Proceeding of the 2004 American Control Conference
Boston, Massachusetts June 30 - July 2, 2004

Recursive Pointwise Design for Nonlinear Systems

Kazuo Tanaka, Hiroshi Ohtake and Hua. O. Wang

Abstract— **This paper presents a recursive pointwise design (RPD) method for a class of nonlinear systems represented by dx(t)/dt = f(x(t))+g(x(t))u(t). A main feature of the RPD method is to recursively design a stable controller by using pointwise information of a system. The design philosophy is that f(x(t)) and g(x(t)) can be approximated as constant vectors in very small local state spaces. Based on the design philosophy, we numerically determine constant control inputs in very small local state spaces by solving linear matrix inequalities (LMIs) derived in this paper. The designed controller switches to another constant control input when the states move to another local state space. Although the design philosophy is simple and natural, the controller does not always guarantee the stability of the original nonlinear system dx(t)/dt = f(x(t))+g(x(t))u(t). Therefore, this paper gives ideas of compensating the approximation caused by the design philosophy. After addressing outline of the pointwise design, we provide design conditions that exactly guarantee the stability of the original system. The controller design conditions requires to give the maximum and minimum values of elements in the functions f(x(t)) and g(x(t)) in each local state space. Therefore, we also present design conditions for unknown cases of the maximum and minimum values. Furthermore, we construct an effective design procedure using the pointwise design. A feature of the design procedure is to subdivide only infeasible regions and to solve LMIs again only for the subdivided infeasible regions. The recursive procedure saves effort to design a controller. A design example demonstrates the utility of the RPD method.**

I. INTRODUCTION

Stability analysis (e.g., [1]) for fuzzy control systems has been mainly discussed in the framework of fuzzy model-based control using Takagi-Sugeno fuzzy systems [2]. A main feature of the Takagi-Sugeno fuzzy systems is to have linear systems in consequent parts. Hence, it is represented by fuzzy blending of the linear systems. It is especially true from theoretical analysis and design points of view that it has been difficult to find clear differences between Takagi-Sugeno fuzzy model-based control and recently developed linear parameter varying (LPV) control. One of future research directions in fuzzy control is to seek a new approach that differs from Takagi-Sugeno fuzzy model-based control. This paper provides a new approach based on pointwise design that utilizes (fuzzy) pointwise information of a nonlinear system.

This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (C), 15560217, 2003.

K. Tanaka and H. Ohtake are with the Department of Mechanical Engineering and Intelligent Systems, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo 182-8585 Japan ktanaka@mce.uec.ac.jp & ohtake@rc.mce.uec.ac.jp

H. O. Wang is with the Department of Aerospace and Mechanical Engineering, Boston University, 10 Cummington Street, Boston, MA 02215 USA wangh@bu.edu

Sugeno and his co-authors [3], [4], [5], [6] have presented an analysis and design method using pointwise information of a nonlinear system. This work gives excellent stability results and brings us a very interesting and important fact to design a controller using pointwise information of a nonlinear system. In this work, pointwise information of a nonlinear system is described as a type of fuzzy models (the so-called TYPE II fuzzy model) that is different from Takagi-Sugeno fuzzy model. Their studies have not explicitly addressed model discrepancies caused via the fuzzy model approximation. That is, although the controllers designed in [3], [4], [5], [6] guarantee the stability of the type of fuzzy model, they do not always guarantee the stability of an original system. In addition, the conditions derived in [3], [4], [5], [6] become complicated (also may be hard to be solved) when a system has many state variables. A simple and effective design is preferred in real system applications.

A pointwise design proposed in this paper achieves a stable controller design for an original system with control input saturations (constraints). Control input constraints have not been discussed in [3], [4], [5], [6]. The pointwise design is simple and can be easily applied even to systems with a large number of states. In addition, to save design effort of a controller, an effective design procedure is constructed by recursively applying the pointwise design.

This paper presents a recursive pointwise design (RPD) method for a class of nonlinear systems $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t)) + g(\boldsymbol{x}(t))\boldsymbol{u}(t)$ by taking into account pointwise information. The design philosophy is that $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$ can be approximated as constant vectors in very small local state spaces. Based on the design philosophy, we numerically determine constant control inputs in very small local state spaces. To facilitate the determination, design conditions are represented in terms of linear matrix inequalities (LMIs). The designed controller switches to another constant control input when the states move to another local state space. Although the design philosophy is simple and natural, the controller does not always guarantee the stability of the original nonlinear system $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t)) + g(\boldsymbol{x}(t))\boldsymbol{u}(t)$. Therefore, this paper gives ideas of compensating the approximation caused by the design philosophy. After addressing outline of the pointwise design, we provide design conditions that exactly guarantee the stability of the original system. The controller design conditions requires to give the maximum and minimum values of elements in the functions $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$ in each local state space. Therefore, we also present design conditions for unknown cases of the maximum and minimum values. Furthermore, we construct an effective design procedure using the pointwise design.

A feature of the design procedure is to subdivide only infeasible regions and to solve LMIs again only for the subdivided infeasible regions. The recursive procedure saves effort to design a controller. A design example demonstrates the utility of the RPD method.

