Properties of Multiobjective Robust Controller Using Difference Signals and Multiple Competitive Associative Nets in Control of Linear Systems

Weicheng Huang, Yuki Ishiguma, and Shuichi Kurogi

Kyushu Institute of Technology, Tobata, Kitakyushu, Fukuoka 804-8550, Japan
{ko@kurolab.,ishiguma@kurolab.,kuro@}cntl.kyutech.ac.jp
http://kurolab.cntl.kyutech.ac.jp/

Abstract. Recently, we have developed multiobjective robust controller using difference signals of nonlinear plant for multiple CAN2s to learn and approximate Jacobian matrices of the nonlinear dynamics. Here, the CAN2 is an artificial neural net for learning efficient piecewise linear approximation of nonlinear function. So far, by means of numerical experiments, we have shown that the controller is capable of coping with the change of plant parameter values as well as the change of control objective by means of switching multiple CAN2s. However, the controller have not been analyzed enough. This paper clarifies several properties of the controller by means of examining the control of linear plants.

Keywords: Multiobjective robust control, Switching of multiple CAN2s, Difference signals, Generalized predictive control, Jacobian matrix of nonlinear dynamics.

1 Introduction

Recently, we have developed multiobjective robust controller using difference signals of nonlinear plant to be controlled and multiple CAN2s (competitive associative nets) [1,2,3]. Here, the CAN2 is an artificial neural net introduced for learning efficient piecewise linear approximation of nonlinear function by means of competitive and associative schemes [5,6,7]. Thus, a CAN2 is capable of leaning piecewise Jacobian matrices of nonlinear dynamics of a plant by means of feeding difference signals of the plant to the CAN2. In [1], we have constructed a robust controller using multiple CAN2s to learn to approximate the plant dynamics for several parameter values. In [2], we have focused on a multiobjective robust control, where we consider two conflicting control objectives for a nonlinear crane system: one is to reduce settling time and the other is to reduce overshoot. Our method enables the controller to flexibly cope with those objectives by means of switching two sets of CAN2s for reducing settling time and overshoot, respectively. In [3], we have tried to improve the control performance by means of replacing single CAN2s by bagging CAN2s and shown several properties of the controller. From the point of view of multiobjective control [8], the settling time is reduced by tuning the number of units of the CAN2s, while the overshoot on average is reduced by bagging CAN2s replacing single CAN2s and an overshoot for the plant with certain parameter values is reduced by an augmentation of bagging CAN2s.

C.K. Loo et al. (Eds.): ICONIP 2014, Part III, LNCS 8836, pp. 58–67, 2014.

However, these properties as well as other properties of the controller have not been analyzed enough so far. In order to examine the controller, we analyze the controller by means of applying it to simple linear plants. In the next section, we show the method to control nonlinear and linear plant. In Sect. 3, we examine the method by means of numerical experiments applied to linear plants involving changeable parameter values.

2 Multiobjective Robust Controller Using Difference Signals and CAN2s

2.1 Plant Model Using Difference Signals

Suppose a plant to be controlled at a discrete time $j = 1, 2, \cdots$ has the input $u_j^{[p]}$ and the output $y_j^{[p]}$. Here, the superscript "[p]" indicates the variable related to the plant for distinguishing the position of the load, (x, y), shown below. Furthermore, suppose that the dynamics of the plant is given by

$$y_j^{[p]} = f(\boldsymbol{x}_j^{[p]}) + d_j^{[p]} \quad , \tag{1}$$

where $f(\cdot)$ is a nonlinear function which may change slowly in time and $d_j^{[p]}$ represents zero-mean noise with the variance σ_d^2. The input vector $\boldsymbol{x}_j^{[p]}$ consists of the input and output sequences of the plant as $\boldsymbol{x}_j^{[p]} \triangleq \left(y_{j-1}^{[p]}, \cdots, y_{j-k_y}^{[p]}, u_{j-1}^{[p]}, \cdots, u_{j-k_u}^{[p]}\right)^T$, where k_y and k_u are the numbers of the elements, and the dimension of $\boldsymbol{x}_j^{[p]}$ is given by $k = k_y + k_u$. Then, for the difference signals $\Delta y_j^{[p]} \triangleq y_j^{[p]} - y_{j-1}^{[p]}$, $\Delta u_j^{[p]} \triangleq u_j^{[p]} - u_{j-1}^{[p]}$, and $\Delta \boldsymbol{x}_j^{[p]} \triangleq \boldsymbol{x}_j^{[p]} - \boldsymbol{x}_{j-1}^{[p]}$, we have the relationship $\Delta y_j^{[p]} \simeq f_{\boldsymbol{x}} \Delta \boldsymbol{x}_j^{[p]}$ for small $\|\Delta \boldsymbol{x}_j^{[p]}\|$, where $f_{\boldsymbol{x}} = \partial f(\boldsymbol{x})/\partial \boldsymbol{x} \Big|_{\boldsymbol{x}=\boldsymbol{x}_{j-1}^{[p]}}$ indicates the Jacobian matrix (row vector). If $f_{\boldsymbol{x}}$ does not change for a while after the time j, then we can predict $\Delta y_{j+l}^{[p]}$ by

