2016 IEEE 55th Conference on Decision and Control (CDC)
ARIA Resort & Casino
December 12-14, 2016, Las Vegas, USA

Learning Control for Task Specific Industrial Robots

Chung-Yen Lin[1], Wenjie Chen[2], and Masayoshi Tomizuka[1]

Abstract— **Iterative learning control (ILC) is a strategy that allows a control system to improve its performance by making use of the error signals collected from previous iterations. A prerequisite of using ILC is that the output reference has to be repetitive from trial to trial. A full run of ILC training (taking non-negligible time) is needed when there exist small changes in the reference signal. This paper introduces a new approach to extrapolate the converged ILC policies to previously unseen tracking problems. A time-frequency domain mapping is constructed to approximate the ILC policy for a group of trajectories used in a particular task, say spot welding. We also introduce the idea of feature-frequency space, where the ILC policies from different trajectories can be encoded into a single model. This model can generate a control policy that performs comparably to the ILC policy while having the advantage of not requiring a full training for a new trajectory. The proposed method implemented on a FANUC R-2000*iC* robot achieved 31.6% of vibration reduction whereas the standard ILC (i.e., with a full training for each particular trajectory) achieved 34.6% of vibration reduction.**

I. INTRODUCTION

Iterative learning control (ILC) [1]–[3] is a useful tracking control method for robot systems that repeatedly execute the same task. The idea is to incorporate the tracking errors from previous iterations to generate a control update (usually as a feedforward) to improve the system performance in the next iteration. The success of the ILC relies on the output reference being repetitive from trial to trial. This condition limits the possible applications of learning control. Namely, a full run of ILC training is needed when there exist changes in the initial condition, the setpoints, and the total traveling time of trajectory. A full training is not preferred since it takes non-negligible engineering efforts to setup the required sensors for learning and to do performance validation. Therefore, it is desirable to develop a learning control method that serves a group of motions with a single learning action.

A possible solution to find a mapping between the reference signal and the ILC update for a group of trajectories used in a particular task (for example, spot welding). A related work called the setpoint variation learning control was presented in [4]. It used a least-squares estimate of finite impulse response (FIR) model to describe the nominal behavior of the ILC. It was shown to be useful for linear systems since most characteristics of the ILC law are simple enough to be captured by a FIR model. In the case of highly nonlinear systems like robot manipulators, we may need a nonlinear model to describe this mapping. In [5], the authors proposed to use a neural network to construct a nonlinear mapping between ILC commands and the system states. However, as it is a static mapping, the method cannot properly describe the dynamic properties in the ILC policy.

This paper presents a learning controller in the time-frequency domain, where the control policy is considered as a complex function that maps the reference signal to the reference update. The idea of feature-frequency space is introduced to encode the ILC policies from different trajectories to a single non-parametric model. With this model, we are able to use the previously learned ILC data to generate a control policy for a previously unseen reference signal. The controller then can be transformed into the time domain to update the reference signal. The proposed method is validated on a FANUC R-2000*i*C robot.

II. PRELIMINARIES

A. Baseline Controller

A standard control structure for industrial manipulators [6, 7] is shown in Fig 1. It consists of a feedforward controller and a feedback controller. The feedforward controller is used to generate required motor torques for trajectory following, while the feedback controller is used to compensate disturbances and model uncertainties. For industrial robots with indirect drive mechanisms, the motor side behavior may often deviate from the load side (i.e., end-effector) behavior due to the robot joint dynamics. In this case, the motor side sensors alone may not be sufficient for achieving good control performance on the load side.

B. Iterative Learning Control

A state-of-the-art solution is to equip a inertial measurement unit (IMU) to measure the end-effector motion (in particular the vibration) and use this information to control the robot [8, 9]. To be precise, an ILC is designed to generate a reference update for achieving better output tracking performance. The structure of the ILC used in this paper is shown in Fig. 2. Here we assume that the load side position information is accessible. It can be achieved by estimating from the accelerometer readings. Details of the ILC design can be found in [8, 9]. In the rest of this paper, we will step by step design a learning controller that is able to generate a reference update performing like the one generated by the ILC.

*This work was supported by FANUC Corporation, Japan.

[1]Chung-Yen Lin and Masayoshi Tomizuka are with Department of Mechanical Engineering, University of California, Berkeley, CA 94720, USA. Email: {chung_yen, tomizuka} at berkeley.edu

[2] Wenjie Chen is with FANUC Corporation, Oshino-mura, Yamanashi-ken, Japan. Email: wjchen at berkeley.edu

