Iterative Learning Control Strategy for a Furuta Pendulum System with Variable-Order Linearization

Ricardo Binz, Stanislav Aranovskiy

Preprints, First IFAC Modeling, Estimation and Control Conference
October 24-27, 2021. Online and UT Austin, USA

Iterative Learning Control Strategy for a Furuta Pendulum System with Variable-Order Linearization

Ricardo Binz * **Stanislav Aranovskiy** *

* *IETR – CentaleSupélec, Avenue de la Boulaie, 35576 Cesson-Sévigné, France.*

Abstract: We consider Iterative Learning Control for the Furuta Pendulum nonlinear mechanical system, where the goal is to learn the input torque such that the pendulum angle follows a reference. We show that the linearization of the considered system is of variable trajectory-dependent order and thus some existing solutions do not apply. We propose a novel method based on the observability matrix inversion allowing to deal with the variable-order minimum realization. The applicability of the proposed method is illustrated with simulations.

Keywords: Iterative learning control, Furuta Pendulum, mechatronic systems, observability

1. INTRODUCTION

Iterative Learning Control (ILC) is a popular methodology for improving transient response and tracking performance when a control task is repeatedly executed. When the classical feedback control produces (up to non-repetitive disturbances) the same trajectory with the same tracking error on each trial, the ILC iteratively adjusts the command between the trials aiming for zero tracking error in plant operation. After each trial, the observed system's behavior is compared to the desired reference. The behavior deviation is then used for the iterative control signal update (learning), improving the control task execution performance. Uchiyama (1978) and Arimoto et al. (1984) initially proposed the ILC for mechanical systems, and nowadays, it is widely used for a broad class of control problems, e.g., in the process control, see the surveys by Bristow et al. (2006) and Wang et al. (2009). It is worth noting one can use the ILC together with a stabilizing feedback control or as a stand-alone feedforward governor. In the stand-alone case, the plant operates in open-loop mode during a trial, and the control signal is then updated between the trials.

The ILC is typically applied to repetitive problems, such as repetitive trajectories in robotics (Norrlof (2002)), chemistry reactors operation (Lee and Lee (2007)), or high-performance positioning systems (Huang et al. (2013)). The use of the information contained in the repetitive task executions distinguishes the ILC from the direct or indirect adaptive control (Åström and Wittenmark (2013); Sastry and Bodson (2011)). Indeed, in adaptive control, one updates the controller parameters directly or indirectly, e.g., via online system identification. In contrast, the ILC methods update the control signal itself, and thus it can be considered a non-parametric approach. Additionally, adaptive control methods typically work along a single-trial trajectory and do not benefit from the repetitive nature of the executed task.

Considering the control solutions motivated by the recent advances in machine learning, a certain similarity can be found between the ILC and the Reinforcement Learning (RL) methods, as was shown by Zhang et al. (2019). However, the RL is typically a *model-free* approach that learns only from the observed reward and does not use the plant model. In contrast, the ILC methods are often *model-based* and assume that a plant model is available and can be used for the control signal update. Thus, the model-based ILC requires a significantly smaller number of trials to arrive at the optimal (in a certain sense) solution than the model-free RL. This difference becomes crucial when the trials must be physically performed and can hardly be simulated with the desired accuracy, i.e. when the simulation-based learning is not possible.

Due to the repetitive task nature, one standard restriction of the ILC was that the trajectories must always start from the same initial conditions. However, this requirement can be relaxed in practice, see, e.g., Hoelzle et al. (2010); Wu et al. (2019); moreover, an extension to non-repetitive trajectories was recently proposed by Jin (2018a).

Nowadays, an important research direction is the ILC for nonlinear systems, where several solutions based on Lipschitz conditions, contraction analysis, or composite energy functions are available, see Xu (2011); Jin (2018b). One common approach is based on linear ILC techniques applied to linearization of the original nonlinear system along the performed trajectory, see Lu et al. (2017). For example, Beuchert et al. (2018) used this idea studying the trajectory-learning problem for a pendulum-on-a-cart system, where they also restricted the ILC algorithm to learn only for small linearization errors.

The paper goal. In this work, we consider the Furuta Pendulum nonlinear mechanical system, where the linearization is not observable at specific trajectory segments. Our goal is to design an ILC approach to learn a reference trajectory for such a system.

It is worth noting that ILC algorithms were also applied in the context of inverted pendulums in previous works. However, the majority of studies considered the cart-pendulum dynamics: Schöllig and D'Andrea (2009) proposed a combination of the ILC with convex optimization techniques, Gao et al. (2009) presented a PID-type ILC algorithm for the trajectory tracking problem, and Zhang et al. (2008) proposed a solution using a closed-loop ILC algorithm based on Lyapunov theory. In contrast to those works, we consider the Furuta Pendulum dynamics, for which we propose a novel linearization-based solution using the observability matrix pseudo-inversion.

