Multirate Controller Design for Resource- and Schedule-Constrained Automotive ECUs

Dip Goswami[1], Alejandro Masrur[1], Reinhard Schneider[1], Chun Jason Xue[2] and Samarjit Chakraborty[1]
[1]Institute for Real-Time Computer Systems, TU Munich, Germany
[2]City University of Hong Kong, Hong Kong
(dip.goswami@tum.de, masrur@rcs.ei.tum.de, reinhard.schneider@rcs.ei.tum.de, jasonxue@cityu.edu.hk and samarjit@tum.de)

***Abstract*—Automotive software mostly consists of a set of applications controlling the vehicle dynamics, engine and many other processes or plants. Since automotive systems design is highly cost driven, an important goal is to maximize the number of control applications to be packed onto a single processor or electronic control unit (ECU). Current design methods start with a controller design step, where the sampling period and controller gain values are decided based on given control performance objectives. However, operating systems (OS) on the ECU (e.g., ERCOSek) are usually pre-configured and offer only a limited set of sampling periods. Hence, a controller is implemented using an available sampling period, which is the shorter period closest to the one determined in the controller design step. However, this increases the load on the ECU (i.e., the processor runs the controller more often than what is actually required by design). This reduces the number of applications that can be mapped, and increases costs of the system. To overcome this predicament, we propose a multirate controller, which switches between multiple available sampling periods offered by the OS on the ECU. Apart from meeting all control objectives, this avoids the unnecessary ECU overload resulting from always sampling at a constant, higher rate.**

I. Introduction

In many cost-sensitive domains, the goal is to pack the maximum possible number of tasks on a processor – e.g., an electronic control unit (ECU) in the automotive domain – while ensuring application specific performance constraints. Towards this, we study a setup where feedback control applications share the same ECU with other tasks. The ECU runs a time-triggered real-time operating system such as ERCOSek [1] which is widely used in the automotive industry. The tasks running on ERCOSek are periodic, and in general only a finite number of task periods are realizable on such platforms. Often, an optimal task period (derived from application-specific constraints) is not directly realizable on a platform such as ERCOSek which configures only a finite number of pre-defined task periods. In such scenarios, a common practice in industry is to assign a shorter task period which is available on the given platform and the closest to the optimal one (i.e., the one determined by design). Evidently, this leads to an inefficient resource-usage in the most cases. In this paper, we address the above aspect in the context of feedback control applications.

The control performance improves with higher sampling rate (or lower task periods). Further, a higher sampling rate results in a higher processor load. Essentially, we have two design requirements:

(i) Control performance constraints,
(ii) Processor load constraints.

In many design cases, a higher sampling rate satisfies (i) but violates (ii). On the other hand, a lower sampling rate satisfies (ii) but not (i). So how to satisfy both? The goal is to design and implement a control application with the optimal (i.e., the longest possible) sampling period which is enough to meet its performance constraints.