To lighten the notation, this paper employs the following particular notions: $R(s_1,\dots,s_n) = R_{\boldsymbol{s}}$, $\boldsymbol{x}(s_1,\dots,s_n) = \boldsymbol{x}_{\boldsymbol{s}}$, $\boldsymbol{u}(s_1,\dots,s_n) = \boldsymbol{u}_{\boldsymbol{s}}$, etc. Furthermore, $\boldsymbol{x}_{\boldsymbol{s}}(k_1,\dots,k_n) = \boldsymbol{x}_{(\boldsymbol{s},\boldsymbol{k})}$. We also shorten the notation $\forall s_1,\cdots,\forall s_n$ as $\forall \boldsymbol{s}$.

II. Outline of Pointwise Design

This section gives outline of the pointwise design. Consider the nonlinear system with n states and m inputs.

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t)) + g(\boldsymbol{x}(t))\boldsymbol{u}(t), \tag{1}$$

where $\boldsymbol{x}(t) \in R^n$ and $\boldsymbol{u}(t) \in R^m$ denote the state and input vectors, respectively. The control purpose is to stabilize the system at $\boldsymbol{x}(t) = 0$.

To discuss the pointwise design, we present a definition of region in the state space.

Definition 1: Each region $R(s_1,\dots,s_n)$ is defined as

$$\begin{aligned} R(s_1,\dots,s_n) &= R_{\boldsymbol{s}} \\ &= \{(x_1,\dots,x_n) |\ x_i^{min} + \Delta S_i\cdot(s_i-1) \\ &\qquad < x_i \le x_i^{min} + \Delta S_i \cdot s_i \quad i=1,\cdots,n\}, \end{aligned}$$

where $s_i = 1, \cdots, \phi_i$. ϕ_i is the number of partitioned guild for x_i axis. x_i^{min} and ΔS_i denote the minimum value and partitioned guild size of x_i, respectively. Note that the number of the region $R_{\boldsymbol{s}}$ becomes $\Pi_{i=1}^n \phi_i$.

For instance, in the case of $n = 2$, the regions $R(s_1,\dots,s_n)$ are divided as shown in Figure 1.

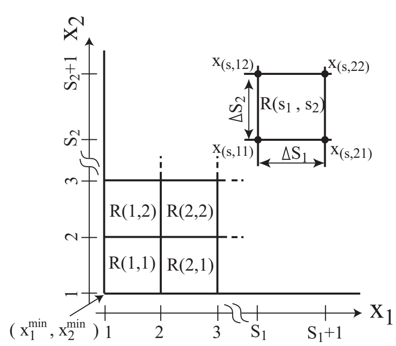

Fig. 1. Definition of regions (two dimenstional case).

To stabilize the system (1), we employ the following controller with control input constraints:

$$\boldsymbol{u}(t) = \boldsymbol{u}(s_1,\dots,s_n) = \boldsymbol{u}_{\boldsymbol{s}} \tag{2}$$
$$\boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}} < \boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}}^{max},\ \forall j,\ \forall \boldsymbol{s} \tag{3}$$
$$\boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}}^{min} < \boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}},\ \forall j,\ \forall \boldsymbol{s} \tag{4}$$

at $R_{\boldsymbol{s}}$, where $\boldsymbol{E}_j$ is a vector whose jth element is one and other elements are zero. The maximum and minimum values of control inputs $\boldsymbol{u}_{\boldsymbol{s}} = [u_{\boldsymbol{s}1}\ \cdots\ u_{\boldsymbol{s}m}]^T$ in each region $R_{\boldsymbol{s}}$ are denoted as $\boldsymbol{u}_{\boldsymbol{s}}^{max} = [u_{\boldsymbol{s}1}^{max}\ \cdots\ u_{\boldsymbol{s}m}^{max}]^T$ and $\boldsymbol{u}_{\boldsymbol{s}}^{min} = [u_{\boldsymbol{s}1}^{min}\ \cdots\ u_{\boldsymbol{s}m}^{min}]^T$. Equations (3) and (4) means that $u_{\boldsymbol{s}j}^{min} < u_{\boldsymbol{s}j} < u_{\boldsymbol{s}j}^{max}$ for $j = 1,2,\cdots,m$. That is, equations (3) and (4) represent control input constrains such as actuator saturations. The constraint is important in practical applications. An example of determining $\boldsymbol{u}_{\boldsymbol{s}}^{max}$ and $\boldsymbol{u}_{\boldsymbol{s}}^{min}$ will be addressed in Remark 3. Equation (2) means that the input $\boldsymbol{u}(t)$ is constant within each region. The controller switches to another constant value when the states move to another region.

To begin with, we consider a simpler case, where it is assumed that $f(\boldsymbol{x}(t)) \equiv f(s_1,\dots,s_n) = f_{\boldsymbol{s}}$ and $g(\boldsymbol{x}(t)) \equiv g(s_1,\dots,s_n) = g_{\boldsymbol{s}}$ at $R_{\boldsymbol{s}}$. This means that $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$ in each region are replaced with the constant vectors $f_{\boldsymbol{s}}$ and $g_{\boldsymbol{s}}$, respectively. Hence the original system (1) is approximated as

$$\dot{\boldsymbol{x}}(t) = f_{\boldsymbol{s}} + g_{\boldsymbol{s}}\boldsymbol{u}(t) \tag{5}$$

in each region.

This approximation is practically reasonable if the Δs_i's are small, i.e., if the regions are small. Theorem 1 gives a stability condition for the approximated system (5). Clearly, Theorem 1 is not sufficient for ensuring stability of the original system (1) due to the approximation.

