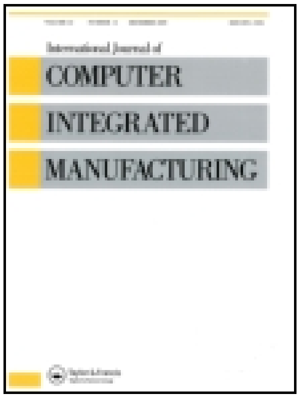

International Journal of Computer Integrated Manufacturing
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tcim20

An iterative learning control method and mathematical model for robotic manipulation at undesired locations

Xiao-Dong Li , J. K. L. Ho & T. W. S. Chow
[a] City University of Hong Kong , Kowloon Tong, Hong Kong
E-mail:
Published online: 19 Feb 2007.

International Journal of Computer Integrated Manufacturing, Vol. 18, No. 6, September 2005, 480 – 486

An iterative learning control method and mathematical model for robotic manipulation at undesired locations

XIAO-DONG LI, J. K. L. HO,* and T. W. S. CHOW

City University of Hong Kong, Kowloon Tong, Hong Kong

This paper presents an iterative learning control (ILC) strategy for a robotic manipulator when objects have to be picked up at undesired locations. The proposed ILC method can ensure that a robotic manipulator achieves its task in real time following a predetermined number of control steps. The paper has provided a robotic control strategy and mathematical formulae for the implementation of a proactive assurance control. The proactive assurance control can improve the quality of production in terms of maintaining the production workflow as a fixed value at all times, even though the components to be assembled are not in their desired locations. This can accelerate the production process owing to the possibility of eliminating the task of mounting the components at their exact locations.

Keywords: Iterative learning control, Robotic manipulation, Undesired locations, proactive assurance

1. Introduction

In assembly processes, robotic manipulators pick and insert components into a workpiece. The time taken for a robotic manipulator to carry out the assembly operation such as picking up a component from a pallet or buffer on the top of an automatic guided vehicle (AGV) is critical in terms of workflow balancing. Therefore, it is very important to keep the time taken at a fixed value for a robotic manipulator to move from the start point to the end point of the pre-described trajectory. This is why the component is usually mounted by a set of fixtures in a predefined location to achieve a single, pre-described robot trajectory. Recently, much research has been carried out to investigate controlling the robotic manipulator to reach an object at varying locations (Gomi 2003, Hosek 2003, Chandra *et al.* 2003). It has been reported that the time taken to reach the objects in different locations is very difficult to accomplish as a fixed value without variation. At present, there is a lack of literature addressing the quality of production in terms of real-time control strategies, theories, and formulae to interrogate and control a robotic manipulator to determine if it can complete its task exactly at a pre-defined number of control steps, when the components are not in their desired locations. The objective of this paper is to provide the theoretical aspect with the mathematical formulae for the interrogation and the control of a robotic manipulator to reach a component, even if it is placed in an undesired location at a pre-defined number of control steps. The proposed strategy and formulae in the paper can be implemented as proactive quality assurance schemes that can handle the unforeseen difficulty of components placed in undesired locations, but can still prevent the variation of production workflow. Therefore, the production workflow is guaranteed at all times to a pre-defined constant, regardless of whether the components are placed at their desired locations. Furthermore, the scheme can accelerate the production cycle because the effort spent on mounting the component at the exact location can be eliminated; this really can improve the quality of production.

ILC has generated considerable research interest since it was first introduced by Arimoto *et al.* (1984). The objective of ILC is to determine a control input so that the tracking of a given reference signal or the output trajectory over a fixed time interval is possible. The control inputs are

*Corresponding author. Email: MEJOHNHO@cityu.edu.hk

International Journal of Computer Integrated Manufacturing
ISSN 0951-192X print/ISSN 1362-3052 online © 2005 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/09511920400030146

updated iteratively after each operation using the error measurements in the previous cycle. This makes the application of the ILC approach increasingly important in many control applications, such as robot manipulators. Until now, a lot of ILC methods have been presented in the control field (Arimoto 1984, Moore 1993, Chow and Fang 1998, Chow and Li 2000, Geng and Jamshidi 1990, Geng *et al*. 1990, Kurek and Zaremba 1993, Choi and Lee 2000, Sugie and Ono 1991). The most widely used ILC method is the proportional-plus-integral-plus-derivative (PID)-type approach because it essentially forms a PID-like system.

