Jitter-Robust LQG Control and Real-Time Scheduling Co-Design

Xu, Yang; Cervin, Anton; Årzén, Karl-Erik

Published in:
Proceedings of the American Control Conference

DOI:
10.23919/ACC.2018.8430953

2018

Document Version:
Peer reviewed version (aka post-print)

Jitter-Robust LQG Control and Real-Time Scheduling Co-Design*

Yang Xu, Anton Cervin and Karl-Erik Årzén

Abstract— **In real-time control systems, varying task response times may lead to delays and jitter in the delays in the feedback control loops, which adversely affects both performance and robustness. Standard LQG control design does not give any guarantees on robustness, while robust control design methods often do not handle controller timing uncertainty. We propose a sampled-data controller synthesis method that minimizes an LQG cost function subject to a jitter margin constraint. By robustifying the LQG controller we are able to retain good stability margins under delay and jitter, while typically paying a small price in terms of nominal performance. We also present a co-design procedure that assigns optimal priorities and sampling periods to a set of controllers based on their performance characteristics and jitter sensitivity. The procedure is evaluated on randomized plant sets, showing an improvement over state-of-the-art methods.**

I. INTRODUCTION

A. Motivation

We explore the problem of scheduling and control co-design in real-time systems. In a real-time control system, for cost-saving reasons, multiple controllers are commonly implemented on a single CPU. In order to have good performance in a digital control systems, the sampling period should be sufficiently short. Although a short sampling period improves the performance of the current task, it requires more CPU resources, and hence potentially degrades the performance of the other tasks. It means that not only the controller, but also the scheduling, affect the closed-loop control performance.

Among many synthesis methods, linear-quadratic-Gaussian (LQG) control is a fundamental optimal control method that combines a Kalman filter with a linear-quadratic feedback regulator. The performance metric in LQG is defined as an average quadratic running cost. It is well known that good LQG performance does not mean that the system is robust [1]. While there are several methods proposed to improve the robustness of LQG (e.g., [2], [3]), very few have focused on uncertainties related to controller timing [4]. This is the reason for exploring LQG control synthesis with a jitter robustness constraint.

In this paper, we present an approach to jitter-robust LQG control synthesis together with priority assignment and period selection methods for the real-time system. The contributions in this paper are:

- We propose a convex optimization-based sampled-data LQG control design method with a jitter robustness constraint. The constraint guarantees that the closed-loop system has a specified *jitter margin* [5], [6].
- We propose a new rule of thumb for initial sampling period selection based on the jitter margin.
- We give a co-design method to assign priorities and sampling periods to obtain optimal robust LQG performance for a set of controllers that share a single CPU.

*This work was supported in part by the LCCC Linnaeus Center and the ELLIIT Excellence Center at Lund University.

The authors are with the Department of Automatic Control, Lund University, Box 118, SE-221 00 Lund, Sweden. Email: {yang, anton, karlerik}@control.lth.se

B. Related Work

A real-time task period selection problem to optimize a system-wide control performance measure under fixed-priority scheduling was initially raised and solved in [7]. The authors assumed, however, that the control performance only depends on the sampling period. [8] analyzed how the LQG cost depends on the sampling period and proposed an on-line adjustment algorithm. In [9], the period assignment problem was solved assuming that the control delay is constant using a formula for the approximate response time. In reality, the response time in the real-time system is not constant, so in the optimization procedure in [10], the delay distribution was used to design stochastic LQG controllers and assign periods to all tasks. The priority assignment problem for multiple (control) tasks has been studied in [11].

The stochastic LQG control with known delay probability distribution was originally solved in [12]. In [13], the periods were perturbed to achieve finite hyperperiod and a periodic LQG control design method that involved the solution of the periodic Riccati equation was proposed. Control system stability under in the presence of real-time system induced jitter was discussed in [14] and [15]. In [14], an integrated design optimization method was proposed for real-time control systems with requirements on robustness and optimized LQG control performance. The jitter margin was used to measure the worst-case control performance. A standard LQG controller was used and the jitter margin constraint was satisfied by designing the scheduling parameters, In the work in this paper, instead, the design of the jitter-robust LQG controller takes the jitter margin constraint into account, and then the scheduling is optimized.

Mixed H_2/H_∞ control design can be used to achieve robust LQG-like control, e.g., [16], [17], but the feedback system is assumed to be linear and time-invariant, i.e., there is no specific analysis for timing uncertainty. H_2 optimal control design with an H_∞ constraint can be reformulated to a convex optimization problem through the Youla parameterization [18]. Using this parameterization, both the H_2 cost and the H_∞ constraint are convex.

C. Outline of This Paper

In Section II, we give the real-time system model and control system model, including the performance metric we use. In Section III, a new rule of thumb for initial sampling period assignment is proposed. The robust LQG control design problem and its solution are described in Section IV. In Section V, we give a jitter-aware priority and period assignment co-design method to optimize the overall system performance. In Section VI, an example is presented to show the entire procedure of scheduling and control co-design. In the same section, the method is evaluated on randomized plant sets, showing an improvement over state-of-the-art methods. Conclusions are given in Section VII.

II. SYSTEM MODEL

A. Real-Time System Model

A real-time system of n tasks running on a single CPU is defined by the following parameters for each task τ_i:

- The period, T_i, is the constant time between two consecutive releases of the task.
- The execution time, E_i, is the time that the CPU needs to run the task. Since the code for the LQG control algorithm is normally quite limited in size without branches and with small memory usage, here it is assumed that the execution time is constant.
- The utilization, U_i, is defined as the fraction of the CPU used by the task; $U_i = E_i/T_i$.
- The worst-case and best-case response times, R_i^w and R_i^b, are the maximum and minimum times, respectively, between the task release and task finishing times.
- The output jitter, J_i, is the difference between the worst-case and best-case response times; $J_i = R_i^w - R_i^b$.

Throughout this paper, preemptive fixed-priority scheduling is used. Unless otherwise stated, we assume that task τ_i has higher priority than τ_j if $i < j$.

