Mechanical Systems and Signal Processing 39 (2013) 280–296

Contents lists available at SciVerse ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

Generalization of norm optimal ILC for nonlinear systems with constraints

Marnix Volckaert [a,*], Moritz Diehl [b], Jan Swevers [a]

[a] Department of Mechanical Engineering, KU Leuven, Celestijnenlaan 300B, 3001 Heverlee, Belgium
[b] Department of Electrical Engineering, KU Leuven, Kasteelpark Arenberg 10, 3001 Heverlee, Belgium

ARTICLE INFO

Article history:
Received 26 December 2011
Received in revised form
27 September 2012
Accepted 11 March 2013
Available online 9 May 2013

Keywords:
Iterative learning control
Optimization
Nonlinear systems
Nonlinear programming

ABSTRACT

This paper discusses a generalization of norm optimal iterative learning control (ILC) for nonlinear systems with constraints. The conventional norm optimal ILC for linear time invariant systems formulates an update equation as a closed form solution of the minimization of a quadratic cost function. In this cost function the next trial's tracking error is approximated by implicitly adding a correction to the model. The proposed approach makes two adaptations to the conventional approach: the model correction is explicitly estimated, and the cost function is minimized using a direct optimal control approach resulting in nonlinear programming problems. An efficient solution strategy for such problems is developed, using a sparse implementation of an interior point method, such that long data records can be efficiently processed. The proposed approach is validated experimentally.

© 2013 Published by Elsevier Ltd.

1. Introduction

Mechatronic systems, such as production machines or industrial robots, often perform the same task repeatedly. In many applications the task is represented by a reference signal $\mathbf{y}_r$ that needs to be tracked by the system's output $\mathbf{y}$. Traditional controllers provide the same performance each time the motion is repeated, even if that performance is suboptimal, for example due to model plant mismatch or repeating disturbances. Moreover, many factors can reduce the performance over time, such as slowly changing operating conditions or dynamics. Typically, these problems lead to recalibration procedures that are often costly.

Iterative learning control (ILC) is an open loop control strategy that aims to reject repeating disturbances and improve tracking control by using information about the tracking performance of the previous motion, also called trial or iteration. Most ILC methods formulate an update equation to calculate the input signal for the next trial as a function of the tracking error of the past trial. An extensive overview of the many variants of ILC algorithms can be found in [1,2].

An early example of an update equation is [3]: $\mathbf{u}_{i+1}=\mathbf{u}_i+\Gamma\mathbf{e}_i$, with $\mathbf{u}_i$ the input of trial i, $\mathbf{e}_i$ the tracking error, defined as $\mathbf{e}_i=\mathbf{y}_r-\mathbf{y}_i$, and Γ a scalar. This approach is applicable to some linear time invariant (LTI) systems and does not require a model. Model based approaches have been developed using for example a frequency response function [4], or a finite impulse response [5,6]. While most ILC algorithms are designed for LTI systems, extensions have been made to linear time-varying (LTV) systems [7–9] and nonlinear systems [10–13]. Algorithms that only use information of the past trial are called first order algorithms, and can be distinguished from higher order algorithms that use multiple past trials [14–16], or current

* Corresponding author. Tel.: +32 488421406.
E-mail addresses: mvolckaert@gmail.com (M. Volckaert), moritz.diehl@esat.kuleuven.be (M. Diehl), jan.swevers@mech.kuleuven.be (J. Swevers).

0888-3270/$ - see front matter © 2013 Published by Elsevier Ltd.
http://dx.doi.org/10.1016/j.ymssp.2013.03.009

trial algorithms, which incorporate a feedback loop [17,18]. Another distinction can be made based on the used update equation. This equation can be designed for example by shaping the learning dynamics in the frequency domain [19–21], or by formulating a closed form solution of an optimization problem [22,5,6]. Traditional ILC algorithms cannot take constraints into account, but approaches have been developed to deal with constrained systems [23–25].

This paper extends the norm optimal ILC algorithm, as described in [5]. This is a first order, model based algorithm for unconstrained LTI systems, for which the update law is a closed form solution of a quadratic cost function. This paper proposes a generalization of this approach. It is based on the same quadratic cost function, but uses a nonlinear model of the system. Furthermore, it can take input and other constraints into account directly, due to the direct solution of the nonlinear optimization problem. It will also be shown that the underlying mechanism of the norm optimal ILC can be regarded as a trial dependent correction to the model, and this correction is made explicit in the proposed algorithm. The approach is based on two dynamic optimization problems: one for estimating the model correction, and one for calculating the next trial's input.

The structure of the presented approach, based on explicit model correction, is also presented in [26,27]. However, this paper also discusses the efficient formulation and solution of the underlying optimization problems for long data records, using a sparse implementation of an interior point method, and includes the experimental validation of the algorithm on a lab scale overhead crane, for both tracking control and point-to-point motion control.

Section 2 briefly discusses the norm optimal ILC for LTI systems. The proposed generalization for nonlinear systems with constraints is described in Section 3. This section discusses the explicit model correction and the formulation of the algorithm as two nonlinear optimal control problems. The relation to norm optimal ILC for LTI systems is discussed in Section 4. Section 5 then describes an efficient solution method for nonlinear optimal control problems, based on a sparse implementation of an interior point method. The proposed algorithm is experimentally validated, as described in Section 6. Finally, conclusions are drawn in Section 8.

2. Norm optimal ILC for LTI systems

Model based ILC algorithms typically use an update law of the following form [1]:

$$\mathbf{u}_{i+1} = Q[\mathbf{u}_i + L(\mathbf{e}_i)], \tag{1}$$

with $Q(\cdot)$ and $L(\cdot)$ a robustness and learning operator respectively. A common notation for the design of ILC algorithms is to consider the signals $\mathbf{u}$,$\mathbf{y}$ and $\mathbf{e}$ as vectors, and to write linear systems as matrices. This notation is called the lifted systems notation. For example, consider a discrete-time, single-input, single-output (SISO), LTI system with relative degree r, characterized by the finite impulse response $[p_1,p_2,\ldots,p_N]$. Assuming zero initial state, the input–output relation of this system can be written as [28]:

$$\begin{bmatrix} y(r) \\ y(r+1) \\ \vdots \\ y(r+N-1) \end{bmatrix} = \underbrace{\begin{bmatrix} p_1 & 0 & \cdots & 0 \\ p_2 & p_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_N & p_{N-1} & \cdots & p_1 \end{bmatrix}}_{\mathbf{P}} \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(N-1) \end{bmatrix}, \tag{2}$$

with $p_1 \neq 0$ because the output is shifted with the relative degree of the system, and $\mathbf{P}$ a lower triangular Toeplitz matrix. The operators $Q(\cdot)$ and $L(\cdot)$ can also be written in matrix form as $\mathbf{Q}$ and $\mathbf{L}$, which are Toeplitz for LTI systems, and lower triangular if these are causal operators.

The lifted systems notation makes it possible to write the learning algorithm as a large MIMO system in the trial domain, while the time domain dynamics are captured in the matrix $\mathbf{P}$. Substituting $\mathbf{e}_i = \mathbf{y}_r - \mathbf{P}\mathbf{u}_i$ in Eq. (1) and rearranging leads to the trial domain dynamic equation

$$\mathbf{u}_{i+1} = \mathbf{Q}(\mathbf{I}-\mathbf{L}\mathbf{P})\mathbf{u}_i + \mathbf{Q}\mathbf{L}\mathbf{y}_r \tag{3}$$

from which the well known necessary and sufficient condition for asymptotic stability (AS) of the ILC algorithm is derived

$$\rho[\mathbf{Q}(\mathbf{I}-\mathbf{L}\mathbf{P})] < 1, \tag{4}$$

with $\rho(\cdot)$ the spectral radius. It can be derived that the tracking error after convergence, $\mathbf{e}_\infty$ is zero if and only if the algorithm is AS and $\mathbf{Q}=\mathbf{I}$. If $\mathbf{L}=\mathbf{P}^{-1}$ then (4) is satisfied for $\mathbf{Q}=\mathbf{I}$. However, typically $\mathbf{L}\neq\mathbf{P}^{-1}$ because of model plant mismatch, especially at higher frequencies. Therefore $\mathbf{Q}$ is used to satisfy asymptotic stability of the algorithm, while achieving perfect tracking at a broad frequency range.

For the norm optimal ILC algorithm, $\mathbf{Q}$ and $\mathbf{L}$ (from now on called $\mathbf{Q}_{\text{ILC}}$ and $\mathbf{L}_{\text{ILC}}$) are designed in order to minimize a quadratic cost function, which is a trade off between the minimization of the tracking error, input effort, and input update rate, and has the following form [5]:

$$J_{i+1}(\mathbf{u}_{i+1}) = \mathbf{e}_{i+1}^{\mathrm{T}}\mathbf{Q}_u\mathbf{e}_{i+1} + \mathbf{u}_{i+1}^{\mathrm{T}}\mathbf{R}_u\mathbf{u}_{i+1} + \Delta\mathbf{u}_i^{\mathrm{T}}\mathbf{S}_u\Delta\mathbf{u}_i, \tag{5}$$

where $\Delta\mathbf{u}_i$ denotes the change in $\mathbf{u}$ from iteration i to $i+1$, such that $\Delta\mathbf{u}_i = (\mathbf{u}_{i+1}-\mathbf{u}_i)$, $\mathbf{Q}_u$ is an $N\times N$ positive-definite matrix, and $\mathbf{R}_u$ and $\mathbf{S}_u$ are $N\times N$ positive-semidefinite matrices. These matrices are typically designed to be diagonal.

Note that (5) depends on the unknown next trial tracking error $\mathbf{e}_{i+1}$, which needs to be approximated by using a model $\hat{P}(\mathbf{u})$ of the system $P(\mathbf{u})$, or in lifted form $\hat{\mathbf{P}}\approx\mathbf{P}$.

