World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:5, No:12, 2011

The Fundamental Reliance of Iterative Learning Control on Stability Robustness

Richard W. Longman

Abstract—Iterative learning control aims to achieve zero tracking error of a specific command. This is accomplished by iteratively adjusting the command given to a feedback control system, based on the tracking error observed in the previous iteration. One would like the iterations to converge to zero tracking error in spite of any error present in the model used to design the learning law. First, this need for stability robustness is discussed, and then the need for robustness of the property that the transients are well behaved. Methods of producing the needed robustness to parameter variations and to singular perturbations are presented. Then a method involving reverse time runs is given that lets the world behavior produce the ILC gains in such a way as to eliminate the need for a mathematical model. Since the real world is producing the gains, there is no issue of model error. Provided the world behaves linearly, the approach gives an ILC law with both stability robustness and good transient robustness, without the need to generate a model.

Keywords—Iterative learning control, stability robustness, monotonic convergence.

I. Introduction

ITERATIVE learning control (ILC) applies to situations where a control system is to perform a specific tracking maneuver repeatedly. First one can give the maneuver as the command to a feedback control system. The response can be given mathematically by a convolution integral of the command over all past time. It is very unlikely that the convolution integral over all previous commands can equal the value of the command at the present time. Hence, feedback control systems generically have deterministic errors that repeat every time a tracking command is given. ILC iteratively adjusts the command given to the feedback control system based on the error observed in the previous run (or iteration, or cycle), aiming to converge to that command that produces zero tracking error. The iterations can also eliminate the influence of disturbances that repeat each time the command is given, such as the gravity torque on a robot link as it follows a given path. See references [1,2] for more information about ILC. This paper collects observations from various previous papers by the author and co-workers, to make observation on the manner in which stability robustness plays a much more fundamental role in ILC than in more routine forms of feedback control.

II. Mathematical formulation of ILC

Consider a single-input, single-output feedback control system

R. W. Longman is with Columbia University, MC4703, New York, NY 10027, USA. RWL4@columbia.edu.

$$\begin{aligned} x_j(k+1) &= Ax(k) + Bu_j(k) \quad ; \quad k = 0,1,2,\ldots,p-1 \\ y_j(k+1) &= Cx_j(k+1) + d(k+1) \end{aligned} \tag{1}$$

where $u_j(k)$ is the command, $y^*(k)$ for $k = 1,2,\ldots,p$ is the p time step desired output history, $d(k)$ represents any repeating disturbance written as an equivalent output disturbance, j indicates the repetition or iteration number, and we aim to make $y_j(k)$ converge to $y^*(k)$ for all k as j tends to infinity.

Following [3], package the whole history of a variable during an iteration into a column vector, and denote this by an underbar:

$$\begin{aligned} \underline{y}_j &= \begin{bmatrix} y_j(1) & y_j(2) & \cdots & y_j(p) \end{bmatrix}^T \\ \underline{u}_j &= \begin{bmatrix} u_j(0) & u_j(1) & \cdots & u_j(p-1) \end{bmatrix}^T \end{aligned} \tag{2}$$

with analogous definitions for $\underline{y}^*$, $\underline{d}$, and the error history $\underline{e}_j = \underline{y}^* - \underline{y}_j$, that all use the time arguments of $\underline{y}_j$. By recursively substituting the solution of (1) for one k into the next k, one can write the output history in terms of convolution sums as

$$\underline{y}_j = \bar{A}x(0) + P\underline{u}_j + \underline{d}$$

$$\bar{A} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^p \end{bmatrix} \quad P = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^{p-2}B & \cdots & CB \end{bmatrix} \tag{3}$$

We assume that CB is not zero. If (1) came from a continuous time system fed by a zero order hold, then CB will not be zero if one is using an appropriate sample rate. Otherwise, one can adjust the definitions in (2) and (3) to account for a different delay through the system. A general linear first order iterative learning control law takes the form

$$\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_j \quad \text{or} \quad \delta_{j+1}\underline{u} = L\underline{e}_j \tag{4}$$

where L is a p by p learning control gain matrix, and the delta operator indicates $\delta_{j+1}\underline{u} = \underline{u}_{j+1} - \underline{u}_j$, and can be used on any quantity. Four possible ILC laws are: pure integral control based learning, e.g. [3,4], Euclidian norm contraction mapping law [4], the partial isometry law [4], and the quadratic cost law