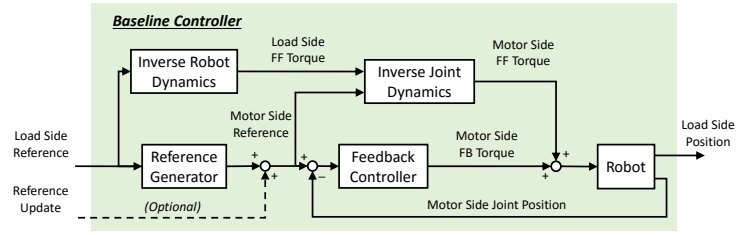


Fig. 1: Structure of the baseline controller

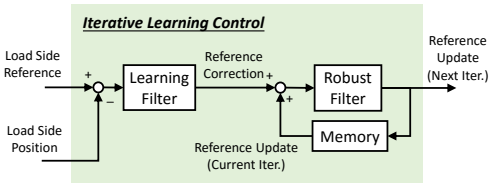


Fig. 2: Structure of the iterative learning control

III. TIME-FREQUENCY ANALYSIS

A. LTV Systems in Time-frequency Domain

The idea to learn a control policy in the time-frequency domain using the previously learned ILC data. To do this, we first derive a multiplicative representation of linear time varying (LTV) systems in the time-frequency domain. The formulation is similar to the spectral modifications [10], but we approach it from a control system viewpoint. This would allow us to easily design and learn a controller in the time-frequency domain.

Given a time domain signal $y(k)$, we perform a time-frequency analysis by swiping the signal with a window function and running a discrete time Fourier transform (DTFT) for each window. This operation is called the discrete time short-time Fourier transform (DT-STFT) and it is formulated as:

$$Y(n,f) = \sum_{k=-\infty}^{\infty} w(n-k)y(k)e^{-jfk} \tag{1}$$

where $Y(n,f)$ is the time-frequency domain representation of the timed signal $y(k)$, n is the time index, f is the frequency, and w is the window function with a window length n_w. The hop size (i.e., the number of time steps from one window frame to the next) is assumed to be one in this paper.

Assume that the signal $y(k)$ is generated from a LTV system with an impulse response $h(k,\tau)$ at time step k and an input signal $x(k)$, the input-output relation can be described by [11]:

$$y(k) = \sum_{r=-\infty}^{\infty} h(k, k-r)x(r) \tag{2}$$

Substituting (2) into (1) yields:

$$Y(n,f) = \sum_{k=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} w(n-k)h(k,k-r)x(r)e^{-jfk}$$

Then by setting $s = k - r$, we have:

$$Y(n,f) = \sum_{s=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} w(n-s-r)h(s+r,s)x(r)e^{-jf(s+r)} \tag{3}$$

By assuming that the window length is longer than the impulse response, we have:

$$h(s+r,s)w(n-s) \simeq h(s+r,s)w(n)$$

This results in an approximate of $Y(n,f)$:

$$Y(n,f) \simeq \sum_{s=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} w(n-r)h(s+r,s)x(r)e^{-jf(s+r)}$$

It is worth pointing out that, for linear time invariant (LTI) systems (i.e., $h(s+r,s) = h(s)$), the above equation can be simplified as a multiplicative relation of the transfer function and the input signal [12], [13]:

$$\begin{aligned} Y(n,f) &\simeq \sum_{r=-\infty}^{\infty} w(n-r)x(r)e^{-jfr} \sum_{s=-\infty}^{\infty} h(s)e^{-jfs} \\ &= X(n,f)H(f) \end{aligned}$$

However, in the case of LTV systems, an additional assumption needs to be made to get this multiplicative relation. Namely, by assuming that the impulse response is invariant within a window, we have:

$$h(s+r,s)w(n-r) \simeq h(n,s)w(n-r) \tag{4}$$

Substituting (4) into (3) yields:

$$\begin{aligned} Y(n,f) &\simeq \sum_{s=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} w(n-r)h(n,s)x(r)e^{-jf(s+r)} \\ &= \sum_{r=-\infty}^{\infty} w(n-r)x(r)e^{-jfr} \sum_{s=-\infty}^{\infty} h(n,s)e^{-jfs} \\ &= X(n,f)H(n,f) \end{aligned}$$

This multiplicative relation is simple enough for controller design.

B. Implementation of STFT

In practice, the time-frequency analysis shown in (1) is not realizable. Therefore, instead of running discrete time Fourier transform in each window, we run discrete Fourier transform (DFT). This gives us the discrete STFT as follows [10]:

$$Y(n,l) = e^{-j\omega_l n}\left(\text{DFT}\left\{\text{SHIFT}\{\mathbf{y}, -n\} \circ \mathbf{w}\right\}\right)_l \tag{5}$$

where $\mathbf{y}$ and $\mathbf{w}$ are respectively the vector representations of the time domain signals $y(k)$ and $w(k)$, SHIFT is the time shifting operator[1], $\circ$ is the element-wise multiplication operator, l is the index for the frequency bin, and ω_l is the l-th normalized frequency.

Similarly, to recover the time domain signal, we perform inverse DFT on each spectrum and shift it back to where it belongs. The algorithm is as follows:

$$\mathbf{y} = \sum_n \text{SHIFT}\left\{\text{DFT}^{-1}\{\mathbf{Y}_n\}, n\right\} \tag{6}$$

where $\mathbf{Y}$ is the matrix representation of the function $Y(n,l)$, while $\mathbf{Y}_n$ is the n-th row of the matrix $\mathbf{Y}$. Hereafter, the transformations (5) and (6) will be denoted as $\mathbf{Y} = \text{STFT}\{\mathbf{y}\}$ and $\mathbf{y} = \text{STFT}^{-1}\{\mathbf{Y}\}$.

