01 Jun 2008

Design of a Linear Time-Varying Cross-Coupled Iterative Learning Controller

K. L. Barton

Douglas A. Bristow
Missouri University of Science and Technology, dbristow@mst.edu

Andrew G. Alleyne

Recommended Citation

K. L. Barton et al., "Design of a Linear Time-Varying Cross-Coupled Iterative Learning Controller," *American Control Conference*, Institute of Electrical and Electronics Engineers (IEEE), Jun 2008.
The definitive version is available at https://doi.org/10.1109/ACC.2008.4587104

2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Design of a Linear Time-Varying Cross-Coupled Iterative Learning Controller

Kira L. Barton, *Member,* IEEE; Douglas A. Bristow, *Member,* IEEE; and Andrew G. Alleyne, *Senior Member*, IEEE

***Abstract*—In many manufacturing applications contour tracking is more important than individual axis tracking. Many control techniques, including Iterative Learning Control (ILC), target individual axis error. Because individual axis error only indirectly relates to contour error, these approaches may not be very effective for contouring applications. Cross-Coupled ILC (CCILC) is a variation on traditional ILC that targets the contour tracking directly. In contour trajectories with rapid changes, high frequency control is necessary in order to meet tracking requirements. This paper presents an improved CCILC that uses a linear time-varying (LTV) filter to provide high frequency control for short durations. The improved CCILC is designed for raster-scan tracking on a Cartesian robotic test platform. Analysis and experimental results are presented.**

I. Introduction

Many manufacturing applications use repetitive processes in order to build numerous identical parts. The tracking performance of these systems is critical to the quality of these parts and ultimately to the success of the company [1]. As these parts move from the macro to the micro and nano-scale, the demand for enhanced precision motion control increases. Iterative learning control (ILC) is a feedforward control technique which improves individual axis tracking of a system through a process known as learning [1,2,3]. ILC uses the tracking errors from past iterations to update the feedforward control input, leading to improved axis tracking. Convergence of the learning process leads to a control input which is optimized for the specific trajectory being learned thereby yielding very low tracking error.

The first order learning algorithm is given by,

$$u_{j+1}(k) = u_j(k) + L(q)e_j(k) \qquad (1)$$

where u is the control input, e is the error, k is the discrete time index, j is the iteration index, q is the forward time-shift operator $qx(k) \triangleq x(k+1)$, and $L(q)$ is a linear time-invariant (LTI) learning function. One of the challenges with (1) is that high-frequency signals can propagate from iteration to iteration. High-frequency propagation can lead to undesirable outcomes such as large transient growth and long convergence times [1,4,3]. Some designs for L, such as the plant inverse method, dynamically couple with the plant dynamics in such a way as to nominally prevent high-frequency propagation. These designs, particularly plant model inversion [5], risk being over-designed and may not be robust to uncertainty in the plant dynamics. To prevent high-frequency propagation and achieve robustness, an additional variable, Q, is included in a modified first-order learning algorithm as

$$u_{j+1}(k) = Q(q)\left(u_j(k) + L(q)e_j(k)\right). \qquad (2)$$

$Q(q)$ is known as the Q-filter and is often a lowpass filter that removes high-frequency signals from the learning algorithm.

For many precision motion control applications, the reference trajectories include short durations of high frequency content combined with large durations of low frequency content [6], such as Fig. 1.

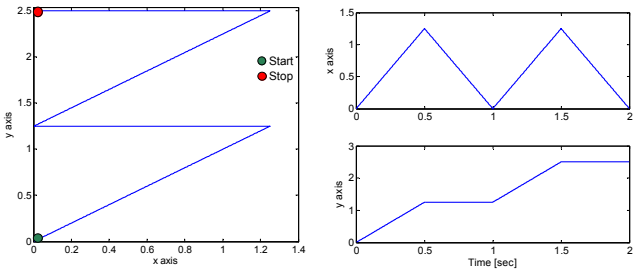

Fig. 1. Reference raster scanning trajectory.

For these types of trajectories, a time-invariant lowpass Q-filter may be the limiting factor in the ability of the learning controller to achieve low converged tracking error. One method for improving the tracking performance may be to redesign L so that the bandwidth of Q can be increased, which may require very accurate models of the plant dynamics. Alternatively, improvements can be made by focusing our attention on the design of a linear time-varying (LTV) Q-filter. Previous work [6] has shown that Q-filter

This work was supported by the University of Illinois at Urbana-Champaign Nano-CEMMS center NSF Award No. DMI-0328162.

Kira L. Barton is with the Department of Mechanical Science and Engineering at the University of Illinois, Urbana-Champaign, IL 61801 USA (phone: 217-244-6556; fax: 217-244-6534; email: barton2@uiuc.edu).

Douglas A. Bristow is with the Department of Mechanical and Aerospace Engineering at the Missouri University of Science and Technology, Rolla, MO 65409 USA.

