# REAL-TIME OPTIMIZATION FOR NONLINEAR SYSTEMS USING ALGORITHMIC CONTROL

**Tomoaki Kobayashi** * **Michiyuki Magono** **
**Joe Imae** * **Kazutaka Yoshimizu** * **Guisheng Zhai** *

* *Dept. Mechanical Engineering, Osaka Prefecture Univ.*
*1-1 Gakuen-cho, Sakai, Osaka 599-8531, JAPAN*
*Email: kobayasi@mecha.osakafu-u.ac.jp*
** *Graduate School of Information Science, Nara Institute of Science and Technology*
*8916-5 Takayama-cho, Ikoma, Nara 630-0192, JAPAN*

Abstract: A new method for on-line optimal control of nonlinear systems is proposed. The proposed method is based on the algorithms for obtaining the numerical solutions of optimal control problems. According to this idea, we can use the iterative computational method for the on-line optimal control problems. In this paper, we adopt the Riccati-equation based algorithm, one of the iterative algorithms, and demonstrate the proposed algorithmic method is applicable to such mechanical systems as fast-sampling is required.*Copyright ©2005 IFAC*

Keywords: Real time optimization, nonlinear system, algorithmic design.

## 1. INTRODUCTION

The design methods for real-time calculation of unconstrained/constrained nonlinear control problems have gained much attention in a variety of realworld applications. Among them, receding horizon schemes (Allgowet *et al.*, 2000), (Ohtsuka, 2004) and state-dependent Riccati-equation techniques (Coutier *et al.*, 1996) are getting their popularity.

In the computational field of control problems, iterative approaches are popular in finding numerical solutions. It is known, however, that such approaches are computationally expensive, and that they are not suitable for real-time control. In this paper, we focus on iterative computational techniques for numerical solutions, and attack the big issue of real-time computation of optimal control problems. In this paper, a new idea is proposed in order to circumvent such computational burden. According to this idea, we can use the iterative computational method for the real-time control of nonlinear optimal control problems.

The outline of the paper is as follows. In Section 2, the nonlinear optimal control problems are formulated. Also, one of the computational methods for optimal control problems is given with its convergence property. In Section 3, based on the computational method, our algorithmic design method is proposed. In Sections 4 and 5, simulation and experiment results are given in order to demonstrate the effectiveness and practicability of our approach, where the Van der Pol problem and the swing control problem of a crane are adopted as design examples.

## 2. OPTIMAL CONTROL PROBLEM

### *2.1 Formulation*

System equation, initial condition, and performance index are given as follows.

$$\dot{x}(t) = f(t, x(t), u(t)) \quad (1)$$

$$x(t_0) = x_0 \in \Re^n \quad (2)$$

$$J = G(x(t_1)) + \int_{t_0}^{t_1} L(t, x(t), u(t))dt \quad (3)$$

where $t_0$, $t_1$ are initial/terminal time given. Then, our goal is to find a controller minimizing the performance index $J$ over a time interval $[t_0, t_1]$. Here, denote the state variable by $x(t) = [x_1(t), \cdots, x_n(t)]^T \in \Re^n$, and the input variable by $u(t) = [u_1(t), \cdots, u_r(t)]^T \in R^r$. Based on the problem formulation (1) to (3), we describe our on-line computational design method, that is to say, algorithmic design method.

Before that, we give some preliminaries. Whether or not the algorithmic design method succeeds depends on how effective the algorithm is to iteratively search the numerical solutions of optimal control problems. In this paper, we adopt one of the so-called Riccati-equation based algorithms (REB algorithms), which is known to be reliable and effective in searching numerical solutions. One of the characteristics is the use of feedback structure in the process of computation of solutions. Details are given later.

*2.2 Riccati-equation based algorithm*

Under the problem formulation (1) to (3), we describe an iterative algorithm for the numerical solutions of optimal control problems, based on Riccati differential equations. In this respect, the algorithm falls in the category of optimal control algorithms, such as the REB algorithms presented in (Nedeljkovic, 1981), (Murray, 1978), (Merriam, 1965), (Bullock & Franklin, 1967), (Dyer & McReynolds, 1970), (Jacobson & Mayne, 1970) and (Imae *et al.*, 1992).