IV. CONTROLLER DESIGN

With the multiplicative relation in the time-frequency domain, it is natural to think of an ILC policy as a complex function that maps the reference signal to the reference update in this space. Now the remaining problem becomes how to generate a time-frequency domain controller that behaves like an ILC for a given trajectory.

In order to encode the ILC policies from different trajectories to a single model, we have to consider the ILC as a linear parameter-varying (LPV) system [14] instead of a LTV system at this point. Namely, the frequency response is varying as a function of certain parameters, where the parameters are assumed to be a function of high level features of the motion. This allows us to transform the ILC policy at each time step into the feature-frequency domain and to perform a policy learning in this space. Once a feature-frequency domain policy is obtained, it can be used to generate a controller for a previously unseen reference (as visualized in Fig. 3). Then the time-frequency domain policy can be obtained directly by flatting and scaling this two-dimensional manifold in the feature-frequency space. Details can be found in the following sections.

A. Training Data Structure

The policy learning is performed in a joint-by-joint manner (i.e., a decentralized policy learning). We collect the converged ILC data from different trajectory tracking problems and partition the data into n_d sets. Each training data set $D^{(i)}$ is a tuple with three elements $(z^{(i)}, \mathbf{v}^{(i)}, \mathbf{u}^{(i)})$ where $z^{(i)} \in \mathbb{R}^{n_z}$ is the feature vector, $\mathbf{v}^{(i)} \in \mathbb{R}^{n_f}$ is the vector representation of a sequence of velocity commands in the configuration space, $\mathbf{u}^{(i)} \in \mathbb{R}^{n_f}$ is the vector representation of the ILC commands in the configuration space, and n_f is the data length. The feature vector can be as simple as the robot kinematics (i.e., position and velocity). It can also consist of high level features like robot inertia or the magnitude of reference at certain frequency components.

[1]SHIFT$\{a, b\}$ is an operator that delays the signal a by b steps.

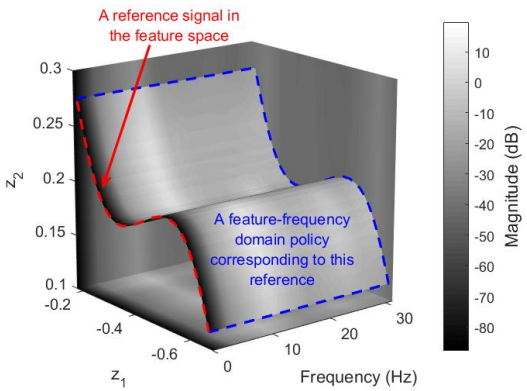


Fig. 3: Search a time-frequency domain feedforward controller in the feature-frequency domain (z_1 and z_2 are features)

B. Policy Learning

The proposed method tries to learn a controller that behaves like the ILC control law in the feature-frequency domain. The method proceeds as follows. First, for each dataset $D^{(i)}$, we apply the DFT on $\mathbf{v}^{(i)}$ and $\mathbf{u}^{(i)}$, and divide the output spectrum by the input spectrum to get the empirical transfer function $\mathbf{c}^{(i)} \in \mathbb{C}^{n_f}$ corresponding to the vector of normalized frequency bins $\mathbf{f} = [0, \frac{2\pi}{n_f}, \frac{4\pi}{n_f}, \ldots, \frac{2(n_f-1)\pi}{n_f}]^{\mathrm{T}} \in \mathbb{R}^{n_f}$. This empirical transfer function can be viewed as a linear approximate of the ILC policy obtained at the feature point $z^{(i)}$. Therefore, the policy learning problem becomes to train a model such that, given an arbitrary feature point and a frequency of interest, it returns the corresponding control policy in terms of frequency domain magnitudes and phases. The model used to approximate the policy is shown as follows:

$$g \sim \mathcal{GP}(m(x;\theta_1), k(x,x';\gamma_1)) \tag{7}$$
$$h \sim \mathcal{GP}(m(x;\theta_2), k(x,x';\gamma_2)) \tag{8}$$
$$|c| \sim \mathcal{N}(g, \sigma_1^2) \tag{9}$$
$$\angle c \sim \mathcal{N}(h, \sigma_2^2) \tag{10}$$

where $\mathcal{GP}$ is the Gaussian process regression (GPR) model [15], $x = [z^{\mathrm{T}}, f]^{\mathrm{T}} \in \mathbb{R}^{n_z+1}$ is the input vector consisting of a feature vector and a frequency bin, $m(x)$ and $k(x,x')$ are respectively the mean function and the kernel function to be designed, while $\{\theta_i, \sigma_i, \gamma_i\}$ are hyperparameters that can be computed by maximizing the marginal likelihood of the model with the training data. In this paper, the mean function is set as the constant mean (i.e., $m(x;\theta) = \theta$)[2] and the kernel function is set as the squared exponential covariance function:

$$k(x,x';\gamma) = \exp\left(-\gamma(x-x')^{\mathrm{T}}(x-x')\right)$$

We then construct the training data for the input channel

[2]An alternative choice of the mean function is a FIR model that approximates the ILC policy.