Andrew G. Alleyne is with the Department of Mechanical Science and Engineering at the University of Illinois, Urbana-Champaign, IL 61801 USA (alleyne@uiuc.edu).

bandwidth can be increased for short periods of time without inducing large transient growth and long convergence times. This LTV Q-filter is highly effective for trajectories with short durations of high frequency content.

In many manufacturing systems, the desired trajectory requires motion from more than one axis. Typically in these multiaxis systems, individual controllers are designed for each motion axis. This decoupled approach works well for linear trajectories defined by single-axis movements because this control design technique focuses on improving individual axis tracking. However, in many applications, the emphasis is placed on the contour tracking performance. MIMO and SISO ILC only indirectly attack the problem of contour tracking through individual axis control. For applications in which the focus is on contour tracking, it may be beneficial to use a controller that directly focuses on minimizing the contour error.

Cross-coupled ILC (CCILC), introduced in [7], is a new variation on ILC that directly targets contour tracking. This new technique, derived from cross-coupled control (CCC) [8], uses the contour error directly in the learning algorithm. Just as CCC is combined with individual axis feedback control to provide low contour error with good command following, the CCILC can be combined with individual axis ILC. This combined design approach has been shown to be effective in reducing contour errors for a multi-axis robotic test platform [9,7].

This paper focuses on the use of the combined ILC and CCILC algorithm for tracking contour trajectories that exhibit large durations of low frequency control combined with short periods of high frequency control, Fig. 1. To this end, we combine CCILC with an LTV Q-filter design [6] to create a new LTV CCILC.

The remainder of this paper is structured as follows. Section II of this paper introduces the class of systems considered in this paper. The LTV Q-filter learning algorithm is described in Section III. Section IV gives a brief overview of the combined ILC and CCILC system. Stability and convergence analysis is provided in Section V, while a design procedure for the LTV CCILC learning algorithm is presented in Section VI. Experimental results are given in Section VII. Conclusion and future work are discussed in Section VIII.

II. Class of Systems

While ultimately we are concerned with controlling multiaxis systems, we focus the initial discussion of this work on single-input single-output (SISO) systems. The process can be extended to additional axes, as well as to the cross-coupled learning controller. In order to use learning controllers, the system is assumed to perform repeated operations in which the initial conditions, disturbances and trajectory are assumed identical from iteration to iteration. The input-output relationship of a discrete-time, LTI, SISO system can be given by [3],

$$y_j(k) = P(q)u_j(k) + d(k) \tag{3}$$

where y is the output signal of the plant model, u is the system control input signal, d is the exogenous signal (e.g. contains information about the initial states, as well as the disturbances) that is time-varying but not iteration varying, and P is the plant that is assumed open loop stable.

One can now consider that each of the above signals actually contains N-sampled values,

$$\begin{aligned} &u_j(k), k \in \{0,1,...,N-1\} \\ &y_j(k), k \in \{0,1,...,N-1\} \\ &d_j(k), k \in \{0,1,...,N-1\} \end{aligned}, \tag{4}$$

including the desired system output defined as

$$y_d(k) \in \{0,1,...,N-1\} \ . \tag{5}$$

The error signal for the j^{th} iteration is defined as

$$e_j(k) = y_d(k) - y_j(k) \ . \tag{6}$$

The goal of ILC is to generate a sequence of control inputs, each an improvement from iteration to iteration, that achieves rapid convergence of the error signal to a low converged value.

III. LTV Q-filter Learning Algorithm [6]

An LTV Q-filter can be written as

$$\begin{aligned} Q_k(q) = \lambda_{k,N_Q} q^{N_Q} + \cdots + \lambda_{k,1} q + \lambda_{k,0} + \lambda_{k,1} q^{-1} + \\ \cdots + \lambda_{k,N_Q} q^{-N_Q} \end{aligned} \tag{7}$$

where $\lambda_{k,i}$ are the impulse response coefficients of a lowpass filter with bandwidth Ω(k). The description, (7), is general enough to include many different types of filters including causal, noncausal, FIR, and IIR filters of any linear type (Gaussian, Butterworth, Chebychev, Bessel, etc.). Although any filter type may be used for the Q-filter, we will use a Gaussian filter because the filter impulse coefficients can be written explicitly as a function of the bandwidth, which is useful for an LTV filter. The coefficients (7) for the Gaussian filter are given by

$$\lambda_{k,i} = \frac{1}{\sum_{r=-N_Q}^{N_Q} \exp\left(\frac{r^2 \left(2\pi\Omega(k)\right)^2}{S^2 \ln 4} \right)} \exp\left(-\frac{i^2 \left(2\pi\Omega(k)\right)^2}{S^2 \ln 4} \right) \tag{8}$$

where S is the sample frequency in Hz.

