

Long Short-term Memory Neural Network-based System Identification and Augmented Predictive Control of Piezoelectric Actuators for Precise Trajectory Tracking

Mayur S. Patil . Bharat Charuku . Juan Ren *

* Iowa State University, Ames, IA 50011, USA (e-mail: juanren@iastate.edu)

Abstract: High-speed and high-precision systems based on piezoelectric actuator (PEA) demand precise real-time trajectory tracking. Model-based control techniques have been proven effective in achieving desired tracking accuracy. However, modeling uncertainty and linearization losses are the biggest hurdles in these techniques to achieve high precision performance over broad frequency ranges at high speed. To overcome these limitations, in this work, we propose a long short-term memory (LSTM) neural network-based inverse system identification and augmented predictive control using a linear model predictive control (MPC) to achieve high precision trajectory tracking of PEAs. An LSTM network was built and trained to model the inversion dynamics of the PEA system. The benefit of using LSTM is that it ensures the long-term dependencies of time series data, and hence it can model the system dynamics for both low and high-frequency ranges. Once the LSTM inversion model accuracy was evaluated, it cascaded with the commercial PEA, which together was mostly linear. This combined system was controlled by augmenting it with a linear MPC controller. The use of augmented predictive control using the nonlinear LSTM inversion model led to improved modeling accuracy and higher speed of operation with a reduced computational load. The results demonstrated the efficacy of the proposed approach in real-time high-speed trajectory tracking of PEAs.

Copyright © 2021 The Authors. This is an open access article under the CC BY-NC-ND license (<https://creativecommons.org/licenses/by-nc-nd/4.0/>)

Keywords: Neural networks, Nonlinear system identification, Identification for control, Model predictive control, Tracking, Real-time control, Piezoelectric actuators.

1. INTRODUCTION

In today's world of nanotechnology, systems and applications are going towards high speed and nano precision in operation. The use of PEAs in such applications for the best possible accuracy has become common practice. The inherited properties such as mechanical stability, high resolution, and fast response time of PEAs make them suitable for a variety of high precision applications, such as atomic force microscope (AFM) (Mollaeian et al. (2018)), scanning tunneling microscopy (Hansma et al. (1988/10/14)), charge-coupled device drives (Yoichi (2006)), and mass flow control (Yoichi (2006)). However, the nonlinearities caused by the hysteresis and creep effects of PEAs are the barriers in achieving the desired modeling and control accuracy.

There have been a lot of efforts to tackle this problem using different real-time control schemes, such as model predictive control (MPC) (Rana et al. (2014)), and iterative learning-based model predictive control (ILMPC) (Xie and Ren (2018)). However, the performance of both MPC and ILMPC were limited due to the rate-dependent nonlinear dynamics of PEAs (Rana et al. (2014) and Xie and Ren (2018)), which becomes more significant as the operating frequency increases (Xie and Ren (2019a)). Discrete-time

quasi-sliding mode control (DQSMC) (Nguyen and Chen (2017)) and adaptive sliding mode (Utkin and Lee (2006), Huang et al. (2009)) have been proposed to ease the control challenges caused by system nonlinearities and modeling uncertainties. However, both approaches suffer from the chattering problem (Utkin and Lee (2006)). Moreover, although both MPC and SMC are effective in real-time trajectory tracking, these techniques significantly rely on the system model, which means that their performance is susceptible to fallacies in the plant or system dynamic model. Hence, obtaining an accurate dynamic model is still the major obstacle for precision trajectory tracking of PEAs.

Historically, physical and phenomenological models of nonlinearities have been used in control theories. Popular modeling approaches such as Preisach model (Ge and Jouaneh (1997), Mrad and Hu (2002)), Maxwell model (Liu et al. (2013)), log (t) model (Wei et al. (1996), Jung and Gweon (2000)) etc. are robust and reliable modeling methods. Although these models are well in modeling nonlinearities, they are exclusive to a certain type of nonlinearity. Additionally, few models are rate-independent in nature, hence limiting their use in high accuracy applications. Recently, machine learning-based models, such as the neural network models, have proven its efficacy in the identification

of system dynamics (Cheng et al. (2015), Xie and Ren (2019b)). Neural networks, such as feedforward neural network (FNN), and recurrent neural network (RNN), have been proposed to model the dynamics of the PEAs (Cheng et al. (2015), Xie and Ren (2019b), Liu et al. (2015)). However, the problem with FNN based model is that it does not consider the system input as a time series (sequence input) for the purpose of training (Xie and Ren (2019b)). On the other hand, RNN is much more effective than FNN and considers the system memory and recognizes inter-temporal dependencies (Xie and Ren (2019b) and Schäfer and Zimmermann (2006)). However, it fails to consider and understand the long-term dependencies of the sequential input and hence cannot capture the correct dynamics of the system over the entire band of operating frequency (Hochreiter and Schmidhuber (1997)), especially with variable frequency drive inputs. To overcome the aforementioned issues, ECNN (error correction neural network) and DCNN (dynamical consistent recurrent neural networks) have been proposed (Haykin et al. (2007)). Although they work better than RNN, their ability in capturing the system's long-term dependencies is still limited, and hence fail to achieve the desired accuracy in some applications (Haykin et al. (2007)). Such a problem can be solved with deep RNN (Pascanu et al. (2013), and Pang et al. (2019)). However, deep RNNs are more susceptible to gradient vanishing or exploding and hence difficult to train (Hochreiter and Schmidhuber (1997)).