**Assumptions**

Let $x : [t_0, t_1] \to \Re^n$ be an absolutely continuous function, and $u : [t_0, t_1] \to \Re^r$ be an essentially bounded measurable function. For each positive integer $j$, let us denote by $AC^j$ all absolutely continuous functions: $[t_0, t_1] \to \Re^j$, and by all essentially bounded measurable functions: $[t_0, t_1] \to \Re^j$. Moreover, we define the following norms on $AC^j$ and $L^j_\infty$ respectively:

$$\| x \| = \max \mid x(t) \mid \text{ for } x \in AC^j, t \in [t_0\ t_1]$$

$$\| y \| = \text{ess sup} \mid y(t) \mid \text{ for } y \in L^j_\infty, t \in [t_0\ t_1]$$

where the vertical bars are used to denote Euclidean norms for vectors.

Now, we make some assumptiions.

**(i)** $G : \Re^r \to \Re^1, f : \Re^1 \times \Re^n \times \Re^r \to \Re^n, L : \Re^1 \times \Re^n \times \Re^r \to \Re^1$ are continuous in all their arguments, and their partial derivatives $G_x(x), f_x(t, x, u), f_u(t, x, u), L_x(t, x, u)$ and $L_u(t, x, u)$ exist and are continuous in all their arguments.

**(ii)** For each compact set $U \subset \Re^r$ there exists some $M_1 \in (0, \infty)$ such that

$$\mid f(t, x, u) \mid \leq M_1(\mid x \mid +1) \quad (4)$$

for all $t \in \Re^1, x \in \Re^n$, and $u \in U$.

**Algorithm**

**Step 0:** Let $\beta \in (0, 1)$ and $M_2 \in (0, 1)$. Select arbitrarily an initial input $u^0 \in L^r_\infty$.

**Step 1:** $i = 0$

**Step 2:** Calculate $x^i(t)$ with $u^i(t)$ from the equation (1).

**Step 3:** Select $A^i \in \Re^{n\times n}$, $B^i_{11} \in L^{n\times n}_\infty$, $B^i_{12} \in L^{n\times r}_\infty$ and $B^i_{22} \in L^{r\times r}_\infty$ so that Kalman's sufficient conditions for the boundedness of Riccati solutions [Jacobson (Jacobson & Mayne, 1970), p.36] hold, that is, for almost all $t \in [t_0, t_1]$,

$$A^i \geq 0, B^i_{22}(t) > 0,$$

$$B^i_{11}(t) - B^i_{12}(t){B^i_{22}}^{-1}(t){B^i_{12}}^T(t) \geq 0 \quad (5)$$

where $A^i$, $B^i_{11}$ and $B^i_{22}$ are symmetric and $(\cdot)^T$ means the transpose of vectors and matrices. We solve $\delta x^i(t)$, $K^i(t)$, $r^i(t)$ from (6), (7), and (8) below,

$$\begin{aligned}\delta\dot{x}(t) = &\{f_x(t, x^i, u^i) + f_u(t, x^i, u^i){B^i_{22}}^{-1} \\ &\times (f^T_u(t, x^i, u^i)K(t) - {B^i_{12}}^T)\}\delta x(t) \\ &+ f_u(t, x^i, u^i){B^i_{22}}^T(f^T_u(t, x^i, u^i)r(t) \\ &- L^T_u(t, x^i, u^i)),\ \delta x(t_0) = 0 \end{aligned} \quad (6)$$

$$\begin{aligned}\dot{K}(t) = &-K(t)f_x(t, x^i, u^i) - f^T_x(t, x^i, u^i)K(t) \\ &+ B^i_{11} + (K(t)f_u(t, x^i, u^i) - B^i_{12}){B^i_{22}}^{-1} \\ &\times ({B^i_{12}}^T - f^T_u(t, x^i, u^i)K(t)),\ K(t_1) = -A^i \end{aligned} \quad (7)$$

$$\begin{aligned}\dot{r}(t) = &-f^T_x(t, x^i, u^i)r(t) + L^T_x(t, x^i, u^i) + \{B^i_{12} \\ &- K(t)f_u(t, x^i, u^i)\}{B^i_{22}}^{-1}(-L^T_u(t, x^i, u^i)) \\ &+ f^T_u(t, x^i, u^i)r(t)),\ r(t_1) = -G(x(t_1)) \end{aligned} \quad (8)$$

and determine $\delta u^i$ from the following.

$$\begin{aligned}\delta u^i(t) = &{B^i_{22}}^{-1}\{((f^T_u(t, x^i, u^i)K^i(t) - {B^i_{22}}^T))\delta x^i \\ &+ f^T_u(t, x^i, u^i)r^i(t) - L^T_u(t, x^i, u^i)\} \end{aligned} \quad (9)$$

