

Control-System Stability under Consecutive Deadline Misses Constraints

Martina Maggio 

Saarland University, Department of Computer Science, Germany

Lund University, Department of Automatic Control, Sweden

Robert Bosch GmbH, Germany

maggio@cs.uni-saarland.de

Arne Hamann

Robert Bosch GmbH, Germany

arne.hamann@de.bosch.com

Eckart Mayer-John

Robert Bosch GmbH, Germany

eckart.mayer@de.bosch.com

Dirk Ziegenbein

Robert Bosch GmbH, Germany

dirk.ziegenbein@de.bosch.com

Abstract

This paper deals with the real-time implementation of feedback controllers. In particular, it provides an analysis of the stability property of closed-loop systems that include a controller that can sporadically miss deadlines. In this context, the weakly hard m-K computational model has been widely adopted and researchers used it to design and verify controllers that are robust to deadline misses. Rather than using the m-K model, we focus on another weakly-hard model, the number of consecutive deadline misses, showing a neat mathematical connection between real-time systems and control theory. We formalise this connection using the joint spectral radius and we discuss how to prove stability guarantees on the combination of a controller (that is unaware of deadline misses) and its system-level implementation. We apply the proposed verification procedure to a synthetic example and to an industrial case study.

2012 ACM Subject Classification Computer systems organization → Real-time systems; Computer systems organization → Embedded and cyber-physical systems; Mathematics of computing → Mathematical analysis; Computer systems organization → Dependable and fault-tolerant systems and networks

Keywords and phrases Real-Time Control, Deadline Misses, Weakly Hard Models

Digital Object Identifier 10.4230/LIPIcs.ECRTS.2020.1

Funding This work was supported by: the ELLIIT Strategic Research Area, the project *ARAMiS II* of the German Federal Ministry for Education and Research with the funding ID 01IS16025. The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871259 (ADMORPH). The responsibility for the content remains with the authors.

Acknowledgements This research was developed while Martina Maggio was on sabbatical at Robert Bosch GmbH.

1 Introduction

The contribution of this paper is a verification procedure to prove the robustness of controller implementations to deadline misses. For this task, literature contributions focus on the computation model where the control task can *miss* at most m deadlines in a window of K



© Martina Maggio, Arne Hamann, Eckart Mayer-John and Dirk Ziegenbein; licensed under Creative Commons License CC-BY

32nd Euromicro Conference on Real-Time Systems (ECRTS 2020).

Editor: Marcus Völöp; Article No. 1; pp. 1:1–1:23



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

45 activations (i.e., the m - K model). This was one of four proposed models [3] to analyse systems
 46 with deadline misses. In this paper we show that there is a natural analytical connection
 47 between another of these four models, the number of maximum consecutive deadline misses,
 48 and control-theoretical tools that can be used to prove properties of closed-loop systems,
 49 such as stability.

50 **Historical Perspective:** During the past couple of decades the real-time systems community
 51 made an effort in formalising requirements, models, and algorithms to handle systems that
 52 can (sporadically) miss deadlines. In this quest, Hamdaoui and Ramanathan analysed tasks
 53 that behave according to the (m, k) model [18]. With this model, tasks can be modelled
 54 as sequences of jobs. In every set of k consecutive jobs, at least m jobs must meet their
 55 deadline. This model was analysed, finding schedulability conditions and scheduling schemes,
 56 e.g., [35]. Building on the ideas from this research, the *weakly hard* model of computation
 57 was formalised [3]. A weakly hard real-time system is a system in which the distribution of
 58 deadline misses and hits during a window of time is precisely bounded.

59 Bernat, Burns and Liamosí [3] give four possible definitions for a weakly hard task τ .

- 60 1. $\tau \vdash \binom{n}{m}$, with $1 \leq n < m$: According to this definition, for each set of m consecutive
 61 deadlines, τ meets at least n of them. With a slight difference in notation, this model
 62 corresponds to the (m, k) model by Hamdaoui and Ramanathan [18].
- 63 2. $\tau \vdash \langle \frac{n}{m} \rangle$, with $1 \leq n < m$: Here, the system guarantees that for each set of m consecutive
 64 deadlines, τ meets consecutively n of them.
- 65 3. $\tau \vdash \overline{\binom{n}{m}}$, with $1 \leq n < m$: This is the dual definition with respect to the first one. In
 66 this case, the system guarantees that for each set of m consecutive deadlines, τ misses at
 67 most n of them.
- 68 4. $\tau \vdash \overline{\langle n \rangle}$, $n \geq 1$: According to this definition the maximum number of consecutive deadline
 69 misses that τ can experience is n .¹

70 The third of these models gained traction in the research community, and the term weakly-
 71 hard task started to indicate a task that can experience a bounded number of misses in a
 72 window of jobs. In particular, with a slightly confusing terminology, this third model is also
 73 often called the m - K model.² This specifies that a task can experience at most m misses in
 74 a window of K consecutive jobs [1, 9, 10, 12, 14, 19, 20, 33, 34, 41–43].

75 **The m - K Model for Control Tasks:** In the attempt to achieve computing and control co-
 76 design, both schedulability results like [43] and analysis using model checking [12] have been
 77 investigated. The research community contributed with criteria to determine stability [5],
 78 determine convergence rates [14], and to design controllers in the presence of deadline
 79 misses [29]. Furthermore, the performance cost of deadline misses was investigated [34, 44],
 80 together with the role of the strategy used to handle the misses [33, 41], i.e., killing the task
 81 or allowing its continuation with different policies for the following iteration. The general
 82 consensus is that sequences of misses and hits (i.e., the m - K model) are to be considered
 83 and analysed to determine physical properties of the system.

84 **Contribution:** In this paper we use the $\tau \vdash \overline{\langle n \rangle}$ model and cast the problem of verifying the
 85 stability of closed-loop systems into a neat mathematical framework. The solution ensures
 86 the stability of the combination of: (1) the physics of the plant under control and, (2) the

¹ The original definition was $\tau \vdash \langle \frac{n}{m} \rangle$, with $1 \leq n < m$. However, according to [3, Theorem 4] there is no
 need to specify the window size, i.e., the task τ can be equivalently defined using any window, hence we
 use $\tau \vdash \overline{\langle n \rangle}$.

² Notice the difference between the (m, k) model (that stated that there were m hits for every sequence
 of k consecutive jobs, with $1 \leq m < k$) and the m - K model (that imposes that there are at most m
 misses in every window of K jobs, with $1 \leq m < K$).

87 execution of the controller (that may miss deadlines). We believe that the $\tau \vdash \overline{\langle n \rangle}$ model is
 88 as relevant as the m - K model from the industrial standpoint.

89 In fact, in industrial products, deadline misses are often caused by transient overload
 90 periods or faults. In many industrial applications, a system load of over 80% is targeted
 91 for cost reasons. Such a high system load can usually not be achieved with purely formal
 92 methods that are based on worst-case considerations, especially on multi-core platforms.
 93 For this reason, many industrial systems are designed for average runtimes plus a safety
 94 margin, in conjunction with rate monotonic scheduling. Our experience is that, following
 95 this practical approach in contrast to analytical ones, deadline violations may occur, e.g.,
 96 due to a transient interrupt load. During these transient periods the miss ratio often is
 97 quite high, making the number of consecutive misses a very relevant indicator of the system
 98 performance. We argue that methods that evaluate and prove the robustness of controllers
 99 to deadline violations in this setup are of high industrial interest.

100 **Outline:** In the remainder of this paper we will recap the necessary control background
 101 and then provide our contribution. In particular, Section 2 explains how a plant is modelled
 102 and how a *state feedback controller* is applied to regulate the plant's behaviour. Section 3
 103 describes the strategies that are typically used to handle deadline misses and provides some
 104 insights on what is the best choice from the system perspective. Section 4 shows how to
 105 guarantee properties of closed-loop systems (like stability) in the presence of deadline misses
 106 with the $\tau \vdash \overline{\langle n \rangle}$ model. Section 5 shows some experimental results validating our claims.
 107 Finally, Section 6 presents an overview of related work and Section 7 concludes the paper.

108 2 Control Background

109 In this section we recap the basic concepts of control theory that are used in the rest of
 110 the paper. We analyse linear time-invariant models and controllers implemented as periodic
 111 tasks with implicit deadlines.

112 **Plant Model:** The starting point for control design is always understanding the object that
 113 the controller should act upon. The control engineer obtains a model \mathcal{P}_c of the plant to
 114 control. In most cases, this model is linear and time-invariant, and represents with ordinary
 115 differential equations the dynamics of the system in the following form.

$$116 \quad \mathcal{P}_c : \begin{cases} \dot{x}(t) = A_c x(t) + B_c u(t) \\ y(t) = C_c x(t) + D_c u(t) \end{cases} \quad (1)$$

117 Here, the system state $x(t) = [x_1(t), \dots, x_p(t)]^T$ evolves depending on the current state and
 118 the input signal $u(t) = [u_1(t), \dots, u_r(t)]^T$, where the superscript T indicates the transposition
 119 operator. We denote with p the number of state variables (i.e., the length of vector x) and
 120 with r the number of input variables (i.e., the length of vector u). The matrices A_c , B_c , C_c ,
 121 and D_c encode the dynamics of the system. In the following, we will make two assumptions:
 122 D_c is a zero matrix of appropriate size, and C_c is the unit matrix of appropriate size.³ The
 123 first assumption means that the system is *strictly proper* and holds for almost all the physical
 124 models used in control. The second assumption means that the state is measurable. This
 125 does not always hold for real systems, but state observers can be built whenever this is not
 126 true [26], to estimate the state $x(t)$.⁴ This means that, without losing generality, we can

³ A_c is a $p \times p$ matrix, B_c is a $p \times r$ matrix, C_c is a $p \times p$ matrix, and D_c is a $p \times r$ matrix.

⁴ As a remark, if a state observer is present its dynamics should be taken into account in the analysis. This extension only requires to augment the system state with the rows and columns corresponding to the execution of the observer, but the analysis method remains the same.

127 represent \mathcal{P}_c as

$$128 \quad \dot{x}(t) = A_c x(t) + B_c u(t), \quad (2)$$

129 and describe the system dynamics using only A_c and B_c .

