A DIFFERENCE BASED EFFICIENT APPROXIMATE ALGORITHM FOR MODEL PREDICTIVE CONTROL OF INPUT-CONSTRAINED LINEAR SYSTEMS

Yutaka Saitou [*] **Jun-ichi Imura** [*]

[*] *Tokyo Institute of Technology,*
Dep. of Information Science and Engineering
8W-1 2-12-1 Ookayama, Meguro-ku,Tokyo 152-8552,
JAPAN
{ysaitou|imura}@cyb.mei.titech.ac.jp

Abstract: This paper proposes a novel efficient algorithm for model predictive control (MPC) of input-constrained linear systems. Based on the fact that the problem of MPC is reduced into the problem of calculating the solution trajectory of the discrete-time linear complementarity (D-LC) systems, the proposed algorithm fully exploits the information on the solution of the problem at the previous time step, as like the difference technique used in the calculation of solution trajectories of a dynamical system. It is shown that the proposed algorithm is much more efficient than the other conventional algorithms at a large prediction horizon. *Copyright© 2005 IFAC.*

Keywords: Prediction control, Linear Complementarity Systems, Constrained Systems

1. INTRODUCTION

It is well known that the Model Predictive Control (MPC) problem of a input-constrained linear system to minimize a quadratic form cost function is reduced into the Quadratic Programming (QP) problem, and further, in terms of the Karush-Kuhn-Tucker condition, in to the Linear Complementarity (LC) problem including the current state of the system as a parameter. Thus it is also known that the closed loop system obtained by the MPC is equivalent to the Discrete-time Piece Wise Affine (D-PWA) system and the Discrete-time Linear Complementarity (D-LC) system (A. Bemporad and Schutter, 2002). Based on this fact, the off-line computation method by the multi-parametric QP algorithm, which is innovative, has been extensively developed (A. Bemporad and Pistikopoulos, 2002). However, its computational cost is too much for a large prediction horizon. On the other hand, some efficient on-line computation methods, although they are traditional, have been still extensively developed (see, e.g., the active set method (Maciejowski, 2002), the Lemke method based algorithm (Camacho, 1993), an algorithm for the special class of LC-problems (Saitou and Imura, 2004)). However, the existing on-line solvers are not still suitable for larger size of MPC problems. This is because they basically solve the QP or LC problem obtained at each time step using the *static* (i.e., usual) QP or LC solver.

This paper thus proposes a novel on-line computation algorithm for the MPC of input-constrained linear systems, which solves a trajectory generation problem of D-LC systems as like the difference technique in the calculation of solution trajectories of dynamical systems. Thus the proposed algorithm is efficient since it fully exploits the information on the solution at the previous step. Although the proposed algorithm gives only an approximated solution, it guarantees that its error converges to 0 as the adjustable parameter of the algorithm goes to infinity. Note that an algorithm based on the similar idea has been also proposed in (Ohtsuka, 2004) for nonlinear MPC; however, its approach solves a two point boundary value problem at each step, and needs to add some heuristic cost function to the original cost function. Thus, it is quite different from that of the proposed algorithm. It is shown by a numerical example that the proposed algorithm solves 30 times faster than the static algorithm based on the Lemke method at 100-step prediction horizon.

In the sequel, the following notation is used. The symbols I,O and $\mathbf{0}$ denote an identity matrix, a zero matrix, and a zero vector of proper dimension, respectively. For a vector x (or a matrix A), '$x \geq 0$' ('$A \geq 0$') means that every element of x (A) is greater than or equal to 0. The symbol a_{ij} denotes the (i,j)-th element of a matrix A, and x_i or $(x)_i$ denotes the i-th row of a vector x. Denote by A_i or $(A)_i$ the i-th row of a matrix A, and by $A_{\alpha\beta}$ a minor matrix of A, which is defined by the index sets $\alpha := \{i_1, \cdots, i_r\}$, $\beta := \{j_1, \cdots, j_r\}$ and composed of $\{i_1, \cdots, i_r\}$-th rows, $\{j_1, \cdots, j_r\}$-th columns of a matrix A. Furthermore, a difference set between α and β is denoted by $\alpha \setminus \beta$, and a complement set of a subset α on the index set α_{all} is denoted by $\alpha^{\mathbf{C}} := \alpha_{all} \setminus \alpha$.

2. PROBLEM FORMULATION

Consider an input-constrained discrete-time linear system given by

$$x_{t+1} = Ax_t + Bu_t \tag{1}$$

$$-w^- \leq u_t \leq w^+ \tag{2}$$

where, $x_t \in \mathbf{R}^n$,$u_t \in \mathbf{R}^m$ are the state and the input at t-step, respectively, $A \in \mathbf{R}^{n\times n}$ and $B \in \mathbf{R}^{n\times m}$ are constant matrices, and each element w_i^-, w_i^+ $(i = 1, \ldots, m)$ of w^-, w^+ is a positive constant.

On the MPC method, a finite-time optimal control problem is solved at the current step $t = k$, and only the first element u_k^* of an optimal input sequence $[u_k^*, u_{k+1}^*, \ldots, u_{k+N_p-1}^*]$ obtained as a solution to the problem is applied to the system, where N_p denotes a length of prediction horizon. Thus, we only have to focus on the finite-time optimal control problem as follows.

