



A limited-preview filtered B-spline approach to tracking control – With application to vibration-induced error compensation of a 3D printer



Molong Duan, Deokkyun Yoon, Chinedum E. Okwudire*

Department of Mechanical Engineering, University of Michigan, 2350 Hayward, Ann Arbor, MI 48109, United States

ARTICLE INFO

Article history:

Received 5 April 2017

Revised 3 August 2017

Accepted 6 September 2017

Available online 29 September 2017

Keywords:

Feedforward control

Non-minimum phase zeros

Limited preview (look-ahead)

Basis functions

3D printer

ABSTRACT

A limited-preview filtered B-spline (FBS) approach for minimizing errors in tracking a desired trajectory is presented. In the full-preview FBS approach, the feedforward control input to a stable linear system, with or without non-minimum phase zeros, is decomposed into B-spline basis functions with unknown coefficients; the basis functions are forward filtered using the (modeled) dynamics of the system, and their coefficients selected to minimize tracking errors of the entire trajectory in one batch. Instead, this paper proposes the use of a receding horizon to recursively compute unknown coefficients that minimize tracking errors for small batches (subsets) of the trajectory at a time, by exploiting the local property of B-splines. This allows optimal control signals to be determined at much lower computational cost compared to full-preview FBS, thus enabling online implementation on real-time controllers. The adverse effects of limited preview on tracking accuracy, relative to full preview, are analyzed, and limited-preview FBS is shown in numerical examples to preserve the versatility of full-preview FBS in tracking systems irrespective of their zero locations. The practicality and effectiveness of the limited-preview FBS approach are demonstrated by employing it for online feedforward compensation of tracking errors caused by structural vibrations of a stepper-motor-driven 3D printer. Alleviation of vibration-induced surface waviness and layer-to-layer registration errors, without sacrificing print speed, are demonstrated.

© 2017 Elsevier Ltd. All rights reserved.

1. Introduction

Tracking control aims to minimize the errors of a system's output(s) in following a desired trajectory. A feedforward controller uses a priori knowledge of a given system and its input(s) to influence the system's output(s) in a pre-defined way; it is very often used in tracking control applications to augment feedback control, which has limited tracking accuracy because it must wait for errors to develop before reacting to them. Moreover, in some tracking control applications – as in stepper-motor-driven 3D printers – feedback control is infeasible due to lack of sensing of the controlled variables; hence, in these cases, feedforward is the only recourse for control. Excellent tracking performance can be achieved using feedforward control by direct inversion of a sufficiently accurate model of a system (i.e., pole-zero cancellation) [1]. However, when applied to systems with non-minimum phase (NMP) zeros, direct model inversion gives rise to unstable control inputs which are unacceptable [1]. Non-minimum phase zeros are prevalent in

practice. For example, they occur in systems with fast sampling rates [2], as well as in systems with non-collocated placement of sensors and actuators [3]. Therefore, it is of great practical benefit for a feedforward tracking control method to perform satisfactorily when applied to minimum phase (MP) systems, as well as systems with NMP zeros.

There have been several feedforward tracking control methods reported in the literature that are applicable to linear systems with NMP zeros. The simplest methods are NMP zero ignore (NPZ-ignore), zero phase error tracking controller (ZPETC), and zero magnitude error controller (ZMETC) [1,4]. However, depending on the system and the performance specifications, NPZ-ignore, ZMETC and ZPETC may not yield satisfactory tracking performance due to the approximations involved [4]. To improve tracking accuracy, advanced methods have been developed, e.g., extended bandwidth ZPETC [5], truncated series [6], direct inversion with bounded reference trajectories [7–11], approximate frequency domain inversion [12], H_∞ matching [13,14], B-spline-based tracking with preview using iterative learning control [15], spline filtering with feedback [16,17], causal/anti-causal dynamics decomposition [18], etc. A major problem faced by most of the advanced methods (e.g., [7,8,10,11,18]) is that they require full preview (i.e., full knowl-

* Corresponding author.

E-mail addresses: molong@umich.edu (M. Duan), ydykyoon@umich.edu (D. Yoon), okwudire@umich.edu (C.E. Okwudire).

Nomenclature

a, b	zero and pole of simulated first order discrete system
$\mathbf{A}_r, \mathbf{B}_r, \mathbf{C}_r$	state-space representation of recursion error dynamics
C	tracking controller
E	number of discrete points minus one
$\mathbf{e}, \bar{\mathbf{e}}$	tracking error vector of FPFBS and LPFBS
$\Delta \mathbf{e}_R$	recursion tracking error due to limited preview
$\Delta \mathbf{e}_T (\Delta \hat{\mathbf{e}}_T)$	(approximated) tracking error due to system dynamics truncation
$\bar{\mathbf{g}}, \mathbf{g}$	normalized and un-normalized (open-ended) knot vector
$H, \mathbf{H} (\bar{\mathbf{H}}), h_k (\bar{h}_k), h_{ss}$	system, its (truncated and rescaled) lifted domain representation, its (truncated and rescaled) impulse response, and its DC gain
i, j, k	indices of preview windows, basis functions, and time steps
K	gain of simulated first order discrete system
L	knot vector spacing
$L_C (L_{C,min})$	(minimum) preview horizon length
L_H	finite impulse response approximation length
$L_r (n_r)$	the number of non-empty rows (columns) in $\bar{\mathbf{N}}_{PC}$
m	B-spline degree
$\mathbf{M}_A$	recursion error dynamic matrices in control canonical form
n	B-spline number of control points minus one
n_b, n_f	minimum preview window needed to account for the contributions of the non-zero portions of $\bar{\mathbf{N}}_{PC}$ and $\bar{\mathbf{N}}_{CF}$
$n_C (n_{up})$	number of control points evaluated (updated) in each preview window
$N_{j,m}, \mathbf{N}$	basis function and basis function matrix
$\bar{N}_{j,m}, \bar{\mathbf{N}}$	filtered basis function and filtered basis function matrix
$\bar{N}_{j,m}, \bar{\mathbf{N}}, \Delta \bar{\mathbf{N}}$	truncated filtered basis function, truncated filtered basis function matrix, $\bar{\mathbf{N}} - \bar{\mathbf{N}}$
$\bar{\mathbf{N}}_{PC} (\mathbf{N}_{PC}), \bar{\mathbf{N}}_{CF}$	partition of $\bar{\mathbf{N}} (\mathbf{N})$ indicating effect from past to current, and from current to future.
$\bar{\mathbf{N}}_{up}$	first n_{up} columns of $\bar{\mathbf{N}}_C$
$\mathbf{p}, \bar{\mathbf{p}}, \Delta \mathbf{p}$	control points of FPFBS without truncation, with truncation and corresponding difference
$\Delta \bar{\mathbf{p}}_{up,i}$	deviation of updated control points in i th preview window due to limited preview
$t (t_k)$	(discrete) time
T_s	sampling time
$u, \mathbf{u}$	control input as time series and vector
$\bar{\mathbf{u}}_{up}$	control input vector updated within current batch
$x, \mathbf{x}$	actual output as time series and vector

$x_d, \mathbf{x}_d$	desired output as time series and vector
$\mathbf{X}_P, \mathbf{X}_C, \mathbf{X}_F$	past, current, future partition of vectors or matrices $\mathbf{X}$ with respect to preview window under evaluation, where $\mathbf{X} \in \{\mathbf{x}_d, \bar{\mathbf{e}}, \bar{\mathbf{p}}, \Delta \bar{\mathbf{p}}, \bar{\mathbf{N}}, \mathbf{N}\}$
ξ	B-spline curve parameter representing normalized time
$\sigma_{\max}, \mathbf{N} (\sigma_{\min}, \bar{\mathbf{N}})$	maximum (minimum) singular value of $\mathbf{N} (\bar{\mathbf{N}})$

edge) of the entire desired trajectory, which presents significant computational challenges for online implementation when the desired trajectory has a large number of samples (i.e., long duration). Moreover, they are not applicable in certain practical situations where only subsets of the desired trajectory are known beforehand, as in most modern computer numerical controllers (CNCs) which calculate desired trajectories in small batches using a look-ahead feature [19]. Some work has been done on alleviating the requirement for full preview in certain advanced methods. For example, a limited-preview approximate solution to full-preview stable inversion [7,8] is proposed in [9], and in [15] iterative learning control is incorporated into the method using B-spline decomposition, to address model uncertainty. However, most of the advanced methods (whether with full or limited preview) are not versatile in terms of the systems and/or the desired trajectories to which they are applicable (e.g., the methods in [9,15] cannot be applied to non-hyperbolic systems) – see [20] for a more detailed discussion of this matter. Moreover, some advanced methods exhibit tracking performance that varies significantly depending on NMP zero location (in the complex plane) [20].

