Designing Temporal Controls *†

Ashok K. Agrawala   Seonho Choi
Institute for Advanced Computer Studies
Department of Computer Science
University of Maryland
College Park, MD 20742
{agrawala, seonho}@cs.umd.edu

Leyuan Shi
Department of Industrial Engineering
University of Wisconsin
Madison, WI 53706
leyuan@ie.engr.wisc.edu

Abstract

Traditional control systems have been designed to exercise control at regularly spaced time instants. When a discrete version of the system dynamics is used, a constant sampling interval is assumed and a new control value is calculated and exercised at each time instant. In this paper we formulate a new control scheme, *temporal control*, in which we not only calculate the control value but also decide the time instants when the new values are to be used. Taking a discrete, linear, time-invariant system, and a cost function which reflects a cost for computation of the control values, as an example, we show the feasibility of using this scheme. We formulate the temporal control scheme as a feedback scheme and, through a numerical example, demonstrate the significant reduction in cost through the use of temporal control.

*This work is supported in part by ONR and DARPA under contract N00014-91-C-0195 to Honeywell and Computer Science Department at the University of Maryland. The views, opinions, and/or findings contained in this report are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, ONR, the U.S. Government or Honeywell. Computer facilities were provided in part by NSF grant CCR-8811954.

†This work is supported in part by ARPA and Philips Labs under contract DASG-60-92-C-0055 to Department of Computer Science, University of Maryland. The views, opinions, and/or findings contained in this report are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the Advanced Research Projects Agency, PL, or the U.S. Government.

1 Introduction

Control systems have been used for the control of dynamic systems by generating and exercising control signals. Traditional approach for feedback controls has been to define the control signals, $u(t)$, as a function of the current state of the system, $x(t)$. As the state of the system changes continuously the controls change continuously, i.e. they are defined as functions of time, t, such that time is treated as a continuous variable. When computers are used for implementing the control systems, due to the discrete nature of computations, time is treated as a discrete variable obtained by regularly spaced sampling of the time axis at Δ seconds. Many standard control formulations are defined for the discrete version of the system, with system dynamics expressed at discrete time instants. In these formulations the system dynamics and the control are expressed as sequences, $x(k)$ and $u(k)$.

Most of the traditional control systems were designed for dedicated controllers which had only one function, to accept the state values, $x(k)$ and generate the control, $u(k)$. However, when a general purpose computer is used as a controller, it has the capabilities, and may, therefore, be used for other functions. Thus, it may be desirable to take into account the cost of computations and consider control laws which do not compute the new value of the control at every instant. When no control is to be exercised, the computer may be used for other functions. In this paper we formulate such a control law and show how it can be used for control of systems, achieving the same degree of control as traditional control systems while reducing computation costs by changing the control at a few, specific time instants. We term this *temporal control*.

To the best of our knowledge this approach to the design and implementation of controls has not been studied in the past. However, taking computation time delay into consideration for real-time computer control has been studied in several research papers [1, 5, 6, 9, 11, 13]. But, all of these papers concentrated on examining computation time delay effects and compensating them while maintaining the assumption of exercising controls at regularly spaced time instants.

The basic idea of temporal control is to determine not only the values for u but also the time instants at which the values are to be calculated and changed. The control values are assumed to remain constant between changes. By exercising control over the time instants of changes the designer has an additional degree of freedom for optimization. In this paper we present the idea and demonstrate its feasibility through an example using a discrete, linear, and time invariant system. Clearly, the same idea can be extended to continuous time as well as non-linear system.

The paper is organized as follows. In Section 2, we formulate the temporal control problem and introduce computation cost into performance index function. The solution approach for temporal control scheme is discussed in *Section 3*. In Section 4, implementation issues are addressed. We provide an example of controlling rigid body satellite in Section 5 . In this example, an optimal temporal controller is designed. Results show that the temporal control approach performs better than the traditional sampled data control approach with the same number of control exercises. Section 6 deals with the application of temporal controls to the design of real-time control systems. Finally, Section 7, we present our conclusions.

2 Problem Formulation

In temporal control, the number of control changes and their exercising time instants within the controlling interval $[0, T_f]$ is decided to minimize a cost function. To formulate the temporal control problem for a discrete, linear time-invariant system, we first discretize the time interval $[0, T_f]$ into M subintervals of length $\Delta = T_f/M$. Let $D_M = \{0, \Delta, 2\Delta, \ldots, (M-1)\Delta\}$ which denote M time instants which are regularly spaced. Here, control exercising time instants are restricted within D_M for the purpose of simplicity. The linear time-invariant controlled process is described by the difference equation:

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

where k is the time index. One unit of time represents the subinterval Δ, whereas $x \in \mathcal{R}^n$ and $u \in \mathcal{R}^l$ are the state and input vectors respectively.

It is well known that there exists an optimal control law [4]

$$u^o(i) = f[x(i)] \;\; i = 0, 1, ..., M-1 \tag{2}$$

that minimizes the quadratic performance index function (Cost)

$$J_M = \sum_{k=0}^{M-1} [x^T(k)Qx(k) + u^T(k)Ru(k)] + x^T(M)Qx(M) \tag{3}$$

where $Q \in \mathcal{R}^{n \times n}$ is positive semi-definite and $R \in \mathcal{R}^{l \times l}$ is positive definite.

As we can see, traditional controller exercises control at every time instant in D_M. However, in temporal control, we are no longer constrained to exercise control at every time instant in D_M. Therefore, we want to find an optimal control law, δ and g for $i = 0, 1, ..., M-1$:

$$\begin{aligned} u^o(i) &= u^o(i-1) \;\; if \; \delta(i) = 0 \\ u^o(i) &= g[x(i)] \;\; if \; \delta(i) = 1 \end{aligned} \tag{4}$$

that minimizes a new performance index function

$$\begin{aligned} J_M^{'} &= \sum_{k=0}^{M-1} [x^T(k)Qx(k) + u^T(k)Ru(k)] + x^T(M)Qx(M) + \sum_{k=0}^{M-1} \delta(k)\mu \\ &= J_M + C_M \end{aligned} \quad (5)$$

Here, μ is the computation cost of getting a new control value at a time instant, and $C_M = \sum_{k=0}^{M-1} \delta(k)\mu$ denotes the total computation cost. Note that $\nu = \sum_{k=0}^{M-1} \delta(k)$ is the number of control changes. Also, let $D_\nu = \{t_0, t_1, t_2, \ldots, t_{\nu-1}\}$ consist of control changing time instants where $t_0 = 0$, $t_1 = n_1\Delta$, ..., $t_{\nu-1} = n_{\nu-1}\Delta$. That is, $n_0, n_1, n_2, \ldots, n_{\nu-1}$ are the indices for control changing time instants and $\delta(n_i) = 1$ for $i = 0, 1, 2, \ldots \nu - 1$.