**Step 4:** Determine $(\tilde{x}^i, \tilde{u}^i)$ satisfying

$\dot{x}(t) = f(t, x(t), u(t)), x(t_0) = x_0 \in \Re^n$

$H^i(t, (x - x^i), (u - u^i), p^i)$

where $= \max_{v\in\Re^r} H^i(t, (x - x^i), (v - v^i), p^i)$

$$\begin{aligned}H^i(t, \delta x, \delta u, p) &= -\{L_x(t, x^i, u^i)\delta x + L_u(t, x^i, u^i)\delta u \\ &+ \tfrac{1}{2}(\delta x^T B^i_{11}\delta x + 2\delta x^T B^i_{12}\delta u + \delta u^T B^i_{22}\delta u)\} \\ &+ p^T(f_x(t, x^i, u^i)\delta x + f_u(t, x^i, u^i)\delta u)\end{aligned}$$

and $p^i$ is a solution of the followings.

$\dot{p}(t) = -f^T_x(t, x^i, u^i)p(t) + L^T_x(t, x^i, u^i)$

$p(t_1) = -G^T_x(x(t_1))$

**Step 5:** $\alpha_i = 1$.

**Step 6:** Set $u^{i+1}(t) = u^i(t) + \alpha_i\delta u^i(t) + \alpha^2_i(\tilde{u}^i(t) - u^i(t) - \delta u^i(t))$. if (10) holds, go to Step 7. Otherwise, set $\alpha_i = \beta\alpha_i$ and repeat Step 6.

$$\begin{aligned}J(u^{i+1}) - J(u^i) \leq \alpha_i M_2\{&G(x_i(t_1))\delta x(t_1) \\ &+ \int_{t_0}^{t_1}(L_x(t, x^i, u^i)\delta x^i + L_u(t, x^i, u^i)\delta u^i)\}\end{aligned} \quad (10)$$

**Step 7:** Set $i = i + 1$, and go to Step 3. Repeat Step 3 to Step 7 until the performance index $J$ converges. Here, the integer $i$ represents the number of iterations.

*2.3 Convergence*

We can prove the convergence property of the algorithm described in the previous subsection. The theorem below tells us that accumulation points generated by Algorithm, if they exist, satisfy the necessary conditions for optimality. See (Imae *et al.*, 1998) for more details.

**Theorem (convergence)**

Let $\{A^i\}_{i=0}^{\infty}, \{B_{11}^i\}_{i=0}^{\infty}, \{B_{12}^i\}_{i=0}^{\infty}, \{B_{22}^i\}_{i=0}^{\infty}, \{u^i\}_{i=0}^{\infty}$, $\{\tilde{u}^i\}_{i=0}^{\infty}$ and $\{\delta u^i\}_{i=0}^{\infty}$ be sequences generated by the above-mentioned algorithm. Suppose that there exists $M_4 \in (0,\ \infty)$ such that, for almost all $t \in [t_0,\ t_1]$, $\mid \tilde{u}^i \mid \leq M_4$ and $\mid \delta u^i \mid \leq M_4$, $i = 0, 1, 2, \cdots$ and also suppose that exist $\bar{A} \in \Re^{n\times n}$, $\bar{B}_{11} \in L_{\infty}^{n\times n}$, $\bar{B}_{12} \in L_{\infty}^{n\times r}$, $\bar{B}_{22} \in L_{\infty}^{r\times r}$, $\bar{u} \in L_{\infty}^{r}$ and a sequence $N \subset (0, 1, 2, 3, \cdots)$ such that $\bar{A} \geq 0$, $\bar{B}_{11}(t) - \bar{B}_{12}(t)(\bar{B}_{22}(t))^{-1}(\bar{B}_{12}^T(t)) \geq 0$ for almost all $t \in [t_0, t_1]$

$\lim_{i\in N} A^i = \bar{A}$ in the norm of $\Re^{n\times n}$

$\lim_{i\in N} B_{11}^i(t)(\text{resp.}, B_{12}^i(t), B_{22}^i(t))$
$= \bar{B}_{11}(t)(\text{resp.}, \bar{B}_{12}^i(t), \bar{B}_{22}^i(t))$
in the norm of $L^{n\times n}(\text{resp.}, L_{\infty}^{n\times r}, L_{\infty}^{r\times r})$

$\lim_{i\in N} u^i(t) = \bar{u}(t)$ in the norm of $L_{\infty}^{r}$

Here, $\bar{A}$, $\bar{B}_{11}$ and $\bar{B}_{22}$ are symmetric. Then, $\bar{u}(t)$ satisfies the weak necessary condition for optimality

