53rd IEEE Conference on Decision and Control
December 15-17, 2014. Los Angeles, California, USA

Design of Norm-optimal Iterative Learning Controllers: The Effect of An Iteration-Domain Kalman Filter For Disturbance Estimation

Nicolas Degen and Angela P. Schoellig

***Abstract*— Iterative learning control (ILC) has proven to be an effective method for improving the performance of repetitive control tasks. This paper revisits two optimization-based ILC algorithms: *(i)* the widely used quadratic-criterion ILC law (QILC) and *(ii)* an estimation-based ILC law using an iteration-domain Kalman filter (K-ILC). The goal of this paper is to analytically compare both algorithms and to highlight the advantages of the Kalman-filter-enhanced algorithm. We first show that for an *iteration-constant* estimation gain and an appropriate choice of learning parameters both algorithms are identical. We then show that the estimation-enhanced algorithm with its *iteration-varying optimal* Kalman gains can achieve both fast initial convergence and good noise rejection by (optimally) adapting the learning update rule over the course of an experiment. We conclude that the clear separation of disturbance estimation and input update of the K-ILC algorithm provides an intuitive architecture to design learning schemes that achieve both low noise sensitivity and fast convergence. To benchmark the algorithms we use a simulation of a single-input, single-output mass-spring-damper system.**

I. Introduction

Iterative learning control (ILC) algorithms have been introduced as a mean to reduce the tracking error of systems that repeatedly execute the same task. In contrast to adaptive control schemes, which use 'learned' information to adapt an underlying feedback controller, ILC algorithms adapt the feed-forward input to the system (which may be the reference signal to a feedback-controlled system) as a function of past tracking errors. Since the initial work of Arimoto *et al.* in 1984 [1], many variations of the algorithm have been published, and ILC has proven to be effective in a variety of application areas ranging from robotic manipulators [2]–[5] and chemical reactors [6], [7] to quadrocopter maneuvering [8]–[10]. A good overview over the field of ILC is given in the survey papers [7], [11], [12].

More recently, starting with the work by Buchheit *et al.* in 1994 [13], researchers have begun to formulate the ILC input update step as an optimization problem minimizing the predicted tracking error of the next iteration. Some recent examples of optimization-based (also called 'norm-optimal') ILC approaches are [3], [7], [8], [14]. The optimization formulation enables the integration of input and output constraints as shown in [7], [8], [15].

In the recent years standard ILC algorithms have been extended with Kalman filter estimators to improve the learning performance, especially for non-repetitive noise. In [8]–[10], a Kalman-filter-enhanced ILC scheme was presented that is used to improve the prediction of the next iteration's tracking error. In contrast to time-domain Kalman-based estimation, which has been proven to not be beneficial in combination with ILC [16], the approach in [8]–[10] applies the Kalman filter to iteration-domain variables. Similar iteration-domain estimation-based ILC schemes have also been proposed by Norrlöff *et al.* in [5], [17]–[19].

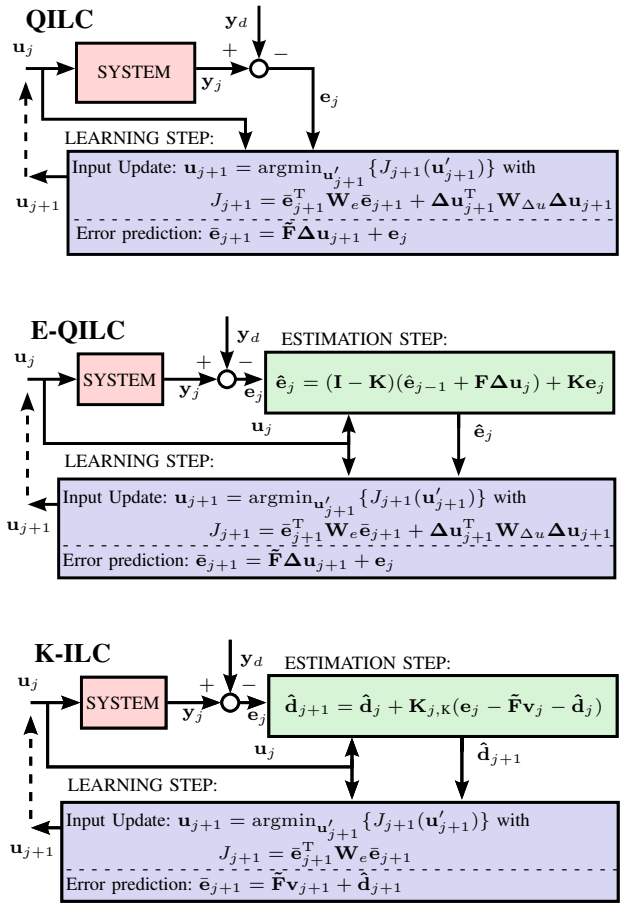

Fig. 1. Typical learning algorithm setups. **Top:** The widely-used quadratic optimal ILC (QILC) [11]. **Middle:** ILC algorithm with integrated estimation step (E-QILC) as presented by Lee *et al.* [7]. **Bottom:** Kalman-filter-based algorithm (K-ILC) presented by Schoellig *et al.* [8]. The dashed lines represent the input for the next iteration.

The goal of this paper is to analytically compare three optimization-based algorithms (see Figure 1): *(i)* the common quadratic-criterion ILC (QILC) [11], [14], *(ii)* an estimation-enhanced quadratic-criterion ILC scheme (E-QILC) [7], and *(iii)* the Kalman-filter based optimal ILC algorithm (K-ILC)

Nicolas Degen is with the Autonomous Systems Lab, ETH Zurich, Switzerland. nidegen@ethz.ch

Angela P. Schoellig is with the University of Toronto Institute for Aerospace Studies, Canada. schoellig@utias.utoronto.ca

as introduced in [8]. The main difference between these algorithms is the method used *to predict the next iteration's tracking error*, which is in turn used in the minimization problem of the input update step. In *(ii)* and *(iii)* (E-QILC and K-ILC, respectively), an estimation algorithm is used for the error prediction, taking into account measurements of all past iterations. While *(ii)* uses an estimation gain that is constant for all iterations, the algorithm in *(iii)* uses a Kalman filter, in which the estimation gain is changing over iterations in an optimal way (based on the assumed noise characteristics). We aim to understand the influence of using an estimation algorithm on the overall performance of the learning algorithm.

In the following, we describe algorithms *(i)-(iii)* in two steps, an error prediction step (A) and an input update step (B) (Section II). We then compare the algorithms analytically (Section III) and finally highlight the found results in a simulation example (Section IV).