The first order learning algorithm with an LTV filter is defined as

$$u_{j+1}(k) = Q_k(q)\left[u_j(k) + L(q)e_j(k) \right] \tag{9}$$

where $L(q)$ is designed using any general learning function design such as PD-type [10], model inverse [11], and

optimal LQ design [12]. The focus for this paper is on the design of the LTV Q-filter. This does leave open the possibility of a future design procedure which incorporates the design of the learning function into the design of the optimal LTV Q-filter.

IV. Combined ILC and CCILC Algorithm

Combined ILC and CCILC [7] is shown in Fig. 2. The combined controller contains three different learning controllers, the individual *x*- and *y*-axis controllers and the cross-coupled learning controller. The individual ILC controllers act on the individual axis errors, while the CC learning controller acts directly on the contour error defined as

$$\varepsilon = -c_x e_x + c_y e_y \tag{10}$$

where c_x and c_y are called the coupling gains [13]. These gains are trajectory dependent and generally time-varying. For this work, linearized coupling gains [13] are used to simplify the design process of the cross-coupled learning controller.

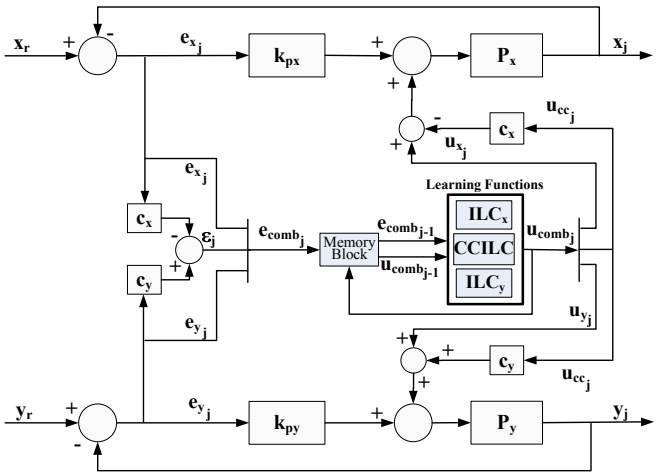

Fig. 2. Block diagram of the combined controller.

A. Learning Algorithm

The general first order learning algorithms for the combined controller are given as

$$\begin{aligned} u_{x_{j+1}}(k) &= Q_k(q)\left[u_{x_j}(k) + L_x(q)e_{x_j}(k) - c_x L_\varepsilon(q)\varepsilon(k)\right] \\ u_{y_{j+1}}(k) &= Q_k(q)\left[u_{y_j}(k) + L_y(q)e_{y_j}(k) + c_y L_\varepsilon(q)\varepsilon(k)\right] \end{aligned} \tag{11}$$

where Li, $i = [x,y,\varepsilon]$, are the learning functions for the x, y and contour controllers, respectively. Substituting the contour error ε in (11) with (10) results in a linear time-varying combined learning controller the authors are calling LTV CCILC.

$$\begin{aligned} u_{x_{j+1}}(k) &= Q_k(q)\left[u_{x_j}(k) + \left(L_x(q) + c_x L_\varepsilon(q)c_x\right)e_{x_j}(k) \right. \\ &\qquad \left. - c_x L_\varepsilon(q)c_y e_{y_j}(k)\right] \\ u_{y_{j+1}}(k) &= Q_k(q)\left[u_{y_j}(k) + \left(L_y(q) + c_y L_\varepsilon(q)c_y\right)e_{y_j}(k) \right. \\ &\qquad \left. - c_y L_\varepsilon(q)c_x e_{x_j}(k)\right] \end{aligned} \tag{12}$$

The new aspect of this approach versus [9] is the introduction of the time-varying Q-filter. Convergence analysis for this combined system is presented in the following section.

V. Stability Analysis

In this section, stability and convergence properties for the combined control system are presented in a lifted system format. Define $\hat{Y}_j$, $\hat{U}_j$ and $\hat{P}$ as lifted matrices [3].

$$\hat{Y}_j = \begin{bmatrix} y_j(1) \\ y_j(2) \\ \vdots \\ y_j(n) \end{bmatrix}; \hat{U}_j = \begin{bmatrix} u_j(0) \\ u_j(1) \\ \vdots \\ u_j(n-1) \end{bmatrix}; \hat{P} = \begin{bmatrix} p_1 & 0 & \cdots & 0 \\ p_2 & p_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_n & p_{n-1} & \cdots & p_1 \end{bmatrix} \tag{13}$$

