NPID-PCA User Guide

- A Simulation Toolkit of Nonlinear PID Control on Scilab/Scicos*

Version 1.0
Released date: September 2004

Bao-Gang Hu[1,2]

[1]LIAMA, Institute of Automation, Chinese Academy of Sciences
[2]Beijing Graduate School, Chinese Academy of Sciences
P.O. Box 2728, Beijing, 100080 China
hubg@nlpr.ia.ac.cn

Abstract

This documentation severs as a user guide for numerical simulations of nonlinear PID controllers based on SCILAB/SCICOS, an "open-source" scientific software developed by INRIA and ENPC of France. The simulation toolkit, entitled "**NPID-PCA**", is developed by the author, and is distributed freely to publics.

NPID-PCA is implemented following the so-called "**P**roportional **C**omponent **A**pproach (or **PCA**)". The significant feature of this approach is the selection of proportional components as the nonlinear functions to synthesize nonlinear PID controllers. Due to their simplest characteristics inherent by proportional actions, the proposed NPID-PCA provides a better means for the design and tuning of the nonlinearity of controllers.

In the structures of NPID-PCA, we adopt the configuration of three independent nonlinear proportional functions in connection to each gain loop. This scheme is important when users desire to adjust the equivalent nonlinear gains independently. NPID-PCA presents the most compatible structure with the conventional, or linear, PID technique. A linear PID controller is conveniently included as a special case for NPID-PCA.

A spline-based function, or Bézier curve, is used for forming nonlinear proportional component functions. For each nonlinear curve, at most four nonlinear parameters (or two control points in 2D) are used. The proposed controller can provide four types of the simplest nonlinear curves to approximate the nonlinear functions of the control output that are implicitly suggested by the process. The significant benefit is obtained by using Bézier curves for the nonlinear design. Users are able to control and visualize the nonlinear functions even without using a graphical means.

Specific attentions are made to the standardization in the implementation of NPID-PCA controllers. All parameters are set within given ranges without lost of generality. It will be helpful to reduce the optimization cost for the design and tuning of controllers. A saturation element is included in the structure of NPID-PCA. Therefore, an actual control force can be analyzed for the control design.

As the first version of NPID-PCA, this simulation toolkit only considers the specification of high performance in process control. We include three case studies with nine individual demos. All cases below are taken from the existing examples in the literature for comparisons:

CASE 1: Step response of first-order plant with or without time delay.

CASE 2: Step response of second-order plant with dead zone.

CASE 3: Step response of second-order plant with small damping.

The simulation results confirm the superior performances on the NPID-PCA controllers in comparing with linear PID technique. And, users can still improve the performance by either manually tuning or optimization-tool using.

To the author's knowledge, NPID-PCA seems to be the first free toolkit available in publics for the subject of nonlinear PID control. It is still in a very preliminary stage, but the author wishes that the present toolkit is useful for users to understand the theoretical fundamentals as well as the implementation and simulation of the NPID-PCA.

License

Contents

1. Introduction

The proportional-integral-derivative (or **PID**) control technique has been in existence for over eighty years [Aström and Hägglund, 1994]. Up to now, however, this technique is still keeping a dominant role in control engineering. We believe that PID controllers will always remain as a basic element of process control in future, even the other advanced control techniques may appear or mature. Both the past history of control engineering as well as PID control technique itself can support this belief. PID control takes the advantages of the "feedback" idea with most intuitive yet simplest means. As long as the feedback control remains, the principle of PID will be behind in working either explicitly or implicitly, or at least partially.

In control engineering, PID control technique has been considered as a matured technique in comparing with other control techniques. In fact, this technique is far away from maturity if PID controllers are used beyond as a linear tool. Even for the study of a linear PID, the improved schemes for adaptation or self-tuning are still in the list of hot topics. In recent years, PID control has received more attentions from the control community [IFAC, 2001, IEE, 2002]. One driving force is from fully understanding and utilizing PID for advanced applications.

In this work, we focus on a study of PID control as a nonlinear tool. Significant investigations have been reported in literature [Rugh, 1987; Jutan, 1989; Han, 1994; Xu, *et al*, 1995; Shahruz and Schwartz, 1994, 1997; Kelly and Carelli, 1996; Seraji, 1998; Hu, *et al*, 1998; Bucklaew and Liu, 1999; Cehn, *et al*, 1999; Liu and Daley, 2000; Armstrong, *et al*, 2001, Tan, *et al*, 2001; Gao, 2002; Ortega, *et al*, 2002; Feng, *et al*, 2002; Chang, *et al*, 2002; Huang, *et al*, 2002; Su, *et al*, 2004]. However, there is no well-accepted synthesizing approach for nonlinear PID controllers. On the other hand, to the author's knowledge, there is no nonlinear PID control toolkit available in publics. Therefore, in this work, we develop a simulation toolkit of nonlinear PID controllers based on Scilab/Scicos. Since we adopt the "**P**roportional **C**omponent **A**pproach (or **PCA**)" [Hu, *et al*, 1998] for synthesizing the nonlinearity of PID, this toolkit is called "**NPID-PCA**".

We wish NPID-PCA can be used for studies by both engineers and the educators. Several demos are given for users to understand and test the proposed nonlinear PID controllers. We choose Scilab/Scicos as a simulation platform since this software is open source as well as its sufficient functions for numerical studies.

The objective of this NPID-PCA toolkit is to stimulate a study in establishing a generalized nonlinear PID controller from both theoretical and practical viewpoints. One of the most challenging tasks for this study, we believe, is to preserve the distinguished features of the conventional PID control technique to the nonlinear PID controllers, such as,

- Generality: it is a general tool applicable for various processes.
- Intuitivity: it works in principles compatible with human intuitions.
- Model free: it dose not require model identifications for parameter tuning.
- Simplicity: it is simple with a few number of tuning parameters.

Therefore, the features above become the principles, or design guidelines, for the NPID-PCA. This documentation does not only propose a new approach but also stress on the principles in each aspect of the NPID-PCA controllers.

2. Proposed Structure of NPID-PCA

There exist variety forms for linear PID controllers [Aström and Hägglund, 1995; Datta, *et al*, 2000]. In this work, we only study the most conventional forms of cascade feedback for linear PID controllers:

$$u = K_P e + K_I \int edt + K_D \frac{de}{dt} = u_P + u_I + u_D, \tag{1}$$

where e is the error signal, K_P, K_I and K_D are proportional, integral and derivative gains, respectively. These gains are constant for a linear PID controller. u is the overall control force which is a summation of three components as u_P, u_I and u_D.

For a nonlinear PID control, it can usually be found by one of two forms, that is, "*Direction-force*" type and "*Gain-scheduling*" type [Mann, *et al*. 1999]. The two types can be expressed in the following forms, respectively:

Direct-action type:

$$u^{(DA)} = u_P(\mathbf{x}, \boldsymbol{\theta}_P) + u_I(\mathbf{x}, \boldsymbol{\theta}_I) + u_D(\mathbf{x}, \boldsymbol{\theta}_D). \tag{2a}$$

Gain-scheduling type:

$$u^{(GS)} = K_P(\mathbf{x}, \boldsymbol{\theta}_P)e + K_I(\mathbf{x}, \boldsymbol{\theta}_I)\int edt + K_D(\mathbf{x}, \boldsymbol{\theta}_D)\frac{de}{dt}. \tag{2b}$$

where $\mathbf{x}$ and θ are the variable and parameter sets, respectively. One can see that two types are different in the selection of items as the nonlinear functions. While the "*Direction-action*" type synthesizes the three components for the nonlinear functions directly, the "*Gain-scheduling*" type designs the nonlinear gains individually.