[5,6]

$$L = \phi I \quad L = \phi P^T \quad L = \phi V U^T \quad L = (P^T Q P + R)^{-1} P^T Q \qquad (5)$$

where $P = U \Sigma V^T$ is the singular value decomposition of P, and $J_j = \underline{e}_j^T Q \underline{e}_j + \delta_j \underline{u}^T R \delta_j \underline{u}$ is the quadratic cost, with Q and R chosen as positive definite matrices. Combining the equations, one can obtain the error propagation equation [3]

$$\underline{e}_{j+1} = (I - PL)\underline{e}_j \qquad (6)$$
$$\underline{u}_{j+1} = (I - LP)\underline{u}_j + L[\underline{y}^* - \overline{A}x(0) - \underline{d}]$$

from which one can conclude that the ILC law L will produce convergence to zero error for all initial error histories, if and only if the absolute value of all eigenvalues of $(I - PL)$ are less than unity (i.e., the spectral radius less than unity).

By taking z-transforms one can produce the analogous equation

$$E_{j+1}(z) = [1 - G(z)L(z)]E_j(z) \qquad (7)$$

provided that the learning law has a Toeplitz structure or is approximated by such a structure. Of course, the associated frequency response thinking is not rigorous in the ILC problem, because it assumes steady state response, and ILC is a finite time problem that technically never reaches steady state in the time domain.

No matter what learning law one uses, if equation (6) shows that it converges to zero error, then the control action converges to the unique solution of the first of equations (3) for the control history after replacing the output history at repetition j, by the desired output history, $\underline{y}^*$. This solution involves the inverse of the matrix P. This inverse is guaranteed to exist analytically since P is lower triangular with nonzero elements on the diagonal. But it is very badly ill conditioned when one starts with any continuous time system fed by a zero order hold, when the pole excess is 3 or more (and the sampling is sufficiently fast), see [7-11]. These reference show how to eliminate this difficulty by asking for zero tracking error every other time step, but allowing the control law to update the zero order hold input to the system every time step. By deleting the odd numbered rows from the matrix P and from the error and output history vectors, one can convert the above control laws in (5) (except for the first one), and also convert the convergence condition in (6). One might want to double the sample rate before applying the method. The phenomenon relates to the fact that the discretization process introduces zeros, and for a pole excess of 3 or more, one or more of the zeros is outside the unit circle making the inverse of the transfer function unstable [7]. The worst singular value of the matrix decays proportional to the inverse of the magnitude of the zero outside the unit circle taken to the power equal to the dimension of the matrix [11]. One might want to call this an internal instability, but in ILC it is not technically an instability since the ILC problem is finite time and hence the control action remains finite although growing exponentially with time step, and alternating in sign each time step. The ILC can produce zero error at every time step, but the error between time steps is also growing exponentially. Hence, if the ILC really gets close to producing zero error for such systems, it is important to delete the rows as described.

III. The Basic Catch 22: Robustness vs. Model Dependence

One can find various definitions of Catch 22 on the web. One says: "A situation in which a person is frustrated by a set of circumstances that prevent any attempt to escape from them." The ILC situation is perhaps not a perfect fit, but it does contain some apparent contradictions and frustrations. By the end of this paper, perhaps we do escape.

A. The Universal ILC Law

The first ILC law in (5) comes very close to be a universal ILC law – i.e. no matter what the system is, just turn on the learning law, come back in a while, and the system will have converged to zero tracking error. Reference [4] discusses this in detail. One does need to know the time delay through the digital system, but when the system is a continuous time differential equation fed by a zero order hold, the delay will be one step unless one is using a sample rate that is unreasonably slow for the system dynamics. Hence, suppose that CB is not zero. Then we can make the following statement: Using the first ILC law in (5) will produce a learning control system that mathematically converges to zero error as the iterations progress, for all sufficiently small gains ϕ if the sign of the gain is the same as that of CB. Furthermore, as the step size gets small, the bound on the size allowed for the gain tends to infinity, so the constraint on the gain to be very easy to satisfy.