II. ILC Algorithms

We consider three different optimization-based iterative learning approaches: *(i)* quadratic optimal ILC (QILC), *(ii)* estimation-enhanced quadratic optimal ILC scheme (E-QILC), and *(iii)* the Kalman-filter based optimal ILC algorithm (K-ILC). We characterize these optimization-based iterative learning schemes by the methods used for (A) the error prediction and (B) the input update. The error prediction step predicts the next iteration's tracking error (see Figure 1) as a function of the new input $\bar{\mathbf{e}}_{j+1}(\mathbf{u}_{j+1})$ taking into account all past measured tracking errors and inputs $\mathbf{e}_j, .., \mathbf{e}_0, \mathbf{u}_j, .., \mathbf{u}_0$. The input update step then computes the input for the next iteration as an optimal solution of a cost function that uses the previously stated error prediction.

In this section, we first introduce the nominal system model and then state the error prediction (Section II-B) and input update algorithms (Section II-C) used in *(i)-(iii)*.

A. Nominal System Model

We assume that the nominal behavior of the system (cf. Figure 1) is modelled by the following input-output relationship

$$\tilde{\mathbf{y}} \quad = \tilde{\mathbf{F}}\mathbf{u}, \tag{1}$$

which maps a discrete-time input signal $\mathbf{u} = [u[1]^{\mathrm{T}}, ..., u[N]^{\mathrm{T}}]^{\mathrm{T}}$ to the corresponding lifted output $\mathbf{y} = [y[m]^{\mathrm{T}}, ..., y[m+N]^{\mathrm{T}}]^{\mathrm{T}}$ via a static, nominal matrix $\tilde{\mathbf{F}}$. In this context, $(N+1)$ samples represent the values at all times for a single iteration j and m the relative degree of the system [10], [11], [20]. We use $\tilde{(\cdot)}$ to denote signals calculated via the nominal model $\tilde{\mathbf{F}}$. In practice, (1) may be derived from a first principles model of the system using linearization and discretization [10], [11], [20].

The goal of ILC is to track a desired output trajectory $\mathbf{y}_d$. We introduce $\mathbf{u}_{\text{nom}}$ as the input that results in the desired output calculated with the nominal model, $\mathbf{y}_d = \tilde{\mathbf{F}}\mathbf{u}_{\text{nom}}$. However, when applying $\mathbf{u}_{\text{nom}}$ to the real system, we typically measure a tracking error $\mathbf{e} = \mathbf{y} - \mathbf{y}_d$, where $\mathbf{y}$ and $\mathbf{e}$ denote measured values. The tracking error is usually a result of unmodeled dynamics, unmodeled external disturbances and noise. The learning algorithm iteratively adapts the input $\mathbf{u}_j$ to decrease the tracking error $\mathbf{e}_j$, where the index j indicates the input and error of iteration j, $j = 0, 1, 2, \ldots$. The nominal, i.e. modelled, error dynamics are given by

$$\tilde{\mathbf{e}}_j = \quad \tilde{\mathbf{y}}_j - \mathbf{y}_d = \tilde{\mathbf{F}}(\mathbf{u}_j - \mathbf{u}_{\text{nom}}) = \tilde{\mathbf{F}}\mathbf{v}_j, \tag{2}$$

where $\mathbf{e}_j = \mathbf{y}_j - \mathbf{y}_d$ represents the error between the measured and the desired output[1] and $\mathbf{v}_j$ the deviation of the input $\mathbf{u}_j$ from the nominal input $\mathbf{u}_{\text{nom}}$. Typically, we choose $\mathbf{u}_0 = \mathbf{u}_{\text{nom}}$ as initial input.

B. Error Prediction

1) QILC Error Prediction: In general, the predicted error of the next iteration, $\bar{\mathbf{e}}_{j+1}$, is a function of the new input $\mathbf{u}_{j+1}$ and all previous iterations' errors $(\mathbf{e}_0, ..., \mathbf{e}_j)$, and is indicated by a bar $\bar{(\cdot)}$. The QILC algorithm uses the most straightforward method to predict the error based on (2),

$$\tilde{\mathbf{e}}_{j+1} - \tilde{\mathbf{e}}_j = \tilde{\mathbf{F}}(\mathbf{v}_{j+1} - \mathbf{v}_j) = \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{u}_{j+1}, \tag{3}$$

where $\boldsymbol{\Delta}\mathbf{u}_{j+1} := \mathbf{u}_{j+1} - \mathbf{u}_j = \mathbf{v}_{j+1} - \mathbf{v}_j$. The model-predicted change of the error (3) added to the measured error, $\mathbf{e}_j$, then results in

$${}^{\text{QILC}}\bar{\mathbf{e}}_{j+1} = \quad \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{u}_{j+1} + \mathbf{e}_j. \tag{4}$$

2) E-QILC Error Prediction: E-QILC extends the prediction procedure of QILC by adding an estimation step [7]. The QILC prediction as given in (4) is used, substituting the measured error $\mathbf{e}_j$ with an estimated error $\hat{\mathbf{e}}_j$:

$${}^{\text{E-QILC}}\bar{\mathbf{e}}_{j+1} = \quad \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{u}_{j+1} + \hat{\mathbf{e}}_j. \tag{5}$$

The estimated error is obtained from

$$\hat{\mathbf{e}}_j = (\mathbf{I} - \mathbf{K})(\underbrace{\hat{\mathbf{e}}_{j-1} + \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{u}_j}_{\bar{\mathbf{e}}_j}) + \mathbf{K}\mathbf{e}_j, \tag{6}$$

where an estimation gain $\mathbf{K}$ weighs the influence of the measured error and the previous prediction. In [7], an iteration-constant gain, $\mathbf{K} = \lambda\mathbf{I}$, $\lambda \in \mathbb{R}$ and $\mathbf{I}$ representing the identity matrix of matching dimension.

3) K-ILC Error Prediction: The K-ILC algorithm as proposed in [8] estimates a disturbance vector $\mathbf{d}_{j+1}$, which represents the difference between nominal model and the real system:

$${}^{\text{K-ILC}}\bar{\mathbf{e}}_{j+1} = \quad \tilde{\mathbf{F}}\mathbf{v}_{j+1} + \hat{\mathbf{d}}_{j+1}, \tag{7}$$

where $\tilde{\mathbf{F}}\mathbf{v}_{j+1}$ is the error according to the nominal model, cf. (2).