130 From this starting point, control systems are usually designed and realised in one of these
131 two ways:

- 132 1. The plant model \mathcal{P}_c is used to synthesise a controller (model) \mathcal{C}_c in continuous-time.
133 Closed-loop system properties, like stability, are proven on the feedback interconnection
134 of \mathcal{C}_c and \mathcal{P}_c . However, when the controller is implemented, digital hardware is used.
135 This means that the controller model \mathcal{C}_c has to be discretised, obtaining \mathcal{C}_d . \mathcal{C}_d describes
136 the behaviour of \mathcal{C}_c at given sampling instants.
- 137 2. The model of the plant in continuous time \mathcal{P}_c is discretised, obtaining a discrete-time
138 plant model \mathcal{P}_d . \mathcal{P}_d describes the behaviour of \mathcal{P}_c at given sampling instants. A controller
139 \mathcal{C}_d is designed directly in the discrete-time framework, using the discrete-time plant model
140 \mathcal{P}_d . Closed-loop properties are proven on the feedback interconnection of \mathcal{C}_d and \mathcal{P}_d .

141 In both cases, when an object (being it the plant or the controller) is discretised, a sampling
142 period π is chosen. With either design methods, we can obtain a discrete-time model of the
143 plant \mathcal{P}_d and of the controller \mathcal{C}_d . In control theory, usually it is possible to prove properties
144 of the interconnection between these two models. In particular, we use \mathcal{C}_d rather than \mathcal{C}_c to
145 prove properties using the controller that is closer to the real implementation. However, on
146 top of what is done in classical control theory, we want to take into account deadline misses.

147 We discretise \mathcal{P}_c from Equation (2). From the representation in terms of ordinary
148 differential equations, we obtain the system of difference equations \mathcal{P}_d as

$$149 \quad \mathcal{P}_d : x_{[k+1]} = A_d x_{[k]} + B_d u_{[k]}. \quad (3)$$

150 Here, k counts the sampling instants (i.e., there is a distance of π [s] between the k -th and
151 the $k + 1$ -th instant). The matrices A_d and B_d are the counterparts of A_c and B_c for the
152 continuous-time system. They describe the evolution of the system in discrete-time, have the
153 same dimensions of the corresponding continuous-time matrices, and their elements depend
154 on the choice of the sampling period π .

155 **Controller Model:** Once a model of the plant is available, control design can be carried
156 out with many different methods. In this paper we tackle periodic controllers expressed as
157 state feedback controllers, i.e., controllers that execute periodically with period π and whose
158 discrete-time form is

$$159 \quad u_{[k]} = K_k x_{[k]}. \quad (4)$$

160 The control design problem is the problem of finding the matrix K_k that stabilises the system
161 and obtains some desired properties. The state feedback formulation is more general than it
162 may seem at a first glance. State feedback controllers are not purely proportional controllers,
163 although their update is proportional to the state vector. It is possible to augment the state
164 vector of the system – for example introducing an error term and its integral – to achieve
165 controllers that are not simply proportional but contain integral action.⁵ Additionally, it

⁵ The most widely adopted controllers in industry are the Proportional and Integral (PI) or the Proportional, Integral and Derivative (PID) controllers. These controllers can be expressed in state-feedback form (as seen later in Section 5 for a specific example), by augmenting the system state $x(t)$ with the difference between the desired state values and the obtained ones, i.e., the error, and its integral, or

166 is possible to use pole placement [26], or to compute optimal controllers using the Linear
167 Quadratic Regulator [25] formulation.

168 In an industrial setting,⁶ many controllers are still designed assuming zero latency and
169 instantaneous computation [46], i.e., assuming that it takes zero time to retrieve the sensor
170 measurement from the plant, compute the control signal, and apply it. When the dynamics
171 of the plant are slow and the controller is able to sample and measure signals at a reasonable
172 speed, this assumption does not significantly affect the behaviour of the system. However, in
173 most cases, basic properties like stability can be violated because of the computational delays
174 that are introduced in the loop. The controller job that is activated at time t_a completes its
175 execution at time t_c , where t_c is in the controller period, i.e., $t_c \in (t_a, t_a + \pi]$, introducing a
176 computational delay $t_c - t_a$.

177 Due to this computational delay, in industry, it became common practice to design control
178 systems following the Logical Execution Time (LET) paradigm and to synchronise input
179 and output exactly to the period boundary. In this case, the control signal is computed
180 within a control period and applied at the beginning of the next period. This enhances the
181 predictability of the system, allows the processor to execute other tasks without affecting the
182 control properties, and ensures a consistent behaviour.

183 In control terms, this means that the controller actuates its control signal computation
184 with a one-step delay. Assuming that the cycle of sampling, computing, and actuating can be
185 always terminated within a control period, this allows the designer to synthesise an optimal
186 controller regardless of the time-varying components of the computational delay such as
187 activation jitter, unpredictable interrupts, uncertain computation times [28]. The equation
188 for the state feedback controller then becomes

$$189 \quad \mathcal{C}_d : u_{[k]} = K x_{[k-1]}, \quad (5)$$

190 where K is the designed controller. With very few exceptions, the vast literature on control
191 design assumes that the deadlines to compute control signals are always met. Recently,
192 Linsenmayer and Allgöwer started to connect the theory of m - K real-time systems (i.e., the
193 $\tau \vdash \overline{\langle \frac{m}{K} \rangle}$ model) with control design [29], showing that it is in some cases possible to design a
194 state feedback controller that is robust (i.e., guarantees stability) to deadline misses. In this
195 paper we will connect the amount of possible consecutive deadline misses (i.e., the $\tau \vdash \overline{\langle n \rangle}$
196 model) to the analysis of stability as a control design property.

197 **Feedback Interconnection:** Assume there are no deadline misses. In this case, we can
198 plug the value of $u_{[k]}$ obtained from Equation (5) into the plant Equation (3), obtaining

$$199 \quad x_{[k+1]} = A_d x_{[k]} + B_d K x_{[k-1]}. \quad (6)$$

200 To analyse the closed-loop system, we define a new state variable $\tilde{x}_{[k]} = [x_{[k]}^T, x_{[k-1]}^T]^T$ (the
201 superscript T indicates the result of the transposition operator). We recall that p denotes
202 the order of the system (i.e., the number of state variables in vector $x_{[k]}$). Using the new

sum, over time. There is a small difference between the controller expressed in state-feedback form and the controller expressed as a state-space system. In the first case, when the controller misses its deadline, the update function for the state is still executed (as it is now part of the system equation). It is however possible to generalise the findings in this paper to handle controllers in state-space form.

⁶ In fact, a survey published in 2001 by Honeywell [11] states that 97% of the existing industrial controllers are PI controllers and use no delay compensation. This does not mean that the control community has not developed solutions to properly address delays in the control design. It simply means that in many industrial settings the design is still simple and limited to considering the computation instantaneous.

203 state variable $\tilde{x}_{[k]}$, Equation (6) can be rewritten as

$$204 \quad \tilde{x}_{[k+1]} = \begin{bmatrix} x_{[k+1]} \\ x_{[k]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & B_d K \\ I_p & 0_{p \times p} \end{bmatrix}}_A \begin{bmatrix} x_{[k]} \\ x_{[k-1]} \end{bmatrix} = A \tilde{x}_{[k]}, \quad (7)$$

205 where I_p and $0_{p \times p}$ are respectively the identity matrix and the zero matrix of size of the
206 number of state variables p .

207 **Stability:** A discrete-time linear time-invariant system is asymptotically stable if and only if
208 all the eigenvalues of its state matrix are strictly inside the unit disk. For the system shown
209 in Equation (7), this means that the eigenvalues of A should have magnitude strictly less
210 than one.

211 Another way of formulating the stability requirement uses the concept of *spectral radius*
212 $\rho(A)$. The spectral radius is defined as the maximum magnitude of the eigenvalues of A . If
213 we denote with $\{\lambda_1, \dots, \lambda_n\}$ the set of eigenvalues of A , this means

$$214 \quad \rho(A) = \max \{|\lambda_1|, \dots, |\lambda_n|\}. \quad (8)$$

215 Requiring that all the eigenvalues have magnitude strictly less than one is equivalent to
216 stating that the spectral radius of the A matrix should be less than 1.

217 This only proves the stability of the system in absence of deadline misses. However, we
218 are aware that sporadic misses can occur, either due to faults [16] or to the chosen period π
219 not satisfying the requirement of worst-case response time for the controller task τ being less
220 than the controller period [12, 33, 34].

221 **3** Deadline Miss

222 In order to properly analyse the closed-loop system properties when deadlines can be missed,
223 it is necessary to define a model of how the system reacts to deadline misses. There are
224 two aspects of this reaction: (i) what is the chosen control signal when a miss occurs [29],
225 and (ii) how is the operating system treating the job that missed the deadline [33]. In the
226 remainder of this section, suppose that in the k -th iteration the controller task τ did not
227 complete its execution before the deadline, i.e., it does not complete its computation before
228 time $(k+1)\pi$ [s]. We denote time $(k+1)\pi$ [s] with t_m .

229 **Control Signal:** At time t_m , a control signal should be applied to the plant. Two alternatives
230 have been identified for how to select the next control signal [29]: zero and hold.

- 231 1. *Zero:* The control signal $u_{[k+1]}$ is set to zero.
- 232 2. *Hold:* The control signal $u_{[k+1]}$ is unchanged, i.e., it is the previous value of the control
233 signal $u_{[k]}$.

234 The choice of these two alternatives often depends on the control goal that should be achieved.

235 When a controller is designed for setpoint tracking (i.e., to ensure that the value of some
236 physical quantity follows a desired profile – e.g., to have a robot follow a desired trajectory),
237 the control signal is usually zero in case the measured physical quantity is equal to its setpoint.
238 In this case, setting the control signal to zero means assuming that the model of the plant is
239 correct and the computation does not need correction. When a controller is designed for
240 disturbance rejection (i.e., to ensure that the effect of some physical disturbance is not visible
241 in the measurements – e.g., to keep the altitude of a helicopter constant despite wind) then
242 the control signal is usually a reflection of the effort needed to counteract the disturbance.
243 In this case, holding the previous value of the control signal means making the assumption
244 that the system is experiencing the same disturbance.

245 **System-Level Action:** The second decision to make is the choice of what to do with the
 246 job that missed the deadline. In this case three alternatives have been proposed [33]: kill,
 247 skip-next, and queue(1).

- 248 1. *Kill:* At time t_m the job that missed the deadline is killed and a new job is activated.
- 249 2. *Skip-next:* At time t_m the job that missed its deadline is allowed to continue with the
 250 same scheduling parameters (e.g., priority or budget) and carries on in the next period.
 251 The job that should have been activated at the deadline missed is not activated, and the
 252 next activation is set to $t_m + \pi$.
- 253 3. *Queue(1):* At time t_m the job that missed its deadline is continued. A new job is activated
 254 with deadline $t_m + \pi$. The two jobs share the scheduling parameters during the period
 255 interval $[t_m, t_m + \pi]$. At time $t_m + \pi$, the most recent update of the control value is
 256 applied. If both jobs finish their computation, the control variable is set to the value
 257 produced by the most recently activated job (i.e., the job that started at time t_m and
 258 was placed in the queue until the old job that missed its previous deadline finished). If
 259 only the first job finishes the computation the control variable is set to the value of the
 260 job that finished and the following one is continued in the subsequent period.

