Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 1909 – 1919

doi:10.3906/elk-1203-19

Optimal iterative learning control design for generator voltage regulation system

Mohsen REZAEI ESTAKHROUIEH,* Aliakbar GHARAVEISI
Electrical Engineering Department, Shahid Bahonar University of Kerman, Kerman, Iran

Received: 05.03.2012 • **Accepted:** 18.06.2012 • **Published Online:** 24.10.2013 • **Printed:** 18.11.2013

Abstract: The purpose of this paper is to design a good tracking controller for the generator automatic voltage regulator (AVR) system. It uses an iterative learning control (ILC) algorithm as a control rule and tries to keep voltage error as low as possible, while replying to set point changes as fast as possible. Two models of ILC are discussed: P-type learning and linear quadratic (LQ) optimal design. The results of designing by LQ optimal method are compared with a Ziegler–Nichols proportional-integral-derivative (PID) controller. The effects of changing different parameters are investigated and illustrated by simulation. In order to provide better robustness for the design, a PID-ILC parallel structure is designed and simulated under system parameter changes.

Key words: Iterative learning control, linear quadratic optimal control, automatic voltage regulation

1. Introduction

Iterative learning control (ILC) is a relatively new control method that was mainly introduced to improve the performance of processes that are repeated periodically over and over. It is used when a precise trajectory tracking is needed, for instance in robotics [1–6], glycemic control in diabetes mellitus [7], hard disk position control [8], iterative system identification [9], human learning behavior [10–12], industrial processes [13–16], electropneumatic servo systems [17], injection molding processes, food production facilities, robotic assembly lines, chemical batch reactors [18], and even for density control of freeway traffic flow [19]. The idea of the ILC algorithm was first introduced by Arimoto et al. in 1984 [20]. The idea was to use the previous iteration data to improve the current iteration process. In other words, it uses the previous iteration control signal and error signal and tries to make a better control signal for the next iteration, and it keeps the error of the next iteration as low as possible the whole time. It can be shown that if some certain conditions are met, after several iterations the tracking error in every moment will tend to zero. This is a fundamental difference between this algorithm and conventional controllers such as the proportional-integral-derivative (PID) controller. By PID controller, error will tend to zero in a steady state. However, in ILC, if we have a precise model of the plant, we can design the algorithm such that the error will be zero from the beginning (after one sample). In other words, ILC makes both steady state error and transient error (or overshoot) zero. This property is called perfect tracking [3]. For this reason, several algorithms of ILC were developed for applications of precision motion and tracking control. Many researchers have worked on different algorithms and have shown new applications of it (for a brief history, see [1]).

Since in a generator, when a voltage drop (or rise) occurs, it is necessary to regulate the voltage quickly

*Correspondence: m.rezaei@math.uk.ac.ir

and precisely, it seems that ILC can be a good choice as an automatic voltage regulator (AVR) control law. The results of simulations prove this.

There may be an argument as to why ILC is used on an AVR system, which does not seem a repetitive process. Applications of ILC in nonrepetitive processes have already been addressed in works such as [21–26]. In our case, since the load usually has a periodic daily or weekly manner, with a fraction of nonperiodic changes, its profile can be used in the ILC problem to generate an optimum signal for 24 h of a day. However, in case the load changes, the stored ILC control signal can be changed proportionally to the load changes and a fraction of the previous control signal is applied to the plant by the intelligent computer controller. In order to have better robustness, a PID controller is added in parallel to ILC at the end of study.

Since we use a software model of a real AVR plant, we can run it as much as we wish. After then designing the ILC, it can be tested on a real system. This means that in some plants that are not repetitive, an offline repetitive design can be done for their approximate models and, after convergence, it can be tested on a real system for verifying its functionality and minor adjustments. Here, an offline method is used to design the ILC. It is considered that the approximate system model is available and one can run the ILC algorithm offline as much as needed until reaching convergence. This is mainly because in practice one is not allowed to run an algorithm on the real plant for many repetitions because of security issues related to the possible divergence of the control action and the high price of such an action. After convergence in an offline simulation, the achieved signal can be tested on the real plant and adjusted for minor changes between the real system and its modeling. However, in the case of parameter changes in system, a periodic system identification method should be used to identify new system parameters and change the ILC settings consequently. This is similar to adaptive control methods.