This convergence result is independent of the system dynamics appearing in the system matrix A. The convergence process in (6) works by having the first step, which is uncoupled from all other steps, converge first. Once it has converged, the next step is like a new first step, and it is uncoupled and will converge. The mathematics producing (6) is based on a linear model, but the result also applies to nearly all non-linear systems as well [12]. The main requirement is the satisfaction of a Lipschitz condition, something that one expects for differential equations governing control systems.

Furthermore, if you do not happen to know the sign of CB, one can alternate the sign of ϕ each iteration and each time step and eliminate any need for such knowledge [13].

We conclude that this learning law has extreme stability robustness. The law is almost model free, works on nearly all systems. So, experiments were performed on a robot with a desired maneuver that rotates all joints by 90 degrees simultaneously, timed so that the base joints reach the maximum speed allowed by the manufacturer. The RMS tracking error decreased by a substantial factor in the first 10 iterations and then started to grow. By iteration 15 the hardware was making so much noise we were afraid of

damaging the robot and stopped the iterations. Of course we had a mathematical proof that it would converge to zero, so we switched to simulation. The result was exponential overflow. Knowing that it must converge, we shortened the trajectory and succeeded in showing that the RMS error reached a peak of 1.1991×10^{51} radians at iteration 62,132, and then started decaying, reaching a rather satisfactory approximation of zero error, an RMS of 1.3145×10^{-48} radians, at iteration 3×10^{5}.

We conclude that this ILC law has very desirable stability robustness, but obtaining good learning transients is the issue. Waiting that many iterations might not be practical in the real world, the actuator limits are not modeled in the mathematics and one might not be able to buy actuators that can reach that many radians in a few second. So the approach appears impractical in spite of its guaranteed convergence.

B. The Use of a Model to Produce Good Learning Transients

The second and third ILC laws in (5), when substituted into the error propagation equation in (6), make the matrix in parenthesis symmetric and positive definite. As a result, for a sufficiently small gain ϕ, all eigenvalues are less than one in magnitude, and in addition all singular values can be make less than one in magnitude

$$\max[\sigma(I-PL)]<1$$

Therefore these laws produce monotonic decay of the Euclidean norm of the error every iteration. The maximum error at any time step for the robot was 9 degrees, and with this monotonic decay property, these laws will have the RMS error getting smaller from there. There is no possibility of going to 10^{51} radians on the way to zero tracking error.

C. Unusual Sensitivity to Model Error

Frequency response methods can apply rigorously to the sister field of repetitive control (RC) [4], but are approximate when applied to the finite time ILC problem. It is instructive to look at the repetitive control problem for indications of behavior. The repetitive controller can be applied to adjust the command to a feedback control system that is subject to a periodic disturbance or is executing a periodic command. Instead of looking at the error in the previous iteration, one looks at the error in the previous period. The repetitive controller is then the controller in a closed loop system where the plant is the original feedback control system whose performance is to be enhanced. When studying stability of feedback control systems it is usual to look at the phase margin associated with the frequency at which the Bode magnitude plot crosses a gain of unity. If one applies this thinking to the RC problem, one sees that the plot goes through the gain of unity around the fundamental frequency of the period involved, but also around every harmonic of the fundamental all the way to Nyquist frequency. In routine controller design, the phase margin indicates stability independent of the high frequency dynamic, provided it does not cross unity again. But in RC, it keeps crossing unity all the way to Nyquist. This indicates that the model accuracy is important at all frequencies up to Nyquist. We are rarely confident of the model accuracy at very high frequency.

Keeping the frequency transfer function version of the square bracket in (7) less than one in magnitude, is suggested in [14] as an approximate way of enforcing monotonic decay – it creates monotonic decay of every frequency component of the error in that part of the trajectory that can be considered to be properly modeled by steady state response, i.e. after roughly a settling time of the system.

In continuous time, each pole contributes 90 degrees phase lag as the frequency gets large. If the Nyquist frequency is high, one expects similar phase lag near Nyquist, although at Nyquist the transfer function is necessarily real, meaning either positive or negative, which indicates the phase must be a multiple of 180 degrees. One expects a larger phase lag in the discrete time system than in the continuous because of the average of one half step delay resulting from sampling.

