Proceedings of the 35th Chinese Control Conference
July 27-29, 2016, Chengdu, China

Consolidated Iterative Learning Control and A Preliminary Case Study

Yang-Guang Cai[1], Zi-Bo Wei[1], Quan Quan[1,2]
[1]School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, P. R. China
[2]Laboratoire International Associé, Beihang University, Beijing 100191, P. R. China
E-mail: yangguang.cai@buaa.edu.cn; whisper@buaa.edu.cn; qq_buaa@buaa.edu.cn

Abstract: In this paper, a new concept of iterative learning control (ILC), namely consolidated ILC (CILC), is introduced. An iterative learning control system is subject to data interrupt when the system output is restricted within a certain area. Data interrupt means that the system output is interrupted at a certain moment due to output restrictions. The output data from this moment to the end of the trail duration cannot be obtained, which is the biggest difference from the classical ILC problems. The ILC problems with data interrupt caused by output restrictions are called CILC problems. The idea to solve the CILC problems is to obtain the complete output data gradually as the number of iterations increases. First of all, three CILC problems in practice are presented as well as the intermittent ILC problems and the relationship between them. The CILC problems are formulated later and the uniform restriction is given. A specific linear case is also studied. According to the simulation results, it is possible to achieve the convergence of the specific system as well as to control the convergence rate by using a proper algorithm.

Key Words: Consolidated iterative learning control, Output restriction, Data interrupt, Convergence, Convergence rate

1 Introduction

Iterative learning control (ILC) is an effective technique for systems that execute the same task multiple times. It can realize a full path track in finite trail durations. ILC uses the bias between the complete output data and the desired path to correct the undesired control signal and generate a new complete control signal. The new control signal is used as the input of the new trail and the performance of the system can be improved during the continuous correction. There have been many researches about ILC [1]-[5]. It has also been successfully applied to industrial robots [6]-[7], autonomous vehicles [8], multicopters [9], etc. According to the feature of ILC, complete data of the system, which includes output data and input data, is required. However, in practice, the condition mentioned before is not always satisfied. The system is subject to data interrupt when the system output is restricted because of safety.

1.1 Background

In this section, three examples are taken to present the situation of data interrupt under the output restrictions.

(i) Maneuver of multicopters passing through a window

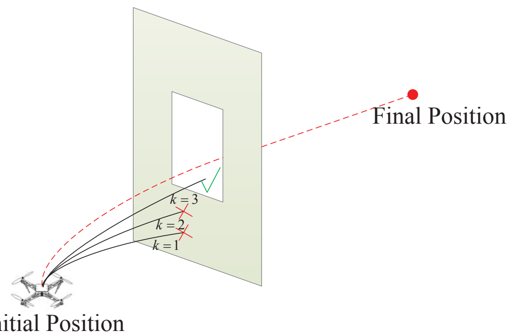


Fig. 1: Maneuver of a multicopter passing through a window

A remote pilot manipulates a multicopter to pass through a window, as shown in Fig. 1, k represents the number of iterations. The dotted line represents the desired trajectory for the multicopter. The multicopter needs to avoid the wall for safety. However, it is highly possible for the multicopter to hit the wall since the window is small compared with the wall. Therefore, when the multicopter is going to hit the wall, the remote pilot needs to manipulate the multicopter to stop flying forward, return to the initial position and do a new trail. After several iterations, the multicopter can finally pass through the window, as shown by the black solid lines in Fig. 1. Before that, the multicopter does not have a complete trajectory, namely a complete output data. The output is restricted by the window.

(ii) Reversing of cars

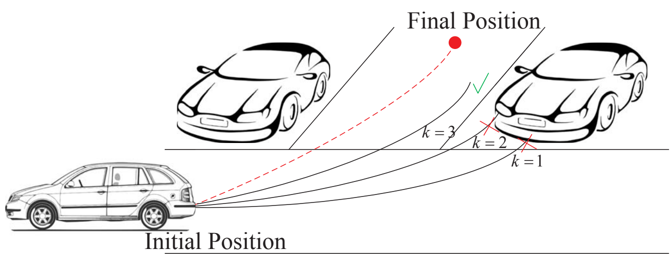


Fig. 2: Reversing of cars

In a similar way, the trajectory of the car is restricted by two other cars next to the parking spot. As shown in Fig. 2, the dotted line represents the desired trajectory for the car. The car needs to avoid hitting other cars during the process of reversing. When it is going to hit other cars, the driver needs to manipulate the car to stop, return to the initial position and do a new trail. The trajectory of the car is not complete. The complete output data cannot be obtained in the first few trials.