Problem 1. Suppose that the system of (1),(2) and the state x_k are given at $t = k$. Then find an input sequence $u_k^*, \ldots, u_{k+N_p-1}^*$ minimizing

$$J(x_k, u_k, u_{k+1}, \ldots, u_{k+N_p-1}) = x_{k+N_p}^\top Q_f x_{k+N_p} + \sum_{t=k}^{k+N_p-1} [x_t^\top Q x_t + u_t^\top R u_t]$$

where $Q, Q_f \in \mathbf{R}^{n\times n}$ are positive semi-definite, and $R \in \mathbf{R}^{m\times m}$ is positive definite. $\triangleleft$

Letting

$$H := \Gamma_{N_p}^\top \mathcal{Q} \Gamma_{N_p} + \mathcal{R} \quad \in \mathbf{R}^{mN_p \times mN_p},$$
$$F := \Psi_{N_p}^\top \mathcal{Q} \Gamma_{N_p} \quad \in \mathbf{R}^{n \times mN_p},$$
$$\mathcal{Q} := \mathrm{diag}[Q, \ldots, Q, Q_f], \ \mathcal{R} := \mathrm{diag}[R, \ldots, R, R],$$
$$\Psi_{N_p} := \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N_p} \end{bmatrix}, \ \Gamma_{N_p} := \begin{bmatrix} B & O \cdots O \\ AB & B \cdots O \\ \vdots & \ddots \ddots \vdots \\ A^{N_p-1}B & \cdots AB\, B \end{bmatrix},$$
$$U_k := [u_k^\top, u_{k+1}^\top, \ldots, u_{k+N_p-1}^\top]^\top \in \mathbf{R}^{mN_p},$$
$$G := \begin{bmatrix} I \\ -I \end{bmatrix} \in \mathbf{R}^{2mN_p \times mN_p}, \bar{W} := \begin{bmatrix} W^+ \\ W^- \end{bmatrix} \in \mathbf{R}^{2mN_p},$$
$$W^+ := [w_1^+ \cdots w_m^+ | \cdots | w_1^+ \cdots w_m^+]^\top \in \mathbf{R}^{mN_p},$$
$$W^- := [w_1^- \cdots w_m^- | \cdots | w_1^- \cdots w_m^-]^\top \in \mathbf{R}^{mN_p},$$

Problem 1 is rewritten as the following QP problem.

Problem 2. Suppose that x_k is given at $t = k$. Then find an input vector U_k^* minimizing the cost function $J(x_k, U_k)$ given by

$$J = \frac{1}{2} U_k^\top H U_k + x_k^\top F U_k \quad \text{s.t. } GU_k \leq \bar{W}. \quad \triangleleft$$

Furthermore, Problem 2 is reduced into the following equivalent LC problem by the KKT-condition.

Problem 3. Suppose that x_k is given at $t = k$. Then find a pair of vectors $\delta_k, \lambda_k \in \mathbf{R}^{2mN_p}$ satisfying

$$\delta_k = \bar{K}\lambda_k + q(x_k), \quad q(x_k) := \bar{S}x_k + \bar{W} \tag{3}$$

$$\delta_k \geq 0, \ \lambda_k \geq 0, \ (\delta_k)_i(\lambda_k)_i = 0 \ \ \forall i \in \mathcal{I} \tag{4}$$

where

$$\bar{K} := GH^{-1}G^\top = \begin{bmatrix} K & -K \\ -K & K \end{bmatrix}, \ \bar{S} := \begin{bmatrix} S \\ -S \end{bmatrix}, \tag{5}$$

$$K := H^{-1}, \ S := H^{-1}F^\top, \ \mathcal{I} = \{1, \cdots, mN_p\}. \ \triangleleft$$

δ_k^*, λ_k^* denote an exact solution for Problem 3. Each element of δ_k, λ_k is usually called the complementarity variable. Since δ_k is obtained by x_k and λ_k, we can consider only λ_k^* as a solution. If $\delta_k \geq 0, \lambda_k \geq 0$ hold, then it is called "λ_k *is feasible*" and if $(\delta_k)_i(\lambda_k)_i = 0 \ \forall i \in \mathcal{I}$ hold, then it is called "λ_k *is complementary*" (R. W. Cottle and Stone, 1992). An exact solution of the LC-problem is a feasible and complementary vector λ_k. For simplicity, the symbol LCP$(q(x_k), \bar{K})$ is used as the LC-problem of (3)-(5) with $q(x_k)$, $\bar{K}$.

Since Problem 2 is equivalent to Problem 3, U_k^* is rewritten by x_k and λ_k^* as follows.

$$U_k^* = -(H^{-1}F^\top x_k + H^{-1}G^\top \lambda_k^*).$$

Then u_k^* is obtained by

$$u_k^* = [I_m\ O]\ U_k^* = -(F_1 x_k + F_2 \lambda_k^*), \quad (6)$$

where $F_1 := [I_m\ O]H^{-1}F^\top$, $F_2 := [I_m\ O]H^{-1}G^\top$.

Now, substituting (6) to (1), the closed loop system in MPC is obtained as follows (A. Bemporad and Schutter, 2002).