261 In Section 4 we will analyse the system in all possible configurations. However, we point
 262 out that, from an implementation perspective, killing the control job may not be feasible
 263 in many industrial settings. In fact, the system has reached an inconclusive intermediate
 264 state. The internal state of the controller could have been updated and the system should
 265 implement a clean rollback of these changes. Implementing a clean rollback procedure is
 266 risky. Furthermore, if the lengthy computation (and subsequent deadline miss) is due to the
 267 received input values, it is likely that the next iteration will start from state values that are
 268 fairly close to the previous ones, with higher than normal risk of missing a deadline.

269 We also notice that enqueueing the task could be beneficial from the control perspective,
 270 because a computation with most recent measurements of the state variables could be
 271 applied. However, the scheduling parameters for τ have most likely been tuned for one single
 272 control job to be executed in a period. For example, if the control task is executed using
 273 reservation-based scheduling, its budget is selected to match one execution. When using
 274 fixed-priority scheduling, the controller priority has been selected. Executing a second control
 275 task may create ripple effects and have a disruptive effect on lower priority tasks.

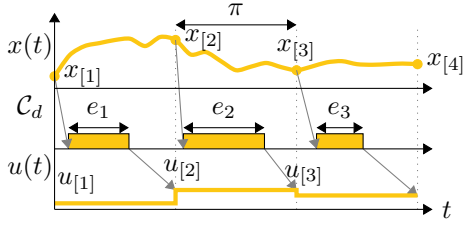
276 Finally, if the deadline is missed, this means that the system is likely experiencing a
 277 transient overload state, which would make *skip-next* the best option to relieve some pressure
 278 from the system.

279 4 System Analysis

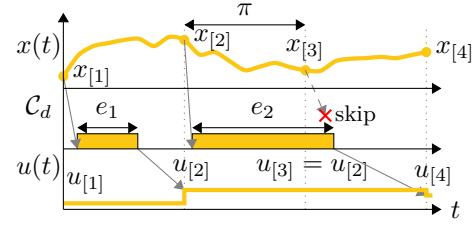
280 In this section we present our analysis of the closed-loop system with deadline misses. We
 281 first discuss the fundamentals of what happens from the physical perspective when a deadline
 282 is missed and then discuss the combinations identified in Section 3 for how the system handles
 283 the miss.

284 **Fundamentals:** Here we present the general methodology that we apply to verify the stability
 285 of closed-loop systems with different strategies. We cast the problem into a switching-systems
 286 stability problem and show how real-world implementations behave.

287 Within one control period, there are two possible realisations. The controller job that was
 288 activated at time $k\pi$ can either hit or miss its deadline. Figure 1 shows the case in which no
 289 deadline is missed, while Figure 2 shows the behaviour of the system when a deadline miss
 290 occur with the *hold and skip-next* strategy. In the figures, we use e_i to indicate the execution



■ **Figure 1** System evolution in case no deadline is missed. The state feedback controller C_d computes the control signal u based on measurements of the state x .



■ **Figure 2** System evolution in case of a deadline miss with the hold policy and skip-next strategy. The controller misses the deadline and completes in the subsequent period.

291 time of the i -th job of the controller. The figure just provides a visual representation of a
 292 lengthy execution, but misses can occur due to other sources of interference, e.g., higher
 293 priority task being executed with a fixed-priority scheduling algorithm, interrupts being
 294 raised and served during the execution of the control task, or access to locked shared resources
 295 being requested. In Figure 2, the control signal $u_{[2]}$ is held as $u_{[3]}$. The next controller
 296 execution instance is skipped and the result of the completion of e_2 is applied as $u_{[4]}$.
 297

The procedure that we follow to analyse the closed loop system is the following:

- 298 1. We express the dynamics of the closed-loop system in the cases of hit and of miss.
 299 Following a procedure similar to the one we used in Equation (7), we determine the state
 300 matrices for the closed-loop systems in case of deadline hit and deadline miss, respectively
 301 A_H and A_M . We then know that the system with (unconstrained) deadline misses can be
 302 expressed as a switching system [27] that arbitrarily switches between these two matrices.
 303 If the original system in Equation (3) was unstable, there is no hope that the switching
 304 system that arbitrarily switches between A_H and A_M is stable (as either an old or no
 305 control action is applied when a miss occurs). However, we still have not introduced any
 306 weakly hard constraint.
- 307 2. We determine the set of possible cases for the evolution of the system when $\tau \vdash \overline{\langle n \rangle}$
 308 guarantees are provided, i.e., the possible realisations of the system behaviour. We
 309 denote with Σ the set of possible matrices that these realise. For $\tau \vdash \overline{\langle n \rangle}$ guarantees,
 310 the set of possible realisations is $\{H, MH, \dots, M^n H\}$. The set contains either a single
 311 hit, or a certain number of misses (up to n) followed by a hit.⁷ This means that
 312 $\Sigma = \{A_H, A_H A_M, A_H A_M^2, \dots, A_H A_M^n\}$. This can be written in a compact form as $\Sigma =$
 313 $\{A_H A_M^i \mid i \in \mathbb{Z}^{\geq}, i \leq n\}$ where \mathbb{Z}^{\geq} indicates the set of integers including zero. Notice
 314 that matrices are multiplied from the right to the left (denoting the standard evolution
 315 of the system from a mathematical standpoint). This step introduces the weakly hard
 316 constraint for which we investigate the system stability.
- 317 3. We compute a generalisation of the spectral radius concept, called *joint spectral radius*
 318 $\rho(\Sigma)$ [21, 36], that allows us to assess the stability of the closed-loop system that switches
 319 between the realisations (i.e., the valid scenarios including a number of misses between

⁷ The notation used to define Σ is slightly simplified here, as the matrix A_H may be different depending on how many deadlines have been missed (for example, with the skip-next strategy the controller uses an old measurement of the state to compute the control signal). We will be more precise in the following when we show how to apply the procedure to the different cases. Furthermore, notice that the matrices in Σ represent the evolution across a different number of time steps: A_H advances the time in the system of π , while $A_H A_M$ advances the system time of 2π . This is not a concern for the system analysis.

320 0 and n followed by a hit) included in Σ . More precisely, the closed-loop system that
 321 can switch between the realisations included in Σ is asymptotically stable *if and only if*
 322 $\rho(\Sigma) < 1$ [21, Theorem 1.2].

323 In order to generalise the spectral radius to a set of matrices, we introduce some notation.
 324 The following paragraphs are using the notation and sequential treatise proposed in [21] to
 325 introduce the concept of the joint spectral radius. We recap only what is needed for the
 326 purpose of understanding our analysis.

327 **Joint Spectral Radius [21, 36]:** The first step for our definition is to determine what
 328 happens when some steps of evolution of the switching system occur. We then denote with
 329 $\rho_\mu(\Sigma)$ the spectral radius of the matrices that we find after μ -steps. Precisely,

$$330 \quad \rho_\mu(\Sigma) = \sup\{\rho(A)^{1/\mu} : A \in \Sigma^\mu\}. \quad (9)$$

331 In this definition we quantify the average growth over μ time steps, as the supremum of the
 332 spectral radius (elevated to the power $1/\mu$) of all the matrices that can be evolutions of the
 333 system after μ matrix multiplications (i.e., after μ evolution steps, where an evolution step
 334 is either a hit or a set of constrained misses followed by a hit). Equation (9) denotes the
 335 supremum of all the possible combinations of products of μ matrices that are included in Σ .

336 Using $\rho_\mu(\Sigma)$ we can define the joint spectral radius of a bounded set of matrices Σ as

$$337 \quad \rho(\Sigma) = \limsup_{\mu \rightarrow \infty} \rho_\mu(\Sigma). \quad (10)$$

338 We are then looking at the evolution of the system for an infinite amount of time, i.e., pushing
 339 μ to the limit.

340 Determining that the switching system is asymptotically stable is equivalent to assessing
 341 that the joint spectral radius of the set of matrices Σ is less than 1. This condition is both
 342 sufficient and necessary [21, Theorem 1.2]. This means that if the joint spectral radius is
 343 higher than 1, there is at least a sequence of switches of hits and misses that destabilises the
 344 closed-loop system.

345 **Joint Spectral Radius Computation:** On the practical side, the problem of computing
 346 if the joint spectral radius is less than 1 is undecidable [8]. In many cases it is possible to
 347 approximate the joint spectral radius with satisfactory precision [6, 7, 17, 32] and obtain upper
 348 and lower bounds for $\rho(\Sigma)$. Clearly, the closest the two bounds are, the more precise is the
 349 estimation of the true value of the joint spectral radius. We can safely say that our controller
 350 design is sufficiently robust to deadline misses if the upper bound on the joint spectral radius
 351 $\rho(\Sigma)$ is less than 1.

352 **Joint Spectral Radius with at most n Consecutive Misses:** If the joint spectral radius
 353 of the set Σ is less than 1, the stability of all the combinations of realisations (of hits and
 354 misses, that include at most n consecutive misses) is proven, regardless of the window size.

355 For example, let us assume that we are analysing a system with the real-time guarantee
 356 that we cannot experience more than two consecutive misses. The realisations that we
 357 analyse are $\{A_H, A_H A_M, A_H A_M A_M\}$ and the joint spectral radius unfolds and checks all
 358 the possible (infinitely long) sequences of combinations of these realisations.

359 For a length of two, this means that we check: (1) $A_H A_H$ as the product of the first
 360 term twice, (2) $A_H A_H A_M$ as the product of the first two terms picking the first as final
 361 (in terms of time evolution of the system), (3) $A_H A_H A_M A_M$ as the product of the first
 362 and last terms picking the first as initial, (4) $A_H A_M A_H$ as the product of the first two
 363 terms picking the second as final, (5) $A_H A_M A_H A_M$ as the product of the second term twice,
 364 (6) $A_H A_M A_H A_M A_M$ as the product of the last two terms, picking the second as final, (7)

365 $A_H A_M A_M A_H$ as the product of the last and first term, (8) $A_H A_M A_M A_H A_M$ as the product
 366 of the last and second term, (9) $A_H A_M A_M A_H A_M A_M$ as the last term twice. This procedure
 367 is repeated for more products, up to *infinitely long* sequences. From the computation side,
 368 the results are an upper and a lower bound on the value of the (true) joint spectral radius.

