ITERATIVE LEARNING CONTROL FOR DISCRETE TIME, LINEAR TIME INVARIANT, SINGLE INPUT SINGLE OUTPUT SYSTEM

A Project Report

submitted by

B. SSVANUSHA

in the partial fulfilment of the requirements

for the award of the degree of

DUAL DEGREE (B. TECH& M. TECH)

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

MAY 2019

THESIS CERTIFICATE

This is to certify that the thesis titled **Iterative learning control for discrete time, LTI, SISO System** submitted by **B. Anusha Subrahmanyam** to the Indian Institute of Technology Madras, for the award of the degree of **Dual Degree (B. Tech & M. Tech)**, is a bonafide record of the research work done by her under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other institute or university for the award of any degree or diploma.

Dr. Ramkrishna Pasumarthy
Research Guide
Associate Professor
Dept. Of Electrical Engineering
IIT-Madras, 600 036.

Place: Chennai

Date: May, 2019.

ACKNOWLEDGEMENTS

I would like to convey warm regards and deepest gratitude to my guide, professor Dr. Ramkrishna Pasumarthy, for giving me an opportunity to work under him. I thank him for his constant and encouraging outlook and guidance towards the course of work done for this project.

I would also like to thank Sailash and Vijayanand, for their valuable guidance throughout the project. I am very grateful to my family for their constant support and motivation. I am also thankful for my friend shahla for her moral support and encouragement. I finally thank IIT madras for giving me this opportunity

ABSTRACT

KEYWORDS: Iterative learning control, Asymptotic stability, Performance, transient response, Robustness.

Iterative learning control (ILC) is a technique for improving the transient response and tracking performance of process, machines or systems that execute the same trajectory over and over under same operating conditions. The objective of ILC is to improve performance by incorporating error information into the control for subsequent iterations. In ILC, improvements are made to the input signal after each trial until desired output is acquired.

The outline of this thesis is separated into four major parts. These are system representation, analysis, design and implementation example. ILC systems are developed using mainly two representations, Time domain and Frequency domain. The Analysis section deals with four basic consequential topics are stability, performance, Transient learning behaviour and robustness.

We analyse the four different design methods, PD type, plant inversion, H∞ and quadratically optimal control. Finally, the design and implementation of an ILC controller for microscale robotic deposition.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

IITM	Indian Institute of Technology Madras
B. Tech	Bachelor of Technology
M. Tech	Master of Technology
ILC	Iterative Learning Control
LTI	Linear and Time Invariant
SISO	Single Input, Single Output
AS	Asymptotically Stable
PD-Type	Proportional and Differential type
***μ*-RD**	Microscale Robot deposition

CHAPTER I

INTRODUCTION

Iterative learning control is based on the notion that the performance of a system that executes the same task multiple times can be improved by learning from previous executions. For example, a basketball player shooting a free throw from a fixed position and an industrial robot that has to repeat the same task.

ILC is closely related to repetitive control and periodic control. Repetitive control is intended for continuous operation whereas ILC is intended for discrete type. Discrete type is natural domain for ILC because it clearly requires the storage of past iteration data, which is typically sampled.

The difference between Repetitive and ILC is the setting of initial conditions for each trial. In ILC, the initial conditions are set to same value on each trial where as in RC, the initial condition are set to the final conditions of previous trial. The resetting of initial conditions differences leads to different analysis methods and different results.

The goal of ILC is to generate a feedforward control that tracks a specific reference or rejects a repetitive disturbance. ILC approach is motivated by the observation that, if the system controller is fixed and initial conditions remains same each time it executes, then any errors in the output response will be repeated during each operation.

The main Objective of ILC is to improve the performance by incorporating error information into the control for the succeeding iterations and achieve high performance with low tracking error despite large model uncertainties and repeating disturbances.

ILC finds its use in Industrial applications where mass production requires repetition. ILC algorithm can also be applied to the systems that do not have identical repetition, but where the different tasks can be equalized by a time scale formation.

ILC Vs Feedback and Feed forward design

Feedback control:

A feedback control reacts to inputs and disturbances, and therefore always has a lag in transient transactions. It can accommodate variations or uncertainties in the system model. Robustness and stability problems can occur. Performance is good at the first trial but there is no improvement later on.

Feed forward:

Feed forward control can eliminate the transient lag but only for known or measurable signals, such as the reference and typically not for disturbances. A feed forward controller performs well only to the extent that the system is accurately known.

ILC:

It generates a feed forward control that can track a specific reference or rejects a repeating disturbance. ILC is anticipatory and can compensate for exogenous signals such as a repeating disturbance, in advance by learning from previous iterations. ILC learns feed forward through practice and it is highly robust to system uncertainties.

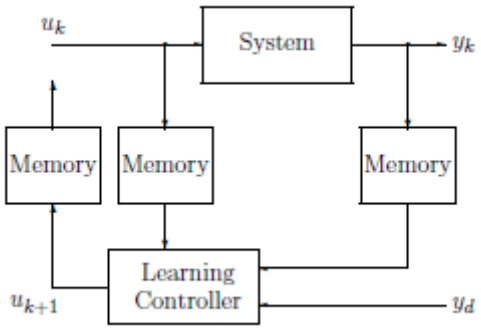


Figure 1.1: An ILC algorithm Block diagram representation

ILC System description:

Consider the following discrete time, LTI, SISO system

$$y_j(k) = p(q)\, u_j(k) + d(k) \qquad (1)$$

Where k corresponds to time index and j corresponds to iteration index and

q is forward time shift operator.

$$qx(k) \equiv x(k+1)$$

$$q^{-1}x(k) \equiv x(k-1)$$

where y_j is the output; u_j is the control input and d is an exogenous signal that repeats after each iteration. p(q) is a proportional rational function of q and has a delay, or equivalently of relative degree m. we assume P(q) is asymptotically stable. A system delay m = 1 is assumed in most of the cases.