With this new setting we need to choose ν, D_ν, and control input values to find an optimal controller which minimizes $J_M^{'}$. This new cost function is different from J_M in two aspects. First, the concept of computational cost is introduced in $J_M^{'}$ as C_M term to regulate the number of control changes chosen. If we do not take this computation cost into consideration ν is likely to become M. If computation cost is high (i.e., μ has a large value) then ν is likely to be small in order to minimize the total cost function. Second, in temporal control, not only do we seek optimal control law $u(x(t))$, but also the control exercising time instants and the number of control changes. In the next section, we present in detail specific techniques for finding an optimal temporal control law.

3 Temporal Control

We develop a three-step procedure for finding an optimal temporal controller.

Step 1. Find an optimal control law given ν and D_ν
Step 2. Find best D_ν given ν
Step 3. Find best ν

First, in the following two subsections(3.1 and 3.2) we derive a temporal control law which minimizes the cost function $J_M^{'}$ when D_ν is given, i.e., both time instants and number of controls are fixed. Since ν and D_ν are fixed we can use J_M defined in (5) as a cost function instead of $J_M^{'}$. Secondly, assume that ν is fixed but D_ν can vary. Then we present an algorithm in section 3.3 to find a D_ν^o such that J_M (and $J_M^{'}$) is minimized. Finally, we will vary ν from 1 to ν_{max} to search an optimal D_ν^o at which temporal control should be exercised. Section 3.4 presents this iteration procedure. Section 3.5 explains how to incorporate terminal state constraints into the above procedure of getting an optimal temporal control law. And a complete algorithm of the above procedure is described in Section 3.6. Finally, in Section 3.7 we explain how to get optimal temporal controllers over an initial state space.

3.1 Closed-loop Temporal Control with D_ν Given

Assume that ν and D_ν are given. Then a new control input calculated at t_i will be applied to the actuator for the next time interval from t_i to t_{i+1}. Our objective here is to determine the optimal control law

$$u^o(n_i) = g[x(n_i)] \quad i = 0, 1, ..., \nu - 1 \tag{6}$$

that minimizes the quadratic performance index function (Cost) J_M which is defined in (5).

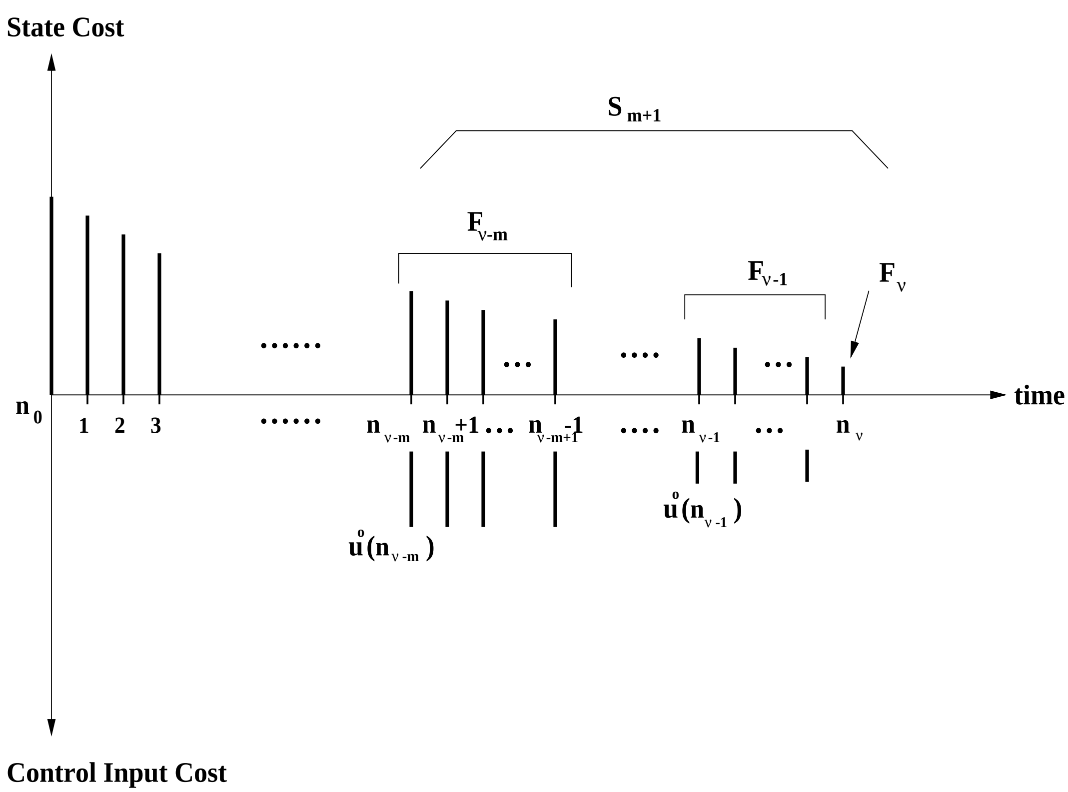


Figure 1: Decomposition of J_M into F_i.

The principle of optimality, developed by Richard Bellman[2, 3] is the approach used here. That is, if a closed loop control $u^o(n_i) = g[x(n_i)]$ is optimal over the interval $t_0 \le t \le t_\nu$, then it is also optimal over any sub-interval $t_m \le t \le t_\nu$, where $0 \le m \le \nu$. As it can be seen from Figure 1, the total cost J_M can be decomposed into F_is for $0 \le i \le \nu$ where

$$\begin{aligned} F_i &= x^T(n_i)Qx(n_i) + x^T(n_i+1)Qx(n_i+1) \\ &+ x^T(n_i+2)Qx(n_i+2) + ... + x^T(n_{i+1}-1)Qx(n_{i+1}-1) \\ &+ (n_{i+1}-n_i)u^T(n_i)Ru(n_i) \end{aligned} \tag{7}$$

That is, from (1),

$$\begin{aligned} F_i &= x^T(n_i)Qx(n_i) + (Ax(n_i) + Bu(n_i))^T Q(Ax(n_i) + Bu(n_i)) \\ &+ (A^2x(n_i) + ABu(n_i) + Bu(n_i))^T Q(A^2x(n_i) + ABu(n_i) + Bu(n_i)) \\ &+ ... + (A^{n_{i+1}-n_i-1}x(n_i) + A^{n_{i+1}-n_i-2}Bu(n_i) + ... + ABu(n_i) + Bu(n_i))^T Q \\ &\quad (A^{n_{i+1}-n_i-1}x(n_i) + A^{n_{i+1}-n_i-2}Bu(n_i) + ... + ABu(n_i) + Bu(n_i)) \\ &+ (n_{i+1}-n_i)u^T(n_i)Ru(n_i) \end{aligned} \tag{8}$$

This can be rewritten as

$$\begin{aligned} F_i &= x^T(n_i)Qx(n_i) + \sum_{j=1}^{n_{i+1}-n_i-1} [A_jx(n_i) + B_ju(n_i)]^T Q[A_jx(n_i) + B_ju(n_i)] \\ &+ (n_{i+1}-n_i)u^T(n_i)Ru(n_i) \end{aligned} \tag{9}$$

where $A_j = A^j$ and $B_j = \sum_{k=0}^{j-1} A^k B$.

