Using Jitterbug to Derive Control Loop Timing Requirements

Cervin, Anton

Published in:
Proceedings of CERTS'03 – Co-Design of Embedded Real-Time Systems Workshop

2003

CERTS'03 – Co-Design of Embedded Real-Time Systems Workshop, Porto, Portugal, July 2003.

Using Jitterbug to Derive Control Loop Timing Requirements

Anton Cervin

Department of Automatic Control
Lund Institute of Technology
Box 118, SE-221 00 Lund, Sweden
anton@control.lth.se

Abstract

Linking scheduling attributes to control performance specifications is a difficult problem. This paper discusses how the MATLAB toolbox Jitterbug can be used to derive timing requirements for control loops from various control performance specifications. The resulting timing requirements include specifications on sampling periods, latencies, and jitter. An overview of the Jitterbug approach is given, and limitations of the tool are pointed out. A control design example is given, and, finally, topics where more research is needed are outlined.

1. Introduction

The design of a real-time control system is essentially a codesign problem, where limited resources should be allocated to control tasks and other tasks such that optimum overall performance is achieved. In this paper, we will focus on the control and scheduling codesign problem. More specifically, we will deal with the problem of scheduling-induced jitter in periodic control loops.

A digital controller is normally designed assuming a fixed sampling period T, and, possibly, assuming a fixed computational delay τ. These simplistic design assumptions are seldom met in the target system. When executing as a task in a real-time system, the controller will suffer from time-varying latencies, induced by preemption from interrupts and higher-priority tasks. The result is degraded control performance. Some performance degradation is normally acceptable, as long as the controller meets its design specifications.

In the scheduling design, a controller is traditionally described as a periodic task with a period T, a deadline D, and a computation time C. It is normally assumed that $D = T$, although a shorter deadline can be used to limit the end-to-end latency in the controller. It can be argued that the traditional timing model is too simplistic, since it does not reflect the fact that a controller is composed of (at least) three distinct operations: the input operation (or *sampling*), the control computation, and the output operations (or *actuation*). To get better control of the latency and jitter in the controller, it is possible to schedule the different part of the controller as separate tasks. Subtask scheduling of control tasks has been treated in, e.g., [Crespo *et al.*, 1999] and [Cervin, 1999]. These papers references assume a particular scheduling policy and that the sampling periods of the controllers are fixed at the scheduling design stage. In reality, the sampling period of the controllers are also design parameters. The sampling periods are typically chosen according to rules of thumb. One such rule [Åström and Wittenmark, 1997] states that the sampling period T should be chosen such that

$$\omega_b T \approx 0.2\text{–}0.6, \tag{1}$$

where ω_b is the bandwidth of the closed-loop system. It should be noted that faster sampling may be required if there is latency and jitter in the control loop.

In order to make correct trade-offs in the scheduling design, the designer needs to what ranges of sampling periods, latencies, and jitter that are acceptable to each control loop. In [Bate *et al.*, 2003], time-domain analysis involving extensive simulations are used to derive timing requirements for digital controllers. In contrast to that work, this paper relies entirely on analytical computations of cost functions and frequency responses to derive the timing requirements. the Jitterbug toolbox [Lincoln and Cervin, 2002] linear model,

The rest of this paper is outlined as follows. In the next section, an overview of control loop timing and its relation to control performance is given. In Section 3, it is described how Jitterbug can be used to model the timing variations in a control loop. Also, an overview of the control design criteria that can be evaluated using Jitterbug are given. In Section 4, the approach is exemplified on a control application, deriving bounds on the sampling period, latency, and jitter given a performance specification. In Section 5, the problem of linking timing requirements to scheduling analysis is discussed. Finally, in Section 6, some concluding remarks are given, and areas where further research are needed are outlined.