$$\widehat{\Delta y}_{j+l}^{[p]} = f_{\boldsymbol{x}} \widetilde{\Delta \boldsymbol{x}}_{j+l}^{[p]} \tag{2}$$

for $l = 1, 2, \cdots$, recursively. Here, $\widetilde{\Delta \boldsymbol{x}}_{j+l}^{[p]} = (\widetilde{\Delta y}_{j+l-1}^{[p]}, \cdots, \widetilde{\Delta y}_{j+l-k_y}^{[p]}, \widetilde{\Delta u}_{j+l-1}^{[p]}, \cdots, \widetilde{\Delta u}_{j+l-k_u}^{[p]})^T$, and the elements are given by

$$\widetilde{\Delta y}_{j+m}^{[p]} = \begin{cases} \Delta y_{j+m}^{[p]} & \text{for } m < 1 \\ \widehat{\Delta y}_{j+m}^{[p]} & \text{for } m \geq 1 \end{cases} \quad \text{and} \quad \widetilde{\Delta u}_{j+m}^{[p]} = \begin{cases} \Delta u_{j+m}^{[p]} & \text{for } m < 0 \\ \widehat{\Delta u}_{j+m}^{[p]} & \text{for } m \geq 0. \end{cases} \tag{3}$$

Here, $\widehat{\Delta u}_{j+m}^{[p]}$ $(m \geq 0)$ is the predictive input (see Sect. 2.3). Then, we have the prediction of the plant output from the predictive difference signals as

$$\widehat{y}_{j+l}^{[p]} = y_j^{[p]} + \sum_{m=1}^{l} \widehat{\Delta y}_{j+m}^{[p]}. \tag{4}$$

For linear plants, the plant function in (1) and the Jacobian matrix in (2) are modified as $f(\boldsymbol{x}_j^{[p]}) = \boldsymbol{A}\boldsymbol{x}_j^{[p]}$ and $f_{\boldsymbol{x}} = \boldsymbol{A}$, where $\boldsymbol{A} \in \mathbb{R}^{1\times k}$ is constant.

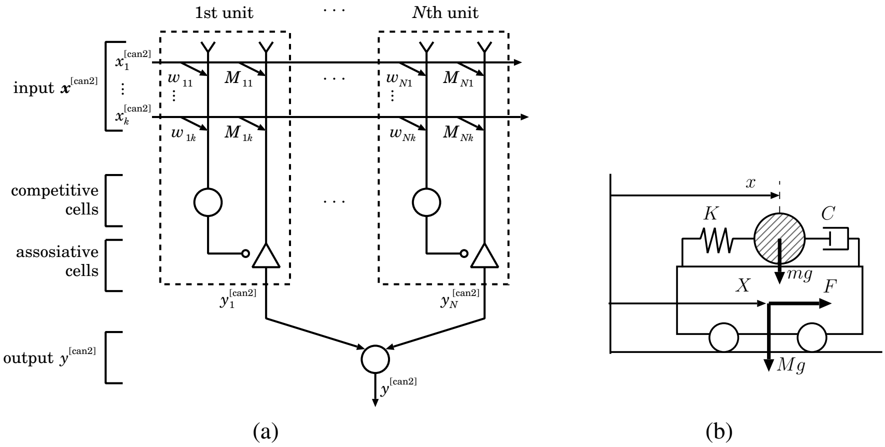

Fig. 1. Schematic diagram of (a) CAN2 and (b) a linear plant model of a car and the load

