# Derivation and Application of Hard Deadlines for Real-Time Control Systems

Kang G. Shin, *Student Member, IEEE* and Hagbae Kim, *Student Member, IEEE*

*Abstract*— **The computation-time delay in the feedback controller of a real-time control system may cause failure to update the control input during one or more sampling periods. If this delay exceeds a certain limit called a hard deadline, either the necessary conditions for system stability are violated or the system leaves the allowed state-space. In such a case a dynamic failure is said to occur to the system. A method for calculating the hard deadlines in linear time-invariant control systems by considering system stability and the allowed state-space is presented. To derive necessary conditions for (asymptotic) system stability, the state difference equation is modified based on an assumed maximum delay and the probability distribution of delays whose magnitudes are less than, or equal to, the assumed maximum delay. Moreover, the allowed state-space—which is derived from given input and state constraints—is used to calculate the hard deadline as a function of time and the system state. A one-shot delay model in which a single event causes a dynamic failure is also considered. The knowledge of hard deadline is then applied to the design of error recovery in a triple modular redundant (TMR) controller computer.**

## I. Introduction

DIGITAL COMPUTERS are commonly used in real-time control systems due mainly to their improved performance and reliability in dealing with increasingly complex controlled processes. A digital computer in the feedback loop of such a control system calculates the control input by executing a sequence of instructions, thereby introducing an unavoidable delay—called the computation-time delay—to the controlled process. This is an extra delay in addition to the system delay commonly seen in the control literature. The computation-time delay is an important part of the delay in the feedback loop, which also includes the other parts of delay related to measurement or sensing, A/D and D/A conversion, and actuation. However, these other parts of delay are usually constant, and thus easy to deal with.

Due to data-dependent loops and conditional branches, and unpredictable delays in sharing resources during the execution of programs that implement control algorithms, the computation-time delay is a continuous random variable that is usually much smaller than one sampling period, $T_s$, if no failure occurs to the controller computer. Since a large number of real-time control systems are life, or safety-critical, controller computers for such systems must be equipped with some fault-tolerance mechanisms. When a component failure or environmental disruption such as an electromagnetic interference (EMI) occurs, the time taken for error detection, fault location, and recovery must be added to the execution time of a control program, thus increasing the computation-time delay significantly. This increase in task execution time seriously degrades the system performance and may even lead to a catastrophe, or a dynamic failure by either violating the necessary conditions for system stability or making the system leave the state-space circumscribed by the given conditions (i.e., the allowed state-space) if the delay exceeds a certain limit called the hard deadline [10].

Several researchers attempted to analyze the effects of computation-time delay on the performance of stability of a control system. The authors of [13] considered the qualitative effect of feedback delay on a multivariable, computer-controlled linear system by proposing an algorithm to compensate for the delay effect. The sufficient (necessary) conditions of stability with a feedback delay and the delay effects on quadratic performance indices were presented for a linear control system [2] (for a nonlinear robot control system [9]). In [8], a more detailed analysis of the stability of a digital control system with feedback delay was carried out by modifying the state difference equation. However, all these analyses are based on the assumption that the feedback delay is fixed or constant. Although the stability problem with a variable-feedback delay was investigated in [4], it was still based on a regular and periodic (i.e., thus deterministic) pattern of delay. In [1], a control system with a random time-varying delay in the feedback loop was modeled with a stochastic-delay differential equation, and sufficient conditions were derived for the almost-sure sample stability under which almost every possible differential equation is an ensemble of stochastic systems has a stable solution. However, this result did not give any explicit relation between the performance (or stability) and the magnitude of delay, but, instead, gave a condition of the coefficients of the state equations and the average rate-of-change of delay for sample stability. Furthermore, this work assumed a delay to be bounded by the "worst-case" intersample period. In [10], the hard deadline in controlling the elevator deflection of the aircraft landing problem was obtained numerically by using the concept of allowed state-space.

In this paper, we analyze computation-time delay effects on system stability and state constraints for linear, time-invariant control systems. Specifically, we derive the hard deadline for such control systems—the critical value of computation-time

Manuscript received October 27, 1991; revised April 10, 1992. This work was supported in part by the Office of Naval Research under Contract N00 014-91-J-1115, and in part by NASA Grant NAG-1-1120.

The authors are with the Real Time Computing Laboratory, the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122.

IEEE Log Number 9202129.

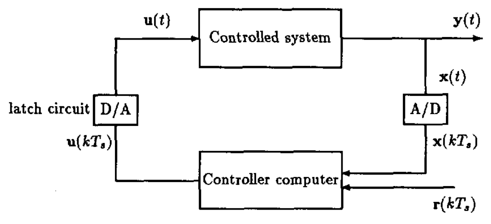


Fig. 1. A computer-controlled system.

delay beyond which a dynamic failure occurs. The feedback delay is assumed to result only from the computation-time delay because other delay elements can be readily dealt with. The computation-time delay is assumed to be stochastically stationary, which corresponds to the characteristics of transient computer failures caused by, for example, electromagnetic interferences. This property can be represented by a binomially-distributed random variable. The system dynamics are modified first according to the *assumed* maximum delay, $NT_s$, and the probability distribution of delays whose occurrence periods $\leq NT_s$, where $N$ is changed from 1 to the actual maximum delay (or hard deadline) denoted by $DT_s$. The pole positions of the modified state equation will then be tested to derive necessary conditions for (asymptotic) system stability. Also, the state and input constraints are used to derive the allowed state-space from which the hard deadline is derived as a function of time and the system state. This analysis is useful for the one-shot delay model, where a single event—a long-lasting failure—may cause a dynamic failure.

Section II addresses the generic problem for qualitative analysis of computation-time delay effects and reviews the basic definition of a hard deadline in real-time digital control systems. Section III presents a method for modifying the state difference equation in the presence of the computation-time delay, and then analyzes system stability by examining the pole positions of the modified state difference equation. the hard deadline associated with the one-shot delay model is also analyzed there. Several simple linear systems are examined to demonstrate the approach in Section IV. Section V deals with the application of the hard-deadline information to the design of a reliable controller computer and the estimation of a control system's reliability. The paper concludes with Section VI.

## II. Computation-Time Delay Effects on Controlled Processes

As shown in Fig. 1, a controller computer calculated the control input at each sampling interval for a linear time-invariant controlled process that is described by the vector difference equation:

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) \tag{1}$$

where $k$ is the time index, one unit of time represents the sampling interval $T_s$, and $\boldsymbol{x} \in \mathcal{R}^n$ and $\boldsymbol{u} \in \mathcal{R}^l$ are the state and input vectors, respectively.

The coefficient matrices, $\boldsymbol{A} \in \mathcal{R}^{n\times n}$ and $\boldsymbol{B} \in \mathcal{R}^{n\times l}$, are obtained from those of the corresponding continuous-time model:

$$\boldsymbol{A} = e^{\boldsymbol{A}_c T_s}, \boldsymbol{B} = \int_0^{T_s} e^{\boldsymbol{A}_c(T_s-s)}\boldsymbol{B}_c \, ds \tag{2}$$

where $\boldsymbol{A}_c$ and $\boldsymbol{B}_c$ are the corresponding coefficient matrices of the continuous-time model. The (digital) controller computer reads sensor values of system output, compares them with desired values, and calculates the control input once every $T_s$ s according to a programmed control strategy. The control input, which is held constant within each sampling interval by a latch circuit, is applied to the continuous-time controlled process.