B. Control System Model

Each task τ_i is controlling a continuous linear time-invariant single-input–single-output plant i with state-space realization (for ease of notation we drop the plant index i)

$$\dot{x}(t) = Ax(t) + Bu(t) + v_1(t),$$
$$y(t) = Cx(t) + v_2(t),$$

where v_1 and v_2 are uncorrelated white noise processes with intensities R_1 and R_2. A time-invariant LQG controller should be designed to minimize the quadratic cost function

$$V = \mathrm{E}\, z^2 = \mathrm{E}\left(x^T Q_1 x + Q_2 u^2\right), \tag{1}$$

where Q_1 and Q_2 are weighting matrices.

The sampling period of the controller is $h = T_i$. The sampling is performed at the task release time, e.g., using external hardware. Hence, there is no sampling (or input) jitter. The actuation is performed when the task finishes. Hence, the actuation (or output) is subject to a time-varying delay $\delta(t)$, $E_i \le \delta(t) \le E_i + J_i$, due to the task output jitter. The control loop is shown in Fig. 1. For a given constant delay or known delay distribution (assuming independent delays between periods), the LQG controller and its cost (1) can be designed and evaluated using, e.g., the Jitterbug toolbox [19].

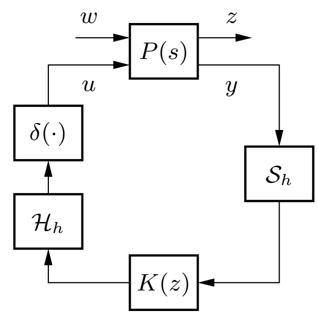

Fig. 1. Control loop with continuous plant $P(s)$, discrete LQG controller $K(z)$, periodic sampler $\mathcal{S}_h$ and hold $\mathcal{H}_h$, and time-varying delay δ.

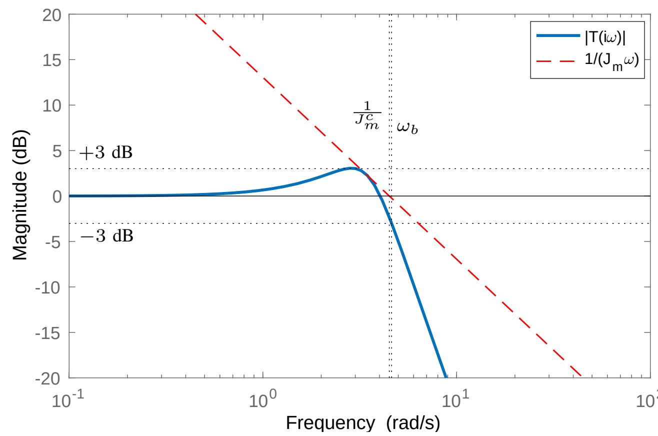

Fig. 2. The continuous-time jitter margin J_m^c is given by the intersection of the closed-loop transfer function $|T(i\omega)|$ and the line $1/(J_m^c\omega)$. For a typical robust design, the 3 dB bandwidth ω_b is close to $1/J_m^c$.

As first shown in [1], an LQG controller has no guaranteed robustness. In order to gain robustness of the real-time control system, we would like to design a controller that has a specified *jitter margin* J_m [5], [6]. This means that the closed-loop system should be stable for any time-varying delays $\delta \in [E_i, E_i + J_m]$. Naturally, we will require that $J_m > J_i$. For a continuous-time control system, the sufficient condition for stability of the closed-loop system is

$$|T(i\omega)| = \left|\frac{P(i\omega)K(i\omega)}{1 + P(i\omega)K(i\omega)}\right| < \frac{1}{J_m\omega}, \quad \forall\omega, \tag{2}$$

where $P(s)$ is the plant and $K(s)$ is the controller [5]. In a Bode plot, this corresponds to the magnitude curve of the complementary sensitivity function $T(s)$ lying below a line with slope -1 and gain $1/J_m^c$ (see Fig. 2). For a sampled-data system, the stability criterion is slightly more complicated (see Section IV below).

C. Co-Design Problem

The goal of the control and scheduling co-design is to optimize the overall control performance, subject to the real-time system utilization constraint. This can be expressed as

$$\text{minimize } V = \sum_i V_i, \quad \text{s.t. } \sum_i U_i \le 1, \tag{3}$$

where the cost V_i for each controller is defined in (1). Moreover, we would like to give guarantees on the robustness for each control loop. The parameters that we can optimize over are the task priorities, the sampling periods, and the controllers themselves.

In order to achieve a small LQG cost and good robustness, the sampling period and the jitter should be small, but this requirement cannot be satisfied simultaneously for every task due to the fixed-priority scheduling algorithm used. This makes the co-design problem nontrivial.

III. Initial Sampling Period Selection

Sampling period selection is typically done based on the properties of the closed-loop system. The rate should be fast enough, so that disturbance rejection and robustness are not affected too much by the sampling and hold operations, while slow enough to avoid numerical problems and allow implementation in resource-constrained systems.

There are several rules of thumb that relate the sampling rate with the speed of the closed-loop system. Franklin *et al.* recommend sampling about 10–40 times faster than the bandwidth [20]. Åström and Wittermark recommend 4 to 10 samples per rise-time of the closed-loop system [21]. These recommendations (and other similar ones [22]) roughly translate into the relation

$$0.15 \le \omega_b h \le 0.6, \tag{4}$$

where ω_b is the 3 dB closed-loop bandwidth and h is the sampling interval.

Basing the sampling period selection on a single point ω_b of the closed-loop frequency response however makes this rule of thumb sensitive to degenerate cases. As an alternative, we propose a new rule of thumb that is based on the continuous-time jitter margin J_m^c of the control system:

$$0.15 \le h/J_m^c \le 0.6. \tag{5}$$

For a typical, robust closed-loop system with a maximum complementary sensitivity of 3 dB, we have $\omega_b \approx 1/J_m^c$, and the new rule produces similar sampling intervals as the old rule (4). This nominal case is illustrated in Fig. 2.