The disturbance estimate is obtained from a Kalman filter based on the following model:

$$\begin{aligned} \mathbf{d}_{j+1} &= \mathbf{d}_j + \omega_j \\ \mathbf{e}_j &= \tilde{\mathbf{F}}\mathbf{v}_j + \mathbf{d}_j + \mu_j, \end{aligned} \tag{8}$$

with $\omega_j \sim \mathcal{N}(0, \mathbf{E}_j)$ and $\mu_j \sim \mathcal{N}(0, \mathbf{H}_j)$. The covariances $\mathbf{E}_j$ and $\mathbf{H}_j$ may be seen as design parameters to adapt the

[1] In the literature, the error is often defined as $\mathbf{e}'_j = \mathbf{y}_d - \mathbf{y}_j$ in contrast to $\mathbf{e}_j = \mathbf{y}_j - \mathbf{y}_d$ as used here.

learning rate of the algorithm. A common choice are diagonal covariances; that is, $\mathbf{H}_j = \eta\mathbf{I}, \mathbf{E}_j = \epsilon\mathbf{I}$, where $\eta, \epsilon \in \mathbb{R}$ and $\mathbf{I}$ represents an identity matrix of appropriate size.

The Kalman filter [8], [21] provides the optimal disturbance estimate $\hat{\mathbf{d}}_{j+1}$ and error estimate $\hat{\mathbf{e}}_{j+1}$ based on the stochastic model (8) taking into account all past measured tracking errors:

$$\hat{\mathbf{d}}_{j+1} = \hat{\mathbf{d}}_j + \mathbf{K}_{j,\mathrm{K}}(\mathbf{e}_j - \tilde{\mathbf{F}}\mathbf{v}_j - \hat{\mathbf{d}}_j) \tag{9}$$

$$\hat{\mathbf{e}}_{j+1} = \tilde{\mathbf{F}}\mathbf{v}_{j+1} + \hat{\mathbf{d}}_{j+1}. \tag{10}$$

The optimal *iteration-varying* Kalman gain $\mathbf{K}_{j,\mathrm{K}}$ is calculated recursively:

$$\begin{aligned} \mathbf{S}_j &= \mathbf{P}_j + \mathbf{E}_j \\ \mathbf{K}_{j,\mathrm{K}} &= \mathbf{S}_j(\mathbf{S}_j + \mathbf{H}_{j+1})^{-1} \\ \mathbf{P}_{j+1} &= (\mathbf{I} - \mathbf{K}_{j,\mathrm{K}})\mathbf{S}_j. \end{aligned} \tag{11}$$

$\mathbf{P}_0 = E[(\mathbf{d}_0 - \hat{\mathbf{d}}_0)(\mathbf{d}_0 - \hat{\mathbf{d}}_0)^{\mathrm{T}}]$ is the covariance matrix of the expected initial disturbance and can be viewed as a design parameter together with the covariances $\mathbf{H}_j, \mathbf{E}_j$. The initial disturbance $\hat{\mathbf{d}}_0$ is typically chosen to be zero as the nominal model is ideally the best possible guess.

Remark 1. In the derivations below, we use the fact that for any arbitrary sequence of gains $\mathbf{K}_j$, there exist corresponding matrices $\mathbf{H}_j, \mathbf{E}_j$, and $\mathbf{P}_0$ such that (11) is satisfied and $\mathbf{K}_j$ can be interpreted as optimal Kalman gains $\mathbf{K}_{j,\mathrm{K}}$ of a specific stochastic model.

Remark 2. In contrast to K-ILC, E-QILC has not been presented with a Kalman-filter-type estimation scheme. In practice, however, the gain $\mathbf{K}_j$ can always be chosen such that the Kalman equations are satisfied. This fact will be used when comparing both approaches in Sec. III.

C. Input Update

Given the predicted error $\bar{\mathbf{e}}_{j+1}$, the optimization-based input update can be stated in its most general form as the minimization of the following cost function:

$$\begin{aligned} J_{j+1}(\mathbf{u}_{j+1}) = \; & \|\mathbf{W}'_e\bar{\mathbf{e}}_{j+1}\|_p^a + \|\mathbf{W}'_{\Delta u}\boldsymbol{\Delta}\mathbf{u}_{j+1}\|_p^a \\ & + \|\mathbf{W}'_u\mathbf{u}_{j+1}\|_p^a + \|\mathbf{W}'_v\mathbf{v}_{j+1}\|_p^a \end{aligned} \tag{12}$$

using p-vector norms to the power of a. The next iteration's input is then given by

$$\mathbf{u}_{j+1} = \underset{\mathbf{u}'_{j+1}}{\operatorname{argmin}}\{J_{j+1}(\mathbf{u}'_{j+1})\}. \tag{13}$$

The cost function penalizes the predicted tracking error $\bar{\mathbf{e}}_{j+1}$, and optionally the change of input $\boldsymbol{\Delta}\mathbf{u}_{j+1}$ from one iteration to the next, the magnitude of the 'shifted' input $\mathbf{v}_j$ as well as the input $\mathbf{u}_{j+1}$ with corresponding weightings. Most literature dealing with ILC considers the squared 2-norm ($p = a = 2$). In principle, as the optimization can be done numerically, any norm is possible. Also, the optimization step allows the implementation of 'hard' constraints on the input and output signals [7], [10], [15]. For our analytical study, we will focus on *unconstrained cost functions using the 2-norm* ($p = a = 2$), since in this case an explicit solution to the optimization problem can be found. In addition, the penalization of the inputs $\mathbf{u}_{j+1}$ and $\mathbf{v}_{j+1}$ will be omitted as it represents 'soft' input constraints. Such 'soft' constraints will act in minimizing $\mathbf{u}_\infty$ and $\mathbf{v}_\infty$ for the converged solution. It further makes an deterministic analysis impossible, without fundamentally affecting the core of the algorithm, it's noise robustness and convergence properties. Overall, this leaves us with

$$J_{j+1} = \; \bar{\mathbf{e}}_{j+1}^{\mathrm{T}}\mathbf{W}_e\bar{\mathbf{e}}_{j+1} + \boldsymbol{\Delta}\mathbf{u}_{j+1}^{\mathrm{T}}\mathbf{W}_{\Delta u}\boldsymbol{\Delta}\mathbf{u}_{j+1}, \tag{14}$$

where $\mathbf{W}_\ell = \mathbf{W}_\ell'^{\mathrm{T}}\mathbf{W}'_\ell, \ell \in \{e, \Delta u\}$, is positive definite and $\mathbf{W}_\ell = \mathbf{W}_\ell^{\mathrm{T}}$. The weighting matrices $\mathbf{W}_e$ and $\mathbf{W}_{\Delta u}$ are design parameters that are chosen to reflect the learning objectives.