Next consider the N – sample sequence of control inputs, outputs and desired outputs.

$$U_j(k),\ k \in \{0,1,2\ldots\ldots,N\text{-}1\}$$

$$y_j(k),\ k \in \{m, m+1\ldots\ldots, N+m\text{-}1\}$$

$$d(k),\ k \in \{m, m+1\ldots\ldots, N+m\text{-}1\}$$

$$y_d(k),\ k \in \{m, m+1\ldots\ldots, N+m\text{-}1\}$$

The performance or error signal is defined as the difference between the desired output and the acquired output. It can be written as

$$e_j(k) = y_d(k) - y_j(k)$$

The Plant can be represented in transition or state space form. Considering non zero initial conditions, state space system is preferred. Consider the system

$$x_j(k+1) = \mathbf{A}\, x_j(k) + \mathbf{B}\, u_j(k) \qquad (2)$$

$$y_j(k) = \mathbf{C}\, x_j(k) \qquad (3)$$

$$\text{With } x_j(0) = x_0,\ \forall\ j$$

The state-space system is equivalent to

$$y_j(k) = \mathbf{C}(q\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\, u_j(k) + \mathbf{C}\mathbf{A}^k x_0$$

Where P(q) corresponds to $\mathbf{C}(q\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$ and d(k) corresponds to $\mathbf{C}\mathbf{A}^k x_0$. The signal d(k) is the free response to the initial conditions.

Most popularly used ILC learning algorithm is given by

$$u_{j+1}(k) = \mathbf{Q}(q)\,[u_j(k) + \mathbf{L}(q)\, e_j(k+1)\,] \qquad (4)$$

From LTI dynamics, Q(q) is defined as Q filter and L(q) is defined as learning function.

At the end of each iteration, the error is filtered through learning filter and then added to the previous control input and passed again through Q-filter. This updated open loop control input is applied to the plant in the next iteration.

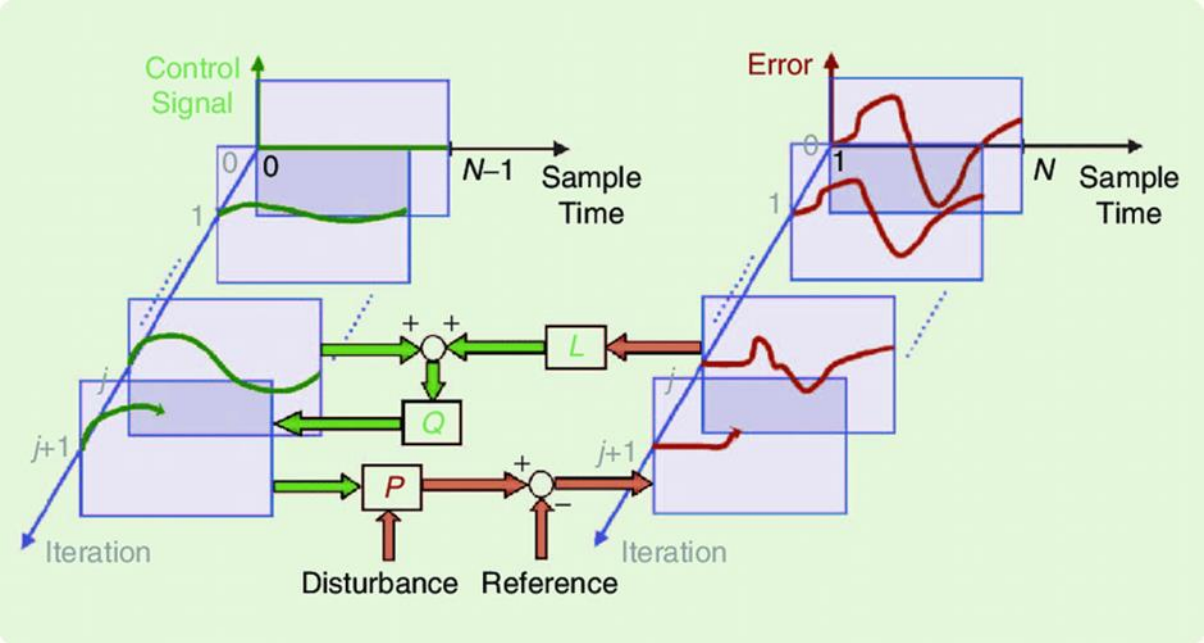


Figure 1.2: A 2D representation of first order ILC system

CHAPTER 2

SYSTEM REPRESENTATION

Analysis of ILC systems is based on two system representations.

1. Time domain analysis using the lifted system framework.
2. Frequency domain analysis using Z – domain representation.

2.1 Time domain analysis:

The lifted system framework is based on impulse response representation of system. To get lifted system representation, the proportional rational function is expanded as an infinite power series by dividing its denominator into its numerator which gives

$$p(q) = p_1 q^{-1} + p_2 q^{-2} + p_3 q^{-3} + \ldots\ldots$$

Here the coefficients p_k are markov parameters and the sequence $p_1, p_2, \ldots$ is the impulse response. Note that $p_1 \neq 0$, as m =1 is assumed.

For the state space description of system (2) and (3), it is given $p_k = C\, A^{k-1} B$.

N x N dimensional lifted system can be written as:

$$\begin{bmatrix} y_j(1) \\ y_j(2) \\ \vdots \\ y_j(n) \end{bmatrix} = \begin{bmatrix} p_1 & 0 & \ldots & 0 \\ p_2 & p_1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_n & p_{n-1} & \ldots & p_1 \end{bmatrix} \begin{bmatrix} u_j(1) \\ u_j(2) \\ \vdots \\ u_j(n) \end{bmatrix} + \begin{bmatrix} d(1) \\ d(2) \\ \vdots \\ d(n) \end{bmatrix}$$

And the performance signal is

$$\begin{bmatrix} e_j(1) \\ e_j(2) \\ \vdots \\ e_j(n) \end{bmatrix} = \begin{bmatrix} y_d(1) \\ y_d(2) \\ \vdots \\ y_d(n) \end{bmatrix} - \begin{bmatrix} y_j(1) \\ y_j(2) \\ \vdots \\ y_j(n) \end{bmatrix}$$

Remark that if time delay is m > 1, the vectors e_j, y_j, y_d and d change, that is their first element is the variable with time index k = m instead of k = 1. Also, the matrix **P** changes; by replacing all the coefficients p_i by p_{m+i-1}.

The ILC algorithm can also be expressed in lifted system representation.