(iii) Aerial Refueling

An aerial refueling process requires an accurate trajectory

This work was supported by National Natural Science Foundation of China (No. 61473012,51375462).

tracking. However, it is highly impossible to have an accurate docking for every trail. A reasonable safety area around the drogue needs to be established. When the receiver aircraft flies out of the area, the receiver aircraft should give up the current trail [10]. A missing and capture criterion is presented in the paper [11]. As shown in Fig. 3, the dotted lines represent the boundary of the safety area, which could be the position, the velocity, the angular, etc. In a word, the trajectory of the receiver airplane is restricted in a certain area.

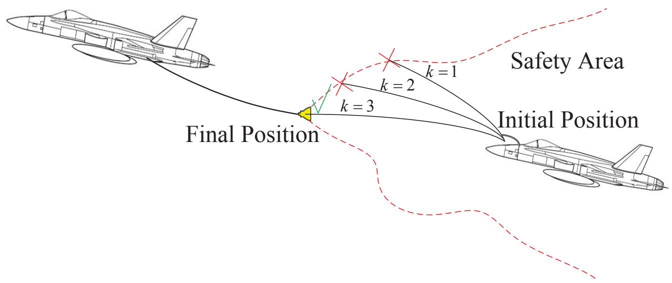


Fig. 3: Aerial refueling

The three examples above show that the output restrictions for the ILC systems are a common phenomenon in practice. Compared with the normal tracking problems, the output of the systems is restricted within a certain area. A complete output data cannot be obtained at the beginning and the ILC algorithms cannot be used directly. Therefore, it is necessary to study the ILC systems with data interrupt caused by output restrictions.

1.2 Related work

Recently, ILC systems with data dropout have been studied from a number of different perspectives, mostly concentrating on the convergence and the stability of the system in a networked control system setting (NCS) [12]-[16]. Data dropout is one of the ILC problems with incomplete data. It is shown that the convergence of the system can be achieved under some given conditions when the output data dropout occurs. In the meantime, the convergence rate is reduced by random data dropout. Different methods are used to design robust iterative learning controllers. For an ILC system with data dropout, the data is dropped with a certain probability. This class of problems can be summarized as the intermittent ILC. A random moment is considered when data dropout occurs. Any two moments are independent when it comes to the probability of data dropout. However, the convergence performance and convergence rate of one moment may be affected by other moments. The other is the CILC problems. Compared with the data dropout, the output data of the CILC systems is interrupted at a certain moment, and the data from this moment to the end of the trail duration cannot be obtained. The concept of CILC is proposed specially in order to analyze this class of problems. The relationship among the classical ILC problems, the intermittent ILC problems and the CILC problems is showed in Fig. 4.

1.3 Layout of this paper

The paper is organized as follows. In the next section, the CILC problems are formulated and a uniform restriction is p-

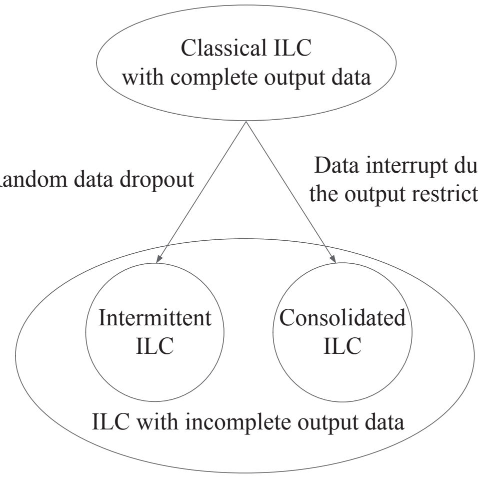


Fig. 4: Relationship among the classical ILC problems, the intermittent ILC problems and the CILC problems

resented in order to simplify the problem. After that, a linear case is studied in Section 3. A couple of key factors including the learning algorithms and the compensation algorithms are analyzed. Then, numerical experiments are given to illustrate preliminary the feasibility of the proposed methods. Based on the simulation results, some discussions are presented in Section 4. Finally, some conclusions are given in Section 5.