To address these issues, in this work, we proposed to use Long Short-Term Memory Network (LSTM) for PEA system dynamics identification. LSTM takes a sequence of the inputs into consideration and identifies the system's long-term temporal dependencies. Such a network can be effectively used for nonlinear system modeling (Gonzalez and Yu (2018)). Moreover, unlike RNN, LSTM does not suffer from the gradient vanishing or exploding problem during the training process. In the proposed work, the LSTM network is used to capture the inverse dynamics of a PEA system, i.e., to obtain an inversion model. The LSTM network is trained based on a gradient descent algorithm (Narendra and Parthasarathy (1991)). Next, the accuracy of the obtained LSTM was evaluated by comparing the LSTM output with the actual measured PEA input-output data. Then, the obtained inverse LSTM system model was cascaded with the commercial PEA system, working together, resulting in a nearly linear behaving system. Hence, this combined system allows the use of linear control schemes without sacrificing the modeling accuracy and eliminates the problem of linearization and linearization losses. Additionally, it significantly reduces the computational load and hence makes the control process faster. Finally, this combined system was augmented and controlled by a linear model predictive controller for high-speed, high precision tracking control of PEA.

2. LONG SHORT-TERM MEMORY NETWORK (LSTM)-BASED SYSTEM IDENTIFICATION

In this section, we provide the details of the LSTM neural network, including the architecture of the LSTM cell, and the process of obtaining the training data set for the LSTM model.

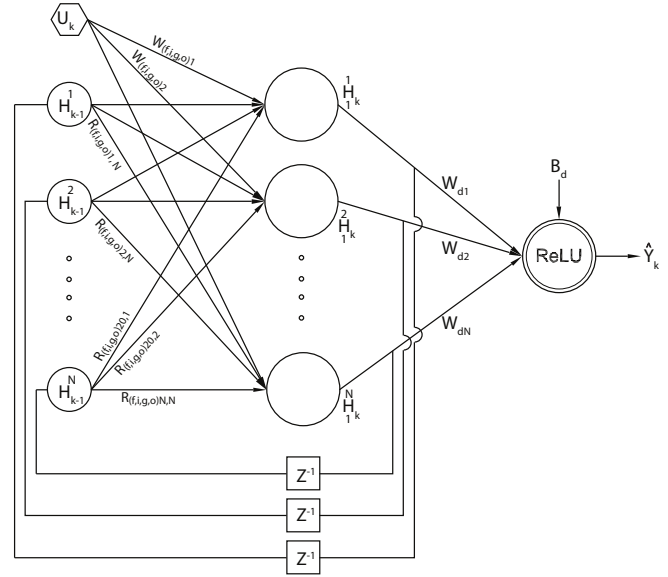


Fig. 1. Structure of the LSTM

2.1 Long Short-Term Memory Network

Fig. 1 presents the structure of the LSTM model for modeling the PEA dynamics. The LSTM model consists of an input layer (hexagon), which is followed by one hidden layer of LSTM neurons (big circles), and an output dense layer (concentric circle). Multiple hidden layers of LSTM units can be stacked together following each other, with each unit of the previous layer connected to all LSTM units of the following layer for more accurate system identification.

As shown in Fig. 1, the output layer has *ReLU* as an activation function, and it can be represented as a measurement function, i.e., $\hat{Y}_k = ReLU(H_k, W_d, B_d)$, with W_d and B_d as weights and biases associated to the output layer, respectively. U_k and $\hat{Y}_k$ are the reference input and the LSTM model output at the sampling instant k , respectively. H_k is the output from the previous layer. Unlike RNN cell, LSTM cell (big circles in Fig. 1) has four gates named input, output, forget, and cell gate, and they can be represented as I_k, O_k, F_k, G_k respectively, as shown in Fig. 2. These four gates have their own activation function, weights, and biases.

The output from this LSTM network at time instant k for given input U_k is obtained by using the following equations,

$$\begin{aligned}
 F_k &= \sigma([W_f \times U_k] + [R_f \times H_{k-1}] + B_f) \\
 O_k &= \sigma([W_o \times U_k] + [R_o \times H_{k-1}] + B_o) \\
 G_k &= \tanh([W_g \times U_k] + [R_g \times H_{k-1}] + B_g) \\
 I_k &= \sigma([W_i \times U_k] + [R_i \times H_{k-1}] + B_i) \\
 C_k &= F_k * C_{k-1} + I_k * G_k \\
 H_k &= O_k * \tanh(C_k) \\
 \hat{Y}_k &= ReLU([W_d \times H_k] + B_d)
 \end{aligned} \tag{1}$$

* signifies element-wise multiplication.
Where,

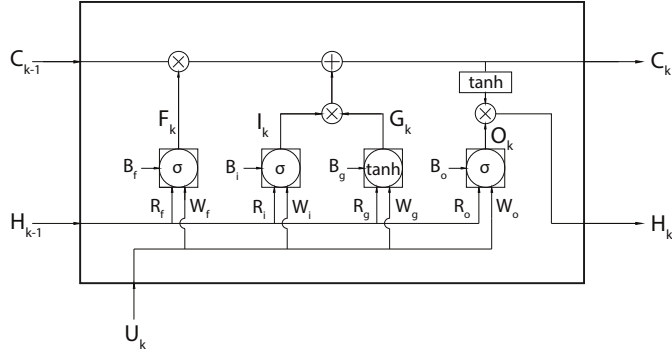


Fig. 2. Internal structure of the LSTM cell.

