, pp. 161–172.

[6] P. Martí, C. Lin, S. A. Brandt, M. Velasco, and J. M. Fuertes, "Draco: Efficient resource management for resource-constrained control tasks," *IEEE Trans. Comput.*, vol. 58, no. 1, pp. 90–105, Jan. 2009.

[7] D. Henriksson and A. Cervin, "Optimal on-line sampling period assignment for real-time control tasks based on plant state information," presented at the 44th IEEE Conf. Decision Control Eur. Control Conf. (ECC), Seville, Spain, Dec. 2005.

[8] R. Castañé, P. Martí, M. Velasco, A. Cervin, and D. Henriksson, "Resource management for control tasks based on the transient dynamics of closed-loop systems," presented at the 18th Euromicro Conf. Real-Time Syst., Dresden, Germany, Jul. 2006.

[9] L. Palopoli, C. Pinello, A. Bicchi, and A. Sangiovanni-Vincentelli, "Maximizing the stability radius of a set of systems under real-time scheduling constraints," *IEEE Trans. Autom. Control*, vol. 50, no. 11, pp. 1790–1795, Nov. 2005.

[10] M.-M. B. Gaid, A. Çela, Y. Hamam, and C. Ionete, "Optimal scheduling of control tasks with state feedback resource allocation," in *Proc. Amer. Control Conf*, Jun. 2006, pp. 310–315.

[11] M.-M. B. Gaid, A. Çela, and Y. Hamam, "Optimal real-time scheduling of control tasks with state feedback resource allocation," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 2, pp. 309–326, Mar. 2009.

[12] A. Cervin, M. Velasco, P. Martí, and A. Camacho, "Optimal on-line sampling period assignment," Dept. Autom. Control, Tech. Univ. Catalonia, Barcelona, Spain, Tech. Rep. ESAII-RR-09-04, Dec. 2009.

[13] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *J. ACM*, vol. 20, no. 1, pp. 40–61, 1973.

[14] J. A. Stankovic, M. Spuri, K. Ramamritham, and G. C. Buttazzo, *Deadline Scheduling for Real-Time Systems—EDF and Related Algorithms*. Norwell, MA: Kluwer, 1998.

[15] K. J. Åström, *Introduction to Stochastic Control Theory*. New York: Academic Press, 1970.

[16] D. Liberzon, *Switching in Systems and Control*. Boston, MA: Birkhäuser, 2003.

[17] Erika Enterprise, "Evidence Srl," 2008. [Online]. Available: http://www.evidence.eu.com/content/view/27/254/

[18] Evidence Srl, "FLEX Full Base Board," 2008. [Online]. Available: http://www.evidence.eu.com/content/view/154/207/