Research Article

INTERNATIONAL JOURNAL OF ADVANCED ROBOTIC SYSTEMS

International Journal of Advanced Robotic Systems
November-December 2021: 1–11

DOI: 10.1177/17298814211057698
journals.sagepub.com/home/arx
SAGE

A neural network-based model predictive controller for displacement tracking of piezoelectric actuator with feedback delays

Zhangming Du[1,2], Chao Zhou[1], Zhiqiang Cao[1], Shuo Wang[1,3], Long Cheng[1] and Min Tan[1]

Abstract

Piezoelectric actuators are widely used in micro/nanoscale robotic manipulators. Due to its hysteresis and dynamic-related nonlinearity, accurate displacement tracking control of piezoelectric actuator is challenging. Besides, in some low-cost practical systems with low sampling rate, transmission delay causes mismatches between feedback and real displacement, further increasing the challenge in tracking control. In this article, a neural network-based model predictive controller (MPC) is proposed for precise tracking control of piezoelectric actuator's displacement in situation where feedback is slow and delayed. The prediction model is based on a nonlinear-autoregressive-moving-average-with-exogenous-inputs framework, which outputs entire prediction horizon of future displacement in a single time, and is fulfilled by a multilayer feedforward neural network. An extended Kalman filter-based estimation for displacement is introduced to relieve the influence of feedback delays so as to improve dynamic performance of the controller. Another neural network is trained to provide initial values for MPC to reduce computation costs and improve performance in dynamic tracking. In a series of tracking experiments, the effectiveness of proposed controller is verified.

Keywords

Piezoelectric actuator, neural network, displacement tracking, model predictive control

Date received: 10 November 2020; accepted: 18 October 2021

Topic: Micro/Nano Robotics
Topic Editor: Quan Zhan
Associate Editor: Qingsong Xu

Introduction

Manipulating objects in nanoscale is a fast growing demand in research of nanotechnology, and robotic nano-manipulators are developed to serve this goal. With excellent performance in precise positioning and capability of generating large forces, piezoelectric actuators (PEAs) have been widely used as motion components of robotic micro/nano-manipulators.[1,2] However, driven by inverse piezoelectric effect,[3] the displacement of a PEA is greatly influenced by its nonlinear characteristics, such as

[1] Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] CAS Center for Excellence in Brain Science and Intelligence Technology, Shanghai, China

Corresponding author:
Chao Zhou, Institute of Automation, Chinese Academy of Sciences, No. 95, Zhongguancun East Road, P.O. Box 2728, Haidian District, Beijing 100190, China.
Email: chao.zhou@ia.ac.cn

hysteresis and creep.[4,5] Hysteresis is a phenomenon that the displacement of an actuator depends not only on current input signal but also on memories of past signals. Creep is a delayed deformation which slowly changes displacement after input signal is unchanged. In practice, the frequency of motion also greatly influence the voltage–displacement relationship, that is, the dynamic nonlinearity of PEAs.[6]

Dealing with the aforementioned nonlinear characteristics is crucial to the control quality of piezoelectric displacement. Traditional feedback strategies like proportional–integral–derivative (PID) can easily deal with creeps but have poor performance handling other nonlinearity, especially in dynamic occasions.[7] So feedforward mechanisms which determine control signals according to prior knowledges are introduced to combine with feedback control for improvement in tracking performance.

The model describing nonlinear characteristics of PEAs is the base of feedforwarding, for which a number of modeling methods have been proposed, generally including physics-based models and phenomenon models. Models in former group try to establish model through inherent relationship of physical variables. Jiles–Atherton model is an example of such kind of models.[8] Physics-based models are difficult to build and lacks generality to be applied in different system. The latter group, namely phenomenon-based models, includes three subgroups: differential equation models, operator models, and others. The first two subgroups include Duhem model,[9] Bouc–Wen model,[10] Preisach hysteresis model,[11] and Prandtl–Ishlinskii (PI) hysteresis model.[12] These models try to extract a mathematical law from phenomenon without involving physical principles. Similar to physics-based models, these mathematical hysteresis models usually have complicated structures and cost much to compute, also have to be appended with additional models of creep and dynamic.[13,14] The last group of models are totally data-driven approaches and have relatively simple forms. Examples include autoregressive-moving-average (ARMA) models, fuzzy models, and neural network-based models. Cao et al.[15] proposed a linear ARMA model for hysteresis. Cheng et al.[14] proposed an adaptive Takagi–Sugeno fuzzy model to describe the nonlinear behaviors of PEAs. Wen and Cheng[16] proposed a recurrent fuzzy model. Liaw et al.[17] designed a radial basis function neural network model. Nonlinear ARMA with exogenous input (NARMAX) form is combined with two neural networks to separately obtain hysteresis submodel and dynamic submodel by Cheng et al.[18] As phenomenon-based models are easier to obtain and have better generality, they're more popular in actual application.

Based on model of PEAs, many control algorithms are proposed for PEAs positioning. Inversion-based methods are most widely used. Usually, an inverse model of hysteresis is cascaded with control plant to carry out feedforward compensation, then feedback approaches are used to deal with dynamic and other minor nonlinearity. Ge and Jouaneh[19] designed a PID controller combined with a feedforward compensator based on numerical inverse Preisach model. Al Janaideh et al.[20] used an analytical generalized PI model inversion as compensation in micropositioning control. Song at al.[21] proposed an inverse extended unparallel PI hysteresis model along with an inverse dynamic model to alter the PEA into highly linearity, then a non-vector space approach is proposed to control PEA-driven scanning probe microscope. Li et al.[22] designed an adaptive internal model control (IMC) scheme with a fuzzy hysteresis model and its inversion working together. Jian et al.[23] combined iterative learning control with direct-inverse-compensated PID approach to reduce the tracking errors caused by incompletely compensated hysteresis. However, accuracy inverse models are hard to acquire whereas the computation costs are usually high, which is a significant drawback for online tracking. Some inversion-free methods are then proposed. Al Janaideh et al.[24] designed a control scheme, in which a PI model acts in a feedback fashion rather than an inversion as a feedforward compensation, with another equivalent linear model derived as internal model of the compensated plant so as to form feedforward–feedback control in an IMC scheme. However, this kind of controller have complex structure and lack the flexibility for practical applications. Different from aforementioned approaches that use model to compensate nonlinearity, model predictive control (MPC) is a more straightforward method, widely used in practical applications.[25,26] Nonlinear model predicts future displacement based on input of model, and actual control signal applied on PEA is determined according to predicted displacement.[18,27] MPC usually takes consecutive predicted future outputs over a certain prediction horizon for optimization, which demands repeated running of the model, thus increases computational burden. A popular solution is linearizing the prediction model to simplify calculation.[28,27] In addition, the accuracy of model greatly influences the performance of MPC.