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad \text{and} \quad ReLU(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

This LSTM network can be represented as a state-space model as follows. The state vector or function,

$$X_k = \begin{bmatrix} C_k \\ H_k \end{bmatrix} = \begin{bmatrix} F_k * C_{k-1} + I_k * G_k \\ O_k * \tanh(C_k) \end{bmatrix} \quad (2)$$

and measurement function,

$$\hat{Y}_k = ReLU(X_k^T \times \begin{bmatrix} 0 \\ W_d \end{bmatrix} + B_d) \quad (3)$$

Note that H_k , and C_k are the outputs from the LSTM cell which are considered as primary states, and secondary states at time instant k as H_k itself is a function of C_k and the concatenation of H_k , and C_k is equivalent to the state vector X_k . The dimensions of $B_o, B_f, B_g, B_i, W_g, W_i, W_f, W_o, W_d$ are $N \times 1$, and R_i, R_o, R_f, R_g are $N \times N$. B_d is a 1×1 scalar. Where N represents the number of units in the LSTM layer.

The recurrent layer, i.e., the feedback layer represented by small circles in Fig. 1 keeps the information of the past input sequence and their dependencies. Hence, only input U at time instant k is sufficient as an external input to the network to produce output $\hat{Y}_k$.

The aforementioned network can be trained with the back-propagation method see Goodfellow et al. (2016), which uses a gradient descent iterative optimization algorithm to obtain trainable parameters, such as W, R , and B for all the gates. Specifically, the training goal is to minimize $\Delta Y = ||\hat{Y} - Y||_2$, where Y is the measured target data, and $\hat{Y}$ is the output of the LSTM model subject to the same input data U .

2.2 LSTM Training Dataset Generation

One of the crucial tasks in obtaining the mathematical system model using a neural network is to generate the training dataset, i.e., U and Y , which covers the entire working frequency and amplitude ranges, respectively. The training data must be unbiased towards specific ranges of frequency and amplitude to achieve a well-generalized model.

The process of generating the training data starts with creating input signals for the PEA (i.e., U). Considering the frequency and amplitude-dependent dynamics of PEA,

the drive input U to the PEA for training the LSTM model can be designed as,

$$U = \prod_{(a_i, f_i) \in S} a_i(\sin(2\pi f_i t)) + C, \quad t \in [0, 1/f_i]. \quad (4)$$

where C denotes the constant DC offset. $\prod$ denotes concatenation of all sinusoidal waves represented by (a_i, f_i) (i.e., (amplitude, frequency)) pairs from S , which is a set consisting p number of (a_i, f_i) pairs and obtained through k -means clustering (Lloyd (1982)). Though it is optimal to use all possible (a_i, f_i) pairs from the $(a - f)$ plane, it is not practical, and hence the use of k -means clustering ensures the best possible (a_i, f_i) pairs to obtain desired modeling accuracy. The value p is the cluster quantity determined through cluster evaluation techniques. Cluster evaluation criterion, such as Silhouette (Rousseeuw (1987), and Kaufman and Rousseeuw (2009)) and CalinskiHarabasz (Caliński and Harabasz (1974)) criterion, are good in evaluating the clusters for optimum p value. The use of a concatenated signal for training, which includes wide frequency and amplitude signals changing rapidly over time, can help obtain a generalized network model. The next step is to measure the corresponding PEA output Y by applying the designed input U to the commercial PEA system.

In order to capture the inverse dynamics of the PEA system, Y and U are used as the input and output of the LSTM model, respectively, in the training process described in Sec. 2.1.

2.3 Combined inverse LSTM and PEA system

Integrating the LSTM inversion model with the physical PEA system in series (represented by a dotted rectangle in Fig. 3) yields a nearly linear system if the LSTM model is accurate in capturing the PEA inverse dynamics. This linear system (referred to as the combined system in the rest of the paper) can be represented in general state-space format as follows,

$$\begin{aligned} X_{k+1} &= AX_k + Bu_k \\ y_k &= CX_k + Du_k \end{aligned} \quad (5)$$

where dimensions of A, B, C, D matrices depend on the degree of the system chosen by the user for better accuracy.

3. PREDICTIVE CONTROL

3.1 Model Predictive Control

The system model represented by (5) can be used by linear MPC for PEA trajectory tracking control (see Fig. 3).

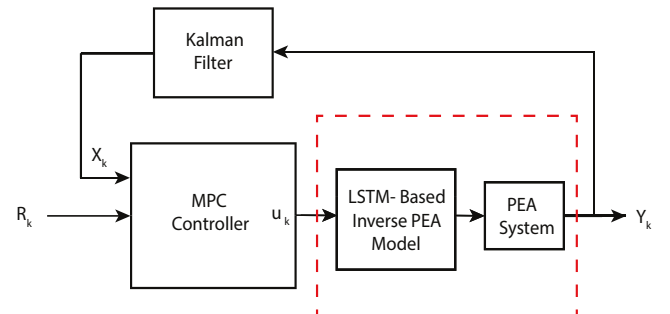


Fig. 3. Block diagram of MPC control framework.

The MPC with internal Kalman filter as a state estimator achieves this goal by predicting the future PEA trajectory $[\hat{y}_k, \hat{y}_{k+1}, \dots, \hat{y}_{k+H_p-1}]^T$ over the prediction horizon H_p and solving the quadratic optimization (QP) problem to obtain optimum drive input, U_{opt} , to the combined system, where U_{opt} is a future input series $[u_k, u_{k+1}, \dots, u_{k+H_c-1}]^T$, where H_c denotes the control horizon.