369 The analysis is sound on the control side, as stability is guaranteed if and only if the joint
 370 spectral radius is less than 1. On the theoretical side, this demonstrates that the $\tau \vdash \overline{\langle n \rangle}$
 371 model can elegantly provide a necessary condition for the stability of the system. The only if
 372 part means that there is at least a sequence of hits and misses (where at most we experience
 373 n consecutive misses) that causes the system to be unstable if the (true value of the) joint
 374 spectral radius is larger than 1.

375 Considering that we only compute an upper and lower bound on the joint spectral radius,
 376 what we can conclude is: if the lower bound that we obtain is above 1, we are entirely certain
 377 that such a sequence exists, while if the lower bound is below 1 and the upper bound is
 378 above 1 we have no mathematical certainty that the system is unstable. Nonetheless, on the
 379 practical side, the bounds obtained with modern approximation techniques [6, 7, 17, 32] are
 380 usually very close to one another, implying that they are a very good estimate of the true
 381 value of the joint spectral radius.

382 It is important to note that even if the control system is able to stabilise the system
 383 in the presence of n consecutive misses, this does not mean that executing the controller
 384 with a period of $n\pi$, rather than its original period π , is a sensible choice. In fact, the
 385 performance (measured for example using the integral of the squared error) of the controller
 386 that is executing with a larger period would be dramatically worse than the performance
 387 of the controller with the shorter period that can experience misses. Being able to tolerate
 388 misses is very different than performing well when these misses occur.

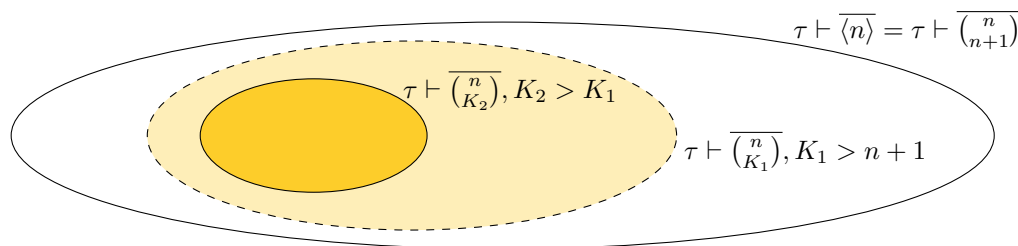
389 **Relation with m - K model:** We here briefly discuss the relation between the guarantees
 390 that we obtain with the $\tau \vdash \overline{\langle n \rangle}$ model and the m - K model, $\tau \vdash \overline{\langle \frac{m}{K} \rangle}$.

391 The $\tau \vdash \overline{\langle n \rangle}$ model includes all the realisations that are contained in the $\tau \vdash \overline{\langle \frac{n}{K} \rangle}$ regardless
 392 of the value of K . However, it can also include additional realisations (that could generate
 393 instability) that are not included in the $\tau \vdash \overline{\langle \frac{n}{K} \rangle}$ model, if $K > n + 1$. The $\tau \vdash \overline{\langle n \rangle}$ model
 394 over-approximates the set of possible realisations that one can obtain with an m - K task
 395 (assuming that $n = m$). This means that there is a chance that an m - K control task stabilises
 396 the system when the corresponding task with $n = m$ consecutive deadline misses would not.

397 The difference between n and K determines the extent of the potential over-approximation.
 398 With a smaller difference, the set of possible realisations converges to the set of realisations
 399 that are included in the $\tau \vdash \overline{\langle n \rangle}$ model. More precisely, the set of possible realisations
 400 considered with the $\tau \vdash \overline{\langle \frac{n}{n+1} \rangle}$ model is the same as the set obtained with the $\tau \vdash \overline{\langle n \rangle}$ model.
 401 However, increasing K , reduces the set of possibilities that are considered, shrinking the
 402 size of the set of valid realisations. Figure 3 shows the relationship between three sets when
 403 $K_2 > K_1 > n + 1$.

404 If the system with $\tau \vdash \overline{\langle n \rangle}$ is found stable, then control task τ that is given m - K guarantees
 405 also stabilises the plant if $m \leq n$. This means that as a first approximation, regardless of the
 406 value of K , when dealing with the m - K model, one can check the stability for a maximum of
 407 m consecutive deadline misses and if the condition is satisfied, then the closed-loop is stable
 408 regardless of the value of K .

409 As a final remark, from the industrial point of view, the analysis of the case when $K \gg n$
 410 is not particularly interesting, because stability and quality of control guarantees are provided
 411 by the design of robust controllers [26] (i.e., a small perturbation in scheduling is anyway
 412 covered by the redundancy and design of control systems). On the contrary, there is a clear



■ **Figure 3** Sets of potential realisations with different models.

413 industrial interest in analysing the $K \approx n + 1$ case, due to transient heavy load perturbations.
 414 **Application:** We now show how to apply the theory to practical case studies. We imple-
 415 mented our analysis methods in MATLAB[®]. The input values for our stability verification
 416 procedures are: n (the number of contiguous deadline that the system can miss), A_d and
 417 B_d (the matrices that determine the dynamics of the system), and K (the controller that is
 418 designed and should be validated). We used the JSR Matlab toolbox [22, 45] to compute
 419 bounds on the joint spectral radius. We constructed the set Σ based on the expressions
 420 derived for the given deadline-handling methods, i.e., for the combination of the control signal
 421 management policy (*zero* or *hold*) and the system-level action (*kill*, *skip-next*, or *queue(1)*).
 422 **Zero&Kill:** The zero and kill strategy is the simplest to analyse. For this strategy we can
 423 look at Equations (3) and (5). The system behaves according to

$$424 \quad \tilde{x}_{[k+1]} = \begin{bmatrix} x_{[k+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & B_d \\ K & 0_{r \times r} \end{bmatrix}}_{A_H} \begin{bmatrix} x_{[k]} \\ u_{[k]} \end{bmatrix} = A_H \tilde{x}_{[k]} \quad (11)$$

425 in case of a hit. Notice that this is in principle exactly the same for each strategy, as when the
 426 controller hits the deadline the behaviour is the same. Kill implies that in case of a deadline
 427 miss there is an abort of what the control task has been computing up to its deadline, which
 428 means there is no need to take into account its (partial) behaviour. In case of a deadline
 429 miss in the k -th iteration, the control signal $u_{[k+1]}$ is set to zero. This means that the system
 430 evolves according to

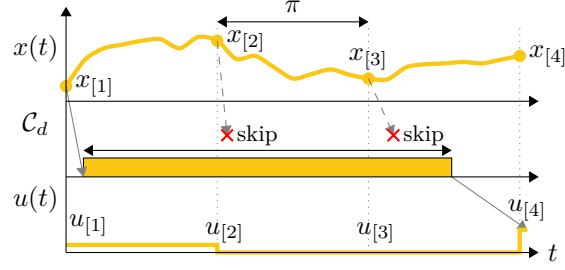
$$431 \quad \tilde{x}_{[k+1]} = \begin{bmatrix} x_{[k+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & B_d \\ 0_{r \times r} & 0_{r \times r} \end{bmatrix}}_{A_M} \begin{bmatrix} x_{[k]} \\ u_{[k]} \end{bmatrix} = A_M \tilde{x}_{[k]} \quad (12)$$

432 in case of a miss. With n maximum consecutive misses, we can then compute all the matrices
 433 in $\Sigma = \{A_H A_M^i \mid i \in \mathbb{Z}^{\geq}, i \leq n\}$ ⁸ and then compute the upper bound on $\rho(\Sigma)$.

434 We would like to remark that while this strategy is simple to analyse, for practical
 435 applications it is hard to guarantee – using kill – that the control task τ will miss at most n
 436 consecutive deadlines (as the state of the running task is always re-set at every period start).

437 **Zero&Skip-Next:** The difference between Zero&Kill and Zero&Skip-Next lays in the
 438 freshness of the measurements that are used for the computation of the control signal when
 439 the task τ hits its deadline. In fact, if the task was killed and a new job was activated then
 440 the state measurement would occur at the beginning of the new activation, while if the task

⁸ As defined in the introductory part of Section 4 (see point 2 in *Fundamentals*), \mathbb{Z}^{\geq} indicates the set of integers including zero.



■ **Figure 4** System evolution in case of two consecutive deadline misses with the skip-next strategy and the zero policy.

441 was allowed to continue, it would use old measurements. Figure 4 shows how the system
 442 evolves in case of consecutive deadline misses. Suppose that the control task τ completes its
 443 execution in the third period. Then this is not equivalent to experiencing two misses and a
 444 hit, because the completion uses old state measurements. We need to then express the state
 445 matrix evolution when there is a *recovery hit*, rather than a regular hit (in the figure $u_{[4]}$
 446 is set using $x_{[1]}$).

447 To properly analyse this system, our state has to include the previous values that can
 448 be used for the control signal computation. With $\tau \vdash \overline{\langle n \rangle}$ guarantees, we can then define
 449 our augmented state vector as $\tilde{x}_{[k]} = [x_{[k]}^T, x_{[k-1]}^T, \dots, x_{[k-n]}^T, u_{[k]}^T]^T$, i.e., the state vector of
 450 the closed-loop system is composed of $n + 1$ elements of the state vector and 1 element for
 451 the control signal.

452 Then we can write the closed-loop system in case of a deadline hit, i.e., the equivalent of
 453 Equation (7), as

$$454 \begin{bmatrix} x_{[k+1]} \\ x_{[k]} \\ \dots \\ x_{[k-n+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & 0_{p \times (n \cdot p)} & B_d \\ & I_{n \cdot p} & 0_{(n \cdot p) \times (p+r)} \\ K & 0_{r \times (n \cdot p)} & 0_{r \times r} \end{bmatrix}}_{A_H} \begin{bmatrix} x_{[k]} \\ x_{[k-1]} \\ \dots \\ x_{[k-n]} \\ u_{[k]} \end{bmatrix}. \quad (13)$$

455 Here, we added some padding to the matrix to identify that state variables are transferred from
 456 one time instant to the next; i.e., to add the trivial equations $x_{[i]} = x_{[i]}, \forall i \mid x - n + 1 \leq i \leq k$.
 457 In fact, when the deadline is hit (not after a miss) there is no use of the previous values of
 458 the state.