Besides the nonlinearity, imperfect feedbacks from sensors can also cause problems, inefficient sampling rate with delays or even data losses exist in practice, especially for low-cost hardware systems. As changing voltage directly acts on PEA's displacement in extremely short time, delay of sensors can cause large mismatches between feedback and real displacement when PEA deforms rapidly, increasing the challenge in dynamic tracking control. The performances of many typical controllers significantly deteriorate in such situation. The simplest solution for this problem is improving the sensors by updating hardwares to increase the sampling/transmission rate which however can be costly and difficult in some actual applications. Hence, some soft methods are proposed to solve similar problems, usually based on estimation of delay and additional model of corresponding states, combined with multiple controllers under switching rules.[29,30] However, few has been adopted

in tracking control of PEA, which has special nonlinear characteristics as aforementioned.

In this article, aiming to improve tracking performance of PEA control systems where low-rate sensor is used, a neural network-based model predictive controller is introduced, which works well when feedback is very inefficient and delayed. The proposed method also requires less computational resources than traditional neural MPC (NMPC), make it easier to be applied in practical systems. First, a NARMAX structure is adopted for nonlinear modeling of PEA, implemented by a multilayer feedforward neural network (MFNN). To ease the computation, the model is designed to output an entire prediction horizon of future displacement altogether when called, avoiding repeated calculation in sequential prediction. Then the control issue is transformed into an optimization problem that iteratively adjusts the control signals to minimize the errors between predicted displacement and reference. Another MFNN is trained as a feedforward controller, outputting an initial control signal for the optimization. In addition, an accumulated-error compensation term is embedded into controller, to reduce steady-state error caused by model mismatches. To relieve the influence of delayed feedbacks, an extended Kalman filter (EKF)-based method is adopted to fuse sensor feedbacks with the predicted displacement, providing controller with better state estimation so as to accordingly modify the control signal. To evaluate the performance of proposed controller, a series of experiments have been conducted versus PID and a typical NMPC, and the result verifies effectiveness of proposed controller.

Neural model of PEAs based on NARMAX

The main nonlinear characteristics to be modeled are hysteresis and rate-dependent dynamic property. As aforementioned, hysteresis is a kind of memory phenomenon, which relates displacement of PEA to states and signals in the past. Meanwhile, frequency can be estimated by extracting differential values from discrete signal sequence, so rate-dependent property can also be viewed as a behavior determined by historical signals and states. Hence, the input–output relationship of PEAs can be concluded as following function

$$\hat{y}(k) = f[y(k-1), ..y(k-n_y), u(k-1), ..u(k-n_u)] \quad (1)$$

where $y(i)$ and $u(i)$ separately indicate the displacement and control signal of a PEA at operating point i. And integers n_y and n_u are the sizes of historical memory of $y(i)$ and $u(i)$, respectively. This is exactly the same form of NARMAX, which has powerful capability of modeling for time series prediction problems with complex, nonlinear, and dynamic nature.[31]

To achieve smoother performance of MPC, a series of predicted outputs $[\hat{y}(k+1), .., \hat{y}(k+n_p)]$ within prediction horizon n_p are demanded at instant k. Usually, they are successively outputted by the model, provided previous predictions as a part of historical states, that is, using $\hat{y}(k+i)$ as $y(k+i)$ when predicting $\hat{y}(k+i+1)$. This repeated process are computational costly. To relieve this problem, the model (1) is modified to output all n_p future predictions in a single time. Then the NARMAX model is rewritten as

$$[\hat{y}(k+1), ..\hat{y}(k+n_p)] = f[y(k), ..y(k-n_y), u(k+n_p-1), ..u(k), ..u(k-n_u)] \quad (2)$$

where $[u(k+n_p-1), ..u(k)]$ are presumed output signals of controller. This model is only effective when prediction horizon is short, otherwise the predictive accuracy may be poor and the training of MFNN will become difficult. In our case, n_p is set to 3.

NARMAX is a framework of input–output representation, whereas obtaining specific model, that is, determining the function $f[\cdot]$ in (2), is the main challenge. In theory, neural networks are able to approximate any continuous function with an arbitrary degree of accuracy, provided that the number of hidden nods is sufficient enough.[32] In practice, neural networks often use fewer parameters and achieve better precision than other approaches in approximation tasks.[25] In this article, an MFNN is adopted to approximate the function $f[\cdot]$ in (2).

Neural model of PEAs

An MFNN with single hidden layer is used as approximator for NARMAX representation $f[\cdot]$. It acts as the model of PEA after sufficiently trained, outputting displacement prediction if provided valid input. As Figure 1 shows, n_y historical displacements $[y(k-1), ..y(k-n_y)]$ and n_u historical control signals $[u(k-1), ..u(k-n_u)]$ together with current states $y(k)$ and presumed control signal $[u(k+n_p-1), ..u(k)]$ are inputted into neural network. $[\hat{y}(k+1), ..\hat{y}(k+n_p)]$ indicates the output of the network, that is, the predicted displacements of PEA. w_{ji}^h indicates connection weight from ith nods in input layer to the jth nods in hidden layer, and b_j^h indicates the bias of jth nods in hidden layer. Similarly, w_{lj}^o indicates the connection weight between jth nods in hidden layer and the lth nods in output layer, whereas b_l^o is the corresponding output bias. Hyperbolic tangent is used as activation function σ in hidden layer

$$\sigma(x) = \tan h(x) = \frac{e^{2x}-1}{e^{2x}+1} \quad (3)$$

And the activation function in output layer is set linear.

Identification of model

Let $X = [y(k), ..y(k-n_y), u(k+n_p-1), ..u(k-n_u)]^T$ indicates the input of neural network and $\hat{Y} = [\hat{y}(k+1), ..\hat{y}(k+n_p)]^T$ indicates corresponding outputs,

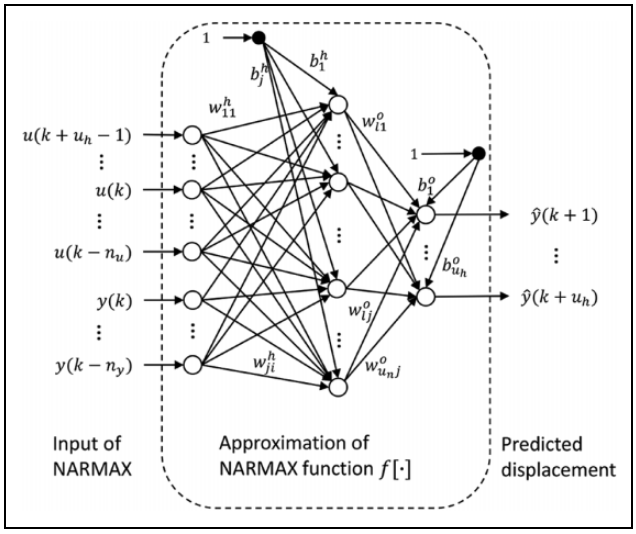

Figure 1. MFNN approximator for NARMAX-based model of PEAs. MFNN: multilayer feedforward neural network; NARMAX: nonlinear-autoregressive-moving-average-with-exogenous-input; PEA: piezoelectric actuator.

then the neural model can be concluded as following function

$$\hat{Y} = f(X) = W^o\overline{\sigma}(W^hX + B^h) + B^o \tag{4}$$

where $W^o \in \mathbb{R}^{n_p \times n_h}$ and $W^h \in \mathbb{R}^{n_h \times (1+n_p+n_u+n_y)}$ are weight matrices for output layer and hidden layer, respectively. n_h is the number of hidden nods. $B^h \in \mathbb{R}^{n_h \times 1}$ and $B^o \in \mathbb{R}^{n_p \times 1}$ are bias matrices of hidden layer and output layer. $\overline{\sigma}(\cdot)$ indicates running $\sigma(x)$ for every element x in corresponding matrix.

