Recurrent-Neural-Network-Based Predictive Control of Piezo Actuators for Trajectory Tracking

Shengwen Xie and Juan Ren

***Abstract*—Precise trajectory tracking of piezo actuators (PEAs) in real time is essential to high-precision systems and applications. However, the real-time tracking accuracy is rather limited as the PEA cannot be accurately modeled over large bandwidth and displacement range due to its nonlinearities. In this article, we propose to use recurrent-neural-network (RNN) to model the PEA system and develop a nonlinear predictive controller for PEA trajectory tracking. Considering the computation efficiency, first, an RNN is trained to model the nonlinear dynamics of the PEA system at high-frequency range. Then, a second-order linear model is proposed to account for the PEA low-frequency dynamics. Therefore, the PEA dynamics is modeled by the nonlinear model consisting of the RNN and the linear model, which is further used for nonlinear predictive control of the displacement. To increase the prediction accuracy, an unscented Kalman filter is designed to estimate the states of the nonlinear model. The nonlinear predictive control problem is solved based on a gradient descent algorithm, in which a method for analytically calculating the gradient of the cost function is developed. The proposed technique was experimentally implemented on a nano piezo stage for demonstration and its performance was compared with that of a PID controller. The accuracy of an iterative learning control approach was used as a benchmark for comparison as well. The results showed that high precision trajectory tracking of PEAs in real time can be achieved using the proposed technique.**

***Index Terms*—Nonlinear predictive control, output tracking, recurrent neural network (RNN).**

I. Introduction

OWING to the fast response and mechanical stability, piezo actuators (PEAs) have been broadly used in high-precision systems and applications [1], such as scanning probe microscope [2], [3], microforming [4], and adaptive optics [5]. The tracking accuracy is critical for these applications. For example, high precision tracking control of PEAs is needed to realize compliance inside the robot's structure [6]. Also, the performance of atomic force microscope (AFM) will be greatly downgraded if the AFM scanner PEAs fail to track the sample surface profile in topography imaging and/or the AFM probe-sample interaction force cannot track the predefined excitation force accurately in AFM mechanical characterization [7]–[9]. However, it is not trivial to achieve precision control of PEAs in real time, especially when operated at high speed due to the system nonlinearities, such as creep effect and hysteresis [1]. Significant efforts have been made to address this challenge. Iterative learning control (ILC) and repetitive control are very effective in precision output (i.e., trajectory) tracking when the tasks are repetitive [10]–[12]. Recently, ILC algorithms aiming for tracking varying trajectories are presented as well [8], [13], [14]. Although high-precision trajectory control can be realized using ILC-based approaches, they are not suitable for real-time control of PEAs as the convergence of these approaches can only be reached through iterations [8], [13], [14]. Real-time output tracking (i.e., trajectory tracking without iterations) of PEAs still remains challenging. Real-time control techniques have been developed for output tracking of PEAs. For example, the dynamics of a PEA was identified with a linear model and controlled with a model predictive controller (MPC) [15]. However, as the operation frequency increases, effect of the system nonlinearities (such as the creep effect and hysteresis) becomes more pronounced which directly results in significantly increased trajectory tracking error. Sliding mode control (SMC) based on linear model of PEAs has been developed as well [16]. Compared to MPC, SMC is more robust to the modeling uncertainties and disturbances. However, the control bandwidth of SMC is quite limited due to the system nonlinearities [16]. Moreover, SMC can also cause the problem of chattering [17]. Therefore, real-time PEA control methods based on linear models are limited as well, and the system nonlinearities and uncertainties must be taken into account.

New approaches have been developed to address the control issues raised by the nonlinearities and disturbances (e.g., creep effect, hysteresis, and mechanical vibration). A feedback-linearized inverse feedforward approach to control the PEAs in the atomic force microscope has been proposed [18], where the hysteresis and creep effects are accounted for using the high-gain feedback control, and the mechanical vibration is addressed using feedforward control [18]. Moreover, at higher operation speed, an inversion model is used to improve the accuracy. As a result, the tracking performance is significantly

Manuscript received December 21, 2018; revised April 17, 2019 and August 12, 2019; accepted September 30, 2019. Date of publication October 9, 2019; date of current version December 31, 2019. Recommended by Technical Editor X. Chen. This work was supported by the National Science Foundation under Grant CMMI-1634592 and Grant CMMI-1751503. *(Corresponding author: Juan Ren.)*

The authors are with the Department of Mechanical Engineering, Iowa State University, Ames, IA 50011 USA (e-mail: swxie@iastate.edu; juanren@iastate.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMECH.2019.2946344

affected by the accuracy of the inversion model [18]. Based on the physical behavior of PEAs, a lot of models are proposed to model the hysteresis and creep effects, such as Domain wall model, Duhem model, and Prandtl–Ishlinskii model [19]–[22]. These models can be used to obtain an inversion model of PEAs which may help to eliminate the effects of system nonlinearities and then real-time control algorithms such as MPC, feedback control, and SMC can be applied [23]–[25]. However, the performances of these inversion model-based approaches are directly affected by the modeling accuracy. Considering the modeling uncertainty, robust control tools, such as H_∞ and adaptive control are developed as well [26], [27]. However, the control bandwidth is rather limited due to the stability issue [26], [27].

Recently, neural networks have been proposed in output tracking applications of PEAs [28], [29]. For example, the feedforward neural network (FNN) has been proposed to model the dynamics of PEAs [28], [29]. However, one issue with the FNN is that the input to FNN is not treated as time series in the training process although the sequence of input can affect the behaviors of PEAs greatly [28]. Other types of neural networks like wavelet neural network [30], neural network with radial basis [31], [32], and recurrent neural network with residue compensation [33] are also proposed for PEA control. In addition, predictors based on neural networks are also proposed to estimate the nonlinear dynamics of time-delay-free dynamic system [34]. However, the bandwidth of the modeled dynamics is restricted to less than 100 Hz and no efficient methods of generating the data set for the neural network training over a large bandwidth have been presented. In contrast to FNN, an RNN is designed to deal with time series [35]. One advantage of the RNN is that it can be represented using nonlinear state space models, thus many nonlinear control tools can be then adopted [35]. Moreover, the RNN has been proved to be a universal approximator in modeling dynamical systems [36]. Therefore, in this article, we propose an RNN-based predictive control framework (RNN+LME) to achieve accurate output tracking of PEAs. Specifically, an RNN is trained to accurately capture the nonlinear dynamics of the PEA system. However, due to the limited length of the RNN training set and available computation resources in real applications, the PEA low frequency behaviors may not be fully captured. Thus, a second-order linear model embedded with an error term (LME) is proposed for modeling the residual dynamics (i.e., low frequency dynamics) [37]. Then a nonlinear predictive controller and an unscented Kalman filter (UKF) are designed to work with this PEA dynamic model (i.e., the nonlinear model consisting of the RNN and LME) to achieve precise output tracking. The proposed technique was experimentally applied on a nano piezo stage for demonstration and its performance was compared with that of a PID controller and an ILC approach.

The main contribution of this article is the development of RNN+LME predictive control framework. First, generation of data for system identification is an open problem, effective approaches for data generation of PEA systems have not been reported yet. In this article, we propose an effective method for generating the training set data by considering both the amplitude and frequency domains. Second, to remedy the computation inefficiency in modeling low-frequency dynamics of using RNN, LME is incorporated. Third, a practical solver is proposed for the nonlinear predictive control based on the gradient descent algorithm. As the RNN+LME framework does not assume any form of the nonlinearities, the integration of RNN and LME provides a flexible framework for solving a series of output tracking problem.

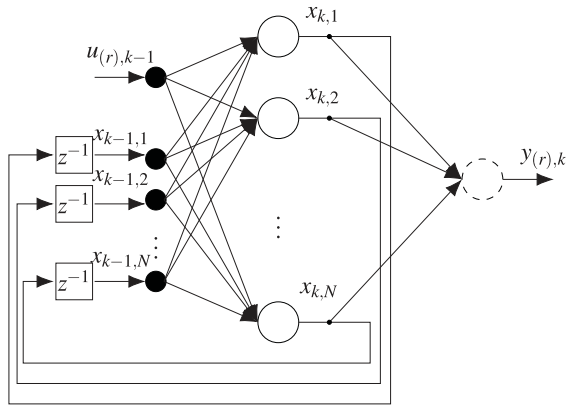


Fig. 1. Recurrent neural network.

