10.24425/acs.2020.132590

Archives of Control Sciences
Volume 30(LXVI), 2020
No. 1, *pages 157–176*

Constrained Output Iterative Learning Control

KALOYAN YOVCHEV, KAMEN DELCHEV and EVGENIY KRASTEV

Iterative Learning Control (ILC) is a well-known method for control of systems performing repetitive jobs with high precision. This paper presents Constrained Output ILC (COILC) for non-linear state space constrained systems. In the existing literature there is no general solution for applying ILC to such systems. This novel method is based on the Bounded Error Algorithm (BEA) and resolves the transient growth error problem, which is a major obstacle in applying ILC to non-linear systems. Another advantage of COILC is that this method can be applied to constrained output systems. Unlike other ILC methods the COILC method employs an algorithm that stops the iteration before the occurrence of a violation in any of the state space constraints. This way COILC resolves both the hard constraints in the non-linear state space and the transient growth problem. The convergence of the proposed numerical procedure is proved in this paper. The performance of the method is evaluated through a computer simulation and the obtained results are compared to the BEA method for controlling non-linear systems. The numerical experiments demonstrate that COILC is more computationally effective and provides better overall performance. The robustness and convergence of the method make it suitable for solving constrained state space problems of non-linear systems in robotics.

Key words: constrained output systems, convergence analysis, iterative learning control, robot manipulators

1. Introduction

The main idea of the Iterative Learning Control (ILC) is to compensate the tracking error for systems that have to perform a repetitive job with high precision. This is done by tracking multiple consequent iterations of the job execution, and between each one the input signals (commands) are improved in order to correct the error based on the data collected so far. Thus, in a natural iterative process of

K. Yovchev (Corresponding author), E-mail: k.yovchev@fmi.uni-sofia.bg, K. Delchev and E. Krastev are with the Faculty of Mathematics and Informatics, Sofia University, 1164 Sofia, 5 James Bourchier Blvd., Bulgaria.

The support by the Fund for Scientific Research at Sofia University “St. Kliment Ohridski” under grant 80-10-7/2019, is acknowledged.

Received 19.07.2019. Revised 2.02.2020.

self-learning, the input signals (commands) for achieving the highest precision are computed.

The idea of ILC is mentioned for the first time in a scientific publication by Uchiyama in 1978 [1], which however is in Japanese and therefore does not gain much popularity. A patent was filed in the United States in 1967, which was accepted in 1971. Its title is “Learning control of actuators in control systems” [2]. The idea is to memorize the control signals in the computer memory and then to perform subsequent iterative update of these signals, according to the difference between the set and the actual behavior of the system. The way in which this improvement is made is not clearly formulated in the patent. Later, in 1984 ILC began to be explored in more detail. Then Arimoto et al. [3, 4], Casalino and Bartolini [5] and Craig [6], independently of each other, published researches about a method that can, by successive iterations of the same assignment, correct errors in the mathematical model, as well as the determined system disturbances. The name Iterative Learning Control was first used by Arimoto after it was originally called a “bettering process”. The ILC study is mainly related to its applications in the field of robotics. A 1998 publication of Moore [7] provides a very good overview of the method studies made so far. At the end of the nineties and the beginning of the new century, the focus of the studies was shifting away from a study of the robustness of the method to its design and performance. Investigations in this direction are by researchers Bien and Xu [8], Norrlof (2000) [9], Lee et al. (2000) and Longman (2000) [10]. The convergence analysis is important for the synthesis of the ILC method. For linear systems there are various proofs of the convergence of the ILC method [7, 8, 10, 11]. For nonlinear systems the convergence is proven in 1989 by Heizinger [12].

One of the main problems of ILC is the growth of the transient error. The essence of the problem consists in the possibility of several iterations, in which the error increases many times before it starts to decrease again and to converge to zero. This problem occurs in real conditions and which may not allow the method to achieve the desired result because it can make it impossible to perform the necessary number of iterations to achieve the required precision. This is a serious problem, especially when using the ILC method in real-world conditions, because such a deviation is beyond the real robot’s running capabilities without disturbing the constraints of the generalized coordinates. This problem is addressed by Longman, R.W. and Huang, Y. in [13]. Our previous research made in 2017 confirms the existence of the transient growth problem through a computer simulation of PUMA 560 robotic manipulator. The result from the computer simulation is shown in Fig. 1. During the first iterations the tracking error is increasing but after that the ILC process is convergent [14].

A rather limited number of solutions to the problem of transient error is known in the scientific literature concerning ILC. A possible solutions to the transient error problem is the slow learning rate ILC [15], the monotone ILC [16], or the Bounded-Error Algorithm (BEA) [17].

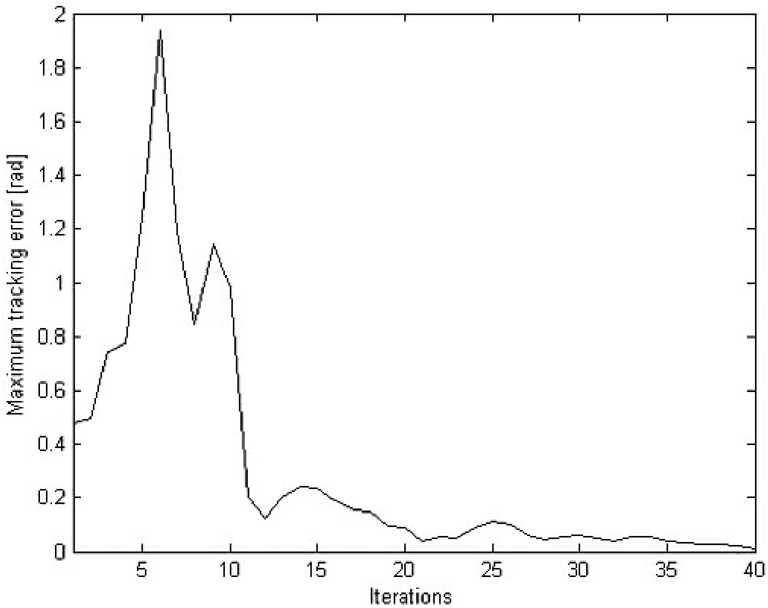

Figure 1: The transient growth error problem of ILC [14]

In 2009 Douglas A. Bristow and John R. Singler, in study [11] investigate whether the slow learning ILC can solve the transient error problem. This method consists of the following: reducing the influence of the learning operator in order to achieve a smaller and smoother correction of the input signals of each iteration. A smaller correction reduces the possibility of a major transient error, but then a larger number of iterations is needed to reach the desired accuracy. In addition, in this case, convergence is only proven for linear systems.

A study by Kwang-Hyun Park and Zeungnam Bien in 2002 [16] shows briefly how to achieve monotonic error convergence. The method proposed there separates the time interval $[0, T]$ into subintervals in which the standard ILC scheme is applied sequentially. Separation is specifically selected so as not to allow a large transient error to occur. This method violates the ILC requirement for equivalence of the initial conditions (since once the trajectory has been reached within a given interval, the beginning is shifted to its end and continued to the next) in the course of the iterations. Moreover, it is not quite clear how to choose the length of the subinterval.

