2005 WSEAS Int. Conf. on DYNAMICAL SYSTEMS and CONTROL, Venice, Italy, November 2-4, 2005 (pp38-43)

Exact Tensor Product Distributed Compensation Based Stabilization of the TORA System

PETER BARANYI
Computer and Automation Research Inst.
Cognitive Vision Systems Laboratory
Kende u. 13-17, H-1111 Budapest
HUNGARY

PETER L. VARKONYI
Computer and Automation Research Inst.
Cognitive Vision Systems Laboratory
Kende u. 13-17, H-1111 Budapest
HUNGARY

Abstract: This paper presents a case study of the TPDC (Tensor Product Distributed Compensation) based control design framework in the stabilization issue of the Translational Oscillator with an eccentric Rotational proof mass Actuator (TORA). First we execute TP model transformation on the linear parameter varying (LPV) model of the TORA to yield its exact TP model representation. As a second step, we substitute the components of the TP model into linear matrix inequalities to derive a controller that guaranties asymptotic stability. We show that the TPDC can be uniformly executed on a large class of LPV models and is capable of involving further control performances beyond stability.

Key–Words: Nonlinear control design, parallel distributed compensation, tensor product transformation, linear matrix inequalities

1 Introduction

The TP model form is a dynamic model representation whereupon Linear Matrix Inequality (LMI) based control design techniques (for instance PDC framework) can immediately be executed. It describes the Linear Parameter Varying (LPV) model by a convex combination of linear time invariant (LTI) models, where the convex combination is defined by the weighting functions of each parameter separately. The TP model representation is somewhat similar to the affine decomposition whereupon convex optimization or linear matrix inequality based control design techniques are applicable [1, 2, 3]. The TP model can always be given in the typical affine model form. An important difference between these model forms is that the convex hull of the given dynamic LPV model can readily be determined and analyzed in the TP model representation. Furthermore, the feasibility of the LMIs can be considerably relaxed in this representation via modifying the convex hull of the LPV model.

The TP model transformation is a recently proposed numerical method to transform LPV models into TP model form [4, 5]. It is capable of transforming different LPV model representations (such as physical model given by analytic equations, fuzzy, neural network, genetic algorithm based models) into TP model form in a uniform way. In this sense it replaces the analytical derivations and affine decompositions (that could be a very complex or even an unsolvable task), and automatically results in the TP model form. Execution of the TP model transformation takes a few minutes by a regular Personal Computer. The TP model transformation minimizes the number of the LTI components of the resulting TP model. Furthermore, the TP model transformation is capable of resulting different convex hulls of the given LPV model.

One can find a number of LMIs under the Parallel Distributed Compensation (PDC) framework which can immediately be applied to the TP model, according to various control design specifications. Therefore it is worth linking the TP model transformation and the PDC design framework.

In conclusion, the TPDC framework is applicable to various LPV model representations, and automatically executable without analytical derivations in reasonable time. Via solving the LMIs, selected according to different control specifications, in the TPDC framework the resulting controller guarantees the desired specifications.

In this paper we investigate the use of the TPDC framework in a control issue of the TORA system.

2 Notation

This section is devoted to introduce the notations being used in this paper: $\{a, b, \ldots\}$: scalar values. $\{\mathbf{a},\mathbf{b},\ldots\}$: vectors. $\{\mathbf{A},\mathbf{B},\ldots\}$: matrices. $\{\mathcal{A},\mathcal{B},\ldots\}$: tensors. $\mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$:vector space of real valued $(I_1\times I_2\times\cdots\times I_N)$-tensors. Subscript defines lower order: for example, an element of matrix $\mathbf{A}$ at row-column number i,j is symbolized as $(\mathbf{A})_{i,j}=a_{i,j}$. Systematically, the ith column vector of $\mathbf{A}$ is denoted as $\mathbf{a}_i$, i.e. $\mathbf{A}=\begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2 & \cdots\end{bmatrix}$. $(\cdot)_{i,j,n},\ldots$: are indices. $(\cdot)_{I,J,N},\ldots$: are index upper bounds: for example: $i=1..I$, $j=1..J$, $n=1..N$ or $i_n=1..I_n$. $\mathbf{A}_{(n)}$: n-mode matrix of tensor $\mathcal{A}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$. $\mathcal{A}\times_n\mathbf{U}$: n-mode matrix-tensor product. $\mathcal{A}\otimes_n\mathbf{U}_n$: multiple product as $\mathcal{A}\times_1\mathbf{U}_1\times_2\mathbf{U}_2\times_3..\times_N\mathbf{U}_N$. Detailed discussion of tensor notations and operations is given in [6].