The linear plant dynamics, (3), can now be written as

$$\hat{Y}_j = \hat{P}\hat{U}_j + \hat{d} \ , \tag{14}$$

where $\hat{d}$ is a vector of the same lifted form as $\hat{Y}_j$ that contains periodic disturbance effects. The elements in the $\hat{P}$ matrix, $\{p_1, p_2, \ldots\}$ represent the impulse response of the plant model, also known as the Markov parameters. The lifted tracking error is given by

$$\hat{E}_j = \hat{Y}_d - \hat{Y}_j \tag{15}$$

with $\hat{E}_j = [e_j(0),\ldots e_j(N-1)]^T$ and $\hat{Y}_d = [y_d(0),\ldots y_d(N-1)]^T$. Using (12), (14), and (15) the lifted form of the closed-loop iteration domain dynamics of the combined ILC system is

$$\begin{aligned} \begin{bmatrix} \hat{U}_{x_{j+1}} \\ \hat{U}_{y_{j+1}} \end{bmatrix} &= \hat{Q}_k\left(\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}\right)\begin{bmatrix} \hat{U}_{x_j} \\ \hat{U}_{y_j} \end{bmatrix} + \hat{D} \\ M_{11} &= \left(I - \left(\hat{L}_x + \hat{C}_x\hat{L}_\varepsilon\hat{C}_x\right)\hat{P}_1\right) \\ M_{12} &= \hat{C}_x\hat{L}_\varepsilon\hat{C}_y\hat{P}_2 \\ M_{21} &= \hat{C}_y\hat{L}_\varepsilon\hat{C}_x\hat{P}_1 \\ M_{22} &= \left(I - \left(\hat{L}_y + \hat{C}_y\hat{L}_\varepsilon\hat{C}_y\right)\hat{P}_2\right) \end{aligned} \ , \tag{16}$$

where

$$\hat{Q}_k = \begin{bmatrix} \lambda_{0,0} & \cdots & \lambda_{0,-N_Q} & & & & \\ \vdots & \ddots & \vdots & \ddots & & & \\ \lambda_{N_Q,N_Q} & \cdots & \lambda_{N_Q,0} & \cdots & \lambda_{N_Q,-N_Q} & & \\ & \ddots & \vdots & \ddots & \vdots & \ddots & \\ & & \lambda_{N-N_Q,N_Q} & \cdots & \lambda_{N-N_Q,0} & \cdots & \lambda_{N-N_Q,-N_Q} \\ & & & \ddots & \vdots & \ddots & \vdots \\ & & & & \lambda_{N-1,N_Q} & \cdots & \lambda_{N-1,0} \end{bmatrix} \quad (17)$$

$$\hat{L}_i = \begin{bmatrix} l_i(0) & l_i(-1) & \cdots & l_i(-N+1) \\ l_i(1) & l_i(0) & \cdots & l_i(-N+2) \\ \vdots & \vdots & \ddots & \vdots \\ l_i(N-1) & l_i(N-2) & \cdots & l_i(0) \end{bmatrix} \quad i = [x, y, \varepsilon]. \quad (18)$$

Using the lifted format, one can see that the control input converges in the iteration domain as long as

$$\left| \lambda_i \left(\hat{Q}_k \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} \right) \right| < 1 \quad (19)$$

where i is defined on the interval $i \in [1, n]$. The spectral radius condition, $\left(\max \left| \lambda \left(\hat{Q}_k M \right) \right| < 1 \right)$, satisfies the stability criteria, but does not ensure monotonicity in the iteration domain [14]. A sufficient condition for monotonic convergence is given by:

$$\left\| \hat{Q}_k \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} \right\|_i < 1 \quad (20)$$

In this equation, $\|\cdot\|_i$ is the induced norm of the matrix. Satisfying (20) with the induced 2-norm ensures monotonicity and stability of the given system.

VI. Design and Optimization of an LTV Q-filter

In this section we present a design methodology for the LTV Q-filter which is used in conjunction with the combined learning controller to generate an LTV CCILC [15]. This controller will then be applied to a Cartesian robotic test platform in Section VII. While the design results in this section are specific to the robotic test platform, the methodology is general enough to be applied to other motion control systems.

As stated previously, this work focuses on a high-performance design for the Q-filter. We assume that the learning functions have been designed using one of the generally accepted design methods mentioned in Section III.

We design the bandwidth Ω(k) of the Q-filter to have roughly the same profile as the initial tracking errors of the multiaxis system, $e_{xo}(k)$ and $e_{yo}(k)$. For example, if we consider the raster trajectory shown in Fig. 1, representative initial tracking errors for the robotic system used in Section VII are given in Fig. 3.