The most common and most straightforward solution to the transient error problem is the use of the Bounded-Error Algorithm (BEA, BEA method, BEILC) proposed in 2013 [18]. This algorithm deals with the problem by tracking the magnitude of the error during the iteration itself. As soon as it reaches a predetermined limit value, the current execution is terminated. The correction of the input signals before the next iteration is performed only until the previous one has been terminated. This violates the postulate about the uniformity of the duration of each iteration, but the convergence of the method is proven in [18].

Other major disadvantage of the ILC methods is that they cannot be applied directly to systems in which there are constraints, e.g. constraints of input (control) signals, state space constraints or velocity constraints [19, 20]. In the study [21] appeared in 2009, two methods are proposed to solve the constraints of the input control signals: the first uses a reference governor to reduce the motion parameters on the given path so as to obtain an executable trajectory and the second uses a barrier function that does not allow the generation of an input signal which violates system constraints. These methods work well and solve the problem of constraints when the specified trajectory is unfeasible due to constraints in input control signals [21]. Optimization ILC methods are proposed in [19, 22] which take into account the dynamics model. As a result, they manage to solve the constraints problem in the case of a swinging pendulum. Research [20] proposes a projection method that solves the problem of constrained input signals.

Most studies mainly address restrictions on input signals or speeds [20–22]. The convergence of these methods has not been proven when used to solve the problem with constraints in the state space coordinates [24], making them unusable for manipulation robots, for example, in cases where the trajectory is planned close to the constraints of the generalized coordinates (Fig. 2). In 2013, Guth et al. proposed a variable pass length ILC method for linear state

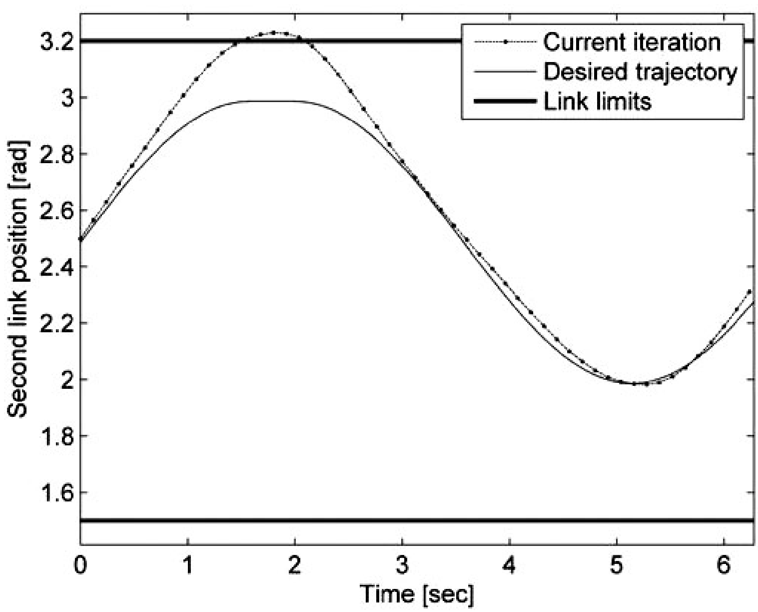

Figure 2: ILC in constrained state space [23]

space constrained systems [25]. In 2018, Sebastian et al. proposed a barrier-function like Lyapunov function to be used to design a new state feedback (or a proportional derivative controller) in order to ensure that output constraints are satisfied in the finite time-domain [26]. In 2019, Zamani et al. proposed model predictive control (MPC) strategy for a constrained and unconstrained linearized system [27]. Those methods are either developed only for linear systems or they are too computationally expensive. Our previous research made in 2018 presents ILC approach which introduces a new condition for the desired trajectory planning with respect to state space constraints which specifies the value of the norm-bound ε on the tracking error for implementation of BEA. The norm-bound ε on the tracking error enables the application of ILC based control scheme within constrained state space for multiple-input multiple-output nonlinear systems [23]. However, the proposed approach enforces tight restrictions over the possible output trajectory.

The main goal of this paper is to propose a new general solution for applying the ILC method within constrained state space. This new solution will be based over the BEA method, where the learning update law will be altered in order to allow the executed output trajectory to be into the whole area defined by the state space constraints. According to the conclusion of our previous research [28] this will lead to a faster convergence rate.

The paper is organized as follows: Section 2 presents the formulation of the problem and describes briefly the BEA method. Section 3 presents the new update law and proof of its convergence. Then, the proposed method for ILC within a constrained state space is validated and evaluated by computer simulations in Section 4.

2. Problem formulation

Let's consider the following class of multi-input-multi-output nonlinear time-varying state-space equations:

$$\begin{aligned} \dot{\boldsymbol{x}}_l(t) &= f\left(\boldsymbol{x}_l(t), t\right) + B\left(\boldsymbol{x}_l(t), t\right) \boldsymbol{u}_l(t) + \omega_l(t), \\ \boldsymbol{y}_l(t) &= g(\boldsymbol{x}_l(t), t), \end{aligned} \tag{1}$$

where: l is iteration number, for $l \in \{0, \ldots, \infty\}$ and all $t \in [0, T]$, $\boldsymbol{x}_l(t) \in \mathbb{R}^n$, $\boldsymbol{y}_l(t) \in \mathbb{R}^m$, $\boldsymbol{u}_l(t) \in \mathbb{R}^r$ are not necessarily continuous, and $\omega_l(t) \in \mathbb{R}^n$ represents both deterministic and random disturbances. The functions $f : \mathbb{R}^n \times [0, T] \to \mathbb{R}^n$ and $B : \mathbb{R}^n \times [0, T] \to \mathbb{R}^{n \times r}$ are piecewise continuous in $t \in [0, T]$ and $g : \mathbb{R}^n \times [0, T] \to \mathbb{R}^n$ is differentiable in $\boldsymbol{x}$ and t, with partial derivatives $g_x(\cdot, \cdot)$ and $g_t(\cdot, \cdot)$. In addition, the following assumptions hold (adopted from [12, 29]):

A. For each fixed initial state $\boldsymbol{x}(0)$ with $\omega(\cdot) \equiv 0$ the output map $O\colon C([0,T],\mathbb{R}^r)\times\mathbb{R}^n \to C([0,T],\mathbb{R}^m)$ and the state map $S\colon C([0,T],\mathbb{R}^r)\times\mathbb{R}^n \to C([0,T],\mathbb{R}^n)$ are one-to-one. In this notation $\boldsymbol{y}_l(\cdot) = O(\boldsymbol{u}_l(\cdot), \boldsymbol{x}_l(0))$ and $\boldsymbol{x}_l(\cdot) = S(\boldsymbol{u}_l(\cdot), \boldsymbol{x}_l(0))$.

B. The disturbance $\omega_l(\cdot)$ is bounded on $[0,T]$ i.e. $\|\omega(t)\| \leqslant b_\omega$, $\|\cdot\|$ is the Euclidean norm.

C. The functions $f(\cdot,\cdot)$, $B(\cdot,\cdot)$, $g_x(\cdot,\cdot)$, and $g_t(\cdot,\cdot)$ are uniformly globally Lipschitz in $\boldsymbol{x}$ on the interval $[0,T]$.