Physical system models are very often missing high frequency dynamics, sometimes called parasitic poles, and in some situations, called residual modes for second order missing terms. To satisfy the design objective in the previous paragraph, the $L(z)$ in (7) should make the product $G(z)L(z)$ have a positive real part, using whatever model $G(z)$ one has. If the model is missing high frequency parasitic poles or residual modes at high frequency, the real phase of the product can very easily go beyond -90 degrees. The result is at least bad transients, if not instability.

Hence, we seem to need the model to fix bad transients, but as soon as we make use of a model, model inaccuracy is very likely to produce eventual instability of the ILC learning process [15].

D. Reaching Hardware Error Levels Smaller than the Error Level of any Feasible Model

One might ask, why do we need ILC? Can't we just create a model, and use it to find the input needed to produce the desired output? On the robot studied in [14], we did produce a model and invert it. The result was a decrease in the RMS error by a factor of 50. This same error level was reached by a simple ILC law after 4 runs, which can be easier than working hard on a model. And when the iterations were continued the error level continued to decrease to a factor of 1000. Thus, the error level reached in hardware was much better than the error level of the model we were using.

We can take this thinking one step further. The ILC in [14] decreased the RMS tracking error of all seven joints of the robot by a factor of about 1000 in roughly 12 iterations. This is a very substantial increase in accuracy for any hardware system, and it is produced without modifying hardware, simply by adjusting the command given to the system. Of course the final error level reached cannot be as low as the reproducibility level of the hardware. When one gives the

same command more than once, there will be some variation in the actual trajectories obtained, and the learning law will look at the error and take corrective action as if the same error would occur next time if the same command were given. Thus, the ILC law will amplify random errors. The factor of 1000 is therefore above this base level, when reproducibility is tested from minute to minute. But it is actually well below the reproducibility level of the hardware when measured from day to day. Hence, the ILC is fixing errors of the size of how different the robot behaves tomorrow than it behaves today. No one is going to be able to make a model that would predict such things. We don't know what is different tomorrow. Maybe the sun is coming in the window. Maybe the air conditioner is on. Maybe the door is open and there is some draft. Of course, we must keep the ILC running in order to correct for such changes.

The conclusion is that when using ILC we want to be able to reach final error levels that are better than the error level for any model we could produce of the system. The ability to do so is in fact one of the very attractive aspects of ILC, and accomplishing this is all based on stability robustness of the learning law.

1) Summary

Now back Catch 22 thinking:

(1) We have an ILC law that is mathematically guaranteed to converge to zero tracking error for almost all systems, but the bad learning transients make it impractical.

(2) We need to make use of a model in the ILC law in order to obtain well behaved, monotonic, learning transients. As long as the model is right, we get what we want.

(3) But, in order to have convergence of the learning process, the model must be accurate to within something like 90 degree phase error all the way to Nyquist frequency. Obtaining such accuracy at high frequencies is very hard, and also an unusual requirement in control. One might say that parasitic poles or residual modes are generic in the world, and they will destroy the convergence.

(4) Hence, we need the model to fix the bad learning transients, but in the process of using the model we easily destroy convergence.

(5) Furthermore, we want ILC to converge to zero error in the real world, not zero error in some model we come up with. So we want the error level to be better than the error level in the model we are using to design the ILC law. As the ILC decreases the error, it should somehow be smart enough to ignore what the model is saying, once the hardware error level is at or below the model error level.

(6) **Conclusion:** robustness of the convergence process is the key to producing the desired ILC behavior. The learning law must feel like converging to zero error in spite of the model being wrong.

IV. Robustness to Model Parameter Uncertainty

Knowing that robustness is fundamental in ILC, one might naturally look at linear robust control theory for guidance. Such methods have been used in ILC, see for example [16]. There are a number of difficulties. In so far as one relies on frequency response modeling, it does not rigorously apply to the finite time ILC problem where we insist on getting zero error every time step, including the transient parts of the trajectory at the beginning. Also, such methods want one to specify the form of the uncertainty in a specific structured way. And this can be rather confining. And finally, such methods are most likely ineffective at handling parasitic poles, i.e. singular perturbations of the model. An advantage of such an approach is that there is guaranteed convergence within the specified parameter range.