2.2 CAN2 for Learning and Identifying Nonlinear and Linear Plants

A CAN2 has N units. The ith unit has a weight vector $\boldsymbol{w}_i \triangleq (w_{i1}, \cdots, w_{ik})^T \in \mathbb{R}^{k\times 1}$ and an associative matrix (row vector) $\boldsymbol{M}_i \triangleq (M_{i1}, \cdots, M_{ik}) \in \mathbb{R}^{1\times k}$ for $i \in I = \{1, 2, \cdots, N\}$ (see Fig. 1(a)). For a given dataset $D^{[n]} = \{(\Delta \boldsymbol{x}_j^{[\mathrm{p}]}, \Delta y_j^{[\mathrm{p}]}) \mid j = 1, 2, \cdots, n\}$ obtained from the plant to be controlled, we train a CAN2 by feeding the input and output pair of the CAN2 as $(\boldsymbol{x}^{[\mathrm{can2}]}, y^{[\mathrm{can2}]}) = (\Delta \boldsymbol{x}_j^{[\mathrm{p}]}, \Delta y_j^{[\mathrm{p}]})$. We employ an efficient batch learning method shown in [10]. Then, for an input vector $\Delta \boldsymbol{x}_j^{[\mathrm{p}]}$, the CAN2 after the learning predicts the output $\Delta y_j^{[\mathrm{p}]} = f_{\boldsymbol{x}} \Delta \boldsymbol{x}_j^{[\mathrm{p}]}$ by

$$\widehat{\Delta y}_j^{[\mathrm{p}]} = \boldsymbol{M}_c \Delta \boldsymbol{x}_j^{[\mathrm{p}]}, \tag{5}$$

where c denotes the index of the unit selected by

$$c = \underset{i\in I}{\mathrm{argmin}}\, \|\Delta \boldsymbol{x}_j^{[\mathrm{p}]} - \boldsymbol{w}_i\|^2. \tag{6}$$

Here, we have assumed the following conjecture shown in [2,3]. Namely, $\boldsymbol{M}_c \simeq f_{\boldsymbol{x}}$ may not be identified via $\Delta \boldsymbol{x}_j^{[\mathrm{p}]}$ because $f_{\boldsymbol{x}}$ is not the function of $\Delta \boldsymbol{x}_j^{[\mathrm{p}]}$ generally. However, an enlarged vector $\Delta \boldsymbol{z}_j^{[\mathrm{p}]} = (\Delta y_{j-1}^{[\mathrm{p}]}, \cdots, \Delta y_{j-k'_y}^{[\mathrm{p}]}, \Delta u_{j-1}^{[\mathrm{p}]}, \cdots, \Delta u_{j-k'_u}^{[\mathrm{p}]})$ for $k'_y = k + k_y$ and $k'_u = k + k_u$ enables a Jacobian matrix $f_{\boldsymbol{z}} = \partial f / \partial \boldsymbol{z}$ to be a function of $\Delta \boldsymbol{z}_j^{[\mathrm{p}]}$ when the elements in $\Delta \boldsymbol{z}_j^{[\mathrm{p}]}$ vary sufficiently and the plant parameter does not change for a while. Thus, the above method with $\Delta \boldsymbol{x}_j^{[\mathrm{p}]}$ in (6) replaced by an enlarged $\Delta \boldsymbol{z}_j^{[\mathrm{p}]}$ is supposed to select an appropriate cth unit in the situation of multiobjective and robust control assuming the change of both plant parameters and control objectives. However, this conjecture is hard to be verified because Jacobian matrix for a certain duration of time involves approximation error in general, thus k_y and k_u to identify the Jacobian matrix depend on the approximation error allowable for the control.

On the other hand, the above conjecture does not seem to be applied to a linear plant with $f(\boldsymbol{x}_j^{[p]}) = \boldsymbol{A}\boldsymbol{x}_j^{[p]}$ because there is only one Jacobian matrix $f_{\boldsymbol{x}} = \boldsymbol{A}$. Thus, the CAN2 with multiple units after the learning of the plant dynamics is considered to have the following associative matrix for the ith unit as

$$\boldsymbol{M}_i = \boldsymbol{A} + \delta \boldsymbol{A}_i \tag{7}$$

where $\delta \boldsymbol{A}_i$ denotes approximation error of $\boldsymbol{A}$. The cth unit with the error $\delta \boldsymbol{A}_c$ is selected by (6) whose weight vector $\boldsymbol{w}_c$ is near to the current difference input vector $\Delta \boldsymbol{x}_j^{[p]}$ (or enlarged $\Delta \boldsymbol{z}_j^{[p]}$) consisting of the trained difference trajectory, i.e. $\Delta y_{j-l}^{[p]}$ and $\Delta u_{j-l}^{[p]}$ for $l = 1, 2, \cdots$. This interpretation of erroneous associative matrices can be also applied to the control of nonlinear plants, and seems more plausible than the above conjecture for nonlinear plants if the present controller also works for linear plants.