The filtered basis function (FBF) method [20–24] is an elegant method for tracking control of linear systems (with or without NMP zeros). It assumes that the desired trajectory to be tracked is fully known and that the control trajectory can be decomposed into a set of basis functions with unknown coefficients. The basis functions are forward filtered using the (modeled) dynamics of the system and the coefficients are selected to minimize the errors in tracking the desired trajectory. The authors have shown in prior work [20,22] that the FBF method is very versatile with regard to the systems and desired trajectories to which it is applicable, and its tracking performance has been observed in case studies to be much less susceptible to NMP zero locations compared to other methods [20,22]. Moreover, interesting features can be incorporated into the FBF method through a proper choice of basis functions. For example the filtered B-spline (FBS) method, which uses B-splines as basis functions, was proposed by the authors as an elegant way of introducing tracking error weighting [22], constraint handling properties [25], and control effort tradeoff [26] into the FBF method. They demonstrated the FBS method's ability to significantly improve the tracking and contouring accuracy of a manufacturing machine experiencing unwanted vibration, without any reduction in total motion time [25]. However, a major shortcoming of FBS (and other FBF methods – that use other basis functions) is that it requires full preview of the desired trajectory. It is therefore desired to relax the full-preview requirement of the FBS method, to make it online implementable, while retaining its versatility. Accordingly, the major contributions of this paper are in:

1. Proposing a limited-preview FBS method which uses a receding horizon to recursively compute B-spline coefficients that minimize tracking errors for small subsets of the desired trajectory, by exploiting the local property of B-splines [27].
2. Analytically deriving conditions that explain the effects of employing limited-preview FBS on tracking accuracy, relative to

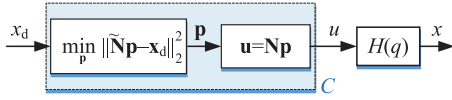


Fig. 1. Block diagram for tracking control using FPFBS method.

full-preview FBS, and showing numerically that limited-preview FBS retains the versatility of the full-preview FBS method.

3. Demonstrating the effectiveness and practicality of the proposed limited-preview FBS method via online feedforward compensation of tracking errors caused by structural vibrations of a stepper-motor-driven 3D printer commanded to follow very lengthy trajectories.

The outline of the paper is as follows: Section 2 provides an overview of the full-preview FBS method. Section 3 then details the proposed limited-preview FBS method, and provides a theoretical and numerical analysis of its tracking accuracy and computational efficiency relative to the full-preview FBS method. The system identification, modeling and feedforward control of a desktop 3D printer using the proposed limited-preview FBS method is presented in Section 4, along with results and discussions, followed by conclusions and future work.

2. Overview of full-preview filtered B-spline approach

Consider the stable linear time-invariant (LTI) discrete-time system (given by transfer function $H(q)$) shown in Fig. 1, controlled by a feedforward tracking controller, C , where q is the forward shift operator. The system $H(q)$ is assumed to have nonzero DC gain, and could represent an open-loop plant or closed-loop controlled system [4]. Given a desired trajectory, $x_d(k)$, where $0 \leq k \leq E$, $k \in \mathbb{Z}$ and $E + 1$ is the number of discrete points in the trajectory, the objective of tracking control is to design C such that the control trajectory, $u(k)$, after passing through $H(q)$, results in an output trajectory $x(k)$ that is sufficiently close to $x_d(k)$. In the full-preview FBS (FPFBS) approach [22,25], x_d is assumed to be entirely known a priori and u is expressed as

$$\begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(E) \end{bmatrix} = \begin{bmatrix} N_{0,m}(\xi_0) & N_{1,m}(\xi_0) & \cdots & N_{n,m}(\xi_0) \\ N_{0,m}(\xi_1) & N_{1,m}(\xi_1) & \cdots & N_{n,m}(\xi_1) \\ \vdots & \vdots & \ddots & \vdots \\ N_{0,m}(\xi_E) & N_{1,m}(\xi_E) & \cdots & N_{n,m}(\xi_E) \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_n \end{bmatrix} \quad (1)$$

$\mathbf{u}$ $\mathbf{N}$ $\mathbf{p}$

where $\mathbf{N}$ is the matrix of B-spline basis functions of degree m , $\mathbf{p}$ is a vector of $n + 1$ unknown coefficients (or control points), $j = 0, 1, \dots, n$, and $\xi \in [0, 1]$ is the spline parameter, representing normalized time, which is discretized in (1) into $E + 1$ uniformly spaced points, $\xi_0, \xi_1, \dots, \xi_E$. The real-valued basis functions, $N_{j,m}(\xi)$, are given by [27]

$$N_{j,m}(\xi) = \frac{\xi - \bar{g}_j}{\bar{g}_{j+m} - \bar{g}_j} N_{j,m-1}(\xi) + \frac{\bar{g}_{j+m+1} - \xi}{\bar{g}_{j+m+1} - \bar{g}_{j+1}} N_{j+1,m-1}(\xi)$$

$$N_{j,0}(\xi) = \begin{cases} 1 & \bar{g}_j \leq \xi \leq \bar{g}_{j+1} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $\bar{\mathbf{g}} = [\bar{g}_0 \ \bar{g}_1 \ \dots \ \bar{g}_{m+n+1}]^T$ is a normalized knot vector defined over $[0, 1]$. For convenience, $\bar{\mathbf{g}}$ is assumed to be uniformly spaced; i.e.

$$\bar{g}_j = \begin{cases} 0 & 0 \leq j \leq m \\ \frac{j-m}{n-m+1} & m+1 \leq j \leq n \\ 1 & n+1 \leq j \leq m+n+1 \end{cases} \quad (3)$$

Similar to $\mathbf{u}$, let vectors $\mathbf{x}_d$ and $\mathbf{x}$ represent the $E + 1$ discrete points of x_d and x , respectively. Accordingly, based on the definition of $\mathbf{u}$ in (1), $\mathbf{x}$ can be written as

$$\mathbf{x} = \tilde{\mathbf{N}}\mathbf{p} \quad (4)$$

where $\tilde{\mathbf{N}}$ is the filtered B-spline matrix, acquired by passing each column of $\mathbf{N}$ through the dynamic system H , or its model. Accordingly, the tracking error can be written as

$$\mathbf{e} = \mathbf{x}_d - \mathbf{x} = \mathbf{x}_d - \tilde{\mathbf{N}}\mathbf{p} \quad (5)$$

By minimizing the two-norm of the tracking error, the optimal control points are given by the well-known least-squares solution [22]

$$\min_{\mathbf{p}} \left((\mathbf{x}_d - \tilde{\mathbf{N}}\mathbf{p})^T (\mathbf{x}_d - \tilde{\mathbf{N}}\mathbf{p}) \right) \Rightarrow \mathbf{p} = (\tilde{\mathbf{N}}^T \tilde{\mathbf{N}})^{-1} \tilde{\mathbf{N}}^T \mathbf{x}_d \quad (6)$$

Note that the calculation of the optimal $\mathbf{p}$ requires full preview of the desired trajectory $\mathbf{x}_d$; the optimal control input, $\mathbf{u} = \mathbf{N}\mathbf{p}$, is thus generated offline and then sent to system H , as discussed in [22,25].

3. Description and analysis of limited-preview filtered B-spline approach

3.1. Limited-preview FBS approach

In contrast to the FPFBS approach described in the preceding section, the proposed limited-preview FBS (LPFBS) approach generates optimal feedforward control inputs in sequential batches, based on a moving preview window (receding horizon) applied to the desired trajectory, x_d . Since x_d is not assumed to be fully known a priori, the knot vector can no longer be normalized over $[0, 1]$ (as in (3)) using the total number of samples in the trajectory ($E + 1$), and the total number of control points ($n + 1$). Therefore, an un-normalized and open-ended knot vector is defined as

$$g_j = \begin{cases} 0 & 0 \leq j \leq m \\ (j - m)LT_s & j \geq m + 1 \end{cases} \quad (7)$$

where $L \geq 1$ represents the uniform spacing of the knot vector elements as an integer multiple of sampling time T_s ; i.e.,

$$g_{j+1} - g_j = LT_s \quad (j = m, m + 1, \dots) \quad (8)$$

Note that L also represents the spacing (in terms of number of discrete samples) between consecutive $N_{j,m}$, for $j \geq m + 1$; therefore, smaller L implies a larger number of control points (for a given length of trajectory) which generally improves tracking accuracy at the expense of higher computational cost (larger matrix sizes) [20,22]. Notice that the terminal elements of $\bar{g}_j$ (i.e., for $n + 1 \leq j \leq m + n + 1$) in (3) are not included in the open-ended g_j , since n is not necessarily known. With the un-normalized g_j , $N_{j,m}$ is expressed as a function of t by replacing ξ with t and $\bar{g}_j$ with g_j in (2), and the function is sampled at $t_k = kT_s$ to formulate $\mathbf{N}$ as in (1).

One significance of the knot vector is that it defines the support (i.e., domain of influence) of $N_{j,m}$. The support, $\text{supp}(f(t))$, of a function $f(t)$ is the domain t for which $f(t) \neq 0$. Based on (2) and (7), the support for each $N_{j,m}$ is given by

$$\text{supp}(N_{j,m}(t)) = \begin{cases} [0, (j + 1)LT_s] & j < m \\ [(j - m)LT_s, (j + 1)LT_s] & j \geq m \end{cases} \quad (9)$$

Notice that each $N_{j,m}$ has finite (compact) support, implying that it has a localized domain of influence. This is a key property of B-splines that facilitates LPFBS, because it limits the influence of any particular control point on its neighbors. The problem, however, is that $\text{supp}(\tilde{N}_{j,m}(t))$ is infinite, in the common scenario where $H(q)$

is an infinite impulse response (IIR) filter. To circumvent this problem, if $H(q)$ is an IIR filter, $\tilde{\mathbf{N}}$ is approximated by its truncated version, $\tilde{\mathbf{N}}$, such that

$$\text{supp}(\tilde{\mathbf{N}}_{j,m}(t)) = \text{supp}(\mathbf{N}_{j,m}(t)) \cup [(j+1)LT_s, (j+1)LT_s + L_H T_s] \quad (10)$$