978-3-9815370-0-0/DATE13/©2013 EDAA

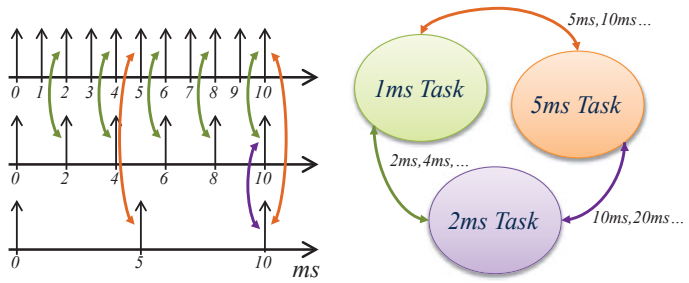

Fig. 1. Allowed switching points between two sampling periods

Our contributions: Towards realizing an optimal sampling period which is not directly supported by the OS configuration (e.g., ERCOSek), we propose a multirate sampling scheme where the sampling period switches among the realizable or available task periods. Fig. 1 shows a typical task model with periods of $1ms$, $2ms$ and $5ms$ on ERCOSek. In this platform, a control application can only run with one or more combinations of these three sampling periods. To incorporate a controller with sampling periods switching between the $1ms$, $2ms$ and $5ms$ tasks, there exists an additional constraint derived from this platform. For example, a controller can switch between $1ms$, $2ms$ or $5ms$ sampling period at times $10ms$, $20ms$, etc. (as illustrated in Fig. 1). Similarly, a controller is allowed to switch between a 1ms and a 5ms sampling period at time $5ms$, $10ms$, $15ms$ etc. That is, it is possible to switch among sampling periods whenever corresponding tasks are released together on ERCOSek. As a result in the example of Fig. 1, only certain sequences of sampling periods are possible such as $\{2ms, 2ms, 2ms, 2ms, 2ms, 5ms, 5ms...\}$, $\{5ms, 5ms, 10ms...\}$ and so on. By using such a sequence of sampling periods, we achieve an "average" sampling period for the control application close to the optimal one. Overall, we can meet the performance requirements of the control applications utilizing "average" sampling period. This way, the processor utilization is hence reduced since the average period of the corresponding control application increases. In this context, we define a *schedule* as a possible order of sampling periods (which are supported by the platform) such that the resulting average sampling period becomes close to the optimal one. The main question that we address in this paper is: For a given schedule, how to analyze and synthesize the controllers which can ensure both the stability and the performance under such multirate scheme? We show that by *deterministic* switch-

ing between two or more available sampling periods and using *appropriately* designed controllers, both requirements (i) and (ii) can be satisfied. This paper introduces a proof of concept on how such *deterministic* switching between sampling periods can be exploited for design improvement. To the best of our knowledge, this work considers the constraints coming from OS configurations in the control design for the first time. We illustrate our scheme taking the control of electro-mechanical braking (EMB) system as a representative example.

A. Related Work

An appropriate choice of a sampling period for feedback control loops in the presence of resource constraints has been studied both in networked architectures [2] and single-processor implementation platforms [3], [4]. In addition, there exists a large body of work in the control theory literature concerning *multirate* sampling schemes [5]. These works consider the scenario where sensing devices have different sampling rates from actuating devices, and the focus is to ensure stability of the overall system under such a scheme.

While these previous works have made significant contributions to this area, the presented technique is motivated by the following two observations: (i) In many real-life settings (e.g., automotive, avionics), the sampling periods (or sampling instants) cannot be chosen arbitrarily. The set of permissible sampling periods is constrained by operating systems running on the particular platform (e.g., ERCOSek) and other *legacy* tasks. (ii) In cost-driven industries such as automotive, the implementation of a control algorithm should be resource-efficient to accommodate as many applications or tasks on as few ECUs or processors as possible. As a result, the sampling period plays an important role in the resource-aware implementation of a feedback loop.

The rest of this paper is organized as follows. Section II introduces the control applications, the EMB system and the platform under consideration. This section further motivates from various design perspectives. In Section III, we introduce the proposed multirate sampling scheme. Section IV presents a set of experimental results conducted upon the electro-mechanical braking system while the conclusions are discussed in Section V.

II. System Architecture

In this section, we describe the setup used throughout the paper. We consider an ECU that runs multiple control applications (tasks) along with other real-time tasks. In the following, we discuss various aspects of the setup.

A. Control Applications

Each control application is responsible for controlling a plant or dynamic system. In particular, we consider linear control applications where the dynamic behavior is modeled by a set of differential equations,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{1}$$

where $x(t) \in R^n$ is the *state*, $y(t)$ is the *output* and $u(t)$ is the *control input* to the system. A is *system matrix* and B is input matrix. The state $x(t)$ is sampled at discrete time instances,

$$0 = t_0 < t_1 < \cdots < t_k < \cdots \tag{2}$$

TABLE I
EMB System Requirements

C1 – Settling time $\tau_s(ms)$	250
C2 – Input Saturation $U_{max}(volt)$	12
Reference $r(meter)$	0.002
Execution Time $e_i(ms)$	0.2

and the control input is piecewise constant between the discrete time instances:

$$u(t) = Kx(t_k) + F(T_k)r, \forall \in [t_k, t_{k+1}), \tag{3}$$

where K is the feedback gain, r is the reference and $F(T_k)$ is the static feedforward gain which depends on the sampling period T_k, $T_k = t_{k+1} - t_k$. Combining systems (1) and (3), the resulting closed-loop system is a discrete-time system [6]:

$$\begin{aligned} x(t_{k+1}) &= G(T_k)x(t_k) + BF(T_k)r \\ y(t_k) &= Cx(t_k) \end{aligned} \tag{4}$$

where

$$G(T_k) = e^{AT_k} + \int_0^{T_k} e^{At} BdtK, \tag{5}$$

$$F(T_k) = 1/C(I - G(T_k))^{-1} \int_0^{T_k} e^{At} Bdt. \tag{6}$$

Clearly, the control law (3) has two components: (i) a feedback component which depends on the state-feedback gain K and (ii) a feedforward component which depends on the gain $F(T_k)$. The stability of the closed-loop system (4) depends only on the choice of K (or resulting system matrix $G(T_k)$) while the reference r is tracked by the feedforward part.