459 When a miss occurs, the control signal is set to zero. This means that we can use the
 460 A_H matrix defined in Equation (13) and substitute the value of K with a zero matrix of
 461 appropriate size, i.e., $0_{r \times p}$ to obtain the A_M matrix,

$$462 \begin{bmatrix} x_{[k+1]} \\ x_{[k]} \\ \dots \\ x_{[k-n+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & 0_{p \times (n \cdot p)} & B_d \\ & I_{n \cdot p} & 0_{(n \cdot p) \times (p+r)} \\ & 0_{r \times (n+1) \cdot p+r} \end{bmatrix}}_{A_M} \begin{bmatrix} x_{[k]} \\ x_{[k-1]} \\ \dots \\ x_{[k-n]} \\ u_{[k]} \end{bmatrix}. \quad (14)$$

463 Now we should remark that a hit and a *recovery hit* have two different matrix realisation,
 464 i.e., a hit that follows a certain number of misses (up to n) has a different matrix with respect
 465 to A_H . In fact, we have to take into account the use of the old state measurement to produce
 466 the control signal of the recovery hit. We denote with A_{R_i} the matrix that represents the

467 evolution of the closed-loop system when a recovery happens after i deadlines were missed.
 468 This matrix can be constructed modifying the last row of A_H , and switching the position
 469 of K to use the correct state vector (i.e., the one that corresponds to the measurements
 470 obtained n steps before).

471 For $i = 1$, i.e., with one deadline miss, we can write

$$472 \begin{bmatrix} x_{[k+1]} \\ x_{[k]} \\ \dots \\ x_{[k-n+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & 0_{p \times (n \cdot p)} & B_d \\ & I_{n \cdot p} & 0_{(n \cdot p) \times (p+r)} \\ 0_{r \times p} & K & 0_{r \times (n-1) \cdot p} \\ & & 0_{r \times r} \end{bmatrix}}_{A_{R_1}} \begin{bmatrix} x_{[k]} \\ x_{[k-1]} \\ \dots \\ x_{[k-n]} \\ u_{[k]} \end{bmatrix}. \quad (15)$$

473 Consistently with our treatise, $A_{R_0} = A_H$. We can then compute the set Σ as $\Sigma = \{A_{R_i} A_M^i \mid$
 474 $i \in \mathbb{Z}^{\geq}, i \leq n\}$ and then compute the upper bound on $\rho(\Sigma)$ using the computed set of
 475 matrices⁹.

476 **Zero&Queue(1)**: The behaviour of the combination of the zero policy and the queue(1)
 477 strategy vary depending on the possibility of the queued job to complete before the deadline
 478 or not. We are going to make the additional hypothesis that the worst-case response time for
 479 a job is less than $n\pi$, where n is the maximum number of consecutive deadline misses. We
 480 can start from the set Σ used for the Zero&Skip-Next combination and add to the set all
 481 the matrices $A_H A_M^i$, that take into account the possibility that the queued job completed
 482 before its deadline. We should also include in the set Σ the matrices A_{R_i} alone, as it could
 483 happen that a queued job doesn't terminate in the period it was started in. We then obtain
 484 $\Sigma = \{A_H A_M^i, A_{R_i}, A_{R_i} A_M^i \mid i \in \mathbb{Z}^{\geq}, i \leq n\}$, and we can use the set to compute the upper
 485 bound on the joint spectral radius $\rho(\Sigma)$.

486 **Hold&Kill**: The hold and kill strategy aborts the task but applies the previously computed
 487 control signal to the plant. An easy way to analyse this case is to include in the state of the
 488 system also the control signal, such that we can determine the switch between two different
 489 matrices without having the matrices grow. We denote with $\tilde{x}_{[k]} = [x_{[k]}^T, u_{[k]}^T]^T$. We recall
 490 that r is used to represent the number of input variables.

491 We can write the evolution of the system when a deadline is hit as

$$492 \begin{bmatrix} x_{[k+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & B_d \\ K & 0_{r \times r} \end{bmatrix}}_{A_H} \begin{bmatrix} x_{[k]} \\ u_{[k]} \end{bmatrix}, \quad (16)$$

493 meaning that the computation (the third row of the A_H matrix) is completed and the new
 494 control variable is updated with the information from the plant. When a deadline is missed,
 495 we compute the system evolution as

$$496 \begin{bmatrix} x_{[k+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & B_d \\ 0_{r \times r} & I_r \end{bmatrix}}_{A_M} \begin{bmatrix} x_{[k]} \\ u_{[k]} \end{bmatrix}, \quad (17)$$

⁹ The drawback of constructing the set Σ as shown above is that the size of the matrices in the set grows linearly with the number of deadline misses. It is possible to construct a compact representation that uses as state vector $\tilde{x}_{[k]} = [x_{[k]}^T, u_{[k]}^T]^T$ but writes the evolution of the system directly as the relation between $\tilde{x}_{[k]}$ and $\tilde{x}_{[k+n+1]}$. This second way of expressing the system dynamics has the disadvantage of hiding misses and hits and only showing the evolution at each hit (still keeping track of what happened at the instants in which the misses occurred). We implemented both versions in our code and checked that the obtained results are the same except for the computational speedup. This remark applies to all the strategies using *skip-next*.

497 encoding the hold as the identity matrix that multiplies the old control value for the equation
 498 that determines the evolution of $u_{[k]}$. As done for the zero&kill alternative, if we assume
 499 there can be a maximum of n consecutive deadline misses, we can then compute all the
 500 matrices in $\Sigma = \{A_H A_M^i \mid i \in \mathbb{Z}^{\geq}, i \leq n\}$ and then compute the upper bound on $\rho(\Sigma)$.

501 **Hold&Skip-Next:** In order to analyse the combination of hold and skip-next we need to
 502 augment the state vector as we did for the zero&skip-next handling strategy. Also in this
 503 case, we use our newly defined state vector $\tilde{x}_{[k]} = [x_{[k]}^T, x_{[k-1]}^T, \dots, x_{[k-n]}^T, u_{[k]}^T]^T$. We obtain
 504 the following expression for the closed-loop system when we hit a deadline,

$$505 \begin{bmatrix} x_{[k+1]} \\ x_{[k]} \\ \dots \\ x_{[k-n+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & 0_{p \times (n \cdot p)} & B_d \\ & I_{n \cdot p} & 0_{(n \cdot p) \times (p+r)} \\ K & 0_{r \times (n \cdot p)} & 0_{r \times r} \end{bmatrix}}_{A_H} \begin{bmatrix} x_{[k]} \\ x_{[k-1]} \\ \dots \\ x_{[k-n]} \\ u_{[k]} \end{bmatrix}. \quad (18)$$

506 When we miss a deadline we use the old control value, introducing an identity matrix in the
 507 last column and last row of the closed-loop state matrix to indicate that the previous control
 508 signal is saved,

$$509 \begin{bmatrix} x_{[k+1]} \\ x_{[k]} \\ \dots \\ x_{[k-n+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & 0_{p \times (n \cdot p)} & B_d \\ & I_{n \cdot p} & 0_{(n \cdot p) \times (p+r)} \\ 0_{r \times (n+1) \cdot p} & & I_r \end{bmatrix}}_{A_M} \begin{bmatrix} x_{[k]} \\ x_{[k-1]} \\ \dots \\ x_{[k-n]} \\ u_{[k]} \end{bmatrix}. \quad (19)$$

510 Finally, we should define the behaviour of system in the case of a recovery hit, as done
 511 for the zero&skip-next strategy, but including the dynamic evolution of the control signal
 512 that has been added to \tilde{x} . For one deadline miss we obtain A_{R_1} as

$$513 \begin{bmatrix} x_{[k+1]} \\ x_{[k]} \\ \dots \\ x_{[k-n+1]} \\ u_{[k+1]} \end{bmatrix} = \underbrace{\begin{bmatrix} A_d & 0_{p \times (n \cdot p)} & B_d \\ & I_{n \cdot p} & 0_{(n \cdot p) \times (p+r)} \\ 0_{r \times p} & K & 0_{r \times (n-1) \cdot p} \\ & & & 0_{r \times r} \end{bmatrix}}_{A_{R_1}} \begin{bmatrix} x_{[k]} \\ x_{[k-1]} \\ \dots \\ x_{[k-n]} \\ u_{[k]} \end{bmatrix}, \quad (20)$$

514 and the following matrices are obtained by moving the position of the term K in the state
 515 evolution matrix to reflect how old is the sensed data that is being used for the computation
 516 of the control signal.

517 Again, $A_{R_0} = A_H$, and we can define the set Σ as $\Sigma = \{A_{R_i} A_M^i \mid i \in \mathbb{Z}^{\geq}, i \leq n\}$. With
 518 this, we can compute the upper bound on $\rho(\Sigma)$.

519 **Hold&Queue(1):** To analyse the hold&queue(1) strategy, we follow the same principles used
 520 for the zero&queue(1) strategy. We start from the hold&skip-next matrices and determine
 521 $\Sigma = \{A_H A_M^i, A_{R_i}, A_{R_i} A_M^i \mid i \in \mathbb{Z}^{\geq}, i \leq n\}$.

522 5 Experimental Validation

523 In this section we present a few examples of how the analysis presented in Section 4 can be
 524 applied to determine the robustness to deadline misses of control system implementations.
 525 In particular, we first present some results obtained with an unstable second-order system,
 526 which could be used to approximate unstable plants such as a segway that has to be stabilised

527 about the top position. Then, we verify the stability of a permanent magnet synchronous
 528 motor for an automotive electric steering application.

529 **Unstable Second-Order System:** We analyse the following continuous-time linear time-
 530 invariant open-loop system,

$$531 \quad \dot{x}(t) = \underbrace{\begin{bmatrix} 10 & 0 \\ -2 & -1 \end{bmatrix}}_{A_c} x(t) + \underbrace{\begin{bmatrix} 5 & 1 \\ 4 & 10 \end{bmatrix}}_{B_c} u(t), \quad (21)$$

532 where both the state and the input vector are composed of two variables. The expression
 533 above is the equivalent of Equation (2). Since the A_c matrix is a lower triangular matrix,
 534 one can immediately see that the poles of the system are 10 and -1 . Since one pole is in
 535 the right half plane, the system has one unstable mode and there is a need for control to
 536 stabilise the system.

537 An optimal linear quadratic regulator [26] is designed for this system, assuming that the
 538 controller execution is instantaneous and there is no one-step delay actuation, obtaining

$$539 \quad K = \begin{bmatrix} -4.7393 & 0.2430 \\ 0.2277 & -0.8620 \end{bmatrix}. \quad (22)$$

540 First, we check the stability of the closed loop system when the controller is executed with
 541 one step delay. We select a sampling period of 10 ms and discretise the system obtaining

$$542 \quad x_{[k+1]} = \underbrace{\begin{bmatrix} 1.1053 & 0.0000 \\ -0.0209 & 0.9900 \end{bmatrix}}_{A_d} x_{[k]} + \underbrace{\begin{bmatrix} 0.0526 & 0.0105 \\ 0.0393 & 0.0994 \end{bmatrix}}_{B_d} u_{[k]}. \quad (23)$$

543 This corresponds to Equation (3). The matrix A_d is also lower triangular, which means that
 544 the poles of the open-loop system are the numbers indicated in the main diagonal. Since one of
 545 them is outside the unit circle, the discretised version of the continuous-time system is (unsur-
 546 prisingly) also unstable and needs control. The poles of the closed-loop system corresponding
 547 to the execution of the LET controller every 10 ms are $\{0.8911, 0.8141, 0.3013, 0.0888\}$ and
 548 they are all inside the unit circle, meaning that the system is stabilised by the LET controller
 549 K from Equation (22), and our control design is a good choice.