Q filter and L(q) can be non-causal functions and their impulse responses are

$$Q(q) = q + q_0 + q_1 q^{-1} + Q_2 q^{-2} + \cdots\cdots$$

$$L(q) = L_0 + L_1 q^{-1} + L_2 q^{-2} + \cdots\cdots$$

$$U_{j+1}(k) = Q(q)\,(U_j(k) + L(q)\, e_j(k+1))$$

$$\begin{bmatrix} U_{j+1}(0) \\ U_{j+1}(1) \\ \vdots \\ U_{j+1}(n-1) \end{bmatrix} = \begin{bmatrix} q_0 & q_{-1} & \cdots & q_{-n+1} \\ q_1 & q_0 & \cdots & q_{-n+2} \\ \vdots & \vdots & \ddots & \vdots \\ q_{n-1} & q_{n-2} & \cdots & q_0 \end{bmatrix} \left(\begin{bmatrix} U_j(0) \\ U_j(1) \\ \vdots \\ U_j(n-1) \end{bmatrix} + \begin{bmatrix} L_0 & L_{-1} & \cdots & L_{-n+1} \\ L_1 & L_0 & \cdots & L_{-n+2} \\ \vdots & \vdots & \ddots & \vdots \\ L_{n-1} & L_{n-2} & \cdots & L_0 \end{bmatrix} \begin{bmatrix} e_j(1) \\ e_j(2) \\ \vdots \\ e_j(n) \end{bmatrix} \right)$$

When Q(q)and L(q) are causal filters, then $q_{-1} = q_{-2} = \cdots\cdots = 0$ and

$L_{-1} = L_{-2} = \cdots\cdots = 0$. Also, the matrices Q and L are lower triangular matrices.

P, Q and L are Toeplitz meaning that all the elements along each diagonal are identical.

2.2 Frequency domain analysis:

The Z-transformation of the signal is given by $X(z) = \sum_{k=0}^{\infty} x(k) z^{-1}$ and Z transformation of the system is obtained by replacing q with z. To apply Z transformations to ILC system we assume N = ∞, because Z transform requires that the signal be defined over an infinite time horizon. Since all the practical applications of ILC have a finite horizon the Z domain is an approximation of ILC system.

$$Y_j(z) = p(z)\, U_j(z) + D(z)$$

$$U_{j+1} = Q(z)\,[U_j(z) + z\, q(z)\, E_j(z)]$$

$$E_j(z) = y_d(z) - y_j(z).$$

CHAPTER 3

ANALYSIS

This section comprises of four basic topics of great importance for understanding ILC systems and behaviour. They are Stability, Performance, Transient learning behaviour and Robustness.

3.1 Stability

If the ILC system is AS then there exists $\bar{u} \in \mathbb{R}$, Such that

$$|u_j(k)| \leq \bar{u}$$

For all $k = \{0, 1, \ldots, n-1\}$ and j = {0, 1, ……}

$$\lim_{j \to \infty} u_j(k) \text{ exists.}$$

We define converged control as

$$u_\infty(k) = \lim_{j \to \infty} u_j(k)$$

$$Y_j(k) = p(q)\, U_j(k) + d(k)$$

$$U_{J+1}(k) = Q(q)\, [U_j(k) + L(q)\, e_j(k+1)]$$

By Substituting $e_j = y_d - y_j$ in learning algorithm we get

$$U_{j+1} = Q\,(I - LP)\, U_j + QL\,(y_d - d)$$

Let $\rho(A) = \max |\lambda_i(A)|$ be the spectral radius of matrix A and $\lambda_i(A)$ be the eigen value of matrix A.

Theorem 1:

The ILC system is AS if and only if

$$\rho\,(Q\,(I - LP)) < 1$$

When the Q filter and the learning filter are causal, the matrix Q (I – LP) is a lower triangular and Toeplitz with repeated eigen values.

$$\lambda = q_0\,(1 - L_0\,p_1)$$

$$|q_0(1 - L_0\,p_1)| < 1$$

Theorem 2:

If $\|Q(z)[1 - z\,L(Z)\,p(z)]\|_\infty < 1$

Then the ILC system with N = ∞ is AS.

When Q(z) and L(z) are causal filters the above theorem also implies AS for a finite duration ILC systems. Their stability conditions are only sufficient and in general much more conservative than the need and sufficient condition ρ (Q (I – LP)) < 1.

3.2 Performance

Performance of ILC is measured by looking at the asymptotic value of the error.

$$e_\infty(k) = \lim_{j\to\infty} e_j(k)$$

$$e_\infty(k) = \lim_{j\to\infty} (y_d(k) - p(q)u_j(k) - d(k))$$

$$e_\infty(k) = y_d(k) - d(k) - p(q)\,U_\infty(k)$$

If the ILC is AS, then the asymptotic error is

$$e_\infty = [I - p\,[I - Q\,(I - LP)]^{-1}\,QL]\,(y_d - d)$$

For the lifted system and

$$E_\infty = \frac{[1-Q(z)][Y_d(z) - D(z)]}{1-[Q(z)[1-z\,L(z)\,p(z)]]}$$

for the z domain representation.

We obtain the above results by replacing the iteration indices with ∞.

$$U_\infty = Q\,(I - LP)\,U_\infty + QL\,(y_d - d)$$

$$U_\infty = [I - Q\,(I - LP)]^{-1}\,QL]\,(y_d - d)$$

Theorem 3:

Suppose that P and L are not identically zero.

Then for the ILC system $e_\infty(k) = 0$ for all values of k, y_d and d, if and only if the system is AS and Q(q) = 1.

Many ILC algorithms set Q(q) = 1 and thus do not include Q filtering and consequently yield a perfect performance. However, Q – filtering can improve the transient learning behaviour and robustness.

In most applications the Q filter is a low pass filter with DC gain equal to one. As a result, it yields perfect performance at low frequencies and gradually shuts off the ILC algorithm to achieve a tracking error equal to $y_d(e^{jw}) - D(e^{jw})$ at high frequencies.

3.3 Transient learning behaviour

Transient learning behaviour can be problematic, even if the stability conditions are satisfied. It is difficult to distinguish transient growth from instability in practice. Large transient growth is a fundamental topic of ILC and preventing it is an essential objective in ILC design

To avoid large learning transients, monotonic convergence is desirable.