Definition 1. (Discrete time -LC system)

$$\begin{cases} x_{t+1} = \bar{A}x_t + \bar{B}\lambda_t \\ \delta_t = \bar{K}\lambda_t + q(x_t) \\ \delta_t \geq 0,\ \lambda_t \geq 0,\ (\delta_t)_i(\lambda_t)_i = 0 \ \ \forall i \in \mathcal{I} \end{cases} \quad (7)$$

where $m' := mN_p$, $\bar{A} := A - BF_1 \in \mathbf{R}^{n\times n}$, $\bar{B} := -BF_2 \in \mathbf{R}^{n\times 2m'}$, $\mathcal{I} = \{1, \cdots, m'\}$. ◁

Thus the MPC problem for the system of (1), (2) is reduced into the following trajectory generation problem of the D-LC system.

Problem 4. Suppose that a solution $(x_{k-1}, \lambda_{k-1}^*, \delta_{k-1}^*)$ of the system (7) at $t = k-1$ is given. Then find a solution $(x_k, \lambda_k^*, \delta_k^*)$ at $t = k$. ◁

For Problem 4, x_{k-1}, λ_{k-1}^* leads to x_k by virtue of the first equation of (7). Furthermore, δ_k^* is obtained by x_k and λ_k^*. Thus, we only have to find λ_k^* for generating a solution trajectory of (7).

2.1 Mathematical preliminaries

We define some symbols used in this paper.

Definition 2. A class of $n \times n$ real matrices for which every principal minor is positive (nonnegative) is denoted as $\mathcal{P}(\mathcal{P}_0)$. The matrix $A \in \mathcal{P}(\mathcal{P}_0)$ is called a P (P_0)-matrix. ◁

An inverse matrix of a P-matrix is also a P-matrix, and a positive definite symmetry matrix is a P-matrix. Furthermore, by the property of eigenvalues of a positive semi-definite matrix and a positive definite matrix, the symmetry matrix H in Problem 2 is positive definite. In summary, the following equations hold.

$$H \in \mathcal{P}, \quad K = H^{-1} \in \mathcal{P}. \quad (8)$$

Thus, Problem 2 is the strictly convex QP problem and it is known that there exists a unique solution for LCP$(q(x_k), \bar{K})$ characterized by (5), (8) and for all $q(x_k)$ (Y. J. Lootsma and Çamlıbel, 1999).

Definition 3. For an arbitrary pair of vectors (λ, δ), the index sets α, β, γ are defined as

$$\alpha := \{i \in \mathcal{I} |\ (\lambda)_i > (\delta)_i\} \quad (9)$$

$$\beta := \{i \in \mathcal{I} \mid (\lambda)_i = (\delta)_i\}, \quad (10)$$

$$\gamma := \{i \in \mathcal{I} \mid (\lambda)_i < (\delta)_i\}. \quad (11)$$

By definition, $\mathcal{I} = \{\alpha \cup \beta \cup \gamma\}$ holds. The index sets for (λ_k, δ_k) at k-step are expressed as $\alpha_k, \beta_k, \gamma_k$, and the index sets for some vector $(\tilde{\lambda}, \tilde{\delta})$ are expressed as $\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}$. For simplicity, we omit their subscript k as α, β, γ as long as it is not confusing.

Definition 4. For index sets α, β, γ, the following permutation between λ and $\hat{\lambda}$ is defined. The permutation of '$\lambda \rightarrow \hat{\lambda}$' is called 'forward permutation', and '$\lambda \leftarrow \hat{\lambda}$' is also called 'backward permutation'. ◁

$$\lambda = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_{2m'} \end{bmatrix} \begin{matrix} FWD \\ \longrightarrow \\ \longleftarrow \\ BWD \end{matrix} \begin{bmatrix} \lambda_\alpha \\ \hline \lambda_\beta \\ \hline \lambda_\gamma \end{bmatrix} =: \hat{\lambda}$$

Remark 1. The permutation between δ and $\hat{\delta}$ is defined in the same way, and a matrix M is also suitably permuted. This type of permutation does not change the property of a P (P_0)-matrix, i.e., $M \in \mathcal{P}(\mathcal{P}_0) \Leftrightarrow \hat{M} \in \mathcal{P}(\mathcal{P}_0)$ holds. In addition, the LC-problem does not lose any generality under this permutation. So we will use λ and $\hat{\lambda}$ as the same vector as long as it is not confusing. ◁

In addition, we define a novel term for a solution, which is useful for discussion hereafter.

Definition 5. Suppose that a pair of vectors (λ, δ) satisfies the following equations

$$\lambda \geq 0,\ (\lambda)_i(\delta)_i = 0 \ \ \forall i \in \mathcal{I}. \quad (12)$$

Then it is called "λ *is semi-complementary*". ◁

3. DIFFERENCE BASED EFFICIENT APPROXIMATE ALGORITHM

The conventional static algorithm solves LCP $(q(x_k), \bar{K})$ at each k-step, not exploiting the information on the solution at the previous step, which is available at k-step. In order to derive an algorithm that fully exploits the previous information, we show the following theorems.