Then J_M can be expressed as

$$J_M = F_0 + F_1 + F_2 + ... + F_\nu. \tag{10}$$

Let S_m be the cost from $i = \nu - m + 1$ to $i = \nu$:

$$S_m = F_{\nu-m+1} + F_{\nu-m+2} + ... + F_{\nu-1} + F_\nu, \;\; 1 \le m \le \nu + 1. \tag{11}$$

These cost terms are well illustrated in the above Figure 1.

Therefore, by applying the principle of optimality, we can first minimize $S_1 = F_\nu$, then choose $F_{\nu-1}$ to minimize $S_2 = F_{\nu-1} + F_\nu = S_1^o + F_{\nu-1}$ where S_1^o is the optimal cost occurred at t_ν. We can continue choosing $F_{\nu-2}$ to minimize $S_3 = F_{\nu-2} + F_{\nu-1} + F_\nu = F_{\nu-2} + S_2^o$ and so on until $S_{\nu+1} = J_M$ is minimized. Note that $S_1 = F_\nu = x^T(n_\nu)Qx(n_\nu)$ is determined only from $x(n_\nu)$ which is independent of any other control inputs.

3.2 Inductive Construction of an Optimal Control Law with D_ν Given

We inductively derive an optimal controller which changes its control at ν time instants t_0, t_1, ..., $t_{\nu-1}$. As we showed in the previous section, the inductive procedure goes backwards in time from S_1^o to $S_{\nu+1}^o$. Since $S_1 = F_\nu = x^T(n_\nu)Qx(n_\nu) + u^T(n_\nu)Ru(n_\nu)$ and $x(n_\nu)$ is independent of $u(n_\nu)$, we can let $u^o(n_\nu) = u^o(M) = 0$ and $S_1^o = x^T(n_\nu)Qx(n_\nu)$ where Q is symmetric and positive semi-definite.

Induction Basis: $S_1^o = x^T(n_\nu)Qx(n_\nu)$ where Q is symmetric.

Inductive Assumption: Suppose that
$S_m^o = x^T(n_{\nu-m+1})P(\nu-m+1)x(n_{\nu-m+1})$
holds for some m where $1 \le m \le \nu$ and $P(\nu-m+1)$ is symmetric.

We can write S_m^o as

$$\begin{aligned} S_m^o \;=\; & [A_{(n_{\nu-m+1}-n_{\nu-m})}x(n_{\nu-m}) + B_{(n_{\nu-m+1}-n_{\nu-m})}u(n_{\nu-m})]^T P(\nu-m+1) \\ & [A_{(n_{\nu-m+1}-n_{\nu-m})}x(n_{\nu-m}) + B_{(n_{\nu-m+1}-n_{\nu-m})}u(n_{\nu-m})] \end{aligned} \tag{12}$$

From the definition of S_m and (9),

$$\begin{aligned} S_{m+1} &= S_m^o + F_{\nu-m} \\ &= S_m^o + x^T(n_{\nu-m})Qx(n_{\nu-m}) \\ &+ \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} [A_j x(n_{\nu-m}) + B_j u(n_{\nu-m})]^T Q [A_j x(n_{\nu-m}) + B_j u(n_{\nu-m})] \\ &+ (n_{\nu-m+1} - n_{\nu-m})u^T(n_{\nu-m})Ru(n_{\nu-m}) \end{aligned} \tag{13}$$

And the above equation becomes

$$\begin{aligned} S_{m+1} &= [A_{n_{\nu-m+1}-n_{\nu-m}}x(n_{\nu-m}) + B_{n_{\nu-m+1}-n_{\nu-m}}u(n_{\nu-m})]^T P(\nu-m+1) \\ & \quad [A_{n_{\nu-m+1}-n_{\nu-m}}x(n_{\nu-m}) + B_{n_{\nu-m+1}-n_{\nu-m}}u(n_{\nu-m})] \\ &+ x^T(n_{\nu-m})Qx(n_{\nu-m}) \\ &+ \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} [A_j x(n_{\nu-m}) + B_j u(n_{\nu-m})]^T Q [A_j x(n_{\nu-m}) + B_j u(n_{\nu-m})] \\ &+ (n_{\nu-m+1} - n_{\nu-m})u^T(n_{\nu-m})Ru(n_{\nu-m}) \end{aligned} \tag{14}$$

If we differentiate S_{m+1} with respect to $u(n_{\nu-m})$, then

$$\begin{aligned}
\frac{\partial S_{m+1}}{\partial u(n_{\nu-m})} &= B^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu-m+1) A_{n_{\nu-m+1}-n_{\nu-m}} x(n_{\nu-m}) \qquad (15)\\
&+ (A^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu-m+1) B_{n_{\nu-m+1}-n_{\nu-m}})^T x(n_{\nu-m})\\
&+ 2B^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu-m+1) B_{n_{\nu-m+1}-n_{\nu-m}} u(n_{\nu-m})\\
&+ \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} [2B_j^T Q A_j x(n_{\nu-m}) + 2B_j^T Q B_j u(n_{\nu-m})]\\
&+ 2(n_{\nu-m+1} - n_{\nu-m}) R u(n_{\nu-m})\\
= \quad & 2\{B^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu-m+1) A_{n_{\nu-m+1}-n_{\nu-m}} \qquad (16)\\
& + \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} B_j^T Q A_j\} x(n_{\nu-m})\\
&+ 2\{B^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu-m+1) B_{n_{\nu-m+1}-n_{\nu-m}}\\
& + \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} B_j^T Q B_j + (n_{\nu-m+1} - n_{\nu-m}) R\} u(n_{\nu-m})
\end{aligned}$$

Note that $P(\nu - m + 1)$ is symmetric and the following three rules are applied to differentiate S_{m+1} above.

$$\begin{aligned}
\frac{\partial}{\partial x}(x^T Q x) &= 2Qx\\
\frac{\partial}{\partial x}(x^T Q y) &= Qy\\
\frac{\partial}{\partial y}(x^T Q y) &= Q^T x
\end{aligned}$$

Let $\frac{\partial S_{m+1}}{\partial u(n_{\nu-m})} = 0$, from Lemma 1 and Lemma 2 given later we can obtain $u^o(n_{\nu-m})$ which minimizes S_{m+1} and thus obtain S^o_{m+1}.

$$\begin{aligned}
u^o(n_{\nu-m}) &= -\{B^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu-m+1) B_{n_{\nu-m+1}-n_{\nu-m}} \qquad (17)\\
&\quad + \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} B_j^T Q B_j + (n_{\nu-m+1} - n_{\nu-m}) R\}^{-1}\\
&\quad \{B^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu-m+1) A_{n_{\nu-m+1}-n_{\nu-m}} + \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} B_j^T Q A_j\} x(n_{\nu-m})\\
&= -K(\nu-m) x(n_{\nu-m})
\end{aligned}$$

where $K(\nu - m)$ is defined in (17).