Since $\mathbf{u}_{i+1}=\mathbf{u}_i+\Delta\mathbf{u}_i$, it follows that $\mathbf{y}_{i+1}=\mathbf{y}_i+\mathbf{P}\Delta\mathbf{u}_i$. Therefore the true value $\mathbf{e}_{i+1}$ can be written as $\mathbf{e}_{i+1}=\mathbf{y}_r-\mathbf{y}_i-\mathbf{P}\Delta\mathbf{u}_i$, and using $\hat{\mathbf{P}}\approx\mathbf{P}$ the error can be approximated by

$$\hat{\mathbf{e}}_{i+1}=\mathbf{y}_r-\mathbf{y}_i-\hat{\mathbf{P}}\Delta\mathbf{u}_i. \tag{6}$$

Substituting $\hat{\mathbf{e}}_{i+1}$ for $\mathbf{e}_{i+1}$ in (5), applying the stationarity condition for optimality and rearranging leads to [5]

$$\begin{aligned}\mathbf{Q}_{\text{ILC}}&=(\hat{\mathbf{P}}^{\mathrm{T}}\mathbf{Q}_u\hat{\mathbf{P}}+\mathbf{R}_u+\mathbf{S}_u)^{-1}(\hat{\mathbf{P}}^{\mathrm{T}}\mathbf{Q}_u\hat{\mathbf{P}}+\mathbf{S}_u)\\ \mathbf{L}_{\text{ILC}}&=(\hat{\mathbf{P}}^{\mathrm{T}}\mathbf{Q}_u\hat{\mathbf{P}}+\mathbf{S}_u)^{-1}\hat{\mathbf{P}}^{\mathrm{T}}\mathbf{Q}_u.\end{aligned} \tag{7}$$

Note that if $\mathbf{Q}_u=\mathbf{I}$ and $\mathbf{R}_u=\mathbf{S}_u=0$, the optimal operators are found to be $\mathbf{Q}_{\text{ILC}}=\mathbf{I}$ and $\mathbf{L}_{\text{ILC}}=\hat{\mathbf{P}}^{-1}$, which is the simplest form of model based ILC. For $\mathbf{S}_u\neq0$, $L(\cdot)$ is typically noncausal, since the expression for $\mathbf{L}_{\text{ILC}}$ in (7) can be rearranged to

$$\mathbf{L}_{\text{ILC}}=(\mathbf{Q}_u\hat{\mathbf{P}}+\hat{\mathbf{P}}^{-T}\mathbf{S}_u)^{-1}\mathbf{Q}_u \tag{8}$$

by eliminating $\hat{\mathbf{P}}^{T}$. The term $\mathbf{Q}_u\hat{\mathbf{P}}$ is lower triangular, while $\hat{\mathbf{P}}^{-T}\mathbf{S}_u$ is upper triangular. Therefore $\mathbf{L}_{\text{ILC}}$ is a full matrix, which means $L(\cdot)$ is noncausal. For $\mathbf{R}_u\neq0$, $Q(\cdot)$ is typically noncausal, by a similar derivation.

A further analysis of (6) shows that the norm optimal ILC implicitly applies a model correction. Applying $\Delta\mathbf{u}_i=\mathbf{u}_{i+1}-\mathbf{u}_i$ to (6) and rearranging leads to

$$\hat{\mathbf{e}}_{i+1}=\mathbf{y}_r-\underbrace{[\hat{\mathbf{P}}\mathbf{u}_{i+1}+(\mathbf{y}_i-\hat{\mathbf{P}}\mathbf{u}_i)]}_{\hat{\mathbf{y}}_{i+1}}. \tag{9}$$

The predicted next iteration output $\hat{\mathbf{y}}_{i+1}$ consists of the term $\hat{\mathbf{P}}\mathbf{u}_{i+1}$, which is the expected next iteration output based on the nominal model, corrected by the term $(\mathbf{y}_i-\hat{\mathbf{P}}\mathbf{u}_i)$, such that the next iteration tracking error is estimated better than using the nominal model only. The form of this correction is fixed, independent of the weighting terms in (5), and 'hidden' in the optimal operators (7).

Other types of model correction have been introduced, for example in [12]. This approach combines a direct minimization method for (5) with a disturbance model structure for the trial domain dynamics in the following form:

$$\begin{aligned}\mathbf{x}_i&=\mathbf{F}\mathbf{u}_i+\mathbf{d}_i+N_\xi\boldsymbol{\xi}_i\\ \mathbf{y}_i&=\mathbf{G}\mathbf{x}_i+\mathbf{H}\mathbf{u}_i+N_v\mathbf{v}_i\end{aligned} \tag{10}$$

with $\mathbf{x}_i$,$\mathbf{u}_i$ and $\mathbf{y}_i$ the lifted system state, input and output vectors respectively, and $\mathbf{F}$,$\mathbf{G}$ and $\mathbf{H}$ lower triangular matrices representing the system dynamics. The vectors $\boldsymbol{\xi}_i$ and $\mathbf{v}_i$ represent process and measurement noise respectively. The vector $\mathbf{d}_i$ represents the model errors and any repeating disturbances, and can be seen as a correction to the nominal model, additive to the states. It is estimated with a linear Kalman filter in the trial domain. However, this correction also has a fixed format.

Other ILC approaches for LTI systems have been developed based on the same cost function (5). For example, the approach described in [6] uses an indirect approach, with $\mathbf{R}_u=0$ and $\mathbf{S}_u\neq0$, using the solution of the Ricatti equation. A faster implementation of this approach is described in [29], and an extension to the algorithm, including future trials in the cost function, is described in [30]. However, these approaches require full state knowledge during the trial, and cannot take constraints into account.

For nonlinear systems, the presented approaches require a linearization around the reference output, and are therefore not always applicable, e.g. for point-to-point motions.

The next section describes a generalization of the norm optimal ILC algorithm that does not have a number of the drawbacks of the presented algorithms: it can use nonlinear models, and it can take constraints into account directly. It explicitly applies a model correction without a predefined format, and therefore provides great flexibility. These advantages come at the cost of only mildly increased computational complexity, due to the efficient application of a sparse optimization algorithm.

3. Generalized nonlinear ILC

This section presents the generalization of norm optimal ILC for nonlinear systems with constraints. The approach is model based, however the only assumption is that the model $\hat{P}$ is a discrete time state space model of the following form:

$$\hat{P}:\begin{cases}\mathbf{x}(k+1)=f[\mathbf{x}(k),u(k)]\\ \hat{y}(k)=h[\mathbf{x}(k),u(k)],\end{cases} \tag{11}$$

with $\mathbf{x}$ the state vector of size n, $\mathbf{u}$ the input and $\hat{\mathbf{y}}$ the model output. The initial state, to which the system is returned after each trial, is denoted by $\mathbf{x}_{\text{init}}$. Note that the algorithm is presented for SISO systems, but it is equally applicable to MIMO systems. The aim of the ILC algorithm is to make the output of the system P track the reference signal $\mathbf{y}_r$, which is a vector of size N. Algorithm 1 summarizes the approach.

Algorithm 1. Generalized nonlinear ILC.

1: Assume an initial guess of the input signal, $\mathbf{u}_1$. This initial guess can for example be the solution of an optimal control problem using the nominal model $\hat{P}$. Set the trial index i to $i=1$.
2: Apply the input signal to the system, and measure and record the output, $\mathbf{y}_i$, with i the trial index.
3: Use $\mathbf{u}_i$ and $\mathbf{y}_i$ to solve a large parameter estimation problem of size N, to improve $\hat{P}$.
4: Calculate a new optimal input signal, based on the improved model.
5: Advance the trial index i and return to step 2.

First, the explicit model correction will be discussed in detail. It will then be shown how steps 3 and 4 can be regarded as two dynamic optimization problems, for which an efficient solution strategy is described in Section 5.

3.1. Explicit model correction

The rearrangement of the predicted next iteration tracking error $\hat{\mathbf{e}}_{i+1}$ from (6) into (9) and the resulting interpretation are based on the LTI property of the model $\hat{P}$. However, the same type of approximation can also be made with nonlinear models if the correction term is made explicit. For example, the next iteration tracking error can be approximated as follows:

$$\hat{\mathbf{e}}_{i+1} = \mathbf{y}_r - [\hat{P}(\mathbf{u}_{i+1}) + \boldsymbol{\alpha}_i] \tag{12}$$

with $\boldsymbol{\alpha}_i$ a correction signal to be estimated after trial i. In this case the sum of the nominal model and the correction signal can be written as the corrected model $\hat{P}_c(\mathbf{u},\boldsymbol{\alpha})$. It is clear that if $\hat{P}(\mathbf{u})$ is a linear model, and $\boldsymbol{\alpha}_i$ is estimated as $(\mathbf{y}_i - \hat{\mathbf{P}}\mathbf{u}_i)$, then $\hat{\mathbf{e}}_{i+1}$ is the same as for the conventional norm optimal ILC. However, explicitly estimating $\boldsymbol{\alpha}_i$ makes it possible to manipulate this signal, for example to constrain it, or to change its features in time- or frequency domain.

Furthermore, $\boldsymbol{\alpha}$ can enter the corrected model $\hat{P}_c(\mathbf{u},\boldsymbol{\alpha})$ in several ways. For (12) $\boldsymbol{\alpha}$ is additive to the model output, but other examples include $\hat{P}_c(\mathbf{u},\boldsymbol{\alpha}) = \hat{P}(\mathbf{u}+\boldsymbol{\alpha})$, or more complex forms. Regardless of this choice, $\hat{P}_c(\mathbf{u},\boldsymbol{\alpha})$ can in general be written as

$$\hat{P}_c : \begin{cases} \mathbf{x}(k+1) = f_c[\mathbf{x}(k), u(k), \alpha(k)] \\ \hat{y}_c(k) = h_c[\mathbf{x}(k), u(k), \alpha(k)] \end{cases}, \tag{13}$$

and the signal $\boldsymbol{\alpha}$ can be considered to be an additional input of $\hat{P}_c$. The presented ILC approach now consists of finding an optimal value of $\boldsymbol{\alpha}_i$ in an estimation step, followed by a control step to calculate $\mathbf{u}_{i+1}$.