For degenerate cases, however, the new rule will typically recommend shorter sampling intervals than the old rule. Three such cases are illustrated in Fig. 3. For closed-loop system T_1, the maximum complementary sensitivity is close to 15 dB, which implies that the system is not robust. For system T_2, the 3 dB bandwidth is quite low, while $|T(i\omega)|$ does not roll off until much later. For system T_3 the closed-loop gain is low throughout and the bandwidth is not well-defined. As seen in the figure, all of these systems actually have the same jitter margin, and the new rule will hence recommend the same sampling interval for all of them.

The new rule aims for a sampled closed loop that should be robust towards delay and jitter amounting to more than

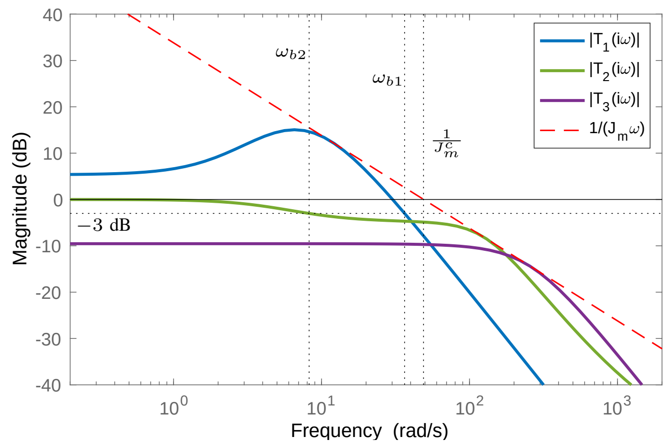

Fig. 3. In degenerate cases, the 3 dB bandwidth ω_b may be significantly smaller than $1/J_m^c$ (systems T_1, T_2) or even undefined (system T_3).

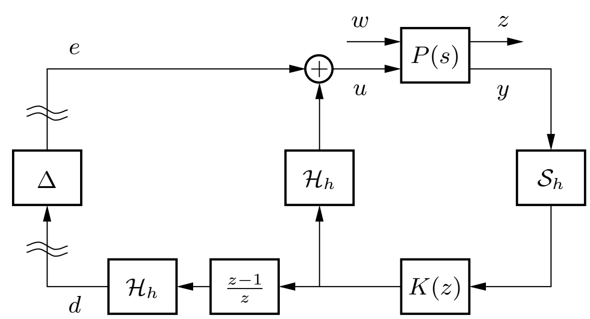

Fig. 4. Transformation of control loop with time-varying output delay. The jitter is represented by the uncertainty Δ.

one and a half sampling interval in total. This aligns with the worst-case situation in typical multirate applications: the sample-and-hold operation can be approximated by a delay of $h/2$, and the output jitter is typically upper bounded by h. Using the rule does not give any hard stability guarantees, however, since the exact performance degradation due to sampling cannot be captured using simple expressions. Once a discrete-time controller has been designed, its jitter margin should be verified using the sampled-data analysis in [5].

IV. Jitter-Robust LQG Control Synthesis

A. Robust LQG Problem Formulation

In order to design a LQG controller with guaranteed jitter robustness, we use the stability criterion in [5] as the constraint. As shown in [5] the sampled-data control loop with output jitter in Fig. 1 can be transformed into the block diagram in Fig. 4. The time-varying operator Δ represents the uncertainty due to jitter and has the worst-case gain

$$\|\Delta\| = \sqrt{(2\lfloor N\rfloor + 1)\,N - \lfloor N\rfloor^2 - \lfloor N\rfloor},$$

where $N = J_i/h$.

Referring to Fig. 4, the jitter-robust control design problem can now be stated as the optimization problem

$$\min_{K(z)} \ \|G_{zw}\|_2^2, \qquad \text{s.t. } \|G_{de}\|_\infty < b, \tag{6}$$

where

$$b = \frac{1}{\sqrt{(2\lfloor N_m \rfloor + 1)\, N_m - \lfloor N_m \rfloor^2 - \lfloor N_m \rfloor}},$$
$$N_m = J_m^{req}/h,$$

and where J_m^{req} is the required jitter margin. The objective function in the optimization problem, namely the square of the H_2 norm of G_{zw}, is equal to the LQG cost (1).

Since both the criterion and the constraint in (6) are nonconvex, it is hard to solve the problem directly. Therefore, we reformulate it as a convex optimization problem below.

B. *Youla Parameterization and Convex Optimization*

Using the Youla parameterization [18], the optimization problem (6) can be reformulated as

$$\begin{aligned} \min_{\Theta} \; & \int_{-2\pi}^{2\pi} \left| P_{zw}(e^{i\omega}) - P_{zu}(e^{i\omega})\Theta(e^{i\omega})P_{yw}(e^{i\omega}) \right|^2 d\omega \\ \text{s.t.} \; & \left\| \frac{e^{i\omega}-1}{e^{i\omega}} P_{yu}\left(e^{i\omega}\right)\Theta\left(e^{i\omega}\right) \right\|_\infty < b, \end{aligned} \tag{7}$$

where the arbitrary stable finite order LTI Θ is defined as

$$\Theta(e^{i\omega}) = \frac{K(e^{i\omega})}{1 + P_{yu}(e^{i\omega})K(e^{i\omega})}.$$

For the sampled plant $P_d(z)$ with noise covariance matrices R_{1d} and R_{2d}, and weighting matrices Q_{1d} and Q_{2d}, the transfer function matrix used in the optimization (7) is

$$\begin{aligned} &\left[\begin{array}{c|c} P_{zw}(z) & P_{yw}(z) \\ \hline P_{zu}(z) & P_{yu}(z) \end{array}\right] \\ &= \left[\begin{array}{cc|c} \sqrt{Q_{1d}}P_d(z)\sqrt{R_{1d}} & 0 & \sqrt{Q_{1d}}P_d(z) \\ 0 & 0 & \sqrt{Q_2} \\ \hline P_d(z)\sqrt{R_{1d}} & \sqrt{R_{2d}} & P_d(z) \end{array}\right]. \end{aligned} \tag{8}$$