This research is motivated by Beuchert et al. (2018) and extends the results presented therein. More precisely, Beuchert et al. (2018) study the pendulum-on-a-cart system where the cart's position is considered an input for the pendulum's dynamics. The authors linearize the system along its trajectory, yielding a transfer function; this transfer function is further inverted, and the ILC is applied to learn the reference trajectory.

Novelty and Contribution. In this work, we show that the method of Beuchert et al. (2018) cannot be directly applied to the mechanical system in which we are interested since the minimum realization of the linearized system is of variable trajectory-dependent order. To deal with this problem, we propose a novel approach based on the observability matrix pseudo-inversion that allows for the desired trajectory learning. We consider the Furuta Pendulum system choosing the torque as the input signal, and thus no extra position controller is required.

The rest of the paper is organized as follows. The problem statement is given in Section 2, and the (motivating to our research) existing solution is discussed in 3. The proposed ILC strategy is presented in 4, and the illustrative learning results for the considered Furuta Pendulum system are given in Section 5. The paper results are discussed and summarized in Conclusion.

2. PROBLEM STATEMENT

Motivated by the Furuta Pendulum example, we consider a mechanical system with two degrees of freedom (DOF) given by [1]

$$\begin{aligned} &\mathcal{M}(q)\ddot{q} + \mathcal{C}(q,\dot{q}) + \mathcal{G}(q) = \tau, \\ &y = Dq, \end{aligned} \tag{1}$$

where $q = [q_1 \ q_2]^\top$, q_1 and q_2 are the generalized coordinates, τ is the generalized torque considered here as an input signal, $\mathcal{M}(q)$ is the inertia matrix, $\mathcal{C}(q,\dot{q})$ is the vector of the centrifugal and Coriolis forces, and $\mathcal{G}(q)$ is the vector of the gravity force. We assume that both the generalized coordinates q and their time derivatives $\dot{q}$ are available, i.e., measured or estimated. The initial conditions of (1) are $q(0) = q_0 \in \mathbb{R}^2$ and $\dot{q}(0) = \dot{q}_0 \in \mathbb{R}^2$. The scalar signal y is the trajectory output specified by the matrix $D \in \mathbb{R}^{1\times 2}$.

The desired trajectory $y_d(t)$ is defined on the finite time interval $[0, T]$ for some $T > 0$, and the initial conditions of (1) satisfy the desired trajectory, i.e., $y_d(0) = y(0) = Dq_0$ and $\dot{y}_d(0) = \dot{y}(0) = D\dot{q}_0$. Moreover, we assume that the desired trajectory y_d is feasible, i.e., there exists an (unknown) input signal τ_d such that the solution of (1) for $\tau(t) \equiv \tau_d(t)$ satisfies $y_d(t) \equiv y(t)$ for all $t \in [0, T]$. Note that the feasibility of y_d implies that it is bounded and twice differentiable.

The goal is to learn in several trials the open-loop control signal τ capable of generating (with a certain accuracy) the desired trajectory y_d, where the control signal is updated between trials. Let us denote the control action applied at j-th trial as $\tau(t, j)$ and the solution of (1) obtained for $\tau(t, j)$ as $q(t, j)$ and $y(t, j)$, where $j \in \mathbb{N}$ and $t \in [0, T]$. Then an iterative learning control algorithm can be written as

$$\tau(t, j+1) = \mathcal{ILC}\left(y_d(t), q(t, j), \dot{q}(t, j), \tau(t, j)\right). \tag{2}$$

In other words, the ILC algorithm (2) updates the control input based on the desired trajectory, the control applied on the current trial, and the observed system behavior.

With these definitions, the goal is to design an ILC algorithm of the form (2) such that given the feasible trajectory y_d and the desired accuracy $\epsilon > 0$, there exists $j_0 \in \mathbb{N}$ such that the solution of (1) with the input $\tau(t, j_0)$ satisfies for all $t \in [0, T]$

$$|y(t, j_0) - y_d(t)| < \epsilon.$$

3. EXISTING SOLUTION

The current research is motivated by Beuchert et al. (2018), where the authors studied ILC design for a pendulum-on-a-cart system. The authors considered the cart's position as an input signal [2] rendering the system (1) to the 1-DOF system

$$\begin{aligned} \ddot{q}_1 &= c_1\left(c_2\left(\ddot{q}_2\cos(q_1) + g\sin(q_1)\right) - c\dot{q}_1\right), \\ y &= q_1, \end{aligned} \tag{3}$$

where q_1 is the pendulum angle (the output signal), q_2 is the cart position (the input signal), and c_1, c_2, c, and g are constant parameters.