This paper is organized as follows: the ILC problem definition and some related theorems are presented in Section 2. The AVR and the plant model are then introduced in Section 3. Simulation results are shown in Section 4 and are compared with a Ziegler–Nichols PID, and the effect of varying different parameters are discussed. Finally, Section 5 presents the conclusions.

2. ILC algorithm

The main ILC algorithm was invented for discrete systems. A discrete model is also used here to describe the plant:

$$Y_j = PU_j + d \tag{1}$$

where

$$Y_j = \begin{bmatrix} y_j(1) \\ \vdots \\ y_j(N) \end{bmatrix}, U_j = \begin{bmatrix} u_j(0) \\ \vdots \\ u_j(N-1) \end{bmatrix} d = \begin{bmatrix} d(0) \\ \vdots \\ d(N-1) \end{bmatrix} \tag{2}$$

This is called the lifted system representation. $y_j(k)$ shows the output in iteration j and the kth sample. $u_j(k)$ is the input to the system, d is disturbance, and P is the static map that maps input U to output Y. It can be shown that P is a matrix of the Markov parameters of the system [27]:

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

where A, B, and C are the main system describing matrices:

$$\begin{cases} X_j(k+1) = AX_j(k) + BU_j(k) \\ Y_j(k) = CX_j(k) + d \end{cases} \quad k \in [0, N] \tag{4}$$

Markov parameters can also be obtained from the impulse response of the plant. The ILC update law is defined as below:

$$U_{j+1} = L_u U_j + L_e E_j \tag{5}$$

where

$$E_j = Y_d - Y_j \tag{6}$$

L_u and L_e are called learning gain matrices. Y_d is the desired trajectory. If we choose learning gains properly, the algorithm will converge. This is stated in the following theorem. A schematic of the ILC controller is shown in Figure 1.

Theorem 1 *For the plant and ILC update law below:*

$$\begin{cases} Y_j = PU_j + d \\ U_{j+1} = L_u U_j + L_e E_j \end{cases} \tag{7}$$

if

$$(\lambda_j(L_u - L_e.P)) < 1 \tag{8}$$

the algorithm asymptotically converges, and the control and error vector will converge to fixed vectors U_∞ and E_∞ such that [28]:

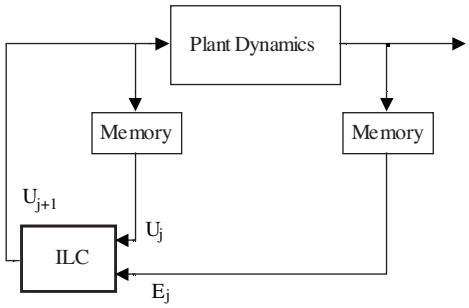

Figure 1. ILC controller.

$$\begin{cases} U_\infty = (I - L_u + L_e P)^{-1} L_e Y_d \\ E_\infty = (I - P\left(I - L_u + L_e P\right)^{-1} L_e) Y_d \end{cases} \tag{9}$$

Note that if $L_u = I$, the error will converge to zero. If we then choose L_e as a diagonal matrix like this:

$$L_e = \begin{bmatrix} \gamma & 0 & \cdots & 0 \\ 0 & \gamma & \cdots & 0 \\ \vdots & \ddots & & \\ 0 & 0 & \cdots & \gamma \end{bmatrix} \tag{10}$$

the convergence conditions will be simply:

$$|1 - p_0.\gamma| < 1 \tag{11}$$

where $p_0 = CB$ is the first Markov parameter. This is a simple way of designing an ILC controller, which is called P-type learning [29,30]. The only thing we need from the plant is one parameter (p_0), but the problem is that in many plants the first Markov parameter is zero and so this method does not guarantee convergence. The solution is choosing other methods. The following theorem helps to solve this problem.

Theorem 2 *Suppose the following cost function:*

$$J_{j+1} = E_{j+1}^T Q E_{j+1} + U_{j+1}^T R U_{j+1} + \delta U_{j+1}^T S \delta U_{j+1} \tag{12}$$

where $\delta U_{j+1} = U_{j+1} - U_j$ *and* Q, R, *and* S *are positive semidefinite weighing matrices. The* L_u *and* L_e *matrices that make this cost function minimum are then:*

$$\begin{cases} L_u = (P^T Q P + R + S)^{-1} \left(P^T Q P + S\right) \\ L_e = (P^T Q P + R + S)^{-1} P^T Q \end{cases} \tag{13}$$

and error will monotonically asymptotically converge to:

$$E_\infty = \left(I - P \left(P^T Q P + R\right)^{-1} P^T Q\right) (Y_d - d) \tag{14}$$