3.2. Two step procedure

The goal of model correction is to make the corrected model $\hat{P}_c(\mathbf{u},\boldsymbol{\alpha})$ correspond better with the real system than the nominal model $\hat{P}(\mathbf{u})$. Using this corrected model, the aim of the ILC algorithm is to find a pair of signals $(\mathbf{u}^*,\boldsymbol{\alpha}^*)$ that satisfy the following equation:

$$\hat{P}_c(\mathbf{u}^*,\boldsymbol{\alpha}^*) = P(\mathbf{u}^*) = \mathbf{y}_r \tag{14}$$

However, $\mathbf{u}^*$ and $\boldsymbol{\alpha}^*$ cannot be calculated simultaneously because it is impossible to differentiate between both signals. The solution lies in solving the problem alternatively for $\boldsymbol{\alpha}$ and for $\mathbf{u}$, using a known value for the other signal, and repeating these two steps for each trial in an iterative manner. In this way $\boldsymbol{\alpha}$ will drive $\hat{P}_c$ closer to the true system P, and the corrected model's output $\hat{\mathbf{y}}_c$ will approach the reference $\mathbf{y}_r$ for increasing i. These two steps can be formulated as optimal control problems with a similar structure.

3.2.1. Step 3 in algorithm 1: estimation

The aim of this step, after trial i, is to find a value for $\boldsymbol{\alpha}_i$, such that the modeled output $\hat{\mathbf{y}}_c$ follows the actual measurement $\mathbf{y}_i$ more closely. Therefore the main objective for this step is

$$\min_{\boldsymbol{\alpha}} J_{\boldsymbol{\alpha}} = (\mathbf{y}_i - \hat{\mathbf{y}}_c)^T \mathbf{Q}_\alpha (\mathbf{y}_i - \hat{\mathbf{y}}_c), \tag{15}$$

with

$$\hat{\mathbf{y}}_c = \hat{P}_c(\mathbf{u}_i,\boldsymbol{\alpha}), \tag{16}$$

and with $\mathbf{Q}_\alpha \in \mathbb{R}^{N\times N}$ a positive semidefinite matrix, assumed to be diagonal. Additional objectives can be added to (15) to tune and shape $\boldsymbol{\alpha}$ at this stage. Possible applications include a frequency dependent weighting on $\boldsymbol{\alpha}$, to limit the learning behavior to a certain bandwidth, or to tune $\boldsymbol{\alpha}$ in time domain, to enable and disable learning at specific time intervals along the trial. Another example is to penalize large deviations of $\boldsymbol{\alpha}$ from trial to trial, in order to increase the robustness of the algorithm to non-repeating disturbances such as measurement noise. Inequality constraints can be applied to the minimization of (15), for example to set a maximum deviation of $\hat{P}_c$ with respect to the nominal model $\hat{P}$.

3.2.2. Step 4 in algorithm 1: control

The aim of this step is to use the estimated $\boldsymbol{\alpha}_i$ in order to find the optimal input signal for the next trial, $\mathbf{u}_{i+1}$, such that $\hat{\mathbf{y}}_c$ is close to $\mathbf{y}_r$. The main objective for this step is

$$\min_{\mathbf{u}} J_{\mathbf{u}} = (\mathbf{y}_r - \hat{\mathbf{y}}_c)^T \mathbf{Q}_u (\mathbf{y}_r - \hat{\mathbf{y}}_c), \tag{17}$$

with

$$\hat{\mathbf{y}}_c = \hat{P}_c(\mathbf{u}, \boldsymbol{\alpha}_i), \tag{18}$$

and again with $\mathbf{Q}_u \in \mathbb{R}^{N \times N}$ a diagonal positive semidefinite matrix. Also for this step additional objectives can be added to (17) to tune the input signal directly, for example to penalize large control efforts, or to smooth $\mathbf{u}$. Inequality constraints can be added, for example to take actuator constraints into account.

Since both the estimation problem and the control problem have a similar structure, they can be solved by a common solution strategy. This strategy is described in Section 5.

4. Relation to norm optimal ILC for LTI systems

This section shows that the presented approach is a generalization of the linear norm optimal ILC approach. It is shown that under certain conditions, a closed form solution of the two optimization problems that constitute the new approach can be formulated, and this solution corresponds to the update equation of linear norm optimal ILC, described in Section 2. These conditions are the following:

1. The system and its model are linear, such that $\mathbf{y} = \mathbf{P}\mathbf{u}$ and $\hat{\mathbf{P}} \approx \mathbf{P}$.
2. No constraints are introduced in the estimation and control steps of the presented approach.
3. The correction signal $\boldsymbol{\alpha}$ is assumed to be additive to the model output, i.e. $\hat{P}_c(\mathbf{u}, \boldsymbol{\alpha}) = \hat{\mathbf{P}}\mathbf{u} + \boldsymbol{\alpha}$.
4. For the estimation step (15), $\mathbf{Q}_\alpha = \mathbf{I}$ and no additional objectives are defined.
5. For the control step (17), $\mathbf{Q}_u$ is an arbitrary positive semidefinite diagonal matrix. Additional objectives are $\mathbf{u}^T \mathbf{R}_u \mathbf{u}$ and $\Delta\mathbf{u}^T \mathbf{S}_u \Delta\mathbf{u}$ (with $\Delta\mathbf{u} = \mathbf{u} - \mathbf{u}_i$), with $\mathbf{R}_u$ and $\mathbf{S}_u$ arbitrary positive semidefinite diagonal matrices.

4.1. Estimation

Under the presented conditions, the objective function for the estimation step can be written as

$$J_\alpha(\boldsymbol{\alpha}) = [\mathbf{y}_i - \hat{P}_c(\mathbf{u}_i, \boldsymbol{\alpha})]^T [\mathbf{y}_i - \hat{P}_c(\mathbf{u}_i, \boldsymbol{\alpha})] \tag{19}$$

Taking the derivative of (19) and using condition (3), the stationarity condition for optimality becomes

$$\frac{\partial J_\alpha(\boldsymbol{\alpha}_i)}{\partial \boldsymbol{\alpha}} = (\mathbf{y}_i - \hat{\mathbf{P}}\mathbf{u}_i + \boldsymbol{\alpha}_i) = 0, \tag{20}$$

yielding

$$\boldsymbol{\alpha}_i = \mathbf{y}_i - \hat{\mathbf{P}}\mathbf{u}_i, \tag{21}$$

which corresponds to the implicit model correction of the conventional norm optimal ILC, discussed in Section 2.

4.2. Control

The objective function for the control step becomes

$$J_{\mathbf{u}}(\mathbf{u}) = [\mathbf{y}_r - \hat{P}_c(\mathbf{u}, \boldsymbol{\alpha}_i)]^T \mathbf{Q}_u [\mathbf{y}_r - \hat{P}_c(\mathbf{u}, \boldsymbol{\alpha}_i)] + \mathbf{u}^T \mathbf{R}_u \mathbf{u} + \Delta\mathbf{u}^T \mathbf{S}_u \Delta\mathbf{u}$$

Taking the derivative of (22), using (21) and condition (3) from above, the stationary condition for optimality becomes

$$\frac{\partial J_{\mathbf{u}}(\mathbf{u}_{i+1})}{\partial \mathbf{u}_{i+1}} = -\hat{\mathbf{P}}^T \mathbf{Q}_u (\mathbf{y}_r - \hat{\mathbf{P}}\mathbf{u}_{i+1} - \mathbf{y}_i + \hat{\mathbf{P}}\mathbf{u}_i) + (\mathbf{R}_u + \mathbf{S}_u)\mathbf{u}_{i+1} = 0, \tag{22}$$

and after rearranging

$$\mathbf{u}_{i+1} = \underbrace{(\hat{\mathbf{P}}^T \mathbf{Q}_u \hat{\mathbf{P}} + \mathbf{R}_u \mathbf{u} + \mathbf{S}_u \mathbf{u})^{-1} \hat{\mathbf{P}}^T \mathbf{Q}_u}_{\mathbf{Q}_{\text{ILC}}} \hat{\mathbf{P}} [\mathbf{u}_i + \underbrace{(\hat{\mathbf{P}}^T \mathbf{Q}_u \hat{\mathbf{P}})^{-1} \hat{\mathbf{P}}^T \mathbf{Q}_u}_{\mathbf{L}_{\text{ILC}}} \mathbf{e}_i]. \tag{23}$$

Eq. (23) gives the update law that is implied by the presented ILC approach under the given conditions, and it is clear that the same update law is found as for the norm optimal ILC algorithm (7). However, none of these conditions are necessary for the application of the new approach.

5. Efficient solution of nonlinear optimal control problems

This section presents a general formulation for nonlinear optimal control problems as nonlinear programming (NLP) problems, and then discusses an efficient solution method using an interior point algorithm. This approach allows the exploitation of the sparse structure of the NLP problem. The estimation and control steps of the presented ILC approach can both be written in this formulation.

5.1. General problem formulation

A general optimal control problem consists of finding control inputs $\mathbf{u}_c$, given a model of the system, exogenous inputs $\mathbf{w}$, and an initial state $\mathbf{x}_{\text{init}}$, such that a norm of a set of signals $\mathbf{e}$ is minimized, and constraints on a set of signals $\mathbf{z}$ are satisfied. Note that this general formulation is valid for MIMO systems. The problem is illustrated in Fig. 1.

Here, $\mathbf{x}(k)\in\mathbb{R}^n$ represents the states at the time instant k, with n the order of the model, $k\in[0,N[$, with N the length of the considered signals, while $\mathbf{w}(k)\in\mathbb{R}^q$ represents the exogenous inputs that are assumed to be known, and $\mathbf{u}_c(k)\in\mathbb{R}^m$ represents the control inputs that need to be calculated. Two sets of outputs are defined: $\mathbf{e}(k)\in\mathbb{R}^p$ are outputs that need to be minimized, while $\mathbf{z}(k)\in\mathbb{R}^s$ are outputs that need to be constrained. The distinction between $\mathbf{e}(k)$ and $\mathbf{z}(k)$ is only made to simplify the notation of the optimization problem. The values of each of these vectors at all time instants k can be combined to create the supervectors $\overline{\mathbf{x}},\overline{\mathbf{w}},\overline{\mathbf{u}}_c,\overline{\mathbf{e}}$ and $\overline{\mathbf{z}}$, such that for example

$$\overline{\mathbf{x}} = [\mathbf{x}(0)^T,\mathbf{x}(1)^T,\ldots,\mathbf{x}(N-1)^T]^T. \tag{24}$$

In other words, the notation $\mathbf{x}(k)$ is used for a vector at time instant k, while $\overline{\mathbf{x}}$ denotes a vector with values for each $k=0,1,\ldots,N-1$.