2 Problem Formulation

2.1 Classical ILC problem

Consider the following nonlinear system

$$\begin{aligned} \dot{x}_k(t) &= f(x_k(t), u_k(t), t) \\ y_k(t) &= g(x_k(t), u_k(t), t) \end{aligned} \tag{1}$$

where $x_k : [0,T] \to \mathbb{R}^n$, $u_k : [0,T] \to \mathbb{R}^m$, and $y_k : [0,T] \to \mathbb{R}^l$ are state, input and output variables, $f : \mathbb{R}^n \times \mathbb{R}^m \times [0,T] \to \mathbb{R}^n$ and $g : \mathbb{R}^n \times \mathbb{R}^m \times [0,T] \to \mathbb{R}^l$ are nonlinear functions. The subscript $k = 0, 1, \cdots$ is used to denote the iteration number and $t \in [0,T]$ is used to denote the continuous time. The system is operated repeatedly in the iteration domain with a desired trajectory $y_d(t)$. The system output error in k-th iteration is denoted by $e_k(t) = y_d(t) - y_k(t)$. Then, for simplicity, the learning algorithm is expressed as

$$u_{k+1}(t) = L(u_k(t), e_k(t)) \tag{2}$$

The objective of an ILC problem is to make $e_k(t) \to 0, \forall t \in [0,T]$ as $k \to \infty$. In this case, the system is convergent. The convergence of the system is the most important issue for the ILC problems.

2.2 CILC problem

The output restrictions are taken into account for the system (1). In this case, the output data is interrupted at a certain moment. This interruption moment is denoted by T_k in the k-th iteration and the restriction zone is denoted by $\Sigma \subset \mathbb{R}^l$. The initial state of the system is assumed to be inside of Σ. The value of the interruption moment T_k is determined as follows

$$
\begin{array}{c} T_k = \sup t_k \\ s.t.\ t_k = \{s_k \mid y_k(t) \in \Sigma, 0 \le t \le s_k \text{ and } s_k \le T\} \end{array} \tag{3}
$$

The system (1) is further expressed as

$$
\begin{array}{c} \dot{x}_k(t) = f(x_k(t), u_k(t), t) \\ y_k(t) = g(x_k(t), u_k(t), t) \end{array} \tag{4}
$$

where $x_k : [0, T_k] \to \mathbb{R}^n$, $u_k : [0, T_k] \to \mathbb{R}^m$, and $y_k : [0, T_k] \to \mathbb{R}^l$ are state, input and output variables, $f : \mathbb{R}^n \times \mathbb{R}^m \times [0, T_k] \to \mathbb{R}^n$ and $g : \mathbb{R}^n \times \mathbb{R}^m \times [0, T_k] \to \mathbb{R}^l$ are nonlinear functions. Here, the output data only exists in $[0, T_k]$. For simplicity, the algorithm for the CILC problems is expressed as

$$
u_{k+1}(t) = \hat{L}(u_k(t), e_k(t)) \tag{5}
$$

where $e_k : [0, T_k] \to \mathbb{R}^l$ and $u_{k+1} : [0, T_{k+1}] \to \mathbb{R}^m$. The objective of a CILC problem is to make $T_k \to T$ and $e_k(t) \to 0, \forall t \in [0, T_k]$ as $k \to \infty$.

The reason that this class of problems is called consolidated iterative learning control is that the system is consolidated trail by trail before the complete output data is obtained. For the interruption moment T_k in k-th iteration, there exists T_{k+m} in $(k+m)$-th $(m > 0)$ iteration such that $T_{k+m} > T_k$. In this case, the output in $[0, T_k]$ is consolidated and the output data in $[T_k, T_{k+m}]$ is obtained. The objective is to make T_k converge to T trail by trail as $k \to \infty$.