We also assume that the system ($\boldsymbol{A}$, $\boldsymbol{B}$) linearized around $\boldsymbol{x}(t) = 0$ are stabilizable, i.e., there exist a positive definite matrix $\boldsymbol{P}$ and a feedback gain $\boldsymbol{H}$ such that $(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{H})^T\boldsymbol{P} + \boldsymbol{P}(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{H}) < \boldsymbol{0}$, where $\boldsymbol{A} = \frac{\partial f}{\partial \boldsymbol{x}}(\boldsymbol{x}(t))|_{\boldsymbol{x}=0}$ and $\boldsymbol{B} = \frac{\partial g}{\partial \boldsymbol{x}}(\boldsymbol{x}(t))|_{\boldsymbol{x}=0}$. This means that the feedback $\boldsymbol{u}(t) = -\boldsymbol{H}\boldsymbol{x}(t)$ stabilizes the system around $\boldsymbol{x}(t) = 0$. We give $\boldsymbol{P} > \boldsymbol{0}$ satisfying the above condition in advance and solve conditions in Theorem 1. By the procedure, conditions in Theorem 1 become LMIs with respect to $\boldsymbol{u}_{\boldsymbol{s}}$. In other words, $\boldsymbol{P}$ is not an LMI variable in Theorem 1. Only $\boldsymbol{u}_{\boldsymbol{s}}$ are LMI variables. The procedure often causes conservative results. However, it provides us the reduction of the number of LMIs to be solved simultaneously. It is an advantage of deigning a controller for a system with a large n. We will address it in Remark 2. In addition, to avoid the conservatism, we bring an interesting idea in design procedure. The idea is to subdivide only infeasible regions and to apply Theorem 1 again only to the subdivided infeasible regions. The recursive procedure also save design effort drastically when n is large. The details will be presented in Section IV

Theorem 1: The nonlinear system (5) is stabilized by the controller (2) if there are the control inputs $\boldsymbol{u}_{\boldsymbol{s}}$ satisfying the following conditions. The designed controller satisfies the input constraints (3) and (4).

$$\begin{aligned} (\boldsymbol{f}_{\boldsymbol{s}} + \boldsymbol{g}_{\boldsymbol{s}}\boldsymbol{u}_{\boldsymbol{s}})^T\boldsymbol{P}\boldsymbol{x}_{(\boldsymbol{s},\boldsymbol{k})} &< 0,\ \forall \boldsymbol{s},\ \forall \boldsymbol{k} & (6) \\ \boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}} &< \boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}}^{max},\ \forall j,\ \forall \boldsymbol{s} & (7) \\ \boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}}^{min} &< \boldsymbol{E}_j\boldsymbol{u}_{\boldsymbol{s}},\ \forall j,\ \forall \boldsymbol{s} & (8) \end{aligned}$$

$\boldsymbol{x}_{(\boldsymbol{s},\boldsymbol{k})}$ is constant defined as

$$\boldsymbol{x}_{(\boldsymbol{s},\boldsymbol{k})} = \boldsymbol{x}_{\boldsymbol{s}}(k_1,\cdots,k_n) = \begin{bmatrix} x_1^{min} + \Delta S_1 \xi_{k_1}(s_1) \\ \vdots \\ x_n^{min} + \Delta S_n \xi_{k_n}(s_n) \end{bmatrix},$$
$$\xi_1(s_i) = s_i - 1,\ \xi_2(s_i) = s_i,\ k_i = 1, 2.$$

(proof) The proof is omitted due to lack of space.

Remark 1: Theorem 1 is not sufficient for ensuring the stability of the original system (1) due to the approximation (5). We will strictly discuss the stability of the original system (1) in Section III.

Remark 2: The number of LMIs in Theorem 1 is $(2^n + 2m)\Pi_{i=1}^n \phi_i$. Note that we do not have to solve all the LMIs simultaneously. It is enough to simultaneously solve a set of $(2^n + 2m)$ LMIs (6)-(8) for each region. This means that the set of $(2^n + 2m)$ LMIs for each region can be solved independently since a common $\boldsymbol{P}$ is given in advance. This is an advantage for control system designs with a large number of n. Computational requirements are drastically reduced when n is large.

Remark 3: Any values of $\boldsymbol{u}_{\boldsymbol{s}}^{max}$ and $\boldsymbol{u}_{\boldsymbol{s}}^{min}$ such that $u_{\boldsymbol{s}j}^{min} < u_{\boldsymbol{s}j}^{max}$ for $j = 1, 2, \cdots, m$ can be separately selected in each region $R_{\boldsymbol{s}}$. The easiest selection is to employ the same values for all the regions, i.e., $u_{\boldsymbol{s}j}^{min} = u_j^{min}$ and $u_{\boldsymbol{s}j}^{max} = u_j^{max}$, according to actuators saturation, mechanical constraints, etc., where u_j^{min} and u_j^{max} are constants such that $u_j^{min} < u_j^{max}$ for $j = 1, 2, \cdots, m$. Alternatively, we can select them so as to be proportional to the distance from the origin (equilibrium point), i.e., $u_{\boldsymbol{s}j}^{max} = \alpha_j \sqrt{\boldsymbol{x}_{\boldsymbol{s}}^{cT} \boldsymbol{x}_{\boldsymbol{s}}^c}$ and $u_{\boldsymbol{s}j}^{min} = -\alpha_j \sqrt{\boldsymbol{x}_{\boldsymbol{s}}^{cT} \boldsymbol{x}_{\boldsymbol{s}}^c}$ for $j = 1, \cdots, m$, where $\boldsymbol{x}_{\boldsymbol{s}}^c$ is the central point in the region $R_{\boldsymbol{s}}$ and α_j is the constant.