An efficient formulation of the optimization problem can be achieved by regarding both $\overline{\mathbf{x}}$ and $\overline{\mathbf{u}}_c$ as optimization variables, and joining them in the vector $\overline{\mathbf{v}}$. Such an approach also provides more degrees of freedom to the optimization algorithm and allows to initialize it at a guess for the states, which are often much better known than the inputs. This makes this simultaneous approach more robust compared to the conventional sequential approach to dynamic optimization, which eliminates the states for given inputs in each iteration. This in particular allows to much better treat unstable or even chaotic systems [31]. The vector $\overline{\mathbf{v}}$ is therefore of length $(n+m)N$ and is constructed as follows:

$$\overline{\mathbf{v}} = [\mathbf{x}(0)^T,\mathbf{u}_c(0)^T,\mathbf{x}(1)^T,\ldots,\mathbf{x}(N-1)^T,\mathbf{u}_c(N-1)^T]^T. \tag{25}$$

The considered application minimizes the 2-norm of $\overline{\mathbf{e}}$, subject to equality and inequality constraints, such that the general optimization problem can be written as

$$\underset{\overline{\mathbf{v}}}{\text{minimize}} \quad J(\overline{\mathbf{v}}) = \overline{\mathbf{e}}^T\mathbf{Q}\overline{\mathbf{e}} \tag{26a}$$

$$\text{subject to} \quad g(\overline{\mathbf{v}}) = 0 \tag{26b}$$

$$h(\overline{\mathbf{v}}) \geq 0. \tag{26c}$$

In general $J(\overline{\mathbf{v}}),g(\overline{\mathbf{v}})$ and $h(\overline{\mathbf{v}})$ are nonlinear functions of $\overline{\mathbf{v}}$, and therefore (26a) is a nonlinear programming (NLP) problem. The matrix $\mathbf{Q}\in\mathbb{R}^{pN\times pN}$ is a positive semidefinite diagonal matrix with diagonal $\overline{\mathbf{q}}$, with $\overline{\mathbf{q}} = [\mathbf{q}(0)^T,\mathbf{q}(1)^T,\ldots,\mathbf{q}(N-1)^T]^T$. Each $\mathbf{q}(k)\in\mathbb{R}^p$ is a vector that contains the weights on the p minimized outputs in $\mathbf{e}(k)$ for the time instant k. The equality constraint function $g(\overline{\mathbf{v}})$ (26b) is defined as

$$g(\overline{\mathbf{v}}) = \begin{cases} \mathbf{x}(0)-\mathbf{x}_{\text{init}} \\ \mathbf{x}(k+1)-f[\mathbf{x}(k),\mathbf{u}_c(k),\mathbf{w}(k)] \quad \text{for } k=0,1,\ldots,N-2 \end{cases}, \tag{27}$$

such that the solution satisfies the model dynamics. It is clear that there are n equality constraints per time instant k, so the size of $g(\overline{\mathbf{v}})$ is nN. The inequality constraints (26c) are defined to constrain the optimization variables and the outputs $\overline{\mathbf{z}}$. The function $h(\overline{\mathbf{v}})$ is defined as

$$h(\overline{\mathbf{v}}) = \begin{cases} \overline{\mathbf{v}}(k)-\overline{\mathbf{v}}_{\min}(k) \\ \overline{\mathbf{v}}_{\max}(k)-\overline{\mathbf{v}}(k) \\ h_z[\mathbf{x}(k),\mathbf{u}_c(k),\mathbf{w}(k)]-\mathbf{z}_{\min}(k) \quad \text{for } k=0,1,\ldots,N-1 \\ \mathbf{z}_{\max}(k)-h_z[\mathbf{x}(k),\mathbf{u}_c(k),\mathbf{w}(k)] \quad \text{for } k=0,1,\ldots,N-1 \end{cases} \tag{28}$$

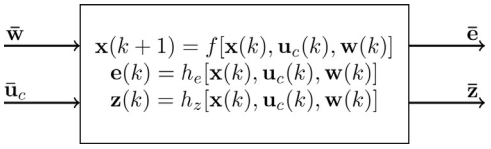


Fig. 1. General form of a nonlinear dynamic system for the formulation of an optimal control problem.

with $\overline{\mathbf{v}}(k)=[\mathbf{x}(k)^T,\mathbf{u}_c(k)^T]^T$, and $\overline{\mathbf{v}}_{\min}(k),\overline{\mathbf{v}}_{\max}(k),\mathbf{z}_{\min}(k),\mathbf{z}_{\max}(k)$ the lower and upper bounds on $\overline{\mathbf{v}}(k)$ and $\mathbf{z}(k)$ respectively. Since the size of $\overline{\mathbf{v}}$ is $(n+m)N$, and the number of constrained outputs is s, the size of $h(\overline{\mathbf{v}})$ is $2(n+m+s)N$.

The optimization problem (26a) has an associated Lagrangian function $\mathcal{L}$, defined as $\mathcal{L}=J(\overline{\mathbf{v}})-\lambda_g^T g(\overline{\mathbf{v}})-\boldsymbol{\mu}_h^T h(\overline{\mathbf{v}})$, with λ_g and $\boldsymbol{\mu}_h$ the Lagrange multipliers of the equality and inequality constraints respectively. Two matrices that are used in the solution of (26a) are the Jacobian of the constraint function J_g, and the Hessian H of $\mathcal{L}$. The vector $g(\overline{\mathbf{v}})$ can be written as $[g(0),g(1),\ldots,g(N-1)]^T$, with each $g(k)$ a vector of n constraints. It is clear that $g(k)$ is a function of $\overline{\mathbf{v}}(k)$ and $\mathbf{x}(k+1)$ only. Therefore J_g has a sparse, banded structure, with N blocks of size $n\times(2n+m)$. The matrix H has a similar sparse structure, and for both matrices the number of non-zeros only increases linearly with N. This means that the problem scales well for higher N.

In order to optimally benefit from the sparse, structured Jacobian and Hessian, the optimization algorithm has to process these matrices in an efficient manner. IPOPT, a software package written in C++ by A. Wächter et al. [32] and available under the Eclipse Public License, is tailored for this kind of large scale, sparse NLPs. Its application to problem (26a) will be discussed in the next section.

5.2. Implementation using IPOPT

A necessary condition for optimality of an optimizer $\mathbf{v}^*$ of (26a) is the following Karush–Kuhn–Tucker (KKT) equation [33]:

$$\nabla J(\overline{\mathbf{v}}^*)-\mathrm{J}_g^T(\overline{\mathbf{v}}^*)\lambda_g-\mathrm{J}_h^T(\overline{\mathbf{v}}^*)\boldsymbol{\mu}_h=0, \tag{29}$$

with $\nabla J(\overline{\mathbf{v}})$ the gradient of $J(\overline{\mathbf{v}})$, J_g and J_h the Jacobian matrices of $g(\overline{\mathbf{v}})$ and $h(\overline{\mathbf{v}})$ respectively, and $\lambda_g,\boldsymbol{\mu}_h$ the corresponding Lagrange multipliers. In order to satisfy the inequality constraints $h(\overline{\mathbf{v}})\geq 0$, the following conditions also have to be satisfied:

$$\begin{aligned} &h(\overline{\mathbf{v}}^*)\geq 0\\ &\boldsymbol{\mu}_h\geq 0\\ &h(\overline{\mathbf{v}}^*)\boldsymbol{\mu}_h=0 \end{aligned} \tag{30}$$

These are non-smooth conditions, preventing the use of Newton's method. The idea of an interior point (IP) method is to replace the conditions (30) by the approximation $h(\overline{\mathbf{v}}^*)\boldsymbol{\mu}_h=\tau$, with $\tau>0$ small. This results in smooth KKT conditions, such that Newton's method can be applied to yield estimates $\overline{\mathbf{v}}(\tau),\lambda_g(\tau)$ and $\boldsymbol{\mu}_h(\tau)$. The optimal solution is found by iteratively lowering τ, since it can be shown that if $\tau\to 0$ then $\overline{\mathbf{v}}(\tau)\to\overline{\mathbf{v}}^*$.

In order to apply the IPOPT solver, the user must create a number of functions that define the optimization problem. The IP method itself is provided by IPOPT and is called to solve the user defined problem in the main function. The user defined functions include:

- *get_nlp_info, get_bounds_info*: these functions define the number of non-zeros in J_g and H, for efficient memory allocation, and define the bounds on $\overline{\mathbf{v}}$.
- *eval_f*: evaluation of the objective function, J.
- *eval_grad_f*: evaluation of the gradient, ∇J.
- *eval_g*: the constraint functions $g(\overline{\mathbf{v}})$ and $h(\overline{\mathbf{v}})$.
- *eval_jac_g*: the Jacobian J_g. Only the non-zero elements have to be provided, with their row and column index.
- *eval_h*: the Hessian H. Since this is a symmetric matrix, only the lower left half has to be provided, again in sparse form.