$H_u(t, \bar{x}(t), \bar{u}(t), \bar{p}(t)) = 0$ a.e.in $t \in [t_0, t_1]$

where $H(t, x, u, p) = -L(t, x, u) + p^T f(t, x, u)$, $\bar{x}(t)$ is a solution of (1) with $\bar{u}$, and $\bar{p}(t)$ is a solution of the followings.

$$\dot{p}(t) = -f_x^T(t, \bar{x}, \bar{u})p(t) + L_x^T(t, x, u)$$
$$p(t_1) = -G_x^T(\bar{x}(t_1))$$

## 3. ALGORITHMIC DESIGN

The key idea is very simple. Roughly speaking, all we have to do is to proceed with such above-mentioned algorithm by one iteration only, periodically. This means that as time passes the number of iterations increases. That is, through sufficiently large number of iterations, it could be expected to eventually reach the possible optimal solutions. More detailed explanations are given in the following description.

From a practical point of view, the calculation time for one-iteration-ahead solution is assumed to be equal to $\Delta T$[ms] (or less than $\Delta T$). The calculation time $\Delta T$ plays a key role in our design method. We here describe how well the algorithmic controller works, together with Fig. 1.

**Algorithm**

**Step 1:** Measure an actual state $x_0$, and select arbitrarily an initial input $u^0$. Set the unit of calculation time $\Delta T$[ms], and apply the input $u^0$ to the plant over the interval of the first unit time of calculation. During the (say, Section 1) interval, proceed with two kinds of calculation. One is to predict the one-unit-time-ahead state $\hat{x}_1$ through system equation (1) with the initial state $x_0$, and the other is to calculate the one-iteration-ahead solution over $[\Delta T, t_1]$ with the updated initial state $\hat{x}_1$ as a new initial state. Then, denote by $k^1$ the feedback gain

$$-B_{22}^{-1}\{f_u(t, x, u)^T K + B_{12}^T\},$$

and by $u^1$ the input associated with Optimal trajectory 1.

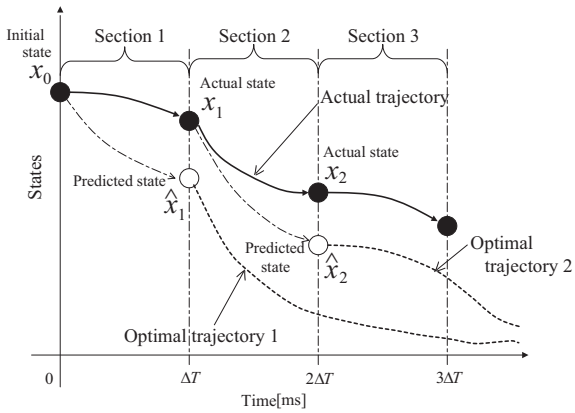


Fig. 1. Optimal/Actual trajectory

**Step 2:** Measure the actual state $x_1$, and apply the input $u^1$ together with the feedback gain $k^1$ to the plant over the interval of the second unit time of calculation. During such interval (say, Section 2), proceed with two kinds of computation. One is to predict the one-unit-time-ahead state $\hat{x}_2$ through the system equation (1) with the state $x_1$. The other is to calculate over $[2\Delta T, t_1]$ the one-iteration-ahead solution (say, Optimal trajectory 2), using $x_2$ as the new initial state. Then, denote by $k^2$ the feedback gain.

$$-B_{22}^{-1}\{f_u(t, x, u)^T K + B_{12}^T\}$$

and by $u^2$ the input corresponding to Optimal trajectory 2.

**Step 3:** Apply to the plant the input $u^3,\ u^4,\ \cdots$.

## 4. NUMERICAL SIMULATIONS

First of all, we consider a simple optimal control problem in order to illustrate how well our design method works.

**Van der Pol Oscillator**

System equation, initial condition, and performance index are as follows.

$$\dot{x}_1(t) = (1 - x_2^2(t))x_1(t) - x_2(t) + u$$
$$\dot{x}_2(t) = x_1(t) \qquad (11)$$
$$x_1(0) = 0, \quad x_2(0) = 1$$
$$J(u) = \frac{1}{2}\int_0^{\infty} (x_1^2(t) + x_2^2(t) + u^2(t))dt \quad (12)$$

Our goal is to find a real-time optimal controller in the algorithmic fashion. We apply the algorithmic design method to this oscillation problem.