2. Control Loop Timing

A control task generally consists of three distinct operations: input data collection, control algorithm computation, and output signal transmission, see Figure 1. The timing of the operations are crucial to the performance of the con-

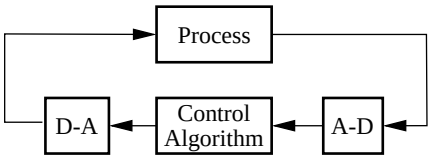

Figure 1 A computer-controlled system. The control task consists of three distinct parts: input data collection (A-D), control algorithm computation, and output signal transmission (D-A).

troller. Ideally, the control algorithm should be executed with perfect periodicity, and there should be zero delay between the reading of the inputs and the writing of the outputs. This will not be the case in a real implementation, where the execution and scheduling of tasks introduce latencies.

The basic timing parameters of a control task are shown in Figure 2. It is assumed that the control task is released periodically at times given by $r_k = kT$, where T is the sampling interval of the controller. Due to preemption from other tasks in the system, the actual start of the task may be delayed for some time L_s. This is called the *sampling latency* of the controller. A dynamic scheduling policy will introduce variations in this interval. The *sampling jitter* is quantified by the difference between the maximum and minimum sampling latencies in all task instances,

$$J_s \stackrel{\text{def}}{=} L_s^{max} - L_s^{min}. \tag{2}$$

Normally, it can be assumed that the minimum sampling latency of a task is zero, in which case we have $J_s = L_s^{max}$.

After some computation time and possibly further preemption from other tasks, the controller will actuate the control signal. The delay from the sampling to the actuation is called the *input-output latency*, denoted L_{io}. Varying execution times or task scheduling will introduce variations in this interval. The *input-output jitter* is quantified by

$$J_{io} \stackrel{\text{def}}{=} L_{io}^{max} - L_{io}^{min}. \tag{3}$$

In general terms, the performance of a digital controller depends on the sampling period and the particular sequences of sampling and input-output latencies, $\{L_s^k\}$ and $\{L_{io}^k\}$. From the controller's point of view, the time-varying latencies can be viewed as random variables (that are independent between periods). Under the simplifying assumption that the distributions of the latencies can be sufficiently

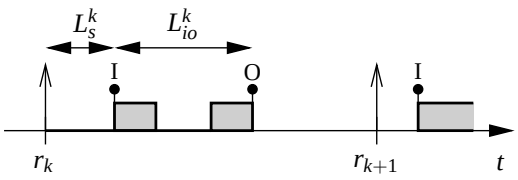

Figure 2 Digital controller timing. Each period, the controller experiences sampling latency, L_s, and input-output latency, L_{io}.

accurately described by their minimum and maximum values, the performance J of the controller can be expressed as a function of the sampling period T, the minimum input-output latency L_{io}^{min}, the sampling jitter J_s, and the input-output jitter J_{io}:

$$J = J(T, L_{io}^{min}, J_s, J_{io}). \tag{4}$$

The goal of the analysis in the next section is to derive bounds on T, L_{io}^{min}, J_s, and J_{io} from various control performance specifications.

3. Analysis Using Jitterbug

Jitterbug [Lincoln and Cervin, 2002] is a MATLAB-based toolbox that is used to analyze linear control systems with time-varying delays. The control system is described by a number of connected continuous-time and discrete-time linear systems, representing the plant and the controller. In the simplest case, a periodic timing model with random delays is used to describe the execution of the discrete-time systems, i.e., the control task.

A Jitterbug model corresponding to the computed-controlled system in Figure 1 is shown in Figure 3. The signal model consists of three connected linear systems. The process is described the continuous-time system $G(s)$. The digital controller is described by two discrete-time blocks, *Samp* and $C(z)$. The first block models the sampling operation, while the second block represents the control algorithm and the actuator. (Implicit in each discrete-time block is a sampler at the input and a zero-order-hold circuit at the output.) The associated timing model consists of three nodes. The first node is periodic (with a given period T) and represents the release of the control task. There is a random delay L_s until the second node where H_1 is updated, and another random delay L_{io} until the third node where H_2 is updated.