In this work, we apply the "*Proportional Component Approach*" [Hu, *et al*, 1998] for the design of nonlinear PID controllers, and call them as NPID-PCA. The main feature of NPID-PCA is the selection of only proportional components, or actions, as the nonlinear functions for synthesizing nonlinear PID controllers. "*This selection is of great significance in nonlinear control design since this action may provide maximum intrinsic simplicity of nonlinear functions than other control actions and tuning gains* [Hu, *et al*, 1998]". For example, one can derive the following heuristic properties for the function of the proportional component:

- $u_P(e,\theta_P)$ is a continuous and monotonic function in respect to the error signal e.
- When e=0, one has $u_P(e,\theta_P)=0$.
- When e=max(e), one has $u_P(e,\theta_P)=\max(u_P)$.

It is understandable that all other two components and three gains do not share the similar properties. And, they do not hold other well-accepted property for simplifying the nonlinear design. In the later simulation examples, we can understand this further by examining the nonlinear gains.

Fig. 1 shows a typical structure of NPID-PCA, in which we adopt the configuration of using three independent proportional actions connected to each gain loop [Mann, *et al*, 1999]. The general expression for NPID-PCA is expressed by:

$$\hat{u} = \hat{K}_P \hat{u}_{P1}(\hat{e}, \boldsymbol{\theta}_{P1}) + \hat{K}_I \int \hat{u}_{P2}(\hat{e}, \boldsymbol{\theta}_{P2})dt + \hat{K}_D \frac{d\hat{u}_{P3}(\hat{e}, \boldsymbol{\theta}_{P3})}{dt}, \tag{3}$$

where $\hat{e}$ and $\hat{u}_{Pi}$ ($\in$[-1,1]) are normalized error signal and normalized proportional component, respectively; and $\hat{K}_P$, $\hat{K}_I$, $\hat{K}_D$ ($\in$[0,1]) are the normalized gains given as constants. We call $\hat{u}$

to be the scaled output. In Fig. 1, S_e and S_u are scaling factor and denormalized factor, respectively. We usually set S_e to be a known constant by the calculation from:

$$S_e = \frac{1}{\max(|e|)}, \tag{4}$$

but consider S_u to be a linear tuning parameter in a given range:

$$S_u = \left[0, \max\left(\left|u^-\right|, \left|u^+\right|\right)\right], \tag{5}$$

where u^- ,u^+ are the lower bound and upper bound of the saturation element, respectively. We include a saturation element in the structure of NPID-PCA controllers (Fig. 1), since most actuators do have limitations to produce the output forces. This strategy will be helpful for analyzing the actual control forces to the processes.

Generally, there have total four linear parameters in the design of NPID-PCA controllers, that is, $\hat{K}_P$, $\hat{K}_I$, $\hat{K}_D$ and S_u. It seems that an additional cost will be added if introducing one more parameter into the conventional linear PID controllers. In fact, the significant benefit is obtained by this scheme. For an optimization design of linear parameters, the conventional approach will search within a half space for each of three gains, but the present approach will search within only a limited space for each of four parameters. Hence, the computational cost can be greatly reduced due to the reduction of the searching space. Moreover, this scheme will also be important for realizing a standard procedure for a controller design. For example, a genetic algorithm can be used directly.

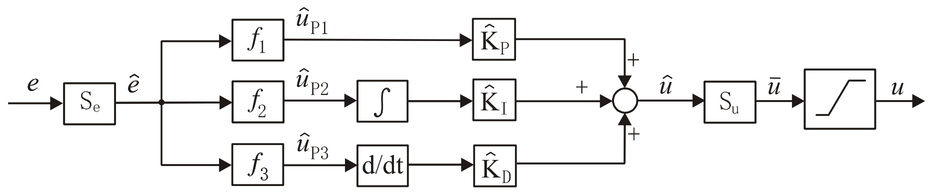

Fig. 1. Typical structure of the proposed NPID-PCA controllers.

3. Synthesis of Nonlinear Proportional Components

Up to now, there exist various approaches to synthesis the nonlinear PID controllers. In apart from the empirical approaches for some given nonlinear functions [Ruth, 1987; Han, 1994; Bucklaew and Liu, 1999; Armstrong, *et al*, 2001], the studies have been found in using fuzzy logic (Zhao, *et al*. 1993; Malki, *et al*, 1994; Li and Gatland, 1996) and neural networks (Ruano, *et al*, 1992; Scott, *et al*, 1992). Most researchers selected approaches based on their experiences and preferences.

Although there is no cure-all solution or approach for control engineering applications, there does exist a need of a general tool like a PID technique from process industry. This tool usually requires handling every plant with different dynamic characteristics. This is also true when PID controllers evolve into a nonlinear technique.

Ideally, we may hope that a NPID controller can work like a "*universal approximators*" (say, fuzzy systems [Wang and Mendel, 1992; Ying, 1994] or neural networks [Hornik, *et al*., 1989; Park and Sandberg, 1991]) to be suitable for all kinds of processes. This means that the controller can approximate any nonlinear control laws or functions. These functions are varied with processes as well as the specific performance criteria. In most cases, however, the knowledge of explicit expressions of nonlinear functions is usually unknown or does not exist to control engineers. In the design of NPID technique, one will face the tasks as followings [Hu, *et al*, 1998]:

"*1). to guess the general (or process-independent) properties of the desired nonlinear functions,*

2). to generate a set of closed-form nonlinear functions compatible with the properties, and

3). to related the nonlinear tuning parameters quantitatively to their associated versatility and flexibility to cover various nonlinear functions".

The task statements above indicate that NPID controllers should not be designed as a conventional concept of "*universal approximators*". If NPID controllers aim to be a "*model-free*" tool, they should be able to produce a group of nonlinear functions that are derived from "*process- or problem-independent*" cases. Only in this way, this type of NPID will preserve the most inherent power to deal with the cases where little or no knowledge is available to the processes.

In our previous work [Hu, *et al*, 1998], we demonstrated the detailed procedures for fulfilling the tasks above. First, we derived the fifteen items of heuristic properties for the nonlinear proportional functions by institutions based on the general applications for nonlinear design. Second, we selected spline-based functions to generate the closed form functions that are compatible with those heuristic properties. In this task, we considered three evaluation criteria for the selection of the nonlinear synthesizing approach. Third, we used so-called "*nonlinearity variation index*" as a "*process-independent*" measure, not the approximation accuracy of the specific function, to evaluate the NPID controllers.

The three criteria proposed in the selection of nonlinear synthesizing approach are "*transparency*", "*versatility*", and "*simplicity*", respectively. Examining with these criteria, we can understand that the methodologies like fuzzy systems and neural networks do not present the best solution for forming a low-level control element of nonlinear PID controllers.

In this work, we will adopt the Bezier functions in [Hu, *et al*, 1998] as the nonlinear synthesizing approach. The readers can refer to the original paper for the theoretical parts of the approach. For a proper understanding and applying the approach, the more detailed procedures of nonlinear design are given below.

Step 1. Select the number for nonlinear proportional functions, $\hat{u}_{Pi}$. According to the "*Simple-first*" strategy, only one function, $\hat{u}_P$, is used, which will be connected to each gain loops.