550 Our research question is how many deadlines can we miss when we execute the controller
 551 with all the possible deadline miss handling strategies. Table 1 summarises the results we
 552 obtain for the analysis.

553 With the zero strategy, irrespective of the choice of how to handle the job that misses the
 554 deadline (kill, skip-next, or queue), the system is robust to missing one deadline (the upper
 555 bound on the joint spectral radius is less than 1 for one deadline miss). A subsequent miss is
 556 not tolerated, and the closed-loop system becomes provably unstable (the lower bound on
 557 the joint spectral radius is above 1). We now look at what happens when the control signal
 558 is kept constant in case of misses, i.e., when we hold. Hold&kill ensures that we could miss
 559 five deadlines in a row without the emergence of unstable behaviour. However, if a sixth
 560 deadline is missed, the system could become unstable. In this case, in fact, the upper-bound
 561 on the joint spectral radius exceeds 1. The lower bound, however, is below 1. This means
 562 that there is no complete certainty that the system is unstable, but there is a high risk. The
 563 true value (for which we have certainty) lies in between the the two bounds. If we continue
 564 our analysis, the situation where the true values is around 1 and uncertain persists up to 7
 565 deadline misses. When we introduce the possibility of missing 8 consecutive deadlines, both

■ **Table 1** Stability Results for the Unstable Second-Order System

	Misses	Stability	Lower Bound	Upper Bound
<i>Zero&Kill</i>	1	✓	0.961037	0.961975
	2	✗	1.071911	1.071915
<i>Zero&Skip-Next</i>	1	✓	0.914298	0.920769
	2	✗	1.059819	1.059822
<i>Zero&Queue(1)</i>	1	✓	0.961037	0.964287
	2	✗	1.071911	1.071915
<i>Hold&Kill</i>	1	✓	0.891089	0.891090
	2	✓	0.891089	0.891090
	3	✓	0.891089	0.891098
	4	✓	0.891089	0.891251
	5	✓	0.891089	0.935272
	6	✗	0.891089	1.004593
	7	✗	0.961344	1.083038
	8	✗	1.065537	1.172249
<i>Hold&Skip-Next</i>	1	✓	0.891089	0.891090
	2	✓	0.914556	0.944458
	3	✗	1.076507	1.091171
<i>Hold&Queue(1)</i>	1	✗	1.347066	1.370827

566 the lower-bound and the upper-bound are above 1, meaning that the system is provably
567 unstable for some sequences. Notice that this does not mean that the instability is found
568 when we repeat the sequence with 8 consecutive misses followed by a hit. It could happen
569 that the unstable sequence is a combination of a number of deadline misses up to 8 followed
570 by a different number, followed by a number of deadline hits, and so forth. In fact, in our
571 investigation we have encountered cases in which the closed-loop system was stable in case of
572 the repetition of the sequence with n misses followed by a hit, but unstable with a number
573 of consecutive misses up to n . For any practical application, terminating the investigation
574 when the upper-bound crosses 1 ensures a safety margin and guarantees the correct system
575 operation.

576 When hold is paired with skip-next the system tolerates 2 misses. When queue(1) is used,
577 on the contrary, the system does not tolerate even a single miss. As a final remark, notice
578 that while the number of tolerated deadline misses is higher for hold&kill, when a task is
579 killed it is difficult to guarantee – with real-time analysis – that the subsequent job will
580 not miss its deadline (especially due to locality effects). On the contrary, in general, when
581 skip-next is applied, it is easier to guarantee the termination of in the consecutive periods.
582 This is true unless the deadline misses are caused by a bug in the control task itself, in which
583 case the control task may never terminate.

584 **Electric Steering Application:** Here we verify the stability of a permanent magnet
585 synchronous motor for an automotive electric steering application in the presence of deadline
586 misses. We first present a standard model for the motor and a proportional and integral (PI)
587 controller for setpoint tracking, written in the state-feedback form.

588 When modeling the motor, our system state is $x(t) = [i_d(t), i_q(t)]^T$ where $i_d(t)$ and $i_q(t)$
589 represent respectively the currents in the d and q coordinates over time. The aim of our
590 control design is to track arbitrary reference values for the currents. The control signal is

591 $u(t) = [u_d(t), u_q(t)]^T$, where $u_d(t)$ and $u_q(t)$ represent respectively the voltages applied to
 592 the motor in the d and q coordinate system (subject to an affine transformation). The model
 593 of the motor can be written as

$$594 \quad \dot{x}(t) = \begin{bmatrix} -R/L_d & L_q \omega_{el}/L_d \\ -L_d \omega_{el}/L_q & -R/L_q \end{bmatrix} x(t) + \begin{bmatrix} 1/L_d & 0 \\ 0 & 1/L_q \end{bmatrix} u(t). \quad (24)$$

595 Here, L_d [ht] and L_q [ht] denote respectively the inductance in the d and q direction, R [Ohm]
 596 is the winding resistance, and ω_{el} [rad/s] is the frequency of the rotor-induced voltage
 597 (assumed as constant). We used the parameters of our motor¹⁰ and discretised the plant
 598 using Tustin's method¹¹, and a sampling period of $10\mu\text{s}$, obtaining

$$599 \quad x_{[k+1]} = \underbrace{\begin{bmatrix} 0.996 & 0.075 \\ -0.052 & 0.996 \end{bmatrix}}_{A_{d,\text{base}}} x_{[k]} + \underbrace{\begin{bmatrix} 0.100 & 0.003 \\ -0.003 & 0.083 \end{bmatrix}}_{B_{d,\text{base}}} u_{[k]}. \quad (25)$$

600 Notice that the eigenvalues of $A_{d,\text{base}}$ are $0.9957 \pm 0.0626i$ and their absolute value is 0.9977,
 601 meaning that the open-loop system is stable (even without control). Control is here added
 602 to constrain the behaviour of the system and make sure it tracks current setpoints without
 603 errors.

604 To achieve zero steady-state error, we would like to design our controller in the PI form,
 605 using a term that is proportional to the error between the actual current vector and the
 606 setpoint vector and a term that is proportional to the integral of the error. We then need to
 607 augment the state vector $x_{[k]}$ and keep track of the error at the current time step and at the
 608 previous time step. We also need to add to the input vector the setpoints for the currents in
 609 the D and Q directions. This allows us to write our PI controller in the state-feedback form.

610 After this transformation, we denote the new system input as $v_{[k]} = [u_{[k]}^T, w_{[k]}^T]^T$ where
 611 $w_{[k]}$ denotes a vector that contains the desired values for the currents i_d and i_q at time k .
 612 We also define the new system state as $s_{[k]} = [x_{[k]}^T, e_{[k]}^T, e_{[k-1]}^T]^T$, where $e_{[k]}$ is the error at
 613 time k , i.e., $e_{[k]} = w_{[k]} - x_{[k]}$. We therefore model the system as

$$614 \quad s_{[k+1]} = \underbrace{\begin{bmatrix} A_{d,\text{base}} & 0_{2 \times 2} & 0_{2 \times 2} \\ -I_2 & 0_{2 \times 2} & 0_{2 \times 2} \\ 0_{2 \times 2} & I_2 & 0_{2 \times 2} \end{bmatrix}}_{A_d} s_{[k]} + \underbrace{\begin{bmatrix} B_{d,\text{base}} & 0_{2 \times 2} \\ 0_{2 \times 2} & I_2 \\ 0_{2 \times 2} & 0_{2 \times 2} \end{bmatrix}}_{B_d} v_{[k]}. \quad (26)$$

615 We design our PI controller as

$$616 \quad K = \begin{bmatrix} & \overbrace{\begin{bmatrix} 5 & 0 \\ 1 & 7 \end{bmatrix}}^{K_1} & \overbrace{\begin{bmatrix} -4 & 0 \\ -3 & 7 \end{bmatrix}}^{K_2} \\ 0_{2 \times 2} & & \\ 0_{2 \times 2} & 0_{2 \times 2} & 0_{2 \times 2} \end{bmatrix}. \quad (27)$$

617 We can test that in absence of deadline misses, when the controller is implemented with LET
 618 (i.e., with one-step delay), the system preserves stability. We now move on to investigate
 619 the stability property of the system when some deadlines are missed. Table 2 presents a
 620 summary of the results we obtained.

¹⁰The constants used for the calculations are: $R = 0.025$ [Ohm], $\omega_{el} = 6283.2$ [rad/s], $L_d = 0.0001$ [ht],
 $L_q = 0.00012$ [ht].

¹¹While Tustin's method introduces frequency distortion, the method is what is currently applied in our
 industrial case study. Similar results can be obtained with the exact matrix exponential, changing the
 discretisation command parameters in the Matlab code.

■ **Table 2** Stability Results for the Electric Steering Application

	Misses	Stability	Lower Bound	Upper Bound
<i>Zero&Kill</i>	∞	✓	0.997713	0.997713
<i>Zero&Skip-Next</i>	1	✓	0.892575	0.892575
	2	✓	0.892575	0.892575
	3	✓	0.892575	0.892575
	4	✓	0.892575	0.892575
	5	✓	0.892575	0.892575
	6	✓	0.892575	0.892575
	7	✓	0.892575	0.892575
	8	✓	0.900922	0.900922
	9	✓	0.912902	0.912902
	10	✓	0.938565	0.938565
	11	✓	0.940823	0.940823
	12	✓	0.942610	0.942610
	13	✓	0.951092	0.951092
	14	✓	0.962846	0.962846
	15	✓	0.973776	0.973776
	16	✓	0.983954	0.983954
	17	✓	0.993436	0.993436
	18	✗	1.002273	1.002273
<i>Zero&Queue(1)</i>	1	✓	0.925966	0.925966
	2	✗	1.001620	1.001620
<i>Hold&Kill</i>	1	✓	0.892575	0.892575
	2	✓	0.938332	0.938332
	3	✗	1.073542	1.073542
<i>Hold&Skip-Next</i>	1	✓	0.968574	0.968574
	2	✗	1.107390	1.107390
<i>Hold&Queue(1)</i>	1	✗	1.423968	1.423968

621 The zero&kill strategy presents a special property. Using this strategy, the closed-loop
622 system is always going to be stable, regardless of the number of deadlines that are missed.
623 In fact, using the joint spectral radius, it is possible to prove that the system that switches
624 between A_H and A_M is always stable, when the matrices are the ones identified in Section 4
625 for zero&kill. This special property comes from the fact that the open-loop system is stable
626 and control is applied only using fresh measurements (assuming that the kill procedure is able
627 to rollback the system to a clean state). The fact that A_M and A_H are stable individually
628 is not enough to guarantee stability, but all of their combinations prove to be contractions,
629 making the switching system stable as well. Notice that this is not generalisable, but only
630 due to the properties of the particular system we are controlling – in fact, this is not true
631 for the the second-order system example above. For this specific system, this tells us that
632 implementing a clean rollback is very beneficial, and goes a long way to ensure fault tolerance.
633 However, it is not always possible to implement a clean rollback with the given hardware
634 and software setup.