Equation (1) must include the delays associated with the measurement or sensing, A/D and D/A conversion, and the execution of control algorithms. The sum of these delays is usually much smaller than one sampling period, $T_s$, in the absence of computer failure(s) or external interferences like an EMI, which was called the delay problem in [9]. For the delay problem where the magnitude of delay, $\Delta$, is smaller than $T_s$, one can change the state (1) to

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}_1\boldsymbol{u}(k) + \boldsymbol{B}_2\boldsymbol{u}(k-1) \tag{3}$$

where

$$\boldsymbol{B}_1 = \int_{\Delta}^{T_s} e^{\boldsymbol{A}_c(T_s-s)}\boldsymbol{B}_c \, ds$$

and

$$\boldsymbol{B}_2 = \int_0^{\Delta} e^{\boldsymbol{A}_c(T_s-s)}\boldsymbol{B}_c \, ds.$$

When a fault[1] occurs in the controller computer, the computation-time delay resulting from the corresponding recovery actions may be large relative to $T_s$. This was termed the loss problem in [9]. Suppose the controller computer fails to update the control input for the next $n$ sampling intervals since time $k_0$. The control input during these intervals will be held constant at $u(k_0)$ by the D/A converter and latch circuit. Since faults occur randomly during the mission lifetime, the failure to update the control input is considered a stochastic disturbance to the controlled process, which can be represented by a model depending on the fault characteristics.

The delay introduced in the feedback loop degrades the performance of the controlled process, and it may even lead to a dynamic failure if the delay exceeds the hard deadline. A dynamic failure occurs if either the necessary conditions for system stability are violated or the system leaves the state-space circumscribed by the given conditions, i.e., the allowed state-space.

When the environment is assumed to be stochastically stationary, the occurrence of computer failure(s) or computation-time delay can be represented by a probability distribution function. The delay may change the pole positions of the controlled process, from which one can derive the necessary conditions for system stability.

Let $\boldsymbol{X}_A$ and $\boldsymbol{U}_A$ be the allowed state-space and the admissible input space, respectively. Suppose the state is evolved

[1] An EMI is considered a fault because it may cause the loss of the controller computer.

from time $k_0$ in the presence of a computation-time delay $N$ according to

$$\boldsymbol{x}(k) = \phi(k, k_0, \boldsymbol{x}(k_0), \boldsymbol{u}(k-N)) \tag{4}$$

where $\phi$ is the state transition map. Then, the hard deadline of a control task starting at time $k_0$ can be represented by

$$D(\boldsymbol{x}(k_0)) = \sup_{\boldsymbol{u}(k-N)\in \boldsymbol{U}_A} \{N : \phi(k, k_0, \boldsymbol{x}(k_0), \boldsymbol{u}(k-N)) \in \boldsymbol{X}_A\}. \tag{5}$$

That is, the hard deadline of this task starting at time $k_0$ is defined as the maximum computation-time delay that the controlled process can tolerate at that time. The hard deadline of a task during the time intervals $[k_0 T_s, k_1 T_s]$ is also defined as

$$D(k_1, \boldsymbol{x}(k_0)) = \inf_{\boldsymbol{u}(k-N)\in \boldsymbol{U}_A} \sup\{N : \phi(k, k_0, \boldsymbol{x}(k_0), \cdot\boldsymbol{u}(k-N)) \in \boldsymbol{X}_A, k_0 \le k \le k_1\}. \tag{6}$$

## III. Derivation of a Hard Deadline

The hard deadline is derived for the controlled process represented by (1). The conditions for asymptotic stability and/or making a controlled process stay in its allowed state-space—which must hold to avoid any dynamic failure—can be used to derive the hard deadline. Let $NT_s$ and $DT_s$ be the assumed maximum and actual maximum delays, respectively. Then, the hard deadline can be obtained by iteratively testing the necessary conditions for system stability and state residence in the allowed state-space while changing $N$ from 1 to $D$.

### *A. Effects of Stationary Occurrence of Delays on System Stability*

When a linear computer-controlled system is (symptotically) stable in the absence of computer failures, we want to derive necessary conditions under which it will remain (asymptotically) stable even in the presence of random failures to the controller computer. These conditions require the knowledge of the system dynamic equations, the control algorithm to be used, and the environmental characteristics.

Consider a simple controlled process represented by a linear time-invariant differential equation, which can be converted to a discrete-time problem by using (2) and a quadratic performance index:

$$J = \frac{1}{2}\left(\sum_{k=0}^{k_f-1} [\boldsymbol{x}^T(k)\boldsymbol{Q}\boldsymbol{x}(k) + \boldsymbol{u}^T(k)\boldsymbol{R}\boldsymbol{u}(k)] + \boldsymbol{x}^T(k_f)\boldsymbol{Q}\boldsymbol{x}(k_f)\right) \tag{7}$$

where the matrix $\boldsymbol{Q} \in \mathcal{R}^{n\times n}$ and $\boldsymbol{R} \in \mathcal{R}^{l\times l}$ are positive semidefinite and positive definite, respectively, and are determined by the control objective of interest. The optimal control input is calculated by minimizing $J$ while maintaining system stability, that is, $\boldsymbol{u}(k) = -\boldsymbol{F}\boldsymbol{x}(k)$, where the state feedback gain matrix $\boldsymbol{F}$ is obtained by solving a discrete Riccati equation [6].

Let the control input have been updated at $t = mNT_s$. If the control inputs were not updated for $i$ sampling periods from that time due to a long computation-time delay, where $0 \le i \le N$, the corresponding state equations for the group of intervals during which the system failed to update the control inputs become

$$\begin{aligned}
\boldsymbol{x}(mN+1) &= \boldsymbol{A}\boldsymbol{x}(mN) + \boldsymbol{B}\boldsymbol{u}(mN)\\
\boldsymbol{x}(mN+2) &= \boldsymbol{A}\boldsymbol{x}(mN+1) + \boldsymbol{B}\boldsymbol{u}(mN)\\
&= \boldsymbol{A}^2\boldsymbol{x}(mN) + (\boldsymbol{A}+\boldsymbol{I})\boldsymbol{B}\boldsymbol{u}(mN)\\
&\vdots\\
\boldsymbol{x}(mN+i) &= \boldsymbol{A}^i\boldsymbol{x}(mN)\\
&\quad + \sum_{j=0}^{i-1} \boldsymbol{A}^j\boldsymbol{B}\boldsymbol{u}(mN)\\
\boldsymbol{x}(mN+i+1) &= \boldsymbol{A}^{i+1}\boldsymbol{x}(mN)\\
&\quad + \sum_{j=1}^{i} \boldsymbol{A}^j\boldsymbol{B}\boldsymbol{u}(mN) + \boldsymbol{B}\boldsymbol{u}(mN+i)\\
&\vdots\\
\boldsymbol{x}((m+1)N) &= \boldsymbol{A}^N\boldsymbol{x}(mN) + \sum_{j=N-i}^{N-1} \boldsymbol{A}^j\boldsymbol{B}\boldsymbol{u}(mN)\\
&\quad + \sum_{j=0}^{N-i-1} \boldsymbol{A}^j\boldsymbol{B}\boldsymbol{u}(mN+N-j-1)
\end{aligned}$$

where $m$ is the time index for the groups of $N$ sampling intervals each. Let $\boldsymbol{X}(m) = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N]^T \equiv [\boldsymbol{x}(mN+1), \boldsymbol{x}(mN+2), \cdots, \boldsymbol{x}((m+1)N)]^T$ and $\boldsymbol{U}(m) = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_N]^T \equiv [\boldsymbol{u}(mN+1), \boldsymbol{u}(mN+2), \cdots, \boldsymbol{u}((m+1)N)]^T$, that is, $\boldsymbol{X}(m)$ and $\boldsymbol{U}(m)$ are respectively the argumented state and control vectors at the group of sampling intervals during which the system failed to update the control inputs (see Fig. 2). When the delay is equal to $i$ sampling periods, the following augmented state equations result:

$$\boldsymbol{X}(m+1) = \boldsymbol{A}_D\boldsymbol{X}(m) + \boldsymbol{B}^1_{D_i}\boldsymbol{U}(m) + \boldsymbol{B}^2_{D_i}\boldsymbol{U}(m+1) \tag{8}$$

$$\boldsymbol{U}(m) = -\boldsymbol{F}_D\boldsymbol{X}(m) \tag{9}$$

where

$$\boldsymbol{A}_D = \begin{bmatrix} \mathbf{o} & \cdots & \mathbf{o} & \boldsymbol{A}\\ \mathbf{o} & \cdots & \mathbf{o} & \boldsymbol{A}^2\\ \vdots & & \vdots & \vdots\\ \mathbf{o} & \cdots & \mathbf{o} & \boldsymbol{A}^N \end{bmatrix}$$

$$\boldsymbol{F}_D = \begin{bmatrix} \boldsymbol{F} & \mathbf{o} & \cdots & \mathbf{o}\\ \mathbf{o} & \boldsymbol{F} & \cdots & \mathbf{o}\\ \mathbf{o} & \mathbf{o} & \ddots & \mathbf{o}\\ \mathbf{o} & \mathbf{o} & \cdots & \boldsymbol{F} \end{bmatrix}$$