5.3. Application to generalized ILC

Both steps of the proposed ILC approach can be written in the general problem formulation introduced in Section 5.1. Consider the corrected model $\hat{P}_c$ given by (13), with state equation f_c and output equation h_c. For the estimation step, given by (15) and (16), the formulation is the following:

$$\begin{aligned} &\mathbf{u}_c(k)=\boldsymbol{\alpha}_i(k)\quad f(k)=f_c[\mathbf{x}(k),\mathbf{w}_2(k),\mathbf{u}_c(k)]\\ &\mathbf{w}(k)=\begin{bmatrix}\mathbf{w}_1(k)\\ \mathbf{w}_2(k)\end{bmatrix}=\begin{bmatrix}\mathbf{y}_i(k)\\ \mathbf{u}_i(k)\end{bmatrix}\quad h_e(k)=\mathbf{w}_1(k)-h_c[\mathbf{x}(k),\mathbf{w}_2(k),\mathbf{u}_c(k)] \end{aligned} \tag{31}$$

The control step, given by (17) and (18), is formulated as

$$\begin{aligned} &\mathbf{u}_c(k)=\mathbf{u}_{i+1}(k)\quad f(k)=f_c[\mathbf{x}(k),\mathbf{u}_c(k),\mathbf{w}_2(k)]\\ &\mathbf{w}(k)=\begin{bmatrix}\mathbf{w}_1(k)\\ \mathbf{w}_2(k)\end{bmatrix}=\begin{bmatrix}\mathbf{y}_r(k)\\ \boldsymbol{\alpha}_i(k)\end{bmatrix}\quad h_e(k)=\mathbf{w}_1(k)-h_c[\mathbf{x}(k),\mathbf{u}_c(k),\mathbf{w}_2(k)] \end{aligned} \tag{32}$$

Additional objectives for the optimization problems (15) and (17) can be introduced by defining multiple outputs $\mathbf{e}$. Constraints on $\boldsymbol{\alpha}$ or $\mathbf{u}$ can be defined by setting the corresponding elements of $\overline{\mathbf{v}}_{\min}$ and $\overline{\mathbf{v}}_{\max}$. Other constraints can be added by defining constrained outputs $\mathbf{z}$ and providing the corresponding $\mathbf{z}_{\min}$ and $\mathbf{z}_{\max}$.

6. Experimental validation

The generalized ILC approach is validated experimentally on a lab scale model of an overhead crane with fixed cable length, which is shown in Fig. 2 and drawn schematically in Fig. 3. The position of the cart is denoted by x_c, the position of the load by x_l, and the length of the cable is denoted as l. The angle of the load with respect to the vertical is θ. The single input u to the system is a desired velocity of the cart, which is tracked by an internal velocity feedback controller. To avoid saturation of the velocity controller, the input signal must be in the interval $[-10, 10]$ V, with a slew rate in the interval $[-10, 10]$ V/s. The single output is the load position, which is calculated as $x_l = x_c + l \sin\theta$. The cart position x_c is measured with an encoder on the cart, while a second encoder measures θ in the point where the cable is attached. The length l is fixed at 50 cm, and therefore the crane has a fixed resonance frequency of 0.705 Hz. The goal is to control the load position without unwanted oscillation, without exact knowledge of the cable length. Therefore a model is constructed using $l = 55$ cm.

6.1. Nonlinear model

The dynamics of the crane are governed by the following nonlinear equation [34]:

$$l\ddot{\theta} + \ddot{x}_c \cos\theta + g \sin\theta = 0. \tag{33}$$

or in discrete form

$$l\ddot{\theta}(k) + \ddot{x}_c(k) \cos\theta(k) + g \sin\theta(k) = 0. \tag{34}$$

The relation between the input u and the actual cart velocity $\dot{x}_c$ is modeled as a first order system

$$\frac{\dot{X}_c(s)}{U(s)} = \frac{A}{\tau s + 1}. \tag{35}$$

Discretizing this equation with a forward Euler method yields

$$\frac{\dot{X}_c(z)}{U(z)} = \frac{AT_s}{\tau(z-1) + T_s} \tag{36}$$

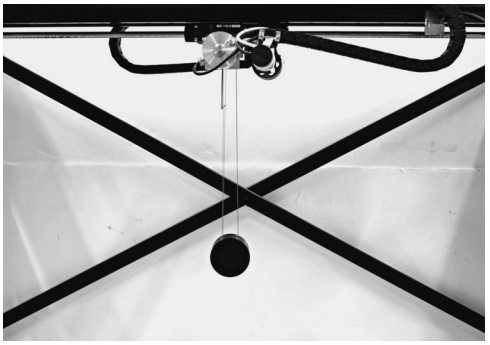


Fig. 2. Picture of the lab scale overhead crane.

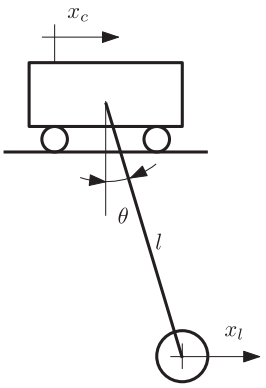


Fig. 3. Schematic representation of the lab scale overhead crane.

with T_s the sample time, and k the discrete time instant. Rearranging this expression and taking the reverse z-transform leads to

$$\dot{x}_c(k+1) = \left[1-\frac{T_s}{\tau}\right]x_c(k)+\frac{AT_s}{\tau}u_c(k) \tag{37}$$

Again using a forward Euler approach, an expression can be found for the instantaneous acceleration $\ddot{x}_c(k)$ and angular acceleration $\ddot{\theta}(k)$

$$\ddot{x}_c(k) = \frac{1}{T_s}[x_c(k+1)\dot{-}x_c(k)]$$
$$\ddot{\theta}(k) = \frac{1}{T_s}[\theta(k+1)\dot{-}\theta(k)]. \tag{38}$$

Inserting equations (37) and (38) into (34) leads to

$$\dot{\theta}(k+1) = \dot{\theta}(k)+\frac{T_s}{\tau l}[x_c-Au(k)]\cos\theta(k)-\frac{gT_s}{l}\sin\theta(k) \tag{39}$$

Eqs. (39) and (37) form the basis of the following 4th order, discrete-time, nonlinear state space model, with $\mathbf{x} = [x_c,\dot{x}_c,\theta,\dot{\theta}]^T$:

$$\mathbf{x}(k+1) = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & p_1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & p_2\cos[x_3(k)] & 0 & 1 \end{bmatrix}\mathbf{x}(k)+\begin{bmatrix} 0 \\ 0 \\ 0 \\ p_3\sin[x_3(k)] \end{bmatrix}+\begin{bmatrix} 0 \\ p_4 \\ 0 \\ p_5\cos[x_3(k)] \\ 0 \end{bmatrix}u(k) \tag{40}$$

$$\hat{y}(k) = x_1(k)+l\sin[x_3(k)] \tag{41}$$

The model parameters p_i for $i\in[1,5]$ are functions of A,τ,T_s,l and g, and are given by

$$p_1 = 1-\frac{T_s}{\tau},\quad p_2 = \frac{T_s}{l\tau},\quad p_3 = \frac{-gT_s}{l},\quad p_4 = \frac{AT_s}{\tau},\quad p_5 = \frac{AT_s}{l\tau} \tag{42}$$

with

$$A = 0.0496,\quad \tau = 0.0128\text{ s},\quad T_s = 0.05\text{ s},\quad l = 55\text{ cm}. \tag{43}$$

In all experiments, the initial state is assumed to be zero, which means both the cart and the load are in rest at the start of the motion.

Two applications of the generalized ILC approach are applied in the experiments: A tracking control problem for which a reference trajectory must be followed over the entire time horizon, and a point-to-point motion control problem for which a fixed load position must be reached in a given amount of time without specifying a trajectory.

6.2. Reference trajectory tracking control

The aim of the experiments is to track a wave shaped reference trajectory, shown in Fig. 4, without residual swinging.

In a first experiment, the nominal model is corrected in a non-parametric approach, additive to the output. In practice, the main source of model plant mismatch is the error in the modeled length l, but in the first experiment the source of the model error is assumed to be unknown. In a second experiment, the model correction is targeted specifically at the parameter l in the model, and both approaches are compared.

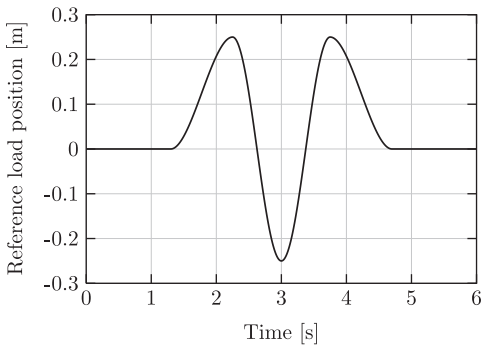


Fig. 4. Reference position for the load for the tracking control experiment.

6.2.1. Non-parametric model correction

The non-parametric model correction adds the term $\boldsymbol{\alpha}$ to the output equation, such that the corrected model is a combination of (40) and

$$\hat{y}_c(k) = x_1(k) + l\sin[x_3(k)] + \alpha(k) \tag{44}$$

The input signal for the first trial is found by solving the control step with the nominal model, so $\boldsymbol{\alpha} = 0$. The generalized ILC algorithm is applied with the following properties:

The *estimation step* is performed without bounds on the optimization variables. An additional objective is added to minimize the trial-by-trial difference of $\boldsymbol{\alpha}$, in order to make the algorithm less sensitive to trial varying disturbances. Therefore there are 2 minimized outputs

$$\mathbf{e}(k) = \begin{bmatrix} e_1(k) \\ e_2(k) \end{bmatrix} = \begin{bmatrix} y_i(k) - \hat{y}_c(k) \\ \alpha_i(k) - \alpha(k) \end{bmatrix} \tag{45}$$

The diagonal $\overline{\mathbf{q}}_\alpha$ of the weighting matrix $\mathbf{Q}_\alpha$ consists of $q_1(k) = 1 \times 10^8$ and $q_2(k) = 1 \times 10^2$ for each k as weights on $e_1(k)$ and $e_2(k)$ respectively.

For the *control step*, an additional objective is to minimize the sample by sample difference of the input signal, $\delta u(k) = u(k) - u(k-1)$, in order to smooth $\mathbf{u}$. Therefore two minimized outputs are defined

$$\mathbf{e}(k) = \begin{bmatrix} e_1(k) \\ e_2(k) \end{bmatrix} = \begin{bmatrix} y_r(k) - \hat{y}_c(k) \\ \delta u(k) \end{bmatrix} \tag{46}$$

In this case, the diagonal $\overline{\mathbf{q}}_u$ of the weighting matrix $\mathbf{Q}_u$ consists of $q_1(k) = 1 \times 10^8$ and $q_2(k) = 5 \times 10^2$ for each k as weights on $e_1(k)$ and $e_2(k)$ respectively. In order to comply with the limitations on input and input slew rate, two constrained outputs $\mathbf{z}$ are defined

$$\mathbf{z}(k) = \begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} = \begin{bmatrix} u(k) \\ \delta u(k) \end{bmatrix} \tag{47}$$

with the corresponding intervals $[\mathbf{z}_{\min}, \mathbf{z}_{\max}]$ equal to $[-10, 10]$ for $z_1(k)$, and $[-0.5, 0.5]$ for $z_2(k)$, for all k.