The optimization problem can be solved numerically using, e.g., the CVX toolbox [23]. Here we choose a pulse response representation of the Youla parameter,

$$\Theta(z) = \sum_{i=0}^{n-1} \frac{\theta_i}{z^k},$$

with $\{\theta_i\}$ being the set of scalar optimization variables. The magnitude constraint is checked over a dense grid of frequency points. Once the problem is solved for $\Theta(z)$, the corresponding controller can be calculated by

$$K(z) = \frac{\Theta(z)}{1 - \Theta(z)P_{yu}(z)}.$$

C. *Example*

Consider control of an inverted pendulum with realization

$$A = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

The continuous-time cost and noise matrices are

$$Q_1 = \begin{bmatrix} 100 & 0 \\ 0 & 0 \end{bmatrix},\; Q_2 = 1,\; R_1 = \begin{bmatrix} 0 & 0 \\ 0 & 100 \end{bmatrix},\; R_2 = 1.$$

A standard LQG design gives the continuous-time jitter margin $J_m^c = 0.195$. The recommended sampling period

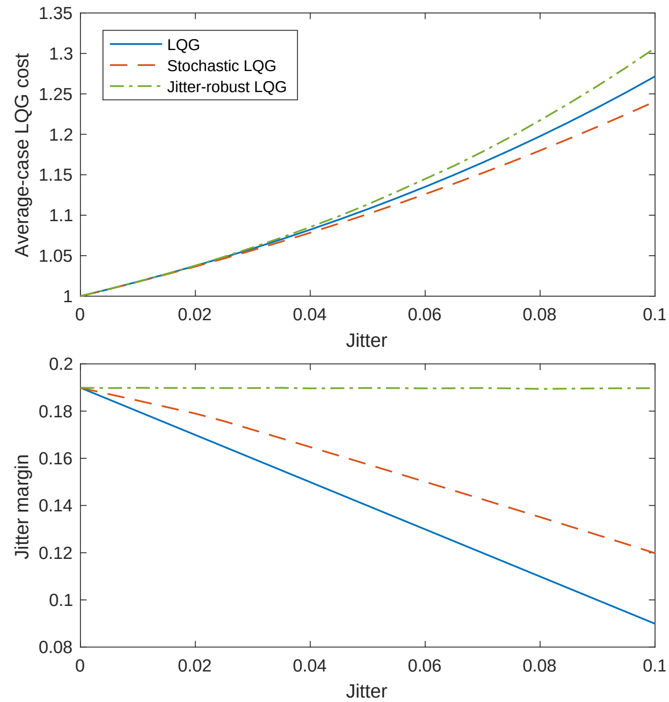

Fig. 5. Comparison of design methods for the inverted pedulum: average-case normalized performance (top) and remaining jitter margin (bottom) vs amount of jitter in the control loop.

range is hence $0.03 < h < 0.12$, and we choose $h = 0.1$. The minimum delay is assumed to be zero, and the output delay is assumed to be independent between periods and uniformly distributed between 0 and J. For zero delay and jitter, the sampled-data jitter margin is $J_m^0 = 0.190$.

For different values of the jitter, $0 \le J \le h$, three different designs are compared:

- Plain LQG design, assuming zero delay and jitter.
- Stochastic LQG design, with perfect knowledge of the delay distribution. This is the optimal design with regards to the LQG cost.
- Jitter-robust LQG design, with the constraint of keeping the remaining jitter margin at the original value J_m^0. This is achieved by setting $J_m^{req} = J_m^0 + J$.

The expected LQG cost under uniform jitter is calculated using Jitterbug [19] and the remaining jitter margin is calculated using [5]. The results, normalized to 1 for the case of zero jitter, are given in Fig. 5. It is seen that the stochastic LQG performs best in terms of average-case performance (it should since it is an optimal design), while its jitter margin degrades as the jitter increases. The situation is worse for the plain LQG controller, which suffers from both performance deterioration and decreasing jitter margin. The jitter-robust LQG is able to keep the jitter margin at its original value (0.190) while paying only a small price in terms of performance degradation.

V. REAL-TIME SYSTEM SCHEDULING CO-DESIGN

We now turn to the problem of implementing several digital controllers on the same CPU. Using the procedure described above, we can design a robust controller with a specified jitter margin for a given sampling period, but it remains to optimize the combined performance of a set of controllers by assigning suitable periods and priorities.

A. Affine Cost Function Approximation

We begin by characterizing the jitter-robust LQG cost V_i as a function of the sampling period and the jitter. We first calculate the initial sampled-data jitter margin $J^0_{m,i}$ assuming a constant delay E_i and zero jitter. For a pair (h_i, J_i) we then design a jitter-robust LQG controller as described in Section IV with the required jitter margin $J^{req}_{m,i} = J^0_{m,i} + J_i$. Solving the convex optimization problem (7) for a given period h_i and jitter J_i, we obtain one data point of the cost function $V_i = f_i(h_i, J_i)$. We store the result of each evaluation as a data point $p_j = (T, J, V)$ $j \in [1, 2, ..., m]$, with m being the number of points.

In order to facilitate an analytical solution for the optimal periods below, we approximate the cost V_i as an affine function of sampling period T_i and jitter J_i, namely,

$$V_i = a_i h_i + b_i J_i + c_i. \tag{9}$$

We want the square sum of the orthogonal distances between the affine approximation and the points as small as possible. Let c be a point on the plane, and n be a unit normal vector to the plane. The optimization problem is

$$\min_{c,n} \sum_{j=1}^{m} \left((p_j - c)^T n \right)^2, \quad \text{s.t. } \|n\| = 1.$$

This plane fitting problem can be efficiently solved by the singular value decomposition (SVD) method [24]. The optimization solution contains two parts, c and n. The point on the plane c can be calculated by $c = 1/m \sum_{j=1}^{m} p_j$. Let $A = [p_1 - c, p_2 - c, ..., p_m - c]$. The unit normal vector n is the last column of U, where $A = USV^T$. Having obtained c and n, the equation of the plane is

$$\left(\begin{pmatrix} h_i & J_i & V_i \end{pmatrix}^T - c \right)^T n = 0,$$

from which we obtain the coefficients in (9).