ILC system is monotonically convergent if

$$\|e_\infty - e_{j+1}\| \leq \gamma \, \|e_\infty - e_j\|$$

$\forall\, j \in \{1, 2, 3, ...\}$ where $0 \leq \gamma < 1$ is the convergence rate.

In the lifted system representation, the asymptotic error result is given by

$$e_\infty - e_{j+1} = pQ\,(I - LP)\,P^{-1}\,(e_\infty - e_j)$$

When P, Q and L are causal, P, Q, L commute.

$$e_\infty - e_{j+1} = Q\,(I - LP)\,(e_\infty - e_j)$$

For z domain

$$E_\infty(z) - E_{j+1}(z) = Q(z)\,(1 - z\,L(z)p(z))\,(E_\infty(z) - E_j(z))$$

Let $\bar{\sigma}$ be the maximum singular value and let us consider the z norm to measure convergence, then the following monotonic convergence conditions can be derived.

Theorem 4:

If ILC system (1) and (4) satisfies

$$\gamma_1 \triangleq \bar{\sigma}\ (PQ\,(I - LP)\,P^{-1}) < 1$$

then

$$\|e_\infty - e_{j+1}\|_2 \leq \gamma_1 \|e_\infty - e_j\|_2 \ \forall\, j \in \{1, 2, 3, \ldots\ldots\ldots\}$$

Theorem 5:

If the ILC system (1) and (4) with $N = \infty$ satisfies

$$\gamma_2 \triangleq \|Q(z)[1 - z\,L(z)\,P(z)]\|_\infty < 1$$

then

$$\|E_\infty(z) - E_{j+1}(z)\|_\infty < \gamma_2 \|E_\infty(z) - E_j(z)\|_\infty$$

$\forall\, j \in \{1, 2, 3, \ldots\ldots\ldots\}$

When Q(z) and L(z) are causal filters also implies that

$$\|e_\infty - e_{j+1}\|_2 \leq \gamma_2 \|e_\infty - e_j\|_2$$

$\forall\, j \in \{1, 2, 3, \ldots\ldots\ldots\}$, for the ILC system with a finite duration N.

3.4 Robustness

An important issue for ILC because if the system model would be known exactly other, more direct approaches for the perfect tracking are preferable.

There we consider robustness as related to stability and monotonic convergence

Consider uncertain system:

$$P(q) = \widehat{P}(q)\,[1 + W(q)\,\Delta(q)] \qquad (5)$$

Where $\widehat{P}(q)$ is normal system model and $W(q)$ is known and stable.

$\Delta(q)$ is unknown and stable with $\|\Delta(z)\|_\infty < 1$

Theorem 6:

If $|W(e^{j\theta})| \leq \frac{\gamma^* - |Q(e^{j\theta})||1 - e^{j\theta}L((e^{j\theta})\hat{P}(e^{j\theta})|}{|Q(e^{j\theta})||e^{j\theta}L((e^{j\theta})\hat{P}(e^{j\theta})|} \quad \forall\, \theta \in [-\pi,\pi]$,

Then ILC is monotonically convergent with $N = \infty$, with convergence rate $\gamma^* < 1$.

From theorem 5, the ILC system is monotonically convergent with rate $\gamma^* < 1$, if

$$\gamma^* \geq \left\|Q(z)\left[1 - z\,L(z)\;\widehat{P}(z)\;[1 + W(z)\,\Delta(z)]\right]\right\|_\infty$$

$$= \max_\theta \left|Q(e^{i\theta})\left[1 - e^{i\theta}\,L(e^{i\theta})\;\widehat{P}(e^{i\theta})\;[1 + W(e^{i\theta})\,\Delta(e^{i\theta})]\right]\right|$$

$$= \max_\theta \left|Q(e^{i\theta})\left[1 - e^{i\theta}\,L(e^{i\theta})\;\widehat{P}(e^{i\theta})\right]\right| +$$

$$\left|Q(e^{i\theta})e^{i\theta}L(e^{i\theta})\;\widehat{P}(e^{i\theta})\,W(e^{i\theta})\right|$$

Which is equivalent to theorem 5 equation.

The monotonic robustness condition depends on dynamics of $P(q)$, $Q(q)$ and $L(q)$.

The most direct way to increase robustness at a given θ is to decrease the Q – filter gain$|Q(e^{j\theta})|$.Decreasing $|Q(e^{j\theta})|$ negatively impacts the converged performance.

CHAPTER 4

DESIGN

Practically the goal of ILC is to generate an open loop signal that approximately inverts the system dynamics to track a reference signal reject a repeating disturbance.

Design section essentially comprises of four methods which are popularly used. These are

1. PD – type and Tunable design
2. Plant inversion method
3. H∞ method
4. Quadratically optimal design

4.1 PD – Type and Tunable design

PD – Type learning function consists of proportional and derivative gain on the error. The P – type, D – type and PD – type are probably most widely used types of learning functions. The integrator term is sparsely used because ILC has natural integrator action from one trail to another. The PD - type learning function relies on tuning and can be applied to a system that does not require an accurate model and analysis.

The PD – type learning function can be written as

$$u_{j+1}(k) = u_j(k) + K_p\, e_j(k+1) + K_d\, [e_j(k+1) - e_j(k)]$$

where K_p is proportional gain and K_d is derivative gain.

The PD – type learning algorithm is AS if and only if $|1 - (K_P + K_d)p_1| < 1$ from theorem 1.

Monotonic convergence is not always possible with PD – type ILC.

The most favourable and generally applicable approach to achieve monotonic convergence is to include a low pass Q – filter in learning algorithm. As Q – filter disables learning at higher frequencies which has the additional benefits of added robustness and high frequency noise filtering.

For tuning of Q – filter, K_p and K_d, select a filter type and order and use the bandwidth of the filter as the tuning variable. For each set of K_p and K_d the learning is reset and run for sufficient iterations to determine the transient behaviour and asymptotic error. Begin with low learning gains and filter bandwidth. Generally learning gains effects the rate of convergence and the Q – filter influences the converged error performance. By increasing the Q – filter bandwidth, robustness decreases whereas the performance increases.