In other words, the IIR filter $H(q)$ is approximated by a finite impulse response filter of length L_H ; hence the support of $\tilde{\mathbf{N}}_{j,m}(t)$ is extended by duration $L_H T_s$. The approximation of $\tilde{\mathbf{N}}$ with $\tilde{\mathbf{N}}$ is reasonable because $H(q)$ is assumed to be stable hence its impulse response $\{h_k\}$ is guaranteed to converge to zero as k grows. (Needless to say that if $H(q)$ is an FIR filter of length L_H then $\tilde{\mathbf{N}} = \tilde{\mathbf{N}}$ – i.e., no approximation is needed). For the purposes of LPFBS where the tracking problem is solved in small windows (batches) using $\tilde{\mathbf{N}}$ instead of $\tilde{\mathbf{N}}$, (5) can be redefined and written in partitioned form as

$$\bar{\mathbf{e}} = \mathbf{x}_d - \tilde{\mathbf{N}}\bar{\mathbf{p}} \Leftrightarrow \begin{bmatrix} \bar{\mathbf{e}}_P \\ \bar{\mathbf{e}}_C \\ \bar{\mathbf{e}}_F \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{d,P} \\ \mathbf{x}_{d,C} \\ \mathbf{x}_{d,F} \end{bmatrix} - \begin{bmatrix} \tilde{\mathbf{N}}_P & \mathbf{0} & \mathbf{0} \\ \tilde{\mathbf{N}}_{PC} & \tilde{\mathbf{N}}_C & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{N}}_{CF} & \tilde{\mathbf{N}}_F \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}}_P \\ \bar{\mathbf{p}}_C \\ \bar{\mathbf{p}}_F \end{bmatrix} \quad (11)$$

where the bar (–) accent attached to $\mathbf{e}$ and $\mathbf{p}$ indicate that they are defined in terms of $\tilde{\mathbf{N}}$ (instead of $\tilde{\mathbf{N}}$, as in (5)); subscript C indicates a quantity pertaining to the current window (e.g., $\mathbf{x}_{d,C}$ is the subset of the desired trajectory currently being evaluated); subscripts P and F denote quantities pertaining to the past and future, relative to the current window. Note that the lower triangular nature of $\tilde{\mathbf{N}}$ is due to causality, while the zero submatrix at the lower left corner of $\tilde{\mathbf{N}}$ arises because of the finite support of $\tilde{\mathbf{N}}_{j,m}(t)$. Given a current subset $\mathbf{x}_{d,C}$ of $\mathbf{x}_d$, as well as its past values, $\mathbf{x}_{d,P}$, the LPFBS method seeks to approximately determine current control points ($\bar{\mathbf{p}}_C$) that minimize current errors $\bar{\mathbf{e}}_C$ without considering $\mathbf{x}_{d,F}$ (future values of $\mathbf{x}_d$). This is achieved via the local least-squares solution given by

$$\bar{\mathbf{p}}_C \cong (\tilde{\mathbf{N}}_C^T \tilde{\mathbf{N}}_C)^{-1} \tilde{\mathbf{N}}_C^T (\mathbf{x}_{d,C} - \tilde{\mathbf{N}}_{PC} \bar{\mathbf{p}}_P) \quad (12)$$

where $\bar{\mathbf{p}}_P$ are the control points already determined in past iterations of the same local least-squares solution. The implication of (12) is that $\bar{\mathbf{p}}_C$ optimally fits the portion of $\mathbf{x}_d$ within the current preview window ($\mathbf{x}_{d,C}$), while taking into consideration the contributions from past control points ($\bar{\mathbf{p}}_P$). The matrix $\tilde{\mathbf{N}}_C$ has dimensions $L_C \times n_C$, where L_C represents the number of time steps considered in the current preview window, and n_C is the associated number of control points. The preview windows are overlapped to facilitate continuity. This is achieved by using only n_{up} of the n_C control points calculated from (12) to update the control input for the next $L n_{up}$ time steps; i.e.,

$$\bar{\mathbf{u}}_{up} = \begin{bmatrix} \mathbf{I}_{L n_{up}} & \mathbf{0} \end{bmatrix} (\mathbf{N}_C \bar{\mathbf{p}}_C + \mathbf{N}_{PC} \bar{\mathbf{p}}_P) \quad (n_{up} < n_C) \quad (13)$$

where $\bar{\mathbf{u}}_{up}$ represents the control inputs updated within current window, and $\mathbf{N}_C$, $\mathbf{N}_{PC}$ are submatrices of the unfiltered basis function matrix, $\mathbf{N}$, obtained following the same partitioning rule and notation applied to $\tilde{\mathbf{N}}$ in (11). For the next batch, $\tilde{\mathbf{N}}_C$ and $\mathbf{N}_C$ advance in time by $L n_{up}$ steps, while $\tilde{\mathbf{N}}_P$, $\mathbf{N}_P$, and the other subvectors/submatrices associated with $\tilde{\mathbf{N}}$ and $\mathbf{N}$ are adjusted accordingly. This process is repeated recursively until the end of the desired trajectory. Fig. 2 graphically illustrates the relationships among $\tilde{\mathbf{N}}_{j,m}$, L , n_C , L_C , n_{up} and $L n_{up}$ within the matrix structure of $\tilde{\mathbf{N}}$; note that variables L_r and n_r appearing in the figure are defined in the remarks below.

Remarks.

1. Apart from its first m columns (basis functions) with varying support, the matrix $\tilde{\mathbf{N}}$ has a uniform banded structure (see

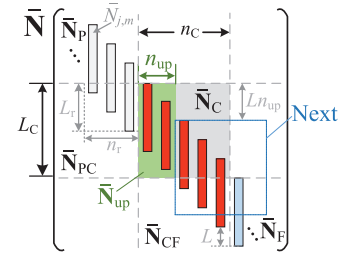


Fig. 2. Pictorial view of $\tilde{\mathbf{N}}$'s structure.

- Fig. 2). It therefore makes sense to initialize the recursive optimization by determining the first m control points using the portion of $\mathbf{x}_d$ associated with the first m columns of $\tilde{\mathbf{N}}$, and then apply a fixed preview window ($\tilde{\mathbf{N}}_C$) of size $L_C \times n_C$ to the rest of $\mathbf{x}_d$. Another way of initializing the optimization is to append mL additional elements to the beginning of $\mathbf{x}_d$ such that $\mathbf{x}_d(0) = \mathbf{x}_d(1) = \dots = \mathbf{x}_d(mL)$. This way the first m control points are given by $\mathbf{x}_d(0)/h_{ss}$, where $h_{ss} \neq 0$ is the DC gain of $H(q)$.
2. The length L_C of the fixed preview window should be selected such that $L_C \geq L_{C,min}$ given by

$$L_{C,min} = \underbrace{L_H + (m+1)L}_{\text{supp}(\tilde{\mathbf{N}}_{j,m}) \text{ for } j \geq m} + \underbrace{(n_{up}-1)L}_{\text{supp}(\tilde{\mathbf{N}}_{j,m}) \text{ for } j \geq m} = L_H + (n_{up} + m)L \quad (14)$$

- The implication of (14) is that the window should at least cover the support of the first n_{up} basis functions associated with $\mathbf{x}_{d,C}$ (see Fig. 2). This condition also guarantees the existence of the zero submatrix at the bottom left corner of $\tilde{\mathbf{N}}$ shown in (11). Moreover, selecting n_C such that $L_C = n_C L$ ensures that $\tilde{\mathbf{N}}_C$ has sufficient number of non-zero elements in all of its columns (see Fig. 2); this avoids ill-conditioned $\tilde{\mathbf{N}}_C$, and is preferred in the pseudoinverse operation in (12).
3. With the use of a fixed preview window, $\tilde{\mathbf{N}}_C$ is unchanged throughout the optimization. As such, to reduce computational cost, the pseudoinverse of $\tilde{\mathbf{N}}_C$ can be calculated offline, stored in a computer's memory and then implemented online as a constant multiplier at each iteration of the recursive optimization.
 4. Although the terms $\tilde{\mathbf{N}}_{PC} \bar{\mathbf{p}}_P$ in (12) and $\mathbf{N}_{PC} \bar{\mathbf{p}}_P$ in (13) seem to include all past control points, in reality, only the last n_r control points of $\bar{\mathbf{p}}_P$ need to be considered. This arises from the fact that $\tilde{\mathbf{N}}_{PC}$ is empty except for the $L_r \times n_r$ portion in its top right corner (see Fig. 2), where

$$L_r = L_H + mL; n_r = \left\lceil \frac{L_H}{L} \right\rceil + m \quad (15)$$

and $\lceil \cdot \rceil$ indicates a rounding-up operation to the closest integer.

3.2. Theoretical analysis of errors of LPFBS relative to FPFBS

There are two key sources of error in the LPFBS approach relative to the FPFBS approach, namely: (i) Truncation error ($\Delta \mathbf{e}_T$): The error induced by determining control points using the truncated $\tilde{\mathbf{N}}$ as opposed to using $\tilde{\mathbf{N}}$, (assuming full preview in both cases); and (ii) Recursion error ($\Delta \mathbf{e}_R$): The error induced by using $\tilde{\mathbf{N}}$ to determine control points recursively with limited preview as opposed to full preview. This section seeks to theoretically analyze the effects of these errors separately. The total error in LPFBS relative to FPFBS is the sum of these two errors. The analysis provides insights on how to properly select parameters of LPFBS, as is illustrated numerically in Section 3.3.

3.2.1. Analysis of truncation error

Recall from (5) and (11) that $\mathbf{p}$ and $\bar{\mathbf{p}}$ are the control points determined with full preview using $\tilde{\mathbf{N}}$ and $\tilde{\mathbf{N}}$, respectively.