Define the tracking error $e := y_d - y$. Assuming that for the given desired trajectory y_d, there exists the corresponding input signal q_2^d and linearizing(3) around a measured (sampled) trajectory q_1^s, q_2^s, we obtain

$$\ddot{e} + \alpha_1(q^s, \dot{q}^s)\dot{e} + \alpha_0(q^s, \dot{q}^s)e \approx \beta_2(q^s, \dot{q}^s)\left(\ddot{q}_2^d - \ddot{q}_2^s\right), \tag{4}$$

where the trajectory-varying coefficients α_0, α_1, and β_2 depend on the sampled trajectory $q_1^s(t)$, $q_2^s(t)$ and are thus time-varying. In the the linearized model (4), the tracking error e and the applied input q_2^s are known, and the goal is to iteratively learn the desired input q_2^d driving the tracking error to zero.

Discretization of (4) with a sampling time T_s yields

$$\begin{aligned} &\left(1 + a_{1,k}z^{-1} + a_{2,k}z^{-2}\right)e[k] \\ &\quad = \left(b_{0,k} + b_{1,k}z^{-1} + b_{2,k}z^{-2}\right)v[k], \end{aligned} \tag{5}$$

where z^{-1} is the time-shift operator, for a signal $x(t)$ the notation $x[k]$ stands for $x(kT_s)$, $v := q_2^d - q_2^s$, and the parameters $a_{i,k}$ for $i = 1, 2$ and $b_{i,k}$ for $i = 0, 1, 2$ are computed based on α_0, α_1, and β_2 values at the k-th sampled trajectory point $q^s[k]$, $\dot{q}^s[k]$.

[1] Here and below the argument of time t is omitted when clear form context.

[2] That implies existence of a position controller not discussed by Beuchert et al. (2018).

Next, the discretized model (5) is inverted and $v[k]$ is computed along the sampled trajectory based on the tracking error $e[k]$. After the j-th trial, the ILC update is performed as

$$q_2[k, j+1] = q_2[k, j] + \mathcal{F}\left[v[k]\right], \tag{6}$$

where $\mathcal{F}[\cdot]$ is a low-pass filtering operator and q_2 is the input signal in (3). Then the updated input signal q_2 is applied to the system, the new sampled trajectory is measured, and the deviation v is computed. These steps are repeated until the desired accuracy is obtained.

Shortcomings. The discussed solution is applied to a simplified 1-DOF system (3), where the linearization (4) and the discretization (5) are well-defined. We will show in the next sections that for the 2-DOF Furuta Pendulum system written in the form (1) where we choose the torque τ as the input (and thus no extra position controller is required), the linearization is not observable at certain trajectory segments. Thus, the minimum realization of the linearized system is of variable trajectory-dependent order, and the ILC via the transfer function inversion can be complicated due to zero-poles cancellation. In the following section, we propose another approach for the linearization-based ILC using the observability matrix pseudo-inversion.

4. PROPOSED SOLUTION FOR FURUTA PENDULUM

4.1 System description

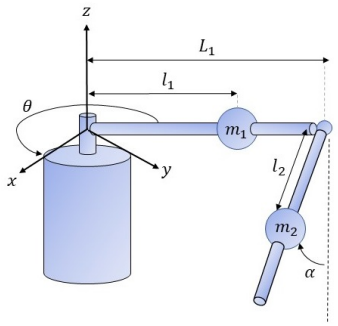
Furuta Pendulum is a mechanism which consists of a driven arm that rotates in the horizontal plane and a pendulum attached to the arm which is free to rotate in the vertical plane, as defined in Cazzolato and Prime (2011). The schematic of the mechanical system is depicted in Fig. 1a, and a photo of the Qunaser Cube equipment used in the paper is given in Fig. 1b. Following the notation shown in the schematic, denote the vertical angle of Arm 2 with respect to z-axis as α, where $\alpha = 0$ is the downward position, and the horizontal angle of Arm 1 with respect to the x-axis as θ. Defining $q := [\theta \ \alpha]^\top$, the dynamics of Furuta Pendulum can be written in the form (1) with

$$\begin{aligned}
\mathcal{M}(q) &:= \begin{bmatrix} p_1 + p_2 \sin^2(\alpha) + p_3 \cos^2(\alpha) & p_4 \cos(\alpha) \\ p_4 \cos(\alpha) & p_2 \end{bmatrix}, \\
\mathcal{C}(q, \dot{q}) &:= \begin{bmatrix} -p_4 \sin(\alpha)\dot{\alpha}^2 + \dot{\theta}\dot{\alpha} \sin(2\alpha)\,(p_2 - p_3) \\ \frac{1}{2}\dot{\theta}^2 \sin(2\alpha)\,(p_3 - p_2) \end{bmatrix}, \\
\mathcal{G}(q) &:= \begin{bmatrix} 0 \\ p_5 \sin(\alpha) \end{bmatrix}, \ \tau := \begin{bmatrix} U \\ 0 \end{bmatrix}.
\end{aligned} \tag{7}$$