635 When zero is paired with skip-next, old measurements of the state are used when a
636 recovery hit happens. This means that the system can be unstable even though the behaviour

637 is similar to the zero&kill strategy one. In fact, differently from zero&kill, the system is
638 not able to tolerate an infinite number of misses. We then ask how many many consecutive
639 deadline misses the system experience without violating the stability property. As reported in
640 Table 2, both the lower-bound and the upper-bound on the joint spectral radius are less than
641 1 for a system that experiences up to 17 consecutive misses. However, with 18 consecutive
642 misses the closed-loop system becomes provably unstable (lower-bound above 1).

643 This result means that if the controller can miss 18 deadlines in a row, then the delay
644 introduced between the sensing and the actuation is harmful for the system and can cause
645 instability. In this case, there is at least one sequence of deadline misses (which satisfies the
646 condition that there cannot be more than 18 consecutive misses) that leads to instability.
647 In this case, it is certain that the closed-loop switching system is unstable. Notice that
648 the harmful sequence does not necessarily have to be a repetition of 18 misses and 1 hit,
649 but can be a combination of different terms (for example it could happen that due to the
650 time constant of the system missing 16 deadlines, hitting 2 of them, and then missing 18
651 would cause instability). For this particular case, however, it is simple enough to check that
652 the spectral radius of the closed-loop matrix with 18 misses and 1 hit is above 1 and this
653 immediately means that the sequence of 18 misses and 1 hit destabilises the system (although
654 it might not be the only one).

655 If we pair zero with queue(1), the results differ dramatically. The number of matrices in
656 the set Σ used to compute the joint spectral radius increases, and there is a combination of
657 events (specific number of misses, recovery, and recovery with immediate information) that
658 can lead to instability even when just 2 consecutive deadlines are missed. This immediately
659 tells us that using the queue(1) strategy is a bad idea for this system and should be avoided.

660 When the control signal is held, the kill strategy guarantees stability for 2 deadline misses,
661 while a third potential miss makes the system provably unstable. The skip-next strategy,
662 paired with hold, can tolerate one miss, but a second one makes the system unstable. The
663 queue(1) strategy cannot even tolerate a single miss.

664 From this experimental campaign, we can conclude that for this system kill is working
665 better than any other system-level strategy, and zero works better than hold in terms of
666 guaranteeing the system stability. We can also conclude that while queuing a task when a
667 previous one is executing could improve the control performance, the risk of harming the
668 system is much higher. We advocate that performing the analysis presented in this paper
669 can give important runtime information to determine how robust control systems are to
670 temporary faults and problems.

671 **6 Related Work**

672 Studying the non-ideal behaviour that emerges from the implementation and execution of
673 control systems is an important problem for practical applications of control. Control tasks
674 for example may have variable periods and may require to be executed with different rates
675 depending on the operating conditions [4]. Also, late information can affect the system's
676 performance [24, 30, 31, 37], especially for networked systems. These timing effects are usually
677 characterised as independent events with stochastic distributions [13], or using worst case
678 bounds [2, 15]. Control researchers proposed deadline-aware design methods to guarantee
679 stability [5, 29] and improve the control performance [38]. Sinopoli et al. [40] proposed
680 an optimal control design for networked system leveraging the probability of packet losses.
681 Lincoln and Cervin [28] proposed a design tool for optimal controllers in the presence of
682 probabilistic delays, that can be exploited for LET design setting the delay to one period.

683 In many circumstances, the control designer can usually trade off computing time and
 684 accuracy [39]. In some cases, an inaccurate and faster solution that can be executed at a
 685 faster rate is preferable to an accurate and precise solution that can only be executed at a
 686 slower rate [23]. However, it is extremely hard in practice to guarantee that the delays will
 687 *never* exceed the control period.

688 These types of systems have been studied both from the schedulability perspective [43] and
 689 using model checking [12]. The performance cost of deadline misses was investigated [34, 44],
 690 together with the role of the strategy used to handle the misses [33, 41, 42]. All these papers
 691 used the m - K model, starting from the assumption that windows of hits and misses have to
 692 be analysed in order to determine the behaviour of the system. We build on the previous
 693 literature to determine how the implementation is going to react to missed deadlines, both in
 694 terms of selection of the control signal [29] and in terms of management of the job that misses
 695 its deadline [33]. There are some important differences between [33] and the contribution
 696 of this paper: [33] presents a control design technique to guarantee probabilistic robustness
 697 to deadline misses, on the contrary we assume that the controller is already designed and
 698 executes without any change and we demonstrate an analysis method to guarantee exactly
 699 that the controller tolerates (in terms of stability) a maximum number of consecutive misses.

700 In this work we showed that it may not be necessary to study windows of hits and misses.
 701 We argue that before looking at the m - K model representation, which is harder to analyse,
 702 one should check the stability of the \overline{m} model. If the \overline{m} is stable, there is no need
 703 to include complex window-based analysis — it happens quite often that the information
 704 needed to study the stability of the closed-loop systems is already contained in the number
 705 of consecutive deadline misses.

706 Ghosh et. al. [16] studied how to design control systems in the presence of faults that cause
 707 them to miss at most n deadlines, providing a synthesis method to achieve fault-tolerance –
 708 without specifying how the system is going to react to the misses (e.g., kill or skip-next).
 709 Here we take a different perspective and want to validate the behaviour of a control system
 710 (in terms of stability) when deadlines are missed, including the deadline management strategy
 711 from a system and implementation perspective.

712 **7 Conclusion**

713 In this paper we revisited the weakly hard real-time model for control tasks. We formalised
 714 the problem of determining the stability of the closed-loop system in the presence of a given
 715 number of consecutive misses that the controller task can experience. With the number
 716 of consecutive deadline misses, we derived stability criteria for systems where the deadline
 717 miss is handled in different ways, both from the perspective of the control signal applied
 718 (either zeroing or holding the previous value of the control signal) and with respect to the
 719 management of the task that misses the deadline (that can be either killed or allowed to
 720 continue in the subsequent control period). We solved this problem using a mathematical
 721 tool called joint spectral radius, for the computation of which open-source toolboxes are
 722 available. We applied our analysis to two different examples: an unstable system and an
 723 industrial application for electric steering. In both cases, we showed the limitations of the
 724 controller implementations that miss a given number of deadlines.

725 In the future, we plan to look at applying the joint spectral radius analysis to systems
 726 specified using the m - K model. This is particularly challenging, because there is no direct
 727 mapping between the potential sequences of hits and misses and a set of matrices. Furthermore,
 728 we want to investigate the performance of controllers that experience deadline misses.

References

- 729 1 Leonie Ahrendts, Sophie Quinton, Thomas Boroske, and Rolf Ernst. Verifying Weakly-Hard
730 Real-Time Properties of Traffic Streams in Switched Networks. In *30th Euromicro Conference*
731 *on Real-Time Systems (ECRTS 2018)*, volume 106 of *Leibniz International Proceedings in*
732 *Informatics (LIPIcs)*, pages 15:1–15:22. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik,
733 2018. doi:10.4230/LIPIcs.ECRTS.2018.15.
- 734 2 Philip Axer, Maurice Sebastian, and Rolf Ernst. Probabilistic response time bound for CAN
735 messages with arbitrary deadlines. In *Design, Automation & Test in Europe Conference &*
736 *Exhibition (DATE), 2012*, pages 1114–1117. IEEE, 2012.
- 737 3 G. Bernat, A. Burns, and A. Liamosí. Weakly hard real-time systems. *IEEE Transactions on*
738 *Computers*, 50(4):308–321, April 2001. doi:10.1109/12.919277.
- 739 4 Alessandro Biondi, Marco Di Natale, Giorgio C. Buttazzo, and Paolo Pazzaglia. Selecting the
740 transition speeds of engine control tasks to optimize the performance. *ACM Transaction on*
741 *Cyber-Physical Systems*, 2(1):1:1–1:26, January 2018. doi:10.1145/3127022.
- 742 5 R. Blind and F. Allgöwer. Towards networked control systems with guaranteed stability:
743 Using weakly hard real-time constraints to model the loss process. In *54th IEEE Conference*
744 *on Decision and Control (CDC)*, pages 7510–7515, December 2015. doi:10.1109/CDC.2015.
745 7403405.
- 746 6 V. Blondel and Y. Nesterov. Computationally efficient approximations of the joint spectral
747 radius. *SIAM Journal on Matrix Analysis and Applications*, 27(1):256–272, 2005. doi:
748 10.1137/040607009.
- 749 7 Vincent Blondel, Yurii Nesterov, and Jacques Theys. On the accuracy of the ellipsoid norm
750 approximation of the joint spectral radius. *Linear Algebra and its Applications*, 394:91–107,
751 2005. doi:https://doi.org/10.1016/j.laa.2004.06.024.
- 752 8 Vincent Blondel and John N. Tsitsiklis. The boundedness of all products of a pair of matrices
753 is undecidable. *Systems & Control Letters*, 41(2):135–140, 2000. doi:https://doi.org/10.
754 1016/S0167-6911(00)00049-9.
- 755 9 Tobias Bund and Frank Slomka. Controller/platform co-design of networked control systems
756 based on density functions. In *Proceedings of the 4th ACM SIGBED International Workshop*
757 *on Design, Modeling, and Evaluation of Cyber-Physical Systems, CyPhy '14*, pages 11–14.
758 ACM, 2014. doi:10.1145/2593458.2593467.
- 759 10 Tobias Bund and Frank Slomka. Worst-case performance validation of safety-critical control
760 systems with dropped samples. In *Proceedings of the 23rd International Conference on Real*
761 *Time and Networks Systems, RTNS '15*, pages 319–326. ACM, 2015. doi:10.1145/2834848.
762 2834860.
- 763 11 Lane Desborough. Increasing customer value of industrial control performance monitoring-
764 honeywell's experience. *Preprints of CPC*, pages 153–186, 2001.
- 765 12 G. Frehse, A. Hamann, S. Quinton, and M. Woehrle. Formal analysis of timing effects on
766 closed-loop properties of control software. In *2014 IEEE Real-Time Systems Symposium*, pages
767 53–62, December 2014. doi:10.1109/RTSS.2014.28.
- 768 13 Maximilian Gaukler, Andreas Michalka, Peter Ulbrich, and Tobias Klaus. A new perspective
769 on quality evaluation for control systems with stochastic timing. In *Proceedings of the 21st*
770 *International Conference on Hybrid Systems: Computation and Control (Part of CPS Week)*,
771 HSCC, pages 91–100. ACM, 2018. doi:10.1145/3178126.3178134.
- 772 14 Maximilian Gaukler, Tim Rheinfels, Peter Ulbrich, and Günter Roppenecker. Convergence
773 rate abstractions for weakly-hard real-time control. *arXiv preprint arXiv:1912.09871*, 2019.
- 774 15 Maximilian Gaukler and Peter Ulbrich. Worst-case analysis of digital control loops with
775 uncertain input/output timing. In Goran Frehse and Matthias Althoff, editors, *ARCH19. 6th*
776 *International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 61
777 of *EPiC Series in Computing*, pages 183–200. EasyChair, 2019. doi:10.29007/c4z1.
- 778 16 Saurav Kumar Ghosh, Soumyajit Dey, Dip Goswami, Daniel Mueller-Gritschneider, and
779 Samarjit Chakraborty. Design and validation of fault-tolerant embedded controllers. In
780