First, MPC predicts the future PEA trajectory as follows,

- i. Compute the innovation e_k

$$e_k = y_k - [CX_{k|k-1}]. \quad (6)$$

- ii. Update the state estimation to account for the current measurement

$$X_{k|k} = X_{k|k-1} + Me_k. \quad (7)$$

- iii. Compute state estimation for the next instant using updated states.

$$X_{k+1|k} = AX_{k|k} + Bu_{k|k} + Le_k. \quad (8)$$

where $X_{k|k-1}$ is a controller state estimates based on previous time instant, $k-1$. Y_k is the current PEA output. L and M are the filter gain and the innovation gain, respectively, obtained using Kalman Estimator.

Similarly, the i^{th} future state prediction $X_{k+i|k}$ and corresponding output $y_{k+i|k}$ can be obtained as follows,

$$X_{k+i|k} = AX_{k+i-1|k} + Bu_{k+i-1|k} + Le_k \quad (9)$$

$$y_{k+i|k} = CX_{k+i|k} \quad i \in [0, H_p - 1] \quad (10)$$

The future output equation i.e. (10) can be represented in matrix form with constant terms written separately as follows,

$$\hat{Y} = \alpha X_{k-1} + \beta U + \Psi u_{k-1}, \quad (11)$$

where

$$\hat{Y} = \begin{bmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k+H_p-1} \end{bmatrix}_{H_p \times 1} \quad U = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+H_c-1} \end{bmatrix}_{H_p \times 1}$$

The terms α, β, Ψ are defined as

$$\alpha = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{H_p} \end{bmatrix} \quad \psi = \begin{bmatrix} CB \\ CA^1 B \\ \vdots \\ CA^{H_p-1} B \end{bmatrix}$$

$$\beta = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{H_p-2} B & CA^{H_p-3} B & \cdots & 0 \end{bmatrix}$$

Next, the following QP is solved to obtain the optimum input U_{opt} .

$$\min_U \zeta = \zeta_y(u) + \zeta_{\Delta u}(u), \quad (12)$$

where

$$\zeta_y(u) = \sum_{i=1}^{H_p} [r_{k+i|k} - y_{k+i|k}]^2,$$

$$\zeta_{\Delta u}(u) = \sum_{i=1}^{H_c} \rho [u_{k+i|k} - u_{k+i-1|k}]^2.$$

Note that $r_{k+i|k}$ is the future reference trajectory to be tracked. $\zeta_y(u)$ and $\zeta_{\Delta u}(u)$ are the costs associated with reference tracking and manipulated variable move suppression, respectively. Specifically, the cost function can be simplified as,

$$\min_U \zeta = (R - \hat{Y})^T (R - \hat{Y}) + \rho U^T D^T D U, \quad (13)$$

with

$$D = \begin{bmatrix} 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}_{H_c \times H_c}$$

and $R = [r_{k+1}, r_{k+2}, \dots, r_{k+H_p}]^T$.

3.2 Kalman Estimator

As the combined system model formulated as (5) is nearly linear if the LSTM inversion model is accurate enough, linear steady-state Kalman state estimator is a suitable choice to use as the internal state estimator with MPC. Kalman estimator estimates the state of the system at each control instant as follows

$$\begin{aligned} X_{k+1} &= AX_k + Bu_k + w_k, \\ y_k &= CX_k + Du_k + v_k \end{aligned} \quad (14)$$

where w_k represents process disturbance, v_k is the measurement noise, and

$$E[w_k w_k^T] = Q, \quad E[v_k v_k^T] = R, \quad E[w_k v_k^T] = N$$

where E represents the estimates of matrices. Then, the states for the next time interval can be estimated as,

$$\hat{X}_{k+1|k} = A\hat{X}_{k|k-1} + Bu_k + L(y_k - C\hat{X}_{k|k-1} - Du_k). \quad (15)$$

Here, L is a filter gain matrix, and it is obtained by solving discrete Riccati equation such that,

$$L = (APC^T + N)(CPC^T + R)^{-1}, \quad (16)$$

with $P = E[\{X - \hat{X}\}\{X - \hat{X}\}^T]$. Then the current Kalman estimator updates the state estimation using all available measured PEA output up to y_k as,

$$X_{k|k} = X_{k|k-1} + M(y_k - C\hat{X}_{k|k-1} - Du_k), \quad (17)$$

where $M = PC^T(CPC^T + R)^{-1}$. Hence, $y_{k|k}$ becomes,

$$y_{k|k} = CX_{k|k} + Du_k. \quad (18)$$

4. EXPERIMENT RESULT AND DISCUSSION

To validate and demonstrate the proposed approach, the trained LSTM inversion model was first evaluated by

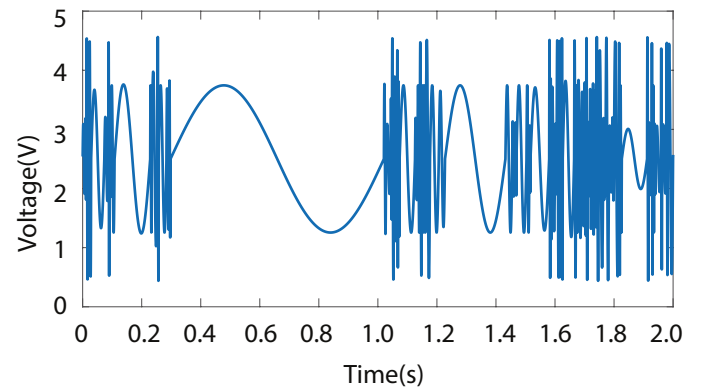


Fig. 4. Designed input signal U .

comparing the LSTM model output with the measured PEA (Nano-OP30, Mad City Labs) data. The LSTM inversion model was built and trained in python using Keras neural network library (Chollet et al. (2015)). Next, it was cascaded to commercial PEA (Nano-OP30, Mad City Labs) to form a combined system as described in Sec. 2.3. This system was further tested in Matlab Simulink (MathWorks, Inc.) on a workstation (Intel Xeon W-2125, 4.00GHz, 32 GB RAM) alongside the data acquisition system (NI PCIe-6353 by National Instruments) for tracking control.