Here p_i, $i = 1, \ldots, 5$ are constants, and U is the DC motor torque. Since the DC motor is equipped with a fast-time-scale current/torque controller, we assume that the input signal U can be directly assigned. We provide equations for the coupled mechanical parameters p_i and the corresponding numerical values in Appendix. In this work, we are interested in learning a control making the (non-actuated) angle α follow a predefined feasible trajectory; thus, the output signal is chosen as $y := [0 \ 1]\, q$.

4.2 Linearization

Applying the input signal U^s to the system, we observe the sampled trajectory α^s, $\dot{\alpha}^s$, and $\dot{\theta}^s$. It is worth noting



(a) Schematic of Furuta Pendulum.

(b) Quanser Cube equipment.

Fig. 1. Schematic of Furuta Pendulum and a Quanser Cube image.

that the angle θ does not affect the system dynamics, see (7), and is not considered here. Then the following approximation holds along the sampled trajectory:

$$\begin{aligned}
\ddot{\theta} - \ddot{\theta}^s \approx\ & a_1(\alpha - \alpha^s) + a_2(\dot{\theta} - \dot{\theta}^s) \\
& + a_3(\dot{\alpha} - \dot{\alpha}^s) + a_4(U - U^s) \\
\ddot{\alpha} - \ddot{\alpha}^s \approx\ & b_1(\alpha - \alpha^s) + b_2(\dot{\theta} - \dot{\theta}^s) \\
& + b_3(\dot{\alpha} - \dot{\alpha}^s) + b_4(U - U^s),
\end{aligned}$$

where the coefficients a_i and b_i for $i = 1, \ldots, 4$ are computed based on the linearization of (1) and (7) along $\alpha^s(t)$, $\dot{\alpha}^s(t)$, and $\dot{\theta}^s(t)$ and are thus time-varying.

Let $\alpha_d \equiv y_d$ be a feasible desired trajectory for the vertical angle of the pendulum's arm, and θ_d, U_d be the corresponding unknown trajectories of θ and U. Define the deviation signals $e_\alpha := \alpha_d - \alpha^s$, $e_\theta := \theta_d - \theta^s$ and $v := U_d - U^s$. Define the state vector

$$x := [\dot{e}_\theta \ e_\alpha \ \dot{e}_\alpha]^\top.$$

Then the linearized model is given by

$$\begin{aligned}
\dot{x} &= \begin{bmatrix} a_2 & a_1 & a_3 \\ 0 & 0 & 1 \\ b_2 & b_1 & b_3 \end{bmatrix} x + \begin{bmatrix} a_4 \\ 0 \\ b_4 \end{bmatrix} v = Ax + Bv, \\
e_\alpha &= Cx,
\end{aligned} \tag{8}$$

where A and B are time-varying matrices, and $C := [0 \ 1 \ 0]$. The linearized system (8) is not observable for $b_2 = 0$, which occurs, for instance, when the angle α is zero, see Appendix B. Indeed, the observability matrix is given by

$$\mathcal{O} := \begin{bmatrix} C \\ CA \\ CA^2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ b_2 & b_1 & b_3 \end{bmatrix}.$$

Here $\operatorname{rank} \mathcal{O} = 2$ for $b_2 = 0$, and $\operatorname{rank} \mathcal{O} = 3$ otherwise. It implies that for some segments of the sampled trajectory the linearization (8) is not the minimum realization of the input-output relation between v and e_α.

Recall that the signals θ_d and U_d are unknown, i.e., the signals e_θ and v are not available, since the only given desired trajectory is α_d. To apply the ILC strategy, it is necessary to estimate the vector x based on the signal e_α, and then find v driving e_α to zero; thus the input U_d will be learned iteratively.

4.3 Discretization and Inversion

Let T_s be the sampling time and let A_d and B_d be the (time-varying) state and input matrices of the discretized linearized system (8), respectively. Then

$$\begin{aligned} x[k+1] &= A_d x[k] + B_d v[k] \\ e_\alpha[k] &= Cx[k]. \end{aligned} \tag{9}$$

Assuming that the linearization changes insignificantly between consecutive steps, i.e., that the sampling frequency is sufficiently high comparing to the trajectory variation, the following relations at step k hold

$$\begin{aligned} e_\alpha[k-3] &= Cx[k-3] \\ e_\alpha[k-2] &= CA_d x[k-3] + CB_d v[k-3] \\ e_\alpha[k-1] &= CA_d^2 x[k-3] + CA_d B_d v[k-3] + CB_d v[k-2] \end{aligned} \tag{10}$$

and

$$\begin{aligned} e_\alpha[k] = CA_d^3 x[k-3] + CA_d^2 B_d v[k-3] \\ + CA_d B_d v[k-2] + CB_d v[k-1]. \end{aligned} \tag{11}$$