The experiment is performed twice, indicated as experiment 1a and 1b, with a slight change in settings. Fig. 5 shows the tracking error measured during trials 1, 2, 3, and 10 of experiment 1a. It is clear that the initial trial shows a large tracking error, due to the model plant mismatch. However, the tracking error reduces over multiple trials, and after 10 trials the tracking error is near zero, except for a small remaining error starting from 2 s and 3.5 s. It will be shown later that this is due to the applied constraints. The correction signal $\boldsymbol{\alpha}$ has converged to the value $\boldsymbol{\alpha}_\infty = \mathbf{y}_r - \hat{P}(\mathbf{u}_\infty)$, and captures the model error.

Fig. 6 shows the load angle θ for trials 1, 2, 3 and 10. The load reaches angles of over 20°, which justifies the use of a nonlinear model to describe the system dynamics. However, it can also be observed that the ILC algorithm has introduced an unwanted oscillatory behavior before and after the wave-like motion of the reference trajectory. This can also be observed from Fig. 7. This figure shows the input and input slew rate of the first trial, and trial 10, of experiment 1a. It is clear that the initial input signal remains zero until the motion starts, but after 10 trials an oscillation has been introduced in the input signal before and after the motion. This oscillation does not increase over successive trials, and from Fig. 5 it is clear that the effect on the tracking error is negligible. However, this behavior is undesirable, because when the cart stops at the end of the time horizon, a large transient oscillation of the load occurs.

In order to remove this unwanted oscillation in a second experiment, called experiment 1b, the input constraint interval $[\mathbf{z}_{\min}, \mathbf{z}_{\max}]$ is set to zero at time instants before and after the desired motion, as can be seen from Fig. 8. The result of this approach on the input and input slew rate in experiment 1b is shown in Fig. 9. It is clear that the oscillation of the input signal before and after the motion has been removed. It had already been observed from Fig. 5 that the effect of this

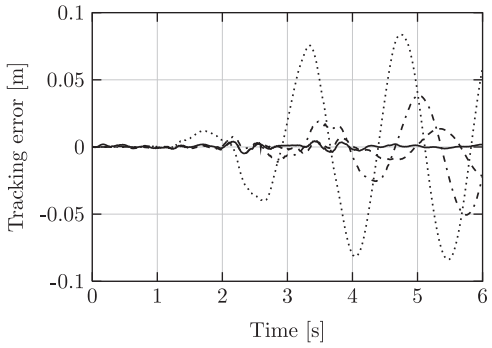


Fig. 5. Tracking error for trial 1 (dotted), 2 (dash-dotted), 3 (dashed) and 10 (full) of experiment 1a.

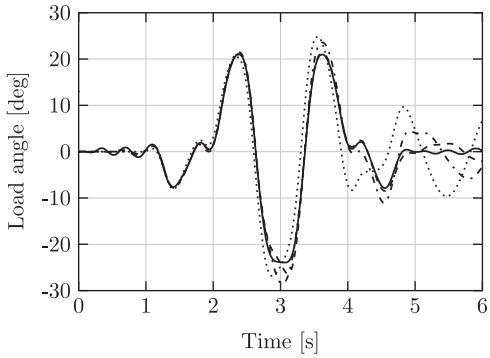


Fig. 6. Load angle for trial 1 (dotted), 2 (dash-dotted), 3 (dashed) and 10 (full) of experiment 1a.

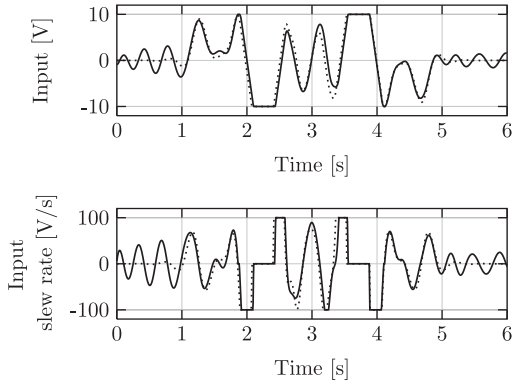


Fig. 7. Input signal (top) and input signal slew rate (bottom) for trials 1 (dotted) and 10 (full) of experiment 1a.

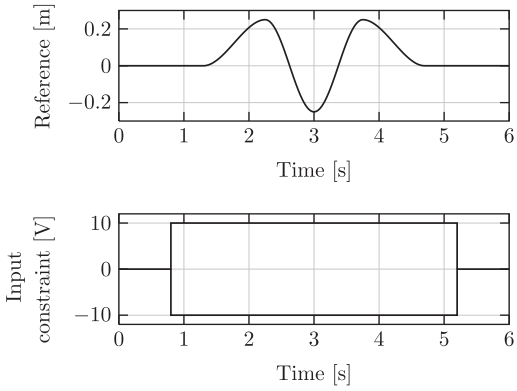


Fig. 8. Reference signal (top) and input constraint interval (bottom) to prevent unnecessary cart motion in experiment 1b.

oscillation on the tracking error over the time horizon is very small. It is therefore expected that experiments 1a and 1b show similar convergence behavior. Fig. 10 shows the evolution of the tracking error for 10 trials of both experiments. In both experiments, a monotonic convergence in 8 trials is observed. A non-zero tracking error remains after convergence, because the input signal hits the constraints around 2 and 3.5 s, which causes a tracking error that cannot be further reduced by learning, as was also observed in Fig. 5.

6.2.2. Non-parametric versus parametric model correction

In experiments 1a and 1b, the correction term $\boldsymbol{\alpha}$ was applied additive to the model output. This means no information is used about the source of the model plant mismatch, in other words which parameter values are not accurately known, since in practical applications this information is often not available. If it is available, the developed ILC approach can apply a correction term directly to the model parameter. This approach is validated in experiments 2a and 2b.

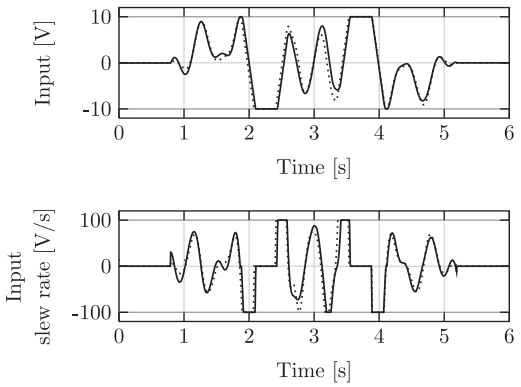


Fig. 9. Input signal (top) and input signal slew rate (bottom) for trials 1 (dotted) and 10 (full) of experiment 1b.

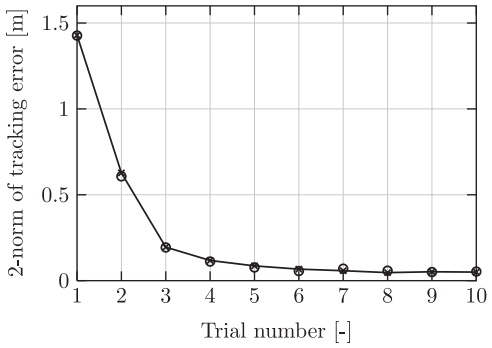


Fig. 10. Evolution of the 2-norm of the tracking error for 10 trials of experiment 1a (circle) and 1b (×).

Note that for all experiments, the real cable length is $l = 0.5$ m, while the model (40) uses $l = 0.55$ m, as written in (43). Therefore the corrected model is constructed with $\boldsymbol{\alpha}$ additive to l, so the parameters p_2,p_3 and p_5 use $l+\alpha(k)$ instead of l. The generalized ILC approach estimates a value of α for each time instant k, while this application requires only a scalar estimate. This is achieved by a suitable choice of constraints in the estimation step. In experiment 2a, the generalized ILC algorithm is applied with the following properties:

For the *estimation step*, a constrained output $\mathbf{z}$ is defined, in order to guarantee that a constant value is estimated for α over all time instants

$$\mathbf{z}(k) = [\delta\alpha(k)] \tag{48}$$

with $\delta\alpha(k) = \alpha(k) - \alpha(k-1)$. Since no change of α is desired from $k-1$ to k, the constraint interval $[\mathbf{z}_{\min},\mathbf{z}_{\max}]$ is set to [0,0]. This experiment uses the same minimized outputs (45) and weights as experiment 1a, to achieve a trial domain regularization of α.

The *control step* is also performed using the same settings as in experiment 1a, using the same minimized outputs 46 and constrained outputs 47, and using the same weights and bounds.

Fig. 11 compares the evolution of the tracking error over 10 trials of experiment 1b with experiment 2a. It is clear that the convergence speed has been increased by targeting the model correction to the main source of model plant mismatch, such that the ILC algorithm converges after 1 trial. However, experiment 2a also shows a much larger tracking error after convergence. This indicates that the inaccurate estimate of l in the nominal model is not the only source of model plant mismatch. Due to the scalar value of $\boldsymbol{\alpha}$, this error cannot be compensated. Therefore a second experiment, called 2b, is performed, for which the correction $\boldsymbol{\alpha}$ is allowed a small variation over the time instants k, in order to also capture the remaining model error. Therefore, in the *estimation step*, the constrained output $\mathbf{z}(k) = \delta\alpha(k)$ is replaced by a minimized output, such that

$$\mathbf{e}(k) = \begin{bmatrix} e_1(k) \\ e_2(k) \\ e_3(k) \end{bmatrix} = \begin{bmatrix} y_i(k) - \hat{y}_c(k) \\ \alpha_i(k) - \alpha(k) \\ \delta\alpha(k) \end{bmatrix} \tag{49}$$

with the corresponding weight matrix $\mathbf{Q}_u$ using the diagonal elements $q_1(k) = 1 \times 10^8$,$q_2(k) = 1 \times 10^3$ and $q_3(k) = 1 \times 10^9$ for each k. The *control step* is performed with the same settings as experiment 2a. The result is also shown in Fig. 11.