3 Tensor Product Distributed Compensation (TPDC)

3.1 Linear Parameter-Varying state-space model

Consider the following parameter-varying state-space model:

$$\dot{\mathbf{x}}(t)=\mathbf{A}(\mathbf{p}(t))\mathbf{x}(t)+\mathbf{B}(\mathbf{p}(t))\mathbf{u}(t), \qquad (1)$$

$$\mathbf{y}(t)=\mathbf{C}(\mathbf{p}(t))\mathbf{x}(t)+\mathbf{D}(\mathbf{p}(t))\mathbf{u}(t),$$

with input $\mathbf{u}(t)$, output $\mathbf{y}(t)$ and state vector $\mathbf{x}(t)$. The system matrix

$$\mathbf{S}(\mathbf{p}(t))=\begin{pmatrix}\mathbf{A}(\mathbf{p}(t)) & \mathbf{B}(\mathbf{p}(t))\\ \mathbf{C}(\mathbf{p}(t)) & \mathbf{D}(\mathbf{p}(t))\end{pmatrix}\in\mathbb{R}^{O\times I} \qquad (2)$$

is a parameter-varying object, where $\mathbf{p}(t)\in\Omega$ is time varying N-dimensional parameter vector, and is an element of the closed hypercube $\Omega=[a_1,b_1]\times[a_2,b_2]\times\cdots\times[a_N,b_N]\subset\mathbb{R}^N$. $\mathbf{p}(t)$ can also include some elements of $\mathbf{x}(t)$.

3.2 Convex state-space TP model

$\mathbf{S}(\mathbf{p}(t))$ can be approximated for any parameter $\mathbf{p}(t)$ as the convex combination of LTI system matrices $\mathbf{S}_r$, $r=1,\ldots,R$. Matrices $\mathbf{S}_r$ are also called *vertex systems*. Therefore, one can define weighting functions $w_r(\mathbf{p}(t))\in[0,1]\subset\mathbb{R}$ such that matrix $\mathbf{S}(\mathbf{p}(t))$ can be expressed as convex combination of system matrices $\mathbf{S}_r$. The explicit form of the convex combination in terms of tensor product becomes:

$$\begin{pmatrix}\dot{\mathbf{x}}(t)\\ \mathbf{y}(t)\end{pmatrix}\approx\mathcal{S}\underset{n=1}{\overset{N}{\otimes}}\mathbf{w}_n(p_n(t))\begin{pmatrix}\mathbf{x}(t)\\ \mathbf{u}(t)\end{pmatrix} \qquad (3)$$

that is

$$\left\|\mathbf{S}(\mathbf{p}(t))-\mathcal{S}\underset{n=1}{\overset{N}{\otimes}}\mathbf{w}_n(p_n(t))\right\|\leq\varepsilon.$$

Here, ε symbolizes the approximation error, row vector $\mathbf{w}_n(p_n)\in\mathbb{R}^{I_n}$ $n=1,\ldots,N$ contains weighting functions $w_{n,i_n}(p_n)$. Function $w_{n,j}(p_n(t))\in[0,1]$ is the j-th univariate weighting function defined on the n-th dimension of Ω, and $p_n(t)$ is the n-th element of vector $\mathbf{p}(t)$. I_n $(n=1,\ldots,N)$ is the number of univariate weighting functions used in the n-th dimension of the parameter vector $\mathbf{p}(t)$. The $(N+2)$-dimensional coefficient tensor $\mathcal{S}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_N\times O\times I}$ is constructed from LTI vertex systems $\mathbf{S}_{i_1i_2\ldots i_N}\in\mathbb{R}^{O\times I}$. For further details we refer to [7, 4, 5]. The convex combination of the LTI vertex systems is ensured by the conditions:

Definition 1 *The TP model (3) is convex if:*

$$\forall n\in[1,N],i,p_n(t):w_{n,i}(p_n(t))\in[0,1]; \qquad (4)$$

$$\forall n\in[1,N],p_n(t):\sum_{i=1}^{I_n}w_{n,i}(p_n(t))=1. \qquad (5)$$

This simply means that $\mathbf{S}(\mathbf{p}(t))$ is within the convex hull of the LTI vertex systems $\mathbf{S}_{i_1i_2\ldots i_N}$ for any $\mathbf{p}(t)\in\Omega$.