B. Control Requirements and Constraints

In general, the stability of the closed-loop system (4) can be ensured by placing its poles inside the unit circle, i.e., with absolute value less than unity. Towards this, *pole-placement* is performed by an appropriate choice of the controller gain K which places the poles of the closed-loop system (4) at desired locations. In addition, in our setting, we need to ensure that the closed-loop system meets the requirements C1 and C2:

- C1 **Settling time τ_s:** This is the maximum time that the output $y(t_k)$ of the control loop takes to reach a close proximity of the reference r. That is, the maximum time to achieve $y(t_k) \approx r$. Each control application has a given settling time requirement τ_s. As a consequence, $u(t)$ should be designed in such a way that the settling time requirement is met.
- C2 **Input Saturation U_{max}:** For every control application, there is a maximum input value (actuator saturation) that can be realized and hence $|u(t)| \leq U_{max}$.

The performance of a closed-loop control application highly depends on its poles and the sampling period used. For a discrete-time system (4), the control action becomes aggressive (i.e., more reactive or responsive) with the system poles closer to *zero* which results in a shorter settling time τ_s. On the other hand, the maximum input signal requirement U_{max} becomes higher when the system poles are placed closer to zero (i.e., with an aggressive controller). As a consequence, the control design mainly hinges on meeting the two conflicting requirements C1 and C2.

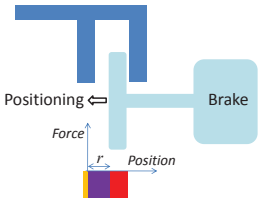

Fig. 2. Electro-Mechanical Braking (EMB) System model

C. Electro-Mechanical Braking (EMB) System

To illustrate various design aspects from the application side, we consider electro-mechanical braking (EMB) system for automobiles as a representative example. The basic braking system is shown in Fig. 2. When the braking system is active and the braking paddle (labeled 'brake' in Fig. 2) is pressed, the position of the braking lever should reach a reference position r within the desired settling time τ_s. This is called *position* mode. Once the braking lever reaches the reference position r, the next task is to achieve a necessary force to stop the vehicle. This is called *force* mode. For ease of exposition, in this work, we consider only the position mode. However, it is possible to trivially extend our approach to the entire braking system as well.

The application in position mode can be represented by a linear system as shown in (1) (model details are omitted due to space limitation). Based on the above model, the requirements for the EMB system is shown in Table I.

D. Platform Requirements and Constraints

As already explained, ERCOSek puts certain constraints on the task periods: (i) On ERCOSek, only a finite number of sampling periods is realizable. The set of available periods is denoted by ϕ and is given by $\{1ms,\ 2ms,\ 5ms,\ 10ms,\ 50ms,\ 100ms,\ 200ms,\ 500ms,\ 1sec\}$. As a consequence, the control applications have to operate with one or more combinations of sampling periods $T_k \in \phi$. (ii) On ERCOSek, the controllers can be switched between two or more different available sampling periods only following a special pattern as illustrated Fig. 1.

Apart from the above two requirements, another design aspect is the *utilization* produced by a particular controller. If the worst-case execution time (WCET) of a control task C_i is e_i and the sampling period is T_k, its utilization (i.e., a measure of its processor demand) is given by the ratio $\frac{e_i}{T_k}$. The overall utilization on the processor or ECU is hence the sum of the individual utilizations of controllers. In general, a system design that results in less processor utilization allows a better, i.e., more cost-efficient, design. This means, for each controller in the system, it is necessary to minimize the following,

$$L = \frac{e_i}{T_k}. \tag{7}$$

Clearly, to minimize L, it is necessary to apply sampling period T_k as high as possible.

E. Design Motivation

To motivate the need for a multirate sampling, we explore various choices of controller designs and their impact on the system performance. The results for the EMB system are shown in Table II.

Case I: In this case, we choose a sampling period of $T_k = 5ms$. The closed-loop systems poles are chosen as close as possible to zero without violating the input saturation constraint C2. Towards meeting the input saturation constraint, the controller becomes less aggressive. The resulting system has a settling time of $400ms$ which violates C1 as shown in Table I. Therefore, we explore a more aggressive controller keeping the same sampling period $T_k = 5ms$ as illustrated in the following.