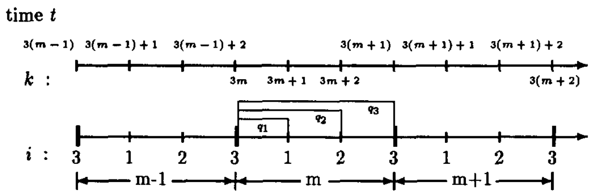


Fig. 2. Time index for newly-defined (argumented) vectors representing the stationary occurrence of computer failures (delays) when $N = 3$.

$$
B^1_{D_i} = \begin{bmatrix}
\mathbf{o} & \cdots & \mathbf{o} & B \\
\mathbf{o} & \cdots & \mathbf{o} & (AB + B) \\
\vdots & \vdots & \vdots & \vdots \\
\mathbf{o} & \cdots & \mathbf{o} & \sum_{j=0}^{i-1} A^j B \\
\mathbf{o} & \cdots & \mathbf{o} & \sum_{j=1}^{i} A^j B \\
\vdots & \vdots & \vdots & \vdots \\
\mathbf{o} & \cdots & \mathbf{o} & \sum_{j=N-i}^{N-1} A^j B
\end{bmatrix}
$$

$$
B^2_{D_i} = \left.\begin{bmatrix}
\mathbf{o} & & \cdots & & \mathbf{o} \\
\vdots & \ddots & \cdots & & \vdots \\
\mathbf{o} & \cdots & \mathbf{o} & \cdots & 0 \\
\mathbf{o} & \cdots & B & \cdots & \mathbf{o} \\
\mathbf{o} & \cdots & AB & \cdots & \mathbf{o} \\
\vdots & \cdots & \vdots & & \vdots \\
\mathbf{o} & \cdots & A^{N-i-1-\delta(i)}B & \cdots \;\; B & \mathbf{o}
\end{bmatrix}\right\}^{N}. \tag{10}
$$

Suppose the occurrence of computer failure(s) is binomially distributed with parameter $P$. Let $q_0, q_1, \cdots, q_N$ be the probabilities of delays 0, $T_s, \cdots, NT_s$, respectively, such that $\Sigma_{i=1}^N q_i = P$, where the maximum delay is assumed to be $NT_s$. This delay distribution can be derived in practice from the knowledge of environmental characteristics. By combining the state equations (2) with the delay distribution, the state equation including the effects of delay $\leq NT_s$ becomes

$$
X(m+1) = A_D X(m) + \sum_{i=0}^{N} \xi_i (B^1_{D_i} U(m) + B^2_{D_i} U(m+1)) \tag{11}
$$

where $\xi_i \in \{0, 1\}$ is a binomially-distributed random variable with parameter $q_i$, i.e., $\Pr[\xi_i = 1] = q_i$. Then, the first moment of (5) is

$$
\overline{X}(m+1) = A_D \overline{X}(m) + \sum_{i=0}^{N} q_i (B^1_{D_i} U(m) + B^2_{D_i} U(m+1)). \tag{12}
$$

Since the period of the index $m$ is $N$, the complex variable $z$ in the $Z$-transform of (6) corresponds to $z_k^N$, where $z_k$ is the complex variable in the $Z$-transforms of equations with index $k$, that is, the period of (12) is $\omega/N$ in the frequency domain, where $\omega$ indicates the period of equations with index $k$ in the frequency domain (subsampling). Using (3) and (12), the characteristic equation of the control system in the presence of stationary occurrences of feedback delay is represented by

$$
\det\left[(I + \sum_{i=0}^{N} q_i B^2_{D_i} F_D) z^N - A_D + \sum_{i=0}^{N} q_i B^1_{D_i} F_D\right] = 0. \tag{13}
$$

The asymptotic stability can be tested with the pole positions of (13) whose characteristic equation reduces to a simple form due to the simple structure of $A_D$ despite its augmented dimension. The characteristic equation of the zero-delay case (i.e., $q_0 = 1$ and thus $P = \Sigma_{i=1}^N q_i = 0$) is

$$
\det[z^N I - (A - BF)^N] = 0. \tag{14}
$$

Further, one can get the following characteristic equation for the worst case in which $q_N = 1$, or the control input is updated only once every $N$ sampling intervals due to the periodic delay of an active duration $NT_s$:

$$
\det\left[z^N I - A^N + \sum_{i=0}^{N-1} A^i BF\right] = \det\left[z^N I - \sum_{i=0}^{N-1} A^i (A - BF) + \sum_{i=1}^{N-1} A^i\right] = 0 \tag{15}
$$

where $A = QDQ^{-1}$ by the similarity transformation if $D = \text{diag}\,[d_1, \cdots, d_n]$ and $d_i$'s are the eigenvalues of $A$. If the system (represented by the transition matrix $A$) without any feedback is unstable—that is, there exists at least one eigenvalue $|d_j| \geq 1$—then the feedback is used to stabilize the system by changing the transition matrix to $A - BF$, which can also be diagonalized by the similarity transformation $R$, i.e., $A - BF = R\Lambda R^{-1}$, where $\Lambda = \text{diag}\,[\lambda_1, \cdots, \lambda_n]$ and $\lambda_i$'s are the eigenvalues of $A - BF$. Then, for example, when $N = 2$:

$$
\begin{aligned}
\text{Eig}[A^2 - (A + I)BF] &= \text{Eig}[(A + I)(A - BF) - A] \\
&= \text{Eig}[QD_1Q^{-1}R\Lambda R^{-1} - QDQ^{-1}]
\end{aligned} \tag{16}
$$

where $D_1 = \text{diag}\,[d_1 + 1, \cdots, d_n + 1]$. The $ij$th element of the matrix $T = A^2 - (A + I)BF$, from which the eigenvalues of the matrix can be calculated numerically, is

$$
[T]_{ij} = \sum_{k=1}^{n} q_{ik} \left( (d_k + 1) \sum_{l,m=1}^{n} \lambda_m q^{-1}_{kl} r_{lm} r^{-1}_{mj} - d_k q^{-1}_{kj} \right) \tag{17}
$$

where $q_{ij}, q^{-1}_{ij}, r_{ij}$, and $r^{-1}_{ij}$ are $ij$th elements of $Q, Q^{-1}, R$, and $R^{-1}$, respectively.