Remark 4: In practice, we need to assign $\boldsymbol{u}_{\boldsymbol{s}} = 0$ at the regions including $\boldsymbol{x}(t) = 0$. However, if $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t))$ is unstable, the controller never stabilizes the system. Therefore, if $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t))$ is unstable, we introduce the feedback

$$\boldsymbol{u}(t) = -\boldsymbol{H}\boldsymbol{x}(t) + \bar{\boldsymbol{u}}(t). \tag{9}$$

Then, the system (1) is rewritten as

$$\dot{\boldsymbol{x}}(t) = \bar{f}(\boldsymbol{x}(t)) + g(\boldsymbol{x}(t))\bar{\boldsymbol{u}}(t), \tag{10}$$

where $\bar{f}(\boldsymbol{x}(t)) = f(\boldsymbol{x}(t)) - g(\boldsymbol{x}(t))\boldsymbol{H}\boldsymbol{x}(t)$. The feedback system (10) reduces to the original system (1) if $\bar{f}(\boldsymbol{x}(t))$ and $\bar{\boldsymbol{u}}(t)$ are replaced with $f(\boldsymbol{x}(t))$ and $\boldsymbol{u}(t)$, respectively. Hence, after introducing the feedback $\boldsymbol{u}(t) = -\boldsymbol{H}\boldsymbol{x}(t) + \bar{\boldsymbol{u}}(t)$, $\bar{\boldsymbol{u}}(t)$ in region $R_{\boldsymbol{s}}$ can be determined by applying Theorem 1 to (10).

Of course, selection of $\boldsymbol{P}$ (and also $\boldsymbol{H}$) directly relates to control performance. Therefore, for instance, we may determine $\boldsymbol{P}$ and $\boldsymbol{H}$ by solving the Riccati equation for the linearized system ($\boldsymbol{A}$, $\boldsymbol{B}$).

In the use of the feedback, $\boldsymbol{u}_{\boldsymbol{s}}^{max}$ and $\boldsymbol{u}_{\boldsymbol{s}}^{min}$ should be selected carefully. The constraint on $\bar{\boldsymbol{u}}(t)$ can be considered as that on $\boldsymbol{u}(t) + \boldsymbol{H}\boldsymbol{x}(t)$. Since the feedback gain $\boldsymbol{H}$ is given, it is possible to select $\boldsymbol{u}_{\boldsymbol{s}}^{max}$ and $\boldsymbol{u}_{\boldsymbol{s}}^{min}$ for $\bar{\boldsymbol{u}}(t)$.

III. Design Conditions based on Fuzzy Model Representation

As addressed in Remark 1, Theorem 1 is not sufficient for ensuring the stability of (1) due to the approximation. In Theorem 2, we guarantee the stability of the original system (1) instead of the approximated system (5) by introducing the following fuzzy model representation:

$$f(\boldsymbol{x}(t)) = \sum_{\eta_1=1}^{2} w_{f_1}(\boldsymbol{x}(t)) \cdots \sum_{\eta_n=1}^{2} w_{f_n}(\boldsymbol{x}(t)) \boldsymbol{f}_{(\boldsymbol{s},\boldsymbol{\eta})},$$
$$g(\boldsymbol{x}(t)) = \sum_{\theta_{11}=1}^{2} w_{g_{11}}(\boldsymbol{x}(t)) \cdots \sum_{\theta_{nm}=1}^{2} w_{g_{nm}}(\boldsymbol{x}(t)) \boldsymbol{g}_{(\boldsymbol{s},\boldsymbol{\theta})},$$

where

$$\boldsymbol{f}_{(\boldsymbol{s},\boldsymbol{\eta})} = \boldsymbol{f}_{\boldsymbol{s}}(\eta_1 \cdots \eta_n) = [\psi_{\boldsymbol{s}f1}(\eta_1)\ \cdots\ \psi_{\boldsymbol{s}fn}(\eta_n)]^T,$$
$$\boldsymbol{g}_{(\boldsymbol{s},\boldsymbol{\theta})} = \boldsymbol{g}_{\boldsymbol{s}} \begin{pmatrix} \theta_{11} & \cdots & \theta_{1m} \\ \vdots & \ddots & \vdots \\ \theta_{n1} & \cdots & \theta_{nm} \end{pmatrix} = \begin{bmatrix} \psi_{\boldsymbol{s}g11}(\theta_{11}) & \cdots & \psi_{\boldsymbol{s}g1m}(\theta_{1m}) \\ \vdots & \ddots & \vdots \\ \psi_{\boldsymbol{s}gn1}(\theta_{n1}) & \cdots & \psi_{\boldsymbol{s}gnm}(\theta_{nm}) \end{bmatrix},$$
$$\psi_{\boldsymbol{s}fi}(\eta_i) = \begin{cases} \max_{\boldsymbol{x}(t)} f_i(\boldsymbol{x}(t)),\ \boldsymbol{x}(t) \in R_{\boldsymbol{s}} & \text{if } \eta_{\rm i} = 1 \\ \min_{\boldsymbol{x}(t)} f_i(\boldsymbol{x}(t)),\ \boldsymbol{x}(t) \in R_{\boldsymbol{s}} & \text{if } \eta_{\rm i} = 2 \end{cases}$$
$$\psi_{\boldsymbol{s}gij}(\theta_{ij}) = \begin{cases} \max_{\boldsymbol{x}(t)} g_{ij}(\boldsymbol{x}(t)),\ \boldsymbol{x}(t) \in R_{\boldsymbol{s}} & \text{if } \theta_{\rm ij} = 1 \\ \min_{\boldsymbol{x}(t)} g_{ij}(\boldsymbol{x}(t)),\ \boldsymbol{x}(t) \in R_{\boldsymbol{s}} & \text{if } \theta_{\rm ij} = 2 \end{cases}$$