In this example, $t_1 = \infty$ should be given. However, from the computational point of view, it is impossible to find a numerical solution with the terminal

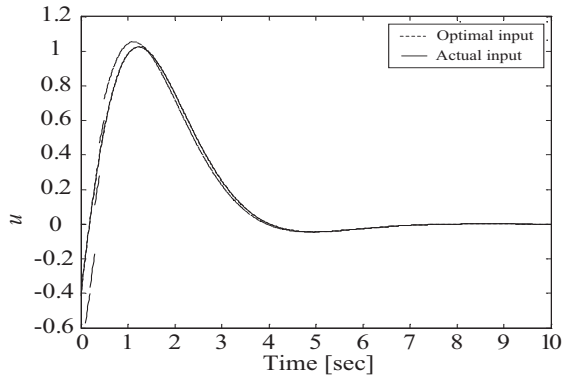

Fig. 2. Control input

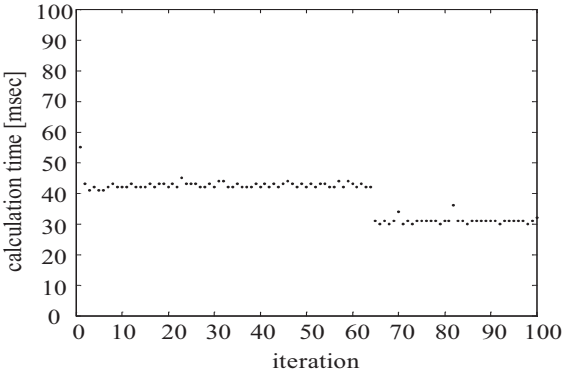

Fig. 3. Computing time

time being infinity. So, we here set $t_1 = 10$, which would be sufficiently large, and let the final time $t_1$ move as time goes by. The integration steps of differential equations are chosen to be 1000, and the input $u^0(t) = 0$ is adopted as an arbitrarily chosen initial one. The computation time for one-iteration-ahead solution is set by $\Delta T = 100$[ms]. Fig. 2 shows the computed input compared with the numerical optimal solution. The CPU time is given in Fig. 3, where the computation time for one-iteration-ahead solution has turned out less than 60[ms]. This implies that the algorithmic design methods are implementable in real-time optimization.
Fig. 4 and Fig. 5 show the behaviors of states and inputs, where the system equation

$$\dot{x}_1(t) = (1.5 - x_2^2(t))x_1(t) - x_2(t) + u$$

is adopted instead of the first equation of eq. (11). The computational results show that our approach is robust to the parameter variations generated in the system. We here note that our approach turns out feedforward in case of the feedback gain being zero.

## 5. SWING CONTROL OF CRANE

We demonstrate the effectiveness and practicability of our algorithmic design method by applying it to the practical optimal control problem, such as swing control of a crane system (see Fig 6).