When the assumed maximum delay is *not* an integral multiple of the sampling period, that is, $T_d = (N - 1)T_s + \delta, 0 < \delta < T_s$, all procedures are the same as that of the assumed maximum delay of $NT_s$, except for $i = N$:

$$
x((m+1)N) = A^N x(mN) + \left( \sum_{j=1}^{N-1} A^j B + B_1 \right) u(mN) + B_2 u((m+1)N - 1)
$$

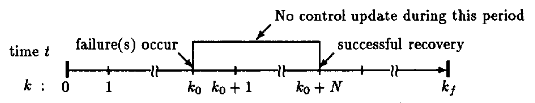


Fig. 3. Time index for the one-shot delay mode.

where $\boldsymbol{B}_1 = \int_0^\delta e^{\boldsymbol{A}_c(T_s-t)}\boldsymbol{B}_c\,dt, \boldsymbol{B}_2 = \int_\delta^{T_s} e^{\boldsymbol{A}_c(T_s-t)}\boldsymbol{B}_c\,dt$. Thus, $\boldsymbol{B}^1_{D_N}$ and $\boldsymbol{B}^2_{D_N}$ in (10) are modified to

$$\boldsymbol{B}^1_{D_N} = \begin{bmatrix} \mathbf{o} & \cdots & \mathbf{o} & \boldsymbol{B} \\ \mathbf{o} & \cdots & \mathbf{o} & \boldsymbol{AB}+\boldsymbol{B} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{o} & \cdots & \mathbf{o} & \sum_{j=0}^{N-2} \boldsymbol{A}^j\boldsymbol{B} \\ \mathbf{o} & \cdots & \mathbf{o} & \sum_{j=1}^{N-1} \boldsymbol{A}^j\boldsymbol{B}+\boldsymbol{B}_1 \end{bmatrix}$$

$$\boldsymbol{B}^2_{D_N} = \begin{bmatrix} \mathbf{o} & \cdots & \mathbf{o} & \mathbf{o} \\ \vdots & & \vdots & \vdots \\ \mathbf{o} & \cdots & \boldsymbol{B}_2 & \mathbf{o} \end{bmatrix}. \tag{18}$$

### B. The Hard Deadline Derived from State Constraints Using a One-Shot Delay Model

The pole locations do not change in case of only one failure with a relatively long ($> T_s$) active period (Fig. 3). The (asymptotic or global) stability condition discussed thus far is therefore no longer applicable. Instead, the terminal state constraints can be used to test whether or not the system leaves its allowed state-space. Note that every critical process must operate within the state-space circumscribed by given constraints, i.e., the allowed state-space. When the control input is not updated for a period exceeding the hard deadline, the system may leave the allowed state-space, thus causing a dynamic failure. The allowed state-space consists of two sets of states $\boldsymbol{X}^1_A$ and $\boldsymbol{X}^2_A$ defined as follows:

- $\boldsymbol{X}^1_A$: the set of states in which the system must stay to avoid an immediate dynamic failure, e.g., a civilian aircraft flying upside down is viewed as an immediate dynamic failure. This set can usually be derived *a priori* from the physical constraints.
- $\boldsymbol{X}^2_A$: the set of states that can lead to meeting the terminal constraints with appropriate control inputs. This set is determined by the terminal constraints, the dynamic equation, and the control algorithm used.

The system must not leave $\boldsymbol{X}^1_A$ nor $\boldsymbol{X}^2_A$ in order to prevent catastrophic failure.

Let $k_0, k_f$, and $N$ denote the indices for the delay/failure occurrence time, the mission completion time, and the period of delay measured in sampling periods, respectively. Then, the dynamic equation of a one-shot delay model is

$$\boldsymbol{x}(k+1) = \boldsymbol{Ax}(k) + \boldsymbol{B}[\boldsymbol{u}(k) + (\boldsymbol{u}(k_0) - \boldsymbol{u}(k))\Pi_{k_0}(N)] \tag{19}$$

where $\Pi_{k_0}(N)$ is a rectangular function from $k_0$ to $k_0+N$, that is, $\Pi_{k_0}(N) = \xi(k-k_0) - \xi(k-k_0-N)$ where $\xi$ is a unit step function.

To test if the constraints at time $k_0+N$ and $k_f$ are met, one must derive $\boldsymbol{x}(k_0+N)$ and $\boldsymbol{x}(k_f)$ by solving (19):

$$\boldsymbol{x}(k_0+N) = \boldsymbol{A}^N\boldsymbol{x}(k_0) + \sum_{i=k_0}^{k_0+N-1} \boldsymbol{A}^{k_0+N-i-1}\boldsymbol{Bu}(k_0) \tag{20}$$

$$\begin{aligned} \boldsymbol{x}(k_f) &= \boldsymbol{A}^{k_f-k_0}\boldsymbol{x}(k_0) + \sum_{i=k_0}^{k_0+N-1} \boldsymbol{A}^{k_f-i-1}\boldsymbol{Bu}(k_0) \\ &\quad + \sum_{i=k_0+N}^{k_f-1} \boldsymbol{A}^{k_f-i-1}\boldsymbol{Bu}(i) \\ &= \boldsymbol{A}^{k_f-k_0-N} \\ &\quad \cdot \left(\boldsymbol{A}^N\boldsymbol{x}(k_0) + \sum_{i=k_0}^{k_0+N-1} \boldsymbol{A}^{k_0+N-i-1}\boldsymbol{Bu}(k_0)\right) \\ &\quad + \sum_{i=k_0+N}^{k_f-1} \boldsymbol{A}^{k_f-i-1}\boldsymbol{Bu}(i) \\ &= \boldsymbol{A}^{k_f-k_0-N}\boldsymbol{x}(k_0+N) \\ &\quad + \sum_{i=k_0+N}^{k_f-1} \boldsymbol{A}^{k_f-i-1}\boldsymbol{Bu}(i). \end{aligned} \tag{21}$$

Let $\Delta = (T_s)/(M)$ for a certain $M$ value determined according to the required accuracy of analysis and let $\boldsymbol{X}^f_A$ be the set of the states at time $k_f$ representing the terminal constraints. When the delay is equal to $T_d = (N-1)T_s + \Delta$, the states at time $k_0T_s + T_d$ and $k_fT_s$ are obtained in the same way as (20) and (21):

$$\begin{aligned} \boldsymbol{x}(k_0T_s+T_d) &= \boldsymbol{A}^N\boldsymbol{x}(k_0T_s) \\ &\quad + \sum_{i=k_0}^{k_0+N-1} \boldsymbol{A}^{k_0+N-i-2}\boldsymbol{Bu}(k_0T_s) \\ &\quad + \int_{(k_0+N-1)T_s}^{k_0T_s+T_d} e^{\boldsymbol{A}_c(T_s-t)}\boldsymbol{B}_c\boldsymbol{u}(k_0T_s)\,dt \\ &\quad + \int_{k_0T_s+T_d}^{(k_0+N)T_s} e^{\boldsymbol{A}_c(T_s-t)} \\ &\quad \cdot \boldsymbol{B}_c\boldsymbol{u}((k_0+N-1)T_s)\,dt \end{aligned} \tag{22a}$$

$$\begin{aligned} \boldsymbol{x}(k_fT_s) &= \boldsymbol{A}^{k_f-k_0-N}\Bigg[E^{\boldsymbol{A}_c(T_s-\Delta)}\boldsymbol{x}(k_0T_s+T_d) \\ &\quad + \int_\Delta^{T_s} e^{\boldsymbol{A}_c(T_s-t)}\boldsymbol{B}_c\boldsymbol{u}((k_0+N-1)T_s)\,dt\Bigg] \\ &\quad + \sum_{i=k_0+N}^{k_f-1} \boldsymbol{A}^{k_f-i-1}\boldsymbol{Bu}(iT_s). \end{aligned} \tag{22b}$$

Then, the pseudocode, shown at the bottom of the page, can be used to derive the hard deadline for the system with the initial state $\boldsymbol{x}_0$ at time $k_0$ where $\boldsymbol{x}_f \in \boldsymbol{X}^f_A$ can be tested indirectly by the following relation:

$$\boldsymbol{x}(k_f) \in \boldsymbol{X}^f_A \Leftrightarrow \boldsymbol{x}(k_0+N) \in \boldsymbol{X}^2_A \tag{23}$$

where

$$\boldsymbol{X}_A^2 = \left\{ \boldsymbol{x} \left| \left[ \boldsymbol{A}^{k_f - k_0 - n}\boldsymbol{x} + \sum_{i=k_0+N}^{k_f-1} \boldsymbol{A}^{k_f-i-1}\boldsymbol{Bu}(i) \right] \in \boldsymbol{X}_A^f \right. \right\} \tag{24}$$

results.