B. Period Assignment

For a given task priority ordering, we now derive a solution to the optimal period assignment problem. To facilitate an analytical solution, we use simple bounds on the best-case and worst-case response times and conservatively overestimate the jitter.

The best-case response time of task i is trivially lower bounded by $R^b_i \geq E_i$. For the worst-case response time, we use the simplistic upper bound (see [25])

$$R^w_i \leq \frac{\sum_{j=1}^{i} E_j}{1 - \sum_{j=1}^{i-1} \frac{E_j}{T_j}}, \tag{10}$$

where $j < i$ indicates that task j has higher priority than task i. It then follows that the jitter is upper bounded by

$$J_i = R^w_i - R^b_i \leq \frac{\sum_{j=1}^{i} E_j}{1 - \sum_{j=1}^{i-1} \frac{E_j}{T_j}} - E_i \tag{11}$$

In order to obtain a solution that is guaranteed not to violate the jitter margin requirement for any controller, we assume that the upper bound in (11) can actually be reached during execution. Using the upper bound for the value of J_i and the relation $U_i = E_i/h_i$, the cost function (9) for task i can be restated as

$$V_i = \frac{a_i E_i}{U_i} + \frac{b_i \sum_{j=1}^{i} E_j}{1 - \sum_{j=1}^{i-1} U_j} - b_i E_i + c_i$$

The period assignment problem (3) can now be expressed in terms of task utilizations as

$$\min_{\{U_i\}} \sum_i V_i, \quad \text{s.t. } \sum_i U_i \leq 1. \tag{12}$$

A similar optimization problem of this form was solved in [9] and we reuse that solution here. Let

$$\alpha_i = a_i E_i, \quad \beta_i = b_i \sum_{j=1}^{i} E_j,$$

and recursively define μ_k and λ_k as

$$\begin{aligned} \mu_k &= \sqrt{\alpha_k}, \quad 1, 2, \cdots, n-1 \\ \lambda_{n-1} &= \sqrt{\alpha_n + \beta_n} \\ \lambda_{k-1} &= \sqrt{\beta_k + (\lambda_k + \mu_k)^2}, \quad k = 2, 3, \cdots, n-1. \end{aligned}$$

The optimal utilizations of each task are then given by

$$\begin{aligned} U_1 &= \frac{\mu_1}{\lambda_1 + \mu_1} \\ U_k &= U_1 \frac{\mu_k}{\mu_1} \prod_{j=1}^{k-1} \frac{\lambda_j}{\lambda_{j+1} + \mu_{j+1}}, \quad k = 2, 3, \cdots, n. \end{aligned}$$

Finally, the optimal periods are recovered as $h_i = E_i/U_i$. The solution always achieves full utilization ($\sum_i U_i = 1$). For details, see [9].

C. Priority Assignment

As discussed in [11], optimal priority assignment in real-time control systems is in general a combinatorial problem with exponential complexity in terms of the number of tasks. Assigning correct priorities is however crucial, since the amount of jitter (and hence also the performance degradation) depends heavily on the task priority (cf. Eq. (11)).

In this paper, we have used the following two methods to assign priorities to a set of controller tasks:

- **Global solution.** Using exhaustive search, we solve the optimal period assignment problem for all permutations of the task priorities and select the priority ordering that gives the smallest cost. Due to complexity, this method can only be used for small task sets in practice.
- **Heuristic solution.** We propose to order the tasks by the value of $E_i/\sqrt{b_i}$ in ascending order. The idea is to give high priority to tasks with short execution times and high jitter sensitivity. This method can be used for arbitrarily large task sets and only requires the period assignment problem to be solved once.

VI. EVALUATION

This section illustrates the jitter-robust LQG control and scheduling co-design procedure in a simple example and in a larger evaluation using randomly generated plants. In summary, the different steps of the co-design method are:

1) Initial sampling period selection. Using (5), initial sampling periods are calculated based on the continuous-time jitter margins. With these periods, the sampled-data closed-loop systems will have some basic robustness against scheduling-induced delay and jitter.
2) Characterization of the LQG cost. Designing a jitter-robust LQG controller and evaluating the resulting cost for a number of different values of the period and jitter, we obtain an affine cost function for each control loop.
3) Priority assignment and sampling period selection. A heuristic method and a global search method are used for the task priority ordering. For each priority assignment, optimal periods are calculated based on the affine cost functions. If any period is outside of the given range, it is clamped at the limit of the range.

A. A Simple Example

The simple example consists of three tasks sharing one CPU. Each task implements an LQG controller that controls one of the following open-loop unstable plants:

$$\begin{aligned} P_1(s) &= \frac{1}{(s-0.71)(s-0.40)}, \\ P_2(s) &= \frac{1}{(s-0.21)(s+0.08)}, \\ P_3(s) &= \frac{1}{(s-0.95)(s+0.76)}. \end{aligned} \tag{13}$$

For the LQG design, the three plants are realized in controllable canonical form. The cost weighting matrices and noise covariance matrices are chosen as as

$$\begin{aligned} Q_{1,i} &= \begin{bmatrix} 0 & 0 \\ 0 & \alpha_i \end{bmatrix}, \quad Q_{2,i} = 1, \\ R_{1,i} &= \begin{bmatrix} \beta_i & 0 \\ 0 & 0 \end{bmatrix}, \quad R_{2,i} = 0.01, \end{aligned} \tag{14}$$

where $\alpha_1 = 1.89$, $\alpha_2 = 72.38$, $\alpha_3 = 102.66$, $\beta_1 = 14.48$, $\beta_2 = 1818.80$, $\beta_3 = 17.86$.