II. System Identification

A. Recurrent Neural Network (RNN) Structure

The RNN used in this article, consisting of an input layer (solid circles), a hidden layer (circles), and an output layer (dashed circles), as shown in Fig. 1. $u_{(r),k}$ and $y_{(r),k}$ denote the input $u_{(r)}$ and output $y_{(r)}$ at the sampling instant k, respectively. $x_k = [x_{k,1}, x_{k,2}, \ldots, x_{k,N}]^T$ is the state vector of the RNN system. The activation functions of the hidden layer $g(x)$ and output layer $h(x)$ are designed as $g(x) = \tanh(x)$ and $h(x) = Wx + b$, respectively, where W is a 1-by-N matrix. Thus, the RNN in Fig. 1 can be represented by the following nonlinear state-space equation as

$$\begin{aligned} x_{k+1} &= \tanh(W_1 x_k + B_2 + B_1 u_{(r),k}) \\ y_{(r),k} &= W_2 x_k + B_3 \end{aligned} \tag{1}$$

where the dimensions of W_1, B_2, B_1, W_2, and B_3 are $N \times N$, $N \times 1$, $N \times 1$, $1 \times N$, and 1×1, respectively. Unlike the FNN which is essentially a "nonlinear autoregressive-moving-average with exogenous inputs" model and takes the past inputs and outputs as inputs to the network, the RNN only needs the current input $u_{(r),k}$ to generate the output signal [28]. Suppose $U_{(ts)}$ is any given time series (i.e., drive voltage) and $Y_{(ts)}$ is the corresponding output time series (i.e., displacement) of a PEA system, the output of the RNN, $Y_{(rts)}$, subject to the input $U_{(ts)}$ should equal $Y_{(ts)}$ if the RNN can accurately model the PEA system dynamics, i.e., $||Y_{(ts)} - Y_{(rts)}|| < \epsilon$ for any $\epsilon > 0$. Therefore, the RNN can be trained [i.e., to obtain the parameters W_1, B_2, B_1, W_2, and B_3 in (1)] by solving the following optimization problem using a predesigned time series input $U_{(ts)}$

and the corresponding PEA system output $Y_{(ts)}$ measured.

$$\begin{aligned} \min_{W_1,B_2,B_1,W_2,B_3} \quad & J_{(r)} = ||Y_{(ts)} - Y_{(rts)}|| \\ \text{subject to :} \quad & x_{k+1} = \tanh(W_1 x_k + B_2 + B_1 U_{(ts),k}) \\ & Y_{(rts),k} = W_2 x_k + B_3 \\ & x_0 = [0, 0, \ldots, 0]^T, k = 1, 2, 3, \ldots, L \end{aligned} \quad (2)$$

where $U_{(ts)} = [U_{(ts),1}, U_{(ts),2}, \ldots, U_{(ts),L}]^T$, $Y_{(ts)} = [Y_{(ts),1}, Y_{(ts),2}, \ldots, Y_{(ts),L}]^T$ and $Y_{(rts)} = [Y_{(rts),1}, Y_{(rts),2}, \ldots, Y_{(rts),L}]^T$. L is the length of the time series. Next, we present how to design $U_{(ts)}$.

B. Training Set Construction for RNN

There are various ways to construct $U_{(ts)}$. Here, we choose sinusoidal signals as building blocks to form $U_{(ts)}$. Define $S(f, A) = A(\sin(2\pi f t + \frac{3\pi}{2}) + 1), t \in [0, 1/f]$, i.e., $S(f, A)$ is a sinusoidal signal in one period with amplitude A and frequency f. In practice, t will be sampled, thus $S(A, f)$ is a time series. Then $U_{(ts)}$ can be written as

$$U_{(ts)} = \bigcup_{(f_i, A_i) \in \Omega} S(f_i, A_i) \quad (3)$$

where Ω is a set consisting of (f_i, A_i) pairs, and $\bigcup$ denotes concatenation. Therefore, each (f_i, A_i) pair represents a point in the $f - A$ plane. Suppose $U_{(ts)}$ consists of N_1 sinusoidal signals with $f_i \in [0, \bar{f}]$ and $A_i \in [0, \bar{A}]$. Ideally, the optimal training set should consist of all the (f_i, A_i) pairs in the wanted ranges, however this is impractical. Here we expect to find N_1 (f_i, A_i) pairs to achieve the highest modeling accuracy (i.e., a suboptimal approach). Finding the suboptimal $U_{(ts)}$ is equivalent to locating N_1 points in the $f - A$ plane such that any point (f_j, A_j) in the plane can be represented by the nearest point (f_k, A_k) (one of the N_1 points) and the distance between (f_j, A_j) and (f_k, A_k) is minimized. The solution to selecting the N_1 (f_i, A_i) points can be obtained using the k-means algorithm [38].

In practice, we can randomly generate a large number of points in the $f - A$ plane to cover all the possibilities as the behaviors of PEAs is frequency and displacement range dependent. Note that this method of choosing $U_{(ts)}$ is not trivial in the sense that the weights of f and A can vary. For example, if frequency is more important for the PEA tracking tasks, we can scale the f axis by multiplying a factor α to f, thus the signal with frequency q and amplitude 1 corresponds to $(q\alpha, 1)$ instead of $(q, 1)$ in the $f - A$ plane. Moreover, some frequency components are better to be avoided (such as the resonance frequency of the piezo actuator) when designing $U_{(ts)}$ by removing the points with f_is close to them, which will be explained with an example in the experiment part.

C. Linear Model Embedded With an Error Term (LME)

It is worthwhile to note that for $S(f, A)$, the smaller f is, the longer the time series $S(f, A)$ will be, e.g., f_s/f sampling points are needed to cover the entire period of this sinusoidal signal, where f_s is the sampling frequency. From (2), it can be seen that increase of the time series length leads to the increase of the number of constraints, which implies that either a more complex RNN is needed or the modeling accuracy will decrease. Therefore, long time series should be avoided considering the modeling accuracy and the computation efficiency of (2). On the other hand, even if a lot of high frequency sinusoidal signals are to be included in $U_{(ts)}$, the length of $U_{(ts)}$ will not be affected too much. Therefore, the above method of constructing $U_{(ts)}$ may result in that the low-frequency dynamics of PEAs cannot be entirely captured. In addition, the drift effect of PEAs (usually in very low frequency range) is hard to be modeled with RNN alone. Therefore, to address this issue, we proposed to use the following linear model embedded with an error term (LME) to deal with the residual dynamics (i.e., the PEA dynamics not captured by RNN) including the low-frequency dynamics and drift (see Fig. 2)

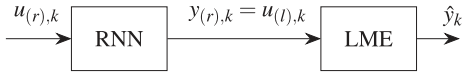


Fig. 2. Block diagram of RNN+LME for modeling the PEA system.

$$\begin{aligned} \eta_{k+1} &= A_e \eta_k + B_e u_{(l),k} + G\hat{e}_k \\ \hat{y}_k &= C_e \eta_k \end{aligned} \quad (4)$$

where $u_{(l),k}$ is the input to the LME, also the RNN output (see Fig. 2), $\hat{e}_k = y_k - \hat{y}_k$ is the model output error with y_k the PEA system output [37]. The error term can act as a feedback term. The dimensions of A_e, B_e, G, and C_e are 2×2, 2×1, 2×1, and 1×2, respectively. Suppose the output of the RNN [with parameters solved from (2)] is $Y_{(rts)}$ subject to the input $U_{(ts)}$, then it is expected that the output of the LME $\hat{Y}_{(ts)}$ will be equal to $Y_{(ts)}$ (i.e., the actual output of the PEA) subject to the input $Y_{(rts)}$, thus we can minimize the difference $||Y_{(ts)} - \hat{Y}_{(ts)}||$ to obtain the LME parameters. Similar to the training of the RNN, the parameters in LME will be determined through solving the following optimization problem with $Y_{(rts)}$ and $Y_{(ts)}$

$$\begin{aligned} \min_{A_e,B_e,G,C_e} \quad & J_1 = ||Y_{(ts)} - \hat{Y}_{(ts)}|| \\ \text{subject to :} \quad & \eta_{k+1} = A_e \eta_k + B_e Y_{(rts),k} + G\hat{e}_k \\ & \hat{Y}_{(ts),k+1} = C_e \eta_{k+1} \\ & \hat{e}_{k+1} = Y_{(ts),k+1} - \hat{Y}_{(ts),k+1} \\ & \eta(0) = [0, 0]^T \\ & \hat{e}_0 = 0, k = 0, 1, \cdots \end{aligned} \quad (5)$$

where $\hat{Y}_{(ts)} = [\hat{Y}_{(ts),1}, \hat{Y}_{(ts),2}, \ldots, \hat{Y}_{(ts),L}]^T$. Before training, $Y_{(rts)}$ and $Y_{(ts)}$ are known and $Y_{(ts)}$ is the same as that in (2).

D. Combine RNN and LME

Since the modeled dynamics in the RNN and the LME is limited by the frequency range of the training set, a low pass filter (LPF) is connected to the PEA model (i.e., RNN+LME) to avoid instability induced by ultra-high frequency dynamics and

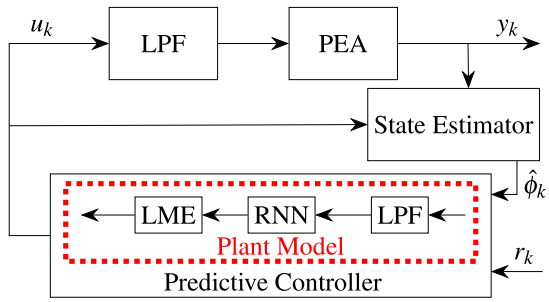


Fig. 3. Block diagram of the proposed RNN+LME. r_k is the reference signal (i.e., the desired trajectory).

disturbances. Suppose the LPF is formulated as

$$\begin{aligned} \beta_{k+1} &= \bar{A}\beta_k + \bar{B}\hat{u}_k \\ z_k &= \bar{C}\beta_k. \end{aligned} \tag{6}$$

With $u_{(r),k} = z_k$ and $u_{(l),k} = y_{(r),k}$, (1), (4), and (6) can be combined as the following nonlinear model:

$$\begin{aligned} \phi_{k+1} = \begin{bmatrix} \beta_{k+1} \\ x_{k+1} \\ \eta_{k+1} \end{bmatrix} &= F(\phi_k, \hat{u}_k) = F\left(\begin{bmatrix} \beta_k \\ x_k \\ \eta_k \end{bmatrix}, \hat{u}_k\right) \\ &= \begin{bmatrix} \bar{A}\beta_k + \bar{B}\hat{u}_k \\ \tanh(W_1 x_k + B_2 + B_1\bar{C}\beta_k) \\ A_e\eta_k + B_e W_2 x_k + B_e B_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} \hat{e}_k \\ \hat{y}_k = H(\phi_k) &= \begin{bmatrix} 0 & 0 & C_e \end{bmatrix} \begin{bmatrix} \beta_k \\ x_k \\ \eta_k \end{bmatrix} \end{aligned} \tag{7}$$

where $\hat{u}_k$ is the input to the nonlinear model. The block diagram of the model represented by (7) is shown in Fig. 3, i.e., the "Plant Model" in the dash box. Note that the dynamics of the LPF, such as phase delay, is already included in (6), and thus in (7).