This is the basis of the linear quadratic (LQ) optimal design of ILC. The role of Q *matrix is weighing error. It will cause the error to be low enough.* R *is a weighting matrix that makes the control signal of ILC low so that saturation does not happen. The* S *matrix ensures that we have a smooth change between iterations and, as a result, big peaks in error do not occur. In fact,* S *is related to the convergence speed.*

It should be noted that in P-type learning, there is no need for system identification and only the bond of the first Markov parameter is sufficient. After running, the algorithm learns the dynamics of the plant and adjusts the control signal. This may seem to be a benefit because of its simplicity, but on the other hand, since monotonic convergence is not guaranteed, there may be large transients before convergence [29]. It may also require much iteration to converge. LQ optimal design, on the other hand, requires system identification, but it has a monotonic convergence and usually has faster convergence. The system identification methods can be used in order to obtain the P matrix.

In this paper, LQ optimal design is used for designing ILC and the effects of different factors on performance will be discussed in Section 4.

3. AVR system

In a generator, it is necessary to keep the output voltage as constant as possible. There are many disturbances in a power system, like temperature rise, speed change, load change, and power factor change, which all affect the voltage level of the generator [31]. It is thus necessary to keep the voltage level constant.

In response to active power changes, the input fuel to the turbine (steam, water) must be increased to match the demanded power, or else the frequency of the network decreases. This can be done automatically by a system called automatic generation control. In addition to this, variation in reactive power may change the voltage level. The exciter should thus be regulated in order to match the voltage drop (or rise). There

must be a voltage regulator device in order to adjust the voltage according to the new conditions. The voltage regulator can be controlled automatically or manually by tap-changing switches, a variable autotransformer, and an induction regulator [32]. When controlling manually, an operator reads the voltage with a voltmeter and decides what to do, but it is not always possible, especially in modern large networks. The AVR system is designed for this purpose. The main duty of AVR in a power plant is to maintain generator voltage automatically [20], which affects the security of the system. As discussed in [32], in general there are 3 important tasks for AVR system:

1. Better voltage regulation,

2. Stability improvement,

3. Reduction of overvoltage on loss of load.

The AVR circuit senses voltage changes and automatically adjusts the exciter field in order to cope with new conditions by changing its output, which is a set point for the plant.

The model used here is a linear synchronous generator that consists of 4 main parts: amplifier, exciter, generator, and sensors, as presented in [31].

Amplifier: Has a simple first order model like this:

$$\frac{V_R(s)}{V_e(s)}=\frac{K_A}{1+\tau_A s} \tag{15}$$

with K_A ranging from 10 to 400 and τ_A ranging from 0.02 to 0.1 s.

Exciter: Proposed as:

$$\frac{V_F(s)}{V_R(s)}=\frac{K_E}{1+\tau_E s} \tag{16}$$

with K_E ranging from 10 to 400 and τ_E ranging from 0.5 to 0.1 s.

Generator: The model is considered as:

$$\frac{V_t(s)}{V_F(s)}=\frac{K_G}{1+\tau_G s} \tag{17}$$

with K_G ranging between 0.7 and 1.0 and τ_G from 1.0 to 2.0 s (full load and no load).

Sensors: Sensors have a fast first-order dynamic compared to the system time constants and are neglected here.

For the AVR controller, we can use classic controllers like the PID or other methods. Several papers have used new methods for doing so and have obtained better results, such as [32]. In this paper, we use ILC and suppose that the control signal is saved in the AVR circuits and that each time the AVR senses a voltage change, it uses a multiplication of the stored signal proportional to the error that occurs. A schematic of the AVR system is shown in Figure 2.

The following ILC controller is designed for this system and then design parameters are changed to show the effects of them on the performance.

4. Simulation

The model that is considered for the simulation is shown in Figure 2. First, a Ziegler–Nichols PID is designed and implemented in the model as a controller [33]. An ILC controller will then be designed. We investigate the effect of changing these parameters in the performance: Q, R, and set point change (in a constant pattern). Simulation results for each of them are presented to observe the effect.

