ROBUST REPETITIVE MODEL PREDICTIVE CONTROL FOR SYSTEMS WITH UNCERTAIN PERIOD-TIME

Approved by:

Dr. Jay H. Lee, Chair

Dr. Matthew Realff

Dr. Pete Ludovice

Date Approved: April 9th, 2004
I would like to express my sincere thanks to my advisor, Dr. Jay H. Lee, for his guidance and support throughout the course of this research. I am very grateful to him for providing me with plentiful opportunities for professional development. His high standards in clear thinking and effective communication are inspiring and will always help me stand in good stead.

I would also like to gratefully acknowledge my other thesis committee members, Dr. Matthew Realff and Dr. Pete Ludovice, for dedicating their valuable time to serve on my committee and for giving me their useful suggestions.

The process control group at Georgia Tech (ISSICL) is simply amazing and composed of very talented individuals. I will always cherish the great experience of having been a part of this group. I want to thank all group members for creating such a wonderful atmosphere for research.

In particular, I would like to acknowledge Niket and Ashwini for their friendship and for their support through thick and thin. I can never thank them enough either with words or by actions
### TABLE OF CONTENTS

**ACKNOWLEDGEMENTS** ........................................... iii

**LIST OF TABLES** ................................................ vii

**LIST OF FIGURES** .............................................. viii

**SUMMARY** ....................................................... x

**CHAPTER 1 INTRODUCTION** .................................... 1

1.1 Motivation ................................................... 1

1.2 Available Methods in Literature ....................... 2

1.3 Objective of the Thesis .................................... 3

**CHAPTER 2 CONTROL OF PERIODIC PROCESSES** .......... 4

2.1 Periodic Processes .......................................... 4

2.2 Examples of Periodic Systems ............................. 6

2.2.1 Continuous Processes .................................. 6

2.2.2 Batch processes ....................................... 6

2.2.3 Hybrid process ........................................ 8

2.3 Shortcomings of Conventional Controllers Applied on Periodic Processes ........................................ 8

2.3.1 PID Controller with Periodic Processes .......... 10

2.3.2 MPC with Periodic Processes .......................... 10

2.4 Challenges of Controlling Periodic Processes ........ 13

2.5 Learning Controller Applied to Periodic Processes  .............................................................................. 13

2.5.1. Iterative Learning Control (ILC) ..................... 14

2.5.2 Repetitive Control (RC) ................................ 16

2.6 Difference between ILC and RC ........................... 16
2.7 Principle of RC
   2.7.1 Internal Model Control  17
   2.7.2 Periodic Signal Generator  17
2.8 Rejection Restriction in RC  19
2.9 Digital Repetitive Controller  19

CHAPTER 3 REPETITIVE MODEL PREDICTIVE CONTROL  21
3.1 Shortcoming of RC  21
3.2 Advantages of Combining Repetitive Control with Model Predictive Control  21
3.3 Time Domain Interpretation of RC  22
3.4 Development of RMPC  22
   3.4.1 General Formulation of Control Problem  22
   3.4.2 Lifting and Augmenting  24
   3.4.3 Run-to-Run Formulation  24

CHAPTER 4 PERIOD MISMATCH PROBLEM IN REPETITIVE CONTROL  29
4.1 Sensitivity analysis  29
4.2 Causes of Mismatch  30
   4.2.1 Frequency Not Known Exactly  30
   4.2.2 Varying Frequency  31
   4.2.3 Non-divisibility of Period  31
4.3 Current Approaches for Handling Period Mismatch  31
   4.3.1 For Frequency Not Known Exactly and/or Changing  31
   4.3.2 For Period Non-integer Multiple of the Sampling Time  32
4.4 Need for Robust RMPC  33

CHAPTER 5 DEVELOPMENT OF ROBUST RMPC  35
5.1 Robust Repetitive Control  35
5.2 Robust Repetitive Model Predictive Controller Construction  38
   5.2.1 Interpretation of Multiple Memory Loops in RMPC Framework  40
   5.2.2 Unconstrained Run-to-Run Robust RMPC  40
5.2.3 Constrained Run-to-Run Robust RMPC 44
5.2.4 Real-time Control for Robust RMPC 44
5.3 Results Comparison between RMPC and Robust RMPC 48
5.4 Application of Robust RMPC to Mechanical Motion Tracking Control 51

CHAPTER 6 LIMITATIONS OF RMPC AND R-RMPC 53
6.1 Period Mismatch in Class I Systems 53
    6.1.2 Results Discussion 53
6.2 Period Mismatch vs. Other Parameters Mismatch 56
6.3 Simple Practical Approach to Tackle Small Period Mismatches for Class I Systems 58

CHAPTER 7 CONCLUSIONS AND FUTURE WORK 60
7.1 Summary of Work 60
7.2 Suggestions for Future Work 62
    7.2.1 Analyzing the Effect on Non-harmonic Frequencies 62
    7.2.2 Reducing Inter-sampling Ripples 62
    7.2.3 Selecting Sampling Frequency 63
    7.2.4 Using Interpolation Schemes for Period Mismatch 63