2.3 Uniform restriction

For the CILC problems, the restriction zone Σ does not have a fixed form. This could bring much difficulty to study the convergence of the system. In this paper, all types of restriction forms are unified into one form. The uniform restriction depends on the restriction degree denoted by σ. Given $y_d(t)$ and Σ, the value of σ can be decided as follows

$$
\sigma = \inf\{\|y_\Sigma(t) - y_d(t)\| \mid y_\Sigma(t) \in \mathbb{R}^l \text{ and } y_\Sigma(t) \notin \Sigma\} \tag{6}
$$

where $\|\cdot\|$ is the norm operator. After the restriction degree is determined, the output of the system $y_k(t)$ should satisfy the following equation

$$
\|y_k(t) - y_d(t)\| < \sigma \tag{7}
$$

Consider the multicopter example mentioned in Section 1. First, it is assumed that the multicopter flies through the window in one plane that includes the desired trajectory, as shown in Fig. 5 (a). In this case, the output restrictions are two radials. As shown in Fig. 5 (b), the closest positions between the desired trajectory and two radials are two endpoints P_1 and P_2. The intersection of P_1P_2 and $y_d(t)$ is denoted by P. The length of P_1P and P_2P are denoted by σ_1 and σ_2, separately. It is assumed that $\sigma_2 > \sigma_1$. Then the restriction degree in this example is $\sigma = \min\{\sigma_1, \sigma_2\} = \sigma_1$ and the uniform restriction zone is the space between two dotted lines L_1 and L_2.

Apparently, the uniform restriction is much tighter than the origin restriction. Moreover, the restriction degree σ is one of the factors that can influence the convergence performance and convergence rate. If the convergence of the system can be achieved, the convergence rate increases as the value of σ increases.

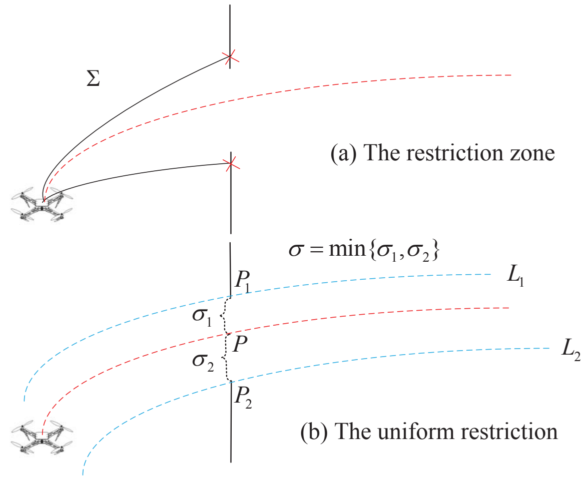


Fig. 5: The plane that includes the desired trajectory

In the next section, a linear case is presented in order to give a more concrete idea about the study of the convergence of the CILC systems and several algorithms are proposed. Simulations are made to verify the feasibility of these algorithms.

3 A preliminary case study

3.1 Research example

In order to simplify the study, a linear discrete-time SISO system is taken as a research example. The form is expressed as

$$
\begin{array}{rcl} x_k(t+1) & = & Ax_k(t) + Bu_k(t) \\ y_k(t) & = & Cx_k(t) \end{array} \tag{8}
$$

where $x_k(t) \in \mathbb{R}^n$, $u_k(t) \in \mathbb{R}$, $y_k(t) \in \mathbb{R}$ and $k = 0, 1, \cdots$ have the same meanings as that of the system (1). The time $t \in \{0, 1, \cdots, T\}$ represents the discrete time index in this case. $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times 1}$, and $C \in \mathbb{R}^{1 \times n}$ are the constant matrices describing the system. If CB is full rank, the desired trajectory $y_d(t)$ is realizable with a unique control input $u_d(t)$. That means the following equations are satisfied

$$
\begin{array}{rcl} x_d(t+1) & = & Ax_d(t) + Bu_d(t) \\ y_d(t) & = & Cx_d(t) \end{array} \tag{9}
$$

where $x_d(t)$ is the desired state. Basic assumptions of this system are: (i) every trail ends in a fixed time of duration; (ii) $y_d(t)$ is iteration invariant; and (iii) every iteration begins at an identical initial condition, namely $x_k(0) = x_0$ for all k [12].

The output restriction $y_d(t) \pm \sigma$ is taken into account. It should be emphasized that T_k in this case represents the discrete interruption time index that satisfies $|e(T_k)| < \sigma < |e(T_k + 1)|$. The study concentrates on the convergence performance and convergence rate. The key factors are given in the following.