Equation (10) can be rewritten as

$$\mathcal{O}_d x[k-3] = \begin{bmatrix} e_\alpha[k-3] \\ e_\alpha[k-2] - CB_d v[k-3] \\ e_\alpha[k-1] - CA_d B_d v[k-3] - CB_d v[k-2] \end{bmatrix}, \tag{12}$$

where $\mathcal{O}_d$ is the discrete-time observability matrix,

$$\mathcal{O}_d := \begin{bmatrix} C \\ CA_d \\ CA_d^2 \end{bmatrix}.$$

To handle both cases when the matrix $\mathcal{O}_d$ is of full rank or not, we propose to estimate the state $x[k-3]$ as

$$\hat{x}[k-3] = \mathcal{O}_d^\dagger \begin{bmatrix} e_\alpha[k-3] \\ e_\alpha[k-2] - CB_d v[k-3] \\ e_\alpha[k-1] - CA_d B_d v[k-3] - CB_d v[k-2] \end{bmatrix}, \tag{13}$$

where $(\cdot)^\dagger$ is the Moore-Penrose pseudo-inverse, and $\hat{x}$ is the estimate of the state vector x. When the observability matrix is of full rank, the unique solution $\hat{x}[k-3] = x[k-3]$ is computed. When the system is not observable, we cannot reconstruct the exact state $x[k-3]$, and we apply the pseudo-inverse approach computing the minimum-norm $\hat{x}[k-3]$ estimate. This approach allows us to uniformly treat all the trajectory points, disregard the possible observability loss.

Given the estimate $\hat{x}[k-3]$, we compute the input signal update for the next trial following (11) as

$$\begin{aligned} v[k-1] = (CB_d)^{-1} \big(e_\alpha[k] - CA_d^3 \hat{x}[k-3] \\ - CA_d^2 B_d v[k-3] - CA_d B_d v[k-2] \big) \end{aligned} \tag{14}$$

and

$$U[k, j+1] = U^s[k, j] + \mathcal{F}\left[v[k]\right],$$

where $U[\cdot, j+1]$ is the input signal to be applied at the next trial, and $\mathcal{F}[\cdot]$ is a low-pass filter operator, cf. (6). The learning is repeated until the desired tracking accuracy ϵ is achieved,

$$|e_\alpha[k]| < \epsilon$$

for all k in the trial, or the maximum trials number N_{max} is performed.

4.4 Discussion

Singularity points. The control update (14) is not feasible when $CB_d = 0$. The trajectory points where $CB_d = 0$ are the singular points, see Beuchert et al. (2018). At these points, the output value $e_\alpha[k]$ does not depend on the previous input $v[k-1]$. The zero division can be avoided, e.g., replacing small values of CB_d with a predefined constant.

Filtering. The low-pass filter $\mathcal{F}$ is introduced to avoid possible instabilities caused by numerical operations. Even though the filter helps smooth the updating signal, it also restricts which trajectories the ILC can learn. Suppose the desired output α_d requires a high oscillating or a discontinuous input U_d. In that case, the ILC will not learn the desired input since the low-pass filter cuts off the signal's high-frequency components. Thus, the filter $\mathcal{F}$ restricts the class of feasible trajectories.

Initial values. As one can see from the update law (13), (14), the computation of $\hat{x}[k-3]$ and $v[k-1]$ depends on the measured deviations $e_\alpha[k-i]$ for $i = 0, 1, 2, 3$ and the previous computation results $v[k-i]$, $i = 2, 3$. Obviously, these equations cannot be used for $k \leq 3$. To this end, the following initialization steps are proposed assuming zero initial conditions for x:

$$\begin{aligned} x[1] &= 0, \\ x[2] &= B_d v[1], \\ x[3] &= A_d x[2] + B_d v[2], \\ x[4] &= A_d x[3] + B_d v[3], \end{aligned}$$

and

$$\begin{aligned} v[1] &= (CB_d)^{-1} e_\alpha[2], \\ v[2] &= (CB_d)^{-1} \left(e_\alpha[3] - CA_d B_d v[1] \right), \\ v[3] &= (CB_d)^{-1} \left(e_\alpha[4] - CA_d B_d v[2] - CA_d^2 B_d v[1] \right). \end{aligned} \tag{15}$$

Learning from small error only. The linearized system is an approximation of the nonlinear model. Therefore, when the tracking error becomes large, the linearization (8) is not a good approximation of (1), and the learning process becomes inaccurate. Following the ideas of Beuchert et al. (2018), in this study the learning is performed only for small tracking errors. Specifically, let $e_{max} > 0$ be a predefined value and let k_e be the first time instance where the absolute value of e_α overpasses e_{max},

$$|e_\alpha[k_e]| \geq e_{max}.$$

Then for this trial, for all $k \geq k_e$ the learning is stopped and instead of (14), the signal v is further updated as

$$v[k-1] = v[k-2].$$

4.5 The ILC algorithm

The ILC realization for Furuta Pendulum is summarized in Algorithm 1.