Theorem 1. Suppose that a solution $(x_{k-1}, \lambda_{k-1}^*, \delta_{k-1}^*)$ of (7) at $k-1$ step is given. Define the difference vector $\Delta x := x_k - x_{k-1}$, the index sets $\alpha_{k-1}, \beta_{k-1}, \gamma_{k-1}$ corresponding to $(\lambda_{k-1}^*, \delta_{k-1}^*)$, and the following linear function;

$$\bar{\Lambda}(\alpha_{k-1}, \Delta x_k) := \begin{bmatrix} \Lambda(\alpha_{k-1}, \Delta x_k) \\ \mathbf{0}_{\alpha_{k-1}^{\mathbf{C}}} \end{bmatrix}, \quad (13)$$

where $\Lambda(\alpha_{k-1}, \Delta x_k) := -(\bar{K}_{\alpha_{k-1}\alpha_{k-1}})^{-1}\bar{S}_{\alpha_{k-1}}\Delta x_k$, $\bar{K}_{\alpha_{k-1}\alpha_{k-1}}$ is nonsingular.

Furthermore, consider $(\tilde{\lambda}, \tilde{\delta})$ given by

$$\tilde{\lambda} = \lambda_{k-1}^* + \bar{\Lambda}(\alpha_{k-1}, \Delta x_k),\ \ \tilde{\delta} = \bar{K}\tilde{\lambda} + q(x_k) (14)$$

and compute the index sets $\tilde{\alpha}$,$\tilde{\beta}$ from $(\tilde{\lambda}, \tilde{\delta})$. Then if $\tilde{\alpha} = \alpha_{k-1}, \tilde{\beta} = \beta_{k-1} = \emptyset$ hold, a pair of vector $(\tilde{\lambda}, \tilde{\delta})$ is equivalent to an exact solution to LCP$(q(x_k), \bar{K})$, i.e., $\lambda_k^* = \tilde{\lambda}$ and $\delta_k^* = \tilde{\delta}$ hold.

The proof is omitted due to the limited space. We call the linear function (13) as the *Difference Value Function (DVF)* hereafter. The nonsingularity of $\bar{K}_{\alpha_{k-1}\alpha_{k-1}}$ will be shown in Lemma 1. Theorem 1 shows that an exact solution of the LC-problem is obtained by adding a DVF, under some conditions. The DVF is based on the stability and the differentiability of a solution for the LC problem. See (R. W. Cottle and Stone, 1992) for the details of these properties. Next, we prove that $\bar{K}_{\alpha_{k-1}\alpha_{k-1}}$ of (13) is nonsingular.

Lemma 1. Suppose that (8) holds and index sets α, β are given for a pair of vector (λ, δ) satisfying (12). Then, for every index set $\xi \subseteq \{\alpha \cup \beta\}$, the matrix $\bar{K}_{\xi\xi}$ is a P-matrix.

The proof is omitted due to the limited space. Since $\bar{K}_{\alpha_{k-1}\alpha_{k-1}}$ is a P-matrix by Lemma 1, it is obviously nonsingular. On the other hand, if $\tilde{\alpha} = \alpha_{k-1}$, $\beta_{k-1} = \emptyset$ do not hold, it is easily shown that a pair of $(\tilde{\lambda}, \tilde{\delta})$ is not feasible by the definition of each index set. However, an error is expected to be smaller as Δx is smaller, since the DVF given by (13) is a linear function of Δx. Based on this idea, we propose an approximate algorithm as follows, where a scaling of Δx is changed according to variation of an index set α.

Proposed algorithm (DBA)
[Main routine]
[Step 0] (Initialization) Given x_0, $(\lambda_0^*, \delta_0^*)$, positive integers ν_1, ν_2. Calculate $x_1 = \bar{A}x_0 + \bar{B}\lambda_0^*$, $k = 1$, $(\lambda_0, \delta_0) = (\lambda_0^*, \delta_0^*)$. Then go to [Step 1].

[Step 1] Define $\Delta x_k := x_k - x_{k-1}$, small intervals $d_{(1)}, d_{(2)}, \ldots\ d_{(\nu_1)}$ such that

$$d_{(\ell_1)} := \left[x_{k-1} + \frac{\ell_1 - 1}{\nu_1}\Delta x, x_{k-1} + \frac{\ell_1}{\nu_1}\Delta x\right].$$

Also define index sets $\tilde{\alpha}_{(\ell_1,1)}, \tilde{\beta}_{(\ell_1,1)}, \tilde{\gamma}_{(\ell_1,1)}$ at the left end point of each $d_{(\ell_1)}$, index sets $\tilde{\alpha}_{(\nu_1+1,1)}$, $\tilde{\beta}_{(\nu_1+1,1)}$, $\tilde{\gamma}_{(\nu_1+1,1)}$ at the right end point of $d_{(\nu_1)}$. Calculate $\tilde{\lambda}_{(1,1)} = \lambda_{k-1}$, $\tilde{\delta}_{(1,1)} = \delta_{k-1}$, $\tilde{\alpha}_{(1,1)} = \alpha_{k-1}$, and $\tilde{\beta}_{(1,1)} = \beta_{k-1}$. Substitute $\ell_1 = 1$ and go to [Step 2].