3.2 Key factors

3.2.1 The learning algorithm (LA)

A learning algorithm is essentially a process to correct the undesired control signal. There are many effective learning algorithms that can be used. The most widely-used one is the PID-type learning algorithm, which is expressed as

$$u_{k+1}(t) = u_k(t) + \Gamma_p e_k(t) + \Gamma_i \int_0^t e_k(\tau) d\tau + \Gamma_d \dot{e}_k(t) \quad (10)$$

where Γ_p, Γ_i and Γ_d are learning gains. The preliminary study in this paper is based on the P-type and D-type learning algorithms. For a discrete-time system, their forms are given as follows

$$\begin{array}{rcl} \text{P-type: } u_{k+1}(t) & = & u_k(t) + \Gamma_p e_k(t+1) \\ \text{D-type: } u_{k+1}(t) & = & u_k(t) + \Gamma_d(e_k(t+1) - e_k(t)) \end{array} \quad (11)$$

The convergence of the classical ILC systems can be achieved by using these two learning algorithms under the same conditions [2].

3.2.2 The compensation algorithm (CA)

As the complete output data is required, the unknown data needs to be compensated. Here, the compensation algorithm is proposed. As there is no data in $[T_k, T]$, the main idea of the compensation in every iteration for the unknown data is essentially a process to set initial input data. Normally, the initial input for the system (1) is $u_0(t) = 0, t \in [0, T]$. The compensation algorithms need to ensure the convergence of the system firstly and then improve the convergence rate.

The new algorithms for the CILC systems in this case are proposed by combining the compensation algorithms and learning algorithms.

3.2.3 Properties of the system

When data interrupt occurs, the learning gain and the system matrices must have a certain relationship among them. It means that the system can influence the convergence performance. Moreover, an effective algorithm applied on the linear CILC system may not be feasible for the nonlinear CILC system. If the nonlinear system is non-minimum-phase, the study of the convergence of the CILC systems could be much more complicated. More discussions are given in Section 4.

3.3 New algorithms for CILC problems and simulations

3.3.1 System description

The matrices in system (8) for the simulations are given as follows

$$A = \begin{bmatrix} 0.5 & 1 \\ -1 & 2 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad (12)$$

where $x_k(0) = 0$ for $k = 0, 1, \cdots$. The output restriction is $|y_d - y_k| < 0.5$, namely $\sigma = 0.5$.

The desired trajectory is

$$y_d(t) = 10(t - t^2), t \in \{0, 1, \cdots, T\} \quad (13)$$

where $T = 1$.

3.3.2 Compararison of the learning algorithms

Here, the design of the compensation algorithm, denoted by *Algo.A*, is to make the unknown output data zero, namely $e_k(t) = 0, t \in \{T_k, T_k + 1, \cdots, T\}$. The D-type learning algorithm is used. In this case, the new algorithm is expressed as

$$\begin{array}{llcl} \text{LA:} & u_{k+1}(t) & = & u_k(t) + \Gamma_d(e_k(t+1) - e_k(t)) \\ & & & \text{for } t \le T_k - 1 \\ \text{CA:} & u_{k+1}(t) & = & u_k(t) \\ & & & \text{for } t > T_k - 1 \end{array} \quad (14)$$

If the P-type learning algorithm is used, the new algorithm is expressed as

$$\begin{array}{llcl} \text{LA:} & u_{k+1}(t) & = & u_k(t) + \Gamma_p e_k(t+1) \\ & & & \text{for } t \le T_k - 1 \\ \text{CA:} & u_{k+1}(t) & = & u_k(t) \\ & & & \text{for } t > T_k - 1 \end{array} \quad (15)$$

The simulation results are presented in Fig. 6.

According to the simulation results, Fig. 6(a) shows a much better convergence performance and a higher convergence rate than Fig. 6(b). That means the algorithm (14), which uses the D-type learning algorithm, is better than the algorithm (15) which uses the P-type learning algorithm.

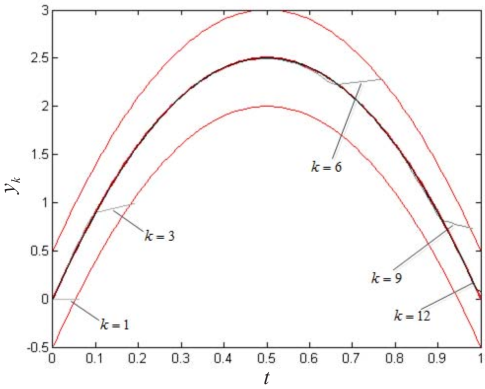
3.3.3 Compararison of the compensation algorithms