Theorem 2: The nonlinear system (1) is stabilized by the controller (2) if there are the control inputs $\boldsymbol{u}_{\boldsymbol{s}}$ satisfying the following conditions. The designed controller satisfies the input constraints (3) and (4).

$$(\boldsymbol{f}_{(\boldsymbol{s},\boldsymbol{\eta})} + \boldsymbol{g}_{(\boldsymbol{s},\boldsymbol{\theta})}\boldsymbol{u}_{\boldsymbol{s}})^T \boldsymbol{P} \boldsymbol{x}_{(\boldsymbol{s},\boldsymbol{k})} < 0,\ \forall\boldsymbol{\eta}, \forall\boldsymbol{\theta}, \forall\boldsymbol{s}, \forall\boldsymbol{k} \tag{11}$$
$$\boldsymbol{E}_j \boldsymbol{u}_{\boldsymbol{s}} < \boldsymbol{E}_j \boldsymbol{u}_{\boldsymbol{s}}^{max},\ \forall j,\ \forall\boldsymbol{s} \tag{12}$$
$$\boldsymbol{E}_j \boldsymbol{u}_{\boldsymbol{s}}^{min} < \boldsymbol{E}_j \boldsymbol{u}_{\boldsymbol{s}},\ \forall j,\ \forall\boldsymbol{s} \tag{13}$$

(proof) The proof is omitted due to lack of space.

Remark 5: The number of LMIs in Theorem 2 is $(2^{2n+mn} + 2m)\prod_{i=1}^n \phi_i$. The number of LMIs in Theorem 2 are larger than that in Theorem 1. As mentioned in Remark 2, we do not have to solve all the LMIs simultaneously. It is enough to simultaneously solve a set of $(2^{2n+mn} + 2m)$ LMIs for each region.

In most of cases, we can find $\boldsymbol{f}_{(\boldsymbol{s},\boldsymbol{\eta})}$ and $\boldsymbol{g}_{(\boldsymbol{s},\boldsymbol{\eta})}$ as in an example later. Even if we can not find these values, we can employ Theorem 3 instead of Theorem 2. The design strategy is to find the maximum value of β_v such that the stability conditions in Theorem 2 are satisfied.

Theorem 3: The nonlinear system (1) is stabilized by the controller (2) if there are the control inputs $\boldsymbol{u}_{\boldsymbol{s}}$ satisfying the following conditions. The designed controller satisfies the input constraints (3) and (4). ϵ_v's are weighting parameters satisfying $\sum_{v=1}^{n+mn} \epsilon_v = 1$.

$$\underset{\boldsymbol{u_s}}{\text{maximize}} \quad \sum_{v=1}^{n+mn} \epsilon_v \beta_v$$

subject to $\beta_v > 0$, (12), (13) and

$$\left((\boldsymbol{f_s} + \boldsymbol{v_{f\eta}}) + (\boldsymbol{g_s} + \boldsymbol{v_{g\theta}})\boldsymbol{u_s}\right)^T \boldsymbol{P}\boldsymbol{x}_{(\boldsymbol{s},\boldsymbol{k})} < 0, \quad \forall\boldsymbol{\eta}, \forall\boldsymbol{\theta}, \forall\boldsymbol{s}, \forall\boldsymbol{k}, \quad (14)$$

where

$$\boldsymbol{v_{f\eta}} = \left[(-1)^{\eta_1}\beta_1 \quad \cdots \quad (-1)^{\eta_n}\beta_n\right]^T,$$
$$\boldsymbol{v_{g\theta}} = \begin{bmatrix} (-1)^{\theta_{11}}\beta_{n+1} & \cdots & (-1)^{\theta_{1m}}\beta_{n+m} \\ \vdots & \ddots & \vdots \\ (-1)^{\theta_{n1}}\beta_{n+m(n-1)+1} & \cdots & (-1)^{\theta_{nm}}\beta_{n+nm} \end{bmatrix},$$

$\eta_i = 1, 2$ and $\theta_{ij} = 1, 2$.

(proof) The proof can be completed in the same procedure as in Theorem 2.

Remark 6: Theorem 3 says that the closed-loop stability is guaranteed if the (convex) vertices made by $\boldsymbol{f_s} + \boldsymbol{v_{f\eta}}$ and the (convex) vertices made by $\boldsymbol{g_s} + \boldsymbol{v_{g\theta}}$ include $\boldsymbol{f}_{(\boldsymbol{s},\boldsymbol{\eta})}$ and $\boldsymbol{g}_{(\boldsymbol{s},\boldsymbol{\theta})}$ in the region $R_{\boldsymbol{S}}$, respectively.