4.1. Comparison with PID

A PID controller is designed and applied to the plant. The parameters for PID are: $K_p = 1.14$, $T_i = 0.5$, $T_d = 0.125$ with an extra far pole for the derivative term in $S = -100$, as a filter, to avoid noise amplification and the sudden kicking of derivative action in the case of set point changes. The ILC algorithm is then applied with $Q = R = S = I$. Figures 3, 4, and 5 show the results. The output settles down in about 0.5 s, while for the PID this time is about 2.5 s. The overshoot of the ILC output is lower than that of the PID, with lower control signal overshoot. This performance is also comparable to the improved PID methods, such as particle swarm optimization (PSO) design of a PID for the AVR system in [34]. In that work, Gaing designed a PSO-PID for the AVR system and the optimized system had a settling down time of about 0.4–0.5 s.

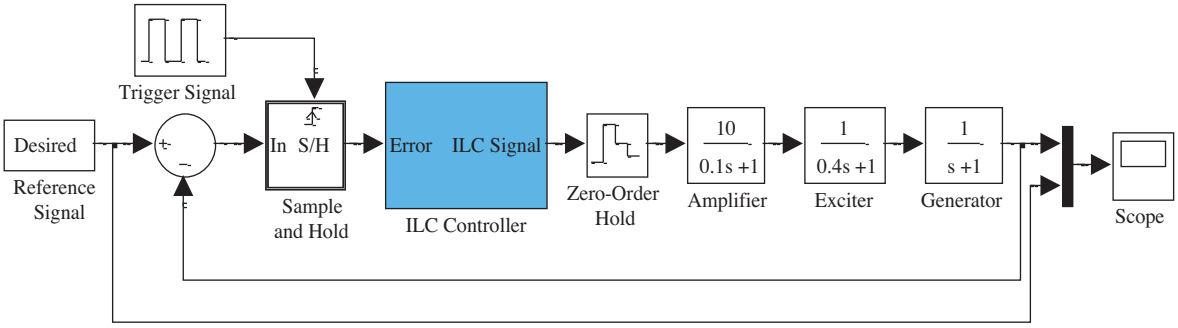

Figure 2. AVR system.

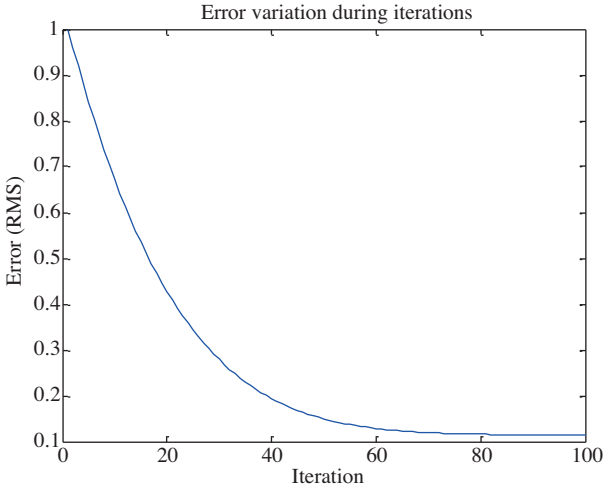

Figure 3. ILC algorithm convergence.

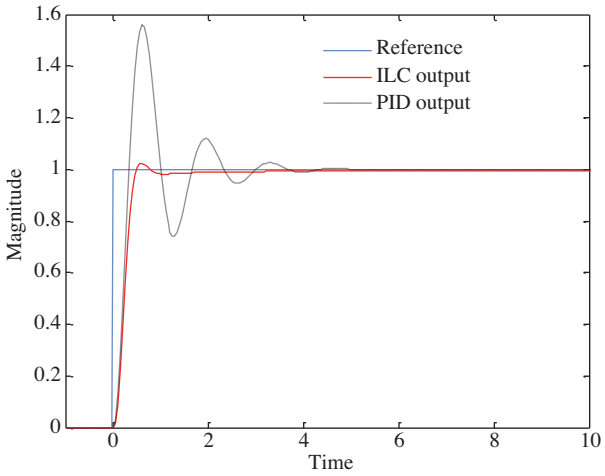

Figure 4. Comparing PID with ILC output.

4.2. Effect of changing Q

In this simulation, $Q = q.I$ is changed 3 times: $q = 1, 10, 100$. As is seen in Figures 6 and 7, the speed of the output increases as q increases, but overshoot in output and control signal increases, too. This is because the weight of the error is increased such that total error in one period must decrease and the process speeds up. It can be concluded that Q regulates the speed of output, but at the price of higher overshoot.