Here, a new design of the compensation algorithm, denoted by *Algo.B*, is applied. This algorithm uses the existing data at time T_k. This method is already used in the paper [16] with a case of data dropout. The missing data at time t is compensated by the existing data at time $t - 1$. The convergence of the system is achieved while the convergence rate gets slower. Here, all the unknown data in $t \in \{T_k, T_k + 1, \cdots, T\}$ can be replaced by the data at time T_k, namely, $e_k(t + 1) = e_k(T_k)$. The D-type learning algorithm is used. In this case, the new algorithm is expressed as

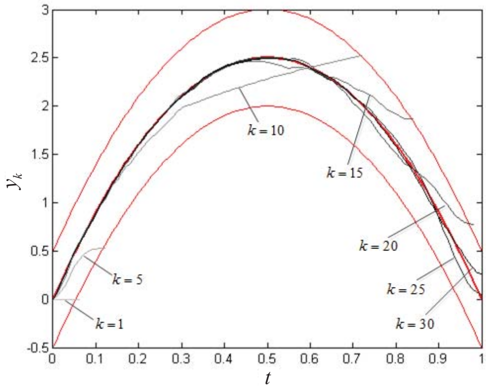
$$\begin{array}{llcl} \text{LA:} & u_{k+1}(t) & = & u_k(t) + \Gamma_d(e_k(t+1) - e_k(t)) \\ & & & \text{for } t \le T_k - 1 \\ \text{CA:} & u_{k+1}(t) & = & u_k(t) + \Gamma_d(e_k(T_k) - e_k(T_k - 1)) \\ & & & \text{for } t > T_k - 1 \end{array} \quad (16)$$

If the P-type learning algorithm is used, the new algorithm is expressed as

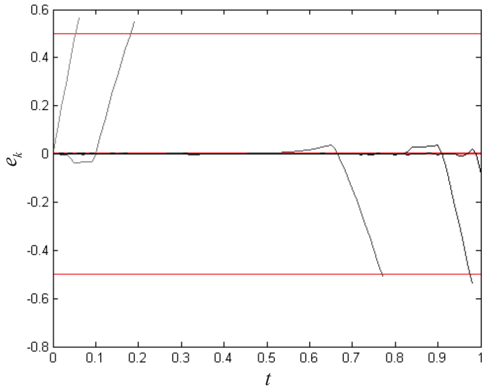
$$\begin{array}{llcl} \text{LA:} & u_{k+1}(t) & = & u_k(t) + \Gamma_p e_k(t+1) \\ & & & \text{for } t \le T_k - 1 \\ \text{CA:} & u_{k+1}(t) & = & u_k(t) + \Gamma_p e_k(T_k) \\ & & & \text{for } t > T_k - 1 \end{array} \quad (17)$$



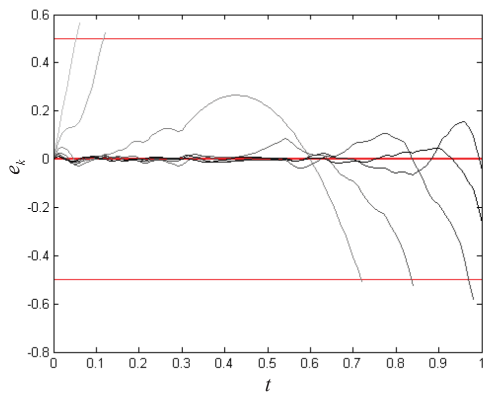
(a) D-type algorithm with $\Gamma_d = 80$ and $k = 12$