Then the problem becomes determining the parameter matrices W^o, W^h, B^h, and B^o. Provided sufficient input–output data pairs sampled in experiment measuring displacement of PEAs, supervised training can be conducted to solve following optimization problem

$$\min_{W^h,W^o,B^h,B^o} \sum_i (\hat{Y}_{X(i)} - Y(i))^T(\hat{Y}_{X(i)} - Y(i)) \tag{5}$$

where $\hat{Y}_{X(i)}$ is the output of model for input X in ith sample, and $Y(i)$ is the corresponding real PEA displacement series. In addition, all aforementioned inputs and desired outputs of model should be separately normalized according to their scales, to ensure properly training of neural network.

There are plenty of proven optimization approaches for neural network training. In this article, adaptive moment estimation algorithm (ADAM) is chosen to play the role. As a gradient-descent-based optimization, ADAM computes individual adaptive learning rates for different parameters according to estimations of both first and second moments of gradients.[33] As a result, ADAM shows good convergence in MFNN training and can still work well when samples are with heavy noises. In our case, ADAM excels other approaches both in quality and speed of training. Coordinated with ADAM, $l2$-regularization is also applied to all weights, which effectively prevents overfitting problem. $l2$-regularization introduces a $l2$-norm of weights as penalty to the original loss function, limiting the scale of average weights, which enhances generalization performance of neural model. Let $E(i) = \hat{Y}_{X(i)} - Y(i)$ indicates prediction errors, then the loss function in (5) to be minimized is rewritten as

$$\text{loss} = \sum_i (E(i)^T E(i) + \alpha_1 Tr(W^h W^{hT}) + \alpha_2 Tr(W^o W^{oT}) \tag{6}$$

where α_1 and α_2 are weighting coefficients of regularization, function $Tr(\cdot)$ calculates the trace of a matrix.

Tracking control of displacement of PEA

With aforementioned model of piezoelectric displacement, the model predictive controller repeatedly tests virtual control signals on the model and adjusts outputs according to the corresponding predictions. As the MFNN model is differentiable, such adjustment can utilize the gradients of predicted errors with respect to control signals, hence turns the control issue into an optimization problem. Besides, a term of static error compensation, a neural feedforward controller, and a feedback estimation are supplemented to the MPC controller to improve tracking performance with feedback delays. The block diagram of the whole control scheme is shown in Figure 2.

Model predictive control scheme

Most elements in neural model's inputting vector $X = [y(k), ..y(k-n_y), u(k+n_p-1), ..u(k), ..u(k-n_u)]^T$ are known and fixed at current operating point, only $U(k) = [u(k+n_p-1), ..u(k)]^T$ is a manipulatable variable. Thus, the predicted displacement can be rewritten as

$$\hat{Y}(k) = f(U(k)) \tag{7}$$

The goal is to minimize the error of predicted displacement

$$\begin{aligned} J(U(k)) &= \sum_{i=1}^{n_p} (\hat{y}(k+i) - r(k+i))^2 \\ &= (\hat{Y}(k) - R(k)^{)T}(\hat{Y}(k) - R(k)) \\ &= (f(U(k)) - R(k)^{)T}(f(U(k)) - R(k)) \end{aligned} \tag{8}$$

where $R(k)$ is the set-point of desired displacement $R(k) = [r(k+1), ..r(k+n_p)]^T$. However, at instant k, except next reference point $r(k+1)$, we cannot exactly know $r(k+i)$ when $i > 1$. Usually, all $r(k+i)$ s are set equal to $r(k+1)$, in this case however, we use an

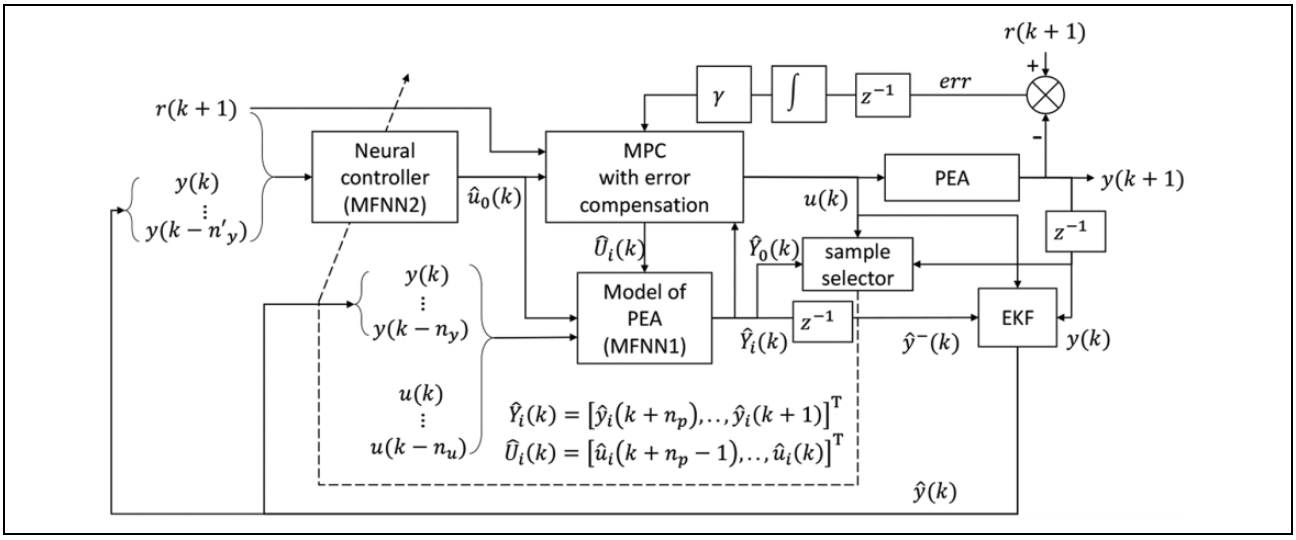

Figure 2. Neural MPC-based tracking control system of PEA. PEA: piezoelectric actuator.

interpolated reference trajectory that can improve response speed of MPC

$$R(k) = [r(k), \ldots, r(k) + i \cdot 2\frac{r(k+1) - r(k)}{n_p - 1}, \ldots, 2r(k+1) - r(k)], \quad i = 0, 1.., n_p - 1 \tag{9}$$

To reduce excessive changes of control signal, a penalty term is added to (8), then the objective function to be minimized becomes

$$J(U(k)) = (f(U(k)) - R(k))^T (f(U(k)) - R(k)) + \beta(\Delta U(k))^T (\Delta U(k)) \tag{10}$$

where β is a penalty coefficient determining the intensity of effort in restraining excessive change of control signal, and $\delta U(k)$ indicates the increment of $U(k)$

$$\Delta U(k) = \begin{bmatrix} 1 & -1 & 0 & \ldots & 0 \\ 0 & 1 & -1 & \ldots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & \ldots & & 1 \end{bmatrix} U(k) + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ u(k-1) \end{bmatrix} \tag{11}$$

As $u(k-1)$ is a known constant at instant k, so $J(U(k))$ in (10) is still determined only by $U(k)$. Then the control issue is transformed into a optimization problem that find an optimal vector $U(k)$ to minimize $J(U(k))$. Levenberg–Marquardt (LM) algorithm is adopted to solve this optimization.