D. The operators $B(\cdot,\cdot)$ and $g_x(\cdot,\cdot)$ are bounded on $[0,T]\times\mathbb{R}^n$.

E. All functions are assumed measurable and integrable.

2.1. Standard ILC procedure

Let us consider the following update law of standard ILC procedure:

$$\boldsymbol{u}_{l+1}(t) = (1-\gamma)\,\boldsymbol{u}_l(t) + \gamma\boldsymbol{u}_0(t) + L\left(\boldsymbol{y}_l(t),t\right)\left(\dot{\boldsymbol{y}}_d(t) - \dot{\boldsymbol{y}}_l(t)\right), \tag{2}$$

where $L\colon \mathbb{R}^m\times[0,T]\to\mathbb{R}^{n\times m}$ is a bounded learning operator and $\gamma\in[0,1)$ allows the influence of a bias term.

Lemma 2 [12]. *If $\{a_l\}$, $l\in\{0,\ldots,\infty\}$ is a sequence of real numbers such that $|a_{l+1}| \leqslant \rho\,|a_l| + \hat{\varepsilon}$, $0\leqslant\rho<1$, then $|a_l| \leqslant (1-\rho)^{-1}\,\hat{\varepsilon}$ when l tends to infinity.*

The proof of Lemma 1 is given in [12].

Theorem 1 [12, 29]. *Let the system described by (1) satisfy assumptions A–E and use the update law (2). Given an attainable desired trajectory $\boldsymbol{y}_d(t) = g(\boldsymbol{x}_d(t),t)$, $t\in[0,T]$, $\boldsymbol{u}_d(t)$ is the corresponding input and $\boldsymbol{x}_d(t)$ is the corresponding state (according to assumption A). If*

$$\begin{aligned}&\left\|(1-\gamma)\boldsymbol{I} - L\left(g(\boldsymbol{x},t),t\right) g_x(\boldsymbol{x},t)B(\boldsymbol{x},t)\right\| \leqslant \rho < 1,\\ &\forall\,(\boldsymbol{x},t)\in\mathbb{R}^n\times[0,T],\end{aligned} \tag{3}$$

and the initial state error $\|\boldsymbol{x}_d(0) - \boldsymbol{x}_l(0)\|$ is bounded by b_{x0}, then, as $l\to\infty$, the error between $\boldsymbol{u}_l(t)$ and $\boldsymbol{u}_d(t)$ is bounded. In addition, the state and output asymptotic errors are bounded. These bounds depend continuously on the bound on the initial state error, bound on the state disturbance, and γ, as b_{x0}, b_ω, and γ tend to zero, these bounds also tend to zero.

The proof of Theorem 1 is presented by Heinzinger et al. in [12, 29], where it is proven that there exists $\lambda = \lambda(T)$:

$$\|\delta \boldsymbol{u}_{l+1}(t)\|_\lambda \leqslant \bar{\rho}\,\|\delta \boldsymbol{u}_l(t)\|_\lambda + \hat{\varepsilon}, \quad 0 < \bar{\rho} < 1, \tag{4}$$

where $\delta \boldsymbol{u}_l(t) \equiv \boldsymbol{u}_d(t) - \boldsymbol{u}_l(t)$, and the time-weighted norm (λ norm) is defined by:

$$\|\delta \boldsymbol{u}_l(t)\|_\lambda \equiv \sup_{t\in[0,T]} e^{-\lambda t}\,\|\delta \boldsymbol{u}_l(t)\|\,. \tag{5}$$

Applying Lemma 1 to the inequality (4) yields:

$$\begin{aligned} &\limsup_{l\to\infty} e^{-\lambda t}\,\|\delta \boldsymbol{u}_l(t)\|_\lambda \leqslant (1-\bar{\rho})^{-1}\,\hat{\varepsilon}, \\ &\limsup_{l\to\infty} e^{-\lambda t}\,\|\delta \boldsymbol{y}_l(t)\|_\lambda \leqslant b_y, \quad \text{where } \delta \boldsymbol{y}_l(t) \equiv \boldsymbol{y}_d(t) - \boldsymbol{y}_l(t). \end{aligned} \tag{6}$$

If $b_{x0} \to 0$, $b_\omega \to 0$ and $\gamma \to 0$, then $\hat{\varepsilon} \to 0$ and $b_y \to 0$.

It has to be mentioned that according to Heizinger [12, 29], from (6), the tracking accuracy of the output trajectory of the standard ILC process (update law (1)) is μ: $\mu = \|\delta \boldsymbol{y}_l(t)\|_\infty \leqslant e^{\lambda T} b_y$ and $\|\delta \boldsymbol{y}_l(t)\|_\infty \equiv \sup_{t\in[0,T]} \|\delta \boldsymbol{y}_l(t)\|$.

It follows, that the standard ILC procedure with update law $\boldsymbol{u}_l$ from (2) is convergent and successfully compensates trajectory tracking errors caused by unmodelled dynamics and deterministic disturbances. However, it does not take into account the existence of the transient growth error problem and due to this problem cannot be applied to systems with constrained output.

2.2. Bounded-Error Algorithm (BEA) for ILC

Let's consider the BEA method for solving the transient growth error problem.

Given an attainable desired output trajectory and an error bound $\varepsilon = e^{\lambda T} b_y + \delta = \mu + \delta$, ($\delta > 0$) the Bounded-Error Algorithm (BEA) for the implementation of the ILC procedure could be formulated as follows:

i. Set the initial iteration number $l = 0$ and begin the iterative procedure.

ii. Starting from the initial position $\boldsymbol{y}_l(0)$ the system is tracking the desired trajectory under the control $\bar{u}_l(t)$ until

$$\left\|\boldsymbol{y}_l\left(\bar{T}_l\right) - \boldsymbol{y}_d\left(\bar{T}_l\right)\right\| = \varepsilon \tag{7}$$

or the end position $\boldsymbol{y}_l(T)$ is reached. When $t = \bar{T}_l$, $\bar{T}_l \in (0, T]$ the tracking process stops.

iii. After the current tracking performance has finished, the learning controller updates the feed-forward control term according to the following learning update law:

$$\begin{aligned}
&\bar{u}_{l+1}\left(t,\bar{T}_l\right) = (1-\gamma)\bar{u}_l\left(t,\bar{T}_{l-1}\right) + \gamma \boldsymbol{u}_0(t) + \bar{u}_l^*\left(t,\bar{T}_l\right),\\
&\bar{u}_0\left(t,\bar{T}_{-1}\right) \equiv \boldsymbol{u}_0(t),\\
&\bar{u}_l^*\left(t,\bar{T}_l\right) = \begin{cases} L\left(\boldsymbol{y}_l(t),\ t\right)\left(\dot{y}_d(t) - \dot{y}_l(t)\right), & t\in\left[0,\bar{T}_l\right],\ \ \bar{T}_l \in (0,\ T];\\ 0, & \forall t\in\left(\bar{T}_l, T\right]. \end{cases}
\end{aligned} \tag{8}$$

iv. If the output error $\|\delta \boldsymbol{y}_l(t)\|_\infty$ is less than or equal to an acceptable tracking accuracy, then exit from the learning procedure, else set $l = l+1$ and go to step (ii).