In two step tuning method, based on the asymptotic stability and monotonic convergence condition,

from theorem 2

$$|1 - e^{i\theta}L((e^{i\theta})P(e^{i\theta})| < \frac{1}{|Q(e^{i\theta})|} \quad \forall\, \theta \in [-\pi, \pi].$$

By using the Nyquist plot of $e^{j\theta}L((e^{j\theta})P(e^{j\theta})$, tune the learning gains to maximize the range $\theta \in [0, \theta_c]$, over which $e^{j\theta}L((e^{j\theta})P(e^{j\theta})$ lies inside the unit circle centred at one.

The Q – filter bandwidth is selected last to satisfy the stability condition.

4.2 Plant inversion method

In this method we use the model of inverted system dynamics in ILC algorithm.

$$u_{j+1}(k) = u_j(k) + \widehat{P}^{-1}(q)\, e_j(k)$$

This corresponds to (4) with $Q(q) = 1$ and $L = q^{-1}\, \widehat{P}^{-1}(q)$, which is causal.

If $\hat{P}(q)$ is an exact model of the system and the inversion of this model does not introduce errors, it can be verified from theorem 4 and theorem 5 that the convergence rate $\gamma = 0$: the convergence occurs in just one iteration and $e_\infty = 0$.

The Problem with this method is that the non-minimum phase system results in an unstable filter. To avoid it, use a stable inversion approach that results in non-causal learning filter.

In case on minimum phase systems, the success depends on the accuracy of the model. Mismatch between model $\widehat{P}(q)$ and actual system $P(q)$ prevents the convergence from occurring in one iteration, and can even lead to poor transient behaviour.

Consider an uncertain system with multiplicative uncertainty (5), and based on theorem 6, monotonic convergence with rate better than γ^* occurs if $|W(e^{j\theta})| < \gamma^*$ $\forall\, \theta \in [-\pi, \pi]$.

If at certain uncertainty θ_0, $|W(e^{j\theta})| > 1$, signifying an uncertainty greater than 100%, then the ILC system will not be robustly monotonically convergent.

Large uncertainty typically occurs at high frequencies: remedy here is to include a low pass Q – filter.

4.3 H∞ methods

Offers a systematic approach to ILC design

The goal of this design is to find the learning function L(q) that offers the fastest convergence rate for a given Q – filter.

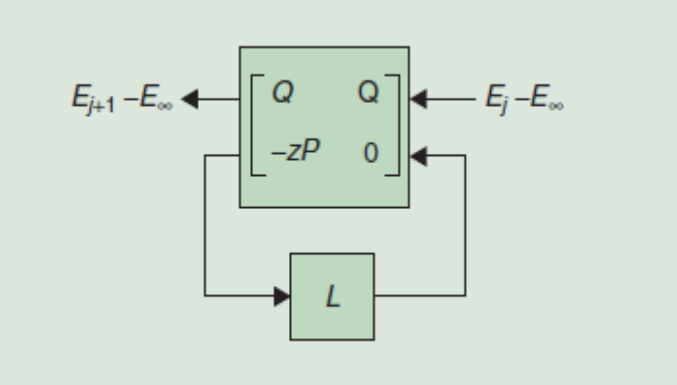


Figure 4.1: Model problem for H∞ design method.

$$L^*(z) = \arg_L \min \|Q(z)(I - z\,L(z)P(Z))\|_\infty$$

$$Q\,(I - z\,LP) = G_{11} + G_{12}\,L\,(I - G_{22}\,L)^{-1} = F_L\,(G, L)$$

$$\text{Where } G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} = \begin{bmatrix} Q & Q \\ -zP & 0 \end{bmatrix}$$

4.4 Quadratically optimal design

In Q – ILC, the learning functions are designed in the lifted system representation to minimize a quadratic next iteration cost criterion.

$$J_{j+1}\,(u_{j+1}) = e_{j+1}{}^T\,Q_{LQ}\,e_{j+1} + u_{j+1}{}^T\,R_{LQ}\,u_{j+1} + \delta_{j+1}\,u^T\,S_{LQ}\,\delta_{j+1}\,u$$

where $\delta_{j+1}u \triangleq u_{j+1} - u_j$,

Q_{LQ} is an N x N positive definite matrix and R_{LQ}, S_{LQ} are N x N positive semi definite matrices.

Minimizing the cost criterion with respect to u_{j+1} gives the optimal Q – filter and learning functions.

$$Q_{opt} = (P^T Q_{LQ} P + R_{LQ} + S_{LQ})^{-1} (P^T Q_{LQ} + S_{LQ})$$

$$L_{opt} = (P^T Q_{LQ} P + S_{LQ})^{-1} P^T Q_{LQ}$$

$$e_{\infty,\, opt} = [I - P (P^T Q_{LQ} P + R_{LQ})^{-1} P^T Q_{LQ}] (y_d - d)$$

CHAPTER 5

IMPLEMENTATION EXAMPLE -Microscale Robotic Deposition

For practical demonstration on the effectiveness of ILC on a real system, we consider the microscale robotic deposition ($\mu - RD$)as shown in figure.

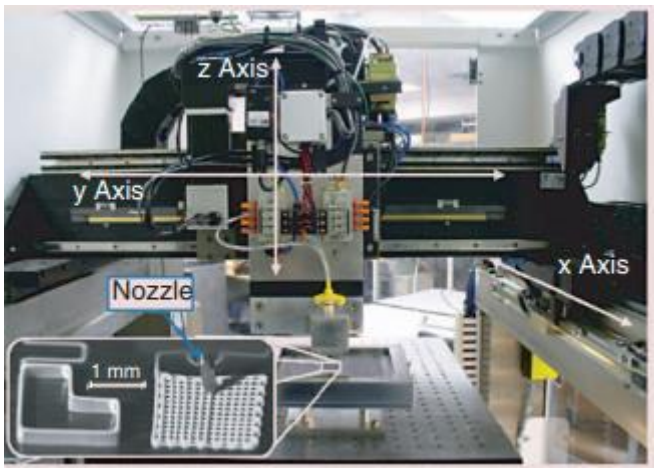


Figure 5.1: Microscale robot deposition system.