Accordingly [28],

$$\tilde{\mathbf{N}}^T(\tilde{\mathbf{N}}\mathbf{p} - \mathbf{x}_d) = \mathbf{0}; \quad \tilde{\mathbf{N}}^T(\tilde{\mathbf{N}}\tilde{\mathbf{p}} - \mathbf{x}_d) = \mathbf{0} \quad (16)$$

Let us further define perturbations $\Delta\mathbf{p}$ and $\Delta\tilde{\mathbf{N}}$ as

$$\tilde{\mathbf{p}} = \mathbf{p} + \Delta\mathbf{p}; \quad \tilde{\mathbf{N}} = \tilde{\mathbf{N}} + \Delta\tilde{\mathbf{N}} = \tilde{\mathbf{N}} + (\tilde{\mathbf{H}} - \mathbf{H})\mathbf{N} \quad (17)$$

where $\mathbf{H}$ and $\tilde{\mathbf{H}}$ are respectively the convolution matrices (lifted representations) of system $H(q)$ and its truncated approximation (of FIR length L_H) rescaled such that the FIR approximation maintains the same DC gain as $H(q)$. Note that $\Delta\tilde{\mathbf{N}}$ does not alter the near-diagonal terms of $\tilde{\mathbf{N}}$, and matrices $\tilde{\mathbf{N}}$ and $\tilde{\mathbf{N}}$ share the same matrix rank. The least-squares solution is continuous with respect to perturbation when the matrix rank is preserved [28]. Substituting (17) in (16) and extracting only the first-order terms yields

$$\Delta\mathbf{p} = (\tilde{\mathbf{N}}^T\tilde{\mathbf{N}})^{-1}\Delta\tilde{\mathbf{N}}^T(\mathbf{x}_d - \tilde{\mathbf{N}}\mathbf{p}) - (\tilde{\mathbf{N}}^T\tilde{\mathbf{N}})^{-1}\tilde{\mathbf{N}}^T\Delta\tilde{\mathbf{N}}\mathbf{p} \quad (18)$$

Accordingly, the first-order approximation of the truncation error, $\Delta\hat{\mathbf{e}}_T$, is derived as

$$\Delta\hat{\mathbf{e}}_T = \tilde{\mathbf{N}}\Delta\mathbf{p} = \tilde{\mathbf{N}}(\tilde{\mathbf{N}}^T\tilde{\mathbf{N}})^{-1}\Delta\tilde{\mathbf{N}}^T\mathbf{e} - \tilde{\mathbf{N}}(\tilde{\mathbf{N}}^T\tilde{\mathbf{N}})^{-1}\tilde{\mathbf{N}}^T\Delta\tilde{\mathbf{N}}\mathbf{p} \quad (19)$$

where $\mathbf{e}$ represents the tracking error in the absence of truncation (calculated based on $\mathbf{p}$). From the definition of matrix norms and singular value decomposition [28],

$$\begin{aligned} \|\tilde{\mathbf{N}}(\tilde{\mathbf{N}}^T\tilde{\mathbf{N}})^{-1}\|_2 &= \frac{1}{\sigma_{\min,\tilde{\mathbf{N}}}}; \quad \|\tilde{\mathbf{N}}(\tilde{\mathbf{N}}^T\tilde{\mathbf{N}})^{-1}\tilde{\mathbf{N}}^T\|_2 = 1; \\ \|\Delta\tilde{\mathbf{N}}\|_2 &\leq \|\tilde{\mathbf{H}} - \mathbf{H}\|_2 \|\mathbf{N}\|_2 \leq \sigma_{\max,\mathbf{N}} \left\| \sum_{k=0}^{L_H} \tilde{h}_k q^{-k} - H(q) \right\|_\infty \end{aligned} \quad (20)$$

where $\sigma_{\min,\tilde{\mathbf{N}}}$ and $\sigma_{\max,\mathbf{N}}$ are the smallest and largest singular values of $\tilde{\mathbf{N}}$ and $\mathbf{N}$, respectively; $\{\tilde{h}_k\}$ represents the first L_H elements $\{h_k\}$ (the IIR of $H(q)$) rescaled such that the FIR approximation maintains the same DC gain as $H(q)$. Accordingly, we get

$$\begin{aligned} \|\Delta\hat{\mathbf{e}}_T\|_2 &\leq \gamma_T \left(\frac{1}{\sigma_{\min,\tilde{\mathbf{N}}}} \|\mathbf{e}\|_2 + \|\mathbf{p}\|_2 \right); \\ \gamma_T &\triangleq \sigma_{\max,\mathbf{N}} \left\| \sum_{k=0}^{L_H} \tilde{h}_k q^{-k} - H(q) \right\|_\infty \end{aligned} \quad (21)$$

Given $\mathbf{e}$, $\mathbf{p}$ and $\mathbf{N}$ from FPFBS, (21) shows that $\|\Delta\hat{\mathbf{e}}_T\|_2$ can be made arbitrarily small by increasing L_H (which, in general, reduces γ_T). However, notice that, for $\gamma_T \neq 0$, having a small $\sigma_{\min,\tilde{\mathbf{N}}}$ can amplify the influence of $\|\mathbf{e}\|_2$ on $\|\Delta\hat{\mathbf{e}}_T\|_2$. Small $\sigma_{\min,\tilde{\mathbf{N}}}$ typically occurs when $H(q)$ has NMP zeros, and $n = E$ (i.e. $L = 1$) such that the small singular values within $\mathbf{H}$ are passed to $\tilde{\mathbf{N}}$ [29]. The influences of L_H and small $\sigma_{\min,\tilde{\mathbf{N}}}$ on $\Delta\mathbf{e}_T$ are examined numerically in Section 3.3.

3.2.2. Analysis of recursion error

Based on (11), we can write

$$\begin{aligned} \tilde{\mathbf{N}}^T(\tilde{\mathbf{N}}\tilde{\mathbf{p}} - \mathbf{x}_d) &= \mathbf{0} \Leftrightarrow \\ \begin{bmatrix} \tilde{\mathbf{N}}_P^T & \tilde{\mathbf{N}}_{PC}^T & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{N}}_C^T & \tilde{\mathbf{N}}_{CF}^T \\ \mathbf{0} & \mathbf{0} & \tilde{\mathbf{N}}_F^T \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{N}}_P & \mathbf{0} & \mathbf{0} \\ \tilde{\mathbf{N}}_{PC} & \tilde{\mathbf{N}}_C & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{N}}_{CF} & \tilde{\mathbf{N}}_F \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{p}}_P \\ \tilde{\mathbf{p}}_C \\ \tilde{\mathbf{p}}_F \end{bmatrix} - \begin{bmatrix} \mathbf{x}_{d,P} \\ \mathbf{x}_{d,C} \\ \mathbf{x}_{d,F} \end{bmatrix} &= \mathbf{0} \end{aligned} \quad (22)$$

The exact solution of $\tilde{\mathbf{p}}_C$ from (22) is given by

$$\tilde{\mathbf{p}}_C = (\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_C)^{-1}\tilde{\mathbf{N}}_C^T(\mathbf{x}_{d,C} - \tilde{\mathbf{N}}_{PC}\tilde{\mathbf{p}}_P) + (\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_C)^{-1}\tilde{\mathbf{N}}_{CF}^T\tilde{\mathbf{e}}_F \quad (23)$$

Accordingly, the control-point error ($\Delta\tilde{\mathbf{p}}_C$) incurred in the current preview window by using the approximate solution in (12) is given by

$$\Delta\tilde{\mathbf{p}}_C = -(\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_C)^{-1}\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_{PC}\Delta\tilde{\mathbf{p}}_P + (\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_C)^{-1}\tilde{\mathbf{N}}_{CF}^T\tilde{\mathbf{e}}_F \quad (24)$$

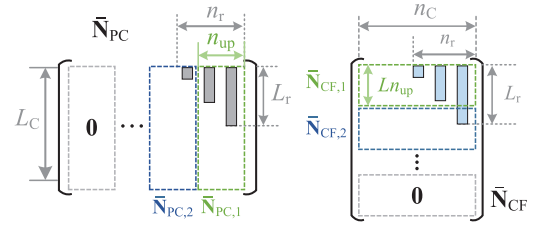


Fig. 3. Partitioning of $\tilde{\mathbf{N}}_{PC}$ and $\tilde{\mathbf{N}}_{CF}$ according to past and future updates.

It consists of errors $\Delta\tilde{\mathbf{p}}_P$ incurred in past preview windows as well as errors incurred by ignoring the contributions of future information in determining the current control points. As discussed in Section 3.1, only the first n_{up} control points of the current preview window are updated. Let $\Delta\tilde{\mathbf{p}}_{up,i}$ represent the error incurred in $\Delta\tilde{\mathbf{p}}_C$ at update $i \in \mathbb{Z}^+$; it is given by

$$\Delta\tilde{\mathbf{p}}_{up,i} = \begin{bmatrix} \mathbf{I}_{n_{up}} & \mathbf{0} \end{bmatrix} \Delta\tilde{\mathbf{p}}_C \quad (25)$$