Optimization by LM algorithm. LM algorithm is a combination of Gauss–Newton and trust region method, useful in solving optimization problems with quadratic forms. In our case, objective function (10) is in quadratic forms

$$J(U(k)) = \eta^T \eta \tag{12}$$

where

$$\eta = \begin{bmatrix} f(U(k)) - R(k) \\ \beta \Delta U(k) \end{bmatrix} \in \mathbb{R}^{2n_p \times 1} \tag{13}$$

Then the basic update rule of LM algorithm goes as following way

$$\hat{U}_{i+1}(k) = \hat{U}_i(k) + \mu_i \tag{14}$$

$$\mu_i = -(H_\eta + \lambda I)^{-1} G_\eta^T (U_i(k)) \eta \tag{15}$$

where μ_i is the update vector of $\hat{u}$ in ith iteration. $G_\eta(U_i(k)) = \partial\eta/\partial U_i(k)$ (simplified as G) is the gradient matrix of η with respect to $U(k)$ in ith iteration, that is, Jacobian matrix of η. H_η originally refers to Hessian matrix of η. However, because of its heavy computational burden, H_η is approximately replaced by Gauss–Newton Hessian matrix: $H_\eta = G^T G$. I is a identity matrix and λ is a adaptive coefficient. The term λI adjusts the rate of convergence and ensures the positive definiteness of $(H_\eta + \lambda I)$ to guarantee convergence. λ is updated in each iteration according to gain ratio

$$\rho = \frac{J(\hat{U}_i(k)) - J(\hat{U}_i(k) + \mu_i)}{\Delta L} \tag{16}$$

If ρ is too big, then decrease λ and vice versa. The numerator in (16) is the actual decrement of objective function whereas ΔL is an estimated decrement

$$\Delta L = -\mu_i^T G^T \eta + \frac{1}{2} \mu^T H_\eta \mu \tag{17}$$

The complete LM algorithm is summarized in Algorithm 1:

Algorithm 1. Calculate MPC output by LM algorithm.

When at instant k (In following expression, all (k) s are ignored, H_η is simplified as H, subscript i indicates iterations):
Given reference $R(k)$
Initialize: Acquire initial $\hat{U}_0, G_0$ and H_0, set maximal iterations n and tolerance $\varepsilon > 0$, initialize $\lambda = \tau \cdot \max(H_0)$, i=0;
0: Calculate η_0 and $J(\hat{U}_0)$;
While $i < n$ **do**
1: Calculate μ_i by formula (15) and then $\hat{U}_{i+1}$ by (14); Regulate $\hat{U}_{i+1}$ into output range;
2: **if** $||\mu_i|| == 0$: **end loop**
3: Acquire output $\hat{Y}(\hat{U}_{i+1})$ of MFNN model with $\hat{U}_{i+1}$ and calculate η_{i+1} and $J(\hat{U}_{i+1})$;
4: Calculate ΔL by (17) and then the gain ratio ρ by (16);
5: **if** $\rho < 0.25$: **do** $\lambda = \lambda * 2$;
else if $\rho > 0.75$: **do** $\lambda = \lambda/3$;
6: $i = i + 1$. Update G_i from model and recalculate H_i;
7: **if** $||\hat{Y}(\hat{U}_i) - R(k)|| < \varepsilon$: **end loop**
8: **output** $\hat{U}_i$ as control voltage signal $U(k)$.

$U(k)$ outputted by the algorithm is a vector composed of series of signals for future n_p instants. But only $u(k)$ will be actually outputted at current operating point, and the process will be re-run when next instant comes.

Model error compensation. The model can never be totally accurate for real plant. Actually, there always exist mismatches between model predictions and actual PEA displacement. As a result, static errors will be accumulated in control progress. Hence, a compensation term is added to controller outputs, functioning the same way as the integral term in PID a controller

$$c(k) = \gamma \sum_{i}^{k} (r(i) - y(i)) \tag{18}$$

$$u_{\text{out}}(k) = u(k) + c(k) \tag{19}$$

where γ is the compensative factor, u_{out} is the formal output signal.

Neural feedforward controller

MPC optimizes predictive errors at every operating points but this repeated process is time-costly and the resulted signals will not help to improve control quality when similar situation occurs. Warm-start technique[25] is proposed to lessen the iterative optimization calculation, usually by setting initial value $u_0(k)$ equal to last output $u(k-1)$. However, this simple trick may not works well when reference trajectory changes too fast. Hence, we introduce another MFNN as feedforward controller (MFNN2), providing initial value for MPC, as Figure 3 shows.

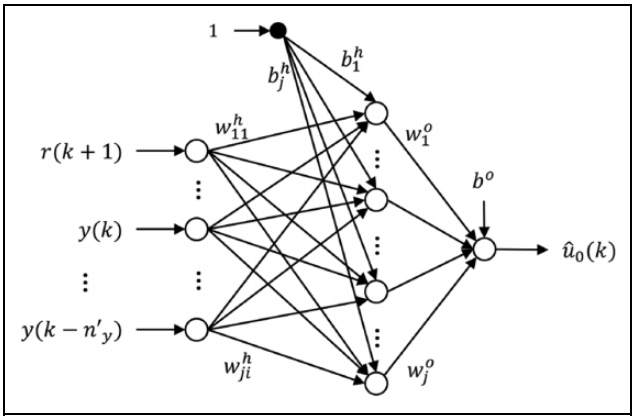

Figure 3. MFNN of neural controller. MFNN: multilayer feedforward neural network.

The set-point reference $r(k+1)$ indicates the desired displacement, and historical displacements $[y(k), .. y(k-n'_y)]$ contain information of current displacement and rate. Thus, the inputs provide enough information for the controller to determine proper guess about $\hat{u}(k)$.

MFNN2 has a single hidden layer, whose activation function $\sigma_1(\cdot)$ is hyperbolic tangent illustrated in (3). The activation function $\sigma_2(\cdot)$ of output layer is linear.

ADAM optimizer is chosen again to train MFNN2. The training can be both conducted off-line and online along with control progress, with small batch of training sets which are updated over time, absorbing newly proven samples and dropping outdated one. In this way, the performance of neural controller will be gradually improved over time before deployment.