2.3 GPC Using Difference Signals

The GPC (Generalized Predictive Control) is an efficient method for obtaining the predictive input $\widehat{u}_j^{[p]}$ which minimizes the following control performance index [9]:

$$J = \sum_{l=1}^{N_y} \left(r_{j+l}^{[p]} - \widehat{y}_{j+l}^{[p]} \right)^2 + \lambda_u \sum_{l=1}^{N_u} \left(\widehat{\Delta u}_{j+l-1}^{[p]} \right)^2, \tag{8}$$

where $r_{j+l}^{[p]}$ and $\widehat{y}_{j+l}^{[p]}$ are desired output and predictive output, respectively. The parameters N_y, N_u and λ_u are constants to be designed for the control performance. We obtain $\widehat{u}_j^{[p]}$ by means of the GPC method as follows: at a discrete time j, use CAN2 to predict $\Delta y_{j+l}^{[p]}$ by (2) and then $\widehat{y}_{j+l}^{[p]}$ by (4). Then, owing to the linearity of these equations, the above performance index is written as

$$J = \|\boldsymbol{r}^{[p]} - \boldsymbol{G}\Delta \boldsymbol{u}^{[p]} - \overline{\boldsymbol{y}}^{[p]}\|^2 + \lambda_u \|\widehat{\Delta \boldsymbol{u}}\|^2 \tag{9}$$

where $\boldsymbol{r}^{[p]} = \left(r_{j+1}^{[p]}, \cdots, r_{j+N_y}^{[p]} \right)^T$ and $\widehat{\Delta \boldsymbol{u}}^{[p]} = \left(\widehat{\Delta u}_j^{[p]}, \cdots, \widehat{\Delta u}_{j+N_u-1}^{[p]} \right)^T$. Furthermore, $\overline{\boldsymbol{y}}^{[p]} = \left(\overline{y}_{j+1}^{[p]}, \cdots, \overline{y}_{j+N_y}^{[p]} \right)^T$ and $\overline{y}_{j+l}^{[p]}$ is the natural response $\widehat{y}_{j+l}^{[p]}$ of the system (1) for the null incremental input $\widehat{\Delta u}_{j+l}^{[p]} = 0$ for $l \geq 0$. Here, we actually have $\overline{y}_{j+l}^{[p]} = y_j^{[p]} + \sum_{m=1}^{l} \overline{\Delta y}_{j+m}^{[p]}$ from (4), where $\overline{\Delta y}_{j+l}^{[p]}$ denotes the natural response of the difference system of (2) with $f_{\boldsymbol{x}}$ replaced by $\boldsymbol{M}_c$. The ith column and the jth row of the matrix $\boldsymbol{G}$ is given by $G_{ij} = g_{i-j+N_1}$, where g_l for $l = \cdots, -2, -1, 0, 1, 2, \cdots$ is the unit step response $y_{j+l}^{[p]}$ of (4) for $\widehat{y}_{j+l}^{[p]} = \widehat{u}_{j+l}^{[p]} = 0$ $(l < 0)$ and $\widehat{u}_{j+l}^{[p]} = 1 (l \geq 0)$. It is easy to derive that the unit response g_l of (4) is obtained as the impulse response of (2). Then, we have $\widehat{\Delta \boldsymbol{u}}^{[p]}$ which minimizes J by $\widehat{\Delta \boldsymbol{u}}^{[p]} = (\boldsymbol{G}^T\boldsymbol{G} + \lambda_u \boldsymbol{I})^{-1}\boldsymbol{G}^T(\boldsymbol{r}^{[p]} - \overline{\boldsymbol{y}}^{[p]})$, and then we have $\widehat{u}_j^{[p]} = u_{j-1}^{[p]} + \widehat{\Delta u}_j^{[p]}$.

2.4 Control and Training Iterations

We execute iterations of the following phases to obtain the training data for CAN2s respectively and train the CAN2s.

(i) **Control Phase:** Control by a default control schedule at the first iteration, and by the GPC using the CAN2s obtained by the previous training phase otherwise.
(ii) **Training Phase:** Train the CAN2s with the dataset $D^{[n]} = \{(\Delta \boldsymbol{x}_j^{[\mathrm{p}]}, \Delta y_j^{[\mathrm{p}]} | j = 1, 2, \cdots, n)\}$ obtained in the control phase.

The control performance at an iteration depends on the CAN2 obtained at the previous iterations. So, for the actual control of the plant, we use the best CAN2s obtained through a number of iterations as shown below.

2.5 Switching Multiple CAN2s For Multiobjective Robust Control

To cope with the change of plant parameters and the change of control objective, we employ the following method to switch CAN2s for each control objective O_l $(l = 1, 2, \cdots)$. Let $\mathrm{CAN2}_{O_l}^{[\theta_s]}$ denote the best CAN2 from the point of view of O_l obtained for the plant with parameter θ_s $(s \in S = \{1, 2, \cdots, |S|\})$ through the above control and training iterations.