$\mathbf{S}(\mathbf{p}(t))$ has a finite element TP model representation in many cases ($\varepsilon=0$ in (3)). However, exact finite element TP model representation does not exist in general ($\varepsilon>0$ in (3)), see Ref. [8]. In this case $\varepsilon\mapsto 0$, when the number of the LTI systems involved in the TP model goes to ∞. In the present control design, we will show that the dynamic model of the TORA system can be exactly represented by a finite TP model.

In order to have a direct link between the TP model and the typical form of affine models and LMI conditions, we define the following index transformation:

Definition 2 *(*Index transformation) *Let*

$$\mathbf{S}_r=\begin{pmatrix}\mathbf{A}_r & \mathbf{B}_r\\ \mathbf{C}_r & \mathbf{D}_r\end{pmatrix}=\mathbf{S}_{i_1,i_2,..,i_N},$$

where $r=ordering(i_1,i_2,..,i_N)$ ($r=1..R=\prod_n I_n$). The function "ordering" results in the linear index equivalent of an N dimensional array's index $i_1,i_2,..,i_N$, when the size of the array is $I_1\times I_2\times..\times I_N$. Let the weighting functions be defined according to the sequence of r:

$$w_r(\mathbf{p}(t))=\prod_n w_{n,i_n}(p_n(t)).$$

By the above index transformation one can write the TP model (3) in the typical form of:

$$\mathbf{S}(\mathbf{p}(t)) = \sum_{r=1}^{R} w_r(\mathbf{p}(t))\mathbf{S}_r.$$

Note that the LTI systems $\mathbf{S}_r$ and $\mathbf{S}_{i_1,i_2,..,i_N}$ are the same, only their indices are modified, therefore the hull defined by the LTI systems is the same in both forms.

3.3 TP model transformation

The TP model transformation starts with the given LPV model (1) and results in the TP model representation (3), where the trade-off between the number of LTI vertex systems and the ε is optimized [4]. The TP model transformation offers options to generate different types of the weighting functions $w(\cdot)$. For instance:

Definition 3 ***SN - Sum Normalisation*** *Vector $\mathbf{w}(p)$, containing weighting functions $w_i(p)$ is SN if the sum of the weighting functions is 1 for all $p \in \Omega$.*

Definition 4 ***NN - Non Negativeness*** *Vector $\mathbf{w}(p)$, containing weighting functions $w_i(p)$ is NN if the value of the weighting functions is not negative for all $p \in \Omega$.*

Definition 5 ***NO - Normality*** *Vector $\mathbf{w}(p)$, containing weighting functions $w_i(p)$ is NO if it is SN and NN type, and the maximum values of the weighting functions are one. We say $w_i(p)$ is close to NO if it is SN and NN type, and the maximum values of the weighting functions are close to one.*

Definition 6 ***RNO - Relaxed Normality*** *Vector $\mathbf{w}(p)$, containing weighting functions $w_i(p)$ is RNO if the maximum values of the weighting functions are the same.*

Definition 7 ***INO - Inverted Normality*** *Vector $\mathbf{w}(p)$, containing weighting functions $w_i(p)$ is INO if the minimum values of the weighting functions are zero.*

All the above definitions of the weighting functions determine different types of convex hulls of the given LPV model. The SN and NN types guarantee (4), namely, they guarantee the convex hull. The TP model transformation is capable of always resulting SN and NN type weighting functions. This means that one can focus on applying LMIs developed for convex affine models only, which considerably relaxes the further LMI design. The NO type determines a tight convex hull where as many of the LTI systems as possible are equal to the $\mathbf{S}(\mathbf{p})$ over some $\mathbf{p} \in \Omega$ and the rest of the LTI's are close to $\mathbf{S}(\mathbf{p}(t))$ (in the sense of L_2 norm). The SN, NN and RNO type guarantee that those LTI vertex systems which are not identical to $\mathbf{S}(\mathbf{p})$ are in the same distance from $\mathbf{S}(\mathbf{p}(t))$. INO guarantees that different subsets of the LTI's define $\mathbf{S}(\mathbf{p}(t))$ over different regions of $\mathbf{p} \in \Omega$.