A very different approach is given in [17]. One can pick any desired probability distribution for each uncertain parameter. Then by picking random numbers from the distribution one constructs a substantial set of models. Then, instead of designing the learning law based on the nominal model as represented in matrix P, for the case of the quadratic cost ILC (the last law in (5)) one sums the associated cost given below (5) over this set of models, and finds one control law to minimize the sum. Following [18], one can do analogous averaging with the Euclidean norm and the partial isometry ILC laws in (5). Because of the nonlinear relationships, the ILC of the average cost is different than the ILC of the average system. And by averaging the cost, one obtains much improved stability robustness of the design.

In [17] 200 models were generated from the uncertainty distributions. For the nominal model based ILC law, stable and monotonic convergence was obtained for 53 models, stable but non-monotonic convergence for 111 models, all of which are practically unacceptable with very large RMS growth, and unstable learning for the remaining 36 models. In contrast, the design based on the cost averaged over the models produced stable monotonic convergence for 186 out of the 200 models, stable non-monotonic convergence for 14 models of which only 2 may be considered unacceptable, and there were no unstable results. This process is surprisingly effective at improving the stability robustness to model parameter errors. The drawback is that there is no guaranteed convergence interval. But the approach is very easy to use, and handles whatever uncertainty distribution one is interested in, and produces a simple learning law.

V. Robustness to Unmodeled High Frequency Dynamics

There is an effective answer to the question of how to make ILC convergence robust to missing parasitic poles or residual modes at high frequency [4,14,19]. One passes the computed control action for the next iteration in (4) through a zero-phase low-pass filter that cuts off the learning above some frequency. One picks this cutoff as the frequency above which one no longer has confidence in the model. One achieves stability of the learning process, but at the expense of no longer aiming for zero error above the cutoff frequency. There are some interesting issues of ill-conditioning in the effective cutoff achieved when one wants a cutoff too near Nyquist

frequency [20]. Of course, this is making use of frequency response thinking. In order to extend the thinking into shorter time trajectories, one can make the singular value cutoff as detailed in [21].

VI. Letting the Real World Take the Place of the Mathematical Model

Reference [22] developed a way to mimic the Euclidean norm contraction mapping ILC law in (5) using reversed time runs between each forward time run. This was studied in the frequency domain in that reference, and [23] reformulates it in the time domain in order to have rigorous mathematical treatment of stability. And [23] also shows how to introduce compensators that improve the learning rate at higher frequencies where the Euclidean norm law can become slow. A summary of the basic process without a compensator is as follows.

A. A Fully Robust Model Free ILC Algorithm with Good Learning Transients

(i) During iteration j, the state starts at $x_{a,j}(0)$, and the input history is given by the column matrix $\underline{u}_j$. The resulting output and error histories are given by

$$\underline{y}_{a,j} = P\underline{u}_j + A_c x_{a,j}(0) + \underline{v}_a$$
$$\underline{e}_{a,j} = \underline{y}^* - \underline{y}_{a,j}$$

Here $\underline{v}_a$ represents any disturbance that appears every time one aims to execute the desired trajectory, assumed to be the same for all iterations.

(ii) Reverse the time history of the resulting error

$$\underline{e}_{b,j} = I_r \underline{e}_{a,j} \quad ; \quad I_r = \begin{bmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 1 & 0 \\ \vdots & \iddots & \vdots & \vdots \\ 1 & \cdots & 0 & 0 \end{bmatrix}$$

(iii) Apply this reversed time error history to the system as a command, starting from initial condition $x_{b,j}(0)$, resulting in the output

$$\underline{e}_{c,j} = P\underline{e}_{b,j} + A_c x_{b,j}(0) + \underline{v}_b$$

Since this command is very different from commands aiming to produce the desired trajectory, this equation allows a different disturbance input $\underline{v}_b$ that is assumed to be the same each time one applies error signals as inputs.

(iv) The time history is again reversed in time

$$\underline{e}_{d,j} = I_r \underline{e}_{c,j}$$

The result is that the tracking error has been low pass filtered by the dynamics of the system, and the reversal of time has produced a phase lead (or lag) that is cancelled by the phase lag (or lead) produced when it next goes through the system, so that there is no net phase change.

(v) Finally, the control update for the next repetition or iteration is given by

$$\underline{u}_{j+1} = \underline{u}_j + \phi \underline{e}_{d,j}$$

with learning gain ϕ.