4.3. Effect of changing R

We suppose that $R = rI$ and change $r = 1, 10, 100$. It is observed that with higher R, output is slower and overshoots in both output and control signals reduce (Figures 8 and 9).

Thus, by changing R, we can adjust the control signal overshoot by the adverse effect of slowing down the process.

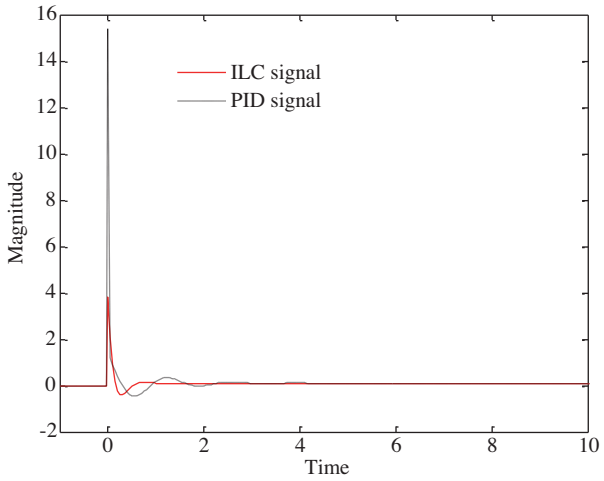

Figure 5. Control effort.

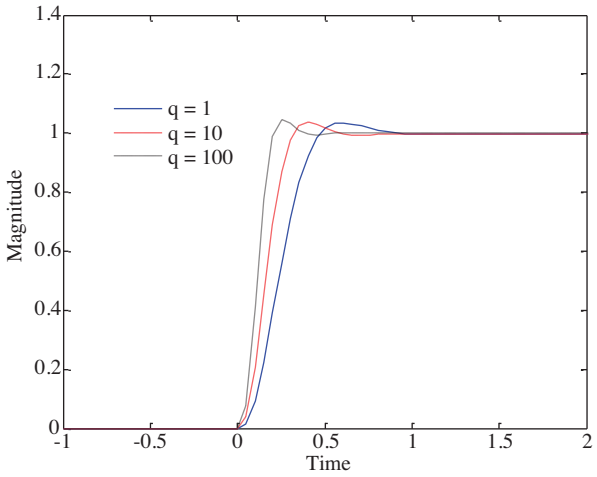

Figure 6. Output for q = 1, 10, 100.

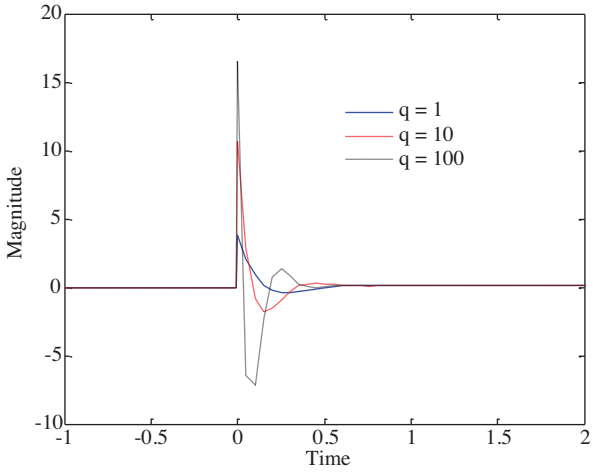

Figure 7. Control effort for q = 1, 10, 100.

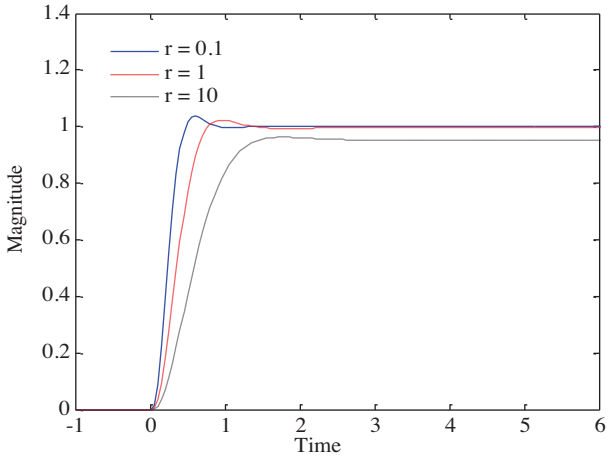

Figure 8. Output for r = 0.1, 1, 10.