In practice, it is difficult to obtain $\boldsymbol{X}_A^2$. Although there may be a one-to-one mapping between $\boldsymbol{x}(k_0+N)$ and $\boldsymbol{x}(k_f), \boldsymbol{X}_A^f$ is usually a continuum, which requires an excessive amount of computation due to the curse of dimensionality. The size of $\boldsymbol{X}_A^2$ will decrease as either $N$ increases or $k$ approaches $k_f$, but the size of $\boldsymbol{X}_A^2$ is usually larger than that of $\boldsymbol{X}_A^f$ due to the (asymptotic) stability of a controlled process.

## IV. Examples and Numerical Results

To demonstrate the concept of hard deadline, we derive the deadlines for several simple, yet practical, example control systems. The first two examples calculate hard deadlines via stability analysis.

*Example 1:* Consider a simple controlled process:

$$x(k+1) = 1.05x(k) + 1.8u(k)$$

where $\boldsymbol{Q} = 2, \boldsymbol{R} = 7$. This system is unstable without any feedback control but is controllable. The optimal feedback control is given by

$$u(k) = [R^{-1}B^TP^{-1}(k+1) + BP^{-1}(k+1)B^T]^{-1}Ax(k) = Fx(k) \tag{25}$$

where $P(k)$ is the solution of the discrete Riccati equation:

$$P(k) = Q + A^TP(k+1)[I + BR^{-1}B^TP(k+1)]^{-1}A$$

where $P(k_f) = 0$. Hence, we obtain a steady-state feedback gain $F = -0.3594$ by plugging $P(k) = P(k+1)$ into (24), that is, $u(k) = -0.3594x(k)$ as $k \to \infty$. The system is stabilized by the feedback control that results from minimizing the performance index $J$ (7), since the pole position $\lambda$ is changed from 1.05 to 0.4013. The relation between the pole position and $N$ for the worst case[2] is given in Table I, yielding the hard deadline $D = 4$ since the pole moves outside the unit circle beginning at $D = 4$. The pole position is affected by the probability distribution of delays as well as the magnitude of the maximum delay ($NT_s$), which is shown in Table II where $q_1 = 1 - P, q_2 = P(1-\beta)$, and $q_3 = P\beta$. Since the pole in the worst case or when $q_N = 1$ is located at 0.9589 for $N = 3$, the pole in all other cases (i.e., $q_3 < 1$), for example $\lambda = 0.9033$ when $P = q_2 = 1, q_3 = 0$, must reside inside of the unit circle. However, the pole approaches the unit circle quickly as the probability of a large delay increases.

*Example 2:* For a more practical example of stationary occurrence of delays/failures, the dynamic behavior of the altitude of a spinning satellite is described in terms of the long-term control of the roll ($\varphi$) and yaw ($\psi$) angles, which is based on the dynamic coupling resulting from the rotation

[2]The "worst case" means the *periodic* occurrence of the largest delay possible, that is, deterministically $q_N = 1$, while updating the control input once every $N$ sampling intervals.

```
\*Recursive testing the allowed state space while increasing the delay $T_d^*$\
$m = 1$    \*subsampling period, $m\Delta$*\
$D$=integer_time($N_{\lim}$)    \*integer times of $T_s^*$\
if  ($D \leq N_{\lim}$)  then
  while($m \neq T_s/\Delta$ or test_constraints() $\neq$ TRUE) do
    $T_d$ := $(D-1)T_d + m\Delta$
    if test_constaints ($T_d$) then return    $T_d$\*hard deadline\, is $T_d^*$\
    else $m : m = 1$    \*test\, larger $T_d$ by increasing $m$*\
  end_while
else return no hard deadline

integer_time($N_{\lim}$)
$N = 1$
while ($N \neq N_{\lim}$ or test_constaints() $\neq$ TRUE) do
  $T_d := NT_s$
if test_constaints ($T_d$) then return $N$ \*$D = N_{\max}$ is obtained*\
else $N$ :=$N+1$ \ test larger $N$*\
end_while
test_constraints($T_d$)
compute $\boldsymbol{x}((k_0+N)T_s)$ from $\boldsymbol{x}_0$
if ($\boldsymbol{x}((k_0+N)T_s) \in \boldsymbol{X}_A^1$) then
  begin
    compute $\boldsymbol{x}(k_fT_s)$ from either $\boldsymbol{x}((k_0+N)T_s)$ or $\boldsymbol{x}_0$
    if ($\boldsymbol{x}(k_fT_s) \in \boldsymbol{X}_A^f$) then return FAILURE
    else return TRUE
  end
else returnTRUE
```

TABLE I
RELATION BETWEEN THE POLE POSITION AND $N$ WHEN $P = q_N = 1$

| $N$ | $\|\lambda\|$ |
|---|---|
| 2 | 0.4729 |
| 3 | 0.9589 |
| 4 | 1.1198 |
| 5 | 1.1811 |
| 6 | 1.2049 |
| 7 | 1.2128 |

TABLE II
RELATION BETWEEN THE POLE POSITION AND THE PROBABILITY DISTRIBUTION OF DELAYS WHEN $N = 3$

| $P = 0.3$ | | $P = 0.7$ | |
|---|---|---|---|
| $\beta$ | $\|\lambda\|$ | $\beta$ | $\|\lambda\|$ |
| 0 | 0.5596 | 0 | 0.7917 |
| 0.1 | 0.5642 | 0.1 | 0.7970 |
| 0.3 | 0.5731 | 0.3 | 0.8076 |
| 0.5 | 0.5818 | 0.5 | 0.8178 |
| 0.7 | 0.5903 | 0.7 | 0.8255 |
| 0.9 | 0.5985 | 0.9 | 0.8375 |
| 1 | 0.6024 | 1 | 0.8423 |

TABLE III
MAXIMUM MAGNITUDE OF EIGENVALUES

| $N$ | $\|\lambda_{\max}\|$ |
|---|---|
| 5 | 0.3017 |
| 6 | 0.5219 |
| 7 | 0.7732 |
| 8 | 1.0381 |
| 9 | 1.3055 |
| 10 | 1.5682 |

of the satellite around the earth:

$$\dot{\varphi}_x = 2.1\varphi_x + 1.5\psi_z + 0.1u_x + 0.2u_z \qquad (26)$$
$$\dot{\psi}_z = -2.3\psi_z + 0.2u_x + 0.1u_z \qquad (27)$$

where the coefficients depend upon the orbital frequency, i.e., the angular velocity of the satellite with respect to the inertial frame, and $u_x$ and $u_z$ are control signals. The goal of the control is to maintain a desired orientation of the satellite in the orbit around the earth (the stabilization problem) with the minimum-control effort, which can be represented by a quadratic performance index (7) of the state and the control input. Discrete state equations are derived from (26) and (27) with $T_s = 1$ s. The corresponding coefficient matrices are then

$$\boldsymbol{A} = \begin{bmatrix} 8.1660 & 2.7490 \\ 0 & 0.1003 \end{bmatrix}$$
$$\boldsymbol{B} = \begin{bmatrix} 0.5470 & 0.7855 \\ 0.0782 & 0.0391 \end{bmatrix}$$
$$\boldsymbol{Q} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$
$$\boldsymbol{R} = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}.$$

One can derive the optimal (feedback) control gain matrix, $\boldsymbol{F}$:

$$\boldsymbol{F} = \begin{bmatrix} 4.7623 & 1.6249 \\ 6.6508 & 2.2661 \end{bmatrix}.$$

This feedback control changes the eigenvalues from $\{8.166, 0.1003\}$ to $\{0.1102 \pm 0.0045j\}$, thus stabilizing the satellite. Let the rate of failure occurrence be 1/500 per hour. Then, the change of poles as a result of incrementing $N$ is derived for the occurrence of the largest delay possible ($P = q_N = 5.556 \times 10^{-7}$) and is given in Table III. Since the hard deadline $D = 8T_s$, the controller computer must have some mechanism of fault-tolerance, which can recover from any controller-computer failure within $8T_s$ in order not to lose the (asymptotic) stability in rotating the satellite. Whereas the previous examples calculated hard deadlines via stability, the following examples will calculate hard deadlines using state constraints.