The proposed LSTM model was built with 4 layers, including one input layer, two LSTM layers, and one fully connected dense layer, with 1, 100, 35, and 1 neuron (unit), respectively. For training the LSTM network, particularly for obtaining a training dataset, the amplitude range was kept at 0–5 V, and the frequency range was set to 1–300 Hz. The sampling frequency was set to 10 kHz throughout the entire experiment. The optimum cluster value p (the number of (a, f) pairs) was 110, obtained by using Silhouette cluster evaluation criterion (Rousseeuw (1987) and Kaufman and Rousseeuw (2009)). Selection of (a, f) pairs were done using k -means clustering (Lloyd (1982)). Finally, the training input was obtained using (4) as shown in Fig. 4. These inputs U and their corresponding outputs Y measured from the PEA were randomly categorized in training and testing datasets in a 0.8:0.2 ratio. Note that training and testing data were not the same, and hence assures the effectiveness of the model in generalization. Next, the experiments were conducted to analyze modeling accuracy and reference trajectory tracking.

First, to validate the accuracy of the LSTM in modeling the inversion dynamics of PEA, different reference input signals were applied only to the combined system (shown by a dotted rectangle in Fig. 3.) without any controller

in the application. Specifically, sinusoidal signals with the frequencies of 10, 50, and 117 Hz as well as complex signals such as variable amplitude sine sweep signal, stepwise signal, and broadband signal (with the frequency range of 1–300 Hz) generated using the approach described in Sec. 2.2 were used as references to demonstrate the performance and generalization ability of the trained model. Note that the frequencies of the testing reference inputs did not overlap with that of the training input. The performance of the trained LSTM inversion model for different sinusoidal inputs and complex inputs is shown in Fig. 5, and Fig. 6, respectively. Further, the modeling accuracy was computed as follows,

$$\xi_{mdl} = \frac{\|Y - \tilde{R}\|_2}{\|\tilde{R}\|_2}. \quad (19)$$

where $\tilde{R}$ and Y are the reference input signal given to the LSTM inversion model and output measured from the PEA system, respectively. The RMS modeling error ξ_{mdl} was 0.351%, 0.356%, 0.60% for the 10, 50, and 117 Hz sinusoidal inputs, and 1.2%, 0.937%, and 0.403% for sine sweep, stepwise signal, and the broadband signal, respectively. It is clear that the model is accurate enough to compensate for the nonlinear dynamics of PEA over the entire trained frequency and amplitude ranges: the input and output of the combined system are mostly the same. These results confirm the modeling accuracy of the LSTM inversion model.

Next, to demonstrate the performance of the proposed approach in a trajectory tracking application, the combined system described in Sec. 2.3 was augmented and controlled by linear MPC, as shown in Fig. 3. The linear MPC was designed in Matlab (MathWorks, Inc.) using the MPC control toolbox. To estimate the states of the system, a linear steady-state Kalman filter was designed in Matlab Simulink (MathWorks, Inc.). For MPC, the

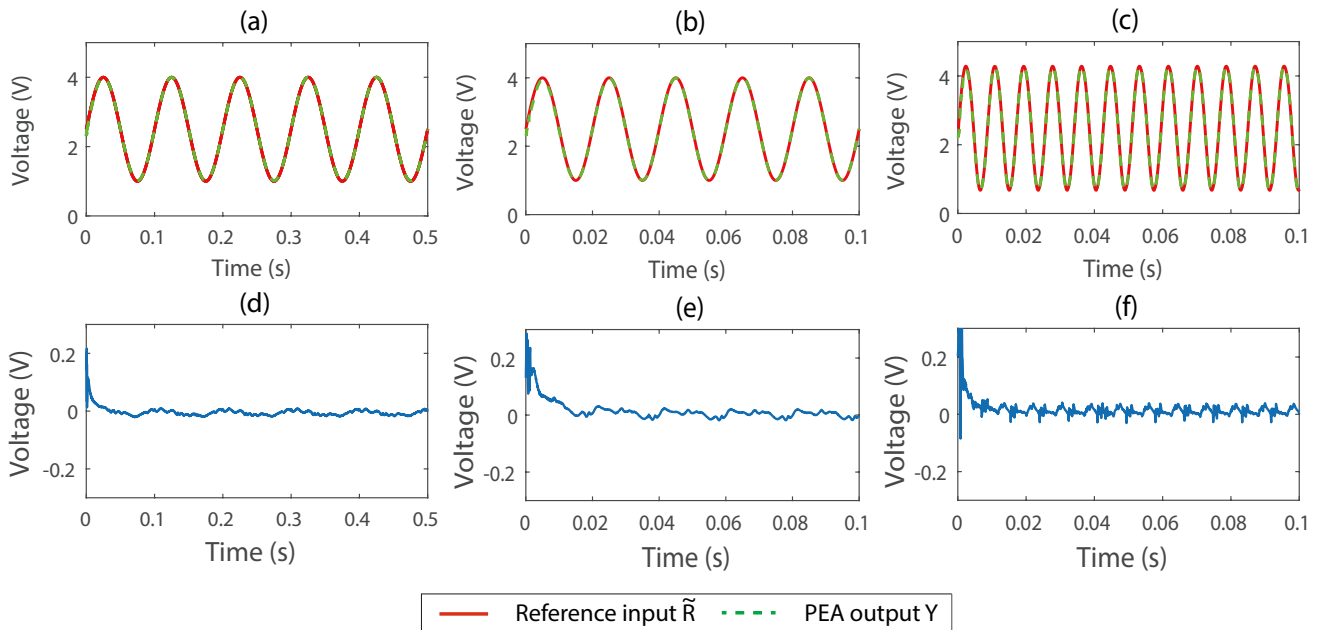


Fig. 5. Modeling Performance of LSTM. (a), (b), (c) are the comparisons of the reference input $\tilde{R}$ and PEA output Y when no controller in action, and (d), (e), (f) are the corresponding tracking errors $(Y - \tilde{R})$, when the reference inputs were 10 Hz, 50 Hz, and 117 Hz sinusoidal signals, respectively.