$\beta_1, \cdots, \beta_{n+mn}$ can be separately determined in Theorem 3. By assuming that $\beta_1 = \cdots = \beta_{n+mn} = \beta$, we can simplify Theorem 3 as follows:

Theorem 4: The nonlinear system (1) is stabilized by the controller (2) if there are the control inputs $\boldsymbol{u_S}$ satisfying the following conditions. The designed controller satisfies the input constraints (3) and (4).

$$\underset{\boldsymbol{u_s}}{\text{maximize}} \quad \beta$$

subject to $\beta > 0$, (12),(13) and

$$\left((\boldsymbol{f_s} + \beta\boldsymbol{h_{f\eta}}) + (\boldsymbol{g_s} + \beta\boldsymbol{h_{g\theta}})\boldsymbol{u_s}\right)^T \boldsymbol{P}\boldsymbol{x}_{(\boldsymbol{s},\boldsymbol{k})} < 0, \quad \forall\boldsymbol{\eta}, \forall\boldsymbol{\theta}, \forall\boldsymbol{s}, \forall\boldsymbol{k}, \quad (15)$$

where

$$\boldsymbol{h_{f\eta}} = \left[\, (-1)^{\eta_1} \quad \cdots \quad (-1)^{\eta_n} \,\right]^T,$$
$$\boldsymbol{h_{g\theta}} = \begin{bmatrix} (-1)^{\theta_{11}} & \cdots & (-1)^{\theta_{1m}} \\ \vdots & \ddots & \vdots \\ (-1)^{\theta_{n1}} & \cdots & (-1)^{\theta_{nm}} \end{bmatrix}. \quad (16)$$

Remark 7: In Theorems 2, 3 and 4, we can select $\boldsymbol{u_S^{max}}$ and $\boldsymbol{u_S^{min}}$ in the same way in Remark 3. In fact, we will utilize $u_{\boldsymbol{S}j}^{max} = \alpha_j\sqrt{\boldsymbol{x_S^{cT}}\boldsymbol{x_S^{c}}}$ and $u_{\boldsymbol{S}j}^{min} = -\alpha_j\sqrt{\boldsymbol{x_S^{cT}}\boldsymbol{x_S^{c}}}$ for $j = 1, \cdots, m$ in Section V.

IV. Design Procedure

We need to give a common $\boldsymbol{P} > \boldsymbol{0}$ in advance and solve the conditions in Theorem 1, 2, 3 or 4. As mentioned before, by the procedure, the conditions in Theorems 1, 2, 3 and 4 become LMIs with respect to $\boldsymbol{u_S}$. The procedure often causes conservative results. However, it provides us the reduction of the number of LMIs to be solved simultaneously. It is an advantage of deigning a controller for a system with a large n. In this section, to avoid the conservatism, we bring an interesting idea in design procedure. The idea is to subdivide only infeasible regions and to solve LMIs again only for the subdivided infeasible regions. The recursive procedure saves effort to design a controller. To perform the subdivision, we introduce a variable L that represents the level of subdivision. In the design procedure, L_{max} denotes the maximum number of L to quit the algorithm.

The design procedure consists of five steps.

[Step 1] Set L=1. Determine L_{max}. Determine x_i^{min}, ΔS_i and ϕ_i for $i = 1, \cdots, n$. Find a positive definite matrix $\boldsymbol{P}$. If necessary, introduce the feedback (9).

[Step 2] Solve a set of the LMIs in Theorems 1, 2, 3 or 4 and obtain the local control inputs $\boldsymbol{u_s}$

[Step 3] If the LMIs are feasible in all the regions, end. If not so, go to Step 4

[Step 4] If there exist regions such that the LMIs used in Step 2 are infeasible, subdivide the infeasible regions into smaller some regions. The way to subdivide the infeasible regions will be presented later. If $L \geq L_{max}$, then go to Step 5, else $L = L + 1$ and go to Step 2

[Step5] Find another $\boldsymbol{P}$ or increase L_{max} and go back to Step 2.

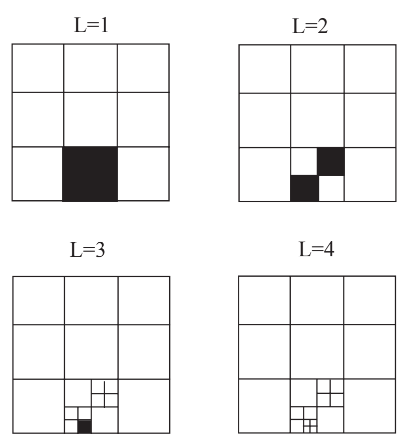

Fig. 2. An example of generating subdivided regions.

In Step 4, if there exist regions such that the LMIs are infeasible, the infeasible regions are subdivided into smaller some regions. The easiest way is to subdivide all the regions into smaller regions by decreasing the partitioned guild size ΔS_i and to solve the LMIs for all the regions. However, it is really inefficient. Hence, to save the design effort, we subdivide only the infeasible regions into smaller some regions and solve again the LMIs only for the subdivided infeasible regions with the same common $\boldsymbol{P}$. As mentioned in Remark 2, we do not have to solve all the LMIs simultaneously, i.e., the LMIs for each region can be solved independently since $\boldsymbol{P}$ is given in advance and the same common $\boldsymbol{P}$ is shared among all the regions. It can drastically save effort to design a controller when n is large. We subdivide equally the infeasible regions into 2^n smaller regions. Figure 2 shows an example of generating subdivided regions in the case of $n = 2$. The black regions show infeasible regions. If some of the generated smaller regions are infeasible regions, we apply the same procedure to the (generated and smaller) infeasible regions again. Thus, it is expected that all the regions become feasible regions by recursively subdividing infeasible regions. This fact will be found in Section V. The recursive procedure is much more effective than the easiest way mentioned above.