The main idea of BEA is that the output trajectory at each iteration must be inside a hyper tube of width 2ε around the desired trajectory i.e.

$$\boldsymbol{y}_l(t) \in \left\{\boldsymbol{y}(t):\ \|\boldsymbol{y}(t) - \boldsymbol{y}_d(t)\| \leqslant \varepsilon,\ \ \forall t\in\left[0,\ \bar{T}_l\right],\ \ \forall \bar{T}_l \in (0,\ T]\right\},\ \ l = 0, 1, \ldots,$$

where $\boldsymbol{y}(t)\ t\in\left[0,\ \bar{T}_l\right]$, $\bar{T}_l \in (0,\ T]$ is the output trajectory and $\varepsilon > e^{\lambda T} b_y$ is a preliminary given error norm bound. The convergence of BEA is proven in [18] by using the following:

Corollary 1 [18]. *If for the system (1) the update law* $\boldsymbol{u}_l$ *(2) is replaced with* $\bar{u}_l$ *(8), then Theorem 1 still holds.*

The proof of Corollary 1 is presented in [18], where, in particular, it is proven that there exists $\bar{k}$:

$$\bar{\rho} \sup_{t\in[0,\bar{T}_l]} e^{-\bar{k}t}\left\|\delta\boldsymbol{u}_l\left(t,\bar{T}_{l-1}\right)\right\| \geqslant \sup_{t\in[\bar{T}_l,T]} e^{-\bar{k}t}\left\|\delta\boldsymbol{u}_l\left(t,\bar{T}_{l-1}\right)\right\|. \tag{9}$$

Our previous research [28] investigates how the BEA parameters influence the convergence rate of the ILC procedure. This research proposes how the BEA parameters should be selected for achieving optimal convergence rate and confirmed the following statement: higher value of BEA parameter ε leads to a faster convergence rate (see Fig. 3).

Thus, the BEA method is convergent and solves the transient growth error problem of ILC. It can be applied to systems with constrained output. However, it does not define how to select the value of δ, respectively the value of ε. The convergence rate depends of the selection of this parameter.

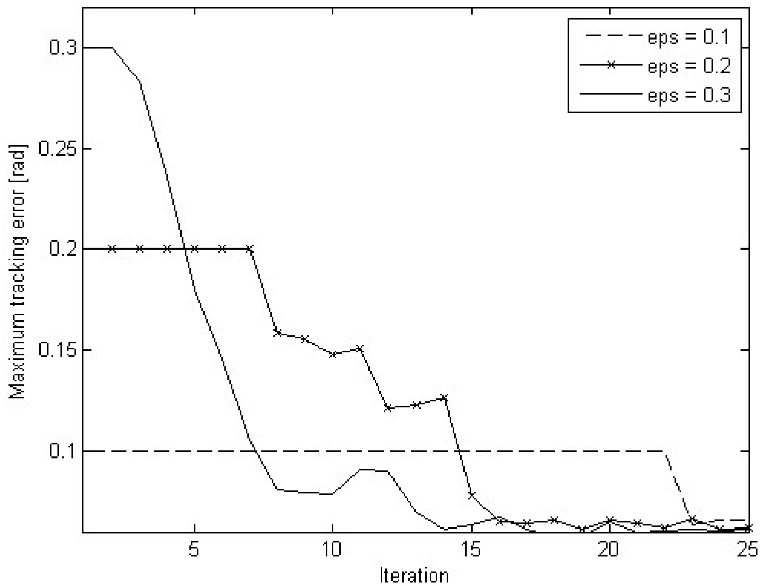

Figure 3: Influence of the value of parameter ε over the BEA convergence rate

2.3. Applying ILC to non-linear system with constrained output

Now, let's consider system with the following output state space constraints for $\boldsymbol{y}(t) \in \mathbb{R}^m$:

$$\boldsymbol{y}(t) = (y_0(t),\ y_1(t),\ \ldots,\ y_m(t)) :\ y_i(t) \in \left[Y_i^{\min},\ Y_i^{\max}\right],\ \ i = 1, 2, \ldots, m. \tag{10}$$

As previously stated, the standard ILC method, defined by update law (2), cannot be directly applied to such system due to the existence of the transient growth problem or when the desired trajectory is planned closely to the constraints (10). In [23] the following solution for applying state-space constrained ILC is proposed. It is based on the BEA method. This solution considers a desired trajectory $\boldsymbol{y}_d = \left(y_1^d, \ldots y_n^d\right) : y_i^d \in \left[Y_i^{\min}, Y_i^{\max}\right]$, $i = 1, \ldots, m$ that satisfies the inequality:

$$\min\left(\min_{t\in[0,T]}\left(Y_i^{\max} - y_i^d(t)\right),\ \min_{t\in[0,T]}\left(y_i^d(t) - Y_i^{\min}\right)\right) > \mu, \tag{11}$$

where μ is the provided accuracy of the ILC method.

From (11) it follows that we can select

$$\delta^* :\ \delta^* = \min\left(\min_{t\in[0,T]}\left(Y_i^{\max} - y_i^d\right),\ \min_{t\in[0,T]}\left(y_i^d - Y_i^{\min}\right)\right) - \mu > 0$$

and $\varepsilon^* = \mu + \delta^*$. Then we can enforce a restriction over the output trajectory at each iteration by applying BEA with learning update law $\bar{u}_l$ from (8). As a result of application of the algorithm in Section 2.2 the entire output trajectory at each iteration $\boldsymbol{y}_l(t)$, $t \in [0, T]$ lies inside a hyper tube of radius ε^* around the desired trajectory $\boldsymbol{y}_d(t)$: $y_i^d(t) \in \left[Y_i^{\min} + \varepsilon^*, Y_i^{\max} - \varepsilon^*\right]$, $t \in [0, T]$. Consequently, this hyper tube lies within the state space constraints (10). It is illustrated in Fig. 4. However, this solution, enforces additional constraints (11) over the set of planned trajectories, as seen in Fig. 4.

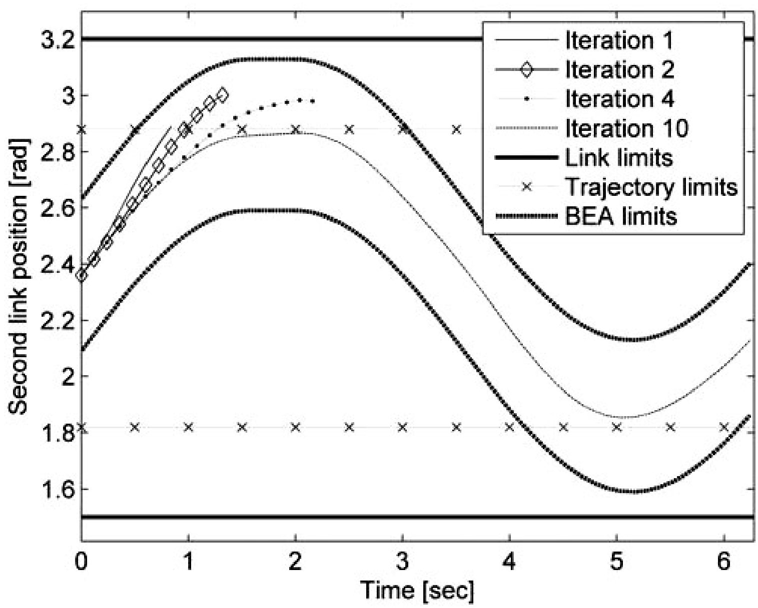

Figure 4: Executed trajectories on different iterations

If we consider the state space constraints in (10) and the inequality (11), then the ILC iteration output trajectories are:

$$\boldsymbol{Y} = \{\boldsymbol{y}(t) : \boldsymbol{y}(t) \in \mathbb{R}^m, \ \|\boldsymbol{y}(t) - \boldsymbol{y}_d(t)\| \leqslant \varepsilon^*, \ \ t \in [0, T], \ i = 1, 2, \ldots, m\}. \quad (12)$$