For the given parameters, a standard continuous-time LQG controller $K_i(s)$ for each plant is designed, and the continuous-time jitter margin $J^c_{m,i}$ is calculated. Initial sampling periods are then chosen as

$$h_i^0 = 0.3 J^c_{m,i}, \tag{15}$$

i.e., the recommendation (5) is fulfilled. The initial utilizations U_i^0 are randomized using the UUniFast algorithm [26]. The execution times are then given by $E_i = U_i^0 h_i^0$. The initial jitter margin $J^0_{m,i}$ is calculated using the initial period h_i^0 and with E_i as the best-case response time. The initial periods, utilizations, execution times, and jitter margins are summarized in Table I.

To obtain cost function approximations, for each task τ_i we choose the sampling period h_i from 7 evenly spaced points between $0.5h_i^0$ and $2h_i^0$ and choose the jitter J_i from 9 evenly spaced points between 0 and $2h_i^0$. For each pair (h_i, J_i) that satisfies $R_i^b + J_i \le h_i$, we design a robust LQG controller with the required jitter margin $J_m^{req} = J^0_{m,i} + J_i$. The corresponding LQG cost is evaluated in Jitterbug, assuming a uniform response time distribution, unif $\left(R_i^b, R_i^b + J_i\right)$. The resulting costs functions $V_i(h_i, J_i)$ for $P_1(s)$, $P_2(s)$, $P_3(s)$ are plotted in Fig. 6, showing that affine approximations are reasonable. Using the SVD plane fitting method, the affine LQG cost functions are

TABLE I

INITIAL PARAMETERS IN THE SIMPLE EXAMPLE

	h_i^0	U_i^0	E_i	$J^0_{m,i}$
$i=1$	0.074	0.35	0.026	0.24
$i=2$	0.082	0.18	0.015	0.27
$i=3$	0.048	0.47	0.022	0.16

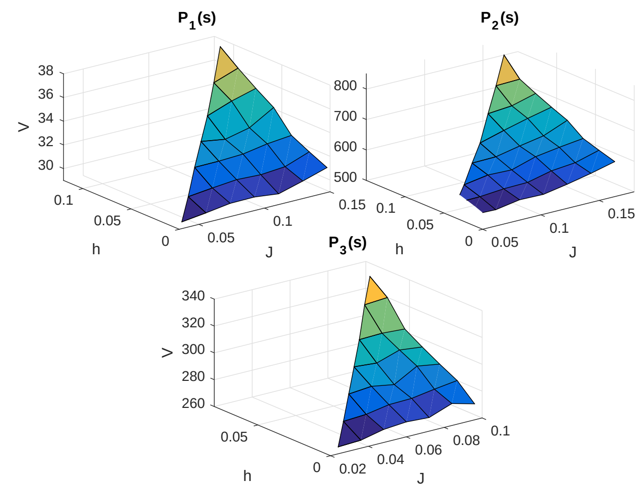

Fig. 6. Cost functions $V_i = f_i(h_i, J_i)$ for three example plants. It is seen that each V_i can be quite well approximated by an affine function.

$$\begin{aligned} V_1(h_1, J_1) &= (0.01h_1 + 0.06J_1 + 0.03) \times 10^3, \\ V_2(h_2, J_2) &= (0.57h_2 + 1.16J_2 + 0.51) \times 10^3, \\ V_3(h_3, J_3) &= (0.12h_3 + 0.80J_3 + 0.26) \times 10^3. \end{aligned} \tag{16}$$

As can be seen from the coefficients, the cost is more sensitive to the size of the jitter than to the value of the sampling period.

In order to optimize the overall cost, priorities and periods need to be assigned. We first enumerate all the permutations of priority assignment and calculate the corresponding cost $\sum_{i=1}^{3} V_i^0$ using the initial periods h_i^0. The results are shown in the first two columns of Table II, where the first column lists the tasks in descending priority order.

Next, for each priority assignment, the periods are optimized using the method in Section V-B. Any period that falls outside the range $\left(0.15J^0_{i,m}, 0.6J^0_{i,m}\right)$ is clamped at the limit, and then the optimization for the remaining periods is repeated. The optimal sampling periods h_1^*, h_2^*, h_3^* and the corresponding total cost $\sum_{i=1}^{3} V_i^*$ are shown in the four last columns of Table II.

In all six cases, the overall cost using optimal periods is

TABLE II

ENUMERATION OF ALL POSSIBLE PRIORITY ORDERINGS

	$\sum_{i=1}^{3} V_i^0$	h_1^*	h_2^*	h_3^*	$\sum_{i=1}^{3} V_i^*$
τ_1, τ_2, τ_3	1000.8	0.15	0.06	0.04	951.3
τ_1, τ_3, τ_2	1285.1	0.15	0.04	0.10	973.9
τ_2, τ_1, τ_3	945.7	0.15	0.08	0.04	914.0
τ_2, τ_3, τ_1	881.0	0.07	0.05	0.07	870.4
τ_3, τ_1, τ_2	1246.8	0.15	0.04	0.10	946.9
τ_3, τ_2, τ_1	926.1	0.06	0.04	0.10	883.0

smaller than the cost using the initial periods. The global optimal solution to the priority and period assignment problem is found when the priority ordering is τ_2, τ_3, τ_1, which gives the optimal total cost $\sum_{i=1}^{3} V_i^* = 870.4$. This can be compared to the initial priority ordering τ_3, τ_1, τ_2 with initial periods, which has the total cost $\sum_{i=1}^{3} V_i^0 = 1246.8$.

Applying the heuristic priority assignment method, the tasks are sorted by $E_i/\sqrt{b_i}$ in ascending order. In this case this results in the same priority order as the one obtained with the global search method.