Therefore, the recursion error ($\Delta\mathbf{e}_{R,i}$) accumulated at update i is given by

$$\Delta\mathbf{e}_{R,i} = \begin{bmatrix} \mathbf{I}_{L_{n_{up}}} & \mathbf{0} \end{bmatrix} (\tilde{\mathbf{N}}_{PC}\Delta\tilde{\mathbf{p}}_P + \tilde{\mathbf{N}}_{up}\Delta\tilde{\mathbf{p}}_{up,i}) \quad (26)$$

where $\tilde{\mathbf{N}}_{up}$ is a matrix consisting of the first n_{up} columns of $\tilde{\mathbf{N}}_C$. Let us partition the columns or rows of $\tilde{\mathbf{N}}_{PC}$, $\Delta\tilde{\mathbf{p}}_P$, $\tilde{\mathbf{N}}_{CF}$ and $\tilde{\mathbf{e}}_F$ in terms of past and future updates as

$$\begin{aligned} \tilde{\mathbf{N}}_{PC} &= \begin{bmatrix} \mathbf{0} & \tilde{\mathbf{N}}_{PC,n_b} & \tilde{\mathbf{N}}_{PC,n_b-1} & \cdots & \tilde{\mathbf{N}}_{PC,2} & \tilde{\mathbf{N}}_{PC,1} \end{bmatrix} \\ \Delta\tilde{\mathbf{p}}_P &= \begin{bmatrix} \cdots & \Delta\tilde{\mathbf{p}}_{up,i-n_b}^T & \Delta\tilde{\mathbf{p}}_{up,i-n_b+1}^T & \cdots & \Delta\tilde{\mathbf{p}}_{up,i-1}^T \end{bmatrix}^T \\ \tilde{\mathbf{N}}_{CF} &= \begin{bmatrix} \tilde{\mathbf{N}}_{CF,1}^T & \tilde{\mathbf{N}}_{CF,2}^T & \cdots & \tilde{\mathbf{N}}_{CF,n_f-1}^T & \tilde{\mathbf{N}}_{CF,n_f}^T & \mathbf{0} \end{bmatrix}^T \\ \tilde{\mathbf{e}}_F &= \begin{bmatrix} \tilde{\mathbf{e}}_{i+1}^T & \tilde{\mathbf{e}}_{i+2}^T & \cdots & \tilde{\mathbf{e}}_{i+n_f-1}^T & \tilde{\mathbf{e}}_{i+n_f}^T & \cdots \end{bmatrix}^T \\ n_b &\triangleq \lceil n_r/n_{up} \rceil; \quad n_f \triangleq \lceil L_r/(L_{n_{up}}) \rceil \end{aligned} \quad (27)$$

The structure of $\tilde{\mathbf{N}}_{PC}$ and $\tilde{\mathbf{N}}_{CF}$ described in (27) arises from the fact they each have non-zero elements only on their top right corners, with n_r columns and L_r rows, as depicted in Fig. 3; hence only small subsets of $\Delta\tilde{\mathbf{p}}_P$ and $\tilde{\mathbf{e}}_F$ affect $\Delta\tilde{\mathbf{p}}_{up,i}$ and $\Delta\mathbf{e}_{R,i}$. Note that n_b and n_f indicate the number of updates needed to fully account for the contributions of the non-zero portions of $\tilde{\mathbf{N}}_{PC}$ and $\tilde{\mathbf{N}}_{CF}$, respectively. Based on (24)–(27), $\Delta\tilde{\mathbf{p}}_{up,i}$ and $\Delta\mathbf{e}_{R,i}$ have discrete update dynamics in i given by:

$$\begin{aligned} \Delta\tilde{\mathbf{p}}_{up,i} &= \sum_{r=1}^{n_b} \mathbf{A}_r \Delta\tilde{\mathbf{p}}_{up,i-r} + \sum_{r=1}^{n_f} \mathbf{B}_r \tilde{\mathbf{e}}_{i+r} \\ \Delta\mathbf{e}_{R,i} &= \sum_{r=0}^{n_b} \mathbf{C}_r \Delta\tilde{\mathbf{p}}_{up,i-r} \end{aligned} \quad (28)$$

where

$$\begin{aligned} \mathbf{A}_r &= -\begin{bmatrix} \mathbf{I}_{n_{up}} & \mathbf{0} \end{bmatrix} (\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_C)^{-1}\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_{PC,r}; \quad (r = 1, 2, \dots, n_b) \\ \mathbf{B}_r &= \begin{bmatrix} \mathbf{I}_{n_{up}} & \mathbf{0} \end{bmatrix} (\tilde{\mathbf{N}}_C^T\tilde{\mathbf{N}}_C)^{-1}\tilde{\mathbf{N}}_{CF,r}^T; \quad (r = 1, 2, \dots, n_f) \\ \mathbf{C}_r &= \begin{cases} \begin{bmatrix} \mathbf{I}_{L_{n_{up}}} & \mathbf{0} \end{bmatrix} \tilde{\mathbf{N}}_{up}; & (r = 0) \\ \begin{bmatrix} \mathbf{I}_{L_{n_{up}}} & \mathbf{0} \end{bmatrix} \tilde{\mathbf{N}}_{PC,r}; & (r = 1, 2, \dots, n_b) \end{cases} \end{aligned} \quad (29)$$

Notice that a necessary and sufficient condition for $\Delta\mathbf{e}_R$ to be bounded as i progresses is that the eigenvalues of matrix $\mathbf{M}_A$ in

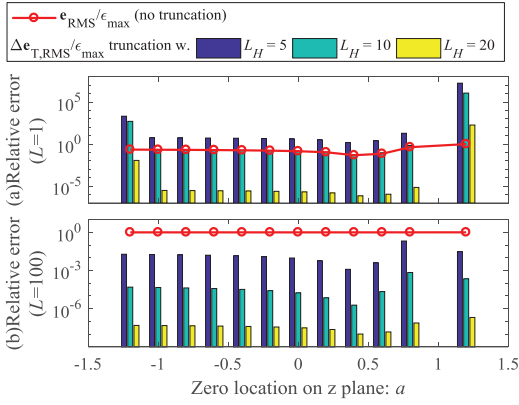


Fig. 4. Comparison of tracking performance of the FPFBS method without truncation to the tracking performance of the LPFBS method with various levels of truncation (L_H values) for: (a) extreme case where $L = 1$ ($n = E$), and (b) more-realistic case where $L = 100$ ($n = E/100$).

(30) lie within the unit circle [30].

$$\begin{bmatrix} \Delta \bar{\mathbf{p}}_{\text{up}, i-n_b+1} \\ \Delta \bar{\mathbf{p}}_{\text{up}, i-n_b+2} \\ \vdots \\ \Delta \bar{\mathbf{p}}_{\text{up}, i-1} \\ \Delta \bar{\mathbf{p}}_{\text{up}, i} \end{bmatrix} = \mathbf{M}_A \begin{bmatrix} \Delta \bar{\mathbf{p}}_{\text{up}, i-n_b} \\ \Delta \bar{\mathbf{p}}_{\text{up}, i-n_b+1} \\ \vdots \\ \Delta \bar{\mathbf{p}}_{\text{up}, i-2} \\ \Delta \bar{\mathbf{p}}_{\text{up}, i-1} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \sum_{r=1}^{n_f} \mathbf{B}_r \bar{\mathbf{e}}_{i+r} \end{bmatrix} \quad (30)$$

$$\mathbf{M}_A \triangleq \begin{bmatrix} \mathbf{0} & \mathbf{I}_{n_{\text{up}}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \mathbf{I}_{n_{\text{up}}} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{I}_{n_{\text{up}}} \\ \mathbf{A}_{n_b} & \mathbf{A}_{n_b-1} & \cdots & \mathbf{A}_2 & \mathbf{A}_1 \end{bmatrix} \quad (30)$$

3.3. Numerical analysis

In this section, the LPFBS method is numerically analyzed relative to the FPFBS method using the first order discrete-time system studied in [4,20]; it is given by

$$H(q) = K \frac{q-a}{q-b}; K = \frac{1-b}{1-a} (a \neq 1) \quad (31)$$

where K , a and $b = 0.5$ are the gain, zero and pole of the system, respectively. To investigate the versatility in tracking performance of the LPFBS method, simulations are performed with $a \in [-1.2, 1.2]$, excluding $a = 1$ for which $H(q)$ is undefined. Notice that the selected range of a accounts for MP systems, as well as hyperbolic and non-hyperbolic NMP systems. Gain K is defined such that the DC gain of the system is unity ($h_{ss} = 1$), and the sampling time is specified to be $T_s = 10^{-4}$ s. Following Ramani et al. [20], $\mathbf{x}_d$ is defined as the double integral of a pseudo random binary sequence acceleration signal of 1 s duration ($E = 10,000$), with acceleration limits $= \pm 10^4$ mm/s². All simulations are carried out using MATLAB 8® on a Windows PC with Intel Core i5-2400 CPU and 8 GB RAM; B-spline degree $m = 5$ is used in all simulations.

Fig. 4 shows normalized RMS tracking errors of the FPFBS method using $\bar{\mathbf{N}}$ (i.e., $\mathbf{e}_{\text{RMS}}$) in comparison with the normalized RMS truncation errors ($\Delta \mathbf{e}_{\text{T,RMS}}$) for various L_H and a values. The errors are normalized using ε_{max} (the maximum value of $\mathbf{e}_{\text{RMS}}$

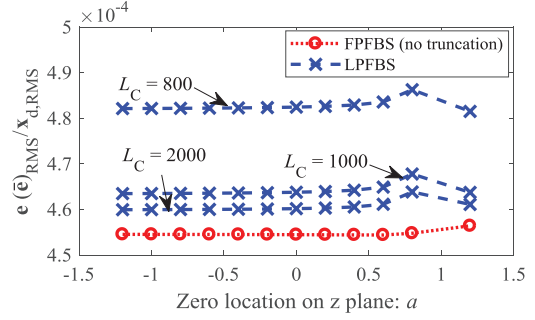


Fig. 5. Comparison of tracking performance of FPFBS (with no truncation) to LPFBS for various preview window lengths, L_C .