(b) P-type algorithm with $\Gamma_p = 6$ and $k = 30$

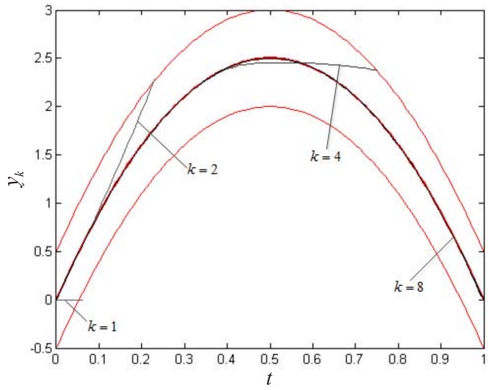


(c) Output error with $\Gamma_d = 80$ and $k = 12$

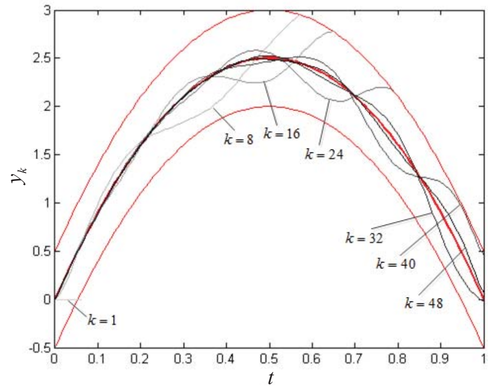


(d) Output error with $\Gamma_p = 6$ and $k = 30$

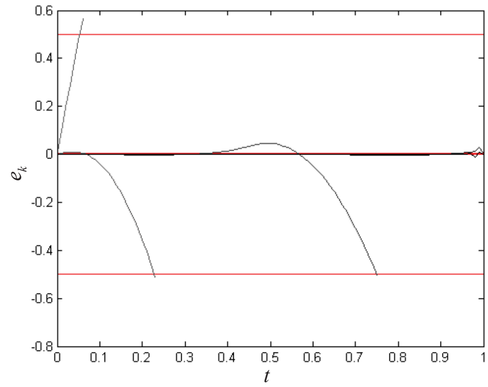
Fig. 6: Simulation results with *Algo.A*



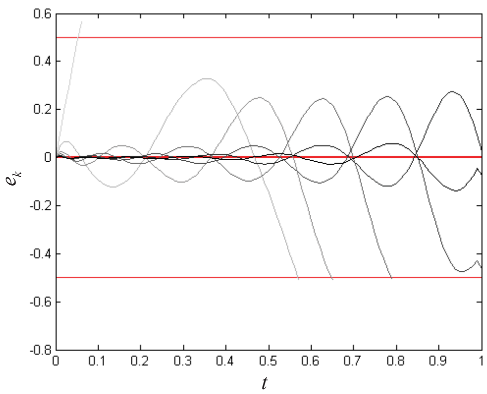
(a) D-type algorithm with $\Gamma_d = 80$ and $k = 8$



(b) P-type algorithm with $\Gamma_p = 6$ and $k = 48$



(c) Output error with $\Gamma_d = 80$ and $k = 8$



(d) Output error with $\Gamma_p = 6$ and $k = 48$

Fig. 7: Simulation results with *Algo.B*

The simulation results are presented in Fig. 7.

According to the simulation results, Fig. 7(a) shows a much better convergence performance and a higher convergence rate than Fig. 7(b). This proves again that the use of D-type learning algorithm is better than that of the P-type learning algorithm. However, by comparing Fig. 7 with Fig. 6, one can observe that the use of *Algo.A* can achieve a better convergence performance when P-type learning algorithm is used and the use of *Algo.B* can achieve a better convergence performance when D-type learning algorithm is used. Since the D-type learning algorithm is better than P-type learning algorithm, it is proved that the first key point to solve the CILC problems is to choose a reasonable learning algorithm.

The simulation results present an idea for studying the CILC systems. There are still many problems that can be discussed.

4 Discussions

4.1 Convergence of CILC systems

The simulation results above show that the convergence of the linear SISO CILC systems can be achieved by using reasonable algorithms. Several simulations using minimum-phase linear MIMO systems have also been performed. All the results point to one conjecture: if the convergence of a classical linear ILC system can be achieve by using a reasonable learning algorithm, the convergence of the corresponding CILC system can also be obtained by using the same learning algorithm with the compensation algorithms proposed in this paper. That means the convergence of the classical ILC systems may be a criterion for the convergence of the CILC systems. Thus, the following part of our future work is to prove this conjecture.

Furthermore, for more complex systems, such as the non-minimum-phase systems and nonlinear systems, this conjecture may be incorrect. The increase of the interruption moment cannot be ensured and it may even decrease progressively. In this case, a new criterion needs to be established for the convergence study of CILC systems.