1) QILC Input Update: We obtain the updated input $\mathbf{u}_{j+1}$ of the QILC algorithm by using ${}^{\text{QILC}}\bar{\mathbf{e}}_{j+1}$ from (4) for the cost function (14). As the cost function is quadratic with positive definite weighting matrices, a minimizing input $\mathbf{u}_{j+1}$ is found by calculating $\frac{dJ_{j+1}}{d\mathbf{u}_{j+1}} = 0$. We obtain:

$${}^{\text{QILC}}\mathbf{u}_{j+1} = \mathbf{u}_j - L\mathbf{e}_j \tag{15}$$

with

$$L = \; (\tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e\tilde{\mathbf{F}} + \mathbf{W}_{\Delta u})^{-1}\tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e. \tag{16}$$

The derivation of (16) can be found in [22]. A non-recursive expression for ${}^{\text{QILC}}\mathbf{u}_{j+1}$ is:

$${}^{\text{QILC}}\mathbf{u}_{j+1} = \mathbf{u}_{\text{nom}} - \textstyle\sum_{i=0}^{j} L\mathbf{e}_i. \tag{17}$$

A special case of QILC is model inversion: for a cost function,

$$J_{j+1} = \bar{\mathbf{e}}_{j+1}^{\mathrm{T}}\mathbf{W}_e\bar{\mathbf{e}}_{j+1}, \tag{18}$$

we get $L = -\tilde{\mathbf{F}}^{-1}$; that is, a straight-forward model inversion:

$$\mathbf{u}_{j+1} = \mathbf{u}_j - \tilde{\mathbf{F}}^{-1}\mathbf{e}_j. \tag{19}$$

2) E-QILC Input Update: As E-QILC for the error prediction step, the input update is analogous to QILC with only the measured error substituted by the estimated error:

$${}^{\text{E-QILC}}\mathbf{u}_{j+1} = \mathbf{u}_j + L\hat{\mathbf{e}}_j \tag{20}$$

3) K-ILC Input Update: For K-ILC, the explicit solution for the optimal update is

$$\begin{aligned} \mathbf{v}_{j+1} = \; & (\tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e\tilde{\mathbf{F}} + \mathbf{W}_{\Delta u})^{-1} \\ & (-\tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e^{\mathrm{T}}\hat{\mathbf{d}}_{j+1} + \mathbf{W}_{\Delta u}^{\mathrm{T}}\mathbf{v}_j) \end{aligned} \tag{21}$$

and is comparable to Norrlöf's result in [5]. The derivation of (21) can be found in [22].

For further analysis, we consider the case for which we choose $\mathbf{W}_{\Delta u} = 0$, only penalizing the tracking error. This represents the scheme as proposed by Schoellig *et al.* [8] without the 'soft' constraints. For this case, (21) is

$$\mathbf{v}_{j+1} = -\tilde{\mathbf{F}}^{-1}\hat{\mathbf{d}}_{j+1}. \tag{22}$$

For the jth iteration this can be transformed to $\tilde{\mathbf{F}}\mathbf{v}_j = -\tilde{\mathbf{F}}\tilde{\mathbf{F}}^{-1}\hat{\mathbf{d}}_j = -\hat{\mathbf{d}}_j$. Plugged into the estimation rule (9), we get

$$\hat{\mathbf{d}}_{j+1} \quad = \hat{\mathbf{d}}_j + \mathbf{K}_{j,\mathrm{K}}\mathbf{e}_j, \tag{23}$$

from which we can derive a non-recursive equation for $\hat{\mathbf{d}}_{j+1}$:

$$\hat{\mathbf{d}}_{j+1} = \quad \sum_{i=0}^{j} \mathbf{K}_{j,\mathrm{K}}\mathbf{e}_i, \tag{24}$$

where we set $\hat{\mathbf{d}}_0 = 0$, assuming that the nominal system model holds the best possible guess for $\mathbf{u}_0$. Furthermore, we can write

$$\mathbf{v}_{j+1} \quad = -\tilde{\mathbf{F}}^{-1}\hat{\mathbf{d}}_{j+1} = -\tilde{\mathbf{F}}^{-1}\sum_{i=0}^{j} \mathbf{K}_{j,\mathrm{K}}\mathbf{e}_i, \tag{25}$$

Note that we refer to a given iteration-dependent sequence of gains as $\mathbf{K}_j$. Such a sequence can always be interpreted as optimal Kalman gains obtained from a Kalman filter ($\mathbf{K}_{j,\mathrm{K}}$), see Remark 1.

III. Comparison of the ILC Algorithms

The goal of this section is to understand the similarities and differences of the ILC algorithms introduced in Section 1. In particular, we are interested in understanding how the K-ILC scheme compares to the widely-used QILC and E-QILC schemes. Although only iteration-constant gains $\mathbf{K}_j = \mathbf{K}$ have been proposed for E-QILC, we generalize it to $\mathbf{K}_j$ for comparison.

A. Equivalency of K-ILC and E-QILC

Proposition 1. For equal choice of gain matrices $\mathbf{K}_j$, $j = \{0, 1, 2, \dots\}$ and cost function (14), the predicted errors $\bar{\mathbf{e}}_{j+1}$ and inputs $\mathbf{u}_{j+1}$ are identical for K-ILC and E-QILC for all $j = \{0, 1, \dots\}$.

Proof. The prediction of the next iteration's error for the E-QILC and K-ILC approach from Section II are:

$$\begin{aligned} {}^{\text{E-QILC}}\bar{\mathbf{e}}_{j+1} &= \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{v}_{j+1} + \hat{\mathbf{e}}_j, \\ {}^{\text{K-ILC}}\bar{\mathbf{e}}_{j+1} &= \tilde{\mathbf{F}}\mathbf{v}_{j+1} + \hat{\mathbf{d}}_{j+1}. \end{aligned}$$

Rearranging the prediction of the K-ILC algorithm, we get

$$\begin{aligned} {}^{\text{K-ILC}}\bar{\mathbf{e}}_{j+1} = \quad & \tilde{\mathbf{F}}\mathbf{v}_{j+1} + \hat{\mathbf{d}}_{j+1} = \tilde{\mathbf{F}}(\mathbf{v}_{j+1} - \mathbf{v}_j + \mathbf{v}_j) + \hat{\mathbf{d}}_{j+1} \\ = \quad & \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{v}_{j+1} + \tilde{\mathbf{F}}\mathbf{v}_j + \hat{\mathbf{d}}_{j+1}. \end{aligned}$$