Therefore, we can write

$$A_{n_{\nu-m+1}-n_{\nu-m}}x(n_{\nu-m}) + B_{n_{\nu-m+1}-n_{\nu-m}}u^o(n_{\nu-m}) = [A_{n_{\nu-m+1}-n_{\nu-m}} - B_{n_{\nu-m+1}-n_{\nu-m}}K(\nu-m)]x(n_{\nu-m}) \tag{18}$$

If we use (17) and (18), we have

$$\begin{aligned} S^o_{m+1} &= \{[A_{n_{\nu-m+1}-n_{\nu-m}} - B_{n_{\nu-m+1}-n_{\nu-m}}K(\nu-m)]x(n_{\nu-m})\}^T P(\nu-m+1) \\ &\quad \{[A_{n_{\nu-m+1}-n_{\nu-m}} - B_{n_{\nu-m+1}-n_{\nu-m}}K(\nu-m)]x(n_{\nu-m})\} \\ &+ x^T(n_{\nu-m})Qx(n_{\nu-m}) \\ &+ \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} \{[A_j - B_jK(\nu-m)]x(n_{\nu-m})\}^T Q\{[A_j - B_jK(\nu-m)]x(n_{\nu-m})\} \\ &+ (n_{\nu-m+1}-n_{\nu-m})[K(\nu-m)x(n_{\nu-m})]^T R[K(\nu-m)x(n_{\nu-m})] \end{aligned} \tag{19}$$

This equation can be rewritten as

$$\begin{aligned} S^o_{m+1} &= x^T(n_{\nu-m})\{[A_{n_{\nu-m+1}-n_{\nu-m}} - B_{n_{\nu-m+1}-n_{\nu-m}}K(\nu-m)]^T P(\nu-m+1) \\ &\quad [A_{n_{\nu-m+1}-n_{\nu-m}} - B_{n_{\nu-m+1}-n_{\nu-m}}K(\nu-m)] \\ &+ Q \\ &+ \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} [A_j - B_jK(\nu-m)]^T Q[A_j - B_jK(\nu-m)] \\ &+ (n_{\nu-m+1}-n_{\nu-m})K^T(n_{\nu-m})RK(\nu-m)\}x(n_{\nu-m}). \\ &= x^T(n_{\nu-m})P(\nu-m)x(n_{\nu-m}) \end{aligned} \tag{20}$$

where $P(\nu-m)$ is obtained from $K(\nu-m)$ and $P(\nu-m+1)$ as in (20). Also note that knowing $P(\nu-m+1)$ is enough to compute $K(\nu-m)$ because other terms of (17) are known a priori.

Therefore, we find a symmetric matrix $P(\nu-m)$ satisfying $S^o_{m+1} = x^T(n_{\nu-m})P(\nu-m)x(n_{\nu-m})$. From (17) and (20), we have the following recursive equations for obtaining $P(\nu-m)$ from $P(\nu-m+1)$ where $m = 1, 2, ..., \nu$.

$$\begin{aligned} K(\nu-m) &= \{B^T_{n_{\nu-m+1}-n_{\nu-m}}P(\nu-m+1)B_{n_{\nu-m+1}-n_{\nu-m}} \\ &+ \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} B_j^T Q B_j + (n_{\nu-m+1}-n_{\nu-m})R\}^{-1} \\ &\quad \{B^T_{n_{\nu-m+1}-n_{\nu-m}}P(\nu-m+1)A_{n_{\nu-m+1}-n_{\nu-m}} + \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} B_j^T Q A_j\} \end{aligned} \tag{21}$$

$$
\begin{aligned}
P(\nu-m) &= [A_{n_{\nu-m+1}-n_{\nu-m}} - B_{n_{\nu-m+1}-n_{\nu-m}} K(\nu-m)]^T P(\nu-m+1) \\
&\quad [A_{n_{\nu-m+1}-n_{\nu-m}} - B_{n_{\nu-m+1}-n_{\nu-m}} K(\nu-m)] \\
&+ Q \\
&+ \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} [A_j - B_j K(\nu-m)]^T Q [A_j - B_j K(\nu-m)] \\
&+ (n_{\nu-m+1} - n_{\nu-m}) K^T(\nu-m) R K(\nu-m)
\end{aligned} \tag{22}
$$

Also, we know that at each time instant $n_{\nu-m}\Delta$

$$
u^o(n_{\nu-m}) = -K(\nu-m)x(n_{\nu-m}) \tag{23}
$$

Hence, with $P(\nu) = Q$, we can obtain $K(i)$ and $P(i)$ for $i = \nu-1, \nu-2, ..., 0$ recursively using (21) and (22). At each time instant $n_i\Delta$, $i = 0, 1, 2, ..., \nu-1$ the new control input value will be obtained using (23) by multiplying $K(i)$ by $x(n_i)$ where $x(n_i)$ is the estimate of the system state at $n_i\Delta$. Also, note that the optimal control cost is $J_M^o = S_{\nu+1}^o = x^T(0)P(0)x(0)$ where $P(0)$ is found from the above procedure.

To prove the optimality of this control law we need the following lemmas.

Lemma 1 *If Q is positive semi-definite and R is positive definite, then $P(i)$, $i = \nu, \nu-1, \nu-2, ..., 0$, matrices are positive semi-definite. Hence, $P(i)$s are symmetric from the definition of a positive semi-definite matrix.*

Proof Since $P(\nu) = Q$, from assumption $P(\nu)$ is positive semi-definite. Assume that for $k = i+1$, $P(k)$ is positive semi-definite. We use induction to prove that $P(i)$ is semi-definite. Note that Q is positive semi-definite and R is positive definite. From (22) we have

$$
\begin{aligned}
P(i) &= [A_{n_{i+1}-n_i} - B_{n_{i+1}-n_i} K(i)]^T P(i+1) \\
&\quad [A_{n_{i+1}-n_i} - B_{n_{i+1}-n_i} K(i)] \\
&+ Q \\
&+ \sum_{j=1}^{n_{i+1}-n_i-1} [A_j - B_j K(i)]^T Q [A_j - B_j K(i)] \\
&+ (n_{i+1} - n_i) K^T(i) R K(i)
\end{aligned} \tag{24}
$$

Since $P(i+1)$ and Q are positive semi-definite, R is positive definite, and $(n_{i+1} - n_i) > 0$, it is easy to verify that for $\forall y \in R^m \ : \ y^T P(i) y \geq 0$. This means that $P(i)$ is positive semi-definite. This inductive procedure proves the lemma.