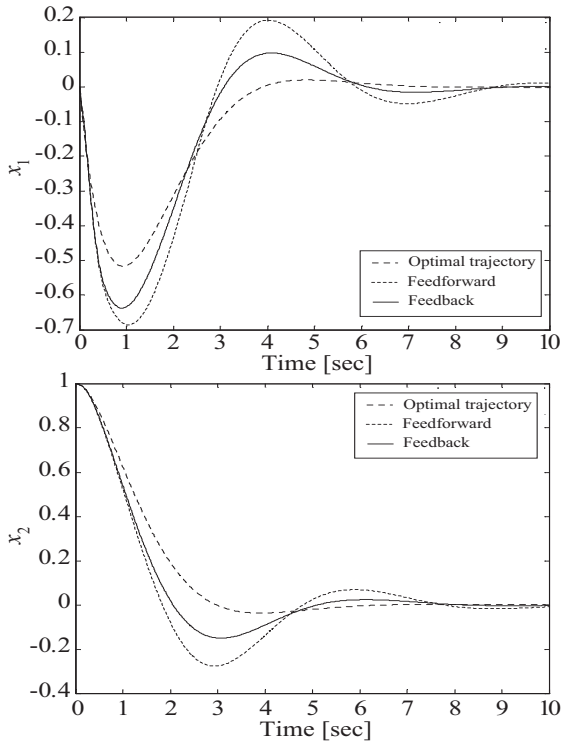

Fig. 4. Trajectories of states

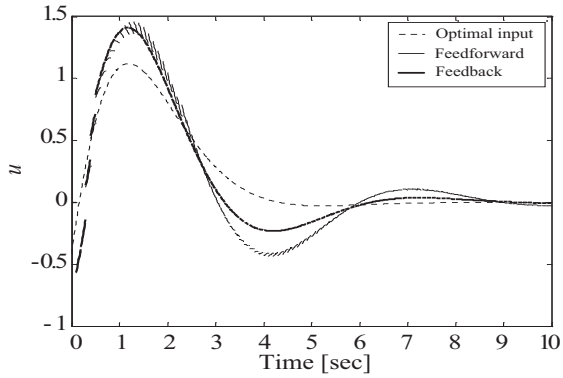

Fig. 5. Control input

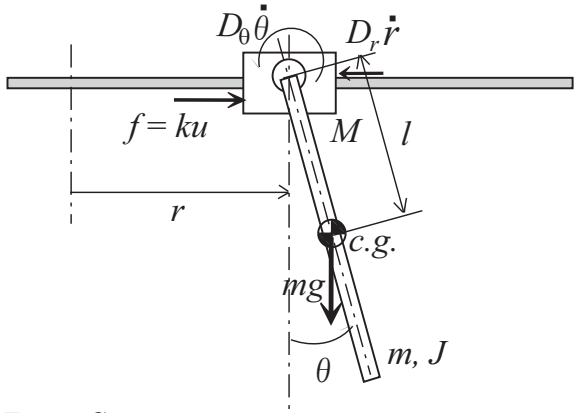

Fig. 6. Crane system

*5.1 Mathematical model*
System equation is given as follows. Denote by $r$ the position of the cart, by $\theta$ the angle of the pendulum, and by the input voltage to the DC motor. Therefore, the state variable $x$ is given as $x = [x_1\ x_2\ x_3\ x_4]^T = [r\ \theta\ \dot{r}\ \dot{\theta}]^T$ and the input variable $u$ is given as $u = v$. See Fig. 6

Table 1. Parameters of crane system

| | | |
|---|---|---|
| $M$ | mass of cart | 0.361 kg |
| $k_f$ | torque coefficient | 2.71 N/V |
| $J$ | moment of inertia | $2.79 \times 10^{-3}$ kgm$^2$ |
| $l$ | length of pendulum | 0.301 m |
| $m$ | mass of pendulum | 0.100 kg |
| $D_r$ | friction coefficient between rail and cart | 9.025 kg/s |
| $D_\theta$ | friction coefficient between cart and pendulum | $6.95 \times 10^{-4}$ kgm$^2$/s |
| $g$ | gravitational acceleration | 9.81m/s$^2$ |

$$\dot{x} = Ax + Bu$$

$$A = \frac{1}{\alpha_1}\begin{bmatrix} 0 & 0 & \alpha_1 & 0 \\ 0 & 0 & 0 & \alpha_1 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & a_{42} & a_{43} & a_{44} \end{bmatrix}, \quad B = \frac{1}{\alpha_1}\begin{bmatrix} 0 \\ 0 \\ b_3 \\ b_4 \end{bmatrix}$$

where

$$a_{32} = -(ml)^2 g \cos x_2 \frac{\sin x_2}{x_2}$$
$$a_{33} = -(J + ml^2)D_r$$
$$a_{34} = -(J + ml^2)mlx_4 \sin x_2 - mlD_\theta \cos x_2$$
$$a_{42} = -(M + m)mgl\frac{\sin x_2}{x_2}$$
$$a_{43} = -mlD_r x_3 \cos x_2$$
$$a_{44} = -(ml)^2 x_4 \cos x_2 \sin x_2 - (M + m)D_\theta$$
$$b_3 = k_f(J + ml^2), \; b_4 = k_f ml \cos x_2$$
$$\alpha_1 = MJ - (ml \cos x_3)$$

and the values of parameters are given in Table 1.

*5.2 Numerical simulations*

First, we introduce the following performance index.

$$J = \frac{1}{2}\int_0^\infty \{\ 100x_1^2(t) + x_2^2(t) + x_3^2(t) + x_4^2(t) + u^2(t)\}dt \quad (13)$$

We set $\Delta T = 100$[ms] and $t_1 = 10$[sec]. The integration steps of differential equations are chosen to be 1000 steps, and the input $u^0(t) = 0$ is adopted as an arbitrarily chosen initial one. It would be important to note that the terminal time $t_1$ is not fixed, that is, the terminal time $t_1$ moves as the time goes.

**Case of** $x_2(0) = 1.0$**[rad]**

We set $[x_1\ x_2\ x_3\ x_4]^T = [0\ 1.0\ 0\ 0]^T$. Fig. 7 tells us that the swing control of the crane system is successful. The CPU time is given in Fig. 8, where the computation time for one-iteration-ahead solution has turned out less than 70 [ms]. This means that the algorithmic method is also implementable in the nonlinear crane system. The behaviors of states are shown in Fig. 9.