These different types of convex hulls strongly effect the feasibility of the further LMI design.

3.4 Parallel Distributed Compensation (PDC)

The PDC framework is applicable to convex TP models [9]. It defines one feedback to each LTI vertex systems, namely, it starts with the LTI vertex system $\mathcal{S}$, and results in the vertex LTI feedbacks $\mathcal{F}$ of the controller. The control value is computed by the help of the same weighting functions as applied in the TP model (3):

$$\mathbf{u}(t) = -\left(\mathcal{F} \bigotimes_{n=1}^{N} \mathbf{w}_n(p_n(t))\right)\mathbf{x}(t). \tag{6}$$

The $\mathcal{F}$ is computed by the LMI based stability theorems selected according to the stability criteria and the desired control performance, see Ref. [9]. By means of the PDC framework one can derive observers as well [10].

4 LPV model of the TORA system

Consider the system shown in Figure 1. which represents the TORA [11, 12, 13, 9]. The nonlinear coupling between the rotational motion of the actuator and the translational motion of the oscillator provides the mechanism for control. Let $x_1(t)$ and $x_2(t)$ denote the translational position and velocity of the cart with $\dot{x_1}(t) = x_2(t)$. Let $x_3(t) = \theta(t)$ and $\dot{x_3}(t) = x_4(t)$ denote the angular position and velocity of the rotational proof mass. Then the system dynamics can be described by the equation:

$$\dot{\mathbf{x}}(t) = f(x_3(t), x_4(t))\mathbf{x}(t) + g(x_3(t))u(t),$$

where u is the torque applied to the eccentric mass, and

$$f(\mathbf{x}(t)) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{-1}{1-\varepsilon^2 cos^2(x_3(t))} & 0 & 0 & \frac{\varepsilon x_4(t) sin(x_3(t))}{1-\varepsilon^2 cos^2(x_3(t))} \\ 0 & 0 & 0 & 1 \\ \frac{\varepsilon cos(x_3(t))}{1-\varepsilon^2 cos^2(x_3(t))} & 0 & 0 & \frac{-\varepsilon x_4(t) sin(x_3(t))}{1-\varepsilon^2 cos^2(x_3(t))} \end{pmatrix},$$

$$g(\mathbf{x}(t)) = \begin{pmatrix} 0 \\ \frac{-\varepsilon cos(x_3(t))}{1-\varepsilon^2 cos^2(x_3(t))} \\ 0 \\ \frac{1}{1-\varepsilon^2 cos^2(x_3(t))} \end{pmatrix},$$

and let $\varepsilon = 0.05$. The linearization around the equilibrium point has a pair of nonzero imaginary eigenvalues and two zero eigenvalues. Hence the system at the origin is an example of a critical nonlinear system. The goal of the control is to asymptotically stabilize the system. In the present example we do not go further than stabilization and apply the simple LMIs guarantying asymptotic stabilization. Note that further control specification can be guarantied by other LMIs.

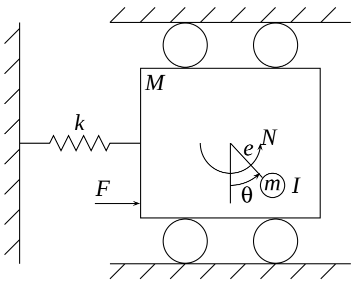

Figure 1: TORA

5 Applying TPDC

5.1 Executing TP model transformation

Let the transformation space be $\Omega = [-a, a] \times [a, a]$, where $a = 55/180\pi$ (note that these intervals can be arbitrarily defined). Let the density of the sampling grid be 101×101. The sampling results in $\mathbf{A}^s_{i,j}$ and $\mathbf{B}^s_{i,j}$, where $i, j = 1..101$. We construct matrix $\mathbf{S}^s_{i,j} = \begin{pmatrix} \mathbf{A}^s_{i,j} & \mathbf{B}^s_{i,j} \end{pmatrix}$. We construct tensor $\mathcal{S}^s \in \mathbb{R}^{101\times101\times2\times3}$ from $\mathbf{S}^s_{i,j}$. When we execute TP model transformation, we find that the rank of $\mathcal{S}^s$ on the first two dimensions are 4 and 2 respectively. This means that the TORA system can be exactly given as convex combination of $4 \times 2 = 8$ LTI vertex models. The LTI systems are:

$$\dot{\mathbf{x}}(t) = \qquad (7)$$

$$\sum_{i=1}^{4}\sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t)\right).$$