5. EXAMPLE

This section presents the simulation results of the proposed ILC method for the Furuta Pendulum system. All pendulum parameters are summarized in Appendix A. To illustrate robustness to possible inaccuracies in model parameters, the values used by the ILC algorithm are not the same as the ones used for the pendulum simulation; see Table A.1 in Appendix A. The pendulum operates in open-loop mode, and the control signal is applied as a

Algorithm 1: ILC algorithm for Furuta Pendulum

Result: signal U_{ILC} that tracks α_d
Load parameters p_1 to p_5 (pendulum parameters)
Load desired trajectory α_d
Set the pendulum's initial state q_0, $\dot{q}_0$ and the ILC algorithm parameters
while $max_t|e_\alpha(t)| > \epsilon$ *and* $N_{trial} < trials_{max}$ **do**
  Run simulation
  Collect sampled trajectory (α^s, $\dot{\alpha}^s$, $\dot{\theta}^s$)
  Compute $e_\alpha(t)$
  Linearize around the sampled trajectory
  Discretize the system
  for *the first 4 sampled points* **do**
    Compute v as in (15)
  end
  for *all other sampled points* **do**
    Compute v as in (14)
  end
  Update U_{ILC}
end

Table 1. The ILC algorithm parameters.

Description	Symbol	Value
Initial pendulum position, rad	q_0	$[0\ 0]^\top$
Initial pendulum velocity, rad/s	$\dot{q}_0$	$[0\ 0]^\top$
Sampling time, s	T_s	0.02
Small-error bound, rad	e_{max}	0.3
Filter $\mathcal{F}$ normalized cutoff frequency, rad/sample	–	0.7
Desired accuracy, rad	ϵ	0.001
Maximum trials number	N_{max}	100

feedforward input. According to the proposed ILC strategy, the control is updated between the trials based on the sampled trajectories.

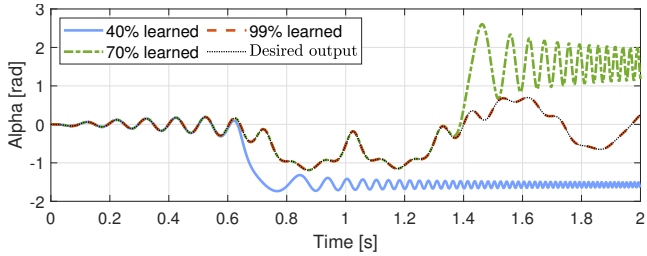
We apply the ILC algorithm as described in Section 4.5; the algorithm implements equations (13), (14) subject to the discussions given in Section 4.4. The algorithm parameters are summarized in Table 1, where the low-pass filter $\mathcal{F}$ is the first order Butterworth filter.

In order to check the capability of the algorithm to track different trajectory profiles, 2 desired outputs are tested with the algorithm. Although some trajectories take more iterations to converge than others, all of them are learned in a finite number of iterations, respecting the constraint given by e_{max}.

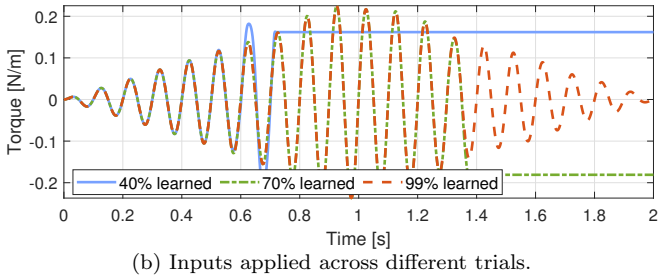
Figures 2 and 3 present the evolution of the output of the algorithm across different trials for different desired trajectories fed to the system.