Lemma 2 *Given D_ν, the inverse matrix in (21) always exists.*

Proof Let $V = B^T_{n_{\nu-m+1}-n_{\nu-m}} P(\nu - m + 1) B_{n_{\nu-m+1}-n_{\nu-m}} + \sum_{j=1}^{n_{\nu-m+1}-n_{\nu-m}-1} B_j^T Q B_j + (n_{\nu-m+1} - n_{\nu-m})R$. From Lemma 1, $P(\nu - m + 1)$ is positive semi-definite. Therefore, $\forall y \in R^m \ :$ $y^T V y > 0$ because Q is positive semi-definite, R is positive definite and $n_{\nu-m+1} - n_{\nu-m} > 0$. This implies that V is positive definite. Hence the inverse matrix exists.

Theorem 1 *Given D_ν, $K(i)$ $(i = 0, 1, 2, ..., \nu-1)$ obtained from the above procedure are the optimal feedback gains which minimize the cost function J_M (and $J_M^{'}$) on $[0, M\Delta]$.*

Proof Note that given D_ν, J_M is a convex function of $u(n_i), i = 0, 1, ..., \nu - 1$. Thus the above feedback control law is optimal.

Lemma 3 *If $p \leq q$ and $D_p \subseteq D_q$, then $J^o_{M_p} \geq J^o_{M_q}$ where $J^o_{M_p}$ and $J^o_{M_q}$ are the optimal costs of controls which change controls at time instants in D_p and D_q respectively.*

Proof Suppose that $J^o_{M_p} < J^o_{M_q}$, then, in controlling the system with D_q, if we do not change controls at time instants in $D_q - D_p$ and change controls at time instants in D_p to the same control inputs that were exercised to get $J^o_{M_p}$ with D_p, we obtain $\hat{J}_{M_q}$ which is equal to $J^o_{M_p}$. This contradicts the fact that $J^o_{M_q}$ is the minimum cost obtainable with D_q since we have found $\hat{J}_{M_q}$ which is equal to $J^o_{M_p}$ and therefore less than $J^o_{M_q}$. Hence, $J^o_{M_p} \geq J^o_{M_q}$.

This lemma implies that if we do not take computation cost, μ, into consideration, then the more control exercising points, the better the controller is (less cost). With the computation cost being included in the cost function, the statement above is no longer true. Therefore we need to search for an optimal D_ν which minimizes the cost function $J_M^{'}$. The following sections provide a detailed discussion on searching for such an optimal solution. Note that if we let $D_\nu = D_M$ then the optimal temporal control law is the same as the traditional linear feedback optimal control law.

3.3 Optimal Temporal Control Law over D_ν Space with ν Given

When the number of control changing points, ν, and an initial system state $x(0)$ are given, we search over a set of possible D_νs and $u(D_\nu)$s such that the cost function J_M is minimized. This can be done by varying $\nu - 1$ control changing time instants, $t_i,\ i = 1, 2, ..., \nu - 1$ (since $t_0 = 0$) over the discrete set, $D_M = \{0, \Delta, 2\Delta, \ldots, (M-1)\Delta\}$ and applying the technique developed in the previous section for each given D_ν. Let us denote such a D_ν which minimizes J_M as D_ν^o. Note that when ν is given, minimizing J_M is equivalent to minimizing $J_M^{'}$. Since both D_ν and $u(D_\nu)$ are control variates, to be able to find a global optimal solution, either an exhaustive search or some global search methods like *Genetic Algorithm* or *Simulated Annealing* should be considered. Later we present a numerical example in which an exhaustive search with *Steepest Descent Search* method is used. Searching for a globally optimal solution for a temporal controller calls for further research.

3.4 Optimal Temporal Control Law

Assume that a maximum number of control changing points, ν_{max}, is given. By varying ν from 1 to ν_{max} we can find $D_{\nu*}^o$ to obtain a globally optimal temporal controller which minimizes $J_M^{'}$. This can be done by first searching for D_ν^o for each given ν and then comparing the cost function $J_M^{'} = J_M + \nu\mu$ at each $D_\nu^o,\ \nu = 1, 2, \ldots, \nu_{max}$. That is, let $J_{M_\nu}^{'o} = x^T(0)P(0)x(0) + \nu\mu$ where $P(0)$ is calculated at D_ν^o as in the previous section. Then we can obtain a global minimum cost $J_M^{'o} = \min_{1 \leq \nu \leq \nu_{max}} \{J_{M_\nu}^{'o}\}$ and an optimal number of control changes, ν^o, at which $J_{M_{\nu^o}}^{'o} = J_M^{'o}$.

3.5 Terminal State Constraints

The terminal state constraints may be used to check if the optimal temporal controller with $D_{\nu*}^o$ can drive the system state to a permissible final state within a given time. Let X_f be a set of allowed terminal states, if $x(n_\nu) \in X_f$, then the control law is said to be *stable* in terms of the terminal state constraints and *not stable* if $x(n_\nu) \notin X_f$. If the globally optimal temporal controller obtained from the above procedure is not stable, ν^* should be increased until a stable one is found. One way of specifying terminal state constraints for regulators might be $| x(M)_i | \leq \epsilon_i$ where $x(M)_i$ is the ith element of $x(M)$ state vector.

3.6 Algorithm to Derive an Optimal Temporal Controller

To summarize the above discussion, we provide in Figure 2 a complete algorithm to search for a globally optimal temporal controller under the assumption that the initial state $x(0)$ is given.

In the algorithm, a neighbor of $D_\nu = \{n_0\Delta, n_1\Delta, n_2\Delta, \ldots, n_{\nu-1}\Delta\}$ is defined to be any member of a set $N(D_\nu) = \{\{n_0^{'}\Delta, n_1^{'}\Delta, \ldots, n_{\nu-1}^{'}\Delta\} \mid \ \mid n_i^{'} - n_i \mid \leq 1, \ i = 1, 2, \ldots, \nu - 1\}$.