- 781 *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1283–1288,
782 2018. doi:10.23919/DATE.2018.8342212.
- 783 17 N. Guglielmi, F. Wirth, and M. Zennaro. Complex polytope extremality results for families
784 of matrices. *SIAM Journal on Matrix Analysis and Applications*, 27(3):721–743, 2005. doi:
785 10.1137/040606818.
- 786 18 M. Hamdaoui and P. Ramanathan. A dynamic priority assignment technique for streams with
787 (m, k)-firm deadlines. *IEEE Transactions on Computers*, 44(12):1443–1451, December 1995.
788 doi:10.1109/12.477249.
- 789 19 Z. A. H. Hammadeh, R. Ernst, S. Quinton, R. Henia, and L. Rioux. Bounding deadline
790 misses in weakly-hard real-time systems with task dependencies. In *Design, Automation*
791 *Test in Europe Conference Exhibition (DATE), 2017*, pages 584–589, March 2017. doi:
792 10.23919/DATE.2017.7927054.
- 793 20 Zain A. H. Hammadeh, Sophie Quinton, Marco Panunzio, Rafik Henia, Laurent Rioux,
794 and Rolf Ernst. Budgeting Under-Specified Tasks for Weakly-Hard Real-Time Systems. In
795 *29th Euromicro Conference on Real-Time Systems (ECRTS 2017)*, volume 76 of *Leibniz*
796 *International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:22. Schloss Dagstuhl–Leibniz-
797 Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ECRTS.2017.17.
- 798 21 R. Jungers. *The Joint Spectral Radius: Theory and Applications*. Lecture Notes in Control
799 and Information Sciences. Springer Berlin Heidelberg, 2009.
- 800 22 Raphael Jungers. JSR Toolbox. [https://www.mathworks.com/matlabcentral/
801 fileexchange/33202-the-jsr-toolbox](https://www.mathworks.com/matlabcentral/fileexchange/33202-the-jsr-toolbox), accessed October 23, 2019.
- 802 23 E. C. Kerrigan, G. A. Constantinides, A. Suardi, A. Picciau, and B. Khusainov. Computer
803 architectures to close the loop in real-time optimization. In *2015 54th IEEE Conference on*
804 *Decision and Control (CDC)*, pages 4597–4611, Dec 2015. doi:10.1109/CDC.2015.7402937.
- 805 24 Antzela Kosta, Nikolaos Pappas, Anthony Ephremides, and Vangelis Angelakis. Age and value
806 of information: Non-linear age case. In *Information Theory (ISIT), 2017 IEEE International*
807 *Symposium on*, pages 326–330. IEEE, 2017.
- 808 25 Huibert Kwakernaak. *Linear Optimal Control Systems*. John Wiley & Sons, Inc., New York,
809 NY, USA, 1972.
- 810 26 W.S. Levine. *The Control Handbook*. Electrical Engineering Handbook. Taylor & Francis,
811 1996.
- 812 27 D. Liberzon. *Switching in Systems and Control*. Systems & Control: Foundations & Applica-
813 tions. Birkhäuser Boston, 2003.
- 814 28 B. Lincoln and A. Cervin. JITTERBUG: a tool for analysis of real-time control performance.
815 In *41st IEEE Conference on Decision and Control*, volume 2, pages 1319–1324, December
816 2002. doi:10.1109/CDC.2002.1184698.
- 817 29 S. Linsensmayer and F. Allgower. Stabilization of networked control systems with weakly hard
818 real-time dropout description. In *IEEE 56th Annual Conference on Decision and Control*
819 *(CDC)*, pages 4765–4770, December 2017. doi:10.1109/CDC.2017.8264364.
- 820 30 L. Palopoli, L. Abeni, G. Buttazzo, F. Conticelli, and M. Di Natale. Real-time control system
821 analysis: an integrated approach. In *Real-Time Systems Symposium, 2000. Proceedings. The*
822 *21st IEEE*, pages 131–140, 2000. doi:10.1109/REAL.2000.896003.
- 823 31 P. Park, S. Coleri Ergen, C. Fischione, C. Lu, and K. H. Johansson. Wireless network design
824 for control systems: A survey. *IEEE Communications Surveys Tutorials*, 20(2):978–1013, 2018.
825 doi:10.1109/COMST.2017.2780114.
- 826 32 Pablo A. Parrilo and Ali Jadbabaie. Approximation of the joint spectral radius using sum
827 of squares. *Linear Algebra and its Applications*, 428(10):2385–2402, 2008. doi:[https://doi.
828 org/10.1016/j.laa.2007.12.027](https://doi.org/10.1016/j.laa.2007.12.027).
- 829 33 Paolo Pazzaglia, Claudio Mandrioli, Martina Maggio, and Anton Cervin. DMAC: Deadline-
830 Miss-Aware Control. In *31st Euromicro Conference on Real-Time Systems (ECRTS 2019)*,
831 volume 133 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:24, 2019.
832 doi:10.4230/LIPIcs.ECRTS.2019.1.

- 833 **34** Paolo Pazzaglia, Luigi Pannocchi, Alessandro Biondi, and Marco Di Natale. Beyond the Weakly
834 Hard Model: Measuring the Performance Cost of Deadline Misses. In *30th Euromicro Confer-*
835 *ence on Real-Time Systems (ECRTS 2018)*, volume 106 of *Leibniz International Proceedings*
836 *in Informatics (LIPIcs)*, pages 10:1–10:22, 2018. doi:10.4230/LIPIcs.ECRTS.2018.10.
- 837 **35** Gang Quan and Xiaobo Hu. Enhanced fixed-priority scheduling with (m,k)-firm guarantee.
838 In *Proceedings of the 21st IEEE Conference on Real-time Systems Symposium, RTSS*, pages
839 79–88. IEEE Computer Society, 2000.
- 840 **36** Gian-Carlo Rota and W. Gilbert Strang. A note on the joint spectral radius. *Indagationes*
841 *Mathematicae*, 63:379–381, 1960. doi:https://doi.org/10.1016/S1385-7258(60)50046-1.
- 842 **37** Abusayeed Saifullah, Chengjie Wu, Paras Babu Tiwari, You Xu, Yong Fu, Chenyang Lu, and
843 Yixin Chen. Near optimal rate selection for wireless control systems. *ACM Transactions on*
844 *Embedded Computing Systems*, 13(4s):128:1–128:25, April 2014. doi:10.1145/2584652.
- 845 **38** Luca Schenato, Bruno Sinopoli, Massimo Franceschetti, Kameshwar Poolla, and S Shankar
846 Sastry. Foundations of control and estimation over lossy networks. *Proceedings of the IEEE*,
847 95(1):163–187, 2007.
- 848 **39** Amir Shahzad, Eric C. Kerrigan, and George A. Constantinides. A stable and efficient method
849 for solving a convex quadratic program with application to optimal control. *SIAM Journal*
850 *on Optimization*, 22(4):1369–1393, 2012. URL: <http://dx.doi.org/10.1137/11082960X>, doi:
851 10.1137/11082960X.
- 852 **40** Bruno Sinopoli, Luca Schenato, Massimo Franceschetti, Kameshwar Poolla, and Shankar
853 Sastry. An lqg optimal linear controller for control systems with packet losses. In *Decision and*
854 *Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference*
855 *on*, pages 458–463. IEEE, 2005.
- 856 **41** D. Soudbakhsh, L. T. X. Phan, A. M. Annaswamy, and O. Sokolsky. Co-design of arbitrated
857 network control systems with overrun strategies. *IEEE Transactions on Control of Network*
858 *Systems*, 5(1):128–141, March 2018. doi:10.1109/TCNS.2016.2583064.
- 859 **42** Damoon Soudbakhsh, Linh T. X. Phan, Oleg Sokolsky, Insup Lee, and Anuradha Annaswamy.
860 Co-design of control and platform with dropped signals. In *Proceedings of the ACM/IEEE 4th*
861 *International Conference on Cyber-Physical Systems, ICCPS '13*, pages 129–140. ACM, 2013.
862 doi:10.1145/2502524.2502542.
- 863 **43** Youcheng Sun and Marco Di Natale. Weakly hard schedulability analysis for fixed priority
864 scheduling of periodic real-time tasks. *ACM Transactions on Embedded Computing Systems*,
865 16(5s):171:1–171:19, September 2017. doi:10.1145/3126497.
- 866 **44** E. P. van Horssen, A. R. B. Behrouzian, D. Goswami, D. Antunes, T. Basten, and W. P.
867 M. H. Heemels. Performance analysis and controller improvement for linear systems with (m,
868 k)-firm data losses. In *2016 European Control Conference (ECC)*, pages 2571–2577, June 2016.
869 doi:10.1109/ECC.2016.7810677.
- 870 **45** Guillaume Vankeerberghen, Julien Hendrickx, and Raphaël M. Jungers. JSR: a toolbox
871 to compute the joint spectral radius. In *Proceedings of the 17th International Conference*
872 *on Hybrid Systems: Computation and Control, HSCC*, pages 151–156. ACM, 2014. doi:
873 10.1145/2562059.2562124.
- 874 **46** Dirk Ziegenbein and Arne Hamann. Timing-aware control software design for automotive
875 systems. In *Proceedings of the 52Nd Annual Design Automation Conference, DAC '15*, pages
876 56:1–56:6, 2015. doi:10.1145/2744769.2747947.