III. Nonlinear Predictive Control

The control scheme of the predictive controller based on the above system model (7) is shown in Fig. 3.

A. Nonlinear Estimator

Since the system model is nonlinear and the states are unavailable through measurement, a state estimator—the UKF is designed to estimate the states during the tracking process in this article.

In the extended Kalman filter, the nonlinear dynamics function is linearized at the current point (first order linearization) in each estimation and the conventional Kalman filter algorithms can be implemented. However, the approximation error through linearization may increase if the nonlinearity at the linearization point is significant [39]. Instead, such an issue can be avoided in the UKF which generates a series of sample points propagated through the nonlinear dynamics function, then the covariance matrix can be computed from the sampling points [39]. For the system model (7), the error term $\hat{e}_k$ is calculated before each estimation and updated at each sample instant. The estimation steps for the UKF used in this article are as follows:

Step 1: Initialize the parameters at $k = 0$

$$\begin{aligned} \hat{\phi}_0 &= E[\phi_0] \\ P_0 &= E[(\phi_0 - \hat{\phi}_0)(\phi_0 - \hat{\phi}_0)]^T \\ \hat{e}_0^- &= 0. \end{aligned} \tag{8}$$

Step 2: Generate sampling points (sigma points)

$$\begin{aligned} \chi_{k-1} &= [\hat{\phi}_{k-1}\hat{\phi}_{k-1} + \gamma\sqrt{P_{k-1}}\hat{\phi}_{k-1} - \gamma\sqrt{P_{k-1}}] \\ k &= 1, 2, 3, \ldots. \end{aligned} \tag{9}$$

Step 3: Time update

$$\begin{aligned} \chi_{k|k-1} &= [F(\chi_{1,k-1}, u_{k-1}), \ldots, F(\chi_{2N+1,k-1}, u_{k-1})] \\ \hat{\phi}_k^- &= \sum_{i=1}^{2N+1} W_m(i)\chi_{i,k|k-1} \\ P_k^- &= \sum_{i=1}^{2N+1} W_c(i)[\chi_{i,k|k-1} - \hat{\phi}_k^-][\chi_{i,k|k-1} - \hat{\phi}_k^-]^T + R_v \\ \mathcal{Y}_{k|k-1} &= [H(\chi_{1,k|k-1}), \ldots, H(\chi_{2N+1,k|k-1})] \\ \hat{y}_k^- &= \sum_{i=1}^{2N+1} W_m(i)\mathcal{Y}_{i,k|k-1}. \end{aligned} \tag{10}$$

Step 4: Measurement update and calculation of the error term

$$\begin{aligned} P_{\delta_y\delta_y} &= \sum_{i=1}^{2N+1} W_c(i)[\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-]^T + R_n \\ P_{\delta_\phi\delta_y} &= \sum_{i=1}^{2N+1} W_c(i)[\chi_{i,k|k-1} - \hat{\phi}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-] \\ \mathcal{K}_k &= P_{\delta_\phi\delta_y} P_{\delta_y\delta_y}^{-1} \\ \hat{\phi}_k &= \hat{\phi}_k^- + \mathcal{K}_k(y_k - \hat{y}_k^-) \\ P_k &= P_k^- - \mathcal{K}_k P_{\delta_y\delta_y}\mathcal{K}_k^T \\ \hat{e}_k^- &= y_k - \hat{y}_k^-. \end{aligned} \tag{11}$$

Note that $\sqrt{P_{k-1}}$ is the square root of a matrix and can be calculated with Cholesky factorization, i.e., $P_{k-1} = \sqrt{P_{k-1}}\sqrt{P_{k-1}}^T$. R_v and R_n are the process and measurement noise covariance matrices, respectively. γ is a constant scalar, W_m and W_c are $(2N+1) \times 1$ vectors determined by the order of the system, N, $W_m(i)$ is the ith element of W_m. $\chi_{i,k|k-1}$ is the ith column of matrix $\chi_{k|k-1}$. The readers are referred to [39] for how to choose γ, W_m, and W_c.

B. Predictive Controller

For the nonlinear predictive control, the following optimization problem is to be solved at each sampling point and the PEA

system drive input u_k for the next sample time can be obtained from the solution

$$\min_U \mathscr{J} = (\hat{Y}^f - R^f)^T(\hat{Y}^f - R^f) + \rho {U^f}^T D^T D U^f$$

$$D = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ 0 & 1 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{12}$$

where $U^f = [u_k, u_{k+1}, \dots, u_{k+N_c-1}]^T$, $\hat{Y}^f = [\hat{y}_{k+1}, \hat{y}_{k+2}, \dots, \hat{y}_{k+N_h}]^T$ and $R^f = [r_{k+1}, r_{k+2}, \dots, r_{k+N_c}, \dots, r_{k+N_c}]^T_{N_h \times 1}$ are the future inputs, predicted outputs, and the desired trajectory), respectively, with N_h the prediction horizon and N_c the control horizon. Notice that the current state ϕ_k and the previous input u_{k-1} are known when solving (12). Here we do not consider the constraints to the input for simplicity, but they can be incorporated later. Since the system dynamics is nonlinear, the objective function $\mathscr{J}$ cannot be written in the quadratic form as in linear model predictive control [40]. As an example, we choose the gradient descent method to solve the unconstrained optimization problem [i.e., (12)].

Next, we show how to analytically compute the gradient $\frac{\partial \mathscr{J}}{\partial U}$ instead of using the numerical method as shown in (13). By using the analytical method, it is not only more precise without considering the increment Δ, but also more computationally efficient [41]. However, the following numerical method can be used to verify the analytic result:

$$\frac{\partial \mathscr{J}}{\partial u_{k+i}} = \lim_{\Delta \to 0} \frac{\mathscr{J}(u_{k+i} + \Delta) - \mathscr{J}(u_{k+i} - \Delta)}{2\Delta}. \tag{13}$$

Let $E = (\hat{Y}^f - R^f)^T(\hat{Y}^f - R^f)$, then the key to compute $\frac{\partial \mathscr{J}}{\partial U}$ is to compute $\frac{\partial E}{\partial U}$ since calculation of the derivative of the other term is trivial. E can be expressed as

$$\begin{aligned} E = \sum_{i=1}^{N_h} E_i^2 &= (\hat{y}_{k+1} - r_{k+1})^2 + (\hat{y}_{k+2} - r_{k+2})^2 \\ &+ \dots + (\hat{y}_{k+N_h} - r_{k+N_c})^2 = (C_e\eta_{k+1} - r_{k+1})^2 \\ &+ (C_e\eta_{k+2} - r_{k+2})^2 + \dots + (C_e\eta_{k+N_h} - r_{k+N_c})^2. \end{aligned} \tag{14}$$

Thus, $\frac{\partial E}{\partial {U^f}^T}$ can be written as

$$\begin{aligned} \frac{\partial E}{\partial {U^f}^T} &= \begin{bmatrix} \frac{\partial E}{\partial u_k} \\ \frac{\partial E}{\partial u_{k+1}} \\ \vdots \\ \frac{\partial E}{\partial u_{k+N_c-1}} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial E_1^2}{\partial u_k} + \frac{\partial E_2^2}{\partial u_k} + \frac{\partial E_3^2}{\partial u_k} + \dots + \frac{\partial E_{N_h}^2}{\partial u_k} \\ \frac{\partial E_2^2}{\partial u_{k+1}} + \frac{\partial E_3^2}{\partial u_{k+1}} + \dots + \frac{\partial E_{N_h}^2}{\partial u_{k+1}} \\ \vdots \\ \frac{\partial E_{N_c}^2}{\partial u_{k+N_c-1}} + \dots + \frac{\partial E_{N_h}^2}{\partial u_{k+N_c-1}} \end{bmatrix}. \end{aligned} \tag{15}$$

Next, we show how to compute $\frac{\partial E}{\partial u_k}$ and the rest follows. Note that $\phi_k = [\beta_k, x_k, \eta_k]^T$ and u_{k-1} are known at the sampling instant k. With (7), we have

$$\begin{aligned} \frac{\partial E_1^2}{\partial u_k} &= 2E_1C_e\frac{\partial \eta_{k+1}}{\partial u_k} \\ &= 2E_1C_e\frac{\partial(A_e\eta_k + B_eW_2x_k + B_eB_3)}{\partial u_k} = 0. \end{aligned} \tag{16}$$