The new formulation suggests the equivalency of ${}^{\text{E-QILC}}\hat{\mathbf{e}}_j = \tilde{\mathbf{F}}\mathbf{v}_j + \hat{\mathbf{d}}_{j+1}$. To prove this assumption we substitute ${}^{\text{E-QILC}}\hat{\mathbf{e}}_j$ in the E-QILC estimation step (6), using $\boldsymbol{\Delta}\mathbf{u}_j = \boldsymbol{\Delta}\mathbf{v}_j$:

$$\begin{aligned} \hat{\mathbf{d}}_{j+1} + \tilde{\mathbf{F}}\mathbf{v}_j &= (\mathbf{I} - \mathbf{K}_{j,\mathrm{K}})((\hat{\mathbf{d}}_j + \tilde{\mathbf{F}}\mathbf{v}_{j-1}) + \mathbf{F}\boldsymbol{\Delta}\mathbf{v}_j) + \mathbf{K}_{j,\mathrm{K}}\mathbf{e}_j \\ \Leftrightarrow \hat{\mathbf{d}}_{j+1} &= \hat{\mathbf{d}}_j - \mathbf{K}_j(\hat{\mathbf{d}}_j + \tilde{\mathbf{F}}\mathbf{v}_j - \mathbf{e}_j). \end{aligned}$$

After some transformation and assuming matching initial conditions, i.e. ${}^{\text{K-ILC}}\hat{\mathbf{d}}_1 = {}^{\text{E-QILC}}\hat{\mathbf{e}}_0$[2], the estimation step is identical to (9). With the prediction used for the cost function being equal, the two algorithms become identical for equivalent cost functions. □

[2]The index difference is only a definition issue as we stick to Lee's *et al.* definition [7].

Interpretation: Although both ILC schemes are based on different approaches, the formulation is equivalent for equal, arbitrary filter gains $\mathbf{K}_j$ and cost function. Although in literature, different cost function have been used, their generalization accounts for both E-QILC and K-ILC. When studying the influence of the choice of $\mathbf{K}_j$ and cost functions J_{j+1}, the findings are hence valid for E-QILC as well as K-ILC. In the following, we will thus only refer to the K-ILC algorithm for the comparison (see also Remark 1).

B. Comparing K-ILC and QILC with Identical Cost Functions

Proposition 2. Choosing $\mathbf{K}_j = \mathbf{I}\ \forall j$ in the K-ILC estimation steps, the resulting K-ILC algorithm is identical to the QILC approach given equal cost functions (14).

Proof. The equivalency of K-ILC and E-QILC being proven, comparing QILC to the E-QILC estimation covers both cases. Setting $\mathbf{K}_j = \mathbf{I}$ in the estimation rule (6), we get $\hat{\mathbf{e}}_j = \mathbf{e}_j$. The estimation for the error becomes thus the last measured error. Plugged into the prediction step of E-QILC, $\hat{\mathbf{e}}_j + \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{u}_{j+1} = \mathbf{e}_j + \tilde{\mathbf{F}}\boldsymbol{\Delta}\mathbf{u}_{j+1}$, which is identical to the QILC prediction, see (4). □

Interpretation: Any arbitrary QILC algorithm is thus a special case of the K-ILC approach with *iteration-constant* gain $\mathbf{K}_j = \mathbf{K}$ and equal cost functions.

Proposition 3. A system that has negligible output noise, i.e. $\mathbf{H}_j \sim 0\ \forall j$ results in an optimal Kalman gain $\mathbf{K}_j = \mathbf{I}\ \forall j$.

Proof. If we analyse the Kalman filter equation in (11), we see that the gain matrix is $\mathbf{K}_j = \mathbf{I}$ in the case of $\mathbf{H}_{j+1} = 0$ or, more generally, $\mathbf{H}_{j+1} \ll \mathbf{P}_j + \mathbf{E}_j$ such that $(\mathbf{P}_j + \mathbf{E}_j)(\mathbf{P}_j + \mathbf{E}_j + \mathbf{H}_{j+1})^{-1} \simeq \mathbf{I}$. □

Interpretation: If the output noise is zero, i.e. $\mathbf{H}_j = 0$, the Kalman filter completely trusts the measurements, resulting in $\mathbf{K}_j \simeq \mathbf{I}$. The covariance $\mathbf{H}_j$ models measurement and process noise. We conclude from Proposition 2 and 3 that QILC can be interpreted as a special case of the K-ILC algorithm assuming *zero output noise*.

C. Comparing QILC and K-ILC with Typical Weighting Matrix Choice

Note that all statements made in Section III-B are valid for general cost functions (14) (and even for (12)). In the following, we will consider (14) with typical weighting matrix choice as found in literature. In particular, for the K-ILC, a standard choice is ${}^{\text{K-ILC}}\mathbf{W}_{\Delta u} = 0$. The idea behind it is that $\mathbf{W}_{\Delta u}$ takes the role of a noise filter in QILC by minimizing the change in input based on the current error signal. Thus, in the QILC algorithm, the cost function couples filtering and input update. In order to compare QILC and K-ILC in this context, we make use of the two explicit formulas for the updated input $\mathbf{u}_{j+1}$, (17) and (25):

$$\begin{aligned} {}^{\text{QILC}}\mathbf{u}_{j+1} &= \mathbf{u}_{\text{nom}} - \sum_{i=1}^{j} {}^{\text{QILC}}L\mathbf{e}_j, \\ {}^{\text{K-ILC}}\mathbf{u}_{j+1} &= \mathbf{u}_{\text{nom}} - \sum_{i=1}^{j} {}^{\text{K-ILC}}L\mathbf{e}_j. \end{aligned} \tag{26}$$

Assuming that we choose the same initial guess and nominal input $\mathbf{u}_{\text{nom}}$, the two approaches only differ in terms of ${}^{\text{QILC}}L$ and ${}^{\text{K-ILC}}L$:

$$\begin{aligned} {}^{\text{QILC}}L &= (\mathbf{W}_{\Delta u} + \tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e\tilde{\mathbf{F}})^{-1}\tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e, \\ {}^{\text{K-ILC}}L &= \tilde{\mathbf{F}}^{-1}\mathbf{K}_j. \end{aligned} \tag{27}$$

We can compare both algorithms by analyzing the two matrices, ${}^{\text{QILC}}L$ and ${}^{\text{K-ILC}}L$. Note that we have two matrices $\mathbf{W}_e$ and $\mathbf{W}_{\Delta u}$ to define ${}^{\text{QILC}}L$ and, therefore, some redundancy. In fact, we can set $\mathbf{W}_e = \mathbf{I}$ without loss of generality. For K-ILC, the influence of $\mathbf{W}_e$ has cancelled out.