In general, Jitterbug can accept arbitrary probability density distributions in the timing model. Here, to limit the design space, we let the controller timing be described by the variables T, L_{io}^{min}, J_s, and J_{io} only. Furthermore, we assume that the latencies are uniformly distributed between their minimum and maximum values. Hence, we let

$$L_s \in U(0, J_s), \tag{5}$$

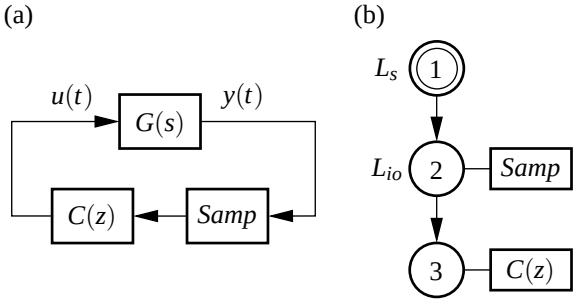

Figure 3 Jitterbug model of a digital control loop: (a) signal model, and (b) timing model.

and

$$L_{io} \in U(L_{io}^{min}, L_{io}^{min} + J_{io}), \tag{6}$$

where $U(a,b)$ denotes a uniform probability distribution between a and b. It should be noted that these uniform latency distributions are quite "nice" to the control loop. A more malign choice of distributions would be to let the latencies vary between the extreme points only. This could result in quite conservative timing requirements, however.

3.1 Performance Criteria and Jitterbug

Below, an overview of the control performance criteria that can be evaluated analytically using Jitterbug are given. Control design always involves trade-offs between various design specifications. A good overview of common performance specifications in computed-controlled systems is given in [Wittenmark *et al.*, 2002]. More material on trade-offs in linear control design can be found in [Boyd and Barratt, 1991].

Stability. A first requirement for any control loop is that it is stable. This property is always checked by Jitterbug. However, since the system is stochastic (due to the time-varying delays), Jitterbug only guarantees so called *mean square stability* of the closed-loop system. This means that there might exist a particular sequence of delays and noises that makes the system go unstable, although the probability of this is zero. (For further discussion on different stability concepts, see [Ji *et al.*, 1991].)

Quadratic Cost Functions. The main purpose of Jitterbug is to facilitate control performance analysis via the computation of *quadratic cost functions*. Such functions are commonly used evaluate the performance of linear controllers. External inputs (reference signals and disturbances) are modeled as white noise processes that enter the control loop at various points. Given a model, Jitterbug can compute a stationary cost function on the form

$$J = \lim_{t\to\infty} \frac{1}{t} \int_0^T x^T(s) Q x(s)\, ds, \tag{7}$$

where x is the state vector for the total system (including the plant states and the controller states) and Q is a chosen semi-definite weighting matrix.

In the LQG design method, a linear controller is explicitly designed to minimize a quadratic cost function. It is then natural to use the same cost function when evaluating the performance of the controller. Given a nominal design, a typical performance specification could be to allow the value of the cost function to increase by, e.g., 10 percent due to scheduling-induced latencies. The same approach can be used also for other linear controllers.

Frequency-Domain Specifications. A classical approach to control design is to use frequency-domain specifications. With Jitterbug, it is possible to compute the *magnitude* of various closed-loop transfer functions, also in the presence of jitter.

Let the sampled-data representation of the process be $P(z)$, while the control algorithm is given by $C(z)$. The response of the closed-loop system is then completely characterized by the four transfer functions

$$\begin{aligned} H_1(z) &= \frac{1}{1+C(z)P(z)}, \quad H_2(z) = \frac{C(z)}{1+C(z)P(z)}, \\ H_3(z) &= \frac{P(z)}{1+C(z)P(z)}, \quad H_4(z) = \frac{C(z)P(z)}{1+C(z)P(z)}. \end{aligned} \tag{8}$$

Performance requirements are commonly expressed as requirements on the magnitudes of these functions. For instance, for reference signal tracking, it can be required that H_4 has a certain bandwidth (i.e., that the magnitude stays above -3 dB up to a certain frequency). The response to input load disturbances is given by H_3, and is typically required to be low at low frequencies, and so on.