In recent years, two-dimensional (2D) system theory has successfully been introduced to the ILC approach (Chow and Fang 1998, Chow and Li 2000, Geng and Jamshidi 1990, Geng *et al*. 1990, Kurek and Zaremba 1993). Owing to the two independent dynamic processes of the 2D system, the 2D model provides an excellent mathematical platform to describe both the dynamics of the control system and the behaviour of the learning iteration. Very promising results on ILC for linear multivariable systems have been obtained (Chow and Fang 1998, Geng and Jamshidi 1990, Geng *et al*. 1990, Kurek and Zaremba 1990).

Section 2 of this paper focuses on the introduction of a 2D system theory and method, which are ILC techniques for the control of the robotic manipulator. This section covers the ILC strategy and theorem for the control of a robotic manipulator and the proof of the formulae. The proactive assurance control, with a case example illustrating how the robotic manipulator can be controlled by the fixed number of control steps to reach the desired location and the location of deviated from the desired, is also discussed in section 3. Section 4 presents conclusions.

2. The ILC strategy for controlling a robotic manipulator

A computer-controlled robotic manipulator for assembling work usually can be described as a linear time-variant discrete system with delay as follows:

$$x(t+1) = A(t)x(t) + A_0(t)x(t-\tau) + B(t)u(t) \qquad (1a)$$

$$y(t) = C(t)x(t) \qquad (1b)$$

where $x(t)\in R^n$ is a state vector, $u(t)\in R^m$ is an input vector, $y(t)\in R^p$ is an output vector, τ is a time-delay parameter, and $A(t)$, $A_0(t)$, $B(t)$, $C(t)$ are real bounded time-variant matrices of appropriate dimensions. The initial condition of the system given in equation (1) is $x(t) = \phi(t)$ for $t = -\tau, -\tau + 1, \ldots, 0$.

It is assumed that a robotic manipulator performs the operation of picking up a component from an AGV or buffer, and the component is laid at the desired location P_0. $u_0(t)$ is signified as the normal input for driving the robotic manipulator to the point of P_0 at the Nth control step along a fixed moving path to produce the output trajectory $y_0(t)$. Therefore, $y_0(N) = P_0$.

Now we investigate the case of a component that is placed away from the desired location P_r. The objective of the investigation is to find an appropriate control input $u_r(t)$ to produce the moving path $y_r(t)$ of the robotic manipulator so that the robotic manipulator can reach the component at the same control step N, that is $y_r(N) = P_r$.

The ILC strategy is employed to tackle the problem of the robotic manipulator picking up a component at the undesired location. For the undesired location P_r of the component, we can set a moving path $y_r(t)$ ($t = 0,1, \ldots, N$) with $y_r(N) = P_r$, which is realizable to equation (1). Then, we iteratively find the corresponding control input $u_r(t)$, $t = 0,1,\ldots N–1$, such that the robotic manipulator follows the moving path $y_r(t)$. Once the control input $u_r(t)$ is obtained, the robotic manipulator can reach the object P_r at the control step N.

If k denotes learning iteration, a general ILC rule is given as

$$u_{k+1}(t) = u_k(t) + \Delta u_k(t), t = 0,\ 1, \cdots, N-1, \qquad (2)$$

where Δu denotes modification of the control input. Sequentially, the system given in equation (1) can be modelled as following 2D time-variant form

$$x_k(t+1) = A\,(t)x_k(t) + A_0(t)x_k(t-\tau) + B(t)\,u_k(t) \qquad (3a)$$

$$y_k(t) = C(t)\,x_k(t) \qquad (3b)$$

The boundary conditions for the 2D system [equation (3)] are

$$x_k(t) = \phi(t) \text{ for } t = -\tau,\ -\tau+1,\ \cdots,\ 0 \text{ and } k = 0, 1, 2\cdots, \qquad (4a)$$

$$u_0(t) \text{ for } t = 0,\ 1, \cdots, N-1. \qquad (4b)$$

Our ILC objective is to find a suitable ILC rule [equation (2)] such that

$$\lim_{k\to\infty} y_k(t) = y_r(t) \text{ for } t = 1, \cdots, N.$$

Let us denote

$$\eta_k(t) = x_{k+1}(t-1) - x_k(t-1), \qquad (5)$$

And

$$e_k(t) = y_r(t) - y_k(t). \qquad (6)$$

Using equations (2) and (3), we obtain for t = 0,1, … N,

$$\begin{aligned}\eta_k(t+1) &= x_{k+1}(t) - x_k(t)\\ &= A(t-1)\eta_k(t) + A_0(t-1)\eta_k(t-\tau) + B(t-1)\Delta u_k(t-1)\end{aligned} \tag{7}$$

$$\begin{aligned}e_{k+1}(t) - e_k(t) &= -C(t)\left[x_{k+1}(t) - x_k(t)\right]\\ &= -C(t)A(t-1)\eta_k(t) - C(t)A_0(t-1)\eta_k(t-\tau)\\ &\quad - C(t)B(t-1)\Delta u_k(t-1).\end{aligned} \tag{8}$$