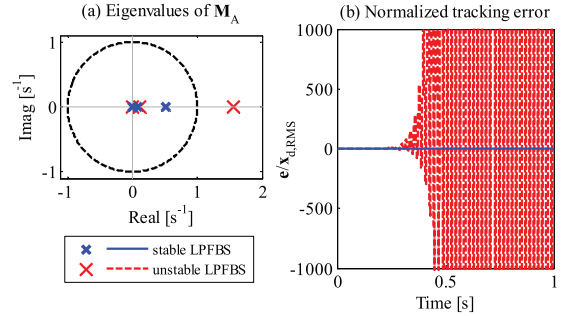


Fig. 6. Comparison of stable and unstable cases of LPFBS via (a) eigenvalues of $\mathbf{M}_A$, and (b) tracking error plots.

across all the a values investigated). Fig. 4(a) illustrates the extreme case where $L = 1$ ($n = E$), while Fig. 4(b) shows a more-typical case where $L = 100$ (i.e., $n = E/100$). Notice that, for both cases, $\mathbf{e}_{\text{RMS}}$ is fairly consistent across all values of a . This confirms the versatility of the FPFBS (and other full-preview FBF approaches – using other basis functions) in tracking MP and NMP systems, as also observed in [20,22]. Note that the slight improvement of $\mathbf{e}_{\text{RMS}}$ around $a = 0.5$ (relative to other values of a) stems from the fact that $H(q) \rightarrow 1$ (unity gain) as $a \rightarrow 0.5$, hence making tracking control notably easy. With the truncated FPFBS, we see in both cases that, as expected, $\Delta \mathbf{e}_{\text{T,RMS}}$ approaches zero as L_H increases. One also observes that, even with truncation-induced errors, the FPFBS method largely retains the versatility of its tracking performance for various values of a . Notice, however, from Fig. 4(a), that $\Delta \mathbf{e}_{\text{T,RMS}}$ increases significantly for NMP systems relative to MP systems. This is caused by the presence of small $\sigma_{\text{min}, \tilde{\mathbf{N}}}$ in NMP systems when $n = E$, which amplifies truncation errors, as discussed in Section 3.2.1. However, in the more-typical situation where $n \ll E$, as in Fig. 4(b), this problem is alleviated. Notice also that there is a slight increase in $\Delta \mathbf{e}_{\text{T,RMS}}$ around $a = 1$. This stems from the fact that the scalar multiplier K of $H(q)$ in (31) grows drastically as $a \rightarrow 1$ thus amplifying truncation errors; since, as explained in Section 3.2.1, the FIR approximation of length L_H is re-scaled to maintain the same DC gain as $H(q)$.

Fig. 5 compares the RMS tracking errors of the FPFBS method (i.e., $\mathbf{e}_{\text{RMS}}$) to those of the LPFBS method (i.e., $\bar{\mathbf{e}} = \mathbf{e} + \Delta \mathbf{e}_{\text{T}} + \Delta \mathbf{e}_{\text{R}}$), each normalized by the RMS value of $\mathbf{x}_d$, for various values of a and L_C , with $L = 100$, $L_H = 20$ and $n_{\text{up}} = 2$. It can be seen that $\bar{\mathbf{e}}_{\text{RMS}}$ approaches $\mathbf{e}_{\text{RMS}}$ as L_C is increased. Moreover, the LPFBS approach demonstrates consistent tracking performance (much like FPFBS) as a is varied. However, as discussed in Section 3.2.2, care must be taken to ensure that the recursion of LPFBS is stable. Fig. 6 compares two cases of LPFBS – a stable case with $n_{\text{up}} = 2$ and $L_C = 800$, and an unstable case with $n_{\text{up}} = 2$ and $L_C = 500$; for both cases, $a = 1.2$, $L = 100$ and $L_H = 20$. Note that the L_C value of the unstable case violates $L_{C,\text{min}} = 720$, as stipulated by (14). The stabil-

Table 1

Comparison of tracking performance and computational efficiency of LPFBS and FPFBS for various durations of desired trajectory.

Duration of desired trajectory [s]		1	4	7	10	13	16	19
Normalized RMS tracking error	LPFBS [%]	0.48	0.50	0.54	0.55	0.50	0.53	0.54
	FPFBS [%]	0.46	0.46	0.49	0.50	0.47	0.49	N/A
Total computation time	LPFBS [ms]	7	18	41	75	113	190	272
	FPFBS [s]	0.5	3.5	9.8	23.7	206.5	1635.1	N/A
Memory usage	LPFBS [KB]	57	57	57	57	57	57	57
	FPFBS [MB]	9	131	397	807	1361	2060	N/A

ity situation of the two cases is predicted by the eigenvalues of $\mathbf{M}_A$ (Fig. 6(a)) and validated in time-domain tracking error plots (Fig. 6(b)), where $\mathbf{x}_{d,RMS}$ is the RMS value of the desired trajectory, $\mathbf{x}_d$.

The tracking performance and computational efficiency of LPFBS relative FPFBS (with no truncation) are compared in Table 1 for increasing durations of desired trajectory $\mathbf{x}_d$ (generated via double integration of a $\pm 10^4$ mm/s² pseudorandom binary sequence acceleration signal with increasing durations); in the simulations, $a = 1.2$, $L = 100$, $L_H = 20$ and $L_C = 800$ are used. The normalized RMS tracking errors of LPFBS ($\bar{\mathbf{e}}_{RMS}/\mathbf{x}_{d,RMS}$) stay within 10% of that of FPFBS ($\mathbf{e}_{RMS}/\mathbf{x}_{d,RMS}$) in all cases investigated. Moreover, the total computation time of the LPFBS method stays within 1.5% of the total duration of $\mathbf{x}_d$, and its memory usage stays constant at 57 KB. However, the computational time and memory usage of the FPFBS method grows drastically with $\mathbf{x}_d$, causing the computer to run out of memory when the duration of $\mathbf{x}_d$ reaches 19 seconds. This shows why the FPFBS method is unsuitable, while the LPFBS method is suitable for online tracking control of systems with long durations of $\mathbf{x}_d$, as in the 3D printing case study discussed in the following section.

4. Application to vibration-induced error compensation of a desktop 3D printer

Commercial desktop 3D printers are designed with light, but flexible, structures and driven with stepper motors in order to reduce their cost, size and weight. Stepper motors are typically open-loop controlled with no feedback measurements to help compensate for un-modeled dynamics. As a result, parts manufactured on such 3D printers suffer from surface waviness (aka. ringing or ghosting) and registration errors, caused by stepper motors skipping counts, due to excessive vibration triggered by the motion of the print head or build platform. Such vibration-induced errors not only mar the aesthetics of 3D printed parts, but often lead to highly distorted and hence scrapped parts. The errors could be mitigated via active vibration control techniques. However, such control techniques require feedback sensors and high-sample-rate real-time control hardware, which are not commercially viable for low-cost desktop 3D printers. In this section, we demonstrate the effectiveness of feedforward control using LPFBS for online compensation of vibration-induced errors of such 3D printers, without sacrificing productivity.

4.1. Experimental set-up: commercial desktop 3D printer

Fig. 7 shows a commercially available desktop 3D printer (HICTOP Prusa i3) and two sample parts used in this paper to demonstrate the compensation of vibration-induced errors based on the LPFBS method. The motion of the printer's build platform is along the x-axis, while its print head moves along the y- and z-axes. All three axes of the printer are controlled by stepper motors, but the focus of this study is on controlling its x- and y-axis motions which generate significant vibration, due to the printer's flexible structure, as its print head and build platform move. Fig. 8

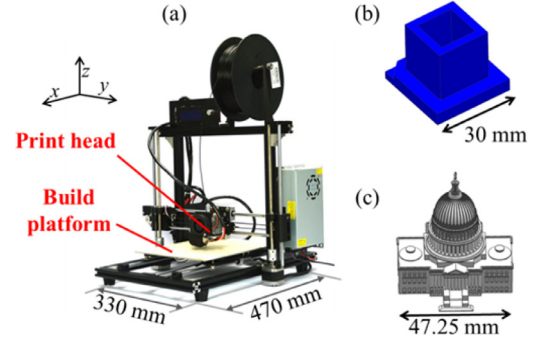


Fig. 7. (a) Commercial desktop 3D printer (HICTOP Prusa i3); (b) CAD model of square block and (c) scale (CAD) model of US Capitol used for the purposes of this study.

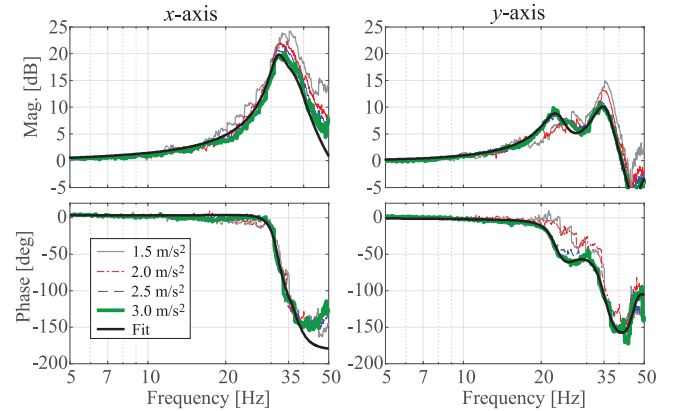


Fig. 8. Frequency response functions of print head relative to print platform for x- and y-axes of the 3D printer for various magnitudes of excitation input (acceleration). Least-squares curve fitting is used to identify the axis-level dynamics based on the 3.0 m/s² data.