The weighting functions $w_{1,i}(x_3(t))$ and $w_{2,j}(x_4(t))$ are depicted on Figures 2 and 3. The resulting vertex systems are ($\mathbf{S}_{i,j} = \begin{pmatrix} \mathbf{A}_{i,j} & \mathbf{B}_{i,j} \end{pmatrix}$):

$$\mathbf{S}_{1,1} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0012 & 0 & 0 & 0.0280 & -0.0352 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0352 & 0 & 0 & -0.0280 & 1.0012 \end{pmatrix};$$

$$\mathbf{S}_{2,1} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0012 & 0 & 0 & -0.0278 & -0.0354 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0354 & 0 & 0 & 0.0278 & 1.0012 \end{pmatrix};$$

$$\mathbf{S}_{3,1} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0030 & 0 & 0 & 0.0003 & -0.0552 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0552 & 0 & 0 & -0.0003 & 1.0030 \end{pmatrix};$$

$$\mathbf{S}_{4,1} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0041 & 0 & 0 & -0.0006 & -0.0756 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0756 & 0 & 0 & 0.0006 & 1.0041 \end{pmatrix};$$

$$\mathbf{S}_{1,2} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0012 & 0 & 0 & -0.0280 & -0.0352 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0352 & 0 & 0 & 0.0280 & 1.0012 \end{pmatrix};$$

$$\mathbf{S}_{2,2} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0012 & 0 & 0 & 0.0278 & -0.0354 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0354 & 0 & 0 & -0.0278 & 1.0012 \end{pmatrix};$$

$$\mathbf{S}_{3,2} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0030 & 0 & 0 & -0.0003 & -0.0552 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0552 & 0 & 0 & 0.0003 & 1.0030 \end{pmatrix};$$

$$\mathbf{S}_{4,2} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -1.0041 & 0 & 0 & 0.0006 & -0.0756 \\ 0 & 0 & 0 & 1.0000 & 0 \\ 0.0756 & 0 & 0 & -0.0006 & 1.0041 \end{pmatrix};$$

5.2 PDC design to achieve asymptotic stability

Having the above resulting TP model (7) one can easily apply PDC control design framework. Before starting with solving LMIs let the result of the TP model transformation be given by linear indexing as (see Definition 2):

$$\dot{\mathbf{x}}(t) = \sum_{r=1}^{8} \omega_r(x_3(t), x_4(t)) \left(\mathbf{A}_r\mathbf{x}(t) + \mathbf{B}_r u(t)\right), \qquad (8)$$

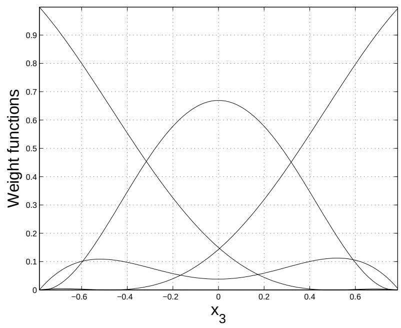

Figure 2: Weighting functions on dimension $x_3(t)$

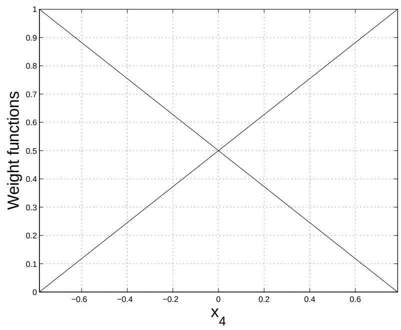

Figure 3: Weighting functions on dimension $x_4(t)$

where $\mathbf{A}_{r=2(i-1)+j} = \mathbf{A}_{i,j}$; $\mathbf{B}_{r=2(i-1)+j} = \mathbf{B}_{i,j}$ and $\omega_{r=2(i-1)+j}(x_3(t), x_4(t)) = w_{1,i}(x_3(t))w_{2,j}(x_4(t))$.