6. CONCLUSION

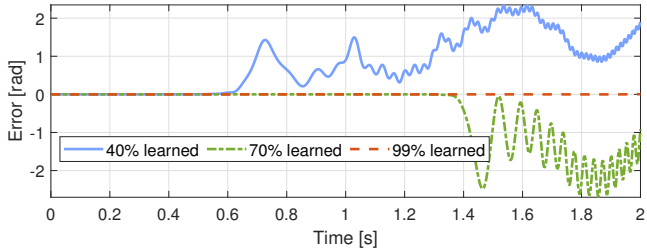
We considered the Iterative Learning Control design for the Furuta Pendulum system. Motivated by the research of Beuchert et al. (2018), the ILC strategy is based on the linearization of the system dynamics along the sampled trajectory. However, for the Furuta Pendulum system, the linearization is not observable at some points, and thus the minimum realization order is trajectory-dependent. To deal with that, we propose a novel method based on the observability matrix inversion that allows uniform processing of all trajectory points. The applicability of the proposed method is illustrated with the Furuta Pendulum simulations. Cases where the desired trajectory crosses singularity points might be an interesting point of attention for future research activities.



(a) Actual and desired output of the system across different trials. The desired and the 99%-learned trajectories are indistinguishable.



(b) Inputs applied across different trials.



(c) Error signal across different trials.

Fig. 2. Simulation results for desired trajectory #1.

ACKNOWLEDGEMENTS

This work was supported by Rennes Metropole, France (project AIS-19C0326).

REFERENCES

Arimoto, S., Kawamura, S., and Miyazaki, F. (1984). Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronics systems. In *The 23rd IEEE Conference on Decision and Control*, 1064–1069. IEEE.

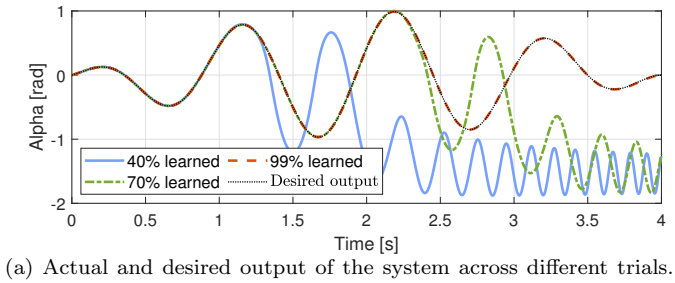
Åström, K.J. and Wittenmark, B. (2013). *Adaptive control*. Courier Corporation.

Beuchert, J., Raischl, J.r., and Seel, T. (2018). Design of an iterative learning control with a selective learning strategy for swinging up a pendulum. In *2018 European Control Conference (ECC)*, 3137–3142. IEEE.

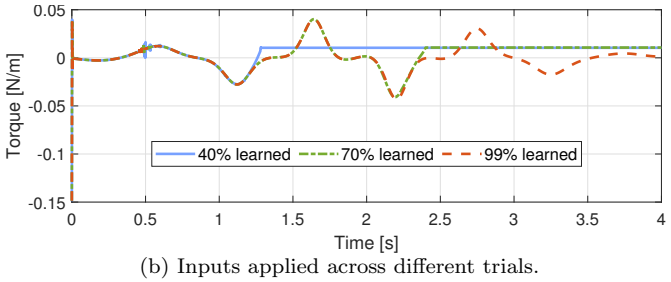
Bristow, D.A., Tharayil, M., and Alleyne, A.G. (2006). A survey of iterative learning control. *IEEE control systems magazine*, 26(3), 96–114.

Cazzolato, B.S. and Prime, Z. (2011). On the dynamics of the furuta pendulum. *Journal of Control Science and Engineering*, 2011.

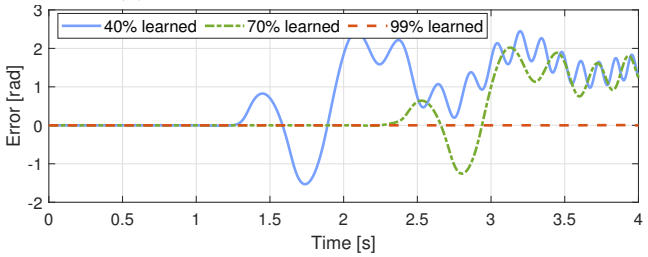
Gao, H., Lu, Y., Mai, Q., and Hu, Y. (2009). Inverted pendulum system control by using modified iterative



(a) Actual and desired output of the system across different trials. The desired and the 99%-learned trajectories are indistinguishable.



(b) Inputs applied across different trials.



(c) Error signal across different trials.

Fig. 3. Simulation results for desired trajectory #2.

learning control. In *International Conference on Intelligent Robotics and Applications*, 1230–1236. Springer.

Hoelzle, D.J., Alleyne, A.G., and Johnson, A.J.W. (2010). Basis task approach to iterative learning control with applications to micro-robotic deposition. *IEEE Transactions on Control Systems Technology*, 19(5), 1138–1148.

Huang, D., Xu, J.X., Venkataramanan, V., and Huynh, T.C.T. (2013). High-performance tracking of piezoelectric positioning stage using current-cycle iterative learning control with gain scheduling. *IEEE Transactions on Industrial Electronics*, 61(2), 1085–1098.