Step 2. Select the number for nonlinear parameters, n_{nl}. In the proposed NPID-PCA controllers, there exist three cases, that is, n_{nl}=1, 2, 4. According to the "*Simple-first*" strategy, only one parameter, n_{nl}=1, is used.

Step 3. According to the number of nonlinear parameters, set the nonlinear parameter set:

$n_{nl}=1$: $\theta_P=\{P_{x1}\}$

$n_{nl}=2$: $\theta_P=\{P_{x1},P_{y1}\}$

$n_{nl}=4$: $\theta_P=\{P_{x1},P_{y1},P_{x2},P_{y2}\}$

where all terms in the nonlinear parameter set are the coordinates of control point(s), which are defined by $\theta_P\in[0,1]$.

Step 4. Establish the nonlinear function $\hat{u}_P=f(\hat{e}, \theta_P)$ for $\hat{u}_P$ and $\hat{e}$ within [0,1] first by the following parametric equations:

$$\begin{aligned}\hat{e} &= 3s(1-s)^2 P_{x1} + 3s^2(1-s)P_{x2} + s^3 \\ \hat{u}_{Pi} &= 3s(1-s)^2 P_{y1} + 3s^2(1-s)P_{y2} + s^3,\end{aligned} \tag{6}$$

where $s(\in[0,1])$ is the functional parameter. Eq. (6) corresponds to the case when n_{nl}=4. If using a smaller number of nonlinear parameter set, NPID-PCA imposes the following constraints for the relation:

$n_{nl}=1$: $P_{x1}= P_{x2}=1-P_{y1}=1-P_{y2}=c$, $c\in[0,1]$

$n_{nl}=2$: $P_{x1}= P_{x2}=c_1$,; $P_{y1}=P_{y2}=c_2$, $c_1,c_2\in[0,1]$

Step 5. Calculate the eq. (6). For a given $\hat{e}$, one can find a unique value of s within [0,1] form the first equation of (6), and obtain $\hat{u}_P$ by substituting the current value s into the second equation.

Step 6. Construct whole $\hat{u}_P=f(\hat{e}, \theta_P)$ function by adding the relations in the range of [-1,0]. For simplicity, say, one can apply an anti-symmetric property to form the whole function.

The nonlinear proportional function can be visualized whenever the control point(s) is known. The most advantage for using the present synthesizing approach is that users can immediately know the curve types from the location of the control point(s).

In the present NPID-PCA, four types of nonlinear curves can be formed as shown in Fig. 2. Fig. 2(a) shows a "C" curve with using a single nonlinear parameter (Note the control point P_1 is located only on the diagonal line indicated). Fig. 2(b) shows an "Inverse-C" curve with using two nonlinear parameters. And Figs. 2(c) and (d) show the "S" and "Inverse-S" curves by using four tuning parameters. We consider those simplest curves are essential for a generic tool of nonlinear PID controllers.

Although this work is about the nonlinear PID controller design, we propose users to start with a linear PID controller first in their initial design or for their tuning process. For example, one can realize a linear PID by imposing $P_x=P_y$ on the controller points, which will arrive at a perfect linear function, $\hat{u}_P=\hat{e}$. A nonlinear tuning process is usually made following the linear one, if a manual operation is used. In practice, several iterations may be engaged for achieving an optimal tuning.

For a given relation of $\hat{u}_P = f(\hat{e}, \theta_P)$, one can obtain explicitly the equivalent nonlinear proportional gain and derivative gain [Hu, *et al*, 1998], respectively:

$$\begin{aligned}(K_P)_{eq} &= \frac{K_P}{s_u \hat{K}_P} = \frac{\hat{u}_P(\hat{e}, \theta_{P2})}{\hat{e}} \\ &= \frac{3(1-s)^2 P_{y1} + 3s(1-s)P_{y2} + s^2}{3(1-s)^2 P_{x1} + 3s(1-s)P_{x2} + s^2},\end{aligned} \tag{7}$$

$$\begin{aligned}(K_D)_{eq} &= \frac{K_D}{s_u \hat{K}_D} = \frac{\partial \hat{u}_P(\hat{e}, \theta_{P2})}{\partial \hat{e}} \\ &= \frac{(1-4s+3s^2)P_{y1} + (2s-3s^2)P_{y2} + s^2}{(1-4s+3s^2)P_{x1} + (2s-3s^2)P_{x2} + s^2},\end{aligned} \tag{8}$$

However, one can find a closed-form solution for the equivalent nonlinear integral gain $(K_I)_{eq}$ only when the error signal $\hat{e}(t)$ is known. Eqs. (7) and (8) remove the linear parameter effects in order to stress only on the nonlinearities of the gains.

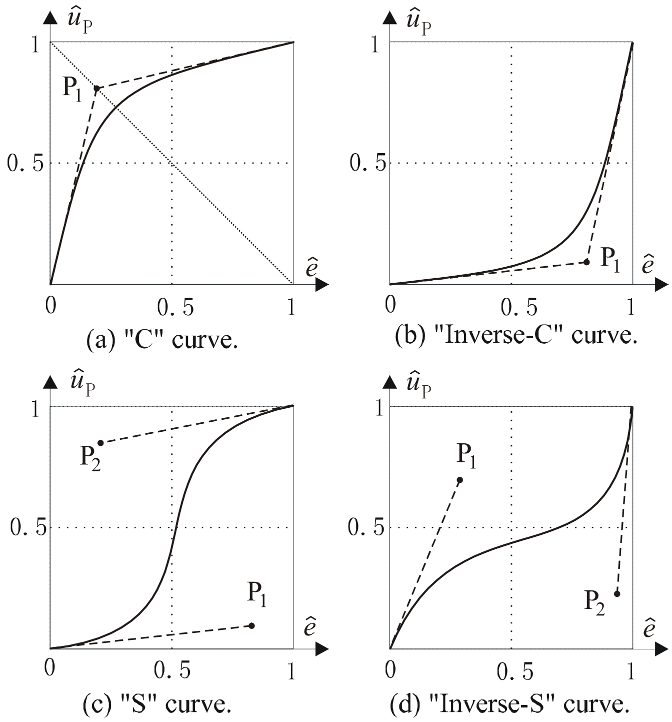

Fig. 2. Four types of simple curves.

4. Case Studies

In this simulation toolkit, we will present several case studies using NPID-PCA controllers on the existing examples from the literature. All studies can be repeated from the giving files in this toolkit, but only in next chapter we will introduce the basic procedures in using the toolkit.

As a preliminary study in this work, we only focus on a single-input-single-output feedback control with the specifications of response performance. That means we would aim at realizing high-performance controllers. Further investigations can be extended conveniently based on the present setup into the other aspects, such as the disturbance response, stability, robustness, etc. Although there exist various advanced configurations on the PID controllers, we only consider the conventional cascade one for the purpose of fast comparisons with other reported works.

Three cases will be studied below with nine individual demos. They will give users a rough idea how an NPID-PCA Controller works, and what the performance it can exhibit. Except for the given parameters from other sources, all other parameters are obtained by a well manually tuning means.

<u>CASE 1: Step response of first-order plant with or without time delay.</u>

This case study is taken from [Hu, et al, 1999], where fuzzy PID controllers were used. From this case we can see that the present NPID-PCA controllers are able to reach the similar high performances as fuzzy PID controllers. However, the present controllers will have simpler expressions of nonlinear functions than those of fuzzy PID controllers.