$\mu - RD$ uses a solid-freeform-fabrication manufacturing technique whereby ink is exited through the nozzle and deposited on a substrate. The highly thixotropic ink solidifies quickly after exiting through the nozzle, allowing the ink to spread out in open gaps and support high aspect-ratio features.

A robot is used to position the nozzle continuously in the three-dimensional space to deposit the ink. We are interested in improving the tracking performance of the robot using ILC. As the robot is cartesian, the axes are dynamically decoupled by design, hence consider an individual axis.

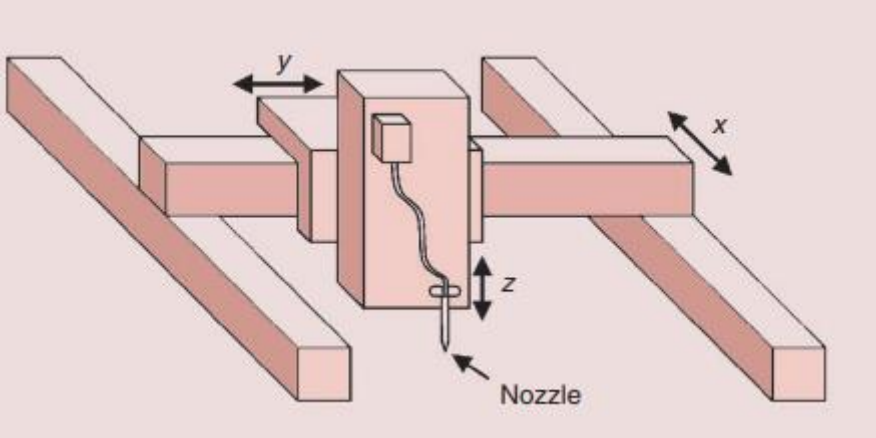


Figure 5.2: Schematic motion system layout.

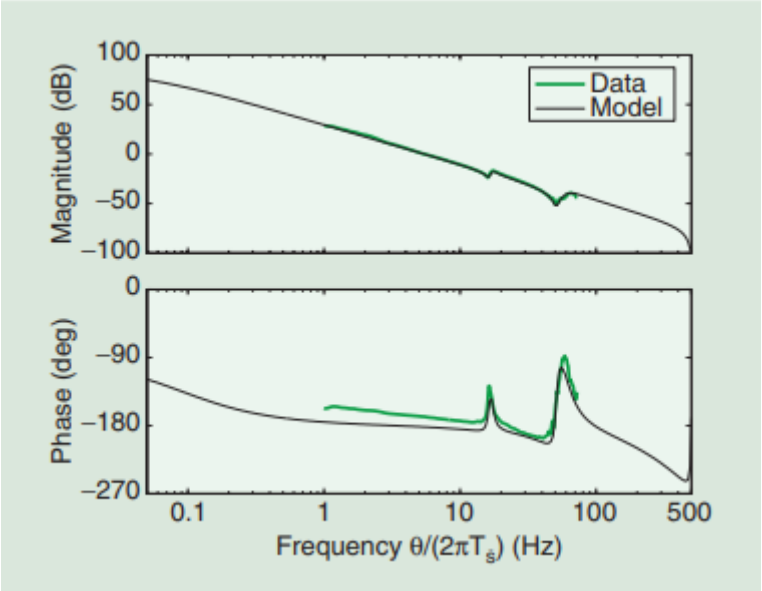


Figure 5.3: Experimental frequency response and model of

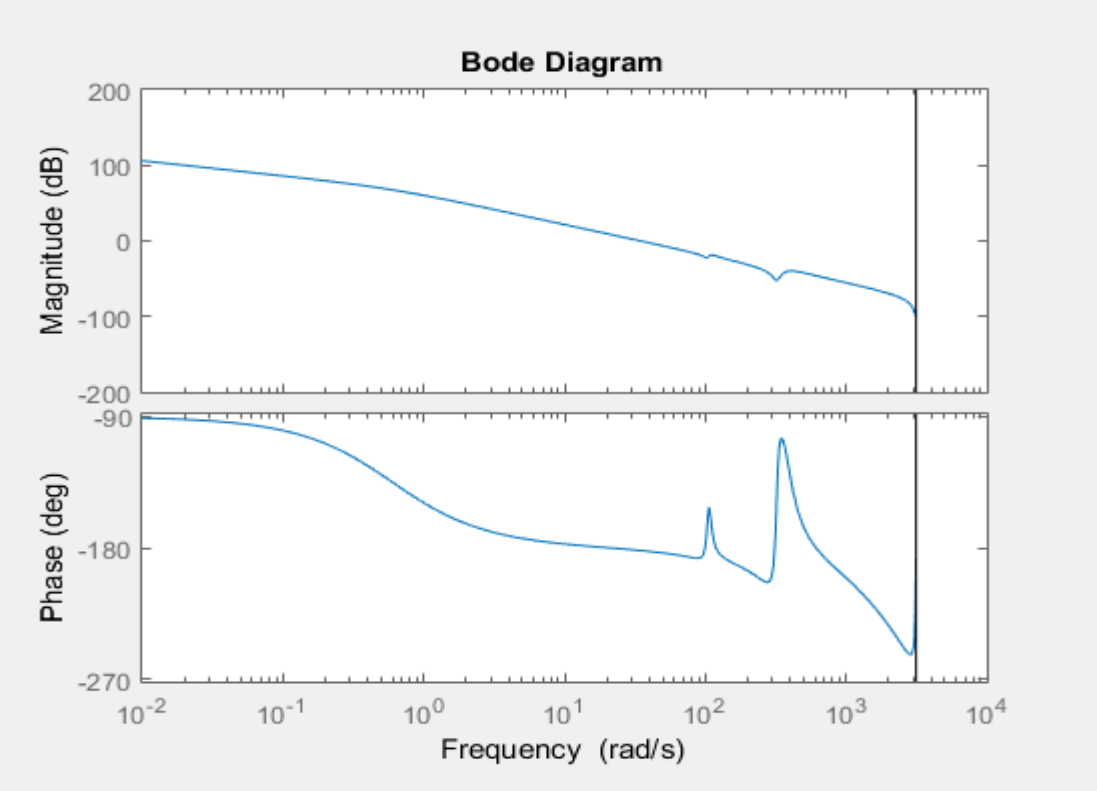
the μ – RD x axis with sampling period T_s = 0.001 s.