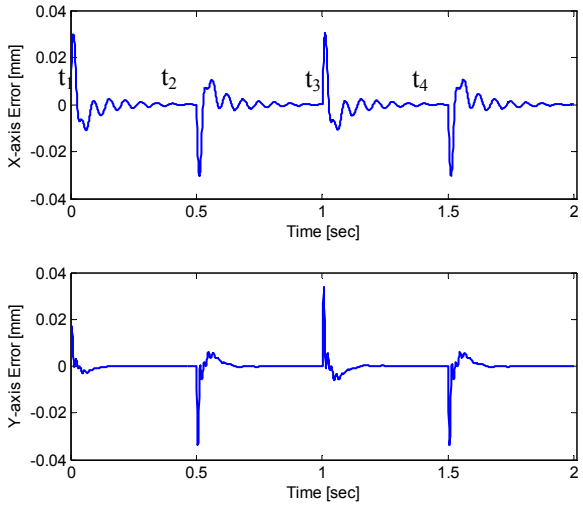

Fig. 3. Initial tracking errors without the use of learning.

Fig. 3 clearly indicates the locations where the most energy from the error signals is focused. Both the *x*- and *y*-axis error signals show relatively small error values, with intermittent areas of large error signals. These locations correspond to the high-frequency component of the error signal which is a direct result of the rapid changes in direction moving from one scanning line to another. From this information, the locations where a higher frequency bandwidth Q-filter can be used have been identified as t_1, t_2, t_3, and t_4, respectively.

The Ω profile can be defined explicitly using the variables a_1, b_1, a_2, b_2 and c as shown in Fig. 4. Two different high-frequency bandwidth peaks have been chosen for this particular LTV Q-filter. The first peak is a short-duration, high-frequency peak used to capture the high frequency dynamics. The second peak is a medium frequency peak that seeks to capture the remaining axis errors. The baseline frequency is described by the c variable and should be set as low as possible, while still achieving good tracking performance during the low frequency sections of the error signals.

The discontinuities in the profiles at each of the peaks will introduce discontinuities in the control, thus decreasing performance. Therefore, after the variables are selected, a lowpass filter, *F(q)*, is used to smooth the bandwidth profile.

$$\underbrace{\Omega'_{a_1,b_1,a_2,b_2,c}(k)}_{smoothed} = F(q)\Omega_{a_1,b_1,a_2,b_2,c}(k) \quad (21)$$

The authors choose to use the same type of Gaussian filter as is used for the LTV Q-filter, with the exception of being time-invariant. The bandwidth of *F(q)* depends on the particular application/trajectory and in this case is chosen to be 10 Hz to smooth the signal, while still permitting distinct peaks.

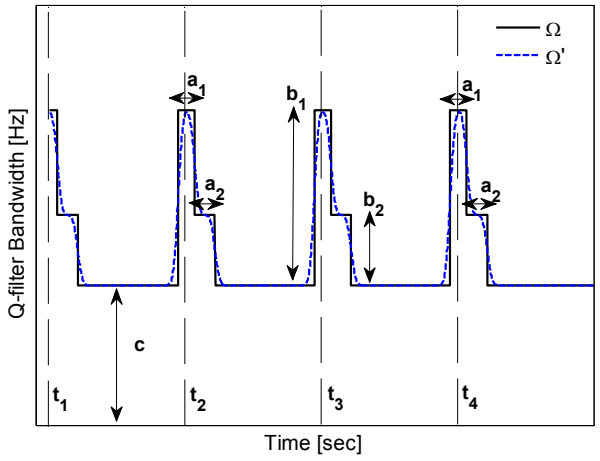

Fig. 4. Parameterized profile.

To optimally select the parameters a_1, b_1, a_2, b_2, and c, we minimize a cost function with two competing objectives. The first objective is to obtain low converged contour error which is measured by the root-mean-square (RMS) of the converged contour error:

$$J_1(a_1,b_1,a_2,b_2,c)=\frac{\|\varepsilon_\infty\|_2}{\sqrt{N}} \tag{22}$$

where ε_∞ is the converged value of the contour error defined loosely as $\varepsilon_\infty=\lim_{j\to\infty}\varepsilon_j$, as long as ε_j is stable and converges as $j\to\infty$. The second objective is fast convergence, which is defined as the number of iterations of learning required to converge within 1% of the converged contour error.

$$J_2(a_1,b_1,a_2,b_2,c)=\min_{j^*}\left\{j^*:\frac{\|\varepsilon_\infty-\varepsilon_j\|_2}{\|\varepsilon_\infty\|_2}\le 0.01,\forall j\ge j^*\right\} \tag{23}$$

The combined cost function is given by

$$J(a_i,b_i,c)=J_1(a_i,b_i,c)+\eta\cdot J_2(a_i,b_i,c),\ i=1,2 \tag{24}$$

where η is a design parameter.

To calculate the costs, J_1 and J_2, for a given set of variables, we first generate the LTV Q-filter using the smooth frequency profile Ω' with (7) and (8). The learning process is then simulated until the contour error converges. Convergence is assumed when the error change, $\|\varepsilon_{j+1}-\varepsilon_j\|$, is less than 10^{-8}, which typically occurs between 10-500 iterations for the system given in Section VII. The converged error is taken as the final value of the contour error for the simulation and J_1 and J_2 are calculated from the simulation results using (24) and (25), respectively.