Formally, transfer functions are only defined for linear, time-invariant systems. However, using the concept of *spectral densities*, Jitterbug can also compute the frequency response of systems with jitter. Given a time-invariant closed-loop system $H(z)$ which is excited by discrete-time white noise with unit intensity, the spectral density ϕ_y of the output is given by

$$\phi_y(\omega) = |H(e^{i\omega})|^2. \tag{9}$$

We hence can find the magnitude of the frequency response by

$$|H(e^{i\omega})| = \sqrt{\phi_y(\omega)}. \tag{10}$$

For systems with jitter, $\phi_y(\omega)$ is still defined, and, furthermore, it can be computed with Jitterbug. The quantity $\sqrt{\phi_y(\omega)}$ should then be interpreted as the average gain of the closed-loop system at a given frequency.

Robustness Measures. Two common robustness measures for control systems are the maximum sensitivity and the maximum complementary sensitivity. The sensitivity function is defined as

$$S(z) = \frac{1}{1+C(z)P(z)}, \tag{11}$$

and the maximum sensitivity is given by

$$M_s = \max_{\omega} |S(e^{i\omega})|. \tag{12}$$

For linear, time-invariant systems, $1/M_s$ can be interpreted as the distance from the loop gain $C(z)P(z)$ to the instability point -1 in the Nyquist diagram. Similar to above, using spectral density calculations, an interpretation for systems with jitter is also possible.

Likewise, the complementary sensitivity function is given by

$$T(z) = \frac{C(z)P(z)}{1+C(z)P(z)}, \tag{13}$$

and the maximum complementary sensitivity by

$$M_t = \max_{\omega} |T(e^{i\omega})|. \tag{14}$$

Common design specifications for M_s and M_t are in the range of 1.2 to 2.0.

3.2 Limitations of Jitterbug Approach

A number of limitations with the proposed approach exist:

- The timing model in Jitterbug is quite simplistic, in that the delays are assumed to be independent from period to period. Hence, the model can not fully describe the timing variations introduced by a dynamic scheduling algorithm. Also, the tool cannot be used to analyze systems where the scheduling parameters change over time (as in feedback scheduling applications).
- The toolbox only computes a *mean* performance index, averaged over an infinite time horizon. Stability is only guaranteed in the *mean square* sense, i.e., the system might become unstable for a particular (but highly unlikely) sequence of delays.
- There is no time-domain analysis in Jitterbug. It is for instance not possible to give specifications on rise-time or maximum overshoot. However, time-domain control specifications can often be translated into frequency-domain specifications, see [Boyd and Barratt, 1991].
- In Jitterbug, it is necessary to specify the distribution of the sampling and input-output latencies. Since these are generally unknown, certain probability distributions must be assumed. Uniform distributions (which are used here) might be too benign, whereas end-point distributions might be to pessimistic.

4. Example

In this section, a design example is given, where we consider LQG (linear-quadratic-Gaussian) control of a servo process, described by the continuous-time transfer function

$$P(s) = \frac{1000}{s(s+1)}.$$

The process is assumed to be disturbed by continuous-time white input noise and with unit variance and discrete-time measurement noise with a variance of 0.1. An LQG controller, denoted $C(z)$, is designed using a sampling interval of T and an assumed a constant input-output latency of L. The controller is designed to minimize the continuous-time cost function

$$J = \lim_{t\to\infty} \frac{1}{t} \int_0^T \left(y^2(s) + u^2(s) \right) ds. \qquad (15)$$

using the Jitterbug command `lqgdesign`. The Jitterbug model of the control system was shown in Figure 3.

4.1 Cost Function Specification

First, we consider a cost function specification, where the value of the cost (15) is evaluated for different values of T, L_{io}^{min}, J_s, and J_{io}. From initial design attempts and time-domain simulation, it has been decided that a cost of at most $J = 100$ gives acceptable performance for the control loop. (Remember that a lower cost means better performance.)