Applying the following rule to equations (7) and (8), respectively, for control calculation

$$\begin{aligned}\Delta u_k(t) &= K_1(t+1)\eta_k(t+1) + K_2(t+1)\eta_k(t-\tau+1)\\ &\quad + K_3(t+1)e_k(t+1)\end{aligned} \tag{9}$$

where $K_1(t)$, $K_2(t)$, and $K_3(t)$ are bounded, one obtains a 2D linear time-variant control error system for t = 0,1, …, N and $k \geqslant 0$

$$\begin{aligned}\eta_k(t+1) &= [A(t-1) + B(t-1)K_1(t)]\eta_k(t) + [A_0(t-1)\\ &\quad + B(t-1)K_2(t)]\eta_k(t-\tau) + B(t-1)K_3(t)e_k(t)\end{aligned} \tag{10a}$$

$$\begin{aligned}e_{k+1}(t) &= -[C(t)A(t-1) + C(t)B(t-1)K_1(t)]\eta_k(t)\\ &\quad - [C(t)A_0(t-1) + C(t)B(t-1)K_2(t)]\eta_k(t-\tau)\\ &\quad + [I - C(t)B(t-1)K_3(t)]e_k(t)\end{aligned} \tag{10b}$$

where (and afterwards) I is simply used to represent an identity matrix of appropriate order. Furthermore, let us make the following matrix denotation

$$\tilde{\eta}_k(t) = \begin{bmatrix}\eta_k(t)\\ \eta_k(t-1)\\ \vdots\\ \eta_k(t-\tau)\end{bmatrix};$$

$$\tilde{A}(t) = \begin{bmatrix}A(t-1)+B(t-1)K_1(t) & 0 & \cdots & 0 & A_0(t-1)+B(t-1)K_2(t)\\ I & 0 & \cdots & 0 & 0\\ \cdots & \cdots & \cdots & \cdots & \cdots\\ 0 & 0 & \cdots & I & 0\end{bmatrix};$$

$$\tilde{B}(t) = \begin{bmatrix}B(t-1)K_3(t)\\ 0\\ \vdots\\ 0\end{bmatrix}; \tilde{C}(t) = [\,C(t) \quad 0 \quad \cdots \quad 0\,];$$

Then, from equation (10), the following 2D linear time-variant discrete Roessor's-type model (Geng *et al.* 1990) can be derived

$$\tilde{\eta}_k(t+1) = \tilde{A}(t)\tilde{\eta}_k(t) + \tilde{B}(t)e_k(t) \tag{11a}$$

$$e_{k+1}(t) = -\tilde{C}(t)\tilde{A}(t)\tilde{\eta}_k(t) + [I - C(t)B(t-1)K_3(t)]e_k(t) \tag{11b}$$

According to equation (4), the boundary conditions of the 2D system [equation (11)] are $\tilde{\eta}_k(1) = 0$ for k = 0,1,2,… and $e_0(t)$ for t = 1,2,…N, which is finite.

For the 2D error system [equation (11)] with the Roessor's-type model, which clearly describes the ILC process for robotic manipulation at undesired locations, we have the following theorem.

Theorem: For a 2D ILC model [see equation (3)], if there exist bounded matrices $K_1(t)$, $K_2(t)$, and $K_3(t)$ to make $||I–C(t)B(t–1)K_3(t)|| < 1$, t = 1,…, N, ($||\cdot||$ represents the matrix norm), then the ILC rule

$$\begin{aligned}u_{k+1}(t) &= u_k(t) + K_1(t+1)[x_{k+1}(t) - x_k(t)]\\ &\quad + K_2(t+1)[x_{k+1}(t-\tau) - x_k(t-\tau)]\\ &\quad + K_3(t+1)e_k(t+1)\end{aligned} \tag{12}$$

can ensure $\lim_{k\to\infty}\begin{pmatrix}\|\tilde{\eta}_k(t)\|\\ \|e_k(t)\|\end{pmatrix} = 0$ for t = 1, …, N.