Obviously, this set $\boldsymbol{Y}$ is dependent on ε^*, which depends on δ^*. δ^* is selected to satisfy (11) based on the desired output trajectory $\boldsymbol{y}_d$. If $\boldsymbol{y}_d$ is planned to pass closely to the output constraints in (10) the set $\boldsymbol{Y}$ in (12) will be highly limited, δ^* defined from (11) will tend to zero and according to [28] this will increase the required number of iterations and will result in an inefficient ILC process. The overall efficiency of this method depends on the selected desired trajectory.

In the following section we present a novel and convergent ILC method for constrained output systems. With this new method the output trajectory at each iteration is no longer constrained within a hyper tube of width $2\varepsilon^*$ around the desired trajectory. The output trajectories will be bounded only by the state space constraints.

3. Constrained Output ILC

The idea of this new method is to constrain the output trajectory at each iteration l, so that the state space constraints cannot be violated during the ILC procedure.

Let's consider state space system (1) constrained by (10) and the following subset of the attainable desired output trajectories defined by the ILC tracking accuracy μ:

$$\begin{aligned} \boldsymbol{Y}_d = \Big\{ & \boldsymbol{y}_d(t) : \boldsymbol{y}_d(t) \in \mathbb{R}^m,\ \boldsymbol{y}_d(t) = \left(y_1^d(t),\ y_2^d(t),\ \ldots,\ y_m^d(t)\right), \\ & y_i^d(t) \in \left(Y_i^{\min} + \mu,\ Y_i^{\max} - \mu\right),\ \ t \in [0,\ T],\ i = 1, 2, \ldots, m \Big\}. \end{aligned} \tag{13}$$

Constrained Output ILC (COILC) method

For the system defined in (1) with the state space constraints defined in (10) we can apply the following ILC procedure for any attainable desired trajectory $\boldsymbol{y}_d \in \boldsymbol{Y}_d$ in (13):

a. Set the initial iteration number $l = 0$ and begin the iterative procedure.

b. Starting from the initial position $\boldsymbol{y}_l(0)$ the system is tracking the desired trajectory under the control $\overline{\overline{\boldsymbol{u}}}_l(t)$ until the first moment S_l for which there exists $j : 1 \leqslant j \leqslant m$ and either $y_j^l(S_l) = Y_j^{\min}$ or $y_j^l(S_l) = Y_j^{\max}$ or the end position $\boldsymbol{y}_l(T)$ is reached. When $t = S_l$, $S_l \in (0, T]$ the tracking process stops.

c. After the current tracking performance has finished, the learning controller updates the feed-forward control term according to the following learning update law:

$$\begin{aligned} & \overline{\overline{\boldsymbol{u}}}_{l+1}(t, S_l) = (1-\gamma)\overline{\overline{\boldsymbol{u}}}_l(t, S_{l-1}) + \gamma \boldsymbol{u}_0(t) + \overline{\overline{\boldsymbol{u}}}_l^*(t, S_l), \\ & \overline{\overline{\boldsymbol{u}}}_0(t, S_{-1}) \equiv \boldsymbol{u}_0(t), \\ & \overline{\overline{\boldsymbol{u}}}_l^*(t, S_l) = \begin{cases} L\left(\boldsymbol{y}_l(t), t\right)\left(\dot{\boldsymbol{y}}_d(t) - \dot{\boldsymbol{y}}_l(t)\right), & t \in [0,\ S_l],\quad S_l \in (0,\ T]; \\ 0, & \forall t \in (S_l,\ T]. \end{cases} \end{aligned} \tag{14}$$

d. If the output error $\|\delta \boldsymbol{y}_l(t)\|_\infty$ is less than or equal to an acceptable tracking accuracy, then exit from the learning procedure, else set $l = l + 1$ and go to step (b).

Corollary 2 *For the system defined in (1) with the state space constraints defined in (10) and for any attainable desired trajectory* $\boldsymbol{y}_d \in \boldsymbol{Y}_d$ *from (13) if the update law* $\boldsymbol{u}_l$ *(2) is replaced with* $\bar{\bar{\boldsymbol{u}}}_l$ *(14), then Theorem 1 still holds.*

Proof. The main idea of the proof of Corollary 2 is to show that a sequence

$$\left\{\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\|_{\lambda^*}\right\}, \quad l \in \{1, \ldots, \infty\}, \quad t \in [0, T] \tag{15}$$

exists and

$$\left\|\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\|_{\lambda^*} \leqslant \bar{\rho}\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\|_{\lambda^*} + \hat{\varepsilon}, \quad 0 < \bar{\rho} < 1 \tag{16}$$

and therefore Lemma 1 can be applied. □

In the case when $S_l \in (0, T)$ from (14) and (5) for $\left\|\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\|_\lambda$ and $\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\|_\lambda$ it follows that:

$$\sup_{t\in[0,T]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\| = \max\begin{pmatrix} \sup\limits_{t\in[0,S_l]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\| \\ \sup\limits_{t\in[S_l,T]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\| \end{pmatrix} \tag{17}$$

and

$$\sup_{t\in[0,T]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\| = \max\begin{pmatrix} \sup\limits_{t\in[0,S_l]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\| \\ \sup\limits_{t\in[S_l,T]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_l)\right\| \end{pmatrix}. \tag{18}$$

From (18), both possible cases are considered for $\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)$, when

$$\sup_{t\in[0,T]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\| = \sup_{t\in[S_l,T]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\| \tag{19}$$

and when

$$\sup_{t\in[0,T]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\| = \sup_{t\in[0,S_l]} e^{-\lambda t}\left\|\delta\bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\|. \tag{20}$$

Let's apply an ε-hyper tube around $\boldsymbol{y}_d(t)$ in (13) for $\bar{\bar{\boldsymbol{u}}}_l$ in (14) and obtain $\bar{\bar{T}}_l$: $\left\|\boldsymbol{y}_l\left(\bar{\bar{T}}_l\right) - \boldsymbol{y}_d\left(\bar{\bar{T}}_l\right)\right\| = \varepsilon$ although the robot motion doesn't stop when $\left\|\boldsymbol{y}_l\left(\bar{\bar{T}}_l\right) - \boldsymbol{y}_d\left(\bar{\bar{T}}_l\right)\right\| = \varepsilon$, because from (13) for any desired output trajectory $\boldsymbol{y}_d \in \boldsymbol{Y}_d$ there exists $\delta > 0$: $\boldsymbol{y}_d \in \left[Y_i^{\min} + \varepsilon, Y_i^{\max} - \varepsilon\right]$ and $\varepsilon = \mu + \delta > \mu$, and therefore $\bar{\bar{T}}_l \leqslant S_l$ (see Fig. 5b). Thus, we can apply Corollary 1 for $\bar{\bar{\boldsymbol{u}}}_l$ in (14),