The plant in this case is a first-order process with a saturation ranges:

$$G = \frac{1}{s+1} e^{-t_d s}, \qquad u^- = 0, \qquad u^+ = 10$$

Two situations are studied for the time delay: t_d=0, and t_d=0.2. In this case study, only PI-type controllers are applied. Figs. 3-4 show the step responses for the closed-loop systems for two situations. Both PI-A and PI-B are linear controllers, and NPI-A and NPI-B are nonlinear controllers. The parameter values are listed in Table 1 for four controllers. In the nonlinear PI controllers, only a single nonlinear function, $\hat{u}_P$, is used. NPI-A and NPI-B controllers employ a single and two nonlinear parameters, respectively.

Table 1. Parameters of linear and nonlinear controllers for step responses of the first-order process with and without time delay.

Time Delay	t_d = 0		t_d = 0.2	
Controller	PI-A	NPI-A	PI-B	NPI-B
Linear Parameters	$\hat{K}_P$ =1, $\hat{K}_I$ =1 S_u =10	$\hat{K}_P$ =1, $\hat{K}_I$ =0.394 S_u =7.32	$\hat{K}_P$ =1, $\hat{K}_I$ =0.874 S_u =2.52	$\hat{K}_P$ =1, $\hat{K}_I$ =0.961 S_u =3.7
Nonlinear Parameters	None.	θ_P={P_{x1}} ={0.1}	None.	θ_P={P_{x1},P_{y1}} ={0.7,0.2}

It can be observed that the nonlinear PI controllers provide better performance than the linear counterparts. When a time delay occurs to the process, the nonlinear proportion function changes from a "C" curve into an "Inverse-C" curve (Fig. 5-6). One can also observe the plots of the equivalent nonlinear proportional gains in two situations. From these examples one can understand why a direct design of nonlinear gains will be more difficult. Designers usually have no knowledge about the gain value at a zero-error point; neither about the changing tendency for the nonlinear functions.

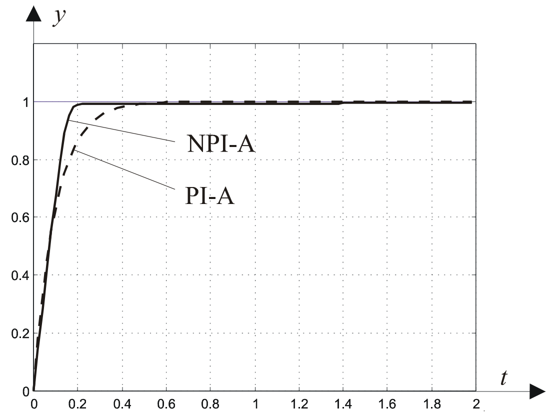

Fig. 3. Step response of the first-order process without time delay (t_d=0).

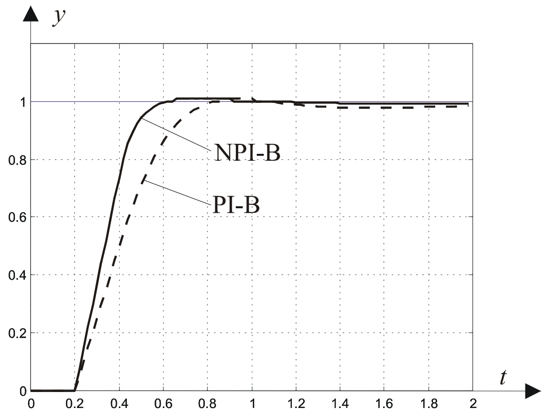

Fig. 4. Step response of the first-order process with time delay (t_d=0.2).

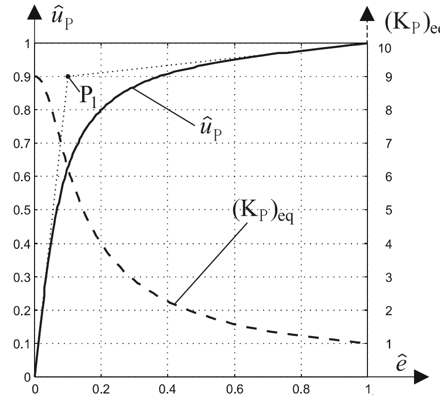

Fig. 5. Plots of $\hat{u}_P$ and $(K_P)_{eq}$ for the NPI-A controller.

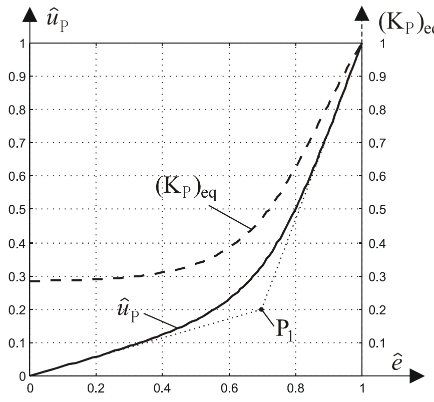

Fig. 6. Plots of $\hat{u}_P$ and $(K_P)_{eq}$ for the NPI-B controller.

CASE 2: Step response of second-order plant with dead zone.

Liu and Daley (2000) proposed a nonlinear PID controller for a dead zone process, which consists of a PID controller plus a dead-zone compensator (Fig. 7). In this case study, we will show that a dead-zone process can also be controlled by smooth nonlinear functions using NPID-PCA controllers. Since they did not provide an explicit expression for the process investigated in [Liu and Daley, 2000], we obtain the estimation of the process from the simulation with the following transfer function:

$$G = \frac{2}{s^2 + 4.4s}, \qquad u^- = -10, \qquad u^+ = 10, \qquad I_1 = \pm 0.5$$

where I_1 is a half width of the dead zone. Fig. 8 shows the step responses from three PD type controllers. NPID-LD is a controller designed from [Liu and Daley, 2000], and NPD-A and

NPD-B are nonlinear controllers based on the present approach. All tuning parameters are given in Table 2 for the three controllers.

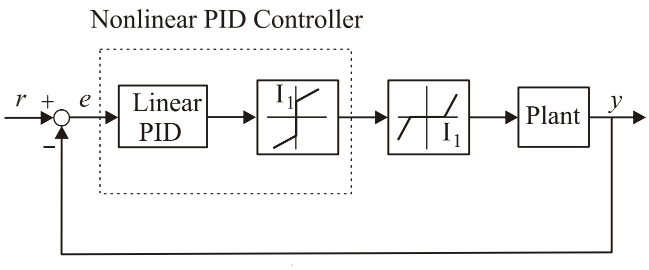

Fig. 7. Nonlinear PID controllers in [Liu and Daley, 2000].

Table 2. Parameters of nonlinear controllers for step responses of the second-order process with a dead zone.

Controller	NPD-LD [Liu, 2000]	NPD-A	NPD-B	
Linear Parameters	K_P=3.21, K_D=0.043	$\hat{K}_P$ =1, $\hat{K}_D$ =0.15 S_u =3.0	$\hat{K}_P$ =1, $\hat{K}_D$ =0.15 S_u =10	
Nonlinear Parameters	A dead-zone Compensator.	θ_P ={P_{x1},P_{y1}} ={0.01,0.9}	θ_{P1} ={P_{x1},P_{y1},P_{x2},P_{y2}} ={0.01,0.9,0.9,0.01}	θ_{P3} ={P_{x1},P_{y1}} ={0.01,0.9}

Note that NPD-A applies a "C" curve for the nonlinear proportional function (Fig. 9), which presents a similar shape as the dead-zone compensator used in Fig. 7. If one increases the value of S_u(=3.0) for the NPD-A controller, the process will result eventually in an overshooting response. In order to improve the performance further, the NPD-B controller employs two independent nonlinear functions (Fig. 10). While $\hat{u}_{P3}$ still remains to be the same "C" curve in the NPD-A controller, $\hat{u}_{P1}$ has changed into an "S" curve. We choose this curve since it preserves the similar shape as the dead-zone compensator around the origin. At this time, the value of S_u(=10) is enlarged greatly, yet without the overshoot.