3.7 Optimal Temporal Controllers over an Initial State Space

Note that D_ν^o might become different if a new initial system state $\hat{x}(0)$ is used instead of $x(0)$ when the state vector is in $R^{m\times 1}$ where $m \geq 2$. This is because the cost function $J_M = x^T(0)P(0)x(0)$ depends on $x(0)$ as well as $P(0)$. Thus, D_ν^o is dependent on the initial state $x(0)$. However, when $m = 1$ it can be shown that D_ν^o is independent of any initial state. To see this let $x(0) = k\hat{x}(0) \in \mathcal{R}^1$ and $P(0)$ and $\hat{P}(0)$ be the optimal matrices with initial states $x(0)$ and $\hat{x}(0)$, respectively. i.e.,

$$J_M(x(0)) = x(0)P(0)x(0)$$
$$J_M(\hat{x}(0)) = \hat{x}(0)\hat{P}(0)\hat{x}(0)$$

From the optimality of $\hat{P}(0)$ with respect to $\hat{x}(0)$,

$$\hat{x}^T(0)P(0)\hat{x}(0) \ \ \geq \ \ \hat{x}^T(0)\hat{P}(0)\hat{x}(0) \tag{25}$$

Multiplying the above inequality by k^2 we have

$$\begin{aligned} k^2\hat{x}^T(0)P(0)\hat{x}(0) &= x^T(0)P(0)x(0) \\ &\geq k^2\hat{x}^T(0)\hat{P}(0)\hat{x}(0) \\ &= x^T(0)\hat{P}(0)x(0) \end{aligned} \tag{26}$$

On the other hand, due to the optimality of $P(0)$ we have

$$x^T(0)\hat{P}(0)x(0) \ \ \geq \ \ x^T(0)P(0)x(0) \tag{27}$$

Therefore, $\hat{P}(0) = P(0)$. This implies the optimality of $\hat{P}(0)$ and $\hat{D}_\nu^o$ for any initial state $x(0) \in \mathcal{R}^1$.

Generally speaking, the above result will not hold for $m \geq 2$ cases. However, using the same argument discussed above we can prove that for any initial state $x(0) = k\hat{x}(0)$, $x(0)$ and $\hat{x}(0)$ will have the same D_ν^o as well as the same $P(0)$.

```
ν^o = 1
J'^o_M = ∞
for ν = 1 to ν_max {
    /* Several different search starting points */
    for i = 1 to NumInitPts_ν {
        D_ν = D_ν^{init,i}
        /* Iterate until a local minimum is found – Steepest Descent Search */
        while (MinimumFound != True) {
            Find optimal costs for neighboring points of D_ν using theorem 1
            if ( J'_M has a Local Minimum at D_ν)
                then {
                        MinimumFound = True
                        J'^i_{M_ν} = Cost(J'_M) at D_ν }
                else
                        D_ν = a neighbor of D_ν with the smallest J'_M
            }
        }
    J'^o_{M_ν} = min_{1≤i≤NumInitPts_ν} {J'^i_{M_ν}}
    if ( J'^o_{M_ν} < J'^o_M )
        then {
            ν^o = ν
            J'^o_M = J'^o_{M_ν} }
    }
```

Figure 2: Complete algorithm to find an optimal temporal controller.

4 Implementation

To implement temporal control, we need to calculate and store $K(i)$ matrices in (22) and use them when controlling the system utilizing (23). Note that in traditional optimal linear control a similar matrix is obtained and used at every time instant in D_M to generate control input value. While the feedback gain matrices for traditional linear optimal controller are independent of initial states, the number of control exercises, ν, and $K(i)$ matrices are dependent on initial states for temporal control systems. But, if the possible set of initial states is in $\mathcal{R}^1$ they are independent of the initial states. Effective deployment of temporal control requires that we know the range of initial state values and generate $K(i)$ matrices for each group. A sensitivity analysis is required to determine how many distinct matrices need to be stored.

In order to implement temporal control we require an operating system that supports scheduling control computations at specific time instants. The Maruti system developed at the University of Maryland is a suitable host for the implementation of temporal control [10, 8, 7]. In Maruti, all executions are scheduled in time and the time of execution can be modified dynamically, if so desired. This is in contrast with traditional cyclic executives often used in real-time systems, which have a fixed, cyclic operation and which are well suited only for the sampled data control systems operating in a static environment. It is the availability of the system such as Maruti that allows us to consider the notion of temporal control, in which time becomes an emergent property of the system.

5 Example

To illustrate the advantages of a temporal control scheme let us consider a simple example of rigid body satellite control problem [12]. The system state equations are as follows:

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0.00125 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 1 & 1 \end{bmatrix} x(k) \end{aligned}$$

where k represents the time index and one unit of time is the discretized subinterval of length $\Delta = 0.05$. The linear quadratic performance index J'_M in (5) is used here with the following parameters.

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

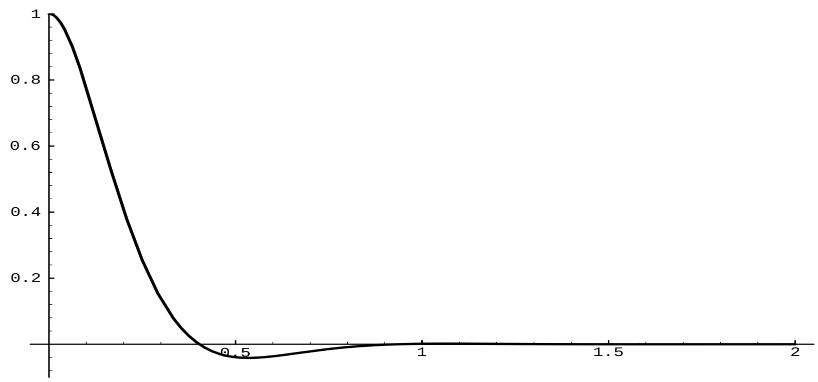


Figure 3: Optimal Linear Control with $\Delta = 0.05$.

$$
\begin{aligned}
R &= 0.0001 \\
\mu &= 0.02 \ \& \ 0.01 \\
M &= 40 \\
\Delta &= 0.05 \\
\epsilon_i &= 0.01, \quad i = 1, 2 \\
x(0) &= \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}
\end{aligned} \tag{28}
$$

The objective of the control is to drive the satellite to the zero position and the desired goal state is $x_f = [0, \quad 0]^T$. The terminal state constraint is $\mid x_i(40) \mid \leq \epsilon_i \quad i = 1, 2$. With the equal sampling interval $\Delta = 0.05$ and $M = 40$ the optimal linear feedback control of this system has cost function $J_M = 0.984678$ (without computational cost) and $J_M^{'} = 1.784678$ (with computational cost) and is shown in Figure 3. The terminal state constraint is satisfied at $0.8sec$.