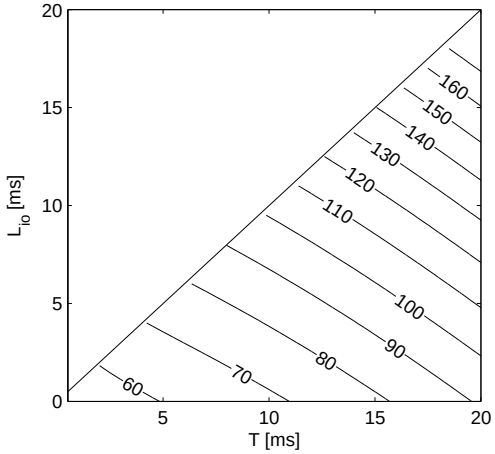

Figure 4 Cost as a function of T and L_{io}^{min}, assuming $J_s = 0$ and $J_{io} = 0$.

By fixing two of the timing parameters, the cost as a function of the remaining parameters can be illustrated in a diagram. In Figure 4, the cost has been computed as a function of T and L_{io}^{min}, assuming zero sampling jitter and zero input-output jitter. It is seen that the control loop is quite sensitive to input-output latency, even though the controller has been designed to compensate optimally for the delay. If a delay is present, faster sampling is required to obtain a cost below 100.

To illustrate what the cost function means, control designs corresponding to three points in the design space in Figure 4 have been evaluated in time-domain simulations. In Figure 5, the responses to an impulse disturbance at time zero have been plotted:

- The full response has the cost $J_1 = 50$, corresponding to the parameters $T = 0.5$ ms and $L_{io} = 0$.
- The dashed response has the cost $J_2 = 100$, corresponding to the parameters $T = 25$ ms and $L_{io} = 0$.
- The dot-dashed response has the cost $J_3 = 100$, corresponding to the parameters $T = 10$ ms and $L_{io} = 10$ ms.

Next, the impact of sampling jitter and input-output jitter on control performance is studied. The sampling period has been fixed to $T = 20$ ms and the minimum latency is set to zero (corresponding to the lower-right corner of Figure 4). The controller is designed assuming a constant latency equal to $J_{io}/2$. The resulting cost is shown in Figure 6. It is seen that, in this example, the control loop is more sensitive to input-output jitter that to sampling jitter. To keep the cost below 100, the both jitters must be less than a fraction of the sampling interval.

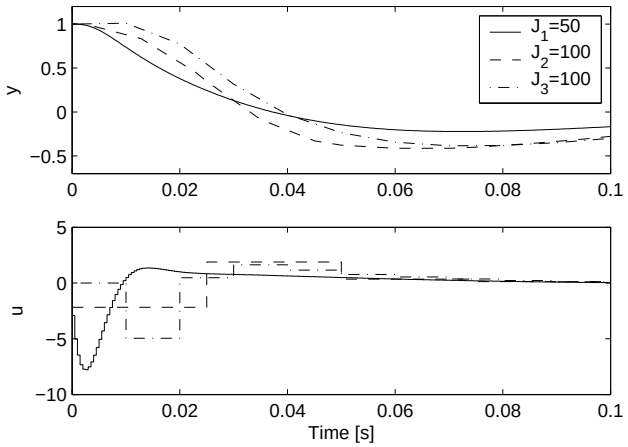

Figure 5 Examples of time-domain control performance corresponding to different values of the cost function.