[Step 2] Branch as follows.
(i) If $\tilde{\alpha}_{(\ell_1,1)} = \tilde{\beta}_{(\ell_1,1)} = \emptyset$, then calculate $\tilde{\lambda}_{(\ell_1+1,1)} = \mathbf{0}$, $\tilde{\delta}_{(\ell_1+1,1)} = q(x_{k-1} + (\ell_1/\nu_1) \cdot \Delta x_k)$ and go to [Step 4].
(ii) If $\tilde{\beta}_{(\ell_1,1)} \neq \emptyset$, then update $\tilde{\beta}_{(\ell_1,1)} \leftarrow \emptyset$, $\tilde{\alpha}_{(\ell_1,1)} \leftarrow \{\tilde{\alpha}_{(\ell_1,1)} \cup \tilde{\beta}_{(\ell_1,1)}\}$ and go to [Step 2]-(iii).
(iii) If $\tilde{\alpha}_{(\ell_1,1)} \neq \emptyset$, $\tilde{\beta}_{(\ell_1,1)} = \emptyset$, then calculate

$$\tilde{\lambda}_{(\ell_1+1,1)} = \tilde{\lambda}_{(\ell_1,1)} + \bar{\Lambda}(\tilde{\alpha}_{(\ell_1,1)}, (1/\nu_1) \cdot \Delta x_k),$$
$$\tilde{\delta}_{(\ell_1+1,1)} = \bar{K}\tilde{\lambda}_{(\ell_1+1,1)} + q(x_{k-1} + (\ell_1/\nu_1) \cdot \Delta x_k),$$

and go to [Step 3].

[Step 3] For brevity of notation, $\tilde{\alpha}_{(\ell_1,1)}$ is denoted as $\tilde{\alpha}$, and the complement set of $\tilde{\alpha}$ as $\tilde{\alpha}^{\mathbf{C}}$ here. Update elements whose index belongs to $\tilde{\alpha}$ as

$$(\tilde{\lambda}_{(\ell_1+1,1)})_{\tilde{\alpha}} \leftarrow (\tilde{\lambda}_{(\ell_1+1,1)})_{\tilde{\alpha}} - (\bar{K}_{\tilde{\alpha}\tilde{\alpha}})^{-1}(\tilde{\delta}_{(\ell_1,1)})_{\tilde{\alpha}}.$$

Then calculate

$$\tilde{\delta}_{(\ell_1+1,1)} = \bar{K}\tilde{\lambda}_{(\ell_1+1,1)} + q(x_{k-1} + (\ell_1/\nu_1) \cdot \Delta x_k),$$

and go to [Step 4].

[Step 4] Calculate $\tilde{\alpha}_{(\ell_1+1,1)}, \tilde{\beta}_{(\ell_1+1,1)}$ by $\tilde{\lambda}_{(\ell_1+1,1)}$, $\tilde{\delta}_{(\ell_1+1,1)}$. Then branch as follows.
(i) If $\{\tilde{\alpha}_{(\ell_1+1,1)} \cup \tilde{\beta}_{(\ell_1+1,1)}\} = \tilde{\alpha}_{(\ell_1,1)}$, then go to [Step 5].
(ii) If $\{\tilde{\alpha}_{(\ell_1+1,1)} \cup \tilde{\beta}_{(\ell_1+1,1)}\} \neq \tilde{\alpha}_{(\ell_1,1)}$, then go to **[Subroutine]**. Note that $\tilde{\alpha}_{(\ell_1,1)}$ in this step has been updated in [Step 2].

[Step 5] Branch as follows.
(i) If $\ell_1 < \nu_1$, then update $\ell_1 \leftarrow \ell_1 + 1$ and go to [Step 2].
(ii) If $\ell_1 = \nu_1$, then calculate $\lambda_k = \tilde{\lambda}_{(\nu_1+1,1)}$ and go to [Step 6].

[Step 6] Calculate

$$x_{k+1} = \bar{A}x_k + \bar{B}\lambda_k, \quad \delta_k = \bar{K}\lambda_k + q(x_k)$$

Then update $k \leftarrow k + 1$ and go to [Step 1]. ◁

[Subroutine]
[Step S-0] (Initialization) $\tilde{\lambda}_{(\ell_1,1)}$, $\tilde{\delta}_{(\ell_1,1)}$, $\tilde{\alpha}_{(\ell_1,1)}$, $\tilde{\beta}_{(\ell_1,1)}$ are given by [Step 4] in the main routine. Define $\bar{\nu} := \nu_1\nu_2$, small intervals $d_{(\ell_1,1)} \cdots d_{(\ell_1,\ell_2)} \cdots d_{(\ell_1,\nu_2)}$ such that

$$d_{(\ell_1,\ell_2)} := \left[x_{k-1} + \left(\frac{\ell_1 - 1}{\nu_1} + \frac{\ell_2 - 1}{\nu_2}\right)\Delta x, \right.$$
$$\left. x_{k-1} + \left(\frac{\ell_1 - 1}{\nu_1} + \frac{\ell_2}{\nu_2}\right)\Delta x\right].$$

Define index sets $\tilde{\alpha}_{(\ell_1,\ell_2)}$, $\tilde{\beta}_{(\ell_1,\ell_2)}$ at the left end point of each $d_{(\ell_1,\ell_2)}$, index sets $\tilde{\alpha}_{(\ell_1,\nu_2+1)}$, $\tilde{\beta}_{(\ell_1,\nu_2+1)}$ at the right end point of $d_{(\ell_1,\nu_2)}$ and also $(\tilde{\lambda}_{(\ell_1,\ell_2)}, \tilde{\delta}_{(\ell_1,\ell_2)})$, which is solution candidate for LCP$(q(x_{k-1} + (\frac{\ell_1-1}{\nu_1} + \frac{\ell_2-1}{\nu_2}) \cdot \Delta x), \bar{K})$. Then calculate $\ell_2 = 1$ and go to [Step S-1].