4.4. Effect of changing the set point

If it is known that in a certain time the set point changes in every cycle of working, the ILC can improve the tracking problem dramatically. Even without knowing the exact time, since the plant is linear, when a sudden set point change occurs we can apply a multiplication of the control signal that is obtained for the unit step (proportional to the set point change). The results for a set point change are shown in Figures 10 and 11. As is seen, the ILC acts much better than the standard PID, while having a lower overshoot than the PID. As mentioned before, in general the ILC algorithm responds very well to set point changes and disturbances that are repeated in every period. However, in the case of unpredictable set point changes or varying disturbances, it is better to use it with another controller like the PID to achieve better performance.

4.5. Using ILC in parallel with PID

One way to provide the ILC with better robustness is to use it with a PID controller either in series or in parallel [29]. Since the PID usually has good robustness and the ILC has good performance, it is expected that their combination has both good robustness and performance. Here a parallel PID-ILC combination is used to study the robustness of the response against system parameter changes, as shown in Figure 10. In this configuration, a conventional ZN-PID is designed for the loop. The ILC is then designed and its control signal is added to that of the PID.

First the system is simulated under normal conditions with no parameter changes in the system. The comparison of the simulation results are given in Figure 11. It is seen that both the ILC and PID-ILC act well in this situation.

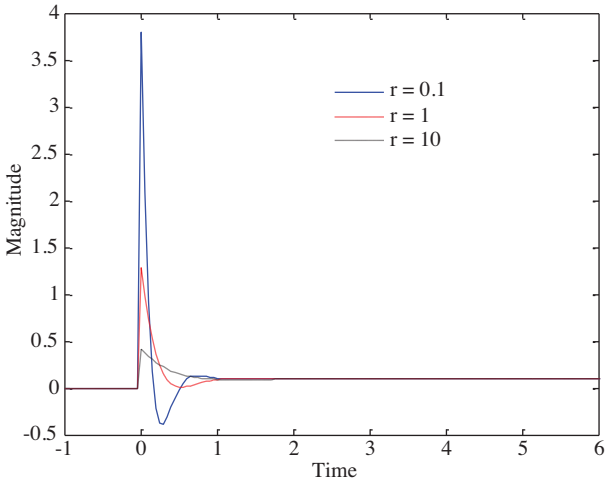

Figure 9. Control effort for r = 0.1, 1, 10.

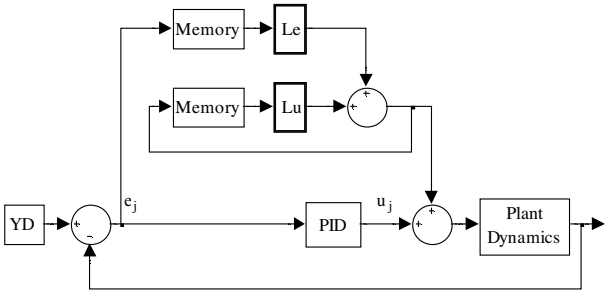

Figure 10. Parallel PID-ILC configuration.

The system is then simulated with 10% typical change in generator parameters (gain is set to 1.1 and time constant is changed to 0.9). It is supposed that the ILC is not trained again for learning the new system dynamics and its control signal is applied with no change. The results are shown in Figure 12. As is shown, the PID-ILC configuration has the best response. This is because of the nature of the PID included in the controller, which gives better robustness to the plant.

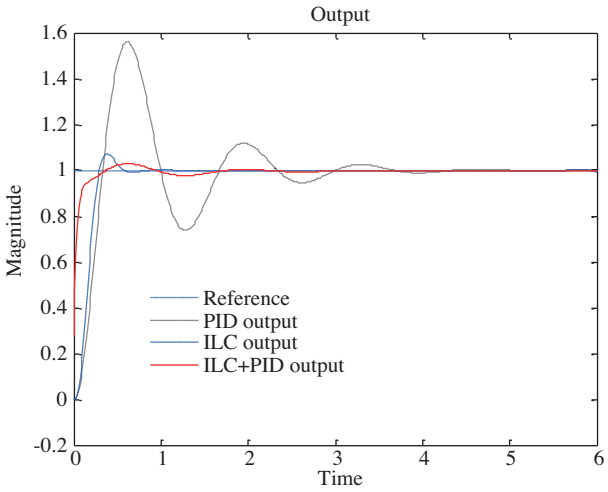

Figure 11. Simulation result.