shows the measured and curve fit x- and y-axis frequency response functions (FRFs) of the 3D printer. The FRFs are measured by applying swept sine acceleration signals (with amplitudes ranging from 1.5 m/s² to 3.0 m/s²) to the printer's stepper motors (each having 12.5 μ m stepping resolution) and measuring the relative acceleration of the build platform and print head using accelerometers (PCB Piezotronics 393B05 and Kistler 8704B100). The resonance peaks of the FRFs increase with decreasing input amplitude. The curve fit models are generated using MATLAB's *invfreqs* function applied to the 3.0 m/s² data, in order to better compensate for vibrations induced by more aggressive motions. The resulting transfer function are given in continuous-time and discrete-time domains (via zero order hold equivalency, $T_s = 1$ ms) as

$$\begin{aligned} & \frac{5.19 \times 10^4 s^3 + 6.45 \times 10^6 s^2 + 2.60 \times 10^9 s + 8.58 \times 10^{10}}{s^5 + 118s^4 + 9.97 \times 10^4 s^3 + 7.33 \times 10^6 s^2 + 2.35 \times 10^9 s + 8.58 \times 10^{10}} \\ & \Rightarrow \frac{0.026q^4 - 0.048q^3 - 0.003q^2 + 0.048q - 0.023}{q^5 - 4.792q^4 + 9.274q^3 - 9.060q^2 + 4.466q - 0.889} \end{aligned} \quad (32)$$

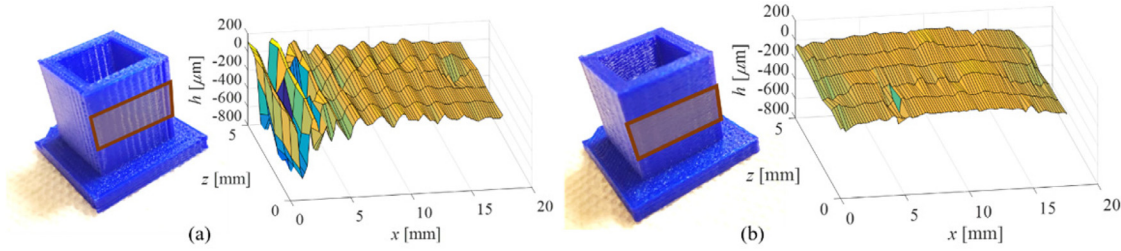


Fig. 9. Comparison of photographs and measured surface profiles (h) of the highlighted surfaces of blocks printed using (a) baseline approach (no vibration compensation) and (b) LPFBS method (both with acceleration limit of 7 m/s²). Vibration-induced surface waviness of baseline case is greatly attenuated using LPFBS.

for the x -axis, and

$$\frac{5.28 \times 10^4 s^4 + 4.97 \times 10^6 s^3 + 5.73 \times 10^8 s^2 + 2.87 \times 10^{11} s + 1.07 \times 10^{14}}{s^6 + 166s^5 + 1.83 \times 10^5 s^4 + 1.64 \times 10^7 s^3 + 8.70 \times 10^9 s^2 + 3.37 \times 10^{11} s + 1.07 \times 10^{14}} \Rightarrow \frac{0.026q^5 - 0.073q^4 + 0.047q^3 + 0.045q^2 - 0.068q + 0.023}{q^6 - 5.672q^5 + 13.56q^4 - 17.48q^3 + 12.83q^2 - 5.076q + 0.8473} \quad (33)$$

for the y -axis. Unity DC gain is enforced in the identified transfer functions to accurately represent the DC response of stepper motors. Notice that the discrete-time transfer functions for the x axis is non-hyperbolic (with a zero at $q = -1.000$); most advanced tracking control methods cannot handle non-hyperbolic systems [20]. Moreover, the y axis has a poorly damped zero (at $q = -0.976$, very close to the unit circle) which also poses a challenge for model-inversion-based tracking controllers [15].

To identify the FRFs and implement the LPFBS method, the printer's proprietary motion controller is bypassed. Instead, axis-level motion (i.e., step and direction) commands are sent to the printer's stepper motors at 1 kHz sampling rate using a real-time controller (dSPACE DS1007 and DS5203) via stepper motor drives (Pololu DRV8825). The real-time controller reads a G-code file (generated using Cura™ software package) and parses the G-code information into axis-level motion commands, emulating motion command generators of 3D printers. The motion commands are then optimized using the LPFBS method following the procedure outlined in Section 3.

4.2. Case study I: mitigation of surface waviness of 3D printed part

The square block model shown in Fig. 7(b) is printed using the 3D printer of Fig. 7(a) with different acceleration limits imposed on the motion commands (while maintaining the feedrate at 60 mm/s for all cases). For each case, the part is printed using the uncompensated motion commands (as the baseline approach), as well as the motion commands compensated through feedforward control via the LPFBS method with $n_{\text{up}} = 28$, $n_c = 56$, $L_c = 952$, $L_H = 384$, $m = 5$ and $L = 17$. It can be verified that the selected preview horizon satisfies the minimum horizon condition in (14) and the eigenvalues of corresponding $\mathbf{M}_A$ lie within the unit circle. The LPFBS parameters are chosen such that acceptable tracking performance is achieved without overly increasing the computational cost. Fig. 9(a) and (b) show the blocks printed using the two methods (both using acceleration limit of 7 m/s²), and the 3D surface profile, h , of the highlighted surfaces, measured using a laser displacement sensor (Keyence LK G-10). Notice that the vibration-induced surface waviness of the baseline case is significantly reduced by the LPFBS method. Fig. 10 compares the surface roughness (RMS h values) of blocks printed using different acceleration limits. Observe that the surface quality of the baseline approach deteriorates significantly at higher acceleration limits compared to that of the LPFBS method, which stays relatively consistent for all acceleration limits investigated. Note that the baseline approach achieves slightly better surface roughness than the compensated case when the lowest acceleration limit of 1 m/s² is used.

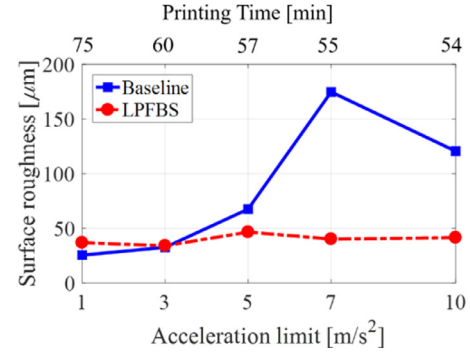


Fig. 10. Surface roughness of 3D printed blocks comparing x - and y -axes commands generated using baseline (no vibration compensation) and LPFBS methods for different acceleration limits (total printing time).

This result is attributed to modeling errors between the fit and measured FRFs, which are particularly large at lower levels of acceleration (as shown Fig. 8). From a practical standpoint, however, the slight loss of performance at 1 m/s² pales in comparison with the large gains in precision at higher acceleration levels, especially considering that higher acceleration levels are preferable because they reduce printing time.

4.3. Case study II: mitigation of registration errors in 3D printed part

The scale model of the US Capitol, shown in Fig. 7(c), is printed to demonstrate the common problem of vibration-induced registration errors in 3D printed parts, and the ability of the LPFBS method to mitigate them. The part is printed using the baseline approach (no vibration compensation) and the LPFBS method (using the same set of parameters as used in Case study I), with different acceleration limits, while keeping the feedrate at 60 mm/s. As shown in Table 2, registration errors occur with the baseline case as the acceleration limit is increased above 3 m/s², causing severe distortion of the printed parts. However, the parts printed using the LPFBS method do not exhibit such registration errors. As a result, the LPFBS method enables quality prints at much shorter printing times compared to the baseline case.

To explain the reason for the registration errors, Fig. 11 compares the measured x -axis acceleration between print head and build platform for the baseline approach and LPFBS method (both with 7 m/s² acceleration limit). Notice that, even though the acceleration of the motion command is limited to 7 m/s² (excepts for a few spikes here and there due to short distance travels), the actual acceleration of the baseline case turns out to be more than double this limit, because the build platform's motion heavily excites the resonance modes of the machine. The excessive vibration (i.e., acceleration) causes inertial loads that exceed the holding torque of the motors, hence they skip counts and lose track of their position. On the other hand, the LPFBS method compensates the

Table 2

3D printed models of US Capitol using different acceleration limits (which influences total printing time). The LPFBS method eliminates the registration errors observed in the baseline case at higher acceleration limits; the feedrate is 60 mm/s for all cases.

Acc. limit	1 m/s ²	3 m/s ²	5 m/s ²	7 m/s ²	10 m/s ²
Printing time	3:59 h	2:42 h	2:22 h	2:12 h	2:06 h
Baseline					
FBS					

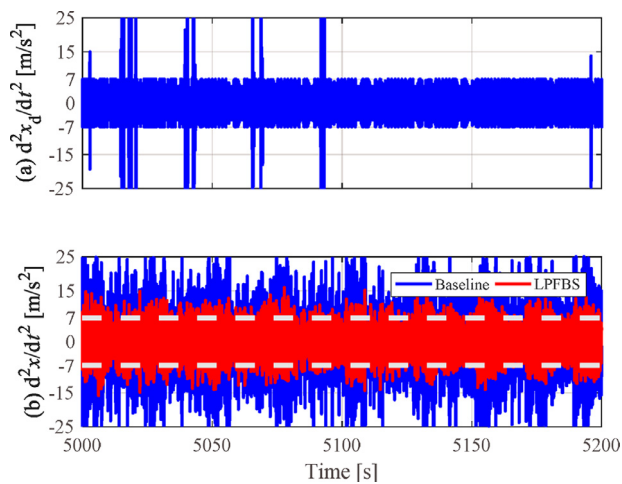


Fig. 11. (a) Acceleration profile of x-axis reference command. (b) On-machine measurement of x-axis acceleration. Dashed lines indicate the acceleration limit ($7 m/s^2$) imposed.

vibration and maintains actual acceleration levels that are closer to the desired limit, hence avoiding the registration errors.