B. Randomly Generated Examples

To further investigate the performance of the co-design method, 10 sets of 3 plants are randomly generated from the following three plant families:

- Family I: All plants have two stable poles and are drawn from $P_a(s)$ and $P_b(s)$ with equal probability, where
$$P_a(s) = \frac{1}{(s+a_1)(s+a_2)}, \quad P_b(s) = \frac{1}{s^2 + 2\zeta\omega s + \omega^2},$$
with $a_1, a_2, \zeta \in \text{unif}(0,1)$, $\omega \in \text{unif}(0,1)$.
- Family II: All plants have two stable or unstable poles, with each plant drawn from $P_c(s)$ and $P_d(s)$ with equal probability, where
$$P_c(s) = \frac{1}{(s+a_1)(s+a_2)}, \quad P_d(s) = \frac{1}{s^2 + 2\zeta\omega s + \omega^2},$$
with $a_1, a_2, \zeta \in \text{unif}(-1,1)$, $\omega \in \text{unif}(0,1)$.
- Family III: All plants have three stable or unstable poles, with each plant drawn where $P_e(s)$ and $P_f(s)$ with equal probability from
$$P_e(s) = \frac{1}{(s+a_1)(s+a_2)(s+a_3)},$$
$$P_f(s) = \frac{1}{(s^2 + 2\zeta\omega s + \omega^2)(s+a_4)},$$
with $a_1, a_2, a_3, a_4, \zeta \in \text{unif}(-1,1)$, $\omega \in \text{unif}(0,1)$.

For each $P_i(s)$, the cost weighting matrices and noise covariance matrices are randomly generated as

$$\begin{aligned} Q_1 &= 10^{3p} C^T C, \quad Q_2 = 1, \\ R_1 &= 10^{3q} B B^T, \quad R_2 = 0.01, \end{aligned} \tag{17}$$

where B and C are state-space matrices of the plant $P_i(s)$, and $p, q \in \text{unif}(0,1)$. We then design a standard LQG controller $K_i(s)$ for each $P_i(s)$ assuming a constant delay E_i. The initial sampling period is chosen using (15). The initial utilization U_i^0 for each task is assigned by the UUniFast method, and the execution times are given by $E_i = h_i^0 U_i^0$.

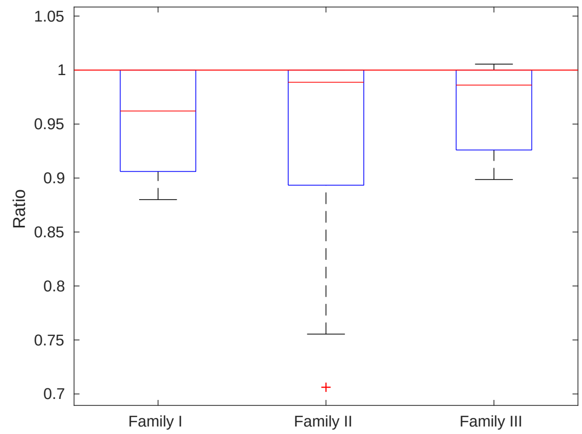

Fig. 7. Ratio of $V_{\text{pri}}/V_{\text{ini}}$ for the randomly generated examples with heuristic priority assignment.

For the given execution time E_i and periods h_i^0, and with the initial priority assignment and using the affine cost function approximations, we calculated the following four overall costs for each set of randomly generated plants:

- Initial overall cost $V_{\textbf{ini}}$. This cost is calculated for the initial periods h_i^0 and with rate-monotonic (RM) priority assignment. This is our baseline approach.
- Overall cost after re-assigning the priorities $V_{\textbf{pri}}$. This cost is calculated for the initial periods h_i^0 with heuristic priority assignment.
- Overall cost after re-assigning priorities and periods $V_{\textbf{heuristic}}$. We re-assign the priorities with the heuristic method and calculate the optimal period assignment and then the overall cost is evaluated.
- Global optimal overall cost $V_{\textbf{global}}$. Here we calculate the optimal periods for all the permutations of priority assignments and find the minimal overall cost.

We compare the ratio of $V_{\text{pri}}/V_{\text{ini}}$ and the ratio of $V_{\text{heuristic}}/V_{\text{ini}}$ and $V_{\text{global}}/V_{\text{ini}}$ in Fig. 7 and 8, respectively.

Fig. 7 shows the performance improvement by priority assignment. In most cases, the overall cost is decreased using the heuristic priority assignment, compared to the initial RM priority assignment. Hence, RM priority assignment which is often used in the real-time control community is not necessarily the best. Here only the priorities were re-assigned using the heuristic method (keeping the same sampling periods), and it gives the lower overall cost.

Fig. 8 shows the performance improvement by period assignment. In all cases, the overall cost is decreased using optimal period assignment. It also shows that $V_{\text{heuristic}}$ is very close to V_{global} in most randomly generated sets of plants. The results imply that the period assignment improves the overall LQG performance, and the performance obtained using the heuristic method gives almost as good results as a much more expensive exhaustive enumeration.

In summary the results show that the heuristic solution

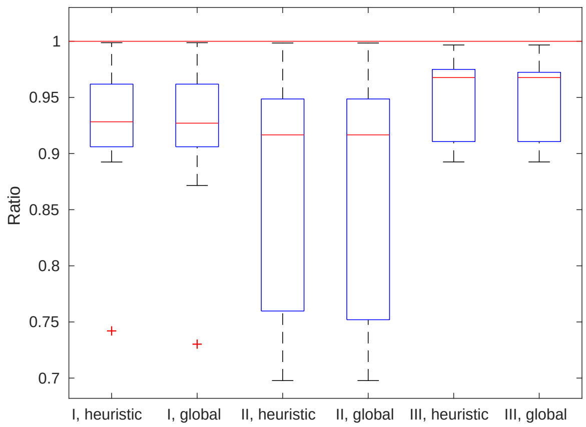

Fig. 8. Ratio of $V_{\text{heuristic}}/V_{\text{ini}}$ and $V_{\text{global}}/V_{\text{ini}}$ for the randomly generated examples with optimized periods.

to the priority assignment problem gives almost the same result as the global solution, and both of them are better than the initial priority assignment, which is the RM priority assignment based on the initial sampling periods. In most cases, the priority assignments by the global solution and by the heuristic solution coincide, but the global solution requires evaluating all the possible priority assignments, which is unfeasible for large task sets. The overall LQG performance can be further improved by using the proposed period assignment method. In the evaluation the performance improvement is visible by comparing Fig. 7 and Fig. 8. The median cost reduction approximately doubles when the period optimization is performed as well.