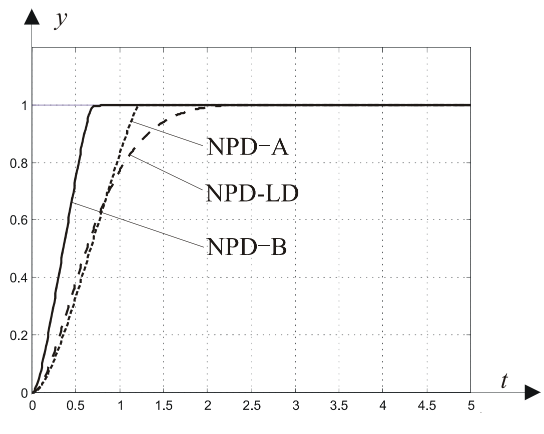

Fig. 8. Step responses of nonlinear controllers for a second-order process with dead zone.

Examining the both equivalent nonlinear proportional gains in Figs. 9 and 10, one can observe that the plot difference between the variable gains is not much in comparison with the "C" curve and "S" curve for the nonlinear proportional functions. This observation confirms the selection of nonlinear proportional functions for the nonlinear design.

The study above demonstrates that the present NPID-PCA controllers are applicable even for processes with dead zones. We can see that the "*Simple first*" strategy is quite helpful in a control design. When the performance is not satisfied to the process, users can add more parameters one by one. Users will not lose the physical meaning for their design. This strategy is usually better than that so called "*Complex reduction*".

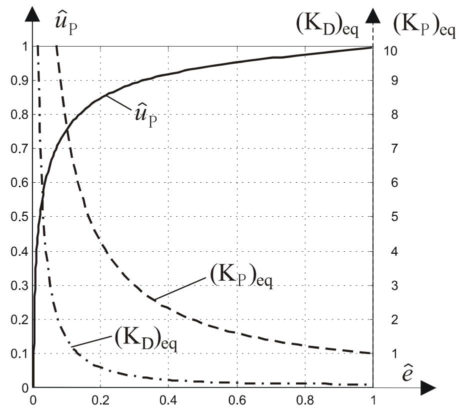

Fig. 9. Plots of $\hat{u}_P$ and $(K_P)_{eq}$ for the NPD-A controller.

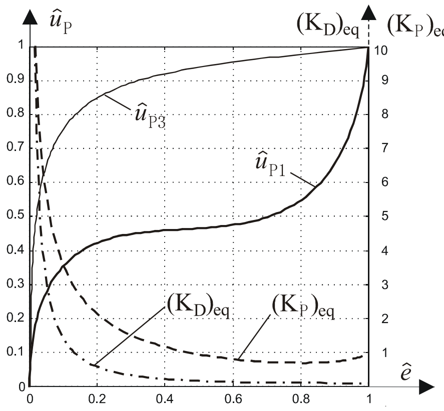

Fig. 10. Plots of $\hat{u}_{Pi}$ and $(K_P)_{eq}$ for the NPD-B controller.

<u>CASE 3: Step response of second-order plant with small damping.</u>

This case study is taken from an investigation given by Shahruz and Schwartz in 1997, where they also proposed two types of high-performance nonlinear PI controllers in the following algorithms:

$$NPI5-SS: \qquad u(t) = [K_P + g_P e^{\lambda|e(t)|}]e(t) + K_I \xi(t),$$

$$NPI6-SS: \qquad u(t) = \frac{a_0 + a_1|e(t)|}{b_0 + b_1|e(t)|} e(t) + K_I \xi(t),$$

$$\frac{d\xi(t)}{dt} = \frac{e(t)}{1+\mu e^2(t)}, \quad \xi(0) = 0.$$

where their designs of NPI5-SS and NPI6-SS controllers applied total 5 and 6 parameters, respectively. Their process example is a second-order plant with small damping:

$$G = \frac{s+1}{s^2 + 0.01s + 1}.$$

In their investigation, a linear PI controller, PI-SS, was also given. For comparing with their methods, we design a controller, NPI-C, based on the present approach, which employs only total 4 parameters. All parameters for the four controllers are given in Table 3. The controllers from [Shahruz and Schwartz, 1997] were obtained by optimization designs on a minimum of both error and control force.

Table 3. Parameters of controllers for step responses of the second-order process with small damping.

Controller	PI-SS [Shahruz, 1997]	NPI5-SS [Shahruz, 1997]	NPI6-SS [Shahruz, 1997]	NPI-C
Linear Parameters	K_P=3.15, K_I=3.38	K_P=2.36, K_I=267.39	K_I=270.0	$\hat{K}_P$ =1, $\hat{K}_D$ =0.15 S_u =10
Nonlinear Parameters	None.	g_P=171.0, λ=-90.99,μ=37.01	a_0=19.36, a_0=19.04, b_0=0.5748, a_0=13.01 μ=30.17	θ_P ={P_{x1}} ={0.01}

Figs. 11-12 show the step responses and control forces of four controllers, respectively. Due to the set-point value is 3, the scaling factor is set as S_e=1/3 for NPI-C. This controller presents the best performance (Fig. 11), but it consumes more energy to the control force (Fig. 12). We can conclude that the NPID-PAC controllers compete closely in performance with the controllers proposed by Shahruz and Schwartz, but present the better physical meanings to the nonlinear tuning parameters.

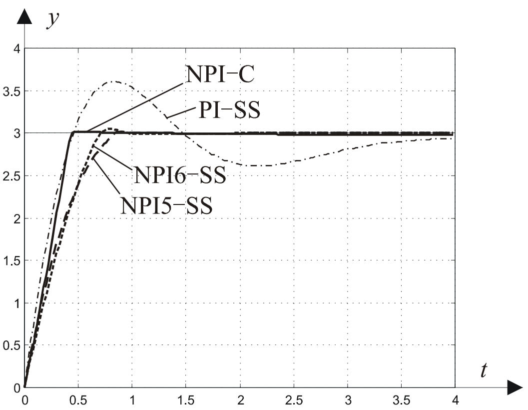

Fig. 11. Step response of the second-order process without small damping.

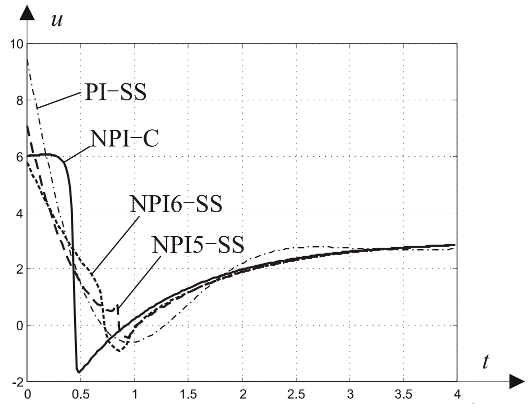

Fig. 12. Control forces for the four controllers in Fig. 11.

From the simulation examples above one can understand that nonlinear PID controllers serve only for approximations of simple nonlinear functions, rather than a conventional concept of a "universal approximators". If a linear function is the first approximation for the control force, then the four types of curves will present the next approximation for the unknown nonlinearities of control forces.