[Step S-1] Update $\tilde{\beta}_{(\ell_1,\ell_2)} \leftarrow \emptyset$, $\tilde{\alpha}_{(\ell_1,\ell_2)} \leftarrow \{\tilde{\alpha}_{(\ell_1,\ell_2)} \cup \tilde{\beta}_{(\ell_1,\ell_2)}\}$. Then calculate $\bar{\Lambda}(\tilde{\alpha}_{(\ell_1,\ell_2)}, (1/\bar{\nu})\Delta x_k)$,

$$\tilde{\lambda}_{(\ell_1,\ell_2+1)} = \tilde{\lambda}_{(\ell_1,\ell_2)} + \bar{\Lambda}(\tilde{\alpha}_{(\ell_1,\ell_2)}, (1/\bar{\nu})\Delta x_k),$$
$$\tilde{\delta}_{(\ell_1,\ell_2+1)} = \bar{K}\tilde{\lambda}_{(\ell_1,\ell_2+1)} + q\left(x_{k-1} + \left(\frac{\ell_1 - 1}{\nu_1} + \frac{\ell_2}{\nu_2}\right)\Delta x_k\right),$$

and go to [Step S-2].

[Step S-2] Branch as follows.
(i) If $\tilde{\lambda}_{(\ell_1,\ell_2+1)} \geq 0$, then go to [Step S-4].
(ii) If $\exists i$, s.t. $(\tilde{\lambda}_{(\ell_1,\ell_2+1)})_i < 0$, then update $(\tilde{\lambda}_{(\ell_1,\ell_2+1)})_i \leftarrow 0$, $\tilde{\alpha}_{(\ell_1,\ell_2)} \leftarrow \tilde{\alpha}_{(\ell_1,\ell_2)} \setminus \{i\}$ for all i s.t. $(\tilde{\lambda}_{(\ell_1,\ell_2+1)})_i < 0$, and go to [Step S-3].

[Step S-3] For brevity of notation, $\tilde{\alpha}_{(\ell_1,\ell_2+1)}$ is denoted as $\tilde{\alpha}$ here.
Update elements whose index belongs to $\tilde{\alpha}$ as

$$(\tilde{\lambda}_{(\ell_1,\ell_2+1)})_{\tilde{\alpha}} \leftarrow (\tilde{\lambda}_{(\ell_1,\ell_2+1)})_{\tilde{\alpha}} - (\bar{K}_{\tilde{\alpha}\tilde{\alpha}})^{-1}(\tilde{\delta}_{(\ell_1,\ell_2)})_{\tilde{\alpha}}.$$

Then calculate

$$\tilde{\delta}_{(\ell_1,\ell_2+1)}=\bar{K}\tilde{\lambda}_{(\ell_1,\ell_2+1)}+q(x_{k-1}+(\frac{\ell_1-1}{\nu_1}+\frac{\ell_2}{\nu_2})\Delta x_k),$$

and go to [Step S-2].

[Step S-4] Calculate $\tilde{\alpha}_{(\ell_1,\ell_2+1)}$, $\tilde{\beta}_{(\ell_1,\ell_2+1)}$ by $\tilde{\lambda}_{(\ell_1,\ell_2+1)}$, $\tilde{\delta}_{(\ell_1,\ell_2+1)}$ and go to [Step S-5].

[Step S-5] Branch as follows.
(i) If $\ell_2 < \nu_2$, then update $\ell_2 \leftarrow \ell_2 + 1$ and go to [Step S-1]. (ii) If $\ell_2 = \nu_2$, then calculate $\tilde{\lambda}_{(\ell_1+1,1)} = \tilde{\lambda}_{(\ell_1,\nu_2+1)}, \tilde{\alpha}_{(\ell_1+1,1)} = \tilde{\alpha}_{(\ell_1,\nu_2+1)}$, $\tilde{\beta}_{(\ell_1+1,1)} = \tilde{\beta}_{(\ell_1,\nu_2+1)}$ and go to [Step 5] in **[Main routine]**. $\triangleleft$

The initial solution $(\lambda_0^*, \delta_0^*)$ can be obtained in advance by solving LCP$(q(x_0), \bar{K})$ by means of the conventional LC solvers. The iteration loop operation in the subroutine ([Step S-2] and [Step S-3]) terminates at most finite times, since the cardinality of $\tilde{\alpha}_{(\ell_1,\ell_2)}$ is finite and it is monotonically non-increasing. We can set positive integers ν_1, ν_2 as the designed parameters. The proposed algorithm replaces Δx in the DVF by $(1/\nu_1)\Delta x$, when index sets at both end points of $d_{(\ell_1)}$ are the same. On the other hand, in the case that index sets are different, it replaces Δx in the DVF by the smaller value $(1/\bar{\nu})\Delta x$ to decrease an error of an approximated solution. Next, we prove that an approximated solution obtained by the proposed algorithm converges to an exact solution as ν_1, ν_2 go to infinity. However, we show only a sketch of proof due to the limited space.