B. Comments

The Euclidean norm contraction mapping ILC law is a particularly appealing law. It produces symmetric matrices that govern convergence, it produces zero phase, and convergence is monotonic in the Euclidean norm of the error so that the transients are well behaved. On the other hand, these properties only hold if the model used is accurate. The method of this section replaces the model in the ILC update gains, by the real world. So we are no longer using a model that could be wrong.

The algorithm stated above does not handle the repeating disturbances, but one can modify it to do so [23]. A difficulty with the Euclidean norm ILC law is that the learning can be very slow at high frequencies, and [23] also has methods that allow the designer to introduce a compensator that speeds up the learning where needed. And finally, [23] shows how to introduce a learning cutoff if desired. Correcting high frequency errors far above the bandwidth of the system can require too large a control action to fix, making a cutoff desirable. Advantages and disadvantages of the method described here include:

(i) Convergence is guaranteed in hardware implementation provided the world behaves linearly.

(ii) There is no need to produce a system model, and no issue of how accurate the model would need to be, since the approach works with the world behavior instead of a mathematical description of the world behavior.

(iii) The transients during the learning process are well behaved, producing monotonic decay of the Euclidean norm of the error from repetition to repetition.

(iv) Gain adjustment is easy, and can be done in hardware if desired. All gains below approximately the reciprocal of the DC gain of the system, or the *M*-peak of the system when there is a resonance, are expected to produce convergence.

(v) The ILC designer can use a compensator to adjust the learning rate as a function of frequency.

(vi) The algorithms has the property that one needs to make an extra reverse time run between each normal run. This suggests that one would normally apply the approach to learn a needed input signal and once convergence is reached, turn off the ILC, and simply use the command that has been learned for all future runs. Because of these extra reverse time runs it is not so convenient to keep the learning process going all the time in order to deal with drifts in the system behavior. Also, the method asks that one extend the desired trajectory to

deal with elimination of forcing functions in the governing equations, and this also suggest doing the learning and then turning off the ILC.

VII. Conclusions

We started with a learning control law that has a mathematical guarantee of convergence to zero error for nearly all systems, linear or nonlinear. Unfortunately this guarantee is of no great use in practice, if the transients can reach 10^{51} radians error and take 10^5 iterations to converge.

Introducing a model into the control law, allowed us to guarantee good learning transients with monotonic decay of the Euclidean norm of the error. But then convergence to zero tracking error is most likely lost if there are any parasitic poles left out of the model at high frequencies. (Note, in practice the resulting unstable ILC can actually be very useful. It is very common that the tracking error decreases dramatically for some time before there is any evidence of the instability, and the instability very often has very slow growth.)

Nevertheless, in the attempt to fix the learning transients, we were forced to use information about the system, with the result that we lost the guarantees of convergence -- to really get zero error, the model has to be rather good all the way to Nyquist frequency, which is unlikely in practice.

One way to address this problem is to use a zero-phase low-pass filter cutoff to cut off the learning above the range for which the model is accurate. This means that we sacrificed zero tracking error for good learning transients.

Finally we present a method of computing iterative learning control updates that does not make use of a model, and hence cannot be leaving out high frequency dynamics, or have sufficient model error to produce instability. The method requires that one make a reverse time run between each iteration to let the real world system give you information for the learning control update. As long as the system behavior is linear, this approach allows one to have good monotonic decay of the learning transients, with guaranteed convergence to zero error, and do so without the need for a model and hence without the danger of destabilization by model error.

References

[1] Z. Bien and J.-X. Xu, editors, *Iterative Learning Control: Analysis, Design, Integration and Applications*, Kluwer Academic Publishers, Boston, 1998.

[2] J.-X. Xu and K. L. Moore, Guest Editors, Special Issue on Iterative Learning Control, *International Journal of Control*, Vol. 73, No. 10, July 2000.

[3] M. Phan and R. W. Longman, "A mathematical theory of learning control for linear discrete multivariable systems," *Proceedings of the AIAA/AAS Astrodynamics Conference*, Minneapolis, Minnesota, pp. 740-746, August 1988.

[4] R. W. Longman, "Iterative learning control and repetitive control for engineering practice," *International Journal of Control*, Special Issue on Iterative Learning Control, Bien and Xu, guest editors, vol. 73, no. 10, pp. 930-954, July 2000.

[5] D. H. Owens and N. Amann, *Norm-Optimal Iterative Learning Control*, Internal Report Series of the Centre for Systems and Control Engineering, University of Exeter, 1994.