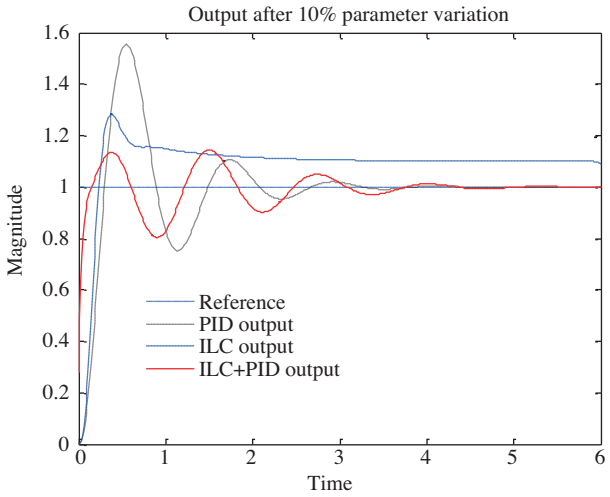

Figure 12. Simulation result with 10% parameter variation.

5. Conclusion

The definition of ILC was stated and some theorems relating to the design of the ILC controller were proposed. A linear AVR model was then considered and simulations were done. The effectiveness of the ILC was shown by simulation and compared with a standard PID. The effect of the important parameters in designing an ILC controller were simulated and discussed, such as R, Q, and set point change. Finally, a parallel PID-ILC structure was used to give better robustness for control action. It was shown that for good design and acceptable performance, there is a compromise between these parameters that a designer with good insight and experience can handle.

References

[1] A. Tayebi, S. Islam, "Adaptive iterative learning control for robot manipulators: experimental results", Control Engineering Practice, Vol. 14, pp. 843–851, 2006.

[2] D. Wang, C.C. Cheah, "An iterative learning control scheme for impedance control of robotic manipulators", International Journal of Robotics Research, Vol. 17, pp. 1091–1104, 1998.

[3] J.H. Moon, T.Y. Doh, M.J. Chung, "An iterative learning control scheme for manipulators", Intelligent Robots and Systems, Vol. 2, pp. 759–765, 1997.

[4] E. Tatlıcıoğlu, "Learning control of robot manipulators in the presence of additive disturbances", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 19, pp. 705–714, 2011.

[5] L. Hladowski, K. Galkowski, Z. Cai, E. Rogers, C.T. Freeman, P.L. Lewin, "Experimentally supported 2D systems based iterative learning control law design for error convergence and performance", Control Engineering Practice, Vol. 18, pp. 339–348, 2010.

[6] C.T. Freeman, D. Tong, K. Meadmore, Z. Cai, E. Rogers, A.M. Hughes, J.H. Burridge, "Phase-lead iterative learning control algorithms for functional electrical stimulation-based stroke rehabilitation", Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, Vol. 225, pp. 850–859, 2011.

[7] Y. Wang, E. Dassau, F.J. Doyle, "Closed-loop control of artificial pancreatic-cell in type 1 diabetes mellitus using model predictive iterative learning control", Biomedical Engineering, Vol. 57, pp. 211–219, 2010.

[8] C.I. Kang, C.H. Kim, "An iterative learning approach to compensation for the servo track writing error in high track density disk drives", Microsystems Technologies, Vol. 11, pp. 623–637, 2005.

[9] A. Fujimori, S. Ohara, "Multi-variable iterative learning identification and its application to estimation of aerodynamic derivatives in an aircraft model", Transactions of the Japan Society for Aeronautical and Space Sciences, Vol. 55, pp. 123–132, 2012.

[10] S.H. Zhou, D. Oetomo, Y. Tan, E. Burdet, I. Mareels, "Human motor learning through iterative model reference adaptive control", International Federation of Automatic Control, pp. 2883–2888, 2011.

[11] A.C. Djelassi, P. Polet, F. Vanderhaegen, "A comparative study between subjective and objective results of predicting human behaviour method based on the expected utility and the iterative learning control: application to the car driving", Systems, Man and Cybernetics, Vol. 5, pp. 4360–4367, 2006.

[12] P. Polet, F. Vanderhaegen, S. Zieba, "Iterative learning control based tools to learn from human error", Engineering Applications of Artificial Intelligence, Vol. 25, pp. 1515–1522, 2012.