Case II: We choose a sampling period of $T_k = 5ms$ with closed-loop system poles closer to zero compared to the poles in Case I. Towards this, the systems poles are chosen such that the settling time meets the requirement C1. In this case, the maximum input signal requirement becomes much higher than requirement C2 in Table I.

From the Cases I and II, we see that both C1 and C2 cannot be simultaneously met with controller (3) and $T_k = 5ms$[1]. Hence, we explore the possibility of utilizing shorter sampling period in the following.

Case III: We choose a sampling period of $T_k = 2ms$. We can see that both C1 and C2 are satisfied in this case. However, this improvement in control performance came at the cost of higher utilization, i.e., higher $\frac{e}{T_k}$. In order to pack as many tasks as possible onto the same ECU and hence, reduce costs, it is absolutely necessary to minimize the utilization L in (7) which is one of the main motivations of our proposed design method. To this end, we illustrate the design for meeting the requirements C1-2 with lesser utilization first without considering the constraints coming from ERCOSek-based platform.

Case IV: We explore another possibility with a sampling period higher than the one in Case III. That is, we choose a sampling period $T_k = 3.5ms$. In this case, Table II shows that we meet both C1-2 with a lower utilization compared to Case III. However, a sampling period of $3.5ms$ is not realizable in our setting because of the constraints on schedules and sampling periods.

In view of the above design examples, it is a common practice in automotive industry to follow the design option presented in Case III. However, since the control application can be run at a lower sampling rate than Case III (as shown in Case IV), such design leads to an inefficient utilization. The main motivation of our proposed multirate scheme is to replace the design option in Case III and to incorporate Case IV as closely as possible. Let us now consider the next design option.

Case V: Instead of keeping constant sampling period T_k, we propose to switch between $2ms$ and $5ms$ ($T_k \in \phi$) in every alternate sample. That is, we apply (3) at $t_k = 0ms, 2ms, 7ms, 9ms, 14ms \cdots$ and so on. As shown in Table II, we can meet this way the control performance requirements C1 and C2. At the same time, the utilization is lesser compared to Case III and same as that of Case IV. By

[1]It is possible to explore the possibility of using a more advanced control algorithm such as Model Predictive Control (MPC). However, such control algorithms will naturally come at the cost of higher computational resource which will open up another front from the design perspective. For this reason, in this paper, we confine our study to a single control algorithm.

TABLE II
CONTROL PERFORMANCE FOR EMB SYSTEM

Cases	Closed-loop poles	$T_k(ms)$	$\tau_s(ms)$	$\lvert u(t_k)\rvert(volt)$ max	J	$\frac{e_i}{T_k}$	Remark
I	$[0, 0, 0.9, 0.9, 0.9]$	5	400	10.5	7.7125×10^{-5}	0.04	C1 violated
II	$[0, 0, 0.3, 0.9, 0.9]$	5	250	73	2.1468×10^{-5}	0.04	C2 violated
III	$[0, 0, 0, 0.9, 0.9]$	2	150	10.7	5.4334×10^{-5}	0.1	Utilization is higher
IV	$[0, 0, 0, 0.8, 0.9]$	3.5	250	9.8	6.7094×10^{-5}	0.0571	Not realizable
V	-	$\{2ms, 5ms, 2ms, 5ms, ...\}$	200	11.2	5.7778×10^{-5}	0.0571	-

switching within the available sampling periods $T_k \in \phi$, it is also possible to achieve a design that is close to Case IV. Motivated by this observation, we present the proposed scheme in the following sections.

III. MULTIRATE SAMPLING SCHEME

Overall, the design problem consists in designing control applications such that their high-level requirements (i.e., C1 and C2) met and the utilization L of individual controllers is as less as possible. Towards this, we first define a schedule S as a sequence of sampling periods (resulting from the previously described period switching) of a control application. For example, $S = \{T_{k,1}, T_{k,2}, T_{k,3}\}$ with $T_{k,i} \in \phi$ implies sampling periods is as follows

$$T_{k,1} \rightarrow T_{k,2} \rightarrow T_{k,3} \rightarrow T_{k,1} \rightarrow T_{k,2} \rightarrow T_{k,3} \rightarrow repeats.$$

For a given schedule S, the resulting closed-loop system becomes

$$G(T) = \prod G(T_{ki}), T_{ki} \in S. \tag{8}$$

The goal is to design the controller gain K in (3) such that $G(T)$ in (8) is stable and the overall closed-loop system meets C1-2. We address this problem in the following.