With a well-trained neural feedforward controller, the original value of control signal can be close to the optimal one, which decreases iterations in optimizing MPC objective function to only one iteration. Combined with the proposed model which prevents repeating predictions, the computational cost is largely lowered from traditional NMPC. Assuming a traditional NMPC requires n_t iteration, then $(n_t \times n_p - 1)$ calls of model and $(n_t - 1)$ optimizations are cut down by proposed method at a small cost of computing a slightly larger neural model and an additional feedforward output.

Modification against feedback delays

The feedbacks in inefficient rate with delay causes mismatches between feedbacks and real displacement can greatly affect the control accuracy and stability, especially in dynamic tracking. When reference changes rapidly, the controlled displacement should also follows quickly, but delayed feedbacks fail to return correct displacement of PEA in time, making control errors larger than reality, which misleads controller to adjusting output even harder. As a result, large overshoot occurs when reference/displacement changes rapidly. All kinds of feedback controllers

are affected by this effect, where MPC is influenced in the prediction process, as NARMAX model requires current displacement as one of inputs. So if current displacement can be better estimated despite of mismatched feedback, the control can be improved based on better prediction. As we have a ready prediction $\hat{y}(k) = f(u(k-1))$ from MPC at previous instant, an estimation can be made according to both $\hat{y}(k)$ and $y_s(k)$ from sensor. Hence, an EKF[34] is introduced for this purpose.

The states functions of prediction and measurement are respectively written as

$$\hat{y}^-(k) = f(\hat{y}(k-1), u(k-1)) + w(k) \tag{20}$$

$$y_s(k) = h \cdot y(k) + v(k) \tag{21}$$

where $f(\hat{y}(k-1), u(k-1))$ is simplified from (2), as only $\hat{y}(k-1)$ and $u(k-1)$ matter now. h is the gain from real displacement $y(k)$ to sampled feedback. $w(k)$ and $v(k)$ are included errors for prediction and measurement, with covariance matrices as Q and R, respectively

$$\begin{cases} p(w(k)) \sim N(0, Q) \\ p(v(k)) \sim N(0, R) \end{cases} \tag{22}$$

From Jacobian G_η, the gradient of $\hat{y}^-(k)$ with respect to $\hat{y}(k-1)$ and $u(k-1)$ can be extracted as $g \in \mathbb{R}^{1\times 2}$. Then the Kalman filtering process can be updated as following

$$P^-(k) = gP(k-1)g^T + Q \tag{23}$$

$$K(k) = P^-(k)h^T(hP^-(k)h^T + R)^{-1} \tag{24}$$

$$P(k) = (I - K(k)h)P^-(k) \tag{25}$$

P is estimation covariance matrix, I is identity matrix, K is Kalman gain. In this case, however, Q, R, P, K, h are all in 1×1 dimension. Finally, the estimated displacement will be outputted as

$$\hat{y}(k) = \hat{y}^-(k) + K(k)(y_s(k) - h \cdot \hat{y}^-(k)) \tag{26}$$

Although the Q and R are defined as covariances of errors, they are not directly related to true errors in our case. They act as two adjustable parameters controlling this fusion process. As (26) shows, the estimation $\hat{y}(k)$ is actually a weighted mean of sensor feedback and model prediction. The purpose of this filter is increasing the weight of prediction, that is, increasing R/Q ratio, when measured displacement changes significantly slower than it should be, which is judged according to

$$\rho(k) = \frac{\Delta u^2}{\Delta y^2 + 0.1} = \frac{(u(k-1) - u(k-2))^2}{(y_s(k) - y_s(k-1))^2 + 0.1} \tag{27}$$

To ensure the coverage and smooth the shifts, the most recent n_ρ ratios will be recorded as $P_\rho = [\rho(k), ..\rho(k-n_\rho)]$, and if one of them is large enough, the situation should be regarded as a significant mismatch caused by feedback delay. Thus, the Q is fixed as a constant c_Q, and R is determined by $\max(P_\rho)$

$$R = c_{R1} + c_{R2} \cdot S(\max(P_\rho)) \tag{28}$$

c_Q, c_{R1}, and c_{R2} are nonnegative constants, $S(\cdot)$ is a sigmoid function

$$S(x) = \frac{1}{1 + e^{-a(x-b)}} \tag{29}$$

where a and b are nonnegative constants. To make it function well, c_Q, c_{R1}, c_{R2}, a, and b should be properly set. c_{R1} and c_{R2} roughly define the range of R as $(c_{R1}, c_{R1} + c_{R2})$, which, cooperated with c_Q, decides the range of R/Q ratio. b is a threshold to tell whether a $\max(P_\rho)$ is big enough for the situation to be regarded as a significant mismatch, that larger b means higher tolerance. a decides the sensitivity of (28), that is, larger a leads to quicker reactions to the changes of $\max(P_\rho)$. With suitable parameters, the $\hat{y}(k)$ after modification (26) will be close to model prediction when significant mismatch happens and be close to sensor feedback when mismatch is slight. The modified $\hat{y}(k)$ is more close to real $y(k)$ in dynamic occasion, thus when inputted to PEA model, it helps to get a better prediction of $\hat{y}(k+1)$.

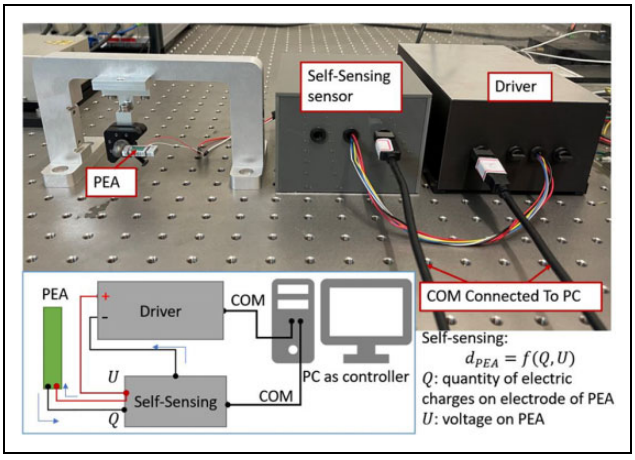

Figure 4. Experiment system. Note that one channel of self-sensing sensor is serially connected between the negative electrodes of PEA and driver. The driving loop is marked with arrows. PEA: piezoelectric actuator.

Experiments and discussion

Experiment setup

As shown in Figure 4, the experiment system are configured as following: a stack PEA produced by Thorlab is driven by a 0–60 V voltage driver with peak-to-peak ripples less than 10 mV. The output voltage can be regarded as linear to control signals. A self-sensing method-based device is used as displacement sensor for feedbacks. It has two channels. One is serially connected between PEA and

Table 1. Tracking performance comparison.