as:

$$\mathbf{x}^{(i)} = \begin{bmatrix} \begin{bmatrix} z^{(i)} & z^{(i)} & \cdots & z^{(i)} \end{bmatrix} \\ \mathbf{f}^{\mathrm{T}} \end{bmatrix} \in \mathbb{R}^{(n_z+1)\times n_f}$$

Then stacking all data together yields the overall input/output data matrices:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \cdots & \mathbf{x}^{(n_d)} \end{bmatrix} \in \mathbb{R}^{(n_z+1)\times n_d n_f}$$

$$|\mathbf{C}| = \begin{bmatrix} |\mathbf{c}^{(1)}| \\ \vdots \\ |\mathbf{c}^{(n_d)}| \end{bmatrix} \in \mathbb{R}^{n_d n_f} \qquad \angle\mathbf{C} = \begin{bmatrix} \angle\mathbf{c}^{(1)} \\ \vdots \\ \angle\mathbf{c}^{(n_d)} \end{bmatrix} \in \mathbb{R}^{n_d n_f}$$

By substituting the training data into the Gaussian process models shown in (7)-(10), we have the joint distributions $|\mathbf{C}| \sim \mathcal{N}(M(\mathbf{X};\theta_1), K(\mathbf{X},\mathbf{X};\gamma_1))$ and $\angle\mathbf{C} \sim \mathcal{N}(M(\mathbf{X};\theta_2), K(\mathbf{X},\mathbf{X};\gamma_2))$ with:

$$M(\mathbf{X};\theta) = \begin{bmatrix} m(x_1;\theta) \\ \vdots \\ m(x_n;\theta) \end{bmatrix}$$

$$K(\mathbf{X},\mathbf{X};\gamma) = \begin{bmatrix} k(x_1,x_1;\gamma) & \cdots & k(x_1,x_n;\gamma) \\ \vdots & \ddots & \vdots \\ k(x_n,x_1;\gamma) & \cdots & k(x_n,x_n;\gamma) \end{bmatrix}$$

where x_i is the i-th row in the data matrix $\mathbf{X}$, $M(\mathbf{X};\theta)$ is the vector representation of the mean function $m(x;\theta)$ with the i-th entry being $m(x_i;\theta)$, and $K(\mathbf{X},\mathbf{X}';\gamma)$ is the matrix representation of the kernel function $k(x,x';\gamma)$ with the (i,j)-th entry being $k(x_i,x'_j;\gamma)$. The mean functions and the kernel functions will be pre-computed for later use when generating a new control policy. A Bayesian network representation of the proposed GPR model is shown in Fig. 4a.

C. Generation of a Time-frequency Domain Controller

Given a previously unseen velocity reference $\mathbf{v}_* \in \mathbb{R}^{n_v}$, we first apply the STFT to obtained a spectrogram:

$$\mathbf{V}_* = \mathrm{STFT}\{\mathbf{v}_*\} \in \mathbb{C}^{n_t \times n_f}$$

where n_f and n_t are the sizes of this spectrogram. The magnitude spectrogram and the phase spectrogram can then be computed by:

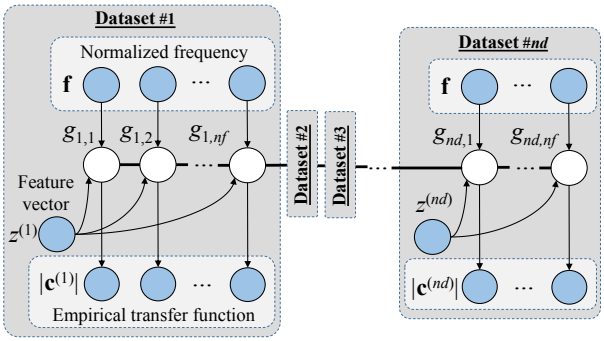
$$\mathbf{V}_* = |\mathbf{V}_*| e^{j\angle\mathbf{V}_*}$$

Note that every entry of V_* would correspond to a feature point z_* and a frequency component f_*. Now we consider the control policy $|c_*|$ and $\angle c_*$ corresponding to $x_* = [z_*^{\mathrm{T}}, f_*]^{\mathrm{T}}$ as random variables with the following joint distributions:

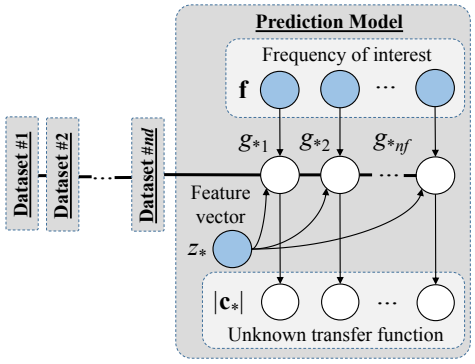
$$\begin{bmatrix} |\mathbf{C}| \\ |c_*| \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} M(\mathbf{X}) \\ m(x_*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{X},\mathbf{X}) & K(\mathbf{X},x_*) \\ K(x_*,\mathbf{X}) & k(x_*,x_*) \end{bmatrix} + \Sigma \right) \tag{11}$$

$$\begin{bmatrix} \angle\mathbf{C} \\ \angle c_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} M(\mathbf{X}) \\ m(x_*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{X},\mathbf{X}) & K(\mathbf{X},x_*) \\ K(x_*,\mathbf{X}) & k(x_*,x_*) \end{bmatrix} + \Sigma \right) \tag{12}$$

where $\Sigma = \sigma^2 I$.



(a) GPR model for the ILC policy



(b) Prediction model

Fig. 4: The proposed GPR model in a Bayesian network representation. The controller magnitude $|c|$ is used as an example.

A graphical representation of this prediction model is shown in Fig. 4b.

Since (11)-(12) are multivariate Gaussian distributions, we can compute the conditional means of $|c_*|$ and $\angle c_*$ given $x_* = [z_*^{\mathrm{T}}, f_*]^{\mathrm{T}}$, $\mathbf{X}$, and $\mathbf{C}$:

$$|c_*| = m(x_*) + K(x_*,\mathbf{X})(K(\mathbf{X},\mathbf{X}') + \Sigma)^{-1}(|\mathbf{C}| - M(\mathbf{X}))$$
$$\angle c_* = m(x_*) + K(x_*,\mathbf{X})(K(\mathbf{X},\mathbf{X}') + \Sigma)^{-1}(\angle\mathbf{C} - M(\mathbf{X}))$$

This gives an estimate of c_* corresponding to every entry of $\mathbf{V}_*$ and therefore can construct a complex control policy $\mathbf{C}_*$.

Now the magnitude spectrogram of feedforward signal can be generated using the multiplicative relation in the time-frequency domain:

$$|\mathbf{U}_*| = |\mathbf{C}_*| \circ |\mathbf{V}_*|$$

where $\mathbf{U}_* \in \mathbb{C}^{n_t \times n_f}$ is the time-frequency space feedforward signal. Similarly, the phase spectrogram $\angle\mathbf{U}_*$ can be computed by:

$$\angle\mathbf{U}_* = \angle\mathbf{C}_* + \angle\mathbf{V}_*$$

The time domain control input can be recovered by performing the inverse STFT on $\mathbf{U}_*$:

$$\mathbf{u}_* = \mathrm{STFT}^{-1}\{\mathbf{U}_*\}$$

V. FILTER DESIGN

Ideally, passing the reference spectrogram through the policy will results in the approximate ILC commands. However, as the training data only explore the state space is a specific way, it is risky to use an estimated control policy to control the system. The control inputs will need to pass though a coherence filter and a lowpass filter for enhancing the robustness of the algorithm. The modified control law (for the magnitude part) is as follows:

$$|\mathbf{U}_*| = \mathbf{Q} \circ \mathbf{S} \circ |\mathbf{C}_*| \circ |\mathbf{V}_*|$$

where $\mathbf{Q}$ and $\mathbf{S}$ are respectively the lowpass filter and the coherence filter, which will be introduced in the following sections.

A. Coherence Filter

Note that at each time index n on the spectrogram, the DFT is computed by a sequence of velocity commends $\mathbf{v}_*(n) \in \mathbb{R}^{n_f}$ corresponding to a feature point $z(n) \in \mathbb{R}^{n_z}$. It is of interest to find a training data set which has the feature point closest to the current feature $z(n)$:

$$i^*(n) = \arg\min_i \|z(n) - z^{(i)}\|_2$$

The idea of coherence filter is to compare the pattern of the testing data $\mathbf{v}_*(n)$ to that of the training data $\mathbf{v}^{(i^*(n))}$. To be precise, the magnitude of the coherence filter at time step n and frequency bin l can be computed as follows:

$$S(n,l) = \sqrt{\text{Coh}\{\mathbf{v}_*(n), \mathbf{v}^{(i^*(n))}\}(l)} \tag{13}$$

where Coh is the coherence function [16]:

$$\text{Coh}\{\mathbf{y}, \mathbf{z}\}(l) = \frac{|\Psi_{\mathbf{yz}}(l)|^2}{\Psi_{\mathbf{yy}}(l)\Psi_{\mathbf{zz}}(l)}$$

while Ψ is the spectral density function [17]. This filter returns one when the testing data $\mathbf{v}_*(n)$ and the training data $\mathbf{v}^{(i^*(n))}$ are identical. It returns zero when two signals are completely unrelated. A matrix representation of the coherence filter is defined as $\mathbf{S} \in \mathbb{R}^{n_t \times n_f}$, where the (n,l)-entry of the matrix $\mathbf{S}$ is equal to $S(n,l)$.