Step 1: At each discrete time j in the control phase, obtain $\boldsymbol{M}_c^{[s]}$ $(= \boldsymbol{M}_c$ in (6)) for all $\mathrm{CAN2}_{O_l}^{[\theta_s]}$ $(s \in S)$.
Step 2: Select the s^*th CAN2, or $\mathrm{CAN2}_{O_l}^{[\theta_{s^*}]}$, which provides the minimum MSE (mean square prediction error) for the recent N_e predictions, or

$$s^* = \underset{s \in S}{\mathrm{argmin}} \frac{1}{N_e} \sum_{l=0}^{N_e - 1} \left\| \Delta y_{j-l}^{[\mathrm{p}]} - \widehat{\Delta y}_{j-l}^{[\mathrm{p}][s]}) \right\|^2 , \tag{10}$$

where $\widehat{\Delta y}_{j-l}^{[\mathrm{p}][s]} = \boldsymbol{M}_c^{[s]} \Delta \boldsymbol{x}_{j-l}^{[\mathrm{p}]}$ (see (5)) denotes the prediction by $\mathrm{CAN2}_{O_l}^{[\theta_s]}$.

3 Numerical Experiments Using Linear Plant Model

3.1 A Car and Load System

We consider a linear model plant of a car and the load shown in Fig. 1(b). This model is derived from the overhead traveling crane system [3] by means of replacing the nonlinear crane by a load (mass) with a spring and a damper. From the figure, we have the motion equations given by

$$m\ddot{x} = -K(x - X) - C(\dot{x} - \dot{X}) \tag{11}$$

$$M\ddot{X} = F + K(x - X) \tag{12}$$

where x and X are the positions of the load and the car, respectively, m and M are the weights of the load and the car, respectively, K the spring constant, C the damping coefficient, and F is the driving force of the car. From the above equations, we have the following state-space representation for the state $\boldsymbol{x} = (x, \dot{x}, X, \dot{X})^T$,

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K}{m} & -\frac{C}{m} & \frac{K}{m} & \frac{C}{m} \\ 0 & 0 & 0 & 1 \\ \frac{K}{M} & 0 & -\frac{K}{M} & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{M} \end{bmatrix} F \tag{13}$$

3.2 Parameter Settings

Suppose that the controller has to move the load on the car from $x = 0$ to the destination position $x_d = 5$m by means of operating F. We obtain discrete signals by $u_j^{[\mathrm{p}]} = F(jT_v)$ and $y_j^{[\mathrm{p}]} = x(jT_v)$ with (virtual) sampling period $T_v = 0.5$s. Here, we use virtual sampling method shown in [4], where the discrete model is obtained with T_v (virtual sampling period) while the observation and operation are executed with shorter actual sampling period $T_a = 0.01$s. We have used $N_y = 20$, $N_u = 1$ and $\lambda_u = 0.01$ for the GPC. and $N_e = 8$ samples for (10).

The parameters of the plant are set as follows; th weight of the car $M = 100$kg, the spring constant $K = 15$ kg/s^2, the damping coefficient $C = 10$ kg/s, and the maximum driving force $F_{\max} = 10$N. To achieve the robustness to the load weight for $m = 10, 15, 20, \cdots, 100$ [kg], we train the CAN2s with $\mathrm{PLANT}^{[\theta_s]}$ for the load weight $\theta_s = m = 10, 40, 70, 100$ [kg] and $s = 1, 2, 3, 4$, respectively, where $\mathrm{PLANT}^{[\theta]}$ indicate the plant with the parameter θ. Let $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_s]}$ and $\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_s]}$ denote the best CAN2s which have achieved smallest overshoot and settling time, respectively, through 10 control and training iterations. Here, at each iteration, we train the CAN2 with the control dataset of two recent iterations, i.e. the current and the previous ones, because the number of obtained data becomes huge and time consuming as the number of iterations increases and the control performance does not seem improved even if we use all data. In order to uniquely select the CAN2, the overshoot x_{OS} and the settling time t_{ST} are ordered by $x_{\mathrm{OS}} + \epsilon t_{\mathrm{ST}}$ and $t_{\mathrm{ST}} + \epsilon x_{\mathrm{OS}}$, respectively, with small $\epsilon = 10^{-2}$. We have used the set of CAN2s, or $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_S]} = \{\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_s]} | s \in S\}$ and $\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_S]} = \{\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_s]} | s \in S\}$ for the switching controller explained in Sect. 2.5, where $S = \{1, 2, 3, 4\}$.