Than one can substitute these vertex models into the following LMIs:

Method 8 ***Asymptotic stability***

Find $\mathbf{X} > 0$ *and* $\mathbf{M}_r$ *satisfying equ.*

$$-\mathbf{X}\mathbf{A}_r^T - \mathbf{A}_r\mathbf{X} + \mathbf{M}_r^T\mathbf{B}_r^T + \mathbf{B}_r\mathbf{M}_r > 0 \quad (9)$$

for all r *and*

$$-\mathbf{X}\mathbf{A}_r^T - \mathbf{A}_r\mathbf{X} - \mathbf{X}\mathbf{A}_s^T - \mathbf{A}_s\mathbf{X} + \quad (10)$$

$$+\mathbf{M}_s^T\mathbf{B}_r^T + \mathbf{B}_r\mathbf{M}_s + \mathbf{M}_r^T\mathbf{B}_s^T + \mathbf{B}_s\mathbf{M}_r \geq 0.$$

for $r < s \leq R$*, except the pairs* (r, s) *such that* $w_r(\mathbf{p}(t))w_s(\mathbf{p}(t)) = 0, \forall \mathbf{p}(t)$.

Since the above conditions (9) and (10) are LMIs with respect to variables $\mathbf{X}$ *and* $\mathbf{M}_r$*, we can find a positive definite matrix* $\mathbf{X}$ *and matrix* $\mathbf{M}_r$ *or determine that no such matrices exist. This is a convex feasibility problem. Numerically, this problem can be solved very efficiently by means of the most powerful tools available in the mathematical programming literature e.g.* MATLAB-LMI *toolbox [2]. The feedback gains can be obtained form the solutions* $\mathbf{X}$ *and* $\mathbf{M}_r$ *as:*

$$\mathbf{F}_r = \mathbf{M}_r\mathbf{X}^{-1} \quad \text{and} \quad \mathbf{P} = \mathbf{X}^{-1}. \quad (11)$$

The above LMIs are feasible in the present case. By the help of $r = ordering(i_1, i_2, .., i_N)$ one can define feedbacks $\mathbf{F}_{i_1,i_2,..,i_N}$ from $\mathbf{F}_r$ obtained in (11) and store into tensor $\mathcal{F}$ of (6):

$$\mathbf{F}_{i,j} = \mathbf{F}_{r=2(i-1)+j}$$

so as:

$$\mathbf{F}_{1,1} = \begin{pmatrix} -11.9254 & 5.3868 & 8.2466 & 16.8868 \end{pmatrix};$$

$$\mathbf{F}_{2,1} = \begin{pmatrix} -7.9777 & 3.2653 & 5.3810 & 11.4298 \end{pmatrix};$$

$$\mathbf{F}_{3,1} = \begin{pmatrix} -12.0212 & 5.5961 & 8.4807 & 17.1318 \end{pmatrix};$$

$$\mathbf{F}_{4,1} = \begin{pmatrix} -12.9699 & 6.2143 & 9.2769 & 18.5103 \end{pmatrix};$$

$$\mathbf{F}_{1,2} = \begin{pmatrix} -7.8927 & 3.2167 & 5.3159 & 11.3096 \end{pmatrix};$$

$$\mathbf{F}_{2,2} = \begin{pmatrix} -11.9615 & 5.4097 & 8.2766 & 16.9395 \end{pmatrix};$$

$$\mathbf{F}_{3,2} = \begin{pmatrix} -11.9892 & 5.5788 & 8.4573 & 17.0875 \end{pmatrix};$$

$$\mathbf{F}_{4,2} = \begin{pmatrix} -13.0336 & 6.2490 & 9.3236 & 18.5984 \end{pmatrix};$$

The control value is generated as (6):

$$u(t) =$$

$$-\left(\sum_{i=1}^{4}\sum_{j=1}^{2} w_{1,i}(x_3(t))w_{2,j}(x_4(t))\mathbf{F}_{i,j}\right)\mathbf{x}(t).$$

5.3 Control results

Figure 4 shows the control result for the initial conditions $\mathbf{x}(t) = \begin{pmatrix} 0.1 & 0 & 60\pi/180 & 0 \end{pmatrix}$ and the control value is switched on at $t = 0$s. The system is asymptotically stable and beyond stability it is capable of tracking a sinusoidal (with amplitude $15\pi/180$ and frequency 0.01 rad/sec) command trajectory assigned to $x_3(t)$.