*Example 1:* Consider the system of a double integrator [3], whose sampled model with $T_s = 0.01s$ is

$$\boldsymbol{x}(k+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \boldsymbol{x}(k) + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u(k)$$

where the constraints on the control and state for $k_0 \le k \le k_f$ are given by

$$\boldsymbol{\Omega} = \{u(k)\colon -1 \le u(k) \le 1\}$$
$$\boldsymbol{X}_A^1 = \{(x_1(k), x_2(k))\colon -25 \le x_1(k) \le 25, -5 \le x_2(k) \le 5\}.$$

The terminal state must belong to the set $\boldsymbol{X}_A^f = \{(x_1, x_2)\colon |x_i| \le 0.2, i = 1, 2\}$. From these constraints, one can find a simple (nonmaximal) $\Omega$-invariant set $\boldsymbol{X}$, which is a polyhedron defined by the vertices: $v_1 = (0, 5), v_2 = (25, 0), v_3 = (25, -5), v_i = -v_{i-3}, i = 4, 5, 6$. A set $\boldsymbol{X}$ is said to be $\Omega$-invariant with respect to $\boldsymbol{G}$ if $\boldsymbol{X} \subseteq \boldsymbol{G}$ such that (i) $\forall \boldsymbol{x}(k_0) = x_0 \in \boldsymbol{X}, \exists u(k) \in \Omega \forall k$, such that if $\boldsymbol{x}(k_0) = \boldsymbol{x}_0$ then $\boldsymbol{x}(k) \in \boldsymbol{G} \forall k$ and $\lim_{k\to\infty} \boldsymbol{x}(k) = 0$, and $\boldsymbol{x}(k) \in \boldsymbol{X} \forall k$. An $\Omega$-invariant set clearly belongs to the allowed state-space $(\boldsymbol{X}_A^1 \cap \boldsymbol{X}_A^2)$, since the state $\boldsymbol{x}(k) \forall k \in [k_0, k_f]$ must stay in the given state constraint set $\boldsymbol{G}$ and satisfy the terminal condition $\boldsymbol{x}(k_f) \in \boldsymbol{X}_A^f$ by the convergence property of $\boldsymbol{x}(k)$. A constant feedback control input was simply derived by using Theorem 3.1 in [3]:

$$u(k) = -0.04x_1(k) - 0.2x_2(k). \qquad (28)$$

The state constraint $\boldsymbol{X}_A^1$, $\Omega$-invariant set $\boldsymbol{X}$ with respect to $\boldsymbol{X}_A^1$, and the state trajectory in the absence of delay, and the state trajectories in the presence of one-shot delay equal to the hard deadline with and without terminal constraints are plotted in Fig. 4, where the curves 1–5 indicate $\boldsymbol{X}_A^1, \boldsymbol{X}$ (or an $\Omega$-invariant set), state trajectory in the absence of delay, state trajectories in the presence of delay equal to the hard deadline with terminal constraints, and without terminal constraints, respectively.

The hard deadlines associated with the states on the trajectory in the absence of delay are derived by the algorithm in Section III for both cases with and without terminal constraints, and are plotted in Fig. 5.

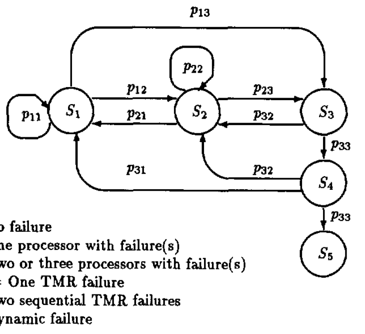


Fig. 4. State trajectory in the absence of delay, and the state trajectories in the presence of one-shot delay equal to the hard deadline with and without terminal constraints.

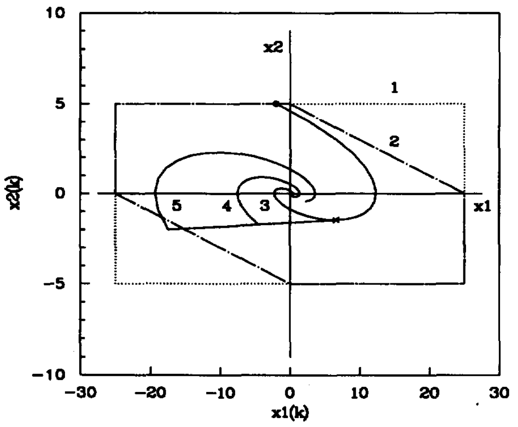


Fig. 5. Hard deadlines of the state trajectory 3 of Fig. 4 in the absence of delay, 1) with terminal constraints, and 2) without terminal constraints.

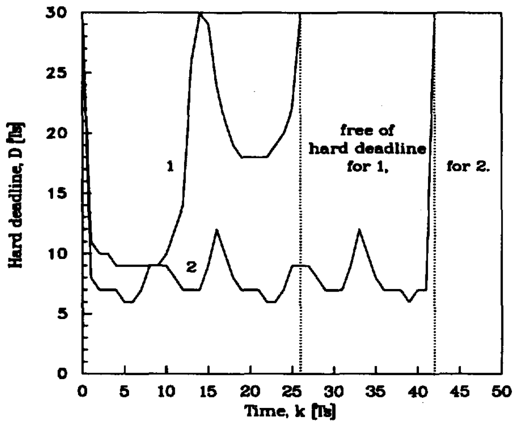


Fig. 6. Hard deadlines of the state trajectory 3 of Fig. 5 in the absence of delay, (1) with terminal constraints and (2) without terminal constraints.

In Fig. 6, the hard deadlines on the subset, $\{5 \le x_1 \le 15; (a) -5 \le x_2 \le -1$ and (b) $0 \le x_2 \le 2\} \subset \boldsymbol{X}$, are derived under the assumption that the remaining mission time is $38T_s$ for all states in the subset.