Reference	Controller	MAE (nm)	RMSE (nm)
0.5 Hz Sinusoid ≈ 500 points/ period	Proposed NMPC	3.8	5.3
	Basic NMPC	7.8	11.2
	PID	6.6	8.2
1 Hz Sinusoid ≈ 250 points/ period	Proposed NMPC	5.5	7.8
	Basic NMPC	12.3	18.5
	PID	13.4	15.7
2 Hz Sinusoid ≈ 125 points/ period	Proposed NMPC	9.8	15.0
	Basic NMPC	26.3	39.0
	PID	27.6	32.2
5 Hz Sinusoid ≈ 50 points/ period	Proposed NMPC	44.3	52.7
	Basic NMPC	74.5	102.8
	PID	74.5	88.2
1 Hz Triangular wave (linear test)	Proposed NMPC	5.6	8.4
	Basic NMPC	8.5	12.3
	PID	8.5	9.7

PID: proportional–integral–derivative; MAE: mean absolute error; RMSE: root mean square error.

negative end of power, measuring the charges on negative electrode of PEA; another is parallel connected with PEA measuring its voltage. With self-sensing principle derived from piezoelectric formula, the device can acquire displacement of PEA indirectly from quantity of electric charges and driving voltage.[35] The control algorithm runs in a PC as controller, communicating with driver and sensor both through COM connections. The sampling rate of the sensor is about 250 Hz whereas the driver has higher maximal output rate. The feedbacks received by controller usually have a delay of one sampling period, which causes mismatches between feedbacks and reality, worsening the control performance in tracking task. In such condition the proposed method is tested and its effectiveness is verified.

Verification of proposed control method

A series of experiments are conducted to evaluate the effectiveness of proposed method, compared with PID and a basic NMPC.[18] For MPC and PID both, there are tradeoffs between performances in tracking of high and low frequencies. Thus, before following experiments, parameters of all controllers are adjusted to achieve balanced performances in sinusoidal tracking of multiple frequencies and in step response. Then all following experiments are conducted under a same set of parameters, which may not make the best performance for specific tasks, but focus on the overall performance.

Firstly, tracking experiments under sinusoidal reference trajectory of different frequencies (0.5, 1, 2, and 5 Hz) are conducted. The results are shown in Table 1 with mean absolute error (MAE) and root mean square error (RMSE) indicating accuracy. Figure 5 also shows performance comparisons in 1, 2, and 5 Hz. The time-displacement charts present zoomed waveforms of displacement changing, and time-error charts show the error recordings during the same periods. When the frequency of desired trajectory goes high, as the amount of control steps for each sinusoidal period decreases, accuracy is deteriorating for all three methods, but proposed controller excels PID/basic NMPC in all tested cases. When sinusoidal trajectory of preference has frequency lower than 2 Hz, control accuracy of proposed controller only slightly deteriorates when desired frequency doubles. However, when desired frequency reaches 5 Hz, the accuracy suffers a dramatic deterioration as control steps become very sparse for each period, but proposed controller still have much better accuracy than PID and basic NMPC. In Figure 5, we can find out different error characteristics between the methods: errors of PID are mainly caused by lag, and the basic NMPC reacts poorly against sudden changes of delay, which causes oscillations, although it has smaller errors than PID when not oscillating, whereas the proposed controller only has large errors around peaks and bottoms of sinusoid, mostly because the sparse and rapidly varying feedbacks become less predictable for model to provide accurate predictions, on which the effective control is based. It can be noticed in error charts that some large errors occur. This is caused when feedbacks suddenly change the duration of delays which results in large mismatches (at the arrowhead in (e) and (f) of Figure 5) and then makes controllers overreact. The proposed controller achieves much better performance than basic NMPC in quelling aftereffects and restrains maximal errors to a smaller extent than PID does. This characteristic is more obvious in step response comparison, as shown in Figure 6.

In Figure 6, a 2 μm step trajectory is set. Proposed controller is compared with PID, basic NMPC and the proposed NMPC without aforementioned delay modification. All the controllers get one-step delayed feedbacks, which means, at the first sampling instant after controllers reacting to the steeply rising reference, the controllers get the feedback showing displacement does not change, although the real displacement has been hugely increased. This mismatch between feedback and reality misleads controllers to continuing to steeply raise output voltage, which causes big overshoots. The overshoot of proposed controller is much smaller than other three controllers, due to the EKF-based modification against feedback delays.

Tracking of mixed sinusoidal reference with random frequency (0.5–5 Hz) and random amplitude are also tested, presented by Figure 7, which shows the proposed controller is effective in random reference tracking. And Figure 8 shows its performance in linear tracking, tested by 1 Hz triangular reference trajectory. Though proposed controller suffers some slight oscillation right after the change of motion direction, it still achieves smaller MAE and RMSE than other two controllers. The results shows that proposed method is less affected by low feedback rate which makes pure-feedback controllers produce large lags.