Thus $\frac{\partial \eta_{k+1}}{\partial u_k} = 0$. Since

$$\frac{\partial x_{k+1}}{\partial u_k} = \frac{\partial \tanh(W_1x_k + B_2 + B_1\bar{C}\beta_k)}{\partial u_k} = 0 \tag{17}$$

then

$$\begin{aligned} \frac{\partial E_2^2}{\partial u_k} &= 2E_2C_e\frac{\partial \eta_{k+2}}{\partial u_k} \\ &= 2E_2C_e\frac{\partial(A_e\eta_{k+1} + B_eW_2x_{k+1} + B_eB_3)}{\partial u_k} = 0. \end{aligned} \tag{18}$$

Then $\frac{\partial \eta_{k+2}}{\partial u_k} = 0$. Similarly

$$\begin{aligned} \frac{\partial E_3^2}{\partial u_k} &= 2E_3C_e\frac{\partial \eta_{k+3}}{\partial u_k} \\ &= 2E_3C_e\frac{\partial(A_e\eta_{k+2} + B_eW_2x_{k+2} + B_eB_3)}{\partial u_k} \\ &= 2E_3C_eB_eW_2\frac{\partial x_{k+2}}{\partial u_k}. \end{aligned} \tag{19}$$

With $\beta_{k+1} = \bar{A}\beta_k + \bar{B}u_k$, we have

$$\begin{aligned} \frac{\partial x_{k+2}}{\partial u_k} &= \frac{\partial \tanh(W_1x_{k+1} + B_2 + B_1\bar{C}\beta_{k+1})}{\partial u_k} \\ &= \frac{\partial \tanh(\mathscr{X}_{k+1,k})}{\partial \mathscr{X}_{k+1,k}}\frac{\partial \mathscr{X}_{k+1,k}}{\partial u_k} = \frac{\partial \tanh(\mathscr{X}_{k+1,k})}{\partial \mathscr{X}_{k+1,k}}B_1\bar{C}\bar{B} \end{aligned} \tag{20}$$

where $\mathscr{X}_{k+1,k} = W_1x_{k+1} + B_2 + B_1\bar{C}\beta_{k+1}$. Therefore, $\frac{\partial E_3^2}{\partial u_k}$ is computed by (19) and (20). At the same time, the resulting $\frac{\partial \eta_{k+3}}{\partial u_k}$, $\frac{\partial x_{k+2}}{\partial u_k}$ and $\frac{\partial \beta_{k+1}}{\partial u_k}$ can be used to compute $\frac{\partial E_4^2}{\partial u_k}$ which is

$$\frac{\partial E_4^2}{\partial u_k} = 2E_4C_e\left(A_e\frac{\partial \eta_{k+3}}{\partial u_k} + B_eW_2\frac{\partial x_{k+3}}{\partial u_k}\right). \tag{21}$$

Note that $\frac{\partial x_{k+3}}{\partial u_k}$ is related to $\frac{\partial x_{k+2}}{\partial u_k}$ and $\frac{\partial \beta_{k+1}}{\partial u_k}$ with

$$\begin{aligned} \frac{\partial x_{k+3}}{\partial u_k} &= \frac{\partial \tanh(W_1x_{k+2} + B_2 + B_1\bar{C}\beta_{k+2})}{\partial u_k} \\ &= \frac{\partial \tanh(\mathscr{X}_{k+2,k})}{\partial \mathscr{X}_{k+2,k}} \cdot W_1\frac{\partial x_{k+2}}{\partial u_k} \cdot B_1\bar{C}\bar{A}\frac{\partial \beta_{k+1}}{\partial u_k}. \end{aligned} \tag{22}$$

Therefore, $\frac{\partial x_{k+i+1}}{\partial u_k}$ can be calculated from the previous calculation of $\frac{\partial x_{k+i}}{\partial u_k}$.

Once the derivative can be computed, the gradient descent method can be implemented with

$${U^f}^{(m+1)} \leftarrow {U^f}^{(m)} + \delta^{(m)}\frac{\partial \mathscr{J}}{\partial {U^f}^{(m)}} \tag{23}$$

Algorithm 1: Compute the Optimal Input.

Input: Current state ϕ_k and previous PEA drive input u_{k-1}.

Output: PEA drive input for next step u_k.

$m \leftarrow 1$
while $m < 15$ **do**
  $l \leftarrow k$
  while $l < N_h + k$ **do**
    $\phi_l \leftarrow F(\phi_l, u_l)$
    Calculate $\frac{\partial \tanh(\mathscr{X}_{l+1,l})}{\partial \mathscr{X}_{l+1,l}}$, $E_{l-k} = C_e\eta_{l+1} - r_{l+1}$ and β_{l+1}.
    $l \leftarrow l + 1$
  $j \leftarrow 0$
  while $j < N_c - 1$ **do**
    for *n=j+3 to* $N_h - 1$ **do**
      Calculate $\frac{\partial \beta_{n+1}}{\partial u_{k+j}}$ with $\frac{\partial \beta_{n+1}}{\partial u_{k+j}} = \bar{A}\frac{\partial \beta_n}{\partial u_{k+j}}$
      Calculate $\frac{\partial x_{k+j-1}}{\partial u_{k+j}}$ with Eq. (22)
      Calculate $\frac{\partial \eta_{k+j-1}}{\partial u_{k+j}}$ with $\frac{\partial \eta_{k+j}}{\partial u_{k+j}} = A_e\frac{\partial \eta_{k+j-1}}{\partial u_{k+j}} + B_eW_2\frac{\partial x_{k+j-1}}{\partial u_{k+j}}$
      Calculate $\frac{\partial E}{\partial u_{k+j}} \leftarrow \frac{\partial E}{\partial u_{k+j}} + E_{j+1}C_e\frac{\partial \eta_{k+j}}{\partial u_{k+j}}$
    $j \leftarrow j + 1$
  Compute $\frac{\partial \mathscr{J}}{\partial U^{f(m)}}$ using $\frac{\partial E}{\partial u_{k+j}}$.
  $U^{f(m+1)} \leftarrow U^{f(m)} + \delta^{(m)}\frac{\partial \mathscr{J}}{\partial U^{f(m)}}$
  $m \leftarrow m + 1$

where m is the iteration number and δ^m is the step length at mth step. In practice, we keep $m < 15$ or smaller to improve the computation efficiency. The above process can be summarized as shown in **Algorithm 1**.

Remark 1: Note that $\frac{\partial \mathscr{J}}{\partial U^{f(m)}}$ is normalized to be a unit vector. To keep $m < 15$, one approach is to use different step lengths at each iteration with the step length decreasing. For example, e^{-m} can be used to tune the step length. Other approaches such as backtracking line search can be applied alternatively [42].

Remark 2: We use "time complexity" to measure the efficiency of the algorithm, which denotes the number of operations to run the algorithm. For example, the time complexity of the matrix multiplication W_1x_k is $\Theta(2N^2)$ (i.e., N^2 multiplications and N^2 additions), which dominates the running time at each iteration, thus the running time is $\Theta(4N_hN_cN^2)$ for the numerical method [i.e., (13)] and $\Theta(2(N_h - N_c/2)N_cN^2)$ for the proposed method. Thus the computational efficiency is reduced by $1 - \frac{2(N_h - N_c/2)N_cN^2}{4N_hN_cN^2} = \frac{1}{2} + \frac{N_c}{4N_h}$, which is up to 75% when $N_c \approx N_h$.

In sum, the proposed controller can be implemented through the following three steps in one sample time.

Step 1: With the previous state ϕ_{k-1} and output of the PEA y_{k-1}, use (8)–(11) to estimate the current state ϕ_k.

Step 2: With the current state ϕ_k and the previous input u_{k-1}, call **Algorithm 1** to compute the next input u_k which is the output of **Algorithm 1**.

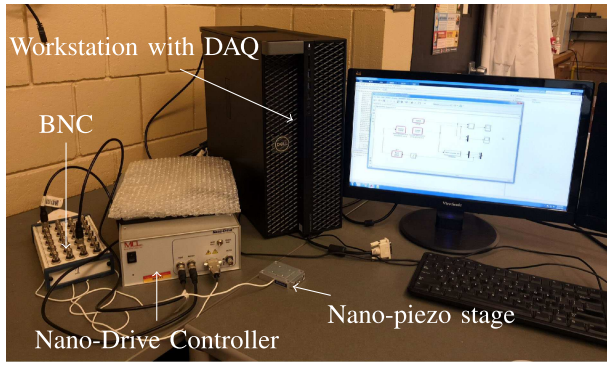


Fig. 4. RNN+LME experimental setup.

Step 3: Apply the computed input u_k to the system and repeat **Steps 1–3**.

Remark 3: Note that the dynamics of the LPF, such as the phase delay, will not affect the tracking performance since the LPF dynamics has been modeled in (6) and thus considered by the predictive controller.