From the experimental results, we obtain a bode plot of frequency range 1 – 70 Hz. From the obtained bode plot the transfer function which is calculated through sampling is given by,

$$G_X(z) = \frac{0.00083315\ (z+0.9604)\ (z^2-1.981z+0.9918)(z^2-1.874z+0.9747)}{(z-0.9994)(z-1)(z^2-1.978z+0.9894)(z^2-1.738z+0.8672)}$$

By executing the transfer function in MATLAB, we verified the bode plot.

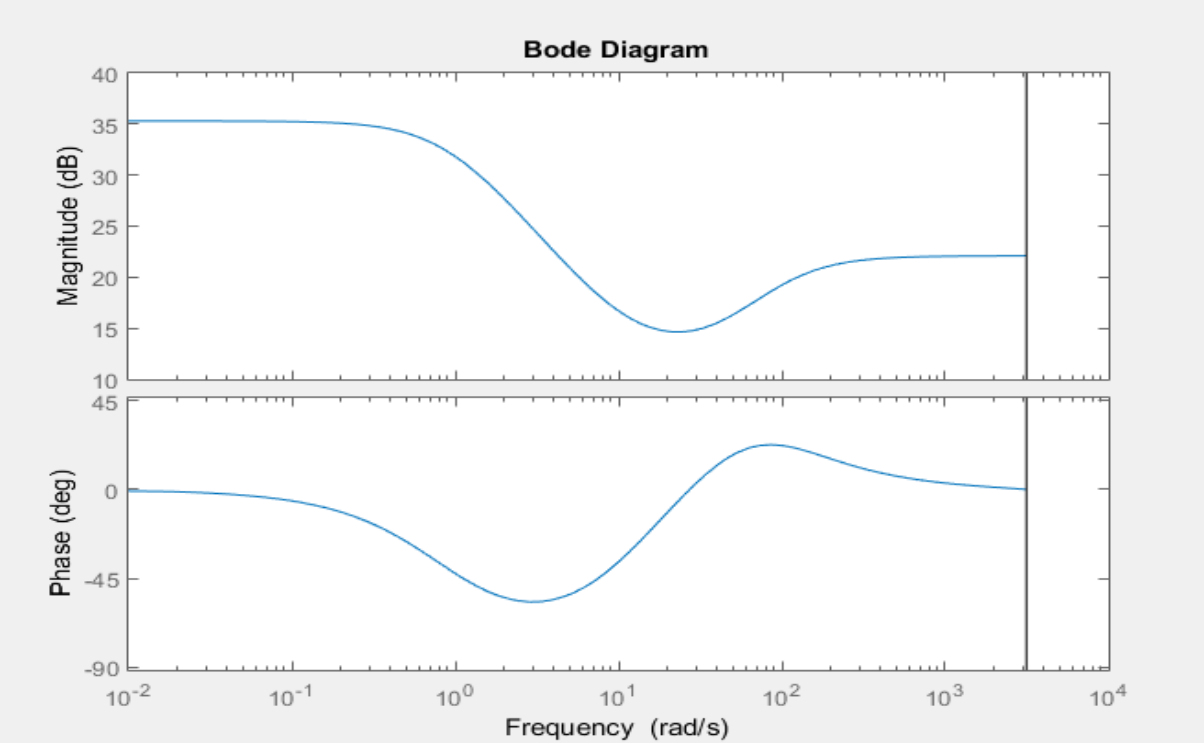
```
>> ts = 0.001;
z1 = [-0.9604];
p1 = [0.9994 1];
k1 = 0.00083315;
num1 = [1 -1.981 0.9918];
den1 = [1 -1.978 0.9894];
num2 = [1 -1.874 0.9747];
den2 = [1 -1.738 0.8672];
t1 = zpk(z1,p1,k1,ts);
t2 = tf(num1,den1,ts);
t3 = tf(num2,den2,ts);
g = t1*t2*t3

g =

  0.00083315 (z+0.9604) (z^2 - 1.981z + 0.9918) (z^2 - 1.874z + 0.9747)
  ---------------------------------------------------------------------
    (z-0.9994) (z-1) (z^2 - 1.978z + 0.9894) (z^2 - 1.738z + 0.8672)

Sample time: 0.001 seconds
Discrete-time zero/pole/gain model.

bode(g)
```



The nominal controller uses both a feedback control and reference feedforward control. The feedback controller is designed using loop shaping for fast transient response.

$$C_{FBK}(z) = \frac{12.3359(z-0.9874)(z-0.9577)(z-0.945)}{(z-0.9991)(z-0.9174)(z-0.9164)}$$

```
>> Ts = 0.001

Ts =

   1.0000e-03

>> z = [0.9874 0.9577 0.945];
>> p = [0.9991 0.9174 0.9164];
>> k = 12.3359;
>> c = zpk(z,p,k,Ts)

c =

  12.336 (z-0.9874) (z-0.9577) (z-0.945)
  --------------------------------------
     (z-0.9991) (z-0.9174) (z-0.9164)

Sample time: 0.001 seconds
Discrete-time zero/pole/gain model.

>> bode(c)
```



This controller designed to provide fast transient response without exciting resonant modes of the motor and the amplifier by tuning with high frequency gain. It also attempts to compensate for friction and cogging forces while minimizing the oscillatory response of the system to reference changes. For further improvement of system response, the reference feedforward controller is given by

$$C_{FFD}(z) = G_X^{-1}(z)\, F(z)$$

Where F(z) is the low pass filter with 100 Hz bandwidth given by,

$$F(z) = \frac{0.1568(z+0.8093)}{z^2-1.25z+0.5535}$$

Which makes C_{FFD} causal.

Tuning:

The design methods in the design section are applicable to the system. We choose the simplest method, the tuned PD learning function and its algorithm is given by

$$u_{j+1}(k) = Q(q)[u_j(k) + K_p\, e_j(k+1) + K_d\, [e_j(k+1) - e_j(k)]]$$

where Q(q) is a zero-phase low pass filter added for robustness.