$\varepsilon = \mu + \delta$ and $\bar{\bar{T}}_l \leqslant S_l$ and there exists $\bar{\bar{k}} > 0$ so that inequality (9) yields $\bar{\rho} \sup\limits_{t\in[0,\bar{\bar{T}}_l]} e^{-\bar{\bar{k}}t} \left\|\delta \boldsymbol{u}_l\left(t, \bar{\bar{T}}_{l-1}\right)\right\| \geqslant \sup\limits_{t\in[\bar{\bar{T}}_l,T]} e^{-\bar{\bar{k}}t} \left\|\delta \boldsymbol{u}_l\left(t, \bar{\bar{T}}_{l-1}\right)\right\|$ and taking into account that $\left[0, \bar{\bar{T}}_l\right] \subset [0, S_l]$ and $[\bar{\bar{T}}_l T] \supset [S_l T]$, it follows:

$$\bar{\rho} \sup_{t\in[0,S_l]} e^{-\bar{\bar{k}}t} \left\|\delta \bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\| \geqslant \sup_{t\in[S_l,T]} e^{-\bar{\bar{k}}t} \left\|\delta \bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\|. \tag{21}$$

Noticing that $0 < \bar{\rho} < 1$ from (17) and (21) follows that:

$$\sup_{t\in[0,T]} e^{-\bar{\bar{k}}t} \left\|\delta \bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\| = \sup_{t\in[0,S_l]} e^{-\bar{\bar{k}}t} \left\|\delta \bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\|. \tag{22}$$

Taking into account that $\hat{\varepsilon} \geqslant 0$ from inequality (21) combined with (19) and (22) by using (5) yields (15) and (16) for $\lambda^* = \max\left(\bar{\bar{k}}, \lambda\right)$.

For the second case when inequality (20) holds, Theorem 1 can be directly applied for $T \geqslant S_l : t \in [0, S_l]$ because update laws (2) and (14) are equivalent. Thus, from (4), using (5) and taking into account that $\lambda(S_l) \leqslant \lambda(T)$ [12], we have:

$$\sup_{t\in[0,S_l]} e^{-\lambda t} \left\|\delta \bar{\bar{\boldsymbol{u}}}_{l+1}(t, S_l)\right\| \leqslant \bar{\rho} \sup_{t\in[0,S_l]} e^{-\lambda t} \left\|\delta \bar{\bar{\boldsymbol{u}}}_l(t, S_{l-1})\right\| + \hat{\varepsilon}. \tag{23}$$

Combining equalities (20) and (22) with inequality (23), for $t \in [0, T]$ and $\lambda^* = \max\left(\bar{\bar{k}}, \lambda\right)$ yields (15) and the inequality (16).

Now assuming that $l \to \infty$ we are in a position to apply Lemma 1 for (15) and (16) and obtain the inequalities (6) for the new update law (14). Consequently Theorem 1 still holds.

It follows, that the COILC method is convergent and it can be applied for constrained output systems. Furthermore, for such systems it also solves the transient growth error problem without requiring knowledge for additional parameters other than the constraints of the system. COILC is supposed to also have a better convergence rate. The actual performance of the proposed COILC method is evaluated in the next section through a computer simulation.

4. Simulation results

This section presents the numerical experiments with the proposed Constrained Output ILC. The objective of the experiments is to evaluate the convergence rate of the method by computer simulation. The simulation setup is similar to those reported in [23]. Both methods are compared. For the computer simulation the dynamics model of the six-linked Puma 560 is used. Two different sets of model parameters are used for realistic simulation. The first set of parameters is for the model of the virtual robotic arm. The second set is used as an imprecise estimation of the parameters of the virtual robot. This allows taking into consideration the errors caused by unmodelled dynamics.

Similarly, to [23], a numerical experiment with 50 iterations was executed. The accuracy μ of ILC is preset to 0.065 rad for this simulation. The constant δ is set to 0.035 rad respectively, so that $\varepsilon = \mu + \delta = 0.10$ rad.

Constraints of the simulated robot arm and the desired trajectory $\boldsymbol{y}_d$, according to inequality (11), are presented in Table 1. For the BEA method the $\boldsymbol{y}_d$ limits are chosen respectively to the described hyper tube approach in Section 2. For the COILC method all of the maximum position limits are the link limits. It is supposed that those limits will prevent the simulated robot from hitting its end-point positions and accounts the accuracy of the method.

Table 1: Robotic manipulator constraints

Link	1		2		3		4		5		6	
	min	max	min	max	min	max	min	max	min	max	min	max
Torque limits [Nm]	−100	100	−180	180	−90	90	−25	25	−25	25	−25	25
Velocity limits [rad/s]	−1	1	−1	1	−1.5	1.5	−2	2	−2	2	−2	2
Position limits [rad]	$-\pi$	π	1.75	2.99	0	π	1.5	3.2	0	π	$-\pi$	π
BEA $\boldsymbol{y}_d$ limits [rad]	−3.04	3.04	1.85	2.89	0.10	3.04	1.6	3.1	0.1	3.04	−3.04	3.04
COILC $\boldsymbol{y}_d$ limits [rad]	$-\pi$	π	1.75	2.99	0	π	1.5	3.2	0	π	$-\pi$	π

The simulated ILC procedure feedback term incorporates computed torque control. For both BEA and COILC in Fig. 5 are shown the desired trajectory for the second robot link with the corresponding joint angle limits (marked as "Link limits" in Fig. 5), the executed output trajectories for selected iterations and the forced limits over the output trajectories (marked as "BEA limits" in Fig. 5). As seen, during the first iterations 1 and 2 both methods abort the iterations before the end of the execution time. However, the COILC method executes during the whole time the 4-th iteration. Another difference is that the ε parameter of BEA accounts the error of the all 6 links, so the tracking process is stopped even before reaching the enforced BEA limits for the second link. During COILC the second link first reaches the stop limits and then the tracking process is interrupted. Both methods prevent successfully the violation of the state space constraints.