Proof: (Mathematical induction). As t = 1, $\tilde{\eta}_k(1) = 0$ for k = 0,1,2,…. From equation (11b), we have

$$\|e_{k+1}(1)\| \leq \|I - C(1)B(0)K_3(1)\| \cdot \|e_k(1)\|. \tag{13}$$

Thus, equation (13) is a contraction in $||e_k(1)||$ as k→∞, and $\lim_{k\to\infty}\|e_k(1)\| = 0$, if the condition $||I\text{-}C(1)B(0)K_3(1)|| < 1$ is satisfied. Sequentially, we have $\lim_{k\to\infty}\begin{pmatrix}\|\tilde{\eta}_k(1)\|\\ \|e_k(1)\|\end{pmatrix} = 0$

Assume that for $t = l$, $\lim_{k\to\infty}\begin{pmatrix}\|\tilde{\eta}_k(l)\|\\ \|e_k(l)\|\end{pmatrix} = 0$. As a direct result of equation (11a), $\lim_{k\to\infty}\|\tilde{\eta}_k(l+1)\| = 0$ because $\tilde{A}(t)$ and $\tilde{B}(t)$ are bounded. Considering the case of $t = l + 1$, from equation (11b), we have

$$\begin{aligned}&\lim_{k\to\infty}\|e_{k+1}(l+1)\|\\ &\leq \lim_{k\to\infty}(\|I - C(l+1)\cdot B(l)\cdot K_3(l+1)\| \cdot \|e_k(l+1)\|).\end{aligned} \tag{14}$$

Therefore, it can be derived from equation (14) that $\lim_{k\to\infty}\|e_k(l+1)\| = 0$ if the condition $||I\text{-}C(l+1)B(l)K_3(l+1)|| < 1$ is satisfied. That is, the statement is also true for $t = l + 1$. Based on mathematical induction, we have $\lim_{k\to\infty}\begin{pmatrix}\|\tilde{\eta}_k(t)\|\\ \|e_k(t)\|\end{pmatrix} = 0$ for t = 1,…,N. **theorem** is proved.

From **theorem**, it is noted that $\lim_{k\to\infty}\begin{pmatrix}\|\tilde{\eta}_k(t)\|\\ \|e_k(t)\|\end{pmatrix} = 0$ (t = 1, …, N) has nothing to do with the matrix $K_1(t)$ and $K_2(t)$. For simplicity, we might as well let $K_1(t) = K_2(t) = 0$ and $K(t) = K_3(t)$ in equation (12), and formulate the following control formula 1.

Control formula 1: For a 2D ILC model [equation (3)], if there exists a matrix $K(t)$ to make $||I\text{-}C(t)B(t\text{-}1)K(t)|| < 1$ for t = 1, …, N, then the ILC rule

$$u_{k+1}(t) = u_k(t) + K(t+1)e_k(t+1), \tag{15}$$

can ensure $\lim_{k\to\infty}\begin{pmatrix}\|\tilde{\eta}_k(t)\| \\ \|e_k(t)\|\end{pmatrix} = 0$ for $t = 1, \ldots, N$.

Remark (1): Note that the condition $\|I\text{-}C(t)B(t\text{-}1)K(t)\| < 1$, $t = 1, \ldots N$ is robust with respect to small perturbations of the system parameters $B(t)$ and $C(t)$, and does not require the information of system matrix $A(t)$ and $A_0(t)$. Thus, the control formula 1 is robust.

Remark (2): According to the relation between matrix norm and spectral radius of matrix, for any given $\varepsilon > 0$, there exists a kind of matrix norm such that

$$\|I - C(t)B(t-1)K(t)\| \leq \rho(I - C(t)B(t-1)K(t)) + \epsilon. \tag{16}$$

As $\rho(I\text{-}C(t)B(t\text{-}1)K(t)) \leqslant p < 1$ for $t = 1,\ldots,N$, we take $0 < \varepsilon < 1\text{-}p$. As a direct result of equation (16), we get $\|I\text{-}C(t)B(t\text{-}1)K(t)\| < 1$ for $t = 1, \ldots N$. Therefore, as the sufficient condition in control formula 1 is changed as $\rho(I\text{-}C(t)B(t\text{-}1)K(t)) \leqslant p < 1$, the conclusion of control formula 1 is also true.