V. Design Example

Consider the following nonlinear system [7]:

$$\begin{cases} \dot{x}_1 = -2x_1 + ax_2 + \sin x_1 \\ \dot{x}_2 = -x_2 \cos x_1 + u \cos 2x_1, \end{cases} \tag{17}$$

where $a = 1$. From (17),

$$\begin{aligned} f_1(\boldsymbol{x}(t)) &= -2x_1 + ax_2 + \sin x_1, \\ g_1(\boldsymbol{x}(t)) &= 0 \\ f_2(\boldsymbol{x}(t)) &= -x_2 \cos x_1, \\ g_2(\boldsymbol{x}(t)) &= \cos(2x_1), \end{aligned}$$

where

$$\begin{aligned} &\boldsymbol{x} = [x_1 \;\; x_2]^T, \\ &f(\boldsymbol{x}(t)) = [f_1(\boldsymbol{x}(t)) \;\; f_2(\boldsymbol{x}(t))]^T, \\ &g(\boldsymbol{x}(t)) = [g_1(\boldsymbol{x}(t)) \;\; g_2(\boldsymbol{x}(t))]^T. \end{aligned}$$

For this system, the following controller can be designed using nonlinear control theory [7]. It can be seen that the controller works only for $-\frac{\pi}{4} < x_1 < \frac{\pi}{4}$.

$$\begin{aligned} u(t) = \frac{1}{\cos(2x_1)}&(-2ax_2 - 2\sin x_1 \\ &- \cos x_1 \sin x_1 + 2x_1 \cos x_1) \end{aligned} \tag{18}$$

Figure 3 plots the surfaces of $f_1(\boldsymbol{x}(t))$, $f_2(\boldsymbol{x}(t))$ and $g_2(\boldsymbol{x}(t))$. It is clear from Figure 3 that $\boldsymbol{f}_{(\boldsymbol{s},\boldsymbol{\eta})}$ and $\boldsymbol{g}_{(\boldsymbol{s},\boldsymbol{\eta})}$ can be found if we divide the regions with $x_1 = \frac{\pi}{4} \times k$ (k is an integer) and $x_2 = 0$. Therefore, Theorem 2 is employed to design a controller. In this example,

$$\begin{aligned} &x_1^{min} = -4\pi,\ \Delta S_1 = \tfrac{\pi}{4},\ \phi_1 = 32, \\ &x_2^{min} = -3,\ \Delta S_2 = 0.5 \text{ and } \phi_2 = 10. \end{aligned}$$

The input constraint, $u_{\boldsymbol{S}j}^{max} = \alpha_j \sqrt{\boldsymbol{x}_{\boldsymbol{S}}^{cT} \boldsymbol{x}_{\boldsymbol{S}}^{c}}$ and $u_{\boldsymbol{S}j}^{min} = -\alpha_j \sqrt{\boldsymbol{x}_{\boldsymbol{S}}^{cT} \boldsymbol{x}_{\boldsymbol{S}}^{c}}$ for $j = 1, \cdots, m$, is used as shown in Remark 3, where $\alpha_j = 10$.

Executing the design procedure, we can design a controller such that all the regions are feasible when $L = 4$. Figures 4-7 show generated subdivisions and infeasible regions for each L. We note that the feasible region (stable region) becomes larger by increasing L. (In $L = 4$, black regions are not infeasible region. Those are feasible regions. Due to very small subdivions (the boundary lines overlap each other), it seems to be black regions.)

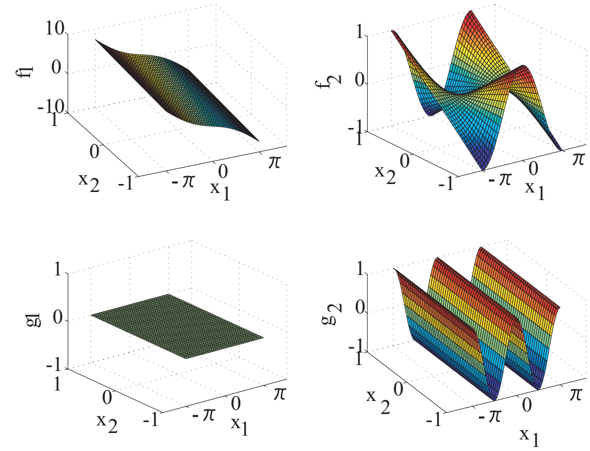

Fig. 3. Surfaces of f1(x(t)), f2(x(t)), g1(x(t)) and g2(x(t)).

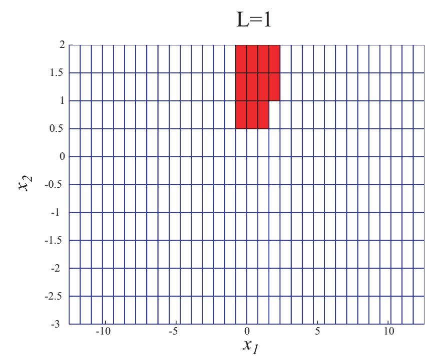

Fig. 4. Generated subdivisions and infeasible regions for $L = 1$.

Figures 8 and 9 illustrate the control inputs $u_{\boldsymbol{S}}$ and control trajectories on phase plane, respectively.

Remark 8: In this example, we give the result for $L = 4$ and the ranges ($-4\pi \le x_1 \le 4\pi$ and $-3 \le x_2 \le 2$). It is confirmed that, with $L = 10$, we can design a stable controller for $-20\pi \le x_1 \le 20\pi$ and $-5 \le x_2 \le 2.45$. Thus, it is generally possible to expand the stable region by increasing L.