For the nonlinearity tuning, we can see that the present NPID-PCA controllers work well for using the "*Simple first*" strategy. The degree of complex can be added step by step. First, users can apply a linear PID controller directly for the initial tuning of the processes. If the performance is not satisfactory, one can impose a single nonlinear function on the controllers. At this stage, the total number of nonlinear tuning parameters can be selected from 1, 2 to 4 eventually. If it is still unsatisfied, multiple nonlinear functions can be used. Therefore, the incremental means of nonlinear parameters in the NPID-PCA is quite straightforward for users to manipulate the complexity of controllers.

5. NPID-PCA Toolkit on Scilab/Scicos

In the present NPID-PCA, most simulations are made on Scicos, which is a simulation toolbox included in Scilab. Scicos provides a user-friendly GUI-based editor for modeling dynamical systems. The most distinguish feature for Scilab/Scicos is open source to publics. This feature is very important for engineering training and university studies. It will encourage and promote fast distributions and communications of new ideas and results freely within engineering and academic communities.

In this chapter, we will introduce some procedures for using the NPID-PCA simulation toolkit. The NPID-PCA has been designed and tested on Scilab/Scicos 3.0 in the Windows environment. One can obtain this toolkit from the web address below:

http://liama.ia.ac.cn/hubg/Scilab

Within the “zip” file entitled “NPID-PCA.zip”, it includes 11 “cos” files, which are used on Scicos. Each file corresponds to the one specific controller as listed in Table 4.

Table 4. Name list for simulation files in NPID-PCA toolkit.

Case Number	File Name	Controller Type	Total Parameters	Referenced Figure(s)
CASE 1	CASE1_PI_A.cos	Linear PI	3	Fig. 3
	CASE1_PI_B.cos		3	Fig. 4
	CASE1_NPI_A.cos	Nonlinear PI	4	Figs. 3, 5
	CASE1_NPI_B.cos		5	Figs. 4, 6
CASE 2	CASE2_NPD_LD.cos	PD+Compensator	2+1	Fig. 8
	CASE2_NPD_A.cos	Nonlinear PD	5	Figs. 8, 9
	CASE2_NPD_B.cos		9	Figs. 8, 10
CASE 3	CASE3_PI_SS.cos	Linear PI	2	Figs. 11, 12
	CASE3_NPI5_SS.cos	Nonlinear PI	5	
	CASE3_NPI6_SS.cos		6	
	CASE3_NPI_C.cos		4	

In apart from the “cos” files, “NPID-PCA.zip” also has other files, namely:

1. readme.txt
2. NPID-PCA.pdf (This documentation)
3. Automatica98.pdf (Reference paper)
4. P_TFS99.pdf (Reference paper)
5. P_SMC99.pdf (Reference paper)

After unzipping those files and installing them into the specified subdirectory, one can start the simulation one by one. Here, we describe the steps in running a demo as follows.

1. Run Scilab and receive a window as shown in Fig. 13. This is main window for Scilab.
2. From the pull-down menu, select “File” and then “Chang Directory”, to move your current directory into the one that includes the all NPID-PCA simulation files.

3. From the pull-down menu, select “Applications” and then “Scicos”. You will see a new window called “Untitled”. This is a main window for Scicos.

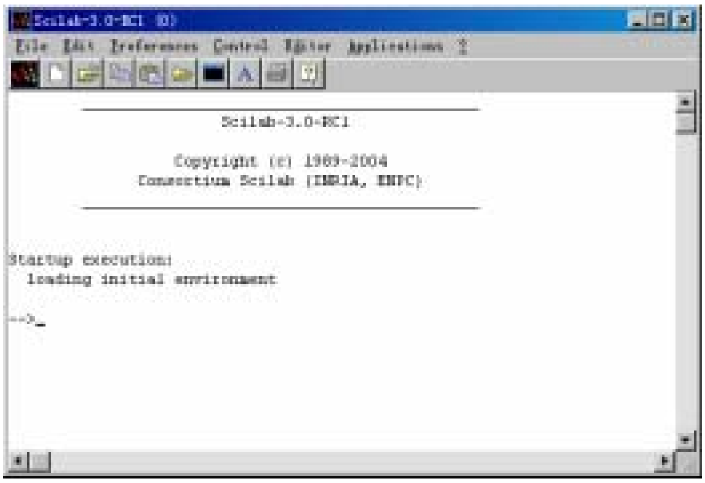

Fig. 13. Scilab main window.

4. From the pull-down menu of the Scicos window, select “Diagram”, “Load”, and then one file, say “CASE1_NPI_B.cos”. The current window is shown as Fig. 14, and renamed automatically as “CASE1_NPI_B”.

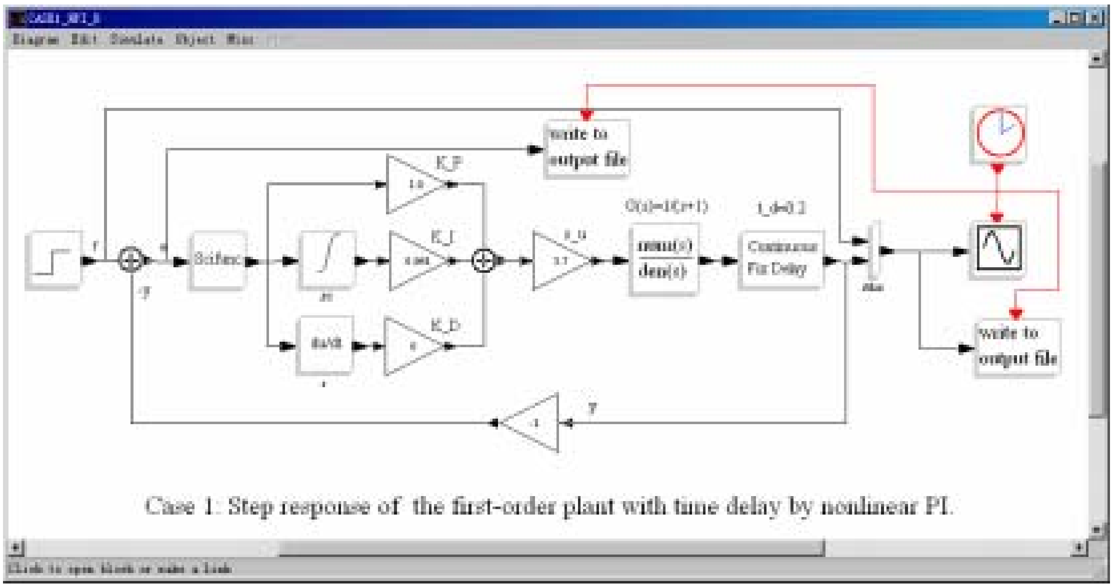

Fig. 14. Setup of the demo of “CASE1_NPI_B” on the Sciocos window.

5. From the pull-down menu of the Scicos window shown in Fig. 14, select “Simulate” and then “Run”, one will see a new window as shown in Fig. 15, which illustrates the step response of the controller. It presents the same results as the plot from the “NPI-B” controller in Fig. 4.

Note that some demos may also present the plot of “$\hat{u}_P - \hat{e}$” after running the files. Users can exam each file to confirm the simulation results presented by this documentation. Each demo gives users a convenience to set their own parameters for the improved performances.