Remark (3): The above **theorem** has been designed for a linear time-variant discrete model with a single time-delay in state. However, a similar result can also be obtained for the ILC problem of a linear time-variant discrete system with multiple time-delays in state.

Remark (4): Despite the fact that control formula 1 provides a simple form of the ILC rule for robotic manipulator systems, the existence of the term $K_1(t+1)[x_{k+1}(t)\text{-}x_k(t)] + K_2(t+1)[x_{k+1}(t\text{-}\tau)\text{-}x_k(t\text{-}\tau)]$ in equation (12) sometimes can increase the convergence rate of the ILC rule given in equation (12) or can make the ILC system present good properties. That can be verified from the following control formula 2.

One of the important features of the ILC is that it requires less prior knowledge about the controlled system in the controller design and set-up phase. From the above analysis, control formula 1 can ensure a robotic manipulator to pick up an object at an undesired location after several tries, if the system parameters $A(t)$, $A_0(t)$, $B(t)$, and $C(t)$ of the robotic manipulator are not accurately known. However, in most cases the system parameters of the robotic manipulator are identified and set at the system set-up phase before operation starts. Therefore, the $K_3(t)$, $K_1(t)$, and $K_2(t)$ can be determined as follows:

$$K_3(t) = (C(t)B(t-1))^T\left[C(t)B(t-1)(C(t)B(t-1))^T\right]^{-1}$$
$$K_1(t) = -K_3(t)C(t)A(t-1) \text{ and } K_2(t) = -K_3(t)C(t)A_0(t-1).$$

Equation (10b) shows $e_1(t) = 0$ for $t = 1, \ldots N$ no matter what $e_0(t)$ is. The robotic manipulator can pick up an object at the undesired location through a single learning iteration, i.e. in one attempt. The robotic manipulator can achieve its task at a given number of control steps in a real-time tuning control fashion, which is very important in balancing the workflow during production. On the other hand, the matrix $(C(t)B(t\text{-}1))^T[C(t)B(t\text{-}1)(C(t)B(t\text{-}1))^T]^{-1}$, which is the right inverse of matrix $C(t)B(t\text{-}1)$, exists if matrix $C(t)B(t\text{-}1)$ has full-row rank. Thus, the following control formula 2 can be drawn as follows:

Control formula 2: For a 2D ILC model [equation (3)], if matrix $C(t)B(t\text{-}1)$ has full-row rank for $t = 1, \ldots, N$, then the ILC rule [equation (12)] with $K_3(t) = (C(t)B(t\text{-}1)^T[C(t)B(t\text{-}1)(C(t)B(t\text{-}1))^T]^{-1}$, $K_1(t) = -K_3(t)C(t)A(t\text{-}1)$, and $K_2(t) = -K_3(t)C(t)A_0(t\text{-}1)$ drives the control error to zero for the pre-defined output trajectory $y_r(t)$ at $t = 1, \ldots, N$ after only one attempt.

3. The proactive assurance control scheme

In this section, the derived theorem is applied to set-up a proactive assurance control scheme. The scheme consists of three essential parts. The first part is to interrogate whether the robotic manipulator has the capability to reach a component at a pre-defined number of control steps. This can be achieved by examining whether $C(t)B(t\text{-}1)$ of the robotic manipulator has a full-row rank for $t = 1, \ldots, N$ as discussed in control formula 2. If a full-row rank exists, this indicates that the robotic manipulator has the ability to pick up the component placed in the varying location at a pre-defined number of control steps.

Assuming the dynamics of a robotic manipulator is modelled as a linear time-variant discrete system with delay as follows:

$$x(t+1) = \begin{bmatrix} -0.24 & 0.1 \\ 0.5\sin t + 0.04 & -3.5 \end{bmatrix} x(t) + \begin{bmatrix} 1 & 0.001t \\ -0.4 & 0.2 \end{bmatrix} x(t-5) + \begin{bmatrix} 1.3 & 0.002t \\ 0 & 1.5 \end{bmatrix} u(t), \tag{17a}$$

$$y(t) = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} x(t), \tag{17b}$$

where the initial condition $x(t) = \left[\begin{smallmatrix} 0 \\ 25 \end{smallmatrix}\right]$ for $t = -5, -4, \ldots, -1.0$, and the matrix $C(t)B(t\text{–}1)$ has a full-row rank for $t = 1, \ldots, 30$. That means the robotic manipulator can be controlled to reach an object at a fixed control step.

The second part of the scheme deals with the component placed in its desired location. The scheme is required to calculate the pre-programmed control input for the robotic manipulator to reach the desired location at the pre-defined number of control steps.