3.3 Results and Analysis

Result Using CAN2s with Single Units. First, we have examined the controller using true linear models and CAN2s with single units ($N = 1$). We use the input vector $\Delta \boldsymbol{x}_j^{[\mathrm{p}]}$ with $k_y = 4$ and $k_u = 1$, which is not enlarged $\Delta \boldsymbol{z}_j^{[\mathrm{p}]}$ but has the original minimum dimension of the true dynamics. From the experimental result shown in Table 1, we can see that the controller using true model has achieved increasing settling time t_{ST} and overshoot x_{OS} with the increase of the load weight $m = 10, 40, 70, 100$ [kg] for θ_i $(i = 1, 2, 3, 4)$. This is because the present controller uses the performance index J to be minimized shown in (9) with fixed control parameter values ($N_y = 20$, $N_u = 1$ and $\lambda_u = 0.01$). The conventional GPC has to tune the control parameters for minimizing t_{ST} and x_{OS} for the plants with different parameter values, while the present

Table 1. Experimental result of settling time t_{ST} [s] and overshoot x_{OS} [mm] obtained by the controller using true linear models $\mathrm{PLANT}^{[\theta_i]}$ and trained $\mathrm{CAN2}^{[\theta_i]}$ with single units. The ith raw from the top shows the result of the control of $\mathrm{PLANT}^{[\theta_i]}$ with $\theta_i = m =$ 10, 40, 70, 100 [kg] for $i = 1, 2, 3, 4$, respectively.

	t_{ST}	x_{OS}		t_{ST}	x_{OS}		t_{ST}	x_{OS}
$\mathrm{PLANT}^{[\theta_1]}$	33.8	0	$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_1]}$	32.4	186	$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_1]}$	32.7	170
$\mathrm{PLANT}^{[\theta_2]}$	35.1	15	$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_2]}$	22.5	9	$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_2]}$	24.2	0
$\mathrm{PLANT}^{[\theta_3]}$	41.5	86	$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_3]}$	15.9	63	$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_3]}$	27.5	9
$\mathrm{PLANT}^{[\theta_4]}$	48.5	141	$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_4]}$	29.9	44	$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_4]}$	30.1	43

controller shows different performances by using different CAN2s for minimizing t_{ST} and x_{OS}, respectively, as shown in Table 1. The difference of the performance is supposed to be obtained from the training datasets for the CAN2s derived from control trajectories which may involve degenerations and/or fluctuations through control and training iterations. However, the performance in Table 1 is not so good as to apply it to the switching control using multiple CAN2s, e.g. $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_1]}$ could not have achieved overshoot less than $x_{\mathrm{OS}} =$170[mm] for the plant θ_1 with $m = 10$ [kg].

Result Using CAN2s with Multiple Units. In order to improve the control performance, we use CAN2s with multiple units. Here, note that the CAN2s with multiple units involve erroneous models as shown in (7). However, the batch learning algorithm of the CAN2 (see [10]) tries to reduce the total approximation error for a given training dataset by means of using the condition called asymptotic optimality to equalize the approximation errors for all units of the CAN2. Thus, we may expect that the error of the associative matrix in (7), $\delta \boldsymbol{A}_i = \boldsymbol{M}_i - \boldsymbol{A}$, does not grow so much for all units and provides a variety of allowable control performances.

A statistical result of settling time t_{ST} and overshoot x_{OS} obtained by the controllers using multiple units is shown in Table 2, and four examples of time course of the input F, the output X and x for the best and the worst control result using multiple CAN2s are shown in Fig. 2. We can see that the best control for reducing settling time (top left) and overshoot (lower right) are reasonable, while the worst control for reducing settling time (top right) and overshoot (bottom right) are not so bad from their objectives.

From Table 2, we can see that the mean, max and std of settling time achieved by the controller using multiple $\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_S]}$ are smaller than those by the controller using single $\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_s]}$ for $s = 1, 2, 3, 4$. Incidentally, the controller using $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_2]}$ has achieved smaller mean, min and std of settling time, but $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_2]}$ is the CAN2 having achieved the minimum overshoot for θ_2 and we cannot find out any reason for this good performance in settling time.