6 conclusion

We investigated the effectiveness of the Tensor Product Distributed Compensation (TPDC) based control design framework conducted trough the benchmark control problem of the TORA system. We observed the following advantages of the design framework: The TPDC framework is a numerical method, and can readily be implemented in MATLAB. It can uniformly be executed to a large class of control problems in a few minutes without analytical interaction. During the execution of the framework to the TORA, we obtained that the linear parameter varying state-space model of the TORA can be given exactly by finite element TP model with 8 LTI systems. We can also conclude that

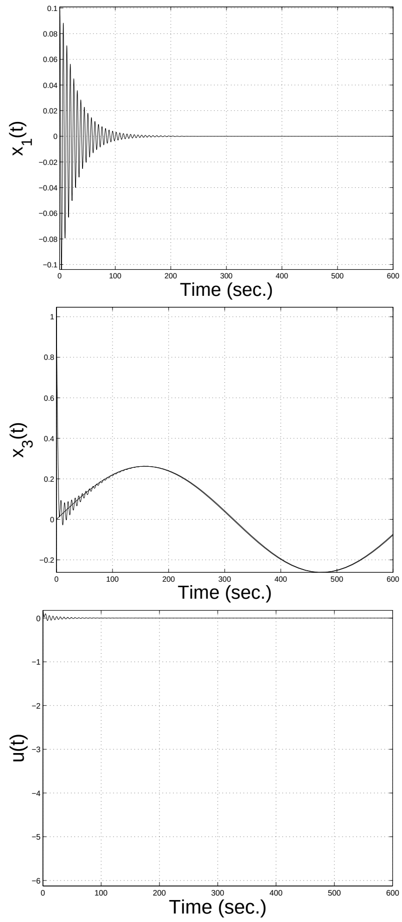

Figure 4: Simulation 2

the TPDC framework is capable of considering various different control specifications in terms of linear matrix inequalities. The whole numerical computation is tractable, the resulting TP model is exact in the present case, and the resulting controller guarantees the desired stability and control performance. The detailed example shows simulation results to validate the controller. We can observe that the control result shown by simulations satisfies the desired performance.

Acknowledgements: The research was supported by the National Found OTKA F 49838.

References:

[1] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, "Linear matrix inequalities in system and control theory," *Philadelphia PA:SIAM, ISBN 0-89871-334-X*, 1994.

[2] P. Gahinet, A. Nemirovski, A.J.Laub, and M.Chilali, LMI *Control Toolbox*, The MathWorks, Inc., 1995.

[3] C. W. Scherer and S. Weiland, *Linear Matrix Iequalities in Control*, DISC course lecture notes. DOWNLOAD: http://www.cs.ele.tue.nl/SWeiland/lmid.pdf, 2000.

[4] P. Baranyi, "TP model transformation as a way to LMI based controller design," *IEEE Transaction on Industrial Electronics*, vol. 51, no. 2, pp. 387–400, April 2004.

[5] P. Baranyi, D. Tikk, Y. Yam, and R. J. Patton, "From differential equations to PDC controller design via numerical transformation," *Computers in Industry, Elsevier Science*, vol. 51, pp. 281–297, 2003.

[6] L. D. Lathauwer, B. D. Moor, and J. Vandewalle, "A multi linear singular value decomposition," SIAM *Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[7] P. Baranyi, "Tensor product model based control of 2-D aeroelastic system," *Journal of Guidance, Control, and Dynamics (in Press)*.

[8] D. Tikk, P. Baranyi, R.J.Patton, and J. Tar, "Approximation capability of TP model forms," *Australian Journal of Intelligent Information Processing Systems*, vol. 8, no. 3, pp. 155–163, 2004.

[9] K. Tanaka and H. O. Wang, *Fuzzy Control Systems Design and Analysis - A Linear Matrix Inequality Approach*, Hohn Wiley and Sons, Inc., 2001.

[10] P. Baranyi, "Output-feedback design of 2-D aeroelastic system," *Journal of Guidance, Control, and Dynamics (in Press)*.

[11] R. T. Bupp, D. S. Bernstein, and V. T. Coppola, "A benchmark problem for nonlinear control design: Problem statement, experiment testbed and passive nonlinear compensation," *American Control Conference, Seattle*, pp. 4363–4367, 1995.

[12] Session on, "Benchmark problem for nonlinear control design," *American Control Conference, Seattle*, pp. 4337–4367, 1995.

[13] M. Jankovic, D. Fontaine, and P. Kokotovic, "Tora example: Cascade and passivity control design," *American Control Conference, Seattle*, pp. 4337–4351, 1995.