*5.3 Experimental result*

Fig. 10 shows the experimental system. We here adopt the same performance index as given in the previous subsection. Similarily, we set $\Delta T = 100$[ms] and $t_1 = 10$[sec]. The integration steps of differential equations are chosen to be 1000, and the input $u^0(t) = 0$ is adopted as an arbitrarily chosen initial one.

**Case of** $x_2(0) = 2.0$**[rad]**

We take as an initial condition $[x_1\ x_2\ x_3\ x_4]^T =$

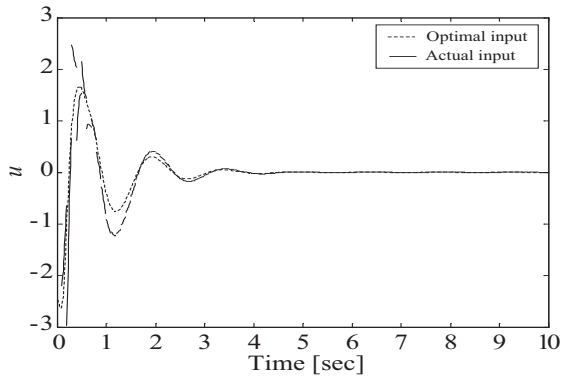


Fig. 7. Control input

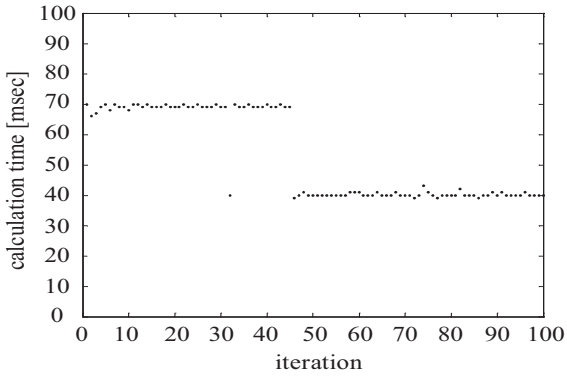


Fig. 8. Computing time

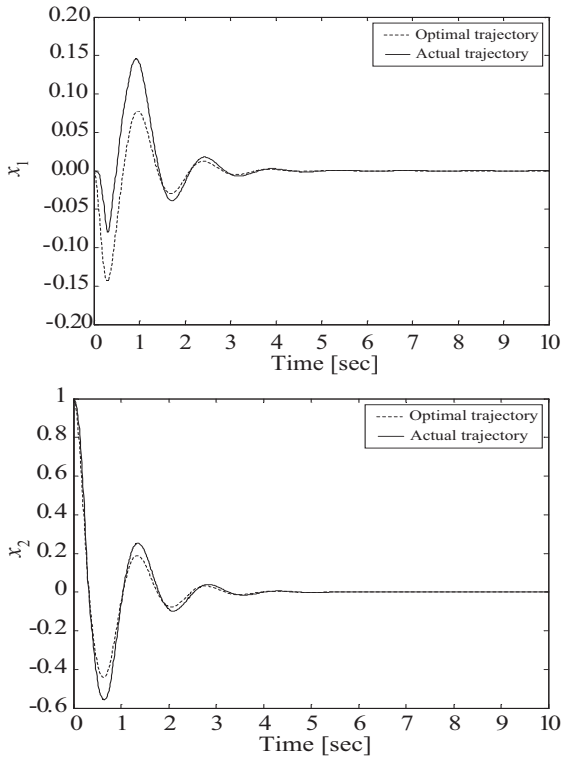


Fig. 9. Trajectories of states

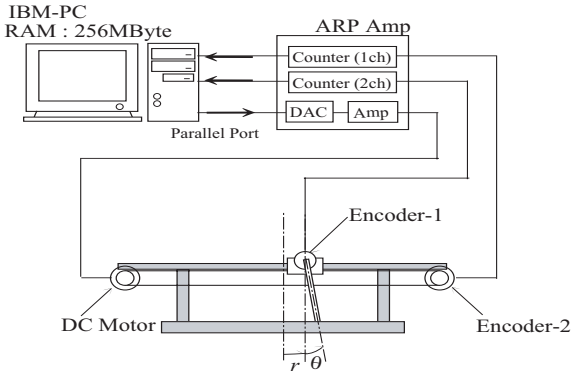


Fig. 10. Experimental system

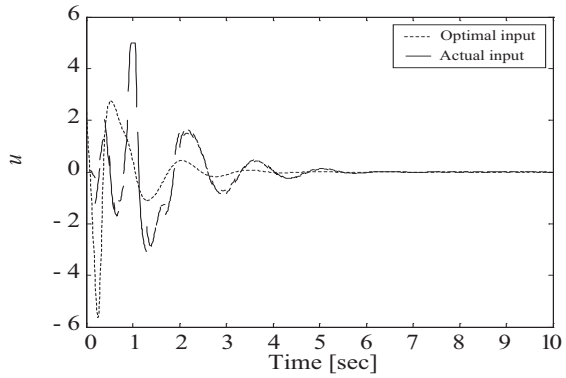


Fig. 11. Control input

$[0\ 2.0\ 0\ 0]^T$. The CPU time for one-iteration-ahead solution has turned out less than 70 [ms]. The behaviors of states and input are shown in Fig. 11 and 12, compared with the numerical optimal solutions.

## 6. CONCLUSION

A design method for real-time computation of nonlinear optimal control problems has been proposed. The proposed method is based on the optimal control algorithms. Therefore, whether or not the real-time computation succeeds depends on how effective the algorithm is. It has been demonstrated that, if the Riccati-equation-based algorithm is adopted, the proposed algorithmic controller is applicable to the optimal control problem. Simulation and experiment have been given in order to demonstrate the effectiveness and practicability of our approach, where the van der Pol problem and the swing control problem of a crane are adopted as design examples.