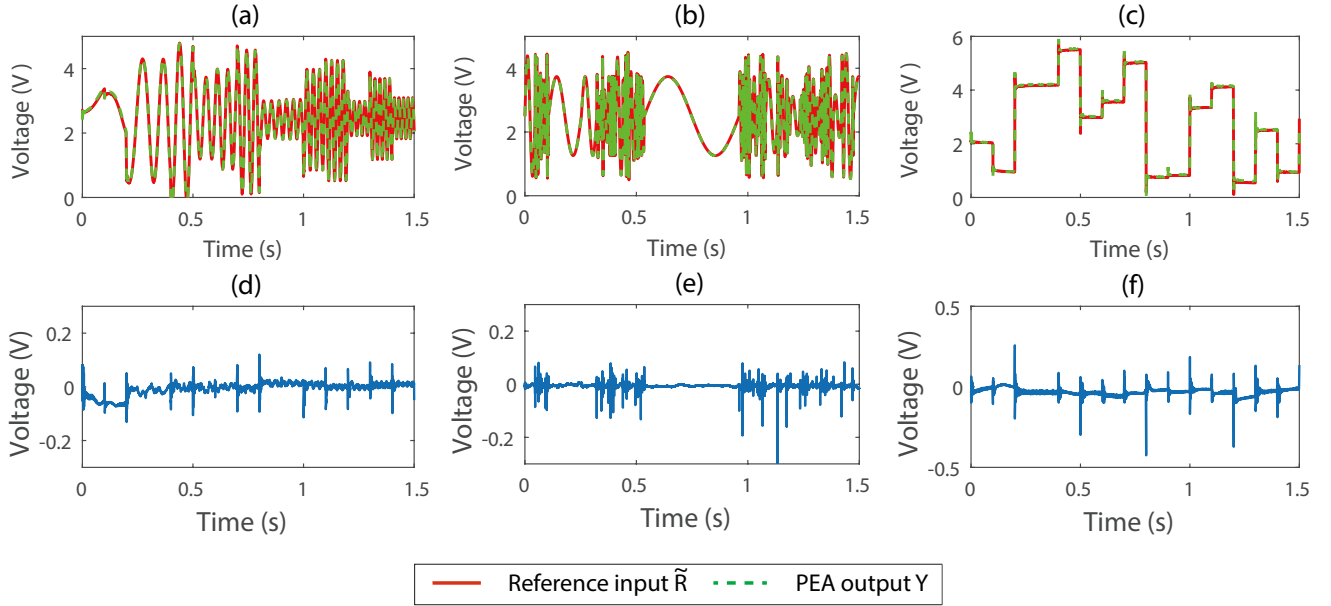


Fig. 6. Modeling Performance of LSTM for complex inputs. (a), (b), (c) are the comparisons of the reference input $\tilde{R}$ and PEA output Y when no controller in action, and (d), (e), (f) are the corresponding tracking errors ($Y - \tilde{R}$), when the reference inputs were sine sweep signal, broadband signal, stepwise signal, respectively.

prediction horizon H_p and the control Horizon H_c were set to 15 and 9, respectively.

The experiments were conducted to track different complex reference signals at a 10 kHz sampling rate. The tracking accuracy was validated by computing the RMS tracking error as follows,

$$\xi_{tr} = \frac{\|Y - R\|_2}{\|R\|_2}, \quad (20)$$

where R is PEA tracking reference and Y is the output of the PEA, respectively.

Fig. 7 shows the comparison between actual nonlinearities exhibited by PEA with nonlinearity compensation achieved using the proposed control approach. Note that the smaller area between two curves (charging curve and discharging curve) corresponds to better hysteretic compensation, and the linear nature of the curve is a measure

of the performance of the model to compensate for overall nonlinearities exhibited by PEA. From Fig. 7, it is clear that the devised method has realized notable nonlinearity compensation. Further, The tracking performance of the proposed LSTM based augmented control scheme for sinusoidal references of different frequencies (10, and 50 Hz), and variable amplitude sine sweep signal with frequency ranging from 1 to 200 Hz are shown in Fig. 8. The RMS tracking errors are shown in Table 1, where the tracking performance of conventional PID feedback subject to the same references is compared. PID is still one of the accurate control schemes for real-time commercial applications, and when tuned properly, it performs superior compared to modern control schemes. Hence, PID can be compared to measure the relative performance of the proposed method. It is clear that the tracking accuracy of the augmented predictive controller (APC) is much better than that of the PID feedback control for all desired tra-