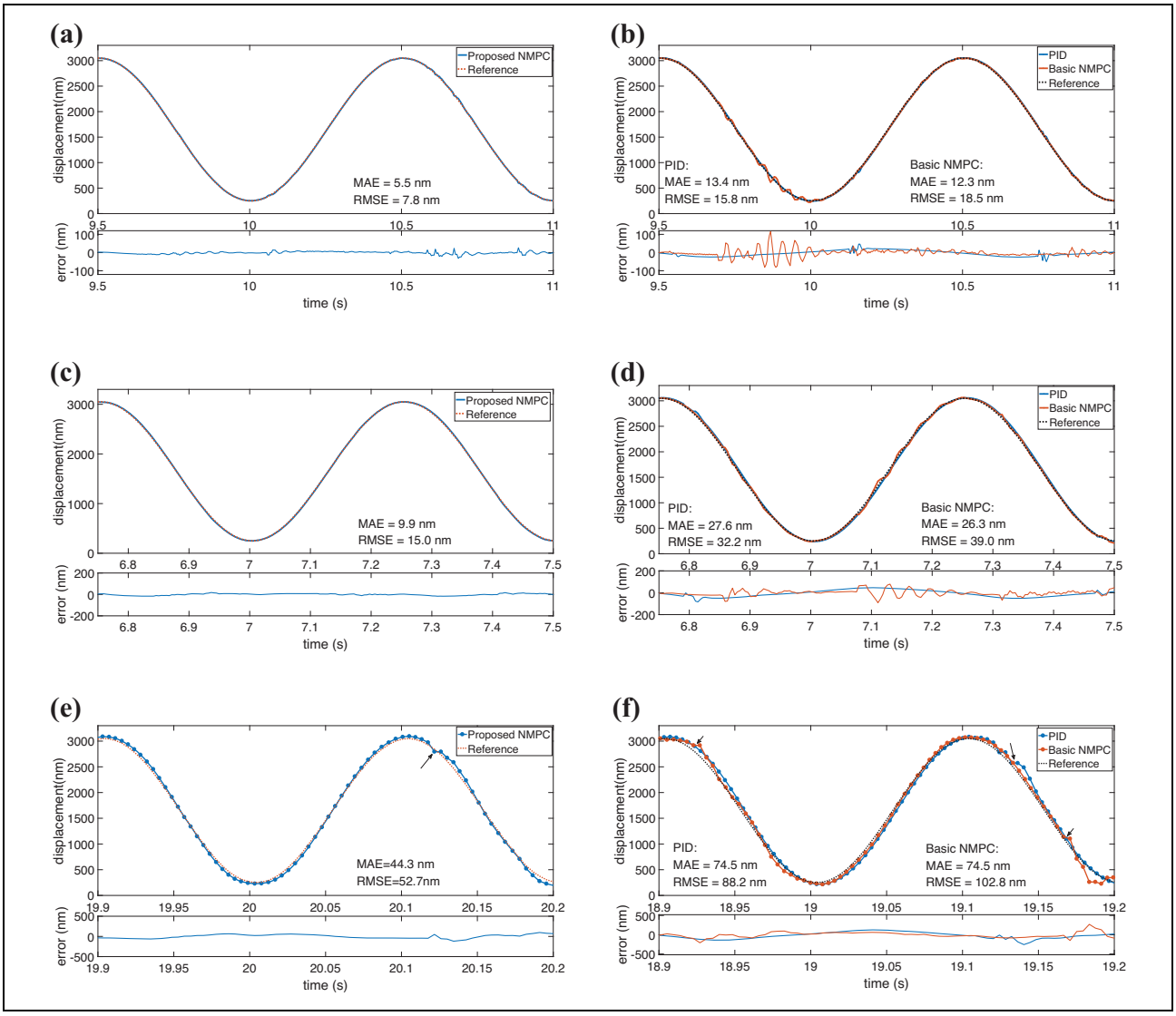

Figure 5. Sinusoidal tracking performance comparison: (a), (c), and (e) show control performance of proposed controller in 1, 2, and 5 Hz sinusoidal tracking, respectively, compared by PID and basic NMPC in (b), (d), and (f) with same reference trajectories. Arrowheads in (e) and (f) indicate the influence of duration changes of feedback delay. Note that MAEs and RMSEs marked in figures are statistics of the entire experiment containing many sinusoidal periods. PID: proportional–integral–derivative; MAE: mean absolute error; RMSE: root mean square error.

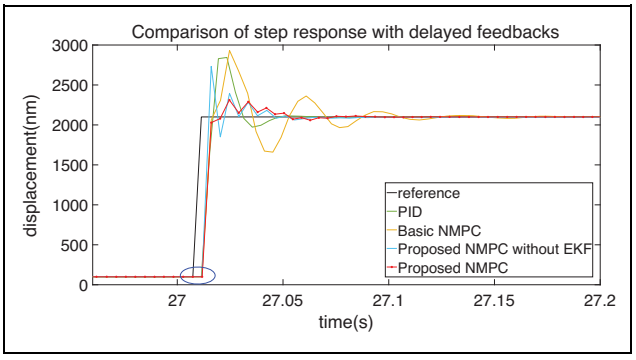

Figure 6. Step response comparison. Circled part indicates the feedback delay where real displacement has been changed but not be sampled yet.

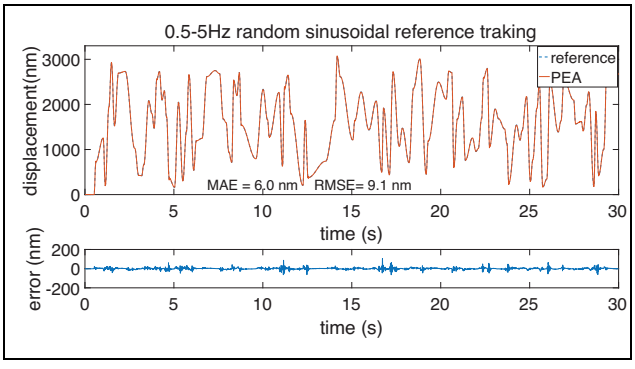

Figure 7. Random sinusoidal tracking by proposed controller.

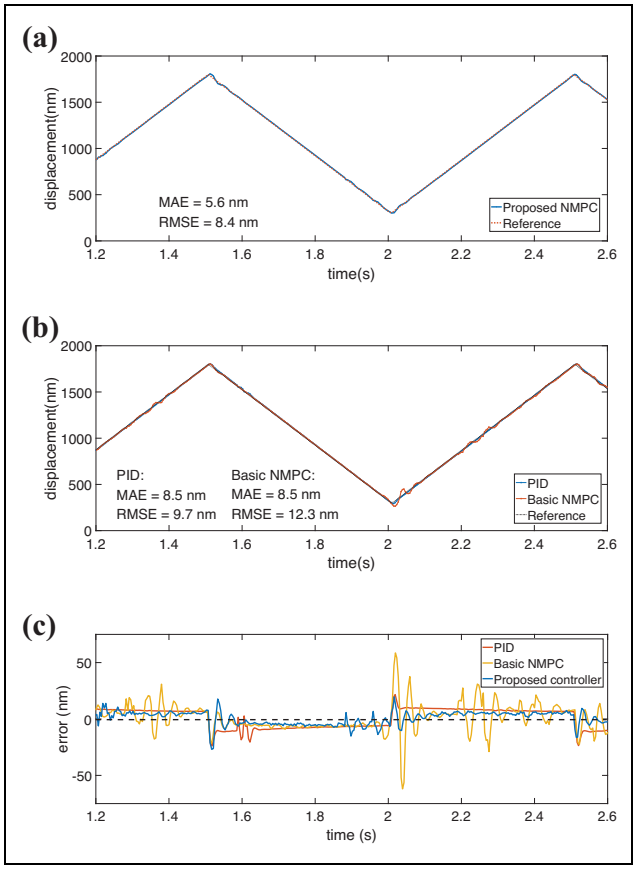

Figure 8. Linear (triangular) reference tracking. (a) Linear tracking performance of proposed controller. (b) Linear tracking performance of PID and basic NMPC. (c) Error comparison between proposed controller, PID and basic NMPC. PID: proportional–integral–derivative.

As the results of above experiments show, with sparse control points, the basic NMPC losses its advantage over PID. The effectiveness of proposed controller in displacement tracking is verified with generally significant advantages over PID and basic NMPC. And the approaches taken to relieve the feedback delay problem have also been validated as effective.

Conclusion

Hysteresis and dynamic-related nonlinearity are challenging problems in displacement tracking control of PEA. The displacement feedback with delay and low frequency causes mismatch between feedback and real state, increasing the difficulty of precise tracking control for pure feedback approaches. In this article, a neural network-based model predictive controller is introduced for tracking control, combined with neural feedforward controller and EKF-based feedback estimation. The proposed method achieves much better performances than basic NMPC and PID controller in tracking experiments, proving its effectiveness in handling the nonlinearity and feedback delay. To achieve further improvements in tracking performance, especially for high-frequency trajectory, more studies are to be conducted. Potential solutions include combining current NARMAX model with more explicit expression of motion trend, which may improve model's ability of providing better prediction and gradient estimation. The proposed controller is also to be applied to systems with higher feedback rate so that tracking the trajectory of higher frequency can be possible. Furthermore, with MPC's ability of multi-goal optimization, current controller has potential to be developed into a space trajectory tracking controller for multi-axis piezoelectric positioning system.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported in part by the National Natural Science Foundation of China under Grant No. 61873268 and No. 62033013, also supported by Strategic Priority Research Program of Chinese Academy of Sciences, Grant No. XDB32050100.