[6] J. A. Frueh and M. Q. Phan, "Linear quadratic optimal learning control (LQL)," *Proceedings of the 37th IEEE Conference on Decision and Control*, Tampa, FL, pp. 678-683, Dec. 1998.

[7] K. Åström, P. Hagander, and J. Strenby, "Zeros of sampled systems," *Proceedings of the 19th IEEE Conference on Decision and Control*, pp. 1077-1081, 1980.

[8] P. A. LeVoci and R. W. Longman, "Intersample error in discrete time learning and repetitive control," *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference*, Providence, RI, August 2004.

[9] R. W. Longman, P. A. LeVoci, and T. Kwon, "Making the impossible possible in iterative learning control," *Proceedings of the Thirteenth Yale Workshop on Adaptive and Learning Systems*, Center for System Science, Yale University, New Haven, CT, pp. 99-106, May-June 2005.

[10] R. W. Longman, T. Kwon, and P. A. LeVoci, "Making the learning control problem well posed – stabilizing intersample error," *Advances in the Astronautical Sciences*, vol. 123, pp. 1143-1162, 2006.

[11] Y. Li and R. W. Longman, "Addressing problems of instability in intersample error in iterative learning control," *Advances in the Astronautical Sciences*, vol. 129, pp. 1571-1591, 2008.

[12] R. W. Longman, C.-K. Chang, and M. Phan, "Discrete time learning control in nonlinear systems," *A Collection of Technical Papers, 1992 AIAA/AAS Astrodynamics Specialist Conference*, Hilton Head, South Carolina, pp. 501-511, August 1992.

[13] R. W. Longman and Y.-C. Huang, "Analysis and frequency response of the alternating sign learning and repetitive control laws," *Journal of the Chinese Society of Mechanical Engineers*, vol. 21, no. 1, pp. 107-118, 2000.

[14] H. Elci, R. W. Longman, M. Phan, J.-N. Juang, and R. Ugoletti, "Discrete frequency based learning control for precision motion control," *Proceedings of the 1994 IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio, TX, pp. 2767-2773, Oct. 1994.

[15] R. W. Longman and Y.-C. Huang, "The phenomenon of apparent convergence followed by divergence in learning and repetitive control," *Intelligent Automation and Soft Computing*, Special Issue on Learning and Repetitive Control, Guest Editor: H. S. M. Beigi, vol. 8, no. 2, pp. 107-128, 2002.

[16] N. Amman, D. H. Owens, W. Rogers, and A. Wahl, "An H-infinity approach to linear iterative learning control design," *International Journal of Adaptive Control and Signal Processing,* vol. 10, pp. 767-681, 1996.

[17] K. Takanishi, M. Q. Phan, and R. W. Longman, "Multiple-model probabilistic design of robust iterative learning controllers," *Transactions of the North American Manufacturing Research Institution*, Society of Manufacturing Engineers, vol. 33, pp. 533-540, May 2005.

[18] R. W. Longman and K. D. Mombaur, "Implementing linear iterative learning control laws in nonlinear systems," *Advances in the Astronautical Sciences*, to appear.

[19] H. Elci, M. Phan, R. W. Longman, J.-N. Juang, and R. Ugoletti, "Experiments in the use of learning control for maximum precision robot trajectory tracking," *Proceedings of the 1994 Conference on Information Sciences and Systems*, Princeton, NJ, pp. 951-958, March 1994.

[20] R. W. Longman and W. Kang, "Issues in robustification of iterative learning control using a zero-phase filter cutoff," *Advances in the Astronautical Sciences*, vol. 127, pp. 1683-1702, 2007.

[21] K. Chen and R. W. Longman, "Creating short time equivalents of frequency cutoff for robustness in learning control," *Advances in the Astronautical Sciences*, vol. 114, pp. 95-114, 2003.

[22] Y. Ye and D. Wang, "Zero phase learning control using reversed time input runs," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 127, pp. 133-139, March 2005.

[23] R. W. Longman, T. Kwon, D. Wang, and Y. Ye, "Practical, model-free, completely robust learning control using reversed time input runs," *Proceedings of the 2006 AIAA/AAS Astrodynamics Specialist Conference*, Keystone, CO, Aug. 2006.