B. Zero-phase Filter

Note that there exist uncertainties when computing empirical transfer function with the training data. Therefore, a zero-phase filter is applied in the time-frequency domain to suppress high frequency control actions. It can be achieve by defining a filter matrix $\mathbf{Q} \in \mathbb{R}^{n_t \times n_f}$ to have the (n,l)-th entry equals to:

$$Q(n,l) = \begin{cases} 1 & \text{if } f(l) \le \omega_c \\ \epsilon & \text{if } f(l) > \omega_c \end{cases}$$

where ϵ is chosen as 10^{-6} in this paper.

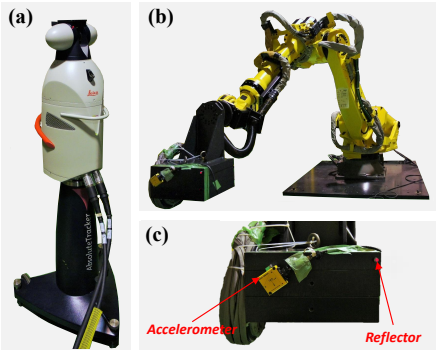


Fig. 5: FANUC R-2000*i*C/210F robot setup

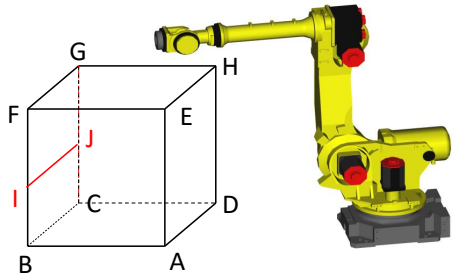


Fig. 6: Box motion with application to spot welding

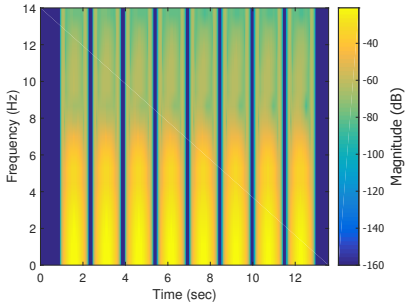
VI. CASE STUDY

A. System Setup

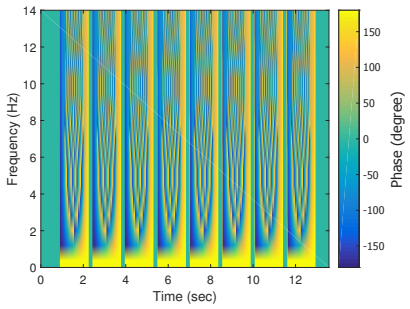
All the experiments in this paper are performed on the FANUC R-2000*i*C/210F robot setup shown in Fig. 5. The FANUC R-2000*i*C is a large size robot designed for heavy duty applications such as material handling and spot welding. It has six joints driven by indirect drive mechanisms. Each joint is equipped with a built-in motor encoder. An accelerometer is equipped to measure the load side acceleration information. In this paper, we assume access to the accelerometer for the ILC, but not for the proposed algorithm. In addition, a three-dimensional position measurement system, the Leica Absolute Tracker AT901B, is utilized to measure the end-effector tool center point (TCP) position in Cartesian space. The Leica AT901B is only used for performance validation. The motor encoders and the robot controller ran at 1kHz. The accelerometer and the Leica AT901B ran at 1kHz. A 210kg payload is mounted on the last joint to mimic the real-world working conditions. A reflector for the Leica AT901B is magnetically mounted to the payload. Detailed specification of the FANUC R-2000*i*C/210F robot can be found in [18].

B. Experimental Results

This paper considers spot welding as a case study of the proposed time-frequency domain feedforward learning (TFFL) method. The training data is collected from a box-motion as shown in Fig. 6. It consists of eight to ten spot welding points on each edge, and the robot motion between each pair of points is designed as a third order smooth time-optimal trajectory. The testing trajectory consists of eight

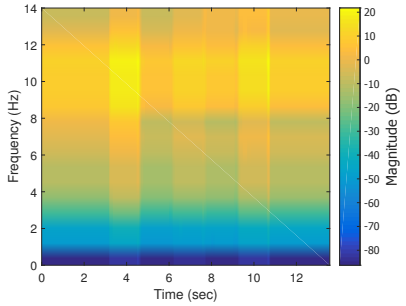


(a) Magnitude spectrogram

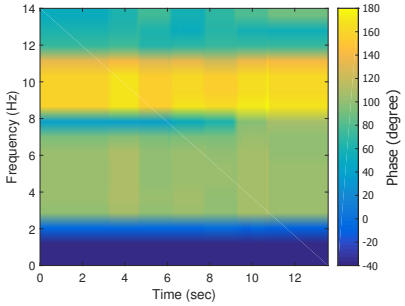


(b) Phase spectrogram

Fig. 7: Velocity commands in the time-frequency domain



(a) Magnitude spectrogram



(b) Phase spectrogram

Fig. 8: Control policy in the time-frequency domain

spot welding points on the $\overline{\text{IJ}}$ edge, which are not included in the training dataset.