Assuming the component is placed at the desired location $P_0 = (30,\ 0)$ as shown in figure 1, there exists a pre-programmed control input of the robotic manipulator $u_0(t) = \begin{bmatrix} u_{01}(t \\ u_{02}(t) \end{bmatrix}$, which can be calculated by control formula 1, as shown in figures 2 and 3 to drive the dynamics of equation (17) to P_0 at the 30th control step ($t = 30$ which is pre-defined) along the trajectory.

$$\begin{bmatrix} y_{01}(t) \\ y_{02}(t) \end{bmatrix} = \begin{bmatrix} 30\cos(0.5\pi(1-0.033t)) \\ 50\sin(0.5\pi(1-0.033t)) \end{bmatrix}. \tag{18}$$

The third part of the scheme handles the component placed in an undesired location. The scheme develops the new trajectory for the undesired location using the trajectory of the desired location as the reference to establish the trajectory. Applying control formula 2 calculates the required control input to reach the undesired location at the pre-defined number of control steps.

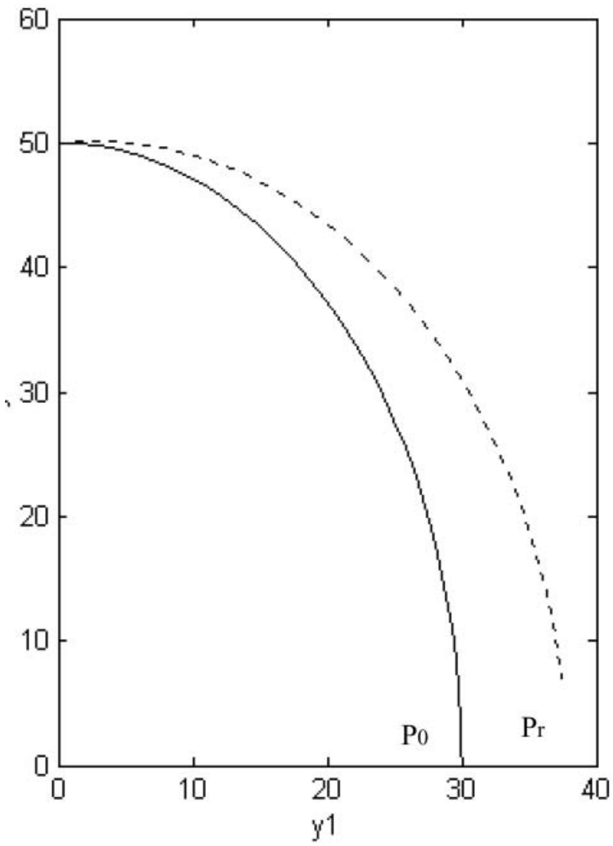

Figure 1. Moving trajectories of system [equation (17)] driven by $u_0(t)$ and $u_r(t)$, respectively. The solid line represents the motion trajectories of system [equation (17)] driven by $u_0(t)$, and the dashed line represents the motion trajectories of system [equation (17)] driven by $u_r(t)$.

Now, assuming that the component is placed at an undesired location $P_r = (37.5,\ 6)$ away from the desired P_0 as shown in figure 1. For the new pick-up point, the moving path for the robotic manipulator becomes as follows:

$$y_r(t) = \begin{bmatrix} y_{r1}(t) \\ y_{r2}(t) \end{bmatrix} = \begin{bmatrix} 30\cos(0.5\pi(1-0.033t)) + 0.25t \\ 50\sin(0.5\pi(1-0.033t)) + 0.2t \end{bmatrix} \tag{19}$$

Along the new moving path $y_r(t)$, the robotic manipulator can definitely reach the pick-up point P_r at the 30th control step. Consequently, the control of the robotic manipulator is to find a suitable input $u_r(t)$ so that the output of equation (17) is $y_r(t)$.