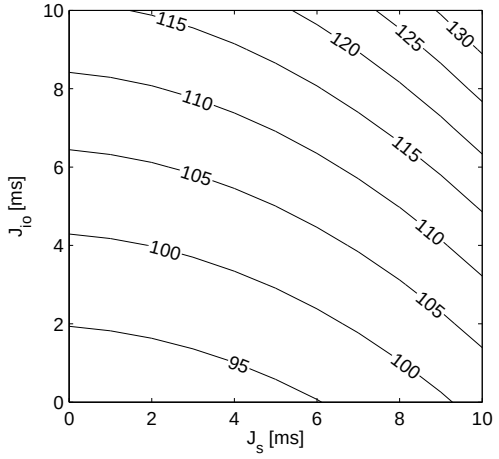

Figure 6 Cost as a function of sampling jitter and input-output jitter, assuming $T = 20$ ms and $L_{io}^{min} = 0$. The controller is designed assuming a constant delay of $J_{io}/2$.

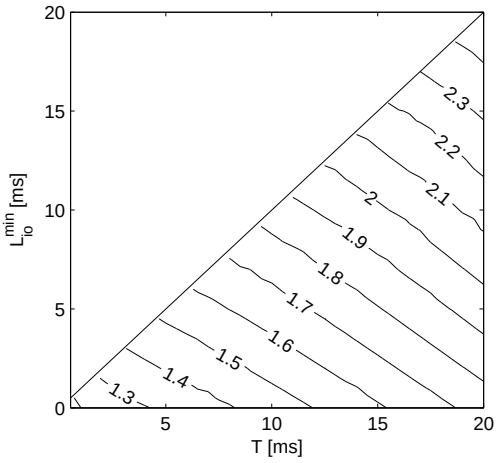

Figure 7 Maximum sensitivity as a function of T and L_{io}^{min}, assuming $J_s = 0$ and $J_{io} = 0$.

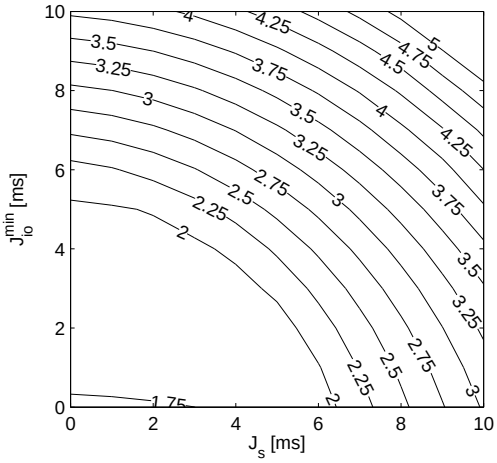

Figure 8 Maximum sensitivity as a function of sampling jitter and input-output jitter, assuming $T = 20$ ms and $L_{io}^{min} = 0$. The controller is designed assuming a constant delay of $J_{io}/2$.

4.2 Robustness Specification

It can sometimes be difficult to select a reasonable upper bound on a cost function, especially in cases where the controller was not directly designed using a cost function (as it is in LQG-control). An interesting alternative is to instead place a bound on the sensitivity function (and possibly also the complimentary sensitivity function). This is often a more simple task, since a value of the maximum sensitivity, M_s, can be chosen independently of the size of the plant and controller parameters.

To continue the example, we assume that a reasonable value of M_s is 2.0. Similar to above, the value of M_s is evaluated as a function of T, L_{io}^{min}, J_s, and J_{io}. In Figure 7, M_s has been computed as a function of T and L_{io}^{min}, assuming zero sampling jitter and zero input-output jitter. Compared to Figure 4, we obtain similar bounds on the timing parameters.

Next, the maximum sensitivity is computed as a function of the amount of sampling jitter and input-output jitter. As before, the sampling period is set to $T = 20$ ms and the minimum latency is set to zero. The result is shown in Figure 8. According to this measure, the system is quite sensitive towards both sampling jitter and input-output jitter (compare with Figure 6).