Equation (24) is typically not convex; therefore we use a brute force approach for searching different combinations of variables. Choosing a moderate baseline frequency, c, based on known performance and convergence properties of LTI Q-filter designs, a broad search pattern is selected for the remaining a and b variables.

As a final step, in keeping with previous design strategies for CCILC [7], we impose the condition of monotonicity. In order to satisfy this requirement, we check monotonicity of each LTV Q-filter combination using (20). The results of the optimization search are plotted in Fig. 5.

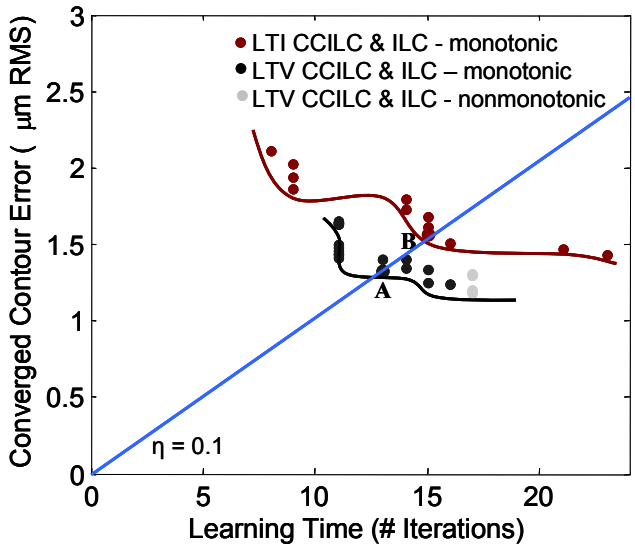

Fig. 5. Numerical optimization results showing the tradeoff between converged error and learning time.

As can be seen in Fig. 5, monotonicity is calculated for each data point and indicated by color variation on the plot. For this particular example, only monotonic LTI combined CCILC & ILC designs were considered, while the optimized combinations of the a, b, and c variables produced both monotonic and non-monotonic LTV combined CCILC & ILC controllers.

For this work, monotonically convergent controllers which fell along the η = 0.1 line were chosen for comparison purposes. These are indicated by the letters **A** and **B** in Fig. 5. Table I shows a summary of the selected LTI and LTV controllers. A comparison of the Ω profiles for the different type of filters is shown in Fig. 6. As can be seen in Table I, the LTV filter resulted in a 14% improvement in the cost function (24) over the LTI filter for the combined system.

TABLE I
COMPARISON OF LTI AND LTV Q-FILTERS

	J_1 μm	J_2 iterations	J	a_1	B_1	A_2	b_2	C
LTI combined	1.57	15	3.07	-	-	-	-	28
LTV combined	1.33	13	2.63	20	20	5	40	20
Improvement	15%	13%	14%					

Fig. 6 shows that the lower converged error and learning time is achieved by implementing higher bandwidth peaks at the locations where the trajectory rapidly changes and lower bandwidth elsewhere.

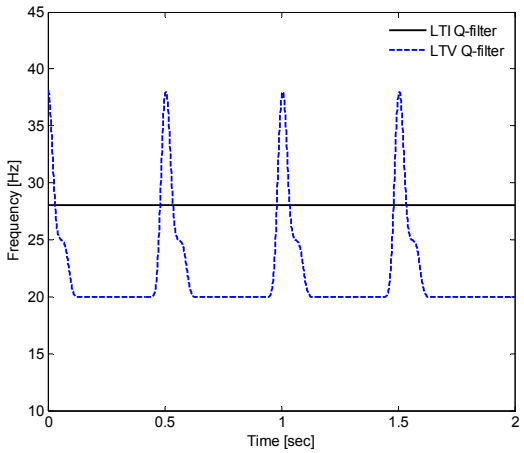

Fig. 6. Optimized LTV versus LTI Q-filter.

VII. Application: Cartesian Robotic Testbed

To demonstrate the effectiveness of this technique, an LTV CCILC controller is applied to a Cartesian robotic testbed system. The system consists of stacked *x*, *y* and *z* axes all mounted orthogonally to one another. Linear models of the *x*- and *y*-axes and proportional-integral-derivative (PID) feedback controllers for the Cartesian robot are given in the appendix.

The individual axis learning functions are designed using model inversion of the open-loop stabilized systems. While nominal plant models are used, the resonances vary depending on the location of the axes within the frame of the robotic testbed. Due to these inconsistencies, the resonances have been eliminated from the learning functions. The individual learning functions are given in the appendix.