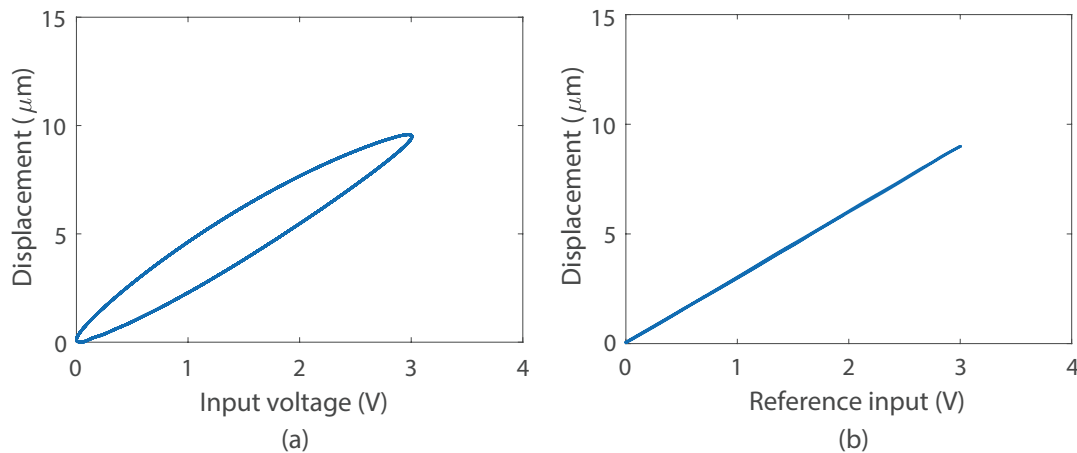


Fig. 7. Comparison of actual nonlinearities vs. experimental nonlinearities. (a) Nonlinearities exhibited by PEA, (b) Nonlinearities after compensation achieved using the proposed control approach.

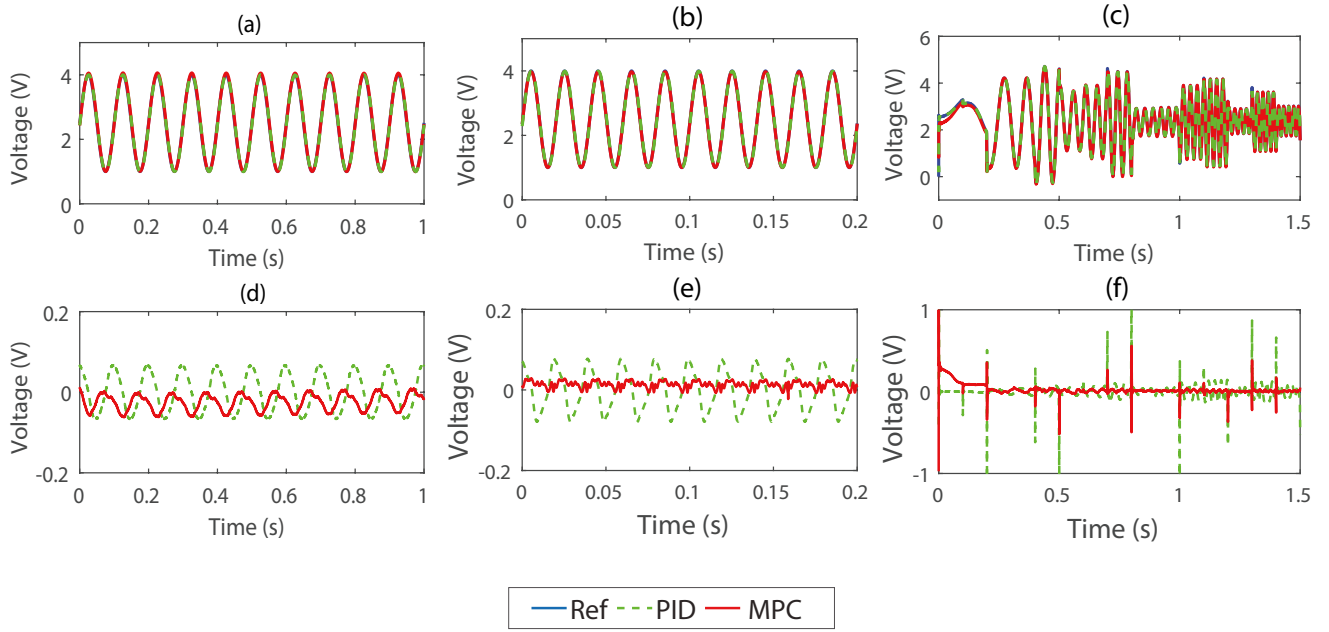


Fig. 8. Tracking performance comparison of MPC and PID. (a), (b), (c) are the PEA outputs, and (d), (e), (f) are the tracking errors for 10 Hz, 50 Hz, and variable amplitude sine sweep signal references, respectively.

jectories. Especially, for the variable frequency references, it outperformed PID, and the accuracy of the proposed approach was up to 5 times higher. We noticed that the performance of the APC was slightly downgraded as the operating frequency increased. This is associated to the small values of H_c and H_p chosen and limited bandwidth training dataset for system identification. The higher values of parameter H_p and H_c can help reduce the tracking error. However, it can impose additional computational complexity leading to difficulties in achieving real-time tracking with limited computing power.

Table 1. RMS tracking error comparison of the proposed augmented predictive controller (APC) and PID

$\xi_{tr}(\%)$	10 Hz	50 Hz	Sine sweep
APC	0.88	0.39	0.96
PID	1.74	1.86	5.13

In summary, the results above have demonstrated that the proposed LSTM inversion model is accurate in capturing the complex dynamics of PEAs. Further, when augmented using linear MPC, it achieved high trajectory tracking accuracy at higher speed and lowered computational cost allowing us to realize the better resolution of operation.