If we apply the temporal control scheme presented above to this problem with $\mu = 0.02$ we find that the optimal number of control changes for this example is 3 and $D_3^o = \{0, 2\Delta, 10\Delta\}$ with a cost $J_M^{'} = 1.08388$. Note that the 40 step optimal linear feedback controller given above has a cost $J_M^{'} = 1.784678$ when computation cost is considered. Table 1 shows how this optimal controller is obtained when we set $\nu_{max} = 7$. Figure 4(a) shows the system trajectory when this three-step optimal temporal controller is used to control the system. This trajectory satisfies the terminal state constraint at $0.8sec$ as well. Also, the maximum control input magnitudes, $\mid u \mid_{max}$, in both

n	D_ν^o	Cost(J_M') with $\mu = 0.02$	Cost(J_M') with $\mu = 0.01$
1	$\{0\}$	$4.63089 + \mu = 4.65089$	$4.63089 + \mu = 4.64089$
2	$\{0, 1\}$	$1.44603 + 2\mu = 1.48603$	$1.44603 + 2\mu = 1.46603$
3	$\{0, 2, 10\}$	$1.02388 + 3\mu = 1.08388$	$1.02388 + 3\mu = 1.05388$
4	$\{0, 2, 9, 11\}$	$1.02224 + 4\mu = 1.10224$	$1.02224 + 4\mu = 1.06224$
5	$\{0, 1, 3, 8, 11\}$	$0.996968 + 5\mu = 1.096968$	$0.996968 + 5\mu = 1.046968$
6	$\{0, 1, 3, 8, 11, 24\}$	$0.996746 + 6\mu = 1.116746$	$0.996746 + 6\mu = 1.056746$
7	$\{0, 1, 3, 8, 11, 23, 25\}$	$0.996745 + 7\mu = 1.136745$	$0.996745 + 7\mu = 1.066745$

Table 1: Calculating optimal temporal controllers.

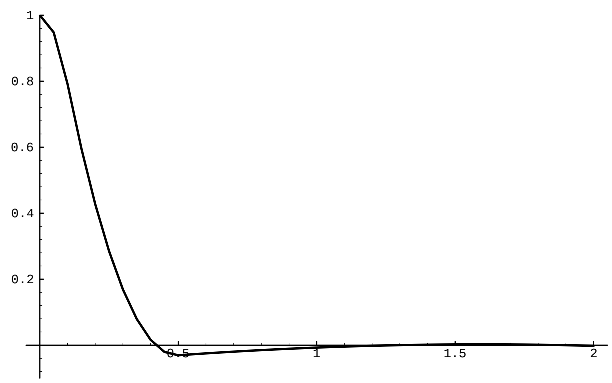
controllers lie within the same bound $B = 50$, which may be another constraint on control.

The optimal temporal controller found with $\mu = 0.01$ has $\nu = 5$ and $D_5^o = \{0, \Delta, 3\Delta, 8\Delta, 11\Delta\}$ with a cost $J_M = 0.996968$. Note that this cost is even less than 1.01269 which is obtained from the optimal controller with equal sampling period $0.1sec$ and 20 control changes.

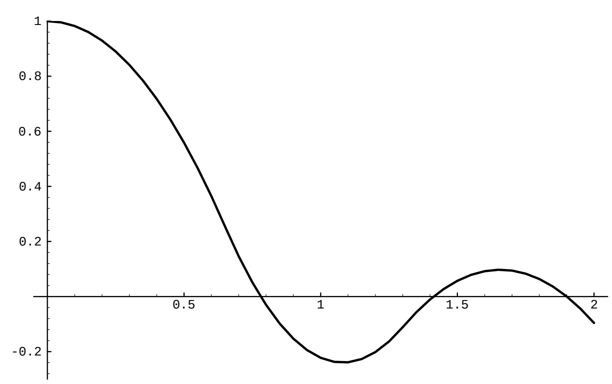
If we change control values only at three time instants with equal sampling period, $13M = 0.65sec$, the total cost incurred is 2.2823(without computational cost) on the time interval $[0, 2]$. The cost is more than twice that of our optimal temporal controller and the terminal state constraint is not satisfied even at the end of the controlling interval of $2.0sec$. Figure 4(b) clearly shows the advantages of using an optimal temporal controller over using an optimal controller of equidistant samplings. Their performances are noticeably different though both of them are changing controls at three time instants. It is clear that the optimal temporal control with three control changes performs almost the same as 40 step linear optimal controller does. This implies that enforcing the constant sampling rate throughout the entire controlling interval may simply waste computational power which otherwise could be used for other concurrent controlling tasks in critical systems.

Obtaining D_3^o for this example was simple since J_{40} has only one minimum over the entire set of possible D_3s on $[0, 40\Delta]$. Figure 5(a) and Figure 5(b) show that J_{40} has only one local(global) minimum at $D_3^o = \{0, 2\Delta, 10\Delta\}$. We got this optimal D_3 by doing steepest descent search with the starting point $D_3^{init} = \{0, \Delta, 10\Delta\}$ after searching for only three points, $\{0, \Delta, 10\Delta\}$, $\{0, 2\Delta, 10\Delta\}$, $\{0, 3\Delta, 10\Delta\}$. Also, Figure 5(a) shows that choosing n_1 has greater influence on the total cost than n_2 since the cost varies more radically along the n_1 axis in the figure. This means that the initial stage of the control needs more attention than the later stage in this linear control problem.

But, if we change one of the parameters of performance index function, R, from 0.0001 to 0.001 we get two local minima at $D_3^1 = \{0, \Delta, 2\Delta\}$ and $D_3^2 = \{0, 3\Delta, 19\Delta\}$, among which D_3^2 is the

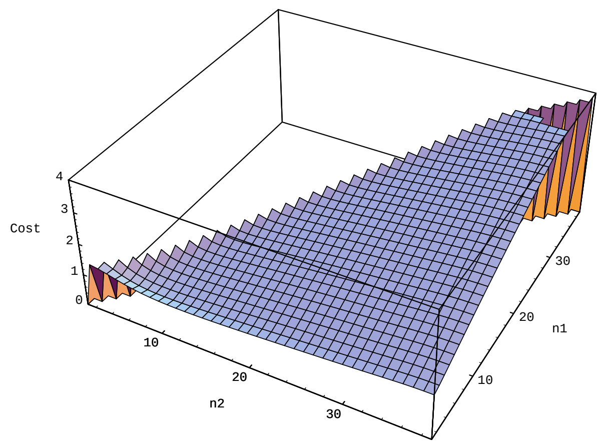


(a)