The cross-coupled learning function selected for this system is a simple PD-type function of the form

$$L_\varepsilon(q) = k_{\varepsilon_p} q + k_{\varepsilon_d}(q-1) \ . \qquad (25)$$

PD parameters, $k_{\varepsilon_p} = .25$ and $k_{\varepsilon_d} = 22$, are tuned to provide satisfactory contour learning behavior, as well as satisfy the monotonic convergence condition (20). The reference command, shown in Fig. 1, is applied to the testbed as y_d and x_d, respectively. Optimal LTI and LTV FIR Gaussian Q-filters for $\eta = 0.1$ are obtained using the design procedure described in Section VI. Implementing the LTI combined CCILC & ILC and LTV combined CCILC & ILC controllers on the robotic testbed, performance improvements resulting from the use of the LTV design can be seen in Fig. 7.

Fig. 7 illustrates that the LTV combined controller converges fairly quickly, while the LTI combined controller requires a much longer learning time. An overview of the cost function results for the experimental testing is presented in Table II. The LTI combined learning controller converged to a neighborhood of 2.92 μm RMS contour error in 60 iterations, while the LTV combined controller converged to a neighborhood of 0.97 μm RMS contour error in approximately 20 iterations.

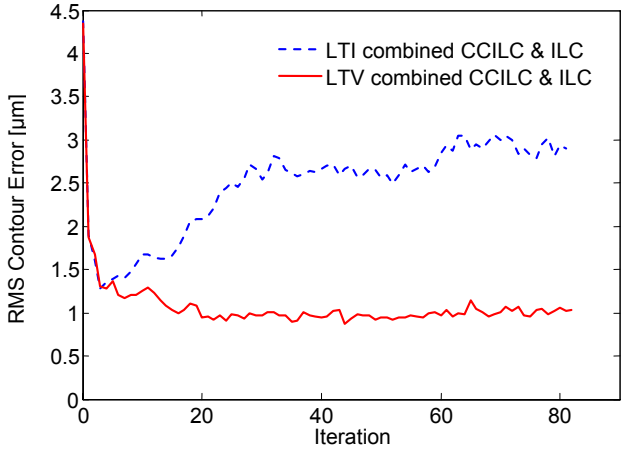

Fig. 7. Experimental RMS contour error convergence.

TABLE II
Comparison of LTI and LTV Q-filters

	J_1 μm	J_2 iterations	J	A_1	B_1	A_2	b_2	C
LTI combined	2.92	60	8.92	-	-	-	-	28
LTV combined	0.97	20	2.97	20	20	5	40	20
Improvement	67%	67%	67%					

Analogous to the simulation results, the experimental results indicate that the lowest cost function was achieved by the LTV combined learning controller with a 67% improvement over the LTI combined learning controller. The increase in the contour error resulting from unmodelled dynamics provides more opportunity for enhanced performance results from the LTV learning controller in the experimental testing. In many applications, one may have a more accurate plant model, in which case the simulation and experimental results would be more closely aligned. However, numerical, simulation and experimental results all indicate that an LTV Q-filter can improve the performance of the combined learning controller over an optimal LTI combined learning controller.

VIII. Conclusion and Future Work

This paper has presented the basic algorithm and design procedure for an LTV Q-filter to be used in conjunction with individual axis ILC and CCILC. An LTV Q-filter provides a means of increasing the bandwidth of the Q-filter and implementing high-frequency control at precise locations within a trajectory. This is particularly useful in trajectories such as a raster scanning trajectory, where the direction of the reference trajectory changes rapidly from one scanning line to the next, requiring short sections of high frequency control. Stability and monotonic convergence analysis for the LTV combined controller (LTV CCILC) was presented and used, along with a two-

part cost function of converged contour error and learning time, to select optimal LTI and LTV Q-filters. The procedure and optimally designed controllers were implemented on a robotic testbed system. Simulation and experimental results demonstrated that the LTV combined learning controller resulted in a lower cost function than the LTI combined controller.

APPENDIX CARTESIAN ROBOT MODEL

Dynamic models for the robotic system are of the form [16]

$$G(z) = \frac{\mathrm{k}(\mathrm{z}+\alpha_1)(\mathrm{z}^2-\alpha_2 z+\alpha_3)(z^2-\alpha_4 z+\alpha_5)}{(z-\beta_1)(z-1)(z^2-\beta_2 z+\beta_3)(z^2-\beta_4 z+\beta_5)} \quad . (26)$$

Numerical values for (26) are presented in Table III.