IV. Experiment Results and Discussion

The proposed RNN+LME was implemented on a PEA (Nano-OP30, Mad City Labs) for demonstration. Fig. 4 shows the experiment setup. The displacement range of OP30 is 0–30 μm. All the signals were acquired through a data acquisition system (NI PCIe-6353, National Instruments) which was installed in the workstation (Intel Xeon W-2125, RAM 32 GB). The proposed RNN+LME was designed using MATLAB Simulink (MathWorks, Inc.). The sampling frequency was set to 10 kHz.

A. Training Set for RNN

To implement the k-means algorithm, 7000 points (blue dots in Fig. 5(a)) were randomly generated in the $f - A$ plane. 120 (f, A) points (red solid circles) were selected as illustrated in Fig. 5(a). The resulting training set $U_{(ts)}$ generated by concatenating all the sinusoidal signals (defined by the 120 $(f - A)$ points as (3)) is shown in Fig. 6. As aforementioned, this method of generating the RNN training set is flexible in the sense that constraints can be taken into account. For example, assuming that 1000 Hz is the resonant frequency of the PEA and the amplitude should decrease as the frequency increases to protect the PEA, these two constraints can be conveniently accounted for through controlling the distributions of the blue dots (see Fig. 5(b)) with the frequency range assumed to be 0–1.2 kHz. Then the training set can be generated using k-means algorithm (red dots) in Fig. 5(b).

B. Accuracy of the RNN, LME, and the State Estimator

We trained the RNN in the range of 0–330 Hz as the sampling frequency of our physical platform is set to 10 kHz. The RNN was generated using the training set shown in Fig. 6 (The RNN parameters are included in the supplementary material). Then, the RNN was tested to predict the PEA hysteresis for sinusoidal

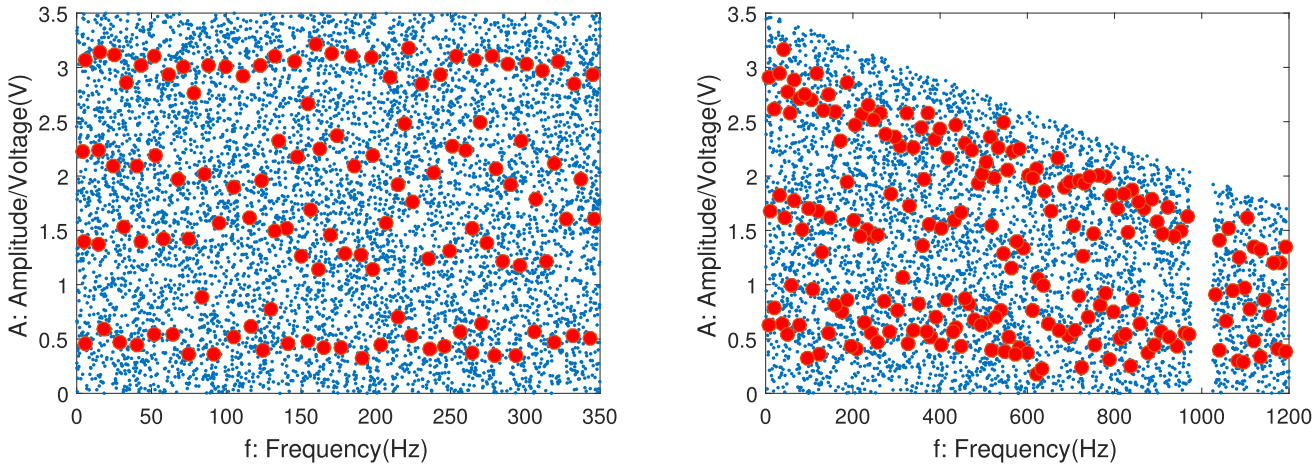


Fig. 5. $U_{(ts)}$ generated using k-means algorithm: (a) without constraints, and (b) with constraints: PEA resonant frequency at 1 kHz and monotonically decreasing amplitude versus frequency relation. Red dots represent the k-means algorithm selection results.

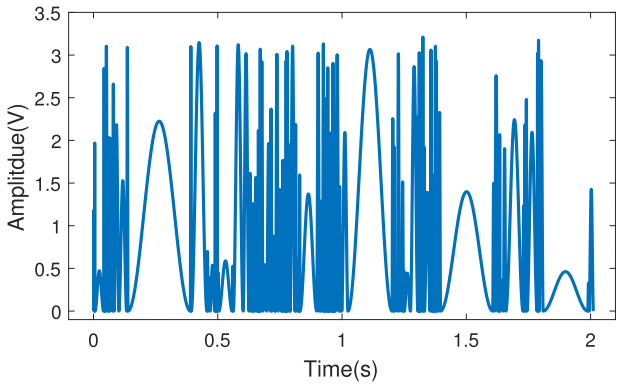


Fig. 6. Generated time series $U_{(ts)}$ for RNN training according to the selected (f, A) points in Fig. 5(a).

signals with different frequencies and amplitudes, see Fig. 7. The obtained RNN can model the hysteresis accurately for signals of high frequencies, as shown in Fig. 7(b) and (c). The modeling error increases as the frequency goes down to 5 Hz as shown in Fig. 7(a). This verifies the limitations of the RNN in modeling low-frequency dynamics, and demonstrates that it is necessary to integrate RNN and LME together for accurate PEA system modeling.

Furthermore, to evaluate the modeling accuracy of RNN in the time domain, the outputs of the obtained RNN model are compared with the actual PEA outputs for different input signals. Specifically, four inputs including sinusoidal signals with frequencies of 5, 120, and 320 Hz and a Γ signal shown below were tested. To highlight the efficacy of the proposed approach, the frequency components of these inputs were not overlapping with that in the training set.

$$\Gamma(t) = [0.8\sin(2\pi 5t + 1.5\pi) + 0.43\sin(2\pi 50t) + 0.12\sin(2\pi 120t + 1.2\pi) + 0.3\sin(2\pi 180t + \pi)]/1.3. \tag{24}$$

TABLE I
STANDARD DEVIATIONS OF MODELING ERRORS OF RNN AND RNN+LME

Signal	5 Hz	120 Hz	320 Hz	Γ(t)
SD (μm)/RNN	0.214	0.091	0.067	0.189
SD (μm)/RNN+LME	0.020	0.065	0.145	0.025

Feeding the same input U to the actual PEA system and RNN, we obtained the outputs from the real system Y_1 and RNN Y_2. Thus the modeling error is $E = Y_1 - Y_2$. Standard deviations of the modeling errors E are presented in Table I. It can be seen that the RNN modeling error increases as the frequency of the signal decreases. The standard deviation increased about 50% for the 5 Hz signal compared to that of high frequency signals—120 and 320 Hz. The modeling results were plotted in Fig. 8, which is in accordance with the data in Table I.

With the obtained RNN model, the unmodeled dynamics of PEA is accounted for with the proposed LME model whose parameters are determined through solving the optimization problem (5). The LME model is as follows:

$$\eta_{k+1} = \begin{bmatrix} 1.5288 & 1.6862 \\ -0.2775 & 0.1252 \end{bmatrix} \eta_k + \begin{bmatrix} -0.7033 \\ -2.1788 \end{bmatrix} u_{(l),k} + \begin{bmatrix} -2.0581 \\ 2.1707 \end{bmatrix} \hat{e}_k$$
$$y_k = \begin{bmatrix} -0.6450 & 0.0548 \end{bmatrix} \eta_k. \tag{25}$$

Similarly, the modeling accuracy of RNN+LME is evaluated as shown in Fig. 8 and Table I. As shown in Table I, the integration of RNN+LME has improved the modeling accuracy significantly, especially at low frequency region. In addition, compared to using RNN alone, RNN+LME is very effective in removing the offset. Since the offset can be tuned to be very small before the experiment, the errors shown in Table I are

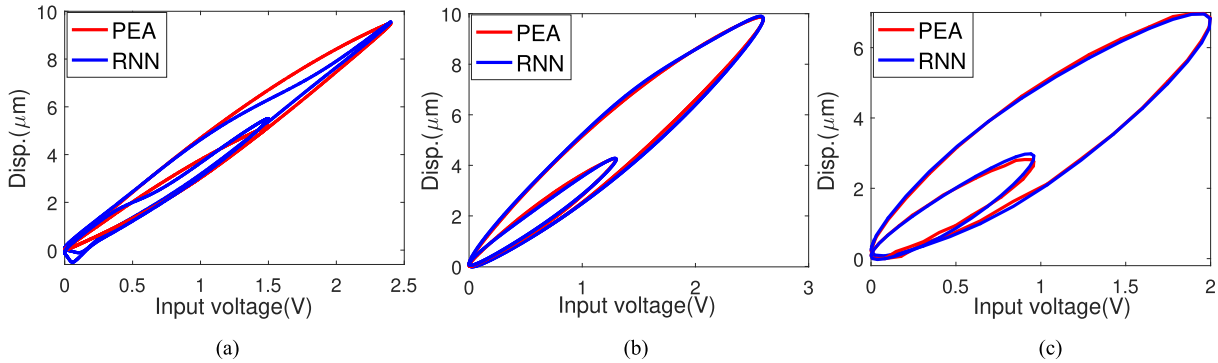


Fig. 7. Comparison of the RNN modeled PEA hysteresis (in blue) versus the experimentally measured results (in red) at the frequencies of (a) 5 Hz, (b) 120 Hz, and (c) 320 Hz with different displacement ranges.