5. Linking Scheduling Analysis to Controller Timing

The above analysis has assumed that values of T, L_{io}^{min}, J_s, and J_{io} are given. Assuming a controller task set and standard fixed-priority scheduling, the values of L_s^{max}, L_{io}^{min}, and L_{io}^{max} can be found using worst-case and best-case response-time analysis [Joseph and Pandya, 1986; Redell and Sanfridson, 2002]:

$$L_{s_i}^{max} = \sum_{j \in hp(i)} \left\lceil \frac{L_{s_i}^{max}}{T_j} \right\rceil C_j. \qquad (16)$$

$$L_{io_i}^{max} = C_i + \sum_{j \in hp(i)} \left\lceil \frac{L_{io_i}^{max}}{T_j} \right\rceil C_j. \qquad (17)$$

$$L_{io_i}^{min} = C_i^b + \sum_{j \in hp(i)} \left\lceil \frac{L_{io_i}^{min} - T_j}{T_j} \right\rceil C_j^b, \qquad (18)$$

Here, C_b^i denotes the *best-case* execution time of task i. As pointed out before, for more accurate analysis, it would be necessary to have the distributions of the latencies as well. This is an area where statistical approaches to scheduling analysis could be used. Also, results regarding minimum response times are lacking under EDF scheduling.

A difficult part of the codesign process is to modify the scheduling parameters such that all performance specifications are met. For this purpose some kind of search procedure must be used. One problem is that the timing attributes (T, L_{io}^{min}, J_s, and J_{io}) depend on the scheduling parameters (T, D, C) in a very nonlinear manner. Other scheduling policies than fixed-priority scheduling could give simpler design problems. One example is the Control Server model [Cervin and Eker, 2003], where T and L_{io} are determined directly by the task utilization factor U.

6. Conclusion

We have described how Jitterbug can be used to derive timing requirements from control performance specifications. The derived requirements are expressed in terms of the sampling interval, the minimum input-output latency, the sampling jitter, and the input-output jitter. The performance specifications can be given in terms of a quadratic cost function, or as constraints on the magnitude of certain closed-loop transfer functions (e.g., the sensitivity function). The analysis is only approximate, since it assumes that the delays introduced by the scheduling can be described by independent random variables with uniform distributions. Jitterbug allows for arbitrary distributions to be used, but the current state of the art in scheduling analysis does not allow the delay distributions to be derived.

References

Åström, K. J. and B. Wittenmark (1997): *Computer-Controlled Systems.* Prentice Hall.

Bate, I., P. Nightingale, and A. Cervin (2003): "Establishing timing requirements and control attributes for control loops in real-time systems." In *Proceedings of the 15th Euromicro Conference on Real-Time Systems.* Porto, Portugal.

Boyd, S. P. and C. H. Barratt (1991): *Linear Controller Design—Limits of Performance.* Prentice Hall.

Cervin, A. (1999): "Improved scheduling of control tasks." In *Proceedings of the 11th Euromicro Conference on Real-Time Systems*, pp. 4–10. York, UK.

Cervin, A. and J. Eker (2003): "The Control Server: A computational model for real-time control tasks." In *Proceedings of the 15th Euromicro Conference on Real-Time Systems.* Porto, Portugal. To appear.

Crespo, A., I. Ripoll, and P. Albertos (1999): "Reducing delays in RT control: The control action interval." In *Proc. 14th IFAC World Congress*, pp. 257–262.

Ji, Y., H. Chizeck, X. Feng, and K. Loparo (1991): "Stability and control of discrete-time jump linear systems." *Control-Theory and Advanced Applications*, **7:2**, pp. 247–270.

Joseph, M. and P. Pandya (1986): "Finding response times in a real-time system." *The Computer Journal*, **29:5**, pp. 390–395.

Lincoln, B. and A. Cervin (2002): "Jitterbug: A tool for analysis of real-time control performance." In *Proceedings of the 41st IEEE Conference on Decision and Control.* Las Vegas, NV.

Redell, O. and M. Sanfridson (2002): "Exact best-case response time analysis of fixed priority scheduled tasks." In *Proc. 14th Euromicro Conference on Real-Time Systems.* Vienna, Austria.

Wittenmark, B., K. J. Åström, and K.-E. Årzén (2002): "Computer control: An overview." Technical Report. IFAC professional brief.