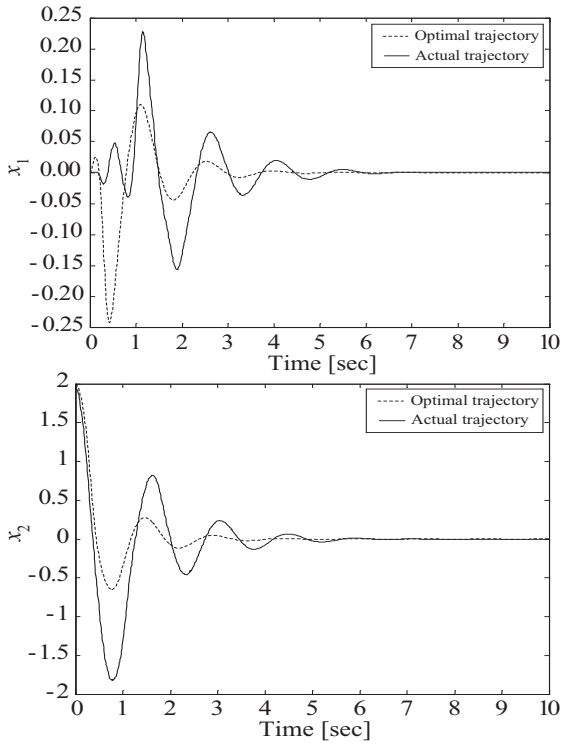


Fig. 12. Trajectories of states

## REFERENCES

C. W. Merriam III (1965) A computational method for feedback control optimization. *Information and Control* **8**, 215–232

D. H. Jacobson, and D. Q. Mayne (1970) *Differential dynamic programming.* American Elsevier.

D. M. Murray (1978) Differential dynamic programming for the efficient solution. of optimal control problems. *University of Arizona, PhD Thesis.*

F.Allgowet, et. al. (2000). *Nonlinear model predictive control.* Birkhauser.

J. Imae, L. Irlicht, G. Obinata and J. B. Moore (1992) Enhancing optimal controllers via techniques. from robust and adaptive control *International Journal of Adaptive Control and Signal Processing* **6**, 413–429

J. Imae, et. al. (1998). A Riccati equation based algorithm for nonlinear optimal control problems. *Proc. of the 37th IEEE CDC* , 4422–4427.

J. R. Coutier, et. al. (1996). Nonlinear regulation and nonlinear control via the state-dependent Riccati equation technique, Part 1&2. *Proc. 1st international conference on nonlinear problems in aviation and aerospace.*

N. B. Nedeljkovic (1981). New algorithms for unconstrained nonlinear optimal control problems. *IEEE Transactions on Automatic Control.* **26**, 868–884.

P. Dyer, and S. R. McReynolds (1970). *The computation and theory of optimal control.* Academic Press.

T. E. Bullock, and G. F. Franklin (1967). A modified quadratic cost problem and feedback stabilization of a linear system. *IEEE Transactions on Automatic Control.* **AC-12-6**,666–673

T. Ohtsuka (2004). A continuation/gmres method for fast computation of nonlinear receding horizon control. *Automatica* **40**, 563–574.