Let us apply control formula 2 with $K_3(t) = (C(t)B(t\text{-}1)^T [C(t)B(t\text{-}1)(C(t)B(t\text{-}1))^T]^{-1,}$ $K_1(t) = \text{-}K_3(t)C(t)A(t\text{–}1)$, and $K_2(t) = -K_3(t)C(t)A_0(t\text{–}1)$. Figures 4 and 5 show the results

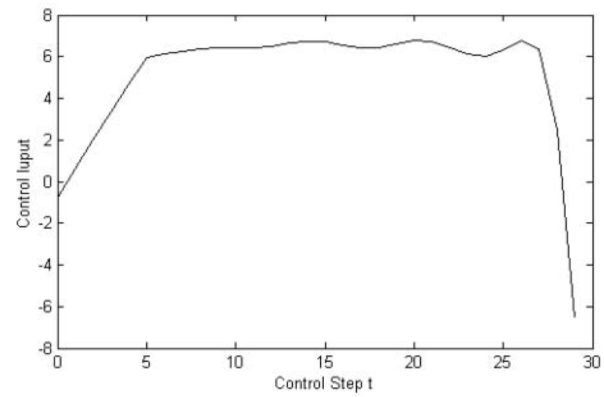

Figure 2. The pre-programmed control input $u_{01}(t)$, which makes the robotic manipulator pick up an object at the desired location P_0.

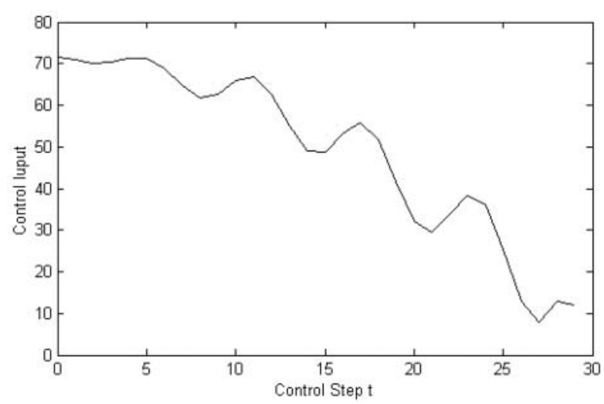

Figure 3. The pre-programmed control input $u_{02}(t)$, which makes the robotic manipulator pick up an object at the desired location P_0.

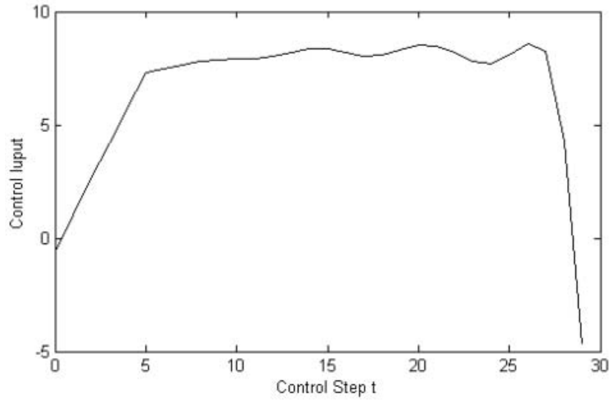

Figure 4. The required control input $u_{r1}(t)$, which makes the robotic manipulator pick up an object at the undesired location P_r.

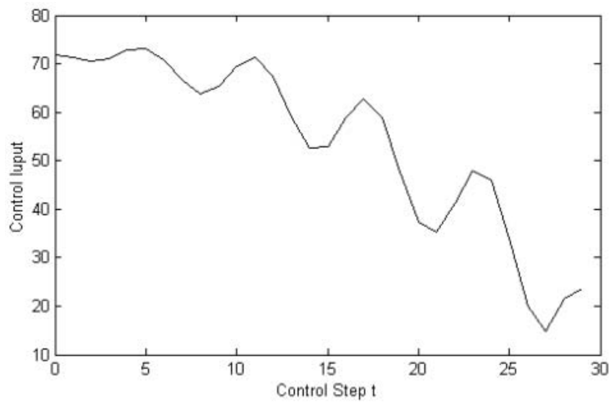

Figure 5. The required control input $u_{r2}(t)$, which makes the robotic manipulator pick up an object at the undesired location P_r.

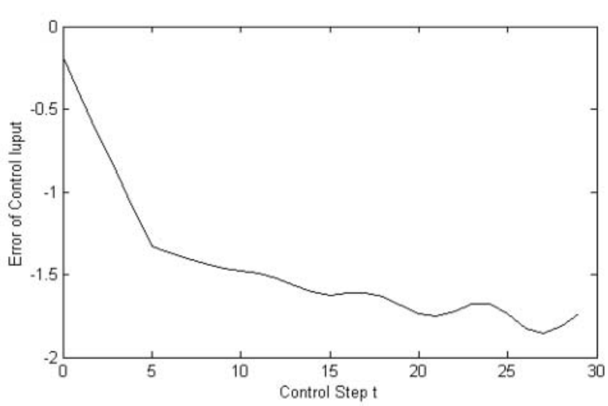

Figure 6. $u_{01}(t)$–$u_{r1}(t)$, the error between the pre-programmed control input and the required control input.