The time-frequency domain reference signal of the testing trajectory is shown in Fig. 7. The window length is set as $n_w = 256$ (i.e., 0.256 sec). These spectrograms are used as inputs of the Gaussian process regression model to generate the control policy as shown in Fig. 8. The corresponding coherence filter can also be obtained by running (13). The resulting filter spectrograms are shown in Fig. 9. It is seen that the filter would deactivate the reference update when the similarity between the testing data and the training data is minor. Therefore, it is expected that when the testing data is totally isolated from the training data, the coherence filter may have small values over the entire spectrogram. The learning controller will not generate any reference update in such case because it is not confident to do so. This provides us a certain level of stability since the controller will degenerate to the baseline controller when the testing data is uncorrelated to the training data.

After passing the reference signal though the controller and the filter, a time domain reference update can be generated. This new feedforward control law was then implemented on the FANUC robot. The experimental results are shown in Fig. 10. As expected, the system with the baseline controller (gray dash-dot) has large tracking errors whenever the acceleration reference has large magnitude. Often times, these errors cannot be properly compensated without load side sensors. However, the TFFL (blue solid) can effectively suppress the errors since the control policy is generated based on the ILC policy over similar trajectories. To further evaluate the proposed method, we perform the

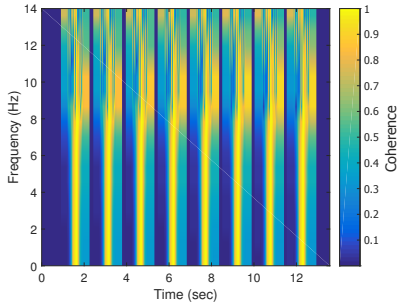


Fig. 9: Coherence filter

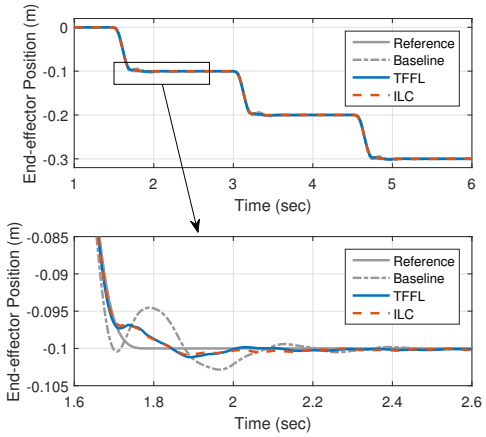


Fig. 10: Position tracking results

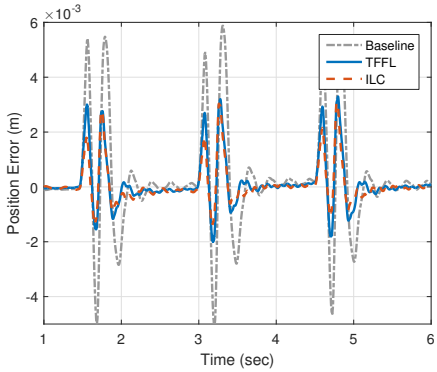
standard ILC (red dash) on this testing trajectory. Note that this ILC information is only used for validation, the TFFL does not have access to it in the training phase. It is seen that the behavior of the TFFL is similar to the one of the ILC, while the advantage is that the TFFL does not require an additional learning action every time when the trajectory is changed.

To further analyze the proposed method, Fig. 11 shows all the position errors, the velocity errors, and the acceleration errors along the moving direction. Among them the acceleration profile is of particular interest since it is a measure of vibration. Again, it is seen that both the ILC and the TFFL provide significant improvement in the tracking performance, while the ILC slightly outperforms the TFFL. This follows the intuition that the TFFL is an approximate of the ILC and therefore cannot perform better than the ILC.

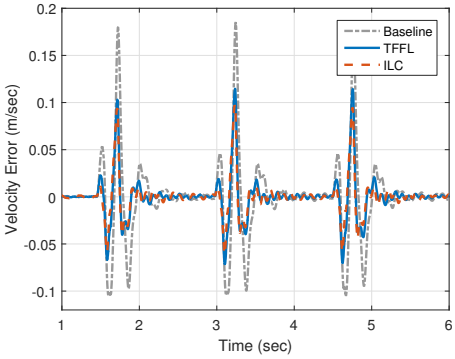
A quantitative comparison can be found in Table I. It is shown that the standard ILC achieved 34.6% of vibration reduction (i.e., acceleration error reduction), whereas the TFFL can achieve 31.6% of vibration reduction. This again demonstrates that the proposed method is able to generate a control policy that performs comparably to the ILC.