*Example 2:* The physical meaning of a hard deadline based on the one-shot delay model can be explained with the real-time controller of a robotic manipulator, where the obstacles in the robot's workplace are translated into state constraints. That is, the system states (robot's positions) must be constrained to avoid collision with the obstacles. In addition to these state constraints, there are usually control input constraints due to the bounds on joint motor torques and energy. In [5], a point robot of Cartesian-coordinate class was modeled by a set of decoupled double integrators:

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} 0 & \boldsymbol{I} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{v} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \boldsymbol{I} & 0 \end{bmatrix} \begin{bmatrix} 0 \\ \boldsymbol{u} \end{bmatrix}$$

where $\boldsymbol{x}, \boldsymbol{y}$, and $\boldsymbol{u}$ are two-dimensional position, velocity, and control vectors, respectively, and $\boldsymbol{I} \in \mathcal{R}^{2\times 2}$. The robot's end-effector is guided by the control input determined by

$$\min_{\boldsymbol{u}} \|\dot{\boldsymbol{v}}_d - \dot{\boldsymbol{v}}\|_2 \tag{29}$$

subject to the control constraints $|u_i| \le u_i^{\max}$ for $i = 1, 2$, and the state constraints specified by the obstacles. The *obstacle avoidance strategy* (OVS) used in [5] considers the dynamic environment and real-time control needs, where the obstacle-related state constraints are transformed into state-dependent control constraints by mapping each end-effector's position relative to an obstacle into the $P$-functionals:

$$P = (\boldsymbol{x}^T\boldsymbol{x})^k - (r^2)^k$$

where the obstacle is assumed to be centered at the origin and covered by a circle of radius $r$. The $k$ in the $\boldsymbol{P}$-functional most be chosen such that the set of permissible controls remains nonempty, and the system state and each obstacle's position determine $\boldsymbol{x} = [x_1 - a, x_2 - b]^T$, where $(a, b)$ is the coordinate of a 2-D obstacle. According to Theorem 1 in [5], $k$ is set to 0.5 so that the control input constraint set may remain nonempty oer the duration from the detection of a potential collision to the disappearance of this collision dánger. A collision-free trajectory is guaranteed if $P(t) > 0 \forall t$. A potential collision was detected by checking

$$P(\hat{t}) = P(0) + \hat{t}\dot{P}(0) + \frac{\hat{t}^2}{2}\ddot{P}_{\max} \ge 0 \forall t \Rightarrow \ddot{P}_{\max} \ge \frac{(\dot{P}(0))^2}{2P(0)} \tag{30}$$

where $\hat{t} = -\dot{P}(0)/\ddot{P}_{\max}$ is the time at which the minimum of $P$ occurs. If the condition (30) is violated, the instantaneous value of $\ddot{P}_{\max}$ is calculated, and an additional constraint (31) is included until $\dot{P}$ becomes positive, i.e., the danger of collision disappears.

$$\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{v}) = \{\boldsymbol{u}: \boldsymbol{P}_{\boldsymbol{x}}^T\boldsymbol{u} + \boldsymbol{v}^T\boldsymbol{P}_{\boldsymbol{xx}}^T\boldsymbol{v} \ge \ddot{P}_{\max}\} \tag{31}$$

where $\boldsymbol{P_x} = (\boldsymbol{x}^T\boldsymbol{x})^{1/2}\boldsymbol{x}, \boldsymbol{P_{xx}} = (m(\boldsymbol{x})\boldsymbol{I})/(\boldsymbol{x}^T\boldsymbol{x}) - (\boldsymbol{x}^T\boldsymbol{x})^{-1/2}$, and $\boldsymbol{m}(\boldsymbol{x}) = 2k(\boldsymbol{x}^T\boldsymbol{x})^{k-1}$. The intersection of these constraints with the admissible control set $\boldsymbol{\Omega}$ results in a polygonal control space:

$$\boldsymbol{\Lambda} = \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{v}) \cap \boldsymbol{\Omega}.$$

The optimal decision strategies (ODS) in [7] can be used to solve such a constrained minimization problem, similarly to a class of pointwise-optimal control laws constrained by hard control bounds. The hard deadline is derived using a pointwise-optimal control law with the OVS. Since the computation-time delay causes the failure to update the control input, a collision (or a dynamic failure) occurs if the computation-time delay is longer than a certain threshold, which is a function of the system state and time. In this example, the state constraints change with system state (time), i.e., the state-dependent control constraints. Thus, the control input must be updated on the basis of new information to avoid any collision.

The trajectories derived in both the absence and presence of delay $DT_s$, and the hard deadlines on these trajectories in the absence of delay are plotted in Figs. 7 and 8.

## V. Application of Hard Deadline Information

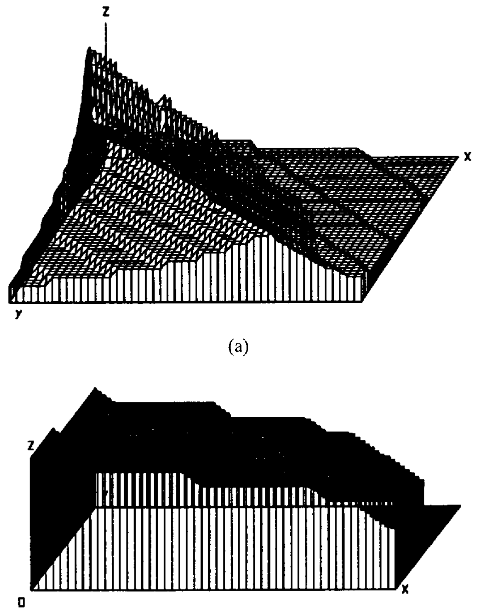
The information on hard deadlines is very useful for modeling system reliability and designing both the hardware and software for a controller computer. When designing a controller computer, one has to make many design decisions in the context of controlled processes that are characterized by their hard deadlines and cost functions [10], including:

- hardware design issues dealing with the number of processors and the type of interconnection network to be used, and how to synchronize the processors,
- software design issues related to the implementation of control algorithms, task assignment and scheduling, redundancy management, error detection and recovery.

From the hard deadline information, one can deduce the knowledge of system inertia, which can, in turn, be used to specify the fault-tolerance requirement of a real-time control system. This knowledge is required to estimate systems' ability of meeting timing constraints in the presence of controller-computer failures, which was characterized in [11] as the *probability of dynamic failure,* $P_{\text{dyn}}$.

To illustrate the general idea of applying the knowledge of system inertia, let us consider a simple example of a triple modular redundant (TMR) controller computer in which three identical processors execute the same set of cyclic tasks. The TMR controller computer updates, once every $T_s$ s or every sampling period, the control input to the controlled process (plant). That is, the period of each cyclic task is equal to $T_s$. The input of the cyclic task is a discretized output of the plant and the output of the cyclic task will be used to control the plant during the next sampling interval. The output of the TMR controller is correct for each task only if at least two of the three processors in the TMR controller produce correct outputs. A *TMR failure* is said to occur if more than one processor in the TMR controller fail during $T_s$. Thus, the output of the TMR controller would not be updated in case of a TMR failure. The condition for a system (dynamic) failure resulting from controller-computer failures[3] is derived from the hard deadline, which is the allowable maximum computation-time delay. In other words, this condition gives us the knowledge about the controlled system's inertia against controller-computer failures.

[3]The other sources of system failure(s), such as failures in actuators or sensors or mechanical parts and failures of A/D and D/A converters, are not considered in this paper, because our main intent is to analyze the coupling between a controlled process and a (fault-tolerant) controller computer.



(b)

Fig. 7. Hard deadlines ($DT_s$) of the region $\{5 \geq x_1 \geq 15. -5 \geq x_2 \geq -1\}$ ($X = x_1, Y = x_2, Z = N$), where the top and bottom values of $Z$-axis are $24T_s$ and $8T_s$, respectively. (b) $\{5 \geq x_1 \geq 15, 0 \geq x_2 \geq 2\}$, where the top and bottom values of $Z$-axis are $10T_s$ and $T_s$.

More than 90% of computer failures have been reported to be transient, especially with short active durations [12]. Thus, the controller computer may recover from most failures in a few sampling intervals, and it can correctly update the control input without causing any dynamic failure, if the active duration of controller-computer failure is smaller than the hard deadline.

Suppose the hard deadline derived from the controlled system is three sampling periods and a TMR controller computer is used. That is, no dynamic failure occurs if the faults inducing computer failures disappear (or are recovered by a fault-tolerant mechanism) within three sampling periods. Then, the reliability model for this controller computer (Fig. 9) is built by extending a Markov chain model whose parameters are to be estimated at a given level of confidence from empirical data. The additional states account for the system inertia, i.e., a dynamic failure results from only three consecutive incorrect (missing the update of) outputs of the controller computer or for a period of $3T_s$, not immediately from one or two incorrect (missing the update of) outputs. Without the information of hard deadline, one can overestimate the probability of a system failure under the assumption that the system has no delay-tolerance, i.e., one incorrect output can lead to a dynamic failure.