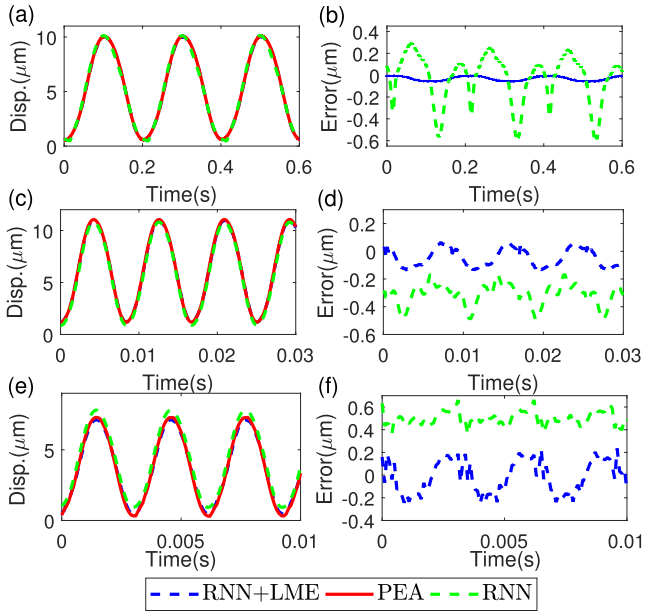


Fig. 8. Outputs of RNN+LME, RNN and PEA for (a) 5 Hz, (c) 120 Hz, and (e) 320 Hz sinusoidal inputs, and the corresponding modeling errors (b), (d), and (f), respectively.

computed with the offset being removed. Note that the modeling error increased a little bit for 320 Hz as the high frequency portion might be affected by LME. Nevertheless, the overall improvement of RNN+LME over RNN is significant.

Moreover, the performance of the estimator was also tested by comparing the actual output and the predicted output for 5 and 200 Hz sinusoidal inputs, respectively, as shown in Fig. 9. It is clear that the state estimator proposed in this RNN+LME framework can accurately predict the PEA output with the prediction error less than 2%.

C. Simulation Results

To show the effect of N_h and N_c on the tracking performance, predictive controllers with different (N_h, N_c)s were simulated to track a 320 Hz sinusoidal trajectory based on the obtained RNN model. The tracking results are shown in Fig. 10, which

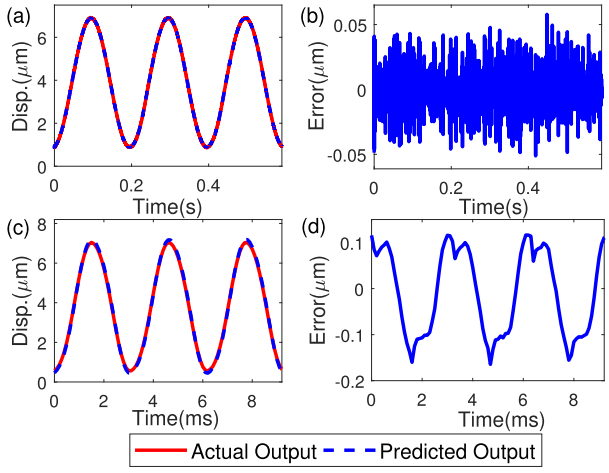


Fig. 9. Actual output and predicted output of the estimator for (a) 5 Hz and (c) 320 Hz sinusoidal inputs, and the corresponding prediction errors (b) and (d), respectively.

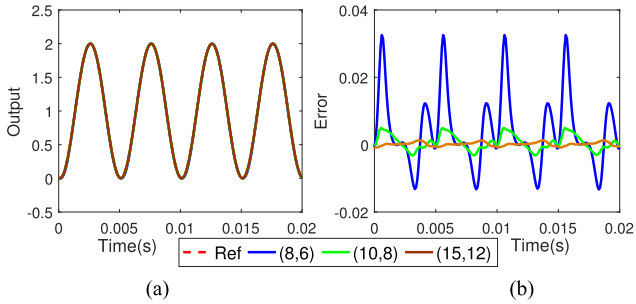


Fig. 10. (a) Tracking results of predictive controllers with different (N_h, N_c) parameters and (b) tracking errors.

clearly shows that the tracking error decreases as N_h and N_c increase. Table II presents the computed tracking errors. Note that $E_{\max}$ and E_{rms} are calculated as shown in the following:

$$E_{\max} \triangleq \frac{||r(\cdot) - y(\cdot)||_\infty}{||r(\cdot)||_\infty}, \; E_{\text{rms}} \triangleq \frac{||r(\cdot) - y(\cdot)||_2}{||r(\cdot)||_2}. \tag{26}$$

where $r(\cdot)$ and $y(\cdot)$ are complex vectors obtained through discrete Fourier transform of the corresponding signals. As shown

TABLE II
TRACKING PERFORMANCE OF THE DESIGNED PREDICTIVE CONTROLLERS WITH DIFFERENT (N_h, N_c)S WHEN 200 HZ SINUSOIDAL SIGNAL WAS USED AS THE REFERENCE TRAJECTORY IN SIMULATION

(N_h, N_c)	(8,6)	(10,8)	(15,12)
Std	0.011	0.002	5.380e-4
$E_{\text{rms}}(\%)$	0.526	0.143	0.026
$E_{\text{max}}(\%)$	0.948	0.196	0.046

TABLE III
TRACKING PERFORMANCE COMPARISON OF RNN+LME, RNN, PID, AND MIIFC

Ref.	Error	RNN+LME	RNN	PID	MIIFC
5 Hz	$E_{\text{rms}}(\%)$	0.89	1.93	0.20	1.92
	$E_{\text{max}}(\%)$	0.47	1.26	0.10	1.05
30 Hz	$E_{\text{rms}}(\%)$	1.27	3.13	2.69	2.01
	$E_{\text{max}}(\%)$	0.39	1.36	2.29	1.21
100 Hz	$E_{\text{rms}}(\%)$	2.32	4.00	9.19	1.81
	$E_{\text{max}}(\%)$	1.74	2.89	7.93	1.18
200 Hz	$E_{\text{rms}}(\%)$	7.48	9.84	19.29	1.69
	$E_{\text{max}}(\%)$	6.31	8.42	16.61	1.11
Γ	$E_{\text{rms}}(\%)$	1.82	2.11	4.81	2.23
	$E_{\text{max}}(\%)$	0.69	1.08	3.30	1.16
stair	$E_{\text{rms}}(\%)$	2.38	3.49	4.07	1.50
	$E_{\text{max}}(\%)$	1.00	2.01	0.35	1.19

in Table II, the tracking error of the predictive controller decreased more than 70% by increasing the prediction and control horizons from (8, 6) to (10, 8). On the other hand, large N_h and N_c imply that much higher computation power is needed. In the experiment part, we chose (8, 6) considering the speed of the hardware thus a downgrade of the tracking performance of the predictive controller was expected.

D. Tracking Performance Comparison

Both LPFs in Fig. 3 were chosen as fourth-order Butterworth LPFs with the cutoff frequency of 1.5 kHz. Five trajectories—sinusoidal signals with frequencies of 5, 30, 100, and 200 Hz, the aforementioned Γ [see (24)], and a stair-case signal were tracked. Apart from the proposed RNN+LME, three other methods were applied. First, to verify the necessity of LME in the proposed approach, we designed the predictive controller without LME (called "RNN" in the rest). The second method is PID which has been popular in PEA control. Moreover, since ILC proved to be very accurate in output tracking through repetitively performing the same task, one ILC-based approach—e.g., modeling-free inversion-based iterative feedforward control (MIIFC) [12], was chosen for comparison as well. Although it is unfair to compare the proposed real-time control framework with the offline controller—MIIFC, the tracking performance of MIIFC was used as a benchmark to demonstrate that the proposed real-time control can achieve similar (or even better) control accuracy when compared to an offline controller.

The calculated tracking errors are shown in Table III and the comparisons in time domain are presented in Fig. 11, where the proposed method RNN+LME are compared with RNN, PID, and MIIFC, respectively.

RNN+LME versus RNN: As seen in Table III, the tracking errors for all trajectories of RNN+LME are lower than that of RNN, which indicates that adding LME could indeed improve the tracking performance. In particular, RNN+LME decreased the tracking errors E_{rms} and E_{max} by at least 59.4% and 71.3%, respectively for the 5 and 30 Hz sinusoidal trajectories, the improvements of which were much more significant than the other cases. This is because the tracking error caused by unmodeled low-frequency dynamics was accounted for by LME. On the other hand, the improvement for high frequency signal tracking (e.g., 200 Hz signal) is not so obvious implying that LME does not contribute to high-frequency modeling. In Fig. 11(d), the tracking errors of RNN+LME and RNN have similar trends showing that the tracking error for high-frequency dynamics was not removed through the LME. Therefore, the issue of RNN for capturing low-frequency dynamics can be solved through incorporating the LME, in which the error term can act as a feedback term.