VII. DISCUSSION & CONCLUSIONS

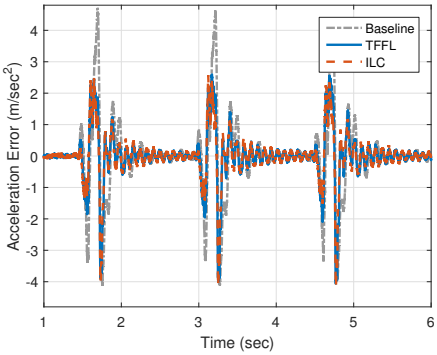
This paper proposed a time-frequency domain feedforward learning method for robot motion control. The idea is to use the previously learned ILC policy to find a feature-frequency domain mapping between the reference signal and the control actions. This mapping is then used to generate a time-frequency domain controller that behaves like an ILC for a new trajectory. In comparison to the standard ILC, the proposed method has the advantage that it is not trajectory specific and it serves a group of trajectories with a single learning action. This feature allows learning controls to be utilized in broader applications. However, further investigation is needed on the robustness of the proposed method. Experimental results show that the proposed method provides significant improvement comparing to the baseline controller. It performs comparably to the ILC while having the advantage of not requiring an additional learning action for a previously unseen trajectory.



(a) Position tracking errors



(b) Velocity tracking errors



(c) Acceleration tracking errors

Fig. 11: Comparisons of the tracking errors

REFERENCES

[1] C.-Y. Lin, L. Sun, and M. Tomizuka, "Matrix factorization for design of Q-filter in iterative learning control," in *2015 IEEE 54th Annual Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 6076–6082.

[2] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *Control Systems, IEEE*, vol. 26, no. 3, pp. 96–114, 2006.

[3] C.-Y. Lin, L. Sun, and M. Tomizuka, "Robust principal component analysis for iterative learning control of precision motion systems with non-repetitive disturbances," in *2015 American Control Conference (ACC)*. IEEE, 2015, pp. 2819–2824.

[4] M. F. Heertjes and R. M. van de Molengraft, "Set-point variation in learning schemes with applications to wafer scanners," *Control Engineering Practice*, vol. 17, no. 3, pp. 345 – 356, 2009.

[5] J. Asensio, W. Chen, and M. Tomizuka, "Feedforward input generation based on neural network prediction in multi-joint robots," *Journal of Dynamic Systems, Measurement, and Control*, vol. 136, no. 3, p. 031002, 2014.

[6] J. J. Craig, *Introduction to robotics: mechanics and control*. Pearson Prentice Hall Upper Saddle River, 2005, vol. 3.

[7] S. B. Niku, *Introduction to Robotics: Analysis, Control, Applications*, 2nd ed. New Delhi: Wiley, 2012.

[8] K. Inaba, C.-C. Wang, M. Tomizuka, and A. Packard, "Design of iterative learning controller based on frequency domain linear matrix inequality," in *American Control Conference, 2009. ACC '09.*, June 2009, pp. 246–251.

[9] W. Chen and M. Tomizuka, "Dual-stage iterative learning control for mimo mismatched system with application to robots with joint elasticity," *Control Systems Technology, IEEE Transactions on*, vol. 22, no. 4, pp. 1350–1361, 2014.

[10] J. O. Smith, *Spectral audio signal processing*. W3K, 2011.

[11] G. Gu, *Discrete-Time Linear Systems: Theory and Design with Applications*. Boston: Springer, 2012.

[12] Y. Avargel and I. Cohen, "On multiplicative transfer function approximation in the short-time fourier transform domain," *Signal Processing Letters, IEEE*, vol. 14, no. 5, pp. 337–340, May 2007.

[13] C. Avendano, "Temporal processing of speech in a time-feature

TABLE I: A Quantitative Comparison

		Baseline	ILC	TFFL
Position (m)	RMS Error	1.74×10^{-3}	6.57×10^{-4}	8.14×10^{-4}
	Standard Deviation	1.73×10^{-3}	6.54×10^{-4}	8.08×10^{-4}
	Peak Error	5.88×10^{-3}	3.09×10^{-3}	3.30×10^{-3}
Velocity (m/sec)	RMS Error	0.0387	0.0182	0.0210
	Standard Deviation	0.0387	0.0182	0.0210
	Peak Error	0.1851	0.0973	0.1146
Acceleration (m/sec^2)	RMS Error	1.0718	0.6864	0.7472
	Standard Deviation	1.0718	0.6864	0.7472
	Peak Error	4.7108	4.1577	4.0616

space," Ph.D. dissertation, Oregon Graduate Institude of Science and Technology, 1997.

[14] C. Hoffmann and H. Werner, "A survey of linear parameter-varying control applications validated by experiments or high-fidelity simulations," *Control Systems Technology, IEEE Transactions on*, vol. 23, no. 2, pp. 416–433, 2015.

[15] C. E. Rasmussen, "Gaussian processes for machine learning." MIT Press, 2006.

[16] J. S. Bendat and A. G. Piersol, "Random data analysis and measurement procedures," *Measurement Science and Technology*, vol. 11, no. 12, p. 1825, 2000.

[17] P. Stoica and R. L. Moses, *Spectral analysis of signals*. Pearson/Prentice Hall Upper Saddle River, NJ, 2005.

[18] *(Sept. 2015) FANUC R-2000iC*. [Online]. Available: http://www.fanuc.eu/be/en/robots/robot-filter-page/r-2000-series/r-2000ic-210f