On the other hand, the controller using multiple $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_S]}$ could not achieved smaller performance than the controller using single $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_3]}$. It seems that this is owing that $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_S]}$ involves $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_2]}$ which has a big mean overshoot 49.4[mm]. In our previous study [3], we have shown a method of augmentation of CAN2s to reduce plant-parameter-specific overshoots, and we apply the method as follows. First, we examined

Table 2. Statistical summary of the performance obtained by the controller using CAN2s with multiple units for the control of test plants with m = 10, 15, 20,$\cdots$, 100 [kg]. The columns of "trained θ_i" indicate the result by the controller applied to the training plants θ_i with m = 10, 40, 70, 100 [kg] for $i = 1, 2, 3, 4$, respectively. The columns of "mean", "min", "max" and "std" for "settling time" and "overshoot" indicate the minimum, maximum and standard deviation of the control result for all test plants. We denote $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_{S'}]}$=$\mathrm{CAN2}_{\mathrm{OS}}^{[70\mathrm{kg}]}\cup\mathrm{CAN2}_{\mathrm{OS}}^{[97\mathrm{kg}]}$ and $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_{S''}]}$=$\mathrm{CAN2}_{\mathrm{OS}}^{[70\mathrm{kg}]}\cup\mathrm{CAN2}_{\mathrm{OS}}^{[90\mathrm{kg}]}\cup\mathrm{CAN2}_{\mathrm{OS}}^{[97\mathrm{kg}]}$. The boldface figures indicate the best (smallest) result in each block, while the italicface figures show the result not corresponding the control objective of the CAN2 shown on the leftmost column.

CAN2 used	settling time t_{ST} [s]					overshoot x_{OS} [mm]				
for the	trained	test				trained	test			
controller	θ_i	mean	min	max	std	θ_i	mean	min	max	std
$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_1]}$	19.6	26.06	**19.6**	35.4	5.91	*98*	*97.3*	*55.0*	*142.0*	*28.5*
$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_2]}$	20.5	26.18	20.3	35.5	5.93	*59*	*108.3*	*51.0*	*172.0*	*43.9*
$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_3]}$	25.1	27.36	23.1	35.1	3.55	*44*	*75.6*	*34.0*	*162.0*	*41.6*
$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_4]}$	28.6	30.68	25.6	39.0	3.74	*94*	***25.9***	***0.0***	***103.0***	***36.1***
$\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_S]}$	—	**25.29**	22.2	**34.9**	**3.38**	—	*52.1*	*11.0*	*136.0*	*42.1*
$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_1]}$	*32.9*	*31.89*	*27.8*	*35.5*	*2.50*	0	11.4	**0.0**	66.0	20.1
$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_2]}$	*21.2*	***24.76***	***15.9***	***33.8***	*3.82*	0	49.4	**0.0**	264.0	72.6
$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_3]}$	*38.8*	*39.89*	*36.8*	*44.2*	***1.89***	0	3.2	**0.0**	28.0	7.4
$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_4]}$	*74.9*	*65.25*	*59.6*	*78.9*	*5.57*	0	6.8	**0.0**	38.0	11.5
$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_S]}$	—	*37.11*	*31.9*	*45.5*	*4.04*	—	6.6	**0.0**	35.0	11.9
$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_{S'}]}$	—	*41.61*	*36.6*	*44.6*	*2.08*	—	3.1	**0.0**	59.0	13.2
$\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_{S''}]}$	—	*38.55*	*35.3*	*43.0*	*2.11*	—	**0.0**	**0.0**	**0.0**	**0.0**

the overshoot obtained by the controller using $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_3]}$=$\mathrm{CAN2}_{\mathrm{OS}}^{[70\mathrm{kg}]}$, and it has the overshoot 3, 11, 18 and 28 [mm] for the plant with m = 85, 90, 95 and 100 [kg], respectively, and 0 [mm] for other test plants. Therefore, we next examined the controller using multiple $\mathrm{CAN2}_{\mathrm{OS}}^{[70\mathrm{kg}]}\cup\mathrm{CAN2}_{\mathrm{OS}}^{[97\mathrm{kg}]}$, and have an overshoot 59[mm] for the plant with m = 100 [kg] and 0[mm] for other test plants. Finally, we executed trial and error, and we have multiple $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_{S''}]}$=$\mathrm{CAN2}_{\mathrm{OS}}^{[70\mathrm{kg}]}\cup\mathrm{CAN2}_{\mathrm{OS}}^{[90\mathrm{kg}]}\cup\mathrm{CAN2}_{\mathrm{OS}}^{[97\mathrm{kg}]}$ which has no overshoot for all test plants as shown in Table 2.