RNN+LME versus PID: RNN+LME outperforms PID for almost all the trajectories as can be seen in Table III and Fig. 11 except for 5 Hz sinusoidal signal. PID is relatively effective for tracking low frequency trajectories such as the 30 Hz sinusoidal signal, however, RNN+LME could further decrease the E_{rms} and E_{max} by about 52.8% and 83.0% compared to PID, respectively. When tracking high frequency signal, such as the 200$\sim$Hz sinusoidal trajectory, RNN+LME decreased the E_{rms} and E_{max} errors by at least 60.0% with respect to the performance of PID, respectively. Fig. 11 also confirms the superiority of RNN+LME over PID.

RNN+LME versus MIIFC: MIIFC has proven to achieve accurate output tracking by repeating the same tasks several times [12]. In this experiment, the MIIFC tracking results of the $8th$ iteration, where MIIFC has converged, were chosen for comparison. The data in Table III show that RNN+LME was comparable (or even better for some cases) with MIIFC when the frequency of the trajectory was not too high, which can also be seen in Fig. 11. When tracking the 30 Hz sinusoidal signal, the tracking errors E_{rms} and E_{max} of RNN+LME were about 36.8% and 67.8% less than that of MIIFC, but increased about 28.2% and 47.5% for the 100 Hz case. When tracking the stair-case signal, it seems that RNN+LME is not as good as that of MIIFC as shown in Fig. 11. This is because the frequency components of the stair-case trajectory exceed the modeling bandwidth of RNN+LME. However, since the high frequency part only accounts for a small portion of the entire trajectory, the tracking result is actually close to that of MIIFC as shown in Table III. Because the low-frequency dynamics is dominated in Γ, it is reasonable that RNN+LME tracked Γ better than MIIFC. Considering the PEA high frequency dynamics has been modeled by the RNN accurately as demonstrated by Figs. 7, 8, and Table I, the main reason of the tracking errors increase for high frequency trajectories is that the chosen N_h and N_c were too small. By increasing N_h and N_c, the tracking accuracy can be improved, which has been verified by the simulation results (Fig. 10). The factor restricting using large N_h and N_c is the

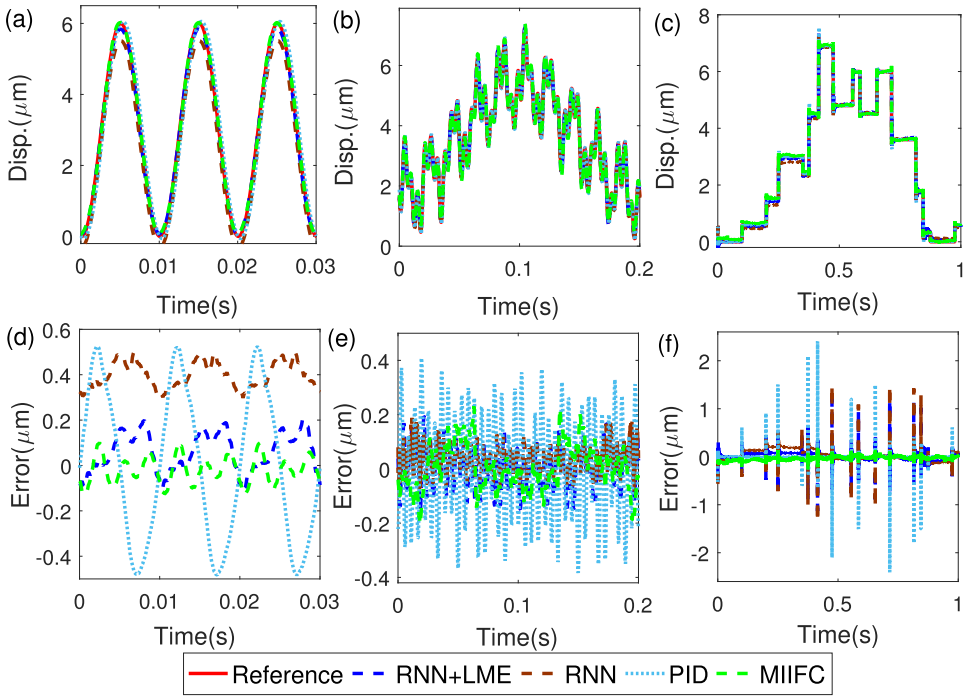


Fig. 11. Comparison of the tracking results of RNN+LME, RNN, PID, and MIIFC for (a) 100 Hz sinusoidal trajectory, (b) Γ, and (c) stair-case trajectory. (b), (e), and (f) are the corresponding tracking errors, respectively.

computation burden. As Algorithm 1 shows that much of the computations lie in the matrix multiplications, one solution is to implement the algorithm in faster hardware platforms such as platform based on FPGA.

In summary, the experiment results demonstrated the efficacy of the proposed method. Applications in which motion control of PEAs is required will benefit significantly from the improvement on the tracking accuracy from the proposed framework. For example, PEAs are used to generate friction force in human-robot interaction [6]. Precise force tracking control of PEAs is needed for the desired damping behaviors. Another important application is AFM as the AFM scanner motion is completely driven by the PEAs [7], [8], [43]. As the sample topography is often irregular in amplitude and frequency, the nonlinear dynamics of the PEAs will be excited during AFM topography scanning, especially at high speed. In such case, the PID feedback controller widely used in commercial AFMs becomes inadequate. In addition, in AFM material mechanical characterization, the AFM probe-sample interaction force needs to track the predesigned excitation force [9], [44], otherwise, the samples can be damaged if the force is not precisely controlled, especially for delicate samples (e.g., living biological materials) [45], [46]. Although ILC-based approaches can achieve high tracking accuracy, iterations should be avoided on delicate samples. Therefore, the tracking accuracy achieved by the proposed technique becomes essential for AFM-related applications. Furthermore, note that the proposed method does not assume the exact form of nonlinearities, it can be conveniently employed to other nonlinear systems, i.e., not limited to PEA systems.

V. Conclusion

In this article, we proposed an RNN-based predictive control approach (RNN+LME) to achieve accurate real-time trajectory tracking of PEAs. An RNN cascaded with an LME was proposed to model the nonlinear dynamics of the PEAs, and the UKF was developed to estimate the states of the nonlinear model. A nonlinear predictive controller, was designed to work with the PEA model, and the optimal input from the controller to the PEA was calculated by using the gradient descent method to solve the optimization problem and the analytical method to compute the gradient was derived as well. Implementation of the proposed approach on a PEA showed that it can achieve high tracking accuracy when the desired trajectory spanned in a broad frequency range.

For future work, the modeling bandwidth needs to be further enlarged with other types of RNNs. Also, we will work on mitigating the effect of the mechanical vibrations through modeling the vibration dynamics or introducing adaptive control laws to reject the disturbances. Finally, how the reference signal will affect the tracking error quantitatively will be investigated.

References

[1] G. M. Clayton, S. Tien, K. K. Leang, Q. Zou, and S. Devasia, "A review of feedforward control approaches in nanopositioning for high-speed spm," *J. Dyn. Syst., Meas., Control*, vol. 131, no. 6, 2009, Art. no. 061101.

[2] Y. F. Dufrêne *et al.*, "Imaging modes of atomic force microscopy for application in molecular and cell biology," *Nature Nanotechnol.*, vol. 12, no. 4, pp. 295–307, 2017.

[3] E. T. Herruzo, A. P. Perrino, and R. Garcia, "Fast nanomechanical spectroscopy of soft matter," *Nature Commun.*, vol. 5, 2014, Art. no. 3126.

[4] Y. Tian, D. Zhang, and B. Shirinzadeh, "Dynamic modelling of a flexure-based mechanism for ultra-precision grinding operation," *Precis. Eng.*, vol. 35, no. 4, pp. 554–565, 2011.

[5] F. Qin, D. Zhang, D. Xing, D. Xu, and J. Li, "Laser beam pointing control with piezoelectric actuator model learning," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, Oct. 25, 2017, doi: 10.1109/TSMC.2017.2754863.

[6] J. Lee, M. Laffranchi, N. Kashiri, N. G. Tsagarakis, and D. G. Caldwell, "Model-free force tracking control of piezoelectric actuators: Application to variable damping actuator," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 2283–2289.

[7] J. Ren and Q. Zou, "High-speed adaptive contact-mode atomic force microscopy imaging with near-minimum-force," *Rev. Sci. Instrum.*, vol. 85, no. 7, 2014, Art. no. 073706.

[8] S. Xie and J. Ren, "High-speed AFM imaging via iterative learning-based model predictive control," *Mechatronics*, vol. 57, pp. 86–94, 2019.

[9] Z. Xu and Q. Zou, "Optimal excitation force design in indentation-based rapid broadband nanomechanical spectroscopy: Poly (dimethylsiloxane) example," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 5, pp. 1618–1628, Sep. 2013.

[10] S. Tien, Q. Zou, and S. Devasia, "Iterative control of dynamics-coupling-caused errors in piezoscanners during high-speed AFM operation," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 6, pp. 921–931, Nov. 2005.

[11] A. A. Eielsen, J. T. Gravdahl, and K. K. Leang, "Low-order continuous-time robust repetitive control: Application in nanopositioning," *Mechatronics*, vol. 30, pp. 231–243, 2015.

[12] K.-S. Kim and Q. Zou, "A modeling-free inversion-based iterative feedforward control for precision output tracking of linear time-invariant systems," *IEEE/ASME Trans. Mechatronics*, vol. 18, no. 6, pp. 1767–1777, Dec. 2013.