Jin, X. (2018a). Fault-tolerant iterative learning control for mobile robots non-repetitive trajectory tracking with output constraints. *Automatica*, 94, 63–71.

Jin, X. (2018b). Iterative learning control for output-constrained nonlinear systems with input quantization and actuator faults. *International Journal of Robust and Nonlinear Control*, 28(2), 729–741.

Lee, J.H. and Lee, K.S. (2007). Iterative learning control applied to batch processes: An overview. *Control Engineering Practice*, 15(10), 1306–1318.

Lu, J., Cao, Z., Zhang, R., and Gao, F. (2017). Nonlinear monotonically convergent iterative learning control for batch processes. *IEEE Transactions on Industrial Electronics*, 65(7), 5826–5836.

Norrlof, M. (2002). An adaptive iterative learning control algorithm with experiments on an industrial robot. *IEEE Transactions on robotics and automation*, 18(2), 245–251.

Table A.1. Used parameter notation and values.

Description	Symbol	Pendulum	ILC algorithm
Mass of the Arm 1, kg	m_1	0.092	0.080
Mass of the Arm 2, kg	m_2	0.024	0.025
Pivot-to-center (of mass) distance, Arm 1, m	l_1	0.016	0.02
Pivot-to-pivot distance, Arm 1, m	L_1	0.085	0.09
Pivot-to-center (of mass) distance, Arm 2, m	l_2	0.058	0.065
Moments of inertia, kg·m^2:			
(minor) axes y and z, Arm 1	J_{1yz}	$6.04 \cdot 10^{-5}$	$6 \cdot 10^{-5}$
(major) axis x, Arm 2	J_{2x}	$3.1 \cdot 10^{-7}$	$3.3 \cdot 10^{-7}$
(minor) axes y and z, Arm 2	J_{2yz}	$3.3 \cdot 10^{-5}$	$3.1 \cdot 10^{-7}$
Gravity acceleration, m·s^{-2}	g	9.81	9.81

Sastry, S. and Bodson, M. (2011). *Adaptive control: stability, convergence and robustness*. Courier Corporation.

Schöllig, A. and D'Andrea, R. (2009). Optimization-based iterative learning control for trajectory tracking. In *2009 European Control Conference (ECC)*, 1505–1510. IEEE.

Uchiyama, M. (1978). Formation of high-speed motion pattern of a mechanical arm by trial. *Transactions of the Society of Instrument and Control Engineers*, 14(6), 706–712.

Wang, Y., Gao, F., and Doyle III, F.J. (2009). Survey on iterative learning control, repetitive control, and run-to-run control. *Journal of Process Control*, 19(10), 1589–1600.

Wu, L., Yan, Q., and Cai, J. (2019). Neural network-based adaptive learning control for robot manipulators with arbitrary initial errors. *IEEE Access*, 7, 180194–180204.

Xu, J.X. (2011). A survey on iterative learning control for nonlinear systems. *International Journal of Control*, 84(7), 1275–1294.

Zhang, X., Wu, J., and Zhan, X. (2008). Iterative learning control study on single inverted pendulum. In *2008 International Symposium on Intelligent Information Technology Application Workshops*, 155–158. IEEE.

Zhang, Y., Chu, B., and Shu, Z. (2019). A preliminary study on the relationship between iterative learning control and reinforcement learning. *IFAC-PapersOnLine*, 52(29), 314–319.

Appendix A. COUPLED MECHANICAL PARAMETERS AND NUMERICAL VALUES

The coupled mechanical parameters p_i, $i = 1, \ldots, 5$ that are introduced in (7), are defined as

$$p_1 := J_{1yz} + m_1 l_1^2 + m_2 L_1^2, \quad p_2 := J_{2yz} + m_2 l_2^2,$$
$$p_3 := J_{2x}, \quad p_4 := m_2 L_1 l_2, \quad p_5 := g m_2 l_2,$$

where J_{1yz}, J_{2yz}, J_{2x}, m_1, m_2, l_1, l_2, L_1, and g are physical parameters explained in Table A.1; the corresponding numerical values are also given therein.

Appendix B. LINEARIZATION COEFFICIENT

The linearization coefficient b_2, see (8), is defined as

$$b_2 := \frac{\cos(\alpha)\sin(\alpha)(2p_2 - 2p_3)\kappa(\alpha, \dot{\theta})}{p_2^2 \sin^2(\alpha) + p_3 p_3 \cos(\alpha)^2 + p_1 p_2 - p_4^2 \cos^2(\alpha)},$$

where

$$\kappa(\alpha, \dot{\theta}) := \left(p_1 + p_2 - (p_2 - p_3)\cos^2(\alpha)\right)\dot{\theta} + p_4 \dot{\alpha} \cos(\alpha).$$