ORCID iD

Zhangming Du https://orcid.org/0000-0003-4324-2109

References

1. Zhou C, Gong Z, Chen BK, et al. A closed-loop controlled nanomanipulation system for probing nanostructures inside scanning electron microscopes. *IEEE/ASME Trans Mechatron* 2016; 21(3): 1233–1241.
2. Ru C, Zhang Y, Sun Y, et al. Automated four-point probe measurement of nanowires inside a scanning electron microscope. *IEEE Trans Nanotechnol* 2011; 10(4): 674–681.
3. Jaffe B. *Piezoelectric ceramics*. 1st ed. London, UK: Academic Press, 1971.
4. Damjanovic D. Hysteresis in piezoelectric and ferroelectric materials. In Mayergoyz I and Bertotti G (eds) *The science of hysteresis*, vol. 3. Amsterdam: Elsevier, 2005, pp. 337–465.
5. Jung H and Gweon DG. Creep characteristics of piezoelectric actuators. *Rev Sci Instrum* 2000; 71(4): 1896–1900.
6. Ang WT, Garmón FA, Khosla PK, et al. Modeling rate-dependent hysteresis in piezoelectric actuators. In: *Proceedings 2003 IEEE/RSJ international conference on intelligent robots and systems*, 27–31 October 2003, vol. 2. Las Vegas, NV, USA, IEEE, pp. 1975–1980.
7. Devasia S, Eleftheriou E, and Moheimani S. A survey of control issues in nanopositioning. *IEEE Trans Control Syst Technol* 2007; 15(5): 802–823.
8. Chi Z and Xu Q. Recent advances in the control of piezoelectric actuators. *Int J Adv Robot Syst* 2014; 11(11): 182.

9. Oh J and Bernstein DS. Semilinear Duhem model for rate-independent and rate-dependent hysteresis. *IEEE Trans Autom Control* 2005; 50(5): 631–645.
10. Rakotondrabe M. Bouc–Wen modeling and inverse multiplicative structure to compensate hysteresis nonlinearity in piezoelectric actuators. *IEEE Trans Autom Sci Eng* 2010; 8(2): 428–431.
11. Zsurzsan TG, Andersen MA, Zhang Z, et al. Preisach model of hysteresis for the piezoelectric actuator drive. In: *IECON 2015-41st annual conference of the IEEE Industrial Electronics Society*, 9–12 November 2015. Yokohama, Japan, IEEE, pp. 002788–002793.
12. Gu GY, Zhu LM, and Su CY. Modeling and compensation of asymmetric hysteresis nonlinearity for piezoceramic actuators with a modified Prandtl–Ishlinskii model. *IEEE Trans Ind Electron* 2013; 61(3): 1583–1595.
13. Changhai R and Lining S. Hysteresis and creep compensation for piezoelectric actuator in open-loop operation. *Sens Actuators A* 2005; 122(1): 124–130.
14. Cheng L, Liu W, Hou ZG, et al. An adaptive Takagi–Sugeno fuzzy model-based predictive controller for piezoelectric actuators. *IEEE Trans Ind Electron* 2017; 64(4): 3048–3058.
15. Cao Y and Chen X. A novel discrete ARMA-based model for piezoelectric actuator hysteresis. *IEEE/ASME Trans Mechatron* 2011; 17(4): 737–744.
16. Wen CM and Cheng MY. An adaptive Takagi–Sugeno fuzzy model-based predictive controller for piezoelectric actuators. *IEEE Trans Ind Electron* 2013; 60(11): 5105–5115.
17. Liaw CH, Shirinzadeh B, and Smith J. Robust neural network motion tracking control of piezoelectric actuation systems for micro/nanomanipulation. *IEEE Trans Neural Netw* 2009; 20(2): 356–367.
18. Cheng L, Liu W, Hou ZG, et al. Neural-network-based nonlinear model predictive control for piezoelectric actuators. *IEEE Trans Ind Electron* 2015; 62(12): 7717–7727.
19. Ge P and Jouaneh M. Tracking control of a piezoceramic actuator. *IEEE Trans Control Syst Technol* 1996; 4(3): 209–216.
20. Al Janaideh M, Rakheja S, and Su CY. An analytical generalized Prandtl-Ishlinskii model inversion for hysteresis compensation in micropositioning control. *IEEE/ASME Trans Mechatron* 2011; 16(4): 734–744.
21. Song B, Sun Z, Xi N, et al. Enhanced nonvector space approach for nanoscale motion control. *IEEE Trans Nanotechnol* 2018; 17(5): 994–1005.
22. Li P, Li P, and Sui Y. Adaptive fuzzy hysteresis internal model tracking control of piezoelectric actuators with nanoscale application. *IEEE Trans Fuzzy Syst* 2016; 24(5): 1246–1254.
23. Jian Y, Huang D, Jiabin L, et al. High-precision tracking of piezoelectric actuator using iterative learning control and direct inverse compensation of hysteresis. *IEEE Trans Ind Electron* 2019; 66(1): 368–377.
24. Al Janaideh M, Rakotondrabe M, Al-Darabsah I, et al. Internal model-based feedback control design for inversion-free feedforward rate-dependent hysteresis compensation of piezoelectric cantilever actuator. *Control Eng Practice* 2018; 72: 29–41.
25. Lawrynczuk M.*Computationally efficient model predictive control algorithms: a neural network approach*. New York, NY: Springer-Verlag, 2014.
26. Wang J, Chen L, and Xu Q. Disturbance estimation-based robust model predictive position tracking control for magnetic levitation system. *IEEE/ASME Trans Mechatron*. DOI: 10.1109/TMECH.2021.3058256.
27. Liu W, Cheng L, Hou ZG, et al. An inversion-free predictive controller for piezoelectric actuators based on a dynamic linearized neural network model. *IEEE/ASME Trans Mechatron* 2016; 21(1): 214–226.
28. Xie S and Ren J. Linearization of recurrent-neural-network-based models for predictive control of nano-positioning systems using data-driven Koopman operators. *IEEE Access* 2020; 8: 147077–147088.
29. Liu J, de la Peña DM, and Christofides PD. Distributed model predictive control of nonlinear systems subject to asynchronous and delayed measurements. *Automatica* 2010; 46(1): 52–61.
30. Sirouspour S and Shahdi A. Model predictive control for transparent teleoperation under communication time delay. *IEEE Trans Robot* 2006; 22(6): 1131–1145.
31. Gao Y and Meng JE. NARMAX time series model prediction: feedforward and recurrent fuzzy neural network approaches. *Fuzzy sets syst* 2005; 150(2): 331–350.
32. Hornik K, Stinchcombe M, and White H. Multilayer feedforward networks are universal approximators. *Neural Netw* 1989; 2(5): 359–366.
33. Kingma DP and Ba JL. Adam: a method for stochastic optimization. In: *Proceedings of the international conference on learning representations* 2015, San Diego, CA, USA.
34. Bishop G and Welch G. An introduction to the Kalman filter. *Proc of SIGGRAPH, Course* 2001; 8(27599-23175): 41.
35. Du Z, Zhou C, Zhang T, et al. A measuring method for nano displacement based on fusing data of self-sensing and time-digit-conversion. *IEEE Access* 2019; 7: 183070–183080.