Theorem 2. Suppose that the initial solution $(x_0, \lambda_0^*, \delta_0^*)$ are given for system (7). For a solution $(x_k, \tilde{\lambda}_k, \tilde{\delta}_k)$, which is obtained by the proposed algorithm at any step k, the following relations hold.

$$\bar{\nu} \to \infty \Rightarrow \|\lambda_k^* - \tilde{\lambda}_k\|_\infty \to 0,\ \|\delta_k^* - \tilde{\delta}_k\|_\infty \to 0,(15)$$

where (λ_k^*,δ_k^*) expresses an exact solution of LCP$(q(x_k), \bar{K})$.

(**Sketch of Proof**) At first, The following statement (i),(ii) hold as the general property of a solution obtained by the proposed algorithm.

(i) Suppose that $\tilde{\lambda}_{(\ell_1,1)}$ given at [Step 2] is semi-complementary. Then $\tilde{\lambda}_{(\ell_1+1,1)}$ obtained at [Step 5] is also semi-complementary.

(ii) For a vector $\tilde{\lambda}$, which is semi-complementary, an error bound between $\tilde{\lambda}$ and λ_k^* is given by $\|\lambda_k^* - \tilde{\lambda}\|_\infty \le \mu_k \|[\tilde{\delta}^-]\|_\infty$, where $\tilde{\delta} := \bar{K}\tilde{\lambda} + q(x_k)$, $\tilde{\delta}^-$ is a vector that is composed of negative elements of $\tilde{\delta}$, and μ_k is a positive scalar depending on $q(x_k)$ and $\bar{K}$.

Next, by (i) and Theorem 1,the case of obtaining an exact solution is shown as follows.

(iii) If [Step 4]-(i) holds, a solution obtained at [Step 5] is an exact solution.

On the other hand, if [Step 4]-(ii) holds, an approximated solution is obtained at [Step 5] via the subroutine. Thus, first, let us consider the error bound for the solution obtained at [Step S-5] in the subroutine.

(iv) Suppose that $\tilde{\lambda}_{(\ell_1,\ell_2)}$, which is given at [Step S-1], is semi-complementary and satisfies $\bar{\nu} \to \infty \Rightarrow \|\tilde{\delta}^-_{(\ell_1,\ell_2)}\|_\infty \to 0$. Then $\tilde{\lambda}_{(\ell_1,\ell_2+1)}$ at [Step S-5] is semi-complementary and satisfies $\bar{\nu} \to \infty \Rightarrow \|\tilde{\delta}^-_{(\ell_1,\ell_2+1)}\|_\infty \to 0$.

The statement (iv) recursively holds from $\ell_2 = 1$ to $\ell_2 = \nu_2$, thus the following statement holds.

(v) Suppose that $\tilde{\lambda}_{(\ell_1,1)}$ is semi-complementary and satisfies $\bar{\nu} \to \infty \Rightarrow \|\tilde{\delta}^-_{(\ell_1,1)}\|_\infty \to 0$. If [Step 4]-(ii) holds, then $\tilde{\lambda}_{(\ell_1+1,1)}$ at [Step 5] is semi-complementary and satisfies $\bar{\nu} \to \infty \Rightarrow \|\tilde{\delta}^-_{(\ell_1+1,1)}\|_\infty \to 0$.

In summary, either of the statement (iii) or (v) recursively holds from $\ell_1 = 1$ to $\ell_1 = \nu_1$. Thus the following statement holds.

(vi) If $\tilde{\lambda}_k$ is semi-complementary and satisfies $\bar{\nu} \to \infty \Rightarrow \|\tilde{\delta}^-_{(k,1)}\|_\infty \to 0$, $\tilde{\lambda}_{k+1}$ is also semi-complementary and satisfies $\bar{\nu} \to \infty \Rightarrow \|\tilde{\delta}^-_{(k+1,1)}\|_\infty \to 0$.

Since $(\lambda_0^*, \delta_0^*)$ is given by assumption, finally, we have (15) for arbitrary k. □

4. EXAMPLE

In this section, the proposed algorithm is applied to a numerical example to show its efficiency. Consider the continuous-time input-constrained linear system given by

$$\dot{x}=A_c x + B_c u,\quad |u_1| \le 0.025,\ |u_2| \le 0.01,$$
$$A_c = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -25 & 25 & -1/2 & 1/6 \\ 300 & -300 & 2 & -2 \end{bmatrix}, B_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & -1 \\ 0 & 1 \end{bmatrix},$$

which models a kind of 2-d.o.f. positioning system (see Fig. 1). The discrete-time linear system is obtained by discretization with a sampling time $T_s = 0.05$ [sec]. We set Q, R as the following matrices:

$$\tilde{C} = \begin{bmatrix} 0 & 2 & 0 & 0 \end{bmatrix},\quad Q = \tilde{C}^\top \tilde{C},\quad R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.2 \end{bmatrix},$$

and use a solution for the discrete Riccati equation as a terminal weight cost function Q_f. The purpose of this problem is to drive the given initial state to an equilibrium state. The prediction horizon N_p is given as $N_p \in [10,\ 100]$, and the initial state is given as $x_0 = [0.1, -0.25, 0, 0\,]^\top$.