VII. CONCLUSIONS

In this paper, we proposed an LQG control synthesis method with a robustness constraint, in the context of real-time system scheduling and control co-design. The initial sampling period assignment and the robustness constraint are both based on the jitter margin of the control system. The approximated affine cost function shows that the cost is more sensitive to jitter than to the value of the sampling periods. A priority and period assignment method is given based on the affine cost functions. The evaluation shows that both priority assignment and period assignment are important in obtaining the overall best performance while retaining the initial robustness of the control loops.

A topic for future work is to extend the co-design method to the case of Earliest Deadline First (EDF) scheduling. In this case the scheduling parameters to decide for each task are the period and the deadline. In the real-time systems community several results on deadline assignment for jitter reduction are available, e.g., [27], that can be exploited.

REFERENCES

[1] J. Doyle, “Guaranteed margins for LQG regulators,” *IEEE Transactions on Automatic Control*, vol. 23, no. 4, pp. 756–757, Aug 1978.

[2] J. Moore, D. Gangsaas, and J. Blight, “Performance and robustness trades in LQG regulator design,” in *IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, vol. 20. IEEE, 1981, pp. 1191–1200.

[3] M. Grimble, “Youla parameterised two and a half degrees of freedom LQG controller and robustness improvement cost weighting,” in *IEE Proceedings D (Control Theory and Applications)*, vol. 139, no. 2. IET, 1992, pp. 147–160.

[4] N. Matni and J. C. Doyle, “Optimal distributed LQG state feedback with varying communication delay,” in *IEEE Conference on Decision and Control*, 2013, pp. 5890–5896.

[5] C.-Y. Kao and B. Lincoln, “Simple stability criteria for systems with time-varying delays,” *Automatica*, vol. 40, no. 8, pp. 1429–1434, Aug. 2004.

[6] A. Cervin, B. Lincoln, J. Eker, K.-E. Årzén, and G. Buttazzo, “The jitter margin and its application in the design of real-time control systems,” in *International Conference on Real-Time and Embedded Computing Systems and Applications*, 2004.

[7] D. Seto, J. P. Lehoczky, L. Sha, and K. G. Shin, “On task schedulability in real-time control systems,” in *IEEE Real-Time Systems Symposium*, Dec 1996, pp. 13–21.

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

[9] E. Bini and A. Cervin, “Delay-aware period assignment in control systems,” in *IEEE Real-Time Systems Symposium*, Nov 2008, pp. 291–300.

[10] Y. Xu, K.-E. Årzén, E. Bini, and A. Cervin, “Response time driven design of control systems,” *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 6098–6104, 2014.

[11] G. M. Mancuso, E. Bini, and G. Pannocchia, “Optimal priority assignment to control tasks,” *ACM Transactions on Embedded Computing Systems*, vol. 13, pp. 161–161, 2014.

[12] J. Nilsson, B. Bernhardsson, and B. Wittenmark, “Stochastic analysis and control of real-time systems with random time delays,” *Automatica*, vol. 34, no. 1, pp. 57–64, 1998.

[13] Y. Xu, K. E. Årzén, A. Cervin, E. Bini, and B. Tanasa, “Exploiting job response-time information in the co-design of real-time control systems,” in *International Conference on Embedded and Real-Time Computing Systems and Applications*, Aug 2015, pp. 247–256.

[14] A. Aminifar, S. Samii, P. Eles, Z. Peng, and A. Cervin, “Designing high-quality embedded control systems with guaranteed stability,” in *IEEE Real-Time Systems Symposium*, Dec 2012, pp. 283–292.

[15] D. Goswami, R. Schneider, and S. Chakraborty, “Co-design of cyber-physical systems via controllers with flexible delay constraints,” in *16th Asia and South Pacific Design Automation Conference*, Jan 2011, pp. 225–230.

[16] K. Zhou, K. Glover, B. Bodenheimer, and J. Doyle, “Mixed H_1 and H_∞ performance objectives. I. Robust performance analysis,” *IEEE Transactions on Automatic Control*, vol. 39, no. 8, pp. 1564–1574, Aug 1994.

[17] R. Muradore and G. Picci, “Mixed H_2/H_∞ control: the discrete-time case,” *Systems & Control Letters*, vol. 54, no. 1, pp. 1–13, 2005.

[18] S. P. Boyd and C. H. Barratt, *Linear controller design: limits of performance*. Prentice Hall, 1991.

[19] B. Lincoln and A. Cervin, “Jitterbug: A tool for analysis of real-time control performance,” in *IEEE Conference on Decision and Control*, 2002.

[20] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*, 3rd ed. Addison-Wesley, 1994.

[21] K. J. Åström and B. Wittenmark, *Computer-Controlled Systems—Theory and Design*, 3rd ed. Prentice Hall, 1997.

[22] W. S. Levine, Ed., *The Control Handbook*. CRC Press, 1996.

[23] M. Grant, S. Boyd, and Y. Ye, “CVX: Matlab software for disciplined convex programming,” 2008. [Online]. Available: http://www.stanford.edu/∼boyd/cvx/

[24] K. S. Arun, T. S. Huang, and S. D. Blostein, “Least-squares fitting of two 3-D point sets,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 5, pp. 698–700, 1987.

[25] E. Bini and S. K. Baruah, “Efficient computation of response time bounds under fixed-priority scheduling,” in *Conference on Real-Time and Network Systems*, 2007, pp. 95–104.

[26] E. Bini and G. C. Buttazzo, “Measuring the performance of schedulability tests,” *Real-Time Systems*, vol. 30, no. 1, pp. 129–154, 2005.

[27] T. Kim, H. Shin, and N. Chang, “Deadline assignment to reduce output jitter of real-time tasks,” *IFAC Proceedings Volumes*, vol. 33, no. 30, pp. 51–56, 2000, 16th IFAC Workshop on Distributed Computer Control Systems, Sydney, Australia.