[13] A. Su, R.M. Cao, H.X. Zhou, "Research on the iterative learning control method for linear motor driven compressor", Electrical Machines and Systems, pp. 1–4, 2011.

[14] B. You, M. Kim, D. Lee, J. Lee, J.S. Lee, "Iterative learning control of molten steel level in a continuous casting process", Control Engineering Practice, Vol. 19, pp. 234–242, 2011.

[15] J. Song, H. Wang, H. Shi, S. Zhang, "Iterative learning control and its application to batch process optimization", Third Pacific-Asia Conference on Circuits, Communications and System, pp. 1–4, 2011.

[16] M. Pandit, S. Baque, "Learning control of cyclic production processes", Emerging Technologies and Factory Automation Proceedings, pp. 64–70, 1997.

[17] S.J. Yu, X.D. Qi, R.C. Han, F. Pan, "Practical design of an iterative learning-sliding mode controller for electro-pneumatic", International Journal of Information Technology, Vol. 11, pp. 1–9, 2004.

[18] J. Ratcliffe, P. Lewin, E. Rogers, J. Hätönen, T. Harte, D. Owens, "Measuring the performance of iterative learning control systems", International Symposium on Intelligent Control, pp. 1213–1218, 2005.

[19] Z. Hou, J.X. Xu, J. Yan, "An iterative learning approach for density control of freeway traffic flow via ramp metering", Transportation Research Part C: Emerging Technologies, Vol. 16, pp. 71–97, 2008.

[20] S. Arimoto, S. Kawamura, F. Miyazaki, "Bettering operation of dynamic systems by learning: a new control theory for servomechanism or mechatronics systems", Decision and Control, pp. 1064–1069, 1984.

[21] W.J.R. Velthuis, Learning Feed-Forward Control, Theory, Design and Applications, PhD, University of Twente, Enschede, the Netherlands, 2000.

[22] W.E. Dixon, E. Zergeroglu, D.M. Dawson, B.T. Costic, "Repetitive learning control: a Lyapunov-based approach", Systems, Man and Cybernetics, Vol. 32, pp. 538–545, 2002.

[23] T.J.A. de Vries, W.J.R. Velthuis, "Toward exploiting the benefits of ILC in non-repetitive motion applications", Iterative Learning and Control Workshop and Roundtable, pp. 87–88, 1998.

[24] X.E. Ruan, Z.Z. Bien, K.H. Park, "Decentralized iterative learning control to large-scale industrial processes for nonrepetitive trajectories tracking", Systems, Man, and Cybernetics, Vol. 38, pp. 238–252, 2008.

[25] O.V. Iftime, M. Verhaegen, "Model-based learning control with non-repetitive initial conditions", International Journal of Intelligent Systems Technologies and Applications, Vol. 2, pp. 161–173, 2007.

[26] C.T. Freeman, Y. Tan, "Iterative learning control with mixed constraints for point-to-point tracking", Control Systems Technology, pp. 1–13, 2012.

[27] R. Tousain, E. Van der Meché, O. Bosgra, "Design strategy for iterative learning control based on optimal control", Decision and Control, Vol. 5, pp. 4463–4468, 2001.

[28] K.L. Moore, Y.Q. Chen, H.S. Ahn, "Iterative learning control: a tutorial and big picture view", Decision and Control, pp. 2352–2357, 2006.

[29] M. Tharayil, Switched Q-Filters in Repetitive Control and Iterative Learning Control, PhD, Graduate College of the University of Illinois, Urbana-Champaign, IL, USA, 2005.

[30] K. Abidi, J. Xu, "Iterative learning control for sampled-data systems: from theory to practice", Industrial Electronics, Vol. 58, pp. 3002–3015, 2011.

[31] F. Naderi, A.A. Gharaveisi, M. Rashidinejad, "Optimal design of type 1 TSK fuzzy controller using GRLA for AVR system", Power Engineering, pp. 106–111, 2007.

[32] M. Htay, K.S. Win, "Design and construction of automatic voltage regulator for diesel engine type stand-alone synchronous generator", World Academy of Science, Engineering and Technology, Vol. 42, pp. 652–658, 2008.

[33] J.G. Ziegler, N.B. Nichols, "Optimum setting for automatic controllers", American Society of Mechanical Engineers, Vol. 64, pp. 759–768, 1942.

[34] Z. Gaing, "A particle swarm optimization approach for optimum design of PID controller in AVR system", Energy Conversion, Vol. 19, pp. 384–391, 2004.