Lemma 1. By choosing the design parameters of the K-ILC algorithm (11) to be $\mathbf{P}_0 = p_0\mathbf{X}, \mathbf{H}_j = \eta\mathbf{X}, \mathbf{E}_j = \epsilon\mathbf{X}$; $p_0, \eta, \epsilon \in \mathbb{R}$ for arbitrary $\mathbf{X}$ (commonly, $\mathbf{X} = \mathbf{I}$), the Kalman gain $\mathbf{K}_{j,\text{K}}$ is diagonal and of the form $\mathbf{K}_{j,\text{K}} = \lambda_j\mathbf{I}$.

Proof. Plugging $\mathbf{P}_0 = p_0\mathbf{X}, \mathbf{H}_j = \eta\mathbf{X}, \mathbf{E}_j = \epsilon\mathbf{X}$ into the Kalman gain equations (11), we see that $\mathbf{P}_j$ and $\mathbf{S}_j$ are of the form $\psi\mathbf{X}$, $\psi \in \mathbb{R}$, , and hence $\mathbf{K}_{j,\text{K}} = \lambda_j\mathbf{X}\mathbf{X}^{-1}$, i.e. a scaled unity matrix. □

Interpretation: If $\mathbf{H}_j, \mathbf{E}_j, \mathbf{P}_0$ are modelled to be structurally similar – that is, they are scaled versions of an arbitrary matrix $\mathbf{X}$ – each iteration's gain $\mathbf{K}_{j,\text{K}}$ is diagonal ($\mathbf{K}_{j,\text{K}} = \lambda_j\mathbf{I}$).

Proposition 4. For any given QILC with weightings as in (14), there exists a corresponding, general estimation gain $\mathbf{K}_j$ resulting in an equivalent K-ILC algorithm with $\mathbf{W}_{\Delta u} = 0$ in the cost function (14), i.e. a cost function as in (18).

Proof. Following directly from the equations (27), we can set ${}^{\text{QILC}}L = {}^{\text{K-ILC}}L$ and solve for $\mathbf{K}_j$:

$$\mathbf{K}_j = \tilde{\mathbf{F}}(\mathbf{W}_{\Delta u} + \tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e\tilde{\mathbf{F}})^{-1}\tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e. \tag{28}$$

□

Interpretation: The intuitive explanation is as follows: the component penalizing $\boldsymbol{\Delta}\mathbf{u}_j$ in the QILC cost function (14) takes a similar role as the filter in the K-ILC algorithm. However, while in the K-ILC algorithm the filter properties can be separately designed via the initial covariance $\mathbf{P}_0$, $\mathbf{H}_j$, and $\mathbf{E}_j$, the QILC scheme combines filtering and input update in one optimization step. In addition, the filtering in the K-ILC case is designed to be *iteration-varying allowing fast adaptation initially and rejecting noise in later stages*. In the QILC case, the 'filtering' via $\mathbf{W}_{\Delta u}$ is kept constant. Note also that this proof assumes no soft or hard input constraints and the squared 2-norm in the cost function.

We can also turn around the argument to determine the QILC weightings depending on $\mathbf{K}_j$:

Proposition 5. For any K-ILC with gain matrix $\mathbf{K}_{j,\text{K}}$ and a cost function as in (18), we can find a QILC set of weighting matrices $\mathbf{W}_e$ and $\mathbf{W}_{\Delta u}$ that results in an equivalent behavior for that specific iteration j.

Proof. Solving (28) for $\mathbf{W}_{\Delta u}$ results in

$$\mathbf{W}_{\Delta u} = \tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e(\mathbf{K}_{j,\text{K}}^{-1} - \mathbf{I})\tilde{\mathbf{F}}. \tag{29}$$

One sees that we can find a matching $\mathbf{W}_{\Delta u}$ for any arbitrary $\mathbf{W}_e$. The choice of $\mathbf{W}_e$ is, in fact, redundant for QILC (given it is positive definite) and is usually set to $\mathbf{W}_e = \mathbf{I}$. □

Interpretation: One can now find a QILC equivalent for every possible $\mathbf{K}_{j,\text{K}}$, i.e. for every j-th iteration of one learning process. Hence it is possible to compare the initial and converged behaviour of K-ILC with an analogous QILC. Such a comparison is found in Section IV.

From Proposition 5 we can derive the formula for the common case of diagonal $\mathbf{K}_j = \lambda_j\mathbf{I}, \lambda_j \in \mathbb{R}$: From (29) we derive $\mathbf{W}_{\Delta u} = \frac{1-\lambda_j}{\lambda_j}\tilde{\mathbf{F}}^{\mathrm{T}}\mathbf{W}_e\tilde{\mathbf{F}}$.

D. K-ILC Convergence Behavior

Using a Kalman filter, a model inversion input update occurs in two practical cases: first, if there is negligible output noise (Proposition 3) and second. at initial iteration steps. As for initially unknown disturbances (i.e. unmodelled dynamics or exogenous disturbances) a large initial state uncertainty $\mathbf{P}_0 \gg \mathbf{H}_1$ is assumed. This leads to an identity matrix as initial gain, $\mathbf{K}_0 = (\mathbf{P}_0+\mathbf{E}_0)(\mathbf{P}_0+\mathbf{E}_0+\mathbf{H}_1)^{-1} \simeq (\mathbf{P}_0+\mathbf{E}_0)(\mathbf{P}_0+\mathbf{E}_0)^{-1} = \mathbf{I}$, and a model inversion for the first update step (see (27)). The gain quickly converges to $\mathbf{K}_\infty$ during the following iterations, as the static errors are expected to be corrected immediately.

For both cases, a model inversion makes sense as the output noise is negligible versus the modelled state uncertainty $\mathbf{P}_j$. Hence, an estimator cannot filter any further information.

As discussed above, the Kalman filter equations (11) provide optimal, iteration-varying gains, which change the characteristics of the learning scheme over the course of a learning experiment. The case of a large $\mathbf{P}_0$ is common in practice, where the modelling errors are the largest source of uncertainty, which result in high initial errors that can be quickly compensated for with a model inversion.

IV. Simulation Example

In the simulation we compare a K-ILC and a QILC algorithm applied on a mass-spring-damper system (see Fig. 2). For comparison we also include the QILC equivalent of both the initial and converged K-ILC algorithm, highlighting the findings of the previous section.