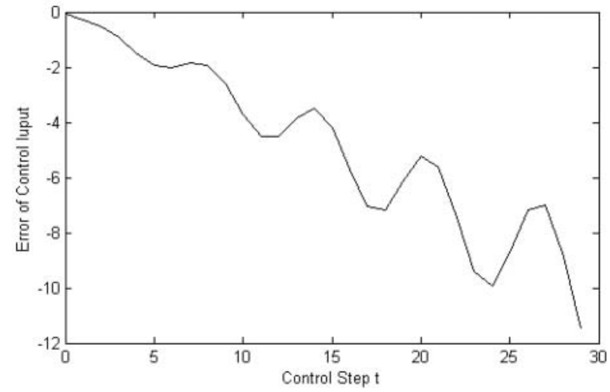

Figure 7. $u_{02}(t)$–$u_{r2}(t)$, the error between the pre-programmed control input and the required control input.

of the simulation illustrating that the required control input $u_r(t)$ can be found in a single learning iteration, and the robotic manipulator reaches the pick-up point P_r along the trajectory given in equation (19) at the 30th control step. The errors between the pre-programmed control input $u_0(t)$ and the new produced control input $u_r(t)$ are plotted in figures 6 and 7, respectively. Figure 1 shows the dynamic trajectories [equations (18) and (19)] of the robotic manipulator [equation (17)] of both components at the desired location $P_0 = (30,0)$ and deviated location $P_r = (37.5, 6)$.

The time interval for each control step is set to 0.1 s as the requirement in balancing the production workflow. This means the robotic manipulator can achieve its task at 3 s exactly. Control formula 2 has provided a real-time tuning control for the robotic manipulator.

4. Conclusion

The proposed ILC control method for the robotic manipulator to pick up an object in an undesired location at a pre-defined number of control steps is guaranteed. As illustrated, the developed theorem and formulae can be implemented as a proactive assurance control scheme to improve the quality of production. The proactive assurance control can avoid the variation of the production workflow owing to components placed in undesired locations and can accelerate the production cycle because the time spent on mounting the components at the exact locations can be eliminated.

References

Arimoto, S., Kawamura, S., and Miyazaki, F., Bettering operation of robots by learning. *Journal of Robot Systems*, 1984, **1**, 123–140.

Chandra, V., Zhongdong Huang, R. and Kumar, R., Automated control synthesis for an assembly line using discrete event system control theory.*IEEE Transactions on Systems, Man and Cybernetics C. Applications and Review*, 2003, **33**(2), 284–289.

Choi, J.Y. and Lee, J.S. Adaptive iterative learning control of uncertain robotic systems. *IEE Proceedings, Control Theory and Applications*, 2000, **147**(2), 217–223.

Chow, T. W. S. and Fang, Y. An iterative learning control method for continuous-time systems based on 2-D system theory. *IEEE Transactions on Circuits and Systems. Part I: Fundamental theory and applications*, 1998, **45**(4), 683–689.

Chow, T. W. S. , and Li, X.-D. A real-time learning control approach for nonlinear continuous-time systems using recurrent neural networks. *IEEE Transactions on Industrial Electronics*, 2000, **47**(2), 478–486.

Geng, Z. and Jamshidi, M. Learning control system analysis and design based on 2-D system theory. *Journal of Intelligent Robotic Systems*, 1992, **4**(1).

Geng, Z., Carroll, R., and Xies, J., Two-dimensional model and algorithm analysis for a class of iterative learning control system. *International Journal of Control*, 1990, **52**, 833–862.

Gomi, T., New AI and service robots. *Industrial Robot*, 2003, **30**(2), 123–138.

Hosek, M., Observer-corrector control design for robots with destabilizing unmodeled dynamics. *IEEE/ASME Transactions Mechatronics*, 2003, **8**(2), 151–164.

Kurek, J. E. and Zaremba, M.B. Iterative learning control synthesis based on 2-D system theory. *IEEE Transactions on Automatic Control*, 1993, **38**, 121–125.

Moore, K.L., *Iterative Learning Control for Deterministic Systems*, 1993 (Springer-Verlag: New York).

Sugie, T. and Ono, T. An iterative learning control law for dynamic systems. *Automatica*, 1991, **27**(4), 729.