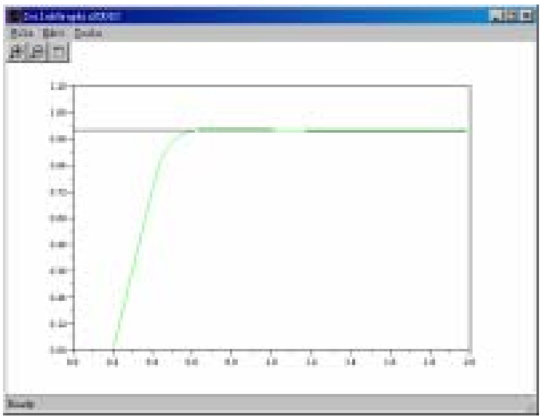

Fig. 15. The plot window for the demo of “CASE1_NPI_B”.

For changing the parameters, there usually have two ways, depending on the setup in the original files. For example, one can select the gain block, and input the specified value to the block directly, like four gain blocks (in a triangle shape) for K_P, K_I, K_D, and s_u in Fig. 14. In the other way, one need to select “Edit” and then “Context” in pull-down menu, and change the values in the new “Scilab Dialog” windows shown in Fig. 16, where Px1, Py1, Px2 and Py2 are nonlinear parameters as the control points. In practice, all parameters can be inputted by the way shown in Fig. 16.

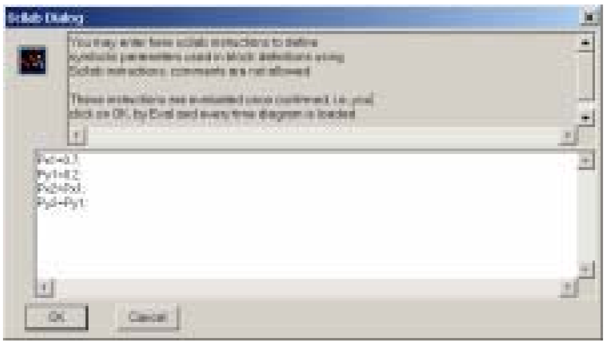

Fig. 16. The “Scilab Dialog” window for changing the parameters.

In Fig. 14, one block called “Scifunc” is used to construct the nonlinear proportional function. It should be written by the Scilab language. The main program for this nonlinear calculation in the “Scifunc” block is given below:

```
er=abs(u1);
if er > 1 then
write(%io(2), 'You need to scale down the error signal within unit range');
write(%io(2), 'er=');
write(%io(2), er);
end
```

```
s=poly(0,"s");
p1=3*s.*(1-s).^2*Px1 +(3*s.^2).*(1-s)*Px2+s.^3;
p=p1-er;
ss=(roots(p));
s1=[];
if er==0 then s1=0; end;
  if er==1 then s1=1; end;
  for j=1:3
   if abs(ss(j))==real(ss(j)) then
   ssr=real(ss(j));
    if ssr >= 0 then
       if ssr <= 1 then
           s1=ssr;
       end
    end
  end
end
if s1==[] then
    write(%io(2), 'u1=');
    write(%io(2), u1);
end
y1= 3*s1*(1-s1)^2*Py1 +(3*s1^2)*(1-s1)*Py2+s1^3;
y1=y1*sign(u1)
```

For a given value of $\hat{e}$, the program above has to find the corresponding value of $\hat{u}_P$. Eq. (6) consists of two basic functions for calculating their relations. We use an absolute value, $|\hat{e}|$, for the calculation of $|\hat{u}_P|$ first. Since $|\hat{e}|$ is limited within a unit range, this program makes a range check first on the current $|\hat{e}|$. If it is out off the range, an output message will be given to the user. The next step in the program is to solve a cubic function for the current parameter value of s from the first equation in Eq. (6). Since three solutions of s will be obtained, the one only within the unit range is selected in the program. If this solution cannot be found from a unit range, a message will be given to the user. Whenever the current value of s is obtained, the corresponding value of $|\hat{u}_P|$ is calculated directly from the second equation in Eq. (6). The last command of the program is to find a real value of $\hat{u}_P$.

After testing on each demo in the NPID-PCA simulation toolkit, users can try different processes and control strategies.

6. Conclusions

It is reported that PID controllers continue to be an important method in control engineering [Aström and Hägglund, 1995; Tan, *et al*, 1999; Datta, *et al*, 2000]. About 90 percent controllers can be related to the PID control technique in process control. This fact indicates that any improvement in this technique may result in a big impact on all related industrials. For reaching such improvement, we believe that a study on nonlinear control will be one of the important directions.

Although an introduction of nonlinear features is able to enhance the overall performances of PID technique, it also brings a much higher degree of complexity in the design and analysis. In principle, adding one nonlinear parameter will require significantly bigger efforts than adding one linear parameter. At the same time, another great challenge is how to synthesize nonlinear PID controllers that will preserve the distinguished features exhibited by the conventional PID technique.

In this work, we describe a new simulation toolkit of nonlinear PID control, namely, NPID-PCA. A "simplicity" principle is persistently followed for the design of NPID-PCA controllers. Since the proportional forces are used for the nonlinear functions, the controllers preserve the simplest properties as well as the most physical meanings in the nonlinearity design. We adopt a spline-based function for forming nonlinear proportional functions. This approach provides a high degree of flexibility in generating four simplest nonlinear curves. Therefore, some existing nonlinear PID controllers reported in literature may fall into a special case into the present controllers if examining their nonlinear curves.

To demonstrate the applicability of the present controllers, several examples are tested in comparing with the other existing nonlinear PID controllers. The simulation results confirm the superior performances of the NPID-PCA controllers. From the demos in the toolkit, users can find that the implementation is also quite important. The present controller does provide an effective way for realizing the "*Simple first*" strategy in the nonlinearity design. At the same time, all parameters in the controllers are well defined within the compact ranges, which may improve the cost of the optimization design of controllers greatly.

The present NPID-PCA seems to be a first free toolkit of nonlinear PID controllers opened in publics. We encourage users to use, distribute, and modify the present simulation toolkit. It is the author's hope this work can stimulate in-depth studies on the nonlinear PID control technique. Within these studies, simulation toolkits also play an important role. We recognize that the simulation toolkit "NPID-PCA" needs to be improved in many aspects. A user interface with a graphic means to the nonlinear curve design is expected. By this way, users can immediately visualize the shape of nonlinear functions and nonlinear gains, respectively. An optimization tool is also necessary for a well tuning of parameters. A genetic algorithm approach is suitable for the purpose. One toolbox in such subject, called "**GATS**" (**G**enetic **A**lgorithm **T**oolbox for **S**cilab) developed by Li (2004), could be used, but a further work is needed for a proper integration of two parts.

The PID control technique has the longest history in our control engineering life. We, sometimes, claim that PID control is the most matured technique in controlling applications. However, when examining it carefully, we have to recognize that some important, yet essential, studies remains for the nonlinear design and analysis of the PID control technique. In fact, we are still far away from realizing a full knowledge or an in-depth understanding to any one of the following open problems:

- **Relationship of “nonlinearity and stability”**
- **Relationship of “nonlinearity and performance”**
- **Relationship of “nonlinearity and robustness”**
- **Tuning rules for nonlinear parameters in together with linear ones**
- **Nonlinear control on real-time control to time-delayed processes**
- **“Smooth nonlinear” control for various “hard nonlinear” compensations**
- **“Time variant” or dynamic design of nonlinear PID controllers**
- **Systematic designs and evaluations of various nonlinear PID controllers**

7. References

1 Armstrong, B., Neevel, D. and Kusik, T., New results in NPID control: Tracking, integral control, friction compensation and experimental results. *IEEE Trans. Control Sys. Tech.*, 2001, **9**(2) 399 – 406.