A position step trapezoidal velocity profile for the above figure is used as sample trajectory for tuning the ILC parameters. The maximum velocity and acceleration of this trajectory are 10 mm/sec and 250 mm/sec^2.

We begin tuning with conservative gains $K_p = 1$ and $K_d = 10$ and Q – filter bandwidth of 20 Hz. By keeping bandwidth constant with different gains, the maximum and RMS errors Vs iteration are plotted.

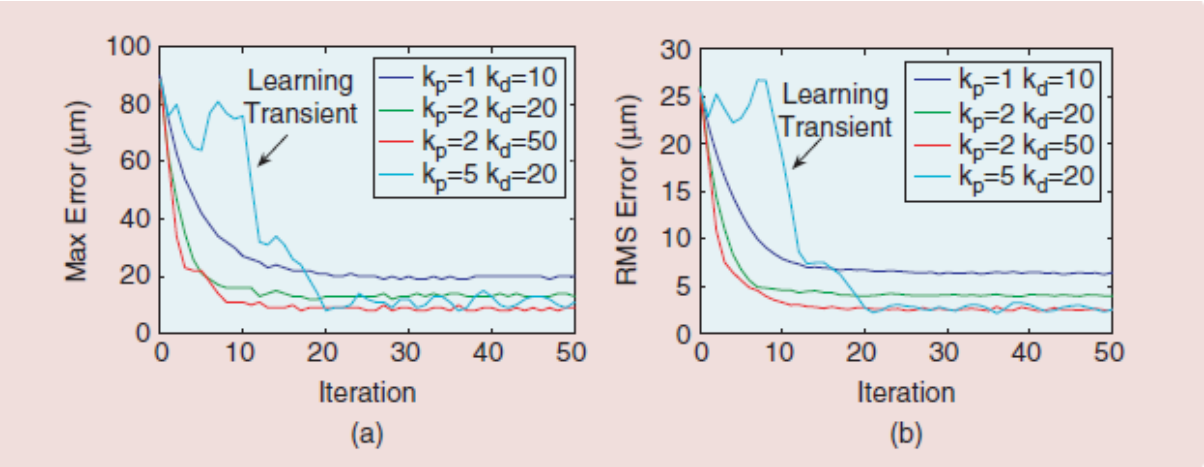


Figure 5.4: Transient behaviour and asymptotic performance

for PD gain tuning.

From the above results the gains K_p = 2 and K_d = 50 are selected. By keeping these gains fixed with different Q – filter bandwidths, the maximum and RMS errors Vs iteration are plotted.

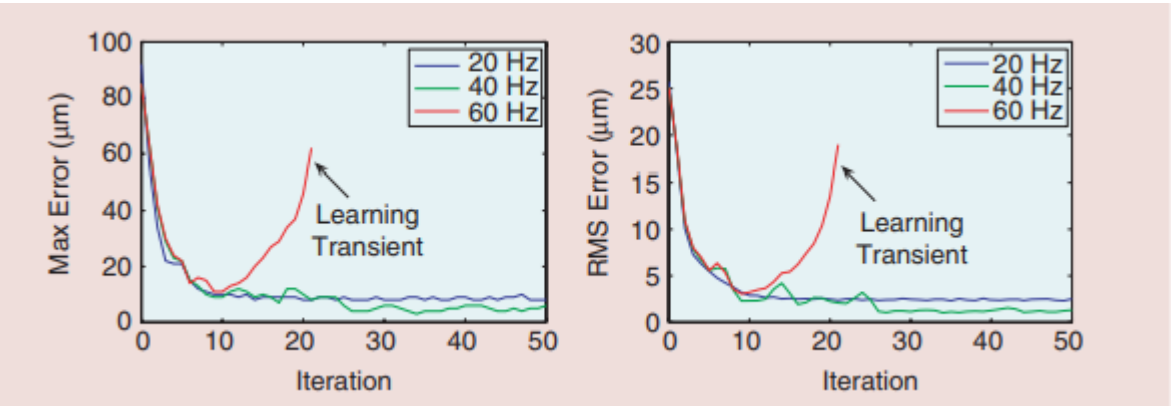


Figure 5.5: Transient behaviour and asymptotic performance

For Q – filter bandwidth tuning.

Tracking performance results:

From the figure (7), the gain combination $K_p = 5$ and $K_d = 20$ has low asymptotic error and the learning transients do not converge monotonically like the others. Hence it may not be good choice of gains. The other three combinations have nearly monotonic transients but $K_p = 2$ and $K_d = 50$ converges fastest with the lowest asymptotic error.

From the figure (8), the plot shows that Q – filter has little effect on convergence rate of the error but instead primarily determines the magnitude of the converged error and the stability of the system. Although increase bandwidth improves the error, a bandwidth that is too high results in learning transients.

CONCLUSION

In this project, we have analysed the ILC algorithm and system representation. For understanding of ILC system behaviour, we have studied the important issues such as stability, performance, transient learning behaviour and robustness. Performance of a system can be improved with ILC for repeating tasks and disturbances. Hence ILC algorithm is widely used in industrial applications where mass production requires repetition. Good learning transient behaviour is identified as the practical stability condition where good transient behaviour is defined as monotonic convergence. When robustness is considered uncertainty of the model leads to limitations on the performance of LTI learning algorithm.

We have discussed the popular design techniques and we implemented the easiest one among them that is PD type for microscale robotic deposition which enhanced its performance.

REFERENCES

1. Douglas A. Bristow, Marina Tharayil, and Andrew G. Alleyne, A survey of iterative learning control, IEEE – volume (26), 2006.
2. Kevin L. Moore and Jian-xin Xu, Editorial - special issue on iterative learning control, 2000.
3. Mikael Norloff and Svante Gunnarsson, Time and frequency domain convergence properties in iterative learning control, International journal of control, 2002.
4. Longman R.W., Iterative learning control and repetitive control for engineering factors, international journal of control volume (73), 2000.
5. Norloff. M, An adaptive iterative learning control algorithm with experiments on an industrial robot, IEEE volume (18), 2002.
6. 2015. MATLAB – math works. URL https://in.mathworks.com/product/ltc/matlab.html.