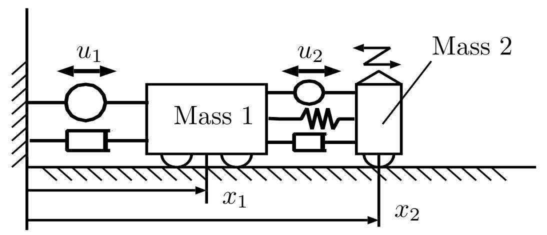

Fig. 1. Two-cart positioning system

This simulation is executed on Matlab 6.1 in a PC with Intel Pentium-M 1.60GHz CPU and a 1GB RAM. Fig. 2 shows trajectories of inputs (u_1, u_2), variation of an error $(e_1(k) = \|\lambda_k^* - \lambda_k\|_\infty)$ and trajectories of states (x_1, x_2) at $N_p = 40$ and $[\nu_1, \nu_2] = [7, 7]$. In this case, both inputs satisfy upper and lower constraints and an error is almost vanished. On the other hand, if $[\nu_1, \nu_2] = [2, 2]$ is chosen, the error becomes $O(10^{-3})$, where the input and position trajectories are still be similar to those in Fig. 2. The Worst Computational Time (W.C.T.) is compared in Fig.3, among the Matlab-QP solver, the Lemke method based solver, and two kinds of the proposed algorithm "DBA". The W.C.T. of the proposed algorithm at $N_p = 100$ is about 220 [msec] for DBA ($[\nu_1, \nu_2] = [2, 2]$) and 1030[msec] for DBA ($[\nu_1, \nu_2] = [7, 7]$), respectively. Thus, in the case of $[\nu_1, \nu_2] = [2, 2]$, it is 30 times faster than the the Matlab-QP or the Lemke method based solver. It is also shown that the W.C.T. of DBA does not increase at so high rate as the prediction horizon increases. In addition, since the source code of the proposed algorithm is written in the M-file format, we can expect to improve the W.C.T., if we use some compiler code.

5. CONCLUSION

This paper has proposed an efficient on-line computation algorithm for solving the MPC problem of input-constrained linear systems, which can efficiently solve the LC-problem that changes according to the linear dynamics, by fully exploiting the information on the solution at the previous step. It has been further proved that an approximated solution converges to an exact solution as some specified parameter goes to infinity. Finally, it has been shown by a numerical example that the proposed algorithm is more efficient than the Lemke method at a large prediction horizon.

REFERENCES

A. Bemporad, M. Morari, V. Dua and E. N. Pistikopoulos (2002). The explicit linear quadratic regulator for constrained systems. *Automatica* **38,No.1**, 3–20.

A. Bemporad, W. P. M. H. Heemels and B. D. Schutter (2002). On hybrid systems and closed-loop systems. *IEEE Trans. on Automatic Control* **47,No.5**, 863–869.

Camacho, E. F. (1993). Constrained generalized predictive control. *IEEE Trans. of AC* **38,No.2**, 327–332.

Maciejowski, J. M. (2002). *Predictive control with Constraints.* Pearson Education Limited. Harlow.

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

R. W. Cottle, J. S. Pang and R. E. Stone (1992). *The Linear Complementarity Problems.* Academic Press. San Diego.

Saitou, Y. and J. Imura (2004). An *M*-matrix based efficient algorithm for model predictive control of input constrained linear systems. *Proc. of Symposium on Large Scale Systems* pp. 777–782.

Y. J. Lootsma, A. J. van der Schaft and M. K. Çamlıbel (1999). Uniqueness of solutions of linear relay systems. *Automatica* **35**, 467/478.

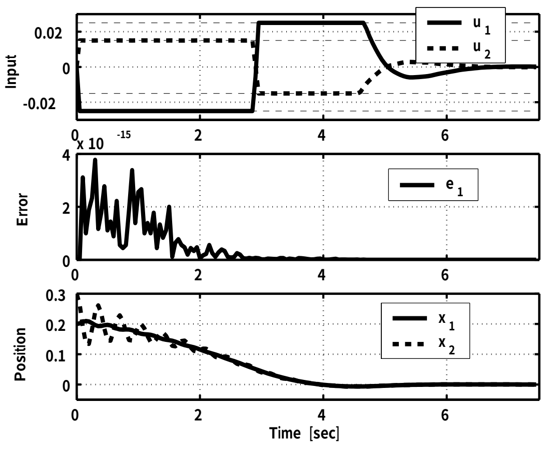

Fig. 2. Solution Trajectory and Error of input (N_p=40, DBA($\nu_1 = 7, \nu_2 = 7$))

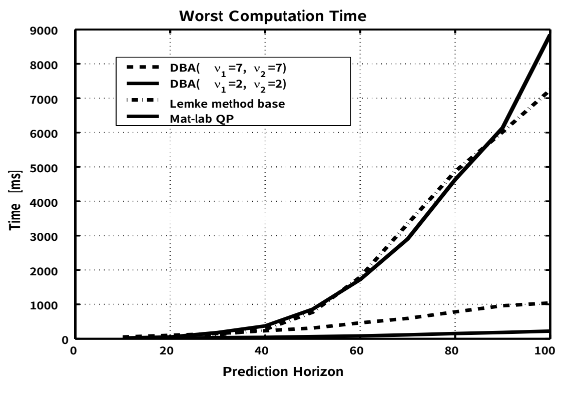

Fig. 3. Worst Computational Time vs Prediction horizon (Matlab-QP, Lemke-method, DBA(ν_1=2, ν_2=2), and DBA(ν_1=7, ν_2=7)