2 Aström, K.J. and Hägglund, T., *PID Controllers: Theory Design and Tuning*, 1995, Instrument Society of America, Research Triangle Park, North Carolina.

3 Bucklaew, T.P., and Liu, C.-S., A new nonlinear gain structure for PD-type controllers in robotic applications. *J Robotic Systems*, 1999, **16**(11), 627-649.

4 Chang, W.-D., Hwang, R.-C., Hsieh, J.-G., A self-tuning PID control for a class of nonlinear systems based on the Lyapunov approach, *Journal of Process Control*, 2002, **12**(2), 233-242.

5 Chen, W.-H., Ballance, D.J., Gawthrop, P.J., Gribble, J.J., O'Reilly, J., Nonlinear PID predictive controller, *IEE Proceedings-Control Theory and Applications*, 1999, **D146**(6), 603 – 611.

6 Datta, A, Ho, M.T., and Bhattacharyya, S.P., *Structure and Synthesis of PID Controllers*, 2000, Springer, London.

7 Feng, W., O'Reilly, J., Ballance, D.J., MIMO nonlinear PID predictive controller, *IEE Proceedings-Control Theory and Applications*, 2002, **D149**(3), 203 – 208.

8 Gao, Z., From linear to nonlinear control means: a practical progression, *ISA Trans.* 2002, **41**(2) 177-89.

9 Han, J.Q., Nonlinear PID controller, *Acta Automtica Sinca*, 1994, **20**(4), 487-490, (in Chinese).

10 Hornik, K., Stinchcombe, M., White, H., Multilayer feedforward networks are universal approximators, *Neural Networks*, 1989, **2**(5), 359 – 366.

11 Hu. B.-G., Mann, G.K.I. and Gosine, R.G., Control curve design for nonlinear (or fuzzy) proportional actions using spline-based functions. *Automatica*, 1998, **34**(9), 1125-1133.

12 Hu, B.-G., Mann, G.K.I. and Gosine, R.G., New methodology for analytical and optimal design of fuzzy PID controllers, *IEEE Trans Fuzzy Sys.* 1999, **7**(5) 521-539.

13 Huang, C.Q., Wang, X.G., and Wang, Z.Q., A class of transpose Jacobian-based NPID regulators for robot manipulators with uncertain kinematics, *J Robotic Systems*, 2002, **19**(11), 527-539.

14 IEE. Special Issue on PID control, *IEE Proc. D, Control Theory and Applications*, 2002, **149**(1), 1-116.

15 IFAC. Special Issue on PID controller, *Control Engineering Practice*, 2001, **9**(11), 1159-1263.

16 INRIA-ENPC, Scilab/Scicos, http://scilabsoft.inria.fr/

17 Jutan, A., A nonlinear PI(D) controller, *Canadian Journal of Chemical Engineering*, 1989, **67**(6), 485-493.

18 Kelly, R., Carelli, R., A class of nonlinear PD-type controllers for robot manipulators, *Journal of Robotic Systems*, 1996, **13**(12), 793-802.

19 Li, H.X. and Gatland, H.B. Conventional fuzzy logic control and its enhancement, *IEEE Transactions on Systems, Man and Cybernetics*, 1996, **26B**(5), 791-797.

20 Li, Z., GATS: Genetic Algorithm Toolbox for Scilab, 2004,

http://liama.ia.ac.cn/SCILAB/winner2003.htm
21 Liu, G.P. and Daley, S., Optimal-tuning nonlinear PID control of hydraulic systems, *Control Eng Practice*, 2000, **8**(9) 1045-1053.
22 Mann, G.K.I., Hu, B.-G. and Gosine, R.G., Analysis of direct action fuzzy PID controller structures, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics,* 1999, **29**(3), 371-388.
23 Malki, H.A., Li H., and Chen, G., New design and stability analysis of fuzzy proportional-derivative control system, *IEEE Trans. Fuzzy Systems*, 1994, **2**(4), 345-354.
24 Ortega, R., Astolfi, A. Barabanov, N.E., Nonlinear PI control of uncertain systems: an alternative to parameter adaptation, *Systems and Control Letters*, 2002, 47(3), 259-278.
25 Park, J., Sandberg, J.W., Universal approximation using radial basis functions network, *Neural Computation*, 1991, **3**(2), 246-257
26 Ruano, A.E.B., Fleming, P.J. and Jones, D.I., Connectionist approach to PID autotuning. *IEE Proc D - Control Theory and Appl.* 1992, **D139**(3), 279 – 285.
27 Rugh, W.J., Design of nonlinear PID controllers, AIChE Journal, 1987, **33**(10), 1738-1742.
28 Scott, G., Shavlik, J. and Ray, W.(1992), Refining PID controllers using neural networks, *Neural Computation*, 1992, **4**, 746-757.
29 Serji, H., A new class of nonlinear PID controllers with robotic applications. *J Robotic Systems*, 1998, **15**(3), 161-181.
30 Shahruz, S.M. and Schwartz, A.L., Design and optimal tuning of nonlinear PI compensators, *Journal Optim. Theory Appli.*, 1994, **83**(1), 181-198.
31 Shahruz, S.M. and Schwartz, A.L., Nonlinear PI Compensators that achieve high performance. *J. Dynam. Sys. Measu. Control., Trans. ASME*, 1997, **119**(2), 105-110.
32 Su, Y.X., Duan, B.Y., Zheng, C.H., Nonlinear PID control of a six-DOF parallel manipulator, *IEE Proceedings-Control Theory and Applications*, 2004, **D151**(1), 95 – 102.
33 Tan, K. K., Wang, Q.- G., Hang, C. C., *Advances in PID*, 1999, Springer, New York.
34 Tan, K.K., Lee; T.H., Zhou, H.X., Micro-positioning of linear-piezoelectric motors based on a learning nonlinear PID controller, *IEEE/ASME Transactions on Mechatronics*, 2001, **6**(4), 428 - 436
35 Wang, L.-X. and Mendel, J.M. Fuzzy basic functions, universal approximation, and orthogonal least-squares learning, *IEEE Trans. Neural Networks*, 1992, **3**(5), 807-814.
36 Xu, Y, Hollerbach, J.M, and Ma, D., A nonlinear PD controller for force and contact transient control. *IEEE Control Systems Magazine*, 1995, **15**(1) 15–21.
37 Ying, H., Sufficient conditions on general fuzzy systems as function approximations, Automatica, 1994, **30**(3), 521-525.
38 Zhao, Z.-Y. Tomizyoshi, M. andIsaka, S., Fuzzy gain scheduling of PID controllers, *IEEE Trans. Syst., Man, Cybern.*, 1993, **23**(5), 1392–1398.

8. Acknowledgement

This work is supported in part by Nature Sciences of Foundation of China (#60275025) and Chinese National 863 Program (2003AA1Z2620). Special thanks also go to M. Goursat, S. Steer, C. Gomez, J.-P. Chancelier, F. Delebecque, and R. Nikoukhah from INRIA for their kind helps and encouraging discussions in using Scilab/Scicos. I always remember the good times with them and Ph. de Reffye at Rocquencourt of INRIA for the pleasant coffee and talks. I also thank to Prof. Guo Ping Liu for the discussion about the PID technique. At last, I would like to thank to LIAMA, which gives me a convenient chance to work on Scilab/Scicos.