[13] B. Altın, J. Willems, T. Oomen, and K. Barton, "Iterative learning control of iteration-varying systems via robust update laws with experimental implementation," *Control Eng. Pract.*, vol. 62, pp. 36–45, 2017.

[14] S. Xie and J. Ren, "Note: Precision control of nano-positioning stage: An iterative learning-based model predictive control approach," *Rev. Sci. Instrum.*, vol. 89, no. 7, 2018, Art. no. 076103.

[15] M. S. Rana, H. R. Pota, and I. R. Petersen, "The design of model predictive control for an AFM and its impact on piezo nonlinearities," *Eur. J. Control*, vol. 20, no. 4, pp. 188–198, 2014.

[16] Q. Xu, "Digital sliding-mode control of piezoelectric micropositioning system based on input-output model," *IEEE Trans. Ind. Electron.*, vol. 61, no. 10, pp. 5517–5526, Oct. 2014.

[17] Y. Pan, C. Yang, L. Pan, and H. Yu, "Integral sliding mode control: Performance, modification and improvement," *IEEE Trans. Ind. Inform.*, vol. 14, no. 7, pp. 3087–3096, Jul. 2018.

[18] K. K. Leang and S. Devasia, "Feedback-linearized inverse feedforward for creep, hysteresis, and vibration compensation in AFM piezoactuators," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 5, pp. 927–935, Sep. 2007.

[19] G.-Y. Gu, L.-M. Zhu, C.-Y. Su, H. Ding, and S. Fatikow, "Modeling and control of piezo-actuated nanopositioning stages: A survey," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 313–332, Jan. 2016.

[20] R. C. Smith and Z. Ounaies, "A domain wall model for hysteresis in piezoelectric materials," *J. Intell. Mater. Syst. Struct.*, vol. 11, no. 1, pp. 62–79, 2000.

[21] J. W. Macki, P. Nistri, and P. Zecca, "Mathematical models for hysteresis," *SIAM Rev.*, vol. 35, no. 1, pp. 94–123, 1993.

[22] X. Chen, C.-Y. Su, Z. Li, and F. Yang, "Design of implementable adaptive control for micro/nano positioning system driven by piezoelectric actuator," *IEEE Trans. Ind. Electron.*, vol. 63, no. 10, pp. 6471–6481, Oct. 2016.

[23] Y. Cao, L. Cheng, X. Chen, and J. Peng, "An inversion-based model predictive control with an integral-of-error state variable for piezoelectric actuators," *IEEE/ASME Trans. Mechatronics*, vol. 18, no. 3, pp. 895–904, Jun. 2013.

[24] G. Song, J. Zhao, X. Zhou, and J. A. De Abreu-García, "Tracking control of a piezoceramic actuator with hysteresis compensation using inverse Preisach model," *IEEE/ASME Trans. Mechatronics*, vol. 10, no. 2, pp. 198–209, Apr. 2005.

[25] M. Al Janaideh, S. Rakheja, and C.-Y. Su, "An analytical generalized Prandtl– Ishlinskii model inversion for hysteresis compensation in micropositioning control," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 4, pp. 734–744, Aug. 2011.

[26] M.-S. Tsai and J.-S. Chen, "Robust tracking control of a piezoactuator using a new approximate hysteresis model," *J. Dyn. Syst., Meas., Control*, vol. 125, no. 1, pp. 96–102, 2003.

[27] M. Quant, H. Elizalde, A. Flores, R. Ramírez, P. Orta, and G. Song, "A comprehensive model for piezoceramic actuators: Modelling, validation and application," *Smart Mater. Struct.*, vol. 18, no. 12, 2009, Art. no. 125011.

[28] L. Cheng, W. Liu, Z.-G. Hou, J. Yu, and M. Tan, "Neural-network-based nonlinear model predictive control for piezoelectric actuators," *IEEE Trans. Ind. Electron.*, vol. 62, no. 12, pp. 7717–7727, Dec. 2015.

[29] W. Liu, L. Cheng, Z.-G. Hou, J. Yu, and M. Tan, "An inversion-free predictive controller for piezoelectric actuators based on a dynamic linearized neural network model," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 1, pp. 214–226, Feb. 2016.

[30] F.-J. Lin, H.-J. Shieh, and P.-K. Huang, "Adaptive wavelet neural network control with hysteresis estimation for piezo-positioning mechanism," *IEEE Trans. Neural Netw.*, vol. 17, no. 2, pp. 432–444, Mar. 2006.

[31] F.-J. Lin, H.-J. Shieh, P.-K. Huang, and P.-H. Shieh, "An adaptive recurrent radial basis function network tracking controller for a two-dimensional piezo-positioning stage," *IEEE Trans. Ultrasonics, Ferroelectrics, Frequency Control*, vol. 55, no. 1, pp. 183–198, Jan. 2008.

[32] X. Zhang *et al.*, "Decentralized adaptive neural approximated inverse control for a class of large-scale nonlinear hysteretic systems with time delays," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 110.1109/TSMC.2018.2827101.

[33] C.-L. Hwang and C. Jan, "A reinforcement discrete neuro-adaptive control for unknown piezoelectric actuator systems with dominant hysteresis," *IEEE Trans. Neural Netw.*, vol. 14, no. 1, pp. 66–78, Jan. 2003.

[34] R. Uddin and J. Ryu, "Predictive control approaches for bilateral teleoperation," *Annu. Rev. Control*, vol. 42, pp. 82–99, 2016.

[35] H. Zimmermann, R. Grothmann, A. Schaefer, and C. Tietz, "Identification and forecasting of large dynamical systems by dynamical consistent neural networks," *New Directions Statistical Signal Processing: From Systems to Brain*. Cambridge, MA, USA: MIT Press, 2006, pp. 203–242.

[36] A. M. Schäfer and H.-G. Zimmermann, "Recurrent neural networks are universal approximators," *Int. J. Neural Syst.*, vol. 17, no. 4, pp. 253–263, 2007.

[37] C. Wang, A. Ohsumi, and I. Djurovic, "Model predictive control of noisy plants using Kalman predictor and filter," in *Proc. IEEE Region 10 Conf. Comput., Commun., Control Power Eng.*, 2002, vol. 3, pp. 1404–1407.

[38] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, Oakland, CA, USA, 1967, vol. 1, pp. 281–297.

[39] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter," in *Kalman Filtering Neural Networks*. Hoboken, NJ, USA: Wiley 2001, pp. 221–280.

[40] M. Rana, H. R. Pota, and I. R. Petersen, "Model predictive control of atomic force microscope for fast image scanning," in *Proc. IEEE 51st Annu. Conf. Decis. Control*, 2012, pp. 2477–2482.

[41] M. A. Nielsen, *Neural Networks and Deep Learning*, vol. 25, San Francisco, CA, USA: Determination Press, 2015.

[42] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[43] H. Lu, Y. Fang, X. Ren, and X. Zhang, "Improved direct inverse tracking control of a piezoelectric tube scanner for high-speed AFM imaging," *Mechatronics*, vol. 31, pp. 189–195, 2015.

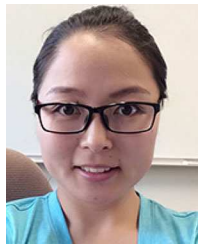
[44] Z. Xu, K. Kim, Q. Zou, and P. Shrotriya, "Broadband measurement of rate-dependent viscoelasticity at nanoscale using scanning probe microscope: Poly (dimethylsiloxane) example," *Appl. Phys. Lett.*, vol. 93, no. 13, 2008, Art. no. 133103.

[45] J. Ren, A. Mousavi, X. Li, Q. Zou, N. Erina, and C. Su, "Enhanced measurement of broadband nanomechanical property of polymers using atomic force microscope," *Appl. Phys. Lett.*, vol. 102, no. 18, 2013, Art. no. 183116.

[46] J. Ren, S. Yu, N. Gao, and Q. Zou, "Indentation quantification for in-liquid nanomechanical measurement of soft material using an atomic force microscope: Rate-dependent elastic modulus of live cells," *Phys. Rev. E*, vol. 88, no. 5, 2013, Art. no. 052711.

Shengwen Xie received the B.S. degree in mechanical engineering from the Huazhong University of Science and Technology, Wuhan, China in 2013, and the M.S. degree in electrical engineering from Tianjin University, Tianjin, China in 2016. He is currently working toward the Ph.D. degree in mechanical engineering at Iowa State University, Ames, IA, USA.

His research interests include system identification, output tracking, and predictive control.



Juan Ren received the B.S. degree from Xian Jiaotong University, China in 2009, and the Ph.D. degree from the State University of New Jersey, New Brunswick, NJ, USA, in June 2015, both in mechanical engineering.

She is currently an Assistant Professor with the Department of Mechanical Engineering at Iowa State University where she has been on the faculty since August 2015. Her research interests include learning-based output tracking and control, control tools for high-speed scanning probe microscope imaging, mechanotransduction modeling and nanomechanical measurement, and mapping of soft and live biological materials.

Dr. Ren was the recipient of the NSF CAREER award in 2018, and currently holds the William and Virginia Binger Professorship in the Department of Mechanical Engineering of ISU. She is the representative of the IEEE Control Systems Society in the IEEE Nanotechnology Council, and an Associate Editor of *Elsevier Mechatronics*.