A. Nominal Model of the Mass-Spring-Damper System

We consider a classic 1-D mass-spring-damper system with the equations of motion,

$$\ddot{x} = F/m - \frac{d}{m}\dot{x} - \frac{c}{m}x, \tag{30}$$

where we neglect the damping and spring in our *nominal* model; i.e. $c = d = 0$ and the force is the applied input $F/m = \gamma u$. Discretized with a zero-order hold and a

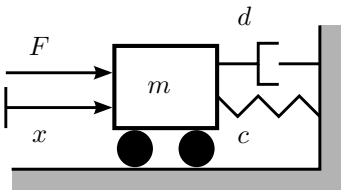

Fig. 2. We use a mass-springer-damper system with the input force F as our simulation example.

sample time T_s, we obtain the following discrete-time system as the nominal model:

$$\begin{aligned} x[k+1] &= \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} x[k] + \begin{bmatrix} T_s^2/2 \\ T_s \end{bmatrix} u[k] \\ y[k] &= \begin{bmatrix} 1 & 0 \end{bmatrix} x[k] + \begin{bmatrix} 0 \end{bmatrix} u[k], \end{aligned} \tag{31}$$

where $x[k] \in \mathbb{R}^2$ represents position and velocity at time kT_s. This model is used in the learning algorithms in the iteration-domain matrix $\tilde{\mathbf{F}}$, refer to [20] on how $\tilde{\mathbf{F}}$ is obtained from (31).

B. Real System Dynamics of the Mass-Spring-Damper

To show the effect of the learning, we choose the 'real' (i.e., simulated) dynamics to be different from the nominal model. We add the spring and damper constants as follows: $c/m = \alpha, d/m = \alpha/10$.

For the simulation of the 'real' system we also use a zero-order-hold discretization. We set $\alpha = 0.01$ and $\gamma = 0.9$ and simulate over $N = 160$ time steps. Note that only the 'real' system with $\alpha \neq 0$ is asymptotically stable, whereas (31) has two eigenvalues at 1. As additional disturbances, we add both a static disturbance $d_{\text{static}}[k] = 0.2\sin^2(\frac{4\pi k}{N})$ and a Gaussian noise with $\mathcal{N} \sim (0, \sigma)$ to the output signal $y[k]$. The desired trajectory is set to $\mathbf{y}_d[k] = \sin^2(k\frac{2\pi}{2N})$.

TABLE I

PARAMETERS USED FOR SIMULATIONS AND CONVERGED ERROR.

	$\mathbf{W}_{\Delta u}$	$\mathbf{W}_e$	$\|\mathbf{e}_\infty\|_{\sigma_I=0.1}$	$\|\mathbf{e}_\infty\|_{\sigma_{II}=0.01}$
K-ILC	0	$\mathbf{I}$	0.1201	0.0115
QILC	$0.3\mathbf{I}$	$\mathbf{I}$	0.1159	0.0114
QILC-c	$\lambda_\infty \tilde{\mathbf{F}}^{\mathrm{T}}\tilde{\mathbf{F}}$	$\mathbf{I}$	0.1197	0.0117
QILC-i	0	$\mathbf{I}$	0.1482	0.0152
K-ILC	$\mathbf{P}_0 = 100\mathbf{I}$, $\mathbf{H}_j = 0.1\mathbf{I}$, $\mathbf{E}_j = 0.001$, $\lambda_\infty = 0.095$			

C. Simulation Results and Discussion

Four different learning algorithms are studied. All use the same linear, discrete-time model $\tilde{\mathbf{F}}$ based on (31). The learning parameters are chosen as stated in Table I. We test two different scenarios with different noise variances $\sigma_I = 0.01$ and $\sigma_{II} = 0.1$. To compare the algorithm's behavior we plot the evolution of each learning iteration's error 2-norm. Also, we average over 120 repetitions of the entire learning process to reduce the noise influence. The first input guess $\mathbf{u}_0 = \mathbf{u}_{\text{nom}}$ is the nominal input given by the model inversion $\tilde{\mathbf{F}}^{-1}\mathbf{y}_d = \mathbf{u}_{\text{nom}}$.

The results are:

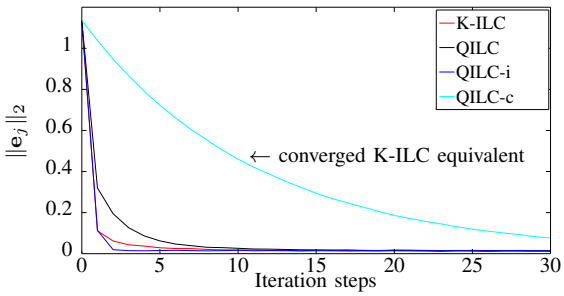

Fig. 3. Average errors of the K-ILC, QILC and model inversion (QILC-i) algorithms with a white output noise with variance $\sigma_I = 0.01$.

K-ILC: The estimation-based learning algorithm is used with optimal Kalman gains based on diagonal covariance matrices for the modelled stochastic variables similar to [8], see Table I. We see fast convergence initially and good noise rejection after convergence for both small and large noise variance σ_I and σ_{II}, see Figures 3 and 4. In Table I we further list the converged error norm $\|\mathbf{e}_\infty\|_2$. The converged error norm was approximated by the average error norms of iterations 190 to 200. Note that the mean (in contrast to the norm of the simulated converged errors) have all magnitudes of the order 10^{-6} and suggest that the error has no significant bias.

QILC-i: Introduced as a special case of QILC (19), we now can interpret the model-inversion learning as a special case of K-ILC for output noise modeled to be negligible (see interpretation of Proposition 3). In both cases, Figures 3 and 4, the initial convergence rate of QILC-i is very high. Note that the initial behavior of K-ILC with high $\mathbf{P}_0$ approximates QILC-i. Especially for the larger amplitude noise $\sigma_{II} = 0.1$ the high error sensitivity of QILC-i produces the worst converged performance.

QILC: QILC uses parameters as commonly chosen in literature, cf. [7], [23]: a unit weighting for the error and a scalar factor for the input change penalization, see Table I. The algorithm performs decently for large noise levels; the slow convergence rate is significant for the small noise amplitude case.

QILC-c: QILC-c uses parameter values that are equivalent to the converged K-ILC, $\mathbf{K}_{\infty,\mathrm{K}} = \lambda_\infty \mathbf{I}$ according to Proposition 4. The converged λ_∞ can be found by simulation or analytically, see [22]. Though robust for noise, it has a very slow learning rate performing badly for initial large disturbances.