A. Controller synthesis

In this section, we describe how to design the controller gain K in (3) such that the resulting system (8) with a given schedule S is stable. For a given schedule, the design steps are outlined in the following:

Step 1: Choose T_{avg} such that: $0 < T_{min} \leq T_{avg} \leq T_{max}$ where

$$T_{min} = min\{T_{k,i}\}, \forall T_{k,i} \in S,$$

$$T_{max} = max\{T_{k,i}\}, \forall T_{k,i} \in S.$$

Step 2: Compute the discrete-time system with a uniform sampling period T_{avg}, $A_d = e^{AT_{avg}}, B_d = \int_0^{T_{avg}} e^{At} B dt$.

Step 3: Choose closed-loop poles P of the new system given by (A_d, B_d) and compute the gain K by pole assignment technique. We have $G(T_{avg}) = A_d + B_d K$.

Step 4: Check the overall stability by computing the poles of system (8). If $G(T)$ is unstable, repeat the steps with another choice of T_{avg} and poles P.

Clearly, we have two design parameters: (i) uniform sampling period T_{avg} and (ii) closed-loops poles P of $G(T_{avg})$. Depending on the choice of these parameters, the stability and the performance of the system varies. The control performance improves with (i) lower T_{avg} which is closer to T_{min} and, (ii) the closed-loop poles of $G(T_{avg})$ closer to zero (i.e., an aggressive controller). On the other hand, the performance degrades (i) as T_{avg} goes closer to T_{max} and, (ii) when the closed-loop poles of $G(T_{avg})$ moves away from zero (i.e., with a less aggressive controller).

IV. EXPERIMENTAL RESULTS

In this section, we revisit the EMB system. We show how to design the controller for a given set of allowable schedules and how the control requirements C1-2 are met with lesser processor utilization under our scheme. We implemented a EMB system and the controllers in MATLAB/Similnk. The multirate scheme is realized using TrueTime 1.5 [7] with a single-processor where multiple periodic control tasks with fixed priorities are running. We choose a schedule

$$S = \{2ms, 2ms, 2ms, 2ms, 2ms, 5ms, 5ms\}.$$

The controller gain K is designed by placing the closed-loop system poles at $[0, 0, 0, 0.8, 0.85]$ with a uniform sampling period $T_{avg} = 3.6ms$. The resulting system has a settling time $190ms$, $|u(t_k)|max = 11.9(volt)$ and utilization $\frac{e_i}{T_k} = 0.07$. Clearly, both C1-2 are met with lower utilization compared to Case III.

V. CONCLUDING REMARKS

In this paper, we presented a multirate sampling scheme for feedback control applications targeted for platforms which support only limited number of task periods and have a limited computational power. From a resource-utilization perspective, a control application should be implemented with the longest possible sample period that can assure stability and performance. However, the underlying implementation platform often cannot realize such sampling periods. Hence, the proposed scheme switches among the realizable or available sampling periods to achieve a period that is optimal from both control and resource-utilization perspectives. Towards this, the main challenge is to design the sampling schedules and the controller to meet certain platform and control constraints.

VI. ACKNOWLEDGEMENT

The authors would like to thank Dr. Arne Hamann, Dr. Matthias Bitzer and Dr. Eckart Mayer-John from Robert Bosch Corporate Research GmbH for the valuable discussions and help with the case-study.

REFERENCES

[1] "OSEK/VDX Specifications," www.osek-vdx.org.
[2] E. Bini and A. Cervin, "Delay-aware period assignment in control systems," in *IEEE RTSS*, 2008.
[3] D. Henriksson and A. Cervin, "Optimal on-line sampling period assignment for real-time control tasks based on plant state information," in *IEEE CDC*, 2005.
[4] F. Zhang, K. Szwaykowska, W. Wolf, and V. J. Mooney, "Task scheduling for control oriented requirements for Cyber-Physical Systems," in *IEEE RTSS*, 2008.
[5] M. Mizuochi, T. Tsuji, and K. Ohnishi, "Multirate sampling method for acceleration control system," *IEEE Transaction on Industrial Electronics*, vol. 54, pp. 1462–1472, 2007.
[6] Y. S. Suh, "Stability and stabilization of nonuniform sampling systems," *Automatica*, vol. 44, no. 12, pp. 3222–3226, 2008.
[7] "TrueTime 1.5," www.control.lth.se/truetime/.