4 Conclusion

We have examined the multiobjective robust controller using difference signals and multiple CAN2s by means of applying it to linear model plants. From the result of numerical experiments as well as theoretical analysis, the following properties are obtained. (1) The dimension of the input vector to select the associative matrix of the CAN2 to approximate the Jacobian matrix of the plant to be controlled does not have to be enlarged, which may reject the conjecture shown in [2,3] that the enlargement is necessary for the present method. (2) The present controller using fixed GPC parameters provides various control performances by means of involving errors of associative

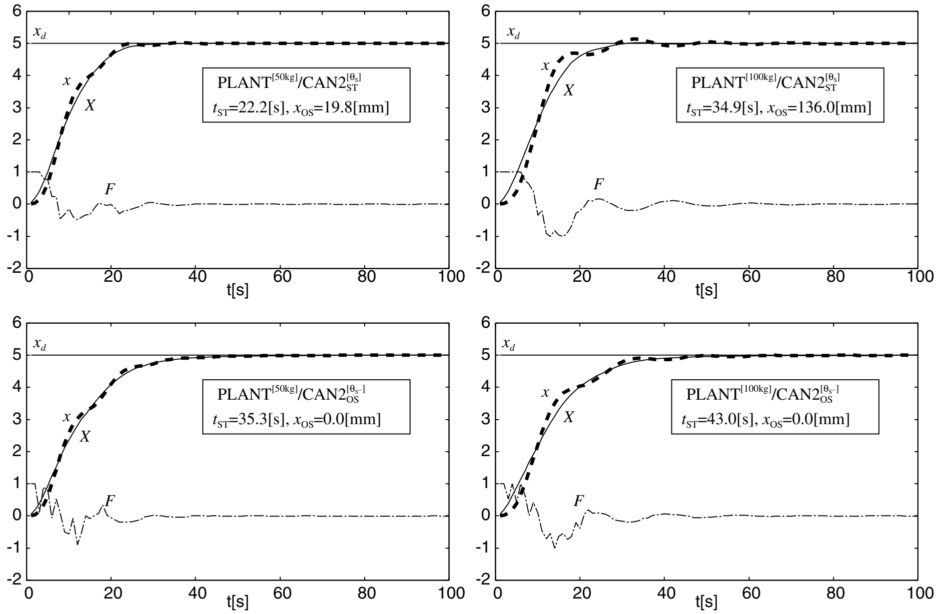

Fig. 2. Examples of time course of x[m], X[m] and F[10N]. Among the control of all test plants, the results of the smallest and biggest settling time by multiple $\mathrm{CAN2}_{\mathrm{ST}}^{[\theta_S]}$ are shown on the top left and right, respectively, and those of the smallest and biggest settling time without overshoot ($x_{\mathrm{OS}} = 0$[mm]) by $\mathrm{CAN2}_{\mathrm{OS}}^{[\theta_{S''}]}$ are shown on the bottom left and right, respectively.

matrices of CAN2s to learn Jacobian matrices, which enables the controller to be multi-objective robust controller by means of switching CAN2s. (3) Plant-parameter-specific overshoots can be reduced by the augmentation of CAN2s, which has also been shown possible for nonlinear plants [3].

Acknowledgement. This work was supported by JSPS KAKENHI Grant Number 24500276.

References

1. Kurogi, S., Yuno, H., Nishida, T., Huang, W.: Robust control of nonlinear system using difference signals and multiple competitive associative nets. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part III. LNCS, vol. 7064, pp. 9–17. Springer, Heidelberg (2011)
2. Huang, W., Kurogi, S., Nishida, T.: Robust controller for flexible specifications using difference signals and competitive associative nets. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) ICONIP 2012, Part V. LNCS, vol. 7667, pp. 50–58. Springer, Heidelberg (2012)
3. Huang, W., Kurogi, S., Nishida, T.: Performance Improvement Via Bagging Competitive Associative Nets for Multiobjective Robust Controller Using Difference Signals. In: Lee, M., Hirose, A., Hou, Z.-G., Kil, R.M. (eds.) ICONIP 2013. LNCS, vol. 8226, pp. 319–327. Springer, Heidelberg (2013)

4. Kurogi, S., Nishida, T., Sakamoto, T., Itoh, K., Mimata, M.: A simplified competitive associative net and a model-switching predictive controller for temperature control of chemical solutions. In: Proc. of ICONIP 2000, pp. 791–796 (2000)
5. Kurogi, S., Ren, S.: Competitive associative network for function approximation and control of plants. In: Proc. NOLTA 1997, pp. 775–778 (1997)
6. Kohonen, T.: Associative Memory. Springer (1977)
7. Ahalt, A.C., Krishnamurthy, A.K., Chen, P., Melton, D.E.: Competitive learning algorithms for vector quantization. Neural Networks 3, 277–290 (1990)
8. Deb, K.: Multi-objective Optimization Using Evolutionary Algorithms. John Wiley & Sons (2009)
9. Clarki, D.W., Mohtadi, C.: Properties of generalized predictive control. Automatica 25(6), 859–875 (1989)
10. Kurogi, S., Sawa, M., Ueno, T., Fuchikawa, Y.: A batch learning method for competitive associative net and its application to function approximation. In: Proc. of SCI 2004, vol. 5, pp. 24–28 (2004)