V. CONCLUSION

In a first step, we proved the equivalency of both estimation-based ILC algorithms, K-ILC as presented by Schoellig *et al.* [8] and E-QILC as proposed by Lee *et al.* [7]. We demonstrated that for equal cost functions, the learning behavior is defined by the choice of gains $\mathbf{K}_j$.

By comparing QILC and K-ILC we further demonstrated that QILC can be viewed as a special case of a generalized estimation-based algorithm with $\mathbf{K}_j = \mathbf{I}$. We compared the

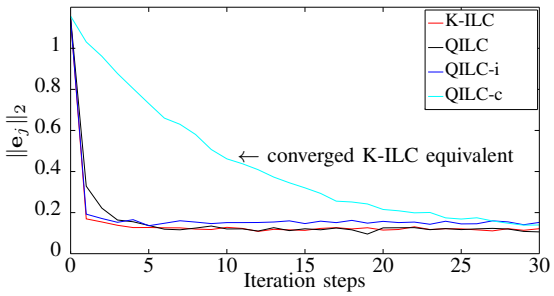

Fig. 4. Average error of the K-ILC, QILC and model inversion (QILC-i) algorithms with a white output noise with variance $\sigma_{II} = 0.1$.

initial and converged K-ILC behavior to the corresponding QILC choice (refer to both Sections III and IV), and saw that an optimal Kalman gain results in an iteration-varying learning action – even for a basic stochastic model with iteration-constant diagonal covariance matrix. The resulting learning performance is improved, as it changes from fast convergence and high sensitivity (model inversion) for the initial iterations to less sensitive converged $\mathbf{K}_{\infty,\mathrm{K}}$.

The estimation gain evolving over the iterations of a learning experiment allows optimization of the scheme for initial as well as converged behavior. With that, it also improves the general robustness towards uncertainties, as it performs well for small noise σ_I (see Fig. 3) and larger noise σ_{II} (see Fig. 3 and 4).

For QILC, the lack of general design rules for the weighting matrices make a trial-and-error tuning necessary. Compared to that, for the estimation-based ILC using Kalman filters the stochastic model provides tuning parameters that have an intuitive interpretation and that can incorporate known noise information; for example, information about the expected noise frequencies.

REFERENCES

[1] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operations of dynamic systems by learning: A new control theory for servomechanism or mechatronic systems," in *Proc. of the Conference on Decision and Control (CDC)*, 1984, pp. 1054–1069.

[2] D. A. Bristow and A. G. Alleyne, "A high precision motion control system with application to microscale robotic deposition," *Transacions on Control Systems Technology*, vol. 14, no. 6, pp. 1008–1020, 2006.

[3] S. Gunnarsson and M. Norrlöf, "On the design of ILC algorithms using optimization," *Automatica*, vol. 37, pp. 2011–2016, 2001.

[4] D. Owens and J. Hätönen, "Iterative learning control – an optimization paradigm," *Annual Reviews in Control*, vol. 29, no. 1, pp. 57–70, 2005.

[5] M. Norrlöf, "An adaptive iterative learning control algorithm with experiments on an industrial robot," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 141–146, 2002.

[6] K. S. Lee and J. H. Lee, "Model predictive control for nonlinear batch processes with asymptotically perfect tracking," *Computers & Chemical Engineering*, vol. 21, pp. S873–S879, 1997.

[7] J. H. Lee, K. S. Lee, and W. C. Kim, "Model-based iterative learning control with a quadratic criterion for time-varying linear systems," *Automatica*, vol. 36, pp. 641–657, 2000.

[8] A. P. Schoellig and R. D'Andrea, "Optimization-based iterative learning control for trajectory tracking," in *Proc. of the European Control Conference (ECC)*, 2009, pp. 1505–1510.

[9] F. L. Mueller, A. P. Schoellig, and R. D'Andrea, "Iterative learning of feed-forward corrections for high-performance tracking," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 3276–3281.

[10] A. P. Schoellig, F. L. Mueller, and R. D'Andrea, "Optimization-based iterative learning for precise quadrocopter trajectory tracking," *Autonomous Robots*, vol. 33, no. 1-2, pp. 103–127, 2012.

[11] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Control Systems Magazine*, pp. 96–114, 2006.

[12] H.-S. Ahn, Y. Chen, and K. L. Moore, "Iterative learning control: Brief survey and categorization," *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1099–1121, 2007.

[13] K. Buchheit, M. Prandit, and M. Befort, "Optimal iterative learning control of an extrusion plant," in *Proc. of IEEE International Conference on Control*, no. 389, 1994, pp. 652–657.

[14] K. L. Moore, *Iterative Learning Control for Deterministic Systems*. Springer, London, 1993.

[15] B. Chu and D. H. Owens, "Iterative learning control for constrained linear systems," *International Journal of Control*, vol. 83, no. 7, pp. 1397–1413, 2010.

[16] B. Panomruttanarug and R. W. Longman, "Using Kalman filter to attenuate noise in learning and repetitive control can easily degrade performance," in *Proc. of the IEEE SICE Annual Conference*, 2008, pp. 3453–3458.

[17] J. Wallen, S. Gunnarsson, R. Henriksson, S. Moberg, and M. Norrlöf, "ILC applied to a flexible two-link robot model using sensor-fusion-based estimates," in *Proc. of the Joint 48th IEEE Conference on Decision and Control and the 28th Chinese Control Conference*, no. 1, 2009, pp. 458–463.

[18] J. Wallen, M. Norrlöf, and S. Gunnarsson, "A framework for analysis of observer-based ILC," *Asian Journal of Control*, vol. 13, no. 1, pp. 3–14, 2011.

[19] M. Norrlöf, "Disturbance rejection using an ILC algorithm with iteration varying filters," *Asian Journal of Control*, vol. 6, no. 3, pp. 432–438, 2004.

[20] N. Amann, D. H. Owens, and E. Rogers, "Iterative learning control for discrete-time systems with exponential rate of convergence," *IEE Proc. Control Theory and Applications*, vol. 143, no. 2, pp. 217–224, 1996.

[21] C. K. Chui and G. Chen, *Kalman Filtering with Real-Time Applications*. Springer, London, 1987, vol. 6.

[22] N. Degen and A. P. Schoellig, "Additional derivations related to the analysis of the Kalman-filter-based ILC scheme," 2014. [Online]. Available: http://www.tiny.cc/DerivationsKalmanILC

[23] D. A. Bristow, "Weighting matrix design for robust monotonic convergence in norm optimal iterative learning control," *Proc. of the American Control Conference (ACC)*, pp. 4554–4560, 2008.