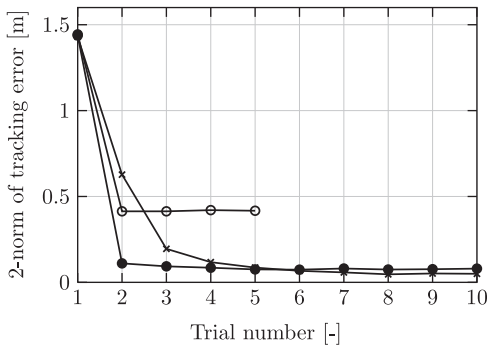


Fig. 11. Evolution of the 2-norm of the tracking error in experiment 3, using non-parametric correction (×), scalar parametric correction (circle), and time varying parametric correction (dot).

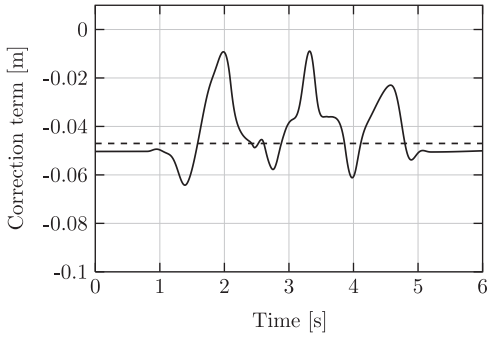


Fig. 12. Scalar parametric correction, 2a (dashed) and time varying parametric correction, 2b (full) in experiment 2.

It is clear that a fast convergence is achieved, similar to experiment 2a. However, the remaining tracking error after convergence has been greatly reduced, to a level only slightly above the remaining error of experiment 1b. This is due to the choice of weight term $q_3(k)$, which is used to smooth the estimated $\boldsymbol{\alpha}$. Reducing $q_3(k)$ results in a smaller tracking error, but reduces the convergence speed of the underlying NLP problem in IPOPT, since the model is very sensitive to changes in l, and l appears in the denominator of p_2,p_3 and p_5.

Fig. 12 shows a comparison of the estimated value of $\boldsymbol{\alpha}$ for experiments 2a and 2b, after convergence. Note that the true model error of l is −0.05 m. The time varying correction term of experiment 2b has converged to this true value before and after the motion, but shows a deviation from this value during the motion, in order to compensate for other model errors. The scalar estimate of experiment 2a does not reach the true value, but compromises between the model error due to the mismatch of l and other model errors.

6.3. Point-to-point motion control

The goal of the point-to-point motion control problem is to make the output of the system reach a given setpoint value in a given amount of time, without specifying the trajectory that the system needs to follow to reach that setpoint, and without control action after the given time. Two experiments, 3a and 3b, are performed.

In experiment 3a, the setpoint is 30 cm from the initial load position, and the motion time is 1.1 s. The load must remain in the setpoint for 0.5 s, and then move back to the initial position, again in 1.1 s. The developed ILC approach is applied with the same settings as experiment 1b, for both steps, except for 2 adaptations.

In the *control step*, the weight matrix $\mathbf{Q}_u$ of the minimized outputs is set to 0 at all time instants before the given motion time. Recall from experiment 1b that the minimized outputs are

$$\mathbf{e}(k) = \begin{bmatrix} e_1(k) \\ e_2(k) \end{bmatrix} = \begin{bmatrix} y_r(k) - \hat{y}_c(k) \\ \delta u(k) \end{bmatrix} \tag{50}$$

and therefore the diagonal $\overline{\mathbf{q}}_u$ of the weighting matrix $\mathbf{Q}_u$ consists of

$$q_1(k) = \begin{cases} 1 \times 10^8 & \text{for } k \in (220,320] \text{ and } (540,800] \\ 0 & \text{for } k \in [0,220] \text{ and } (320,540] \end{cases} \tag{51}$$

and $q_2(k) = 5 \times 10^2$. After the given motion time, no control action is allowed, so the bounds on the first constrained output z are set to vary over the time horizon as well

$$[z_{1,\min}(k), z_{1,\max}(k)] = \begin{cases} [0,0] & \text{for } k \in (220,320] \text{ and } (540,800] \\ [-10,10] & \text{for } k \in [0,220] \text{ and } (320,540] \end{cases} \tag{52}$$

The results of experiment 3a, which were carried out for 8 trials, are shown in Figs. 13 and 14. It is clear that for the initial input signal, the load keeps swinging at the time instants where no control action is given, so the output cannot be kept at the required setpoint. After 8 trials however, no residual swinging is observed after arriving at the setpoint. From Fig. 14, it is clear that control action is only given at the time instants where it is allowed. The input signal during the motion is determined by the constraints and additional objectives, and the objective to reach the setpoint in time. Example 3a is set up with a given motion time that is large enough for the system to complete the required motion. A second experiment, 3b, is carried out to study the behavior of the algorithm for a motion time that is too short to complete the motion, given the system limitations. First, the same motion of 30 cm in 1.1 s is required (and needs to be held for 1 s), but it is followed by a motion of 15 cm in 0.75 s, and another motion of 15 cm in 1 s. The result is shown in Fig. 15, again for 8 trials. It is clear that the second setpoint, at 15 cm, cannot be reached in the given motion time. The ILC algorithm converges to the most optimal

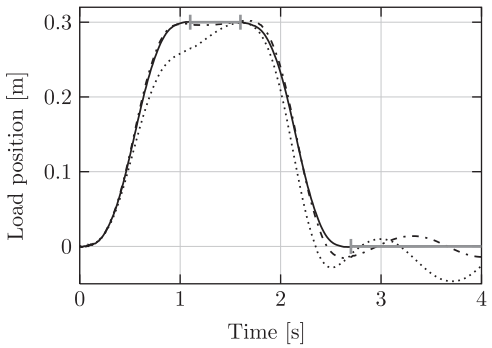


Fig. 13. Load position for trial 1 (dotted), 2 (dash-dotted) and 8 (full) of experiment 3a (setpoints shown in gray).

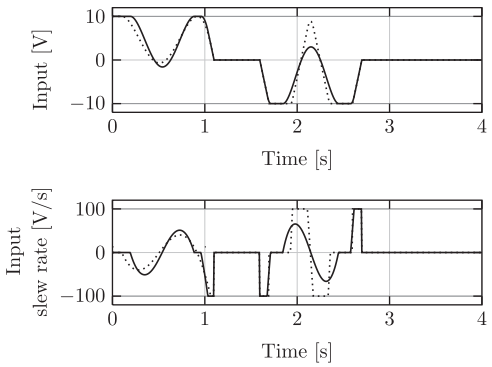


Fig. 14. Input signal (top) and input signal slew rate (bottom) for trials 1 (dotted) and 8 (full) of experiment 3a.

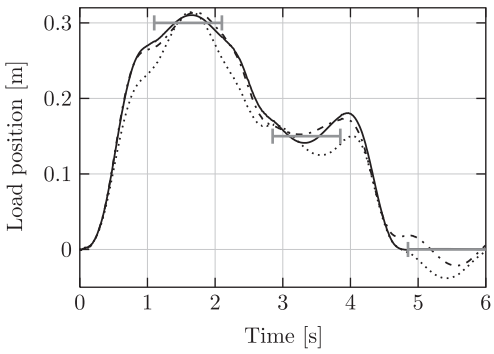


Fig. 15. Load position for trial 1 (dotted), 2 (dash-dotted) and 8 (full) of experiment 3b (setpoints shown in gray).

Table 1
Computation times per step in seconds, as a function of reference length, for experiment 1a.

N	1200	2100	3000
Time spent in IPOPT	0.856	1.184	1.86
Time spent in function evaluations	0.052	0.104	0.136
Number of IP iterations	23	20	21
Number of optimization variables	7200	12 600	18 000
Number of constraints	10 800	18 900	27 000

feasible solution, in the sense that the 2-norm of the difference between output and setpoint is minimal. The third setpoint can be reached without residual swinging of the load, after convergence of the algorithm.

It is interesting to note that the first point-to-point motion is not completed with the same accuracy as in experiment 3a, even though it has the same motion time. This is because the developed ILCapproach finds the input signal that optimizes the motion over the entire time horizon, which includes all three point-to-point motions, and does not consider these motions to be independent.

7. Discussion

The developed, generalized norm optimal ILC algorithm is based on the solution of two NLP problems: one for the estimation of the model correction term, and one for the calculation of the next trial's input signal. The design parameters of the algorithm are the definition of additional objectives and constraints, and the selection of appropriate weights and bounds, for both NLP problems. This offers a lot of freedom to adapt the algorithm to the considered application.

An example, which was used in the experimental validation in Section 6, is to apply additional objectives to deal with trial varying disturbances, such as measurement or process noise. The estimation of a model correction term based on a noisy measurement can lead to an amplification of high frequent components in the calculated input signal, and this can be countered in several ways. One way is to apply a minimization of the time difference of the new input signal, $\delta u(k) = u(k)-u(k-1)$, which acts as a low pass filter for u and therefore smooths the signal. Another option is to apply a trial domain regularization to either the correction term or the input, to minimize the change of the signal from one trial to the next, and therefore making the algorithm less sensitive to trial varying disturbances.

The addition of such regularization terms affects the performance of the algorithm. Trial domain regularization typically slows down the convergence, since the updates after each trial are smaller, while smoothing of the input signal typically leads to a larger remaining tracking error after convergence, since the low pass effect of the regularization is not perfect. The presented experimental results use a combination of both strategies, which results in reasonably fast convergence and smooth input signals.

An advantage of the developed approach is that the structure of the model correction can be chosen freely. This allows the algorithm to use information on the source of model errors in the nominal model, by applying correction directly to certain parameters. This parametric model correction can also be used to detect other sources of model errors, or to assess the quality of the nominal model or model structure.

A drawback of optimization based ILC approaches is that it is not always easy to select appropriate values for the weight terms in the objective functions, since it is hard to quantify the effect of such regularization terms on the solution of the optimization problem, and therefore on the performance of the ILC algorithm. Model quality or uncertainty is often better assessed in the frequency domain, but it is hard to translate design choices from the frequency domain to the time domain optimization problems.

The presented ILC approach solves two NLP problems, and therefore is subject to the specific issues related to these problems, such as the existence of local minima, the need for a suitable initialization of the optimization variables, and possible slow convergence or divergence of the interior point method. The implementation with IPOPT is not guaranteed to provide a solution under all circumstances, and may require some tuning of the applied regularization terms to improve the convergence of the optimization algorithm.