## VI. Conclusion

The hard deadline for a critical-control task is usually assumed to be given *a priori.* This presupposes the existence of a precise definition of the hard deadline and a method to derive it, which, however, have not been addressed in detail.

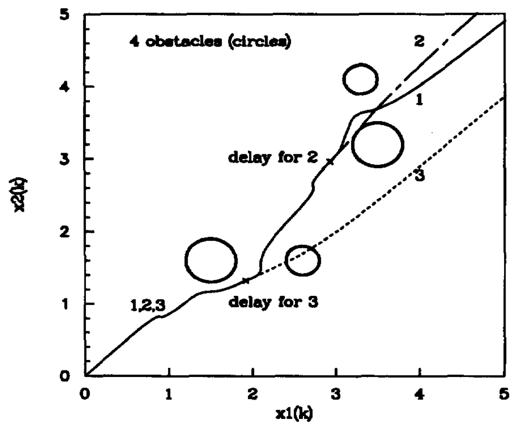


Fig. 8. State trajectories for a point robot with four obstacles, 1) in the absence of delay and 2, 3) in the presence of delay equal to $DT_s$.

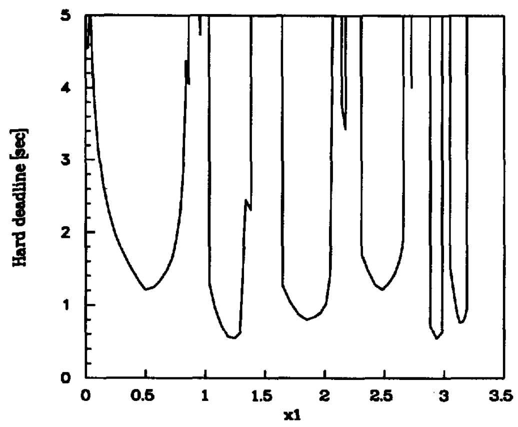


Fig. 9. A Markov reliability model with knowledge of the system inertia.

The knowledge of hard deadlines, which must be derived from real control applications, is very important for task assignment and scheduling, specification and evaluation of fault-tolerant controller computers.

In this paper, hard deadlines are derived for linear time-invariant systems whose dynamic properties and control algorithms are given along with the characteristics of their environment (controller computer's fault behavior). More complex control systems can be approximated by such simple systems. First, when the occurrence of computer failures is stationary or represented by an appropriate probability density function, the hard deadline is obtained as an integer multiple of the sampling period by using the (asymptotic) stability condition for a modified state equation. Second, a heuristic algorithm is proposed to iteratively compute the hard deadline as a function of the system state and time by testing for the given (or derived) constraints of control inputs and states.

## ACKNOWLEDGMENT

The authors would like to thank Allan White, Chuck Meissner, and Felix Pitts of the NASA Langley Research Center, and Jim Smith of the Office of Naval Research for their technical and financial assistance.

## REFERENCES

[1] A. Belleisle, "Stability of systems with nonlinear feedback through randomly time-varying delays," *IEEE Trans. Automat. Contr.*, vol. AC-20, pp. 67–75, Feb. 1975.
[2] A. Gosiewski and A. Olbrot, "The effect of feedback delays on the performance of multivariable linear control systems," *IEEE Trans. Automat. Contr.*, vol. AC-25, no. 4, pp. 729–734, Aug. 1980.
[3] P. Gutman and M. Cwikel, "Admissible sets and feedback control for discrete-time linear dynamic systems with bounded controls and states," *IEEE Trans. Automat. Contr.*, vol. AC-31, no. 4, pp. 373–376, Apr. 1986.
[4] K. Hirai and Y. Satoh, "Stability of a system with variable time delay," *IEEE Trans. Automat. Contr.*, vol. AC-25, no. 3, pp. 552–554, June 1980.
[5] S. Kheradpir and J. S. Thorp, "Real-time control of robot manipulators in the presence of obstacles," *IEEE J. Robotics Automat.*, vol. 4, no. 6, pp. 687–698, Dec. 1988.
[6] G. Leitmann, *An Introduction to Optimal Control.* New York: McGray-Hill, 1969.
[7] D. G. Luenberger, *Optimization by Vector Space Methods.* New York: Wiley, 1969.
[8] Z. Rekasius, "Stability of digital control with computer interruption," *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 356–359, Apr. 1986.
[9] K. G. Shin and X. Cui, "Effects of computing time delay on real-time control systems," in *Proc. 1988 Amer. Contr. Conf.*, 1988.
[10] K. G. Shin, C. M. Krishna, and Y.-H. Lee, "A unified method for evaluating real-time computer controller and its application," *IEEE Trans. Automat. Contr.*, vol. AC-30, pp. 357–366, Apr. 1985.
[11] K. G. Shin and C. Krishna, "New performance measures for design and evaluation of real-time multiprocessors," *Int. J. Comput. Sci. Eng.*, vol. 1, no. 4, pp. 179–191, Oct. 1986.
[12] D. Siewiorek and R. S. Swarz, *The Theory and Practice of Reliable System Design.* Bedford, MA: Digital Equipment Corporation, 1982.
[13] K. Zahr and C. Slivinsky, "Delay in multivariable computer controlled linear systems," *IEEE Trans. Automat. Contr.*, pp. 442–443, Aug. 1974.

**Kang G. Shin** (S'75–M'78–SM'83–F'92) He received the B.S. degree in electronics engineering from Seoul National University, Seoul, Korea, in 1970, and both the M.S. and Ph.D. degrees in electrical engineering from Cornell University, Ithaca, NY, in 1976 and 1978, respectively.

He is a Professor of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, which he joined in 1982. He also chairs the Computer Science and Engineering Division, EECS Department. He has authored and coauthored more than 230 technical papers (more than 100 of these are in archival journals) in the areas of fault-tolerant computing, distributed real-time computing, computer architecture, and robotics and automation. From 1978 to 1982, he was on the faculty of Rensselaer Polytechnic Institute, Troy, NY. He has held visiting positions at the U.S. Air Force Flight Dynamics Laboratory, AT&T Bell Laboratories, Computer Science Division, within the Department of Electrical Engineering and Computer Science at UC Berkeley, and International Computer Science Institute, Berkeley, CA.

Dr. Shin received the Outstanding IEEE TRANSACTIONS ON AUTOMATIC CONTROL Paper Award in 1987 for a paper on robot trajectory planning. In 1989, he also received the Research Excellence Award from the University of Michigan, in 1985, he founded the Real-Time Computing Laboratory, where he and his colleagues are building a 19-node hexagonally mesh multicomputer a called Harts, to validate various architectures and analytic results in the area of distributed real-time computing. He was the Program Chair of the 1986 Real-Time Systems Symposium (RTSS), the General Chairman of the 1987 RTSS and the Guest Editor of the Special Issue on Real-Time Systems of the IEEE TRANSACTIONS ON COMPUTERS in Aug. 1987. He is a Distinguished Visitor of the Computer Society of the IEEE, an Editor for IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, and an Area Editor of *International Journal of Time-Critical Computing Systems.*

**Hagbae Kim** (S'88–M'90) received the B.S. degree from in electronics engineering from Seoul National University, Seoul, Korea, in 1988, and the M.S. degree in electrical engineering and computer science from the University of Michigan, Ann Arbor, in 1990. Currently, he is working toward the Ph.D. degree in electrical engineering and computer science at the University of Michigan, Ann Arbor. His research interests include real-time control systems, reliability modeling, and fault-tolerant computing.