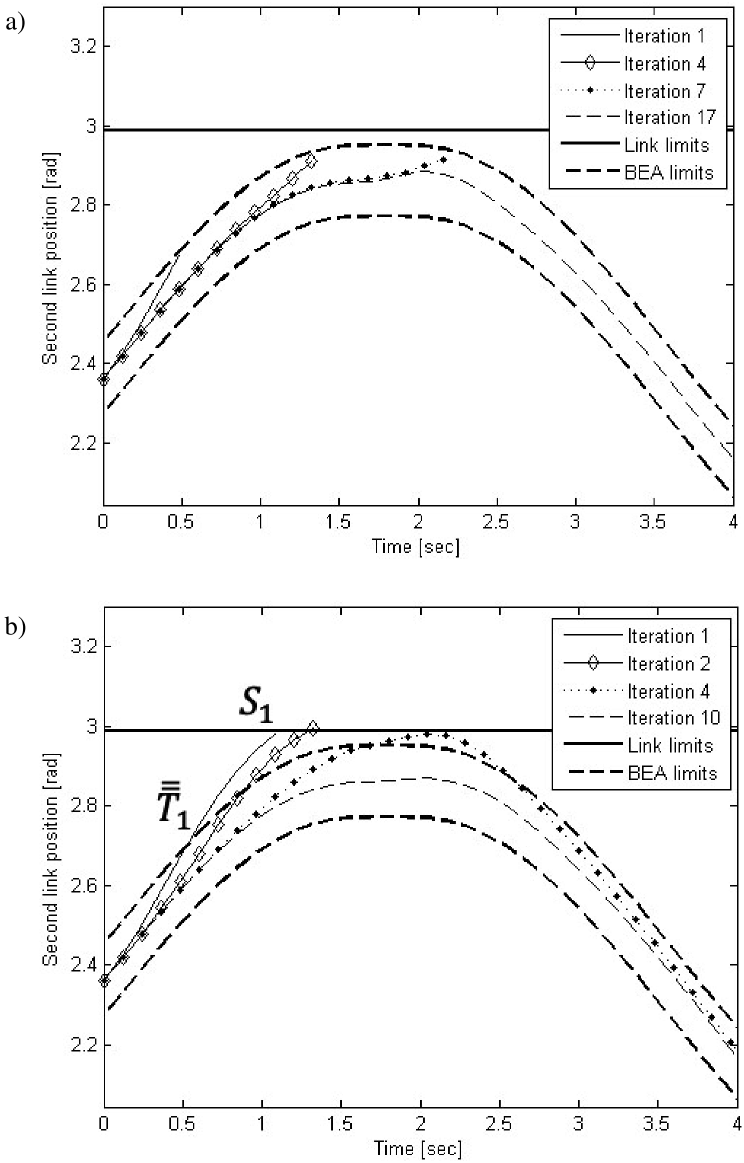

Figure 5: a) BEA tracked trajectories of second link; b) COILC tracked trajectories of second link

The following Fig. 6. shows the time and error comparison of the execution of both BEA and COILC. The maximum tracking error of iteration l is measured as $\sup\limits_{t\in[0,T]} \|\mathbf{y}_d(t) - \mathbf{y}_l(t)\|$. As seen, the COILC method converges faster than BEA. It reaches optimal solution on iteration 14-th while BEA needs 4 more iterations.

This is a result of longer initial iterations (Fig. 7). And for the test case, COILC gives 25% faster convergence than BEA. The application of this new method was thoroughly tested through this simulation. Furthermore, the state space constrained ILC problem was resolved in a computationally effective way. The overall performance of the COILC method is better than the reported approach in [23].

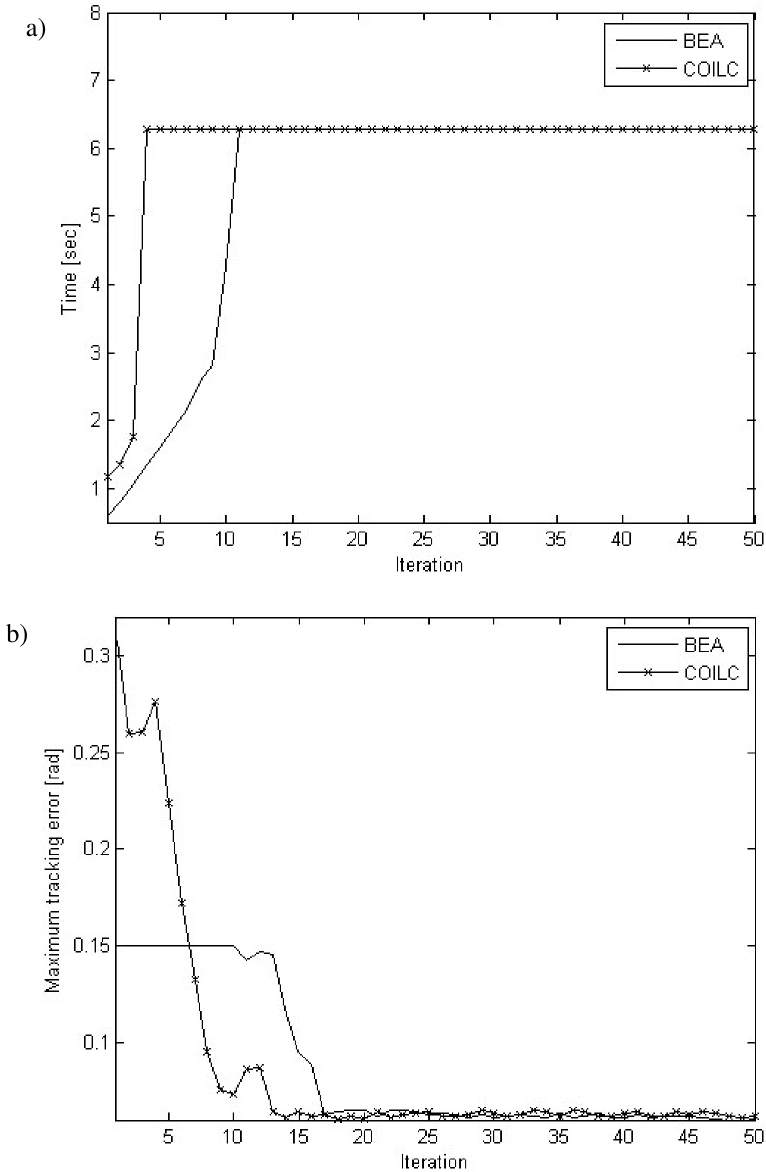

Figure 6: a) BEA vs COILC time comparison; b) BEA vs COILC maximum tracking error comparison