The generalized ILC algorithm assumes one or more model correction terms of length N, in other words of the same length as the input and output of the system. For scalar parametric model correction, only a single estimate is required, and therefore the proposed implementation introduces an unnecessary increase in the computational cost for such a correction strategy.

However, due to the sparse structure of the Jacobian matrices in both optimization problems, which is introduced by including the states into the optimization variable vector, the presented algorithm has a low computational cost, which increases linearly with N. For example, in experiment 1a, described in Section 6.2.1, IPOPT requires less than 1 s to solve each step, run on a PC with a quad-core Intel Xeon CPU, running at 2.53 GHz, using 1 Gb of the available 12 Gb, in Linux 3.0. Since this example has 1 input and 1 correction term, the calculation time for both steps is roughly equal. Experiment 1a has been repeated with longer reference trajectories, for which the periods of the wave in the middle of the trajectory have been increased (experiment 1a uses 1 period, see Fig. 4). The total calculation time per step for three different reference lengths is written in Table 1, together with the number of optimization variables and constraints, and the required iterations of the

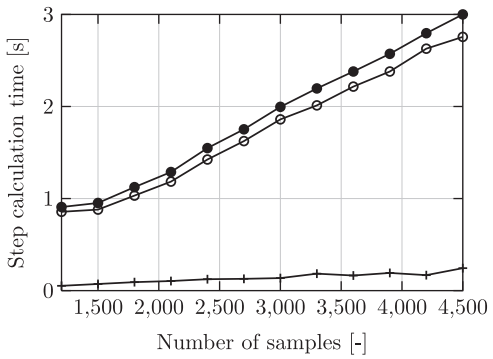


Fig. 16. Total calculation time (dot), time spent in IPOPT (circle), and time spent in function evaluations (+) per step as a function of reference trajectory length.

interior point method. Note that in this example, the number of states is 5, due to the applied constraint on δu, and the number of constrained outputs is 2. There are $5N$ equality constraints and $2N$ inequality constraints, which have a lower and upper bound. These computation times are in the same order of magnitude as other modern optimization based ILC algorithms. For example, the authors of [12] describe an example of a 5th order model with 2 inputs and 6 constraints per sample, such that the number of optimization variables is 226, with 1356 constraints. A total calculation time of 0.2 s is achieved on a standard desktop PC, in combination with an offline precalculation which takes 1.5 s. The presented approach needs about 2 s (for 2 steps) in the case $N=1200$. This is a 10-fold of the calculation time compared to [12], but the problem has around 30 times the number of optimization variables. Furthermore, the calculation time required by IPOPT in the proposed approach scales linearly with N, as can be observed from Fig. 16. This figure shows the total calculation time, time spent in IPOPT, and time spent in function evaluations, for increasing problem size. Note that the number of iterations required for the solution of the NLP problem is not equal, for example the case $N=1200$ requires 23 iterations, whereas $N=2100$ requires 20.

8. Conclusions

This paper discusses a generalization of norm optimal ILC for nonlinear systems with constraints. The approach is based on the explicit estimation of a model correction signal. An optimal value of this signal is estimated after each trial, based on the known input and measured output of the past trial. The corrected model is then used in a second step to calculate an optimal input signal for the next trial. It is shown that under a number of conditions, the optimal solution of the next trial's input signal can be written in closed form, and this solution corresponds to the update law of the linear norm optimal ILC.

Both the estimation and the control step are formulated as an NLP problem. A general formulation and solution strategy for this class of problems is also discussed in this paper. It is shown that the formulation leads to sparse and block structured matrices, that can be efficiently processed by IPOPT, a sparse implementation of an interior point method.

The proposed ILC approach is versatile and can be easily adapted by selecting appropriate weights, constraints, or additional objectives of the optimization problems. It can exploit information about the source of the model error, by using a suitable structure of the model correction term. It can handle a broad class of nonlinear models, including black box models, and it is applicable both to tracking control problems and point to point motion control problems.

Acknowledgments

This work benefits from FWO-project G.0422.08, K.U.Leuven-BOF PFV/10/002 Center-of-Excellence Optimization in Engineering (OPTEC), IWT-SBO-project 80032 (LeCoPro) and the Belgian Programme on Interuniversity Attraction Poles, initiated by the Belgian Federal Science Policy Office (DYSCO).

References

[1] D.A. Bristow, M. Tharayil, A.G. Alleyne, A survey of iterative learning control: a learning-based method for high-performance tracking control, IEEE Control Syst. Mag. 26 (2006) 96–114.
[2] H.-S. Ahn, K. Moore, Iterative Learning Control: Robustness and Monotonic Convergence for Interval Systems (Communications and Control Engineering), Springer, 2007.
[3] S. Arimoto, S. Kawamura, F. Miyazaki, Bettering operations of robots by learning, J. Robotic Syst. 1 (1984) 123–140.
[4] J. De Cuyper, M. Verhaegen, J. Swevers, Off-line feed-forward and feedback control on a vibration rig, Control Eng. Pract. 11 (2003) 129–140.
[5] S. Gunnarsson, M. Norrlof, On the design of ILC algorithms using optimization, Automatica 37 (2001) 2011–2016.
[6] N. Amann, D. Owens, E. Rogers, Iterative learning control for discrete time systems with exponential rates of convergence, Proc. Inst. Electr. Eng. Control Theory Appl. 143 (1996) 217–224.
[7] W.B.J. Hakvoort, Iterative Learning Control for LTV Systems with Applications to an Industrial Robot, Ph.D. Thesis, Universiteit Twente, Enschede, 2009.

[8] C. Chien, A discrete iterative learning control for a class of nonlinear time-varying systems, IEEE Trans. Autom. Control 43 (1998) 748–752.
[9] M. Arif, T. Ishihara, H. Inooka, A learning control for a class of linear time varying systems using double differential of error, J. Intell. Robotic Syst. 36 (2003) 223–234.
[10] J.-X. Xu, Y. Tan, Linear and Nonlinear Iterative Learning Control, Springer, 2003.
[11] J. Ghosh, B. Paden, Iterative learning control for nonlinear nonminimum phase plants, Meas. Control 123 (2001) 21–30.
[12] A.P. Schoellig, F.L. Mueller, R. D'Andrea, Optimization-based iterative learning for precise quadrocopter trajectory tracking, Auton. Rob. 33 (2012) 103–127.
[13] K. Smolders, M. Volckaert, J. Swevers, Tracking control of nonlinear lumped mechanical continuous-time systems: a model-based iterative learning approach, Mech. Syst. Signal Process. 22 (2008) 1896–1916.
[14] Y. Chen, Z. Gong, C. Wen, Analysis of a high-order iterative learning control algorithm for uncertain nonlinear systems with state delays, Automatica 34 (1998) 345–353.
[15] T. Al-Towaim, P. Lewin, Higher order ILC versus alternatives applied to chain conveyor systems, in: Proceedings of the 15th IFAC World Congress, 2002.
[16] M. Sun, D. Wang, Analysis of nonlinear discrete-time systems with higher-order iterative learning control, Dyn. Control 11 (2001) 81–96.
[17] Y. Chen, C. Wen, M. Sun, A robust high-order P-type iterative learning controller using current iteration tracking error, Int. J. Control 68 (1997) 331–342.
[18] T.-Y. Doh, J.-H. Moon, K. Jin, M. Chung, Robust iterative learning control with current feedback for uncertain linear systems, Int. J. Syst. Sci. 30 (1999) 39–47.
[19] N. Amann, D.H. Owens, E. Rogers, A. Wahl, An H_∞ approach to linear iterative learning control design, Int. J. Adaptive Control Signal Process. 10 (1996) 767–781.
[20] D. De Roover, O. Bosgra, Synthesis of robust multivariable iterative learning controllers with application to a wafer stage motion system, Int. J. Control 73 (2000) 968–979.
[21] H.-S. Ahn, K. Moore, Y.Q. Chen, LMI approach to iterative learning control design, IEEE Mt. Workshop Adaptive Learn. Syst. (2006) 72–77.
[22] J. Lee, K. Lee, W. Kim, Model-based iterative learning control with a quadratic criterion for time-varying linear systems, Automatica 36 (2000) 641–657.
[23] J. Xu, Y. Tan, T. Lee, Iterative learning control design based on composite energy function with input saturation, Automatica 40 (2004) 1371–1377.
[24] B. Chu, D.H. Owens, Iterative learning control for constrained linear systems, Int. J. Control 83 (2010) 1397–1413.
[25] P. Janssens, G. Pipeleers, J. Swevers, Model-free iterative learning control for LTI systems and experimental validation on a linear motor test setup, in: American Control Conference, 2011, pp. 4287–4292.
[26] M. Volckaert, J. Swevers, M. Diehl, A two step optimization based iterative learning control algorithm, in: ASME Dynamic Systems and Control Conference, 2010.
[27] M. Volckaert, M. Diehl, J. Swevers, Iterative learning control for nonlinear systems with input constraints and discontinuously changing dynamics, in: American Control Conference (ACC), 2011, pp. 3035–3040.
[28] M.Q. Phan, R.W. Longman, K.L. Moore, Unified formulation of linear iterative learning control, Adv. Astronaut. Sci. 105 (2000) 93–111.
[29] J. Ratcliffe, P. Lewin, E. Rogers, J. Hatonen, D. Owens, Norm-optimal iterative learning control applied to gantry robots for automation applications, IEEE Trans. Robotics 22 (2006) 1303–1307.
[30] N. Amann, D.H. Owens, E. Rogers, Predictive optimal iterative learning control, Int. J. Control 69 (1998) 203–226.
[31] E. Baake, M. Baake, H. Bock, K. Briggs, Fitting ordinary differential equations to chaotic data, Phys. Rev. A 45 (1992) 5524–5529.
[32] A. Wächter, L.T. Biegler, On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming, Math. Programming 106 (2006) 25–57.
[33] J. Nocedal, S.J. Wright, Numerical Optimization, Springer, 2006.
[34] M. Fliess, J. Levine, P. Rouchon, Flatness and defect of nonlinear systems: introductory theory and examples, Int. J. Control 61 (1995) 1327–1361.