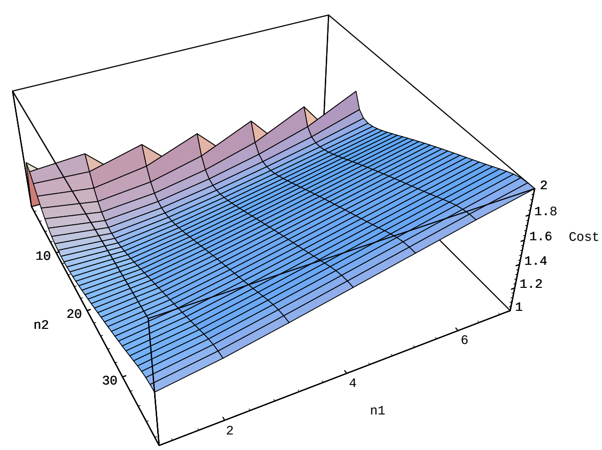


(b)

Figure 4: Control trajectories with 3 control changes. (a)Optimal temporal control with $D_3^o = \{0, 2\Delta, 10\Delta\}$. (b)Optimal linear control with 13Δ ($0.65sec$) period.



(a)



(b)

Figure 5: Cost function distribution over (n_1, n_2). (a)Costs on D_3 space. (b)Costs near $D_3^o = \{0, 2\Delta, 10\Delta\}$.

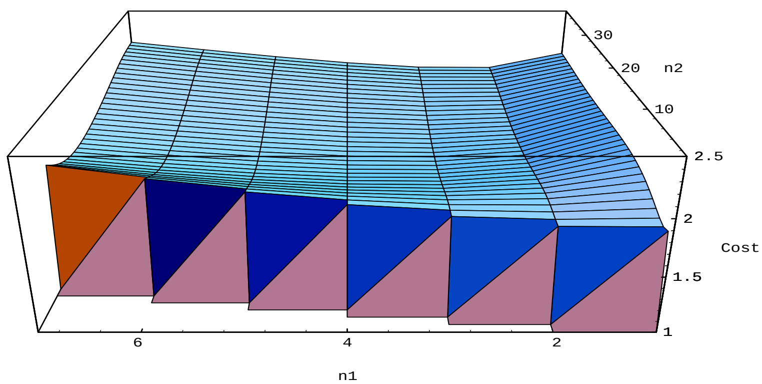


Figure 6: Costs near D_3^1 and D_3^2 with $R = 0.001$.

optimal one with less cost. Figure 6 shows this fact. In this case we need to use steepest descent search method at least twice with different search starting points to get an optimal solution. We implemented this steepest descent search algorithm in Mathematica and used it to generate D_ν^o for several examples by varying ν. For our examples of linear time invariant system control problems the number of local minima was not so large that we could efficiently apply this search method just a few times with different initial D_ν^{init}s to get a global minimum without doing an exhaustive search over the entire D_ν space.

6 Discussion

Employing the temporal control methodology in concurrent real-time embedded systems will have a significant impact on the way computational resources are utilized by control tasks. A minimal amount of control computations can be obtained for a given regulator by which we can achieve almost the same control performance compared to that of traditional controller with equal sampling period. This significantly reduces the CPU times for each controlling task and thus increases the number of real-time control functions which can be accommodated concurrently in one embedded system. Particularly, in a hierarchical control system if temporal controllers can be employed for lower level controllers the higher level controllers will have a great degree of flexibility in managing resource usages by adjusting computational requirements of each lower level controller. For example, in emergency situations the higher level controller may force the lower level controller to run as infrequently as they possibly can (thus freeing computational resources for handling the emergency). In contrast, during normal operations the temporal control tasks may run as necessary, and the additional computation time can be used for higher level functions such as monitoring and planning, etc.

In addition, the method developed in Section 3.2, which calculates an optimal controller when control changing time instants are given, can be applied to the case in which the control computing time instants cannot be periodic. For example, when a small embedded controller is used to control several functions, it may be a lot better to design a temporal controller for each function such that the required computational resources are appropriately scheduled while retaining the required degree of control for each function.

7 Conclusion

In this paper we proposed a *temporal control* technique based on a new cost function which takes into account computational cost as well as state and input cost. In this scheme new control input values are defined at time instants which are not necessarily regularly spaced. For the linear control problem we showed that almost the same quality of control can be achieved while much less computations are used than in a traditional controller.

The proposed formulation of temporal control is likely to have a significant impact on the way concurrent embedded real-time systems are designed. In hierarchical control environment, this approach is likely to result in designs which are significantly more efficient and flexible than traditional control schemes. As it uses less computational resources, the lower level temporal controllers will make the resources available to the higher level controllers without compromising the quality of control.

References

[1] A. Belleisle. Stability of systems with nonlinear feedback through randomly time-varying delays. *IEEE Transactions on Automatic Control*, AC-20:67–75, February 1975.

[2] R. Bellman. *Adaptive Control Process: A Guided Tour.* Princeton,NJ: Princeton University Press, 1961.

[3] R. Bellman. Bellman special issue. *IEEE Transactions on Automatic Control*, AC-26, October 1981.

[4] P. Dorato and A. Levis. Optimal linear regulators: The discrete time case. *IEEE Transactions on Automatic Control*, AC-16:613–620, December 1971.

[5] A. Gosiewski and A. Olbrot. The effect of feedback delays on the performance of multivariable linear control systems. *IEEE Transactions on Automatic Control*, AC-25(4):729–734, August 1980.

[6] K. Hirai and Y. Satoh. Stability of a system with variable time delay. *IEEE Transactions on Automatic Control*, AC-25(3):552–554, June 1980.

[7] S. T. Levi, Satish K. Tripathi, Scott Carson, and Ashok K. Agrawala. The MARUTI hard real-time operating system. *ACM Symp. on Op. Syst. Principles, Op. Syst. Review*, 23(3), July 1989.

[8] Shem-Tov Levi and Ashok K. Agrawala. *Real Time System Design.* McGraw Hill, 1990.

[9] Z. Rekasius. Stability of digital control with computer interruptions. *IEEE Transactions on Automatic Control*, AC-31:356–359, April 1986.

[10] Manas Saksena, James da Silva, and Ashok K. Agrawala. *Design and Implementation of Maruti-II*, chapter 4. Prentice Hall, 1995. In *Advances in Real-Time Systems*, edited by Sang H. Son.

[11] K. G. Shin and H. Kim. Derivation and application of hard deadlines for real-time control systems. *IEEE Transactions on Systems, Man and Cybernetics*, 22(6):1403–1413, November 1992.

[12] G.S. Virk. *Digital Computer Control Systems*, chapter 4. McGraw Hill, 1991.

[13] K. Zahr and C. Slivinsky. Delay in multivariable computer controlled linear systems. *IEEE Transactions on Automatic Control*, pages 442–443, August 1974.