5. Conclusion and future work

A limited-preview filtered B-spline (LPFBS) approach is proposed to minimize errors in tracking long-duration desired trajectories which may or may not be entirely known a priori. The feedforward control input to a stable linear system is decomposed into B-spline basis functions and is locally optimized within pre-defined preview horizons (windows) by exploiting the local property of B-splines. Compared to the traditional full-preview filtered B-spline (FPFBS) approach, the proposed LPFBS approach has significantly lower computational cost and can thus be implemented online, even when the entire desired trajectory is unknown. It is shown analytically and numerically that the parameters of the LPFBS method can be selected to keep its tracking performance degradation relative to FPFBS bounded and arbitrarily small. The LPFBS method is also shown, numerically, to preserve the versatility of the FPFBS method in tracking arbitrary desired trajectories applied to minimum and non-minimum phase systems. The practical benefits of the LPFBS method are demonstrated via on-

line feedforward compensation of vibration-induced errors of a commercial stepper-motor-driven desktop 3D printer. Compared to baseline cases without feedforward compensation, the proposed LPFBS method is shown to significantly reduce surface waviness and eliminate registration errors of 3D printed parts, thus allowing the production of high quality parts at higher speed (i.e., shorter manufacturing time). Future work will seek to extend the LPFBS approach to time-varying linear systems, which are common in practice, e.g., as 3D printer dynamics change with build height; constraint handling and robustness to variations in plant dynamics will also be investigated.

Acknowledgments

The authors would thank Mr. Yifan Ding for his preliminary investigation into the LPFBS method, and Mr. Andrew Edoimiyoa for his assistance in setting up the 3D printer used for experiments.

References

- [1] Tomizuka M. Zero phase error tracking algorithm for digital control. *J Dyn Syst Meas Control* 1987;109:65. doi:10.1115/1.3143822.
- [2] Astram KF, Wittenmark B. *Computer controlled systems: theory and design*. Prentice Hall; 1984.
- [3] Miu DK. *Mechatronics: electromechanics and contromechanics*. Springer Science & Business Media; 2012.
- [4] Butterworth JA, Pao LY, Abramovitch DY. Analysis and comparison of three discrete-time feedforward model-inverse control techniques for nonminimum-phase systems. *Mechatronics* 2012;22:577–87. doi:10.1016/j.mechatronics.2011.12.006.
- [5] Torfs D, De Schutter J, Swevers J. Extended bandwidth zero phase error tracking control of nonminimal phase systems. *J Dyn Syst Meas Control* 1992;114:347. doi:10.1115/1.2897354.
- [6] Gross E, Tomizuka M. Experimental flexible beam tip tracking control with a truncated series approximation to uncanceled inverse dynamics. *IEEE Trans Control Syst Technol* 1994;2:382–91. doi:10.1109/87.338659.
- [7] Devasia S, Chen D, Paden B. Nonlinear inversion-based output tracking. *IEEE Trans Autom Contr* 1996;41:930–42. doi:10.1109/9.508898.
- [8] Hunt LR, Meyer G, Su R. Noncausal inverses for linear systems. *IEEE Trans Autom Contr* 1996;41:608–11. doi:10.1109/9.489285.
- [9] Zou Q, Devasia S. Preview-based stable-inversion for output tracking of linear systems. *J Dyn Syst Meas Control* 1999;121:625. doi:10.1115/1.2802526.
- [10] Marconi L, Marro G, Melchiorri C. A solution technique for almost perfect tracking of non-minimum-phase, discrete-time linear systems. *Int J Control* 2001;74:496–506. doi:10.1080/00207170010014557.
- [11] Benosman M, Le Vey G. Stable inversion of SISO nonminimum phase linear systems through output planning: an experimental application to the one-link flexible manipulator. *IEEE Trans Control Syst Technol* 2003;11:588–97. doi:10.1109/TCST.2003.813372.
- [12] Trautt TA, Bayo E. Inverse dynamics of non-minimum phase systems with non-zero initial conditions. *Dyn Control* 1997;7:49–71.

- [13] Rigney BP, Pao LY, Lawrence DA. Nonminimum phase dynamic inversion for settle time applications. *IEEE Trans Control Syst Technol* 2009;17:989–1005. doi:[10.1109/TCST.2008.2002035](#).
- [14] Wen JT, Potsaid B. An experimental study of a high performance motion control system. In: *Proceedings of 2004 American control Conference*, 6; 2004. p. 5158–63.
- [15] Wang H, Kim K, Zou Q. B-spline-decomposition-based output tracking with preview for nonminimum-phase linear systems. *Automatica* 2013;49:1295–303. doi:[10.1016/j.automatica.2013.01.044](#).
- [16] Jetto L, Orsini V, Romagnoli R. Spline based pseudo-inversion of sampled data non-minimum phase systems for an almost exact output tracking. *Asian J Control* 2015;17:1866–79. doi:[10.1002/asjc.1079](#).
- [17] Jetto L, Orsini V, Romagnoli R. Accurate output tracking for nonminimum phase nonhyperbolic and near nonhyperbolic systems. *Eur J Control* 2014;20:292–300. doi:[10.1016/j.ejcon.2014.09.001](#).
- [18] Kwon D-S, Book WJ. A time-domain inverse dynamic tracking control of a single-link flexible manipulator. *J Dyn Syst Meas Control* 1994;116:193–200. doi:[10.1115/1.2899210](#).
- [19] Suh S-H, Kang S-K, Chung D-H, Stroud I. *Theory and design of CNC systems*. London: Springer; 2008. doi:[10.1007/978-1-84800-336-1](#).
- [20] Ramani KS, Duan M, Okwudire CE, Ulsoy AG. Tracking control of linear time-invariant nonminimum phase systems using filtered basis functions. *J Dyn Syst Meas Control* 2017;139:11001. doi:[10.1115/1.4034367](#).
- [21] Frueh JA, Phan MQ. Linear quadratic optimal learning control (LQL). *Int J Control* 2000;73:832–9. doi:[10.1080/002071700405815](#).
- [22] Duan M, Ramani KS, Okwudire CE. Tracking control of non-minimum phase systems using filtered basis functions: a NURBS-based approach. In: *Proceedings of ASME 2015 dynamic systems and control conference*; 2015. V001T03A006, doi: [10.1115/DSCC2015-9859](#).
- [23] Lunenburg JJM. *Technical report. Inversion-based MIMO feedforward design beyond rigid body systems*. Eindhoven University of Technology; 2010.
- [24] Ronde M, van den Bulk J, van de Molengraft R, Steinbuch M. Feedforward for flexible systems with time-varying performance locations. In: *Proceedings of 2013 American control conference*. IEEE; 2013. p. 6033–8. doi:[10.1109/ACC.2013.6580783](#).
- [25] Okwudire C, Ramani K, Duan M. A trajectory optimization method for improved tracking of motion commands using CNC machines that experience unwanted vibration. *CIRP Ann – Manuf Technol* 2016;65:373–6. doi:[10.1016/j.cirp.2016.04.100](#).
- [26] Ramani KS, Okwudire CE. A comparison of two methods for energy efficient tracking control using filtered basis functions. In: *Proceedings of international symposium flexible automation (ISFA)*. IEEE; 2016. p. 477–82.
- [27] Piegl L, Tiller W. *The NURBS book*. Berlin, Heidelberg: Springer; 1995. doi:[10.1007/978-3-642-97385-7](#).
- [28] Björck Å. *Numerical methods for least squares problems*. Society for Industrial and Applied Mathematics; 1996. doi:[10.1137/1.9781611971484](#).
- [29] Owens DH, Chu B. Modelling of non-minimum phase effects in discrete-time norm optimal iterative learning control. *Int J Control* 2009;83:2012–27. doi:[10.1080/00207179.2010.501458](#).
- [30] Ogata K. *Discrete-time control systems*, 2. Englewood Cliffs, NJ: Prentice Hall; 1995.

Molong Duan received his B.S. degree from Peking University, Beijing, China, and his M.S.E. degree from the University of Michigan (U-M), Ann Arbor, in 2012 and 2013, respectively. He is currently pursuing a Ph.D. degree in Mechanical Engineering at U-M. His research interests are control of hybrid and redundantly actuated systems, sustainable manufacturing, and intelligent motion command generation. Mr. Duan was granted the Rackham Centennial Fellowship from U-M in 2013. He received the best poster award at the 2014 International Forum on Sustainable Manufacturing and the best student paper award at the 2015 Dynamic Systems and Controls Conference.

Deokkyun Yoon received the B.S. and M.S. degrees in 2009 and 2010 and is working toward the Ph.D. degree, all in mechanical engineering from the University of Michigan. Prior to returning to continue working toward the Ph.D. degree, he was with Korea Institute of Machinery and Materials as a research staff member from 2010 to 2013. His research interests include mechatronics system design for improved precision, throughput, and power efficiency, applied to advanced manufacturing processes.

Chinedum E. Okwudire received his Ph.D. degree in Mechanical Engineering from the University of British Columbia in 2009 and joined the Mechanical Engineering faculty at the University of Michigan in 2011. Prior to joining Michigan, he was the mechatronic systems optimization team leader at DTL (Mori Seiki, Ltd.) based in Davis, CA. His expertise lies in smart and sustainable automation, where he leverages the fundamental engineering disciplines of machine design, structural dynamics, and control theory to tackle challenging problems in precision, throughput, and energy-efficiency faced by the manufacturing and vehicle automation industries. His scholarly and teaching contributions have been recognized by several awards that include the International Symposium on Flexible Automation Young Investigator Award; the Society of Manufacturing Engineers Outstanding Young Manufacturing Engineer Award; the SAE International Ralph Teetor Educational Award; the National Science Foundation CAREER award; and a number of best paper awards.