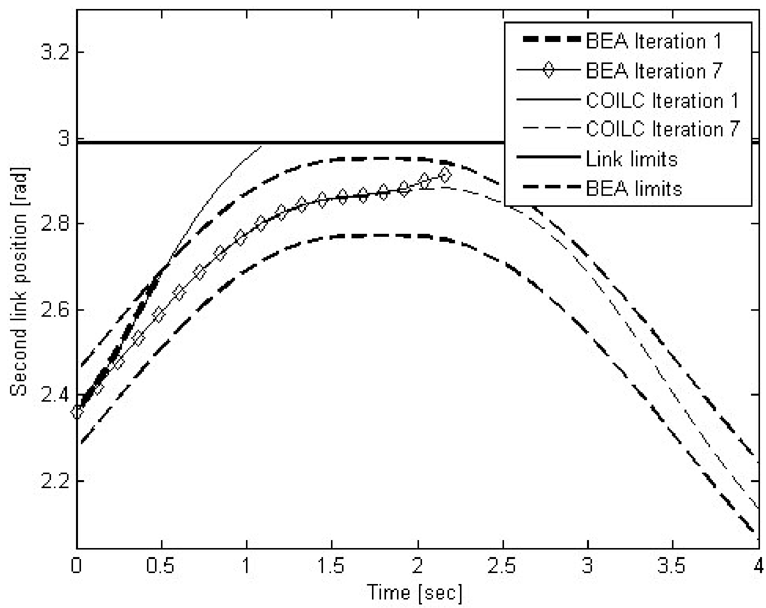

Figure 7: Comparison of BEA and COILC trajectory tracking process

5. Conclusion

This paper presents a novel and computationally efficient general solution of the state space constrained ILC problem – the Constrained Output ILC method (COILC). The convergence of COILC is proven for non-linear state space constrained systems. It extends the application of the Bounded Error Algorithm (BEA) for ILC in real-life work operations. Initially, BEA has been designed to solve the transient error growth problem. The proposed COILC method modifies the learning update law of BEA and introduces a different stop condition. This results in more computationally effective method. COILC is compared through the computer simulation to the previously studied approach for applying ILC to state space constrained non-linear systems. The results allow concluding that COILC can be successfully applied to such systems. The experiments implicate that this method is with a better overall performance and that it is robust and convergent. Furthermore, the COILC method solves the transient growth problem and prevents violation of the state space constraints. Also, since this method only monitors the current output during each iteration and aborts it if simple inequalities are in force it can be concluded to be the most computationally effective method for applying ILC to nonlinear state space constrained systems. COILC is expected to be of a considerable interest for robotic applications.

References

[1] M. Uchiyama: Formulation of high-speed motion pattern of a mechanical arm by trial, *Trans. SICE (Soc. Instrum. Contr. Eng.)*, **14**(6) (1978), 706–712.

[2] G. Murray: *Learning control of actuators in control systems*, United States Patent US3555252 A, 1971.

[3] S. Arimoto, S. Kawamura and F. Miyazaki: Iterative learning control for robot systems, *Proceedings of IECON*, Tokyo, Japan, 1984.

[4] S. Arimoto, S. Kawamura and F. Miyazaki: Bettering operation of robots by learning, *Journal of Robotic Systems*, **1**(2) (1984), 123–140.

[5] G. Casalino and G. Bartolini: A learning procedure for the control of movements of robotic manipulators, *IASTED Symposium on Robotics and Automation*, Amsterdam, The Netherlands, (1984), 108–111.

[6] J. Craig: Adaptive control of manipulators through repeated trials, *Proc. of ACC*, San Diego, CA, (1984).

[7] K.L. Moore: Iterative learning control – an expository overview., *Applied and Computational Control, Signals, and Circuits*, Birkhäuser Boston, (1999), 151–214.

[8] J.-X. Xu and Y. Tan: *Linear and Nonlinear Iterative Learning Control*, Springer-Verlag Berlin Heidelberg, 2003.

[9] M. Norrlof: *Iterative Learning Control: Analysis, Design, and Experiments, Linkoping Studies in Science and Technology*, PhD Dissertations, Linkoping, Sweden, 2000.

[10] R. Longman: Iterative learning control and repetitive control for engineering practice, *International Journal of Control*, **73**(10) (2000), 930–954.

[11] D. Bristow and J. Singler: Analysis of Transient Growth in Iterative Learning Control Using Pseudospectra, *Symposium on Learning Control*, Shanghai, China, 2009.

[12] D. Heizinger, B. Fenwick, B. Paden and F. Miyazaki: Robust Learning Control, *Proc. of 28th Conference on Decision and Control*, Tampa, FL, (1989), 436–440.

[13] R. Longman and Y. Huang: The Phenomenon of Apparent Convergence Followed by Divergence in Learning and Repetitive Control, *Intelligent Automation and Soft Computing*, **8**(2) (2002), 107–128.

[14] K. Yovchev, K. Delchev and E. Krastev: Computer Simulation of Bounded Error Algorithm for Iterative Learning Control, *Advances in Robot Design and Intelligent Control. RAAD 2016. Advances in Intelligent Systems and Computing*, **540**, A. Rodić and T. Borangiu, Eds., Springer, Cham, 2017.

[15] D.A. Bristow and J.R. Singler: Robustness Analysis of Slow Learning in Iterative Learning Control Systems, *2011 American Control Conference on O'Farrell Street*, San Francisco, CA, USA, (2011).

[16] K.-H. Park and Z. Bien: A Study On Iterative Learning Control with Adjustment of Learning Interval for Monotone Convergence in the Sense of Sup-Norm, *Asian Journal of Control*, **4**(1), (2002), 111–118.

[17] K. Delchev: Iterative learning control for robotic manipulators: A bounded-error algorithm. *International Journal of Adaptive Control and Signal Processing*, **28**(12), (2013).

[18] K. Delchev: Iterative Learning Control for Nonlinear Systems: A Bounded-Error Algorithm, *Asian Journal of Control*, **15**(3) (2013), 1–8.

[19] S. Mishra, U. Topcu and M. Tomizuka: Optimization-based constrained iterative learning control, *IEEE Trans. Control Syst. Technol.*, **19**(6) (2011), 1613–1621.

[20] B. Chu and D. Owens: Iterative learning control for constrained linear systems using projection method, *International Workshop on Human Adaptive Mechatronics*, Loughborough, GB, (2010).

[21] Y. Tan and J.X. Xu: On iterative learning control for nonlinear time-varying systems with input saturation, *Symposium on Learning Control at IEEE CDC 2009*, Shanghai, (2009).

[22] A. Schöllig and R. D'Andrea: Optimization-Based Iterative Learning Control for Trajectory Tracking, *European Control Conference*, Budapest, Hungary, (2009), 1505–1510.

[23] K. Yovchev, K. Delchev and E. Krastev: State Space Constrained Iterative Learning Control for Robotic Manipulators, *Asian Journal of Control*, **20**(1) (2018), 1–6.

[24] G. Oriolo: An Iterative Learning Controller for Nonholonomic Mobile Robots, *The International Journal of Robotics Research*, **17**(9) (1998), 954–970.

[25] M. Guth, T. Seel and J. Raisch: Iterative Learning Control with variable pass length applied to trajectory tracking on a crane with output constraints, *Proc. of 52nd IEEE Conf. on Decision and Control*, Florence, Italy, (2013).

[26] G. Sebastian, Y. Tan, D. Oetomo and I. Mareels: Feedback-based iterative learning design and synthesis with output constraints for robotic manipulators, *IEEE Control Systems Letters*, **2**(3) (2018), 513–518.

[27] M.R. Zamani, Z. Rahmani and B. Rezaie: A novel model predictive control strategy for constrained and unconstrained systems in presence of disturbance, *IMA Journal of Mathematical Control and Information*, (2019).

[28] K. Yovchev: Finding the Optimal Parameters for Robotic Manipulator Applications of the Bounded Error Algorithm for Iterative Learning Control, *Journal of Theoretical and Applied Mechanics*, **47**(4) (2017), 3–11.

[29] D. Heizinger, B. Fenwick, B. Paden and F. Miyazaki: Stability of Learning Control with Disturbances and Uncertain Initial Condition, *IEEE Trans. Autom. Control*, **37**(1) (1992), 110–114.