TABLE III
VALUES FOR DYNAMIC MODELS AND LEARNING CONTROLLERS

SYMBOL	QUANTITY				
Numerator	α_1	α_2	α_3	α_4	α_5
G_x	0.9604	1.981	0.9918	1.874	0.9747
G_y	1.0	1.983	0.9912	1.873	0.9547
Denominator	β_1	β_2	β_3	β_4	β_5
G_x	.9994	1.978	0.9894	1.738	0.8672
G_y	.9994	1.983	0.9911	1.87	0.9539
Gain	k				
G_x	.00083				
G_y	.00185				

PID feedback controllers used to stabilize the *x*- and *y*-axis plant models of (26) are given below

$$C_{G_x}(z) = \frac{38\left(z^2-1.941z+.9423\right)}{\left(z-1\right)\left(z-.7408\right)} \quad , \qquad (27)$$

$$C_{G_y}(z) = \frac{36.5\left(z^2-1.949z+.9506\right)}{\left(z-1\right)\left(z-.7408\right)} \quad . \qquad (28)$$

The open-loop stable plant model used in (3) and all subsequent equations is of the form

$$P_j\left(z\right) = \frac{G_i\left(z\right)}{1+G_i\left(z\right)C_{G_i}\left(z\right)}, \mathrm{j}=1,2,\ \mathrm{i}=x,y \quad . \qquad (29)$$

The individual axis learning functions, designed as modified inverse plants, are given as

$$L_x(z) = \frac{34.1(z-.9235)(z-.9666)(z^2-1.815z+.8303)}{(z-1)(z-.8187)^2(z-.7408)} \quad , (30)$$

$$L_y(z) = \frac{11.43(z^2-1.939z+.9409)(z^2-1.732z+.815)}{(z-1)(z-.8187)^2(z-.7408)} \quad . (31)$$

REFERENCE LIST

[1] Longman, Richard W., "Iterative Learning Control and Repetitive Control for Engineering Practice," *International Journal of Control*, vol. 73, no. 10, pp. 930-954, 2000.

[2] Moore, Kevin L., Dahleh, Mohammed, and Bhattacharyya, S. P., "Iterative Learning Control: a Survey and New Results," *Journal of Robotic Systems*, vol. 9, no. 5, pp. 563-594, 1992.

[3] Bristow, Douglas A., Tharayil, Marina, and Alleyne, Andrew G., "A Survey of Iterative Learning Control," *Control Systems Magazine*, vol. 26, no. 3, pp. 96-114, 2006.

[4] Huang, Yi-Cheng and Longman, Richard W., "Source of the often observed property of initial convergence followed by divergence in learning and repetitive control," *Advances in Astronautical Sciences*, vol. 90, no. 1, pp. 555-572 , 1996.

[5] Kavli, Tom, "Frequency domain synthesis of trajectory learning controllers for robot manipulators," *Journal of Robotic Systems*, vol. 9, no. 5, pp. 663-680, 1992.

[6] Bristow, D. A., Tharayil, M., and Alleyne, A. G., "Optimizing Learning Convergence Speed and Converged Error for Precision Motion Control," *Journal of Dynamic Systems Measurement and Control*, to appear.

[7] Barton, K. and Alleyne, A., "A Cross-Coupled Iterative Leaning Control Design for Precision Motion Control," *IEEE Transactions on Control Systems Technology*, to appear.

[8] Koren, Yoram and Lo Ch. Ch., "Variable-Gain Cross-Coupling Controller for Contouring," *CIRP Annals*, vol. 40, pp. 371-374, 1991.

[9] Barton, K. and Alleyne, A., "Cross-Coupled ILC for Improved Precision Motion Control: Design and Implementation," *Proceedings of IEEE ACC 2007,* 2007.

[10] Chen, YangQuan and Moore, Kevin L., "An Optimal Design of PD-type Iterative Learning Control with Monotonic Convergence," *Proceedings of the IEEE International Symposium on Intelligent Control,* pp. 55-60, 2002.

[11] Lee, J. H., Lee, K. S., and Kim, W. C., "Model-based Iterative Learning Control with a Quadratic Criterion for Time-Varying Linear Systems," *Automatica*, vol. 36, no. 5, pp. 641-657, 2000.

[12] Frueh, J. A. and Phan, M. Q., "Linear quadratic optimal learning control (LQL)," *Proceedings of the 37th IEEE Conference on Decision and Control, 16-18 Dec. 1998,* pp. 678-83, 1998.

[13] Koren, Yoram, "Cross-coupled biaxial computer control for manufacturing systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 102, pp. 265-272, 1980.

[14] Longman, Richard W. and Huang, Yi-Cheng, "The Phenomenon of Apparent Convergence Followed by Divergence in Learning and Repetitive Control," *Intelligent Automation and Soft Computing*, vol. 8, no. 2, pp. 107-128, 2002.

[15] Bristow, D., Alleyne, A., and Tharayil, M., "A Time-Varying Q-Filter Design for Iterative Learning Control," *Proceedings of the 2007 American Control Conference,* 2007.

[16] Bristow, Douglas A. and Alleyne, Andrew G., "A high precision motion control system with application to microscale robotic deposition," *IEEE Control Systems Technology*, vol. 14, no. 6, pp. 1008-1020, 2006.