Remark 9: In [5], a controller was designed for (17). However, their controller guarantees the stability of the type II fuzzy model (that approximates the dynamics of (17)) for the regions $-5/3\pi \le x_1 \le 5/3\pi$ and $-2 \le x_2 \le 2$. Therefore, their controller does not always guarantees the stability of the original system (17). As mentioned in Remark 8, our controller guarantees the stability of the original system (17) for much wider ranges (at least for the regions $-20\pi \le x_1 \le 20\pi$ and $-5 \le x_2 \le 2.45$). As emphasized above, a larger L achieves extension of the stable region. L_{max} can be selected by taking into account

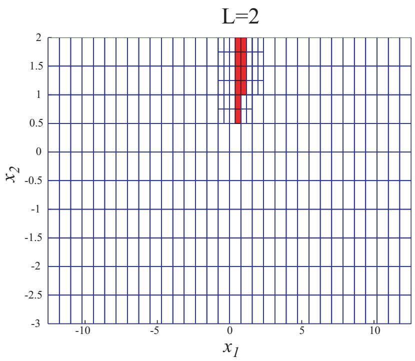

Fig. 5. Generated subdivisions and infeasible regions for $L = 2$.

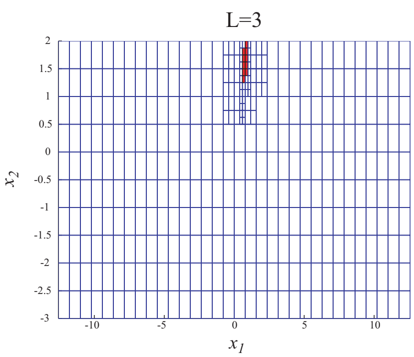

Fig. 6. Generated subdivisions and infeasible regions for $L = 3$.

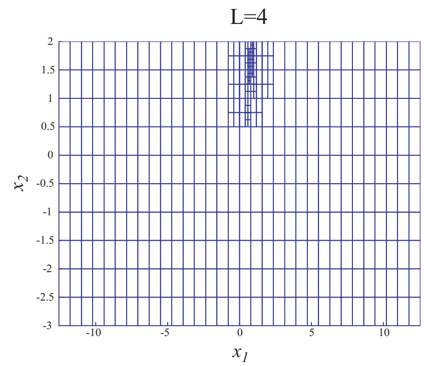

Fig. 7. Generated subdivisions and infeasible regions for $L = 4$.

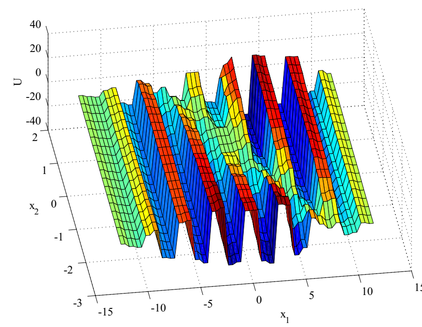

Fig. 8. Control input.

both extension of the stable regions and design effort.

VI. Conclusions

This paper has presented a recursive pointwise design (RPD) method for a class of nonlinear systems represented by $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t)) + g(\boldsymbol{x}(t))\boldsymbol{u}(t)$. The design philosophy is that $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$ can be approximated as constant vectors in very small local state spaces. This paper has given ideas of compensating the approximation caused by the design philosophy. We have provided design conditions that exactly guarantee the stability of the original system. The controller design conditions requires to give the maximum and minimum values of elements in the functions $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$ in each local state space. Therefore, we have presented design conditions for unknown cases of the maximum and minimum values. To save design effort of a controller, we have constructed an effective design procedure by recursively applying the pointwise design. A design example has demonstrated the utility of the RPD method.

We would like to thank Mr. M. Sakaguchi for his contribution for this research.

References

[1] K. Tanaka and H. O. Wang, "Fuzzy Control Systems Design and Analysis : A Linear Matrix Inequality Approach", John Wiley & Sons (2001).

[2] T. Takagi and M. Sugeno, "Fuzzy Identification of Systems and Its Applications to Modelling and Control", IEEE SMC-15, pp. 116-132, (1985)

[3] M. Sugeno, "On Stability of Fuzzy Systems Expressed by Fuzzy Rules with Singleton Consequents", IEEE TRANSACTIONS ON FUZZY SYSTEMS, VOL.7, NO.2, pp.201-224, 1999.

[4] T. Taniguchi and M. Sugeno, Stabilization of Nonlinear Systems based on Type II fuzzy models, Proceedings of the 2003 American Control Conference, Denver, June, pp.1920- 1925 (2003).

[5] T. Taniguchi and M. Sugeno, Stabilizing Control of Type II Fuzzy Systems, 18th Fuzzy System Symposium, pp.455-458, 2002, in Japanese.

[6] M. Sugeno, A Piecewise Parametric Approximation Model for Nonlinear Systems and Lyapunov Stability Conditions, 31th SICE Symposium on Control Theory, pp.129-134, October, 2002.

[7] Jean-Jacques E. Slotine and Weiping Li, "Applied Nonlinear Control" Prentice Hall, pp.213-216, 1991.

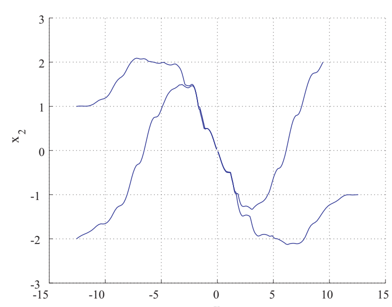

Fig. 9. Control trajectories on phase plane.