5. CONCLUSION

In this work, we proposed a system identification and control approach of PEA using the LSTM neural network for precise trajectory tracking. The LSTM was trained with the backpropagation method using a gradient descent optimization algorithm. The proposed LSTM inversion model effectively captured the PEA system dynamics for a wide range of operating frequencies. The LSTM network successfully modeled PEA dynamics with less than 2% modeling error. Furthermore, the obtained LSTM inversion model was cascaded to PEA and augmented using

a linear model predictive controller for PEA trajectory tracking. The results have shown that the proposed control approach can track the complex reference trajectories with less than 1% RMS tracking error. It is clear from the results that using the LSTM inversion model to capture the PEA dynamics and linear MPC to control reference tracking can achieve high accuracy and precision real-time trajectory tracking of PEA.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation (NSF) (CMMI-1634592) and Iowa State University.

REFERENCES

- Caliński, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1), 1–27.
- Cheng, L., Liu, W., Hou, Z.G., Yu, J., and Tan, M. (2015). Neural-network-based nonlinear model predictive control for piezoelectric actuators. *IEEE Transactions on Industrial Electronics*, 62(12), 7717–7727.
- Chollet, F. et al. (2015). Keras. <https://keras.io>.
- Ge, P. and Jouaneh, M. (1997). Generalized preisach model for hysteresis nonlinearity of piezoceramic actuators. *Precision engineering*, 20(2), 99–111.
- Gonzalez, J. and Yu, W. (2018). Non-linear system modeling using lstm neural networks. *IFAC-PapersOnLine*, 51(13), 485–489.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.
- Hansma, P., Elings, V., Marti, O., and CE, B. (1988/10/14). Scanning tunneling microscopy and atomic force microscopy: application to biology and technology. *Science*, 209–216.
- Haykin, S., Principe, J.C., Sejnowski, T.J., and Mcwhirter, J. (2007). Modeling large dynamical systems with dynamical consistent neural networks.

- Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735–1780.
- Huang, S., Tan, K.K., and Lee, T.H. (2009). Adaptive sliding-mode control of piezoelectric actuators. *IEEE Transactions on Industrial Electronics*, 56(9), 3514–3522.
- Jung, H. and Gweon, D.G. (2000). Creep characteristics of piezoelectric actuators. *Review of scientific Instruments*, 71(4), 1896–1900.
- Kaufman, L. and Rousseeuw, P.J. (2009). *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons.
- Liu, W., Cheng, L., Hou, Z.G., Yu, J., and Tan, M. (2015). An inversion-free predictive controller for piezoelectric actuators based on a dynamic linearized neural network model. *IEEE/ASME Transactions on Mechatronics*, 21(1), 214–226.
- Liu, Y., Shan, J., Gabbert, U., and Qi, N. (2013). Hysteresis and creep modeling and compensation for a piezoelectric actuator using a fractional-order maxwell resistive capacitor approach. *Smart Materials and Structures*, 22(11), 115020.
- Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2), 129–137.
- Mollaeian, K., Liu, Y., Bi, S., Wang, Y., Ren, J., and Lu, M. (2018). Nonlinear cellular mechanical behavior adaptation to substrate mechanics identified by atomic force microscope. *International journal of molecular sciences*.
- Mrad, R.B. and Hu, H. (2002). A model for voltage-to-displacement dynamics in piezoceramic actuators subject to dynamic-voltage excitations. *IEEE/ASME transactions on mechatronics*, 7(4), 479–489.
- Narendra, K.S. and Parthasarathy, K. (1991). Gradient methods for the optimization of dynamical systems containing neural networks. *IEEE Transactions on Neural networks*, 2(2), 252–262.
- Nguyen, L. and Chen, X. (2017). Discrete time quasi sliding mode control for piezo-actuated positioning systems: A prescribed performance control approach. *IFAC-PapersOnLine*, 50(1), 5121–5126.
- Pang, B., Zha, K., Cao, H., Shi, C., and Lu, C. (2019). Deep rnn framework for visual sequential applications. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 423–432.
- Pascanu, R., Gulcehre, C., Cho, K., and Bengio, Y. (2013). How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026*.
- Rana, M.S., Pota, H.R., and Petersen, I.R. (2014). The design of model predictive control for an afm and its impact on piezo nonlinearities. *European Journal of Control*, 20(4), 188–198.
- Rousseeuw, P.J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20, 53–65.
- Schäfer, A.M. and Zimmermann, H.G. (2006). Recurrent neural networks are universal approximators. In *International Conference on Artificial Neural Networks*, 632–640. Springer.
- Utkin, V. and Lee, H. (2006). Chattering problem in sliding mode control systems. In *International Workshop on Variable Structure Systems, 2006. VSS'06.*, 346–350. IEEE.
- Wei, C., Zhang, H., Tao, L., Li, W., and Shi, H. (1996). Compensation method based on formularizing hysteresis of piezoelectric tube scanners. *Review of scientific instruments*, 67(10), 3594–3598.
- Xie, S. and Ren, J. (2018). Note: Precision control of nano-positioning stage: An iterative learning-based model predictive control approach. *Review of Scientific Instruments*, 89(7), 076103.
- Xie, S. and Ren, J. (2019a). High-speed afm imaging via iterative learning-based model predictive control. *Mechatronics*, 57, 86–94.
- Xie, S. and Ren, J. (2019b). Recurrent-neural-network-based predictive control of piezo actuators for precision trajectory tracking. In *2019 American Control Conference (ACC)*, 3795–3800. IEEE.
- Yoichi, M. (2006). Applications of piezoelectric actuator. *NEC Technical Journal*.