4.2 Convergence rate of CILC systems

A high convergence rate of CILC systems is required in practice, such as the three examples listed in Section 1. Thus, after the convergence of a CILC system is ensured, one needs to improve the convergence rate. According to the case study in Section 3, both LA and CA can influence the convergence rate of CILC systems. On one hand, many classical ILC learning algorithms [17] can be employed to improve the convergence rate of CILC systems. On the other hand, CA is also important for the convergence rate. If the internal information of the system is available, the optimal method [18] can be employed to design CA. But, in practice, the accurate system model cannot always be obtained. In this situation, a reasonable offline CA is required, such as the CAs in Section 3. Moreover, an online CA may result in a better convergence rate. For example, the system model can be estimated by online system identification, and the corresponding mapping is denoted by $\hat{G}_k$ in the k-th iteration. Then, $\hat{G}_k$ can be used to design CA, such as $u_{k+1}(t) = \hat{G}_k^{-1} y_d(t), t \geq T_k$, where $\hat{G}_k^{-1}$ is the inversion of $\hat{G}_k$. In order to obtain a high convergence rate, the original restriction should be employed instead of the uniform restriction because more output data can be obtained under the original restriction.

5 Conclusions

In this paper, the CILC problems are presented through several practical examples. The output restrictions are unified into a fixed form in order to simplify the problems. A linear case is analyzed in order to illustrate the current studies for the CILC systems and several factors that can influence the convergence and convergence rate are presented. A few new algorithms are also proposed to study the linear CILC systems. Simulations are made to prove the feasibility of the proposed methods. However, there are still many research topics deserved to study in the future, for example, the stability issues of CILC systems.

References

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

[2] M.X. Sun, B.J. Huang, *Iterative learning control*. Beijing: National Defense Industry Press, 1999.5: 1-7. (In Chinese)

[3] S. Liu, Iterative learning control systems design and its applications, Zhejiang University, China, 2002. (In Chinese)

[4] D. Bristow, M. Tharayil, A.G. Alleyne, A survey of iterative learning control, *IEEE Control Systems*, 26(3): 96-114, 2006.

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

[6] W. Messner, R. Horowitz, W.W. Kao, et al, A new adaptive learning rule, *IEEE Trans. on Automatic Control*, 36(2): 188-197, 1991.

[7] M. Norrlof, An adaptive iterative learning control algorithm with experiments on an industrial robot, *IEEE Trans. on Robotics and Automation*, 18(2): 245-251, 2002.

[8] Y.Q. Chen, K.L. Moore, A practical iterative learning path-following control of an omni-directional vehicle, *Asian Journal of Control,* 4(1): 90-98, 2002.

[9] O. Purwin, R.D. Andrea, Performing aggressive maneuvers using iterative learning control, in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, 2009: 1731-1736.

[10] Q. Quan, Z.B. Wei, J. Gao, et al, A survey on modeling and control problems for probe and drogue autonomous aerial refueling at docking stage, *Acta Aeronautica et Astronautica Sinica*, 35(9): 2390-2410, 2014. (In Chinese)

[11] R.P. Dibley, M.J. Allen, N. Nabaa, et al, Autonomous airborne refueling demonstration phase I flight-test results, *AIAA Paper*, 2007 (6639).

[12] H.S. Ahn, Y.Q. Chen, K.L. Moore, Intermittent iterative learning control, in *Proceedings of the 2006 IEEE International Symposium on Intelligent Control*, 2006: 832-837.

[13] H.S. Ahn, K.L. Moore, Y.Q. Chen, Stability of discrete-time iterative learning control with random data dropouts and delayed controlled signals in networked control systems, in *Proceedings of 10th Intl. Conf. on Control, Automation, Robotics and Vision*, 2008: 757-762.

[14] C. Liu, J. Xu, J. Wu, Iterative learning control for network systems with communication delay or data dropout, in *Proceedings of 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, 2009: 4858-4863.

[15] X.H. Bu, F.S. Yu, Z.S. Hou, et al, Iterative learning control for a class of nonlinear systems with random packet losses, *Nonlinear Analysis: Real World Applications*, 14(1): 567-580, 2013.

[16] Y.J. Pan, H. J.Marquez, T. Chen, et al, Effects of network communications on a class of learning controlled non-linear systems, *International Journal of Systems Science*, 40(7): 757-767, 2009.

[17] M. Norrlof, S. Gunnarsson, Experimental comparison of some classical iterative learning control algorithms, *IEEE Trans. on Robotics and Automation*, 18(4): 636-641, 2002.

[18] S. Gunnarsson, M. Norrlof, On the design of ILC algorithms using optimization, *Automatica*, 37(12): 2011-2016, 2001.