REFERENCES 64
LIST OF TABLES

Table 5.1   Steady state error with RMPC and R-RMPC 50
LIST OF FIGURES

Figure 2.1  Block diagram of processes with periodic references/trajectories  5
Figure 2.2  Polyamine process and optimal periodic trajectories of operation  7
Figure 2.3  Different phases of batch processes and its repetitive nature  7
Figure 2.4  A continuous process with underlying discrete periodic events  9
Figure 2.5  Simulated moving bed  9
Figure 2.6  PID controller with periodic trajectory tracking  11
Figure 2.7  Basic structure of MPC  11
Figure 2.8  Receding horizon MPC  12
Figure 2.9  MPC with periodic trajectory tracking  12
Figure 2.10  Basic structure of learning controller  15
Figure 2.11  Block diagram of a memory loop/periodic signal generator  18
Figure 2.12  Block diagram of a repetitive controller  18
Figure 3.1  Time domain interpretation of period-wise integration in repetitive control  23
Figure 4.1  Gain of periodic signal generator with respect to frequency  30
Figure 5.1  Block diagram of basic structure of a repetitive controller  36
Figure 5.2  Standard single memory loop  36
Figure 5.3  Multiple memory loops for generalized repetitive controller  36
Figure 5.4  Frequency response of single memory loop repetitive controller  39
Figure 5.5  Frequency response of multiple memory loop repetitive controller (N=1); generalized repetitive controller (N=2,3)  39
Figure 5.6  Change in periodic signal generator magnitude with respect to mismatch standard repetitive controller (N=1); generalized repetitive controller (N=2,3)  40
Figure 5.7  Block diagram of a period-wise integrator  41
Figure 5.8  Alternate representation of a periodic integrator  41
Figure 5.9  Block diagram of a double memory loop $W_1 = 2, W_2 = -1$  42
Figure 5.10  Discrete time equivalent of a double memory loop structure  42
Figure 5.11  Alternative representation of double memory loop repetitive control  42
Figure 5.12  Performance of standard RMPC with and without period mismatch  49
Figure 5.13  Performance of Robust RMPC with and without period mismatch  49
Figure 5.14  Performance of RMPC and R-RMPC as a function of period mismatch  50
Figure 5.15  RMPC vs. Robust RMPC with mechanical motion tracking control  51
Figure 6.1  A prototypical switching system  55
Figure 6.2  Instability due to period mismatch in the switching system  55
Figure 6.3  Illustration of plant/model error due to period mismatch in type 1 systems  56
Figure 6.4  Illustration of plant/model error due to parameter mismatch only  57
Figure 6.5  Control of SMB using heuristic approach in presence of period mismatch  59
SUMMARY

Repetitive Model Predictive Control (RMPC) incorporates the idea of Repetitive Control (RC) into Model Predictive Control (MPC) to take full advantage of the constraint handling, multivariable control features of MPC in periodic processes. The RMPC achieves perfect asymptotic tracking/rejection in periodic processes, provided that the period length used in the control formulation matches the actual period of the reference/disturbance exactly. Even a small mismatch between the actual period of process and the controller period can deteriorate the RMPC performance significantly. The period mismatch occurs either from an inaccurate estimation of actual frequency of disturbance due to resolution limit or from trying to force the controller period to be an integer multiple of sampling time. An extension of RMPC called Robust Repetitive Model Predictive Control (R-RMPC) is proposed for such cases where period length cannot be predetermined accurately, or where period is not an integer multiple of sampling time. This robust RMPC borrows the idea of using weighted, multiple memory loops in RC for robustness enhancement. The modified RMPC is more robust in the sense that small changes in period length do not diminish the tracking/rejection properties by much. Simulation results show that R-RMPC achieves significant improvement over the standard RMPC in case of a slight period mismatch. The effectiveness of this Robust RMPC is demonstrated by applying it to a mechanical motion tracking machine whose function is to follow a constant trajectory while rejecting periodic disturbances of an uncertain period.
CHAPTER 1

INTRODUCTION

The objective of this thesis is to develop a Robust Repetitive Model Predictive Control (R-RMPC) technique that can deal with a ‘small’ mismatch between the period assumed by the controller and the period of reference/disturbance to be tracked/rejected. This mismatch can occur due to an inaccurate estimation of period or to the restriction of controller period being an integer multiple of sample-time. The R-RMPC approach developed in this thesis maintains the controller performance in an acceptable range even in the presence of period-length mismatch.

1.1 Motivation

Periodic processes are often encountered in numerous engineering applications. These are the processes that show periodic characteristics due to periodic operation and/or acting periodic disturbances. The periodicity can arise either naturally or through intentional periodic forcing of the process in order to improve its performance. An example of naturally occurring periodic process is a wastewater treatment plant with diurnal influent variations [1]. An example of the latter kind is a simulated moving bed chromatography process where periodic switching of inlet-outlet ports drives the underlying continuous dynamics [2].

Model Predictive Control (MPC) is an advanced control technique that uses a dynamic model of the plant to predict and optimize the effect of future control moves on the controlled variables. MPC has been widely popular in industry due to its ability to handle many practical issues such as multivariable interactions and operational constraints [3].

Conventionally formulated MPC, which extrapolates the feedback error as a constant bias, does not handle periodic errors effectively, leaving significant periodic offset [4]. To deal with the periodicity more effectively within MPC, Lee and Natarajan
developed a suitable extension of MPC called Repetitive Model Predictive Control (RMPC), which borrows an idea from Repetitive Control (RC) on how to reject periodic errors [5]. The new formulation uses the technique of “lifting” to represent a periodically time varying system in time invariant form and then embedding a period–wise integrator to asymptotically track periodic trajectory or reject periodic disturbances without offset.

The performance of standard RMPC strongly depends on the precise match between the controller’s period and the actual period of the error signal. The controller generates a comb-filter shaped closed-loop sensitivity function. The notches of this sensitivity function are sharply narrow. Hence, even for a slight period mismatch the controller’s performance degrades substantially [6]. This performance loss can be even higher for higher harmonic frequencies.

In this thesis, the idea of using multiple memory loops in robust repetitive control [7] is borrowed to modify the standard RMPC into Robust RMPC (R-RMPC) that can deal effectively with the period mismatch. The new R-RMPC shows better robustness to (slight) period-length mismatch than the standard RMPC.

1.2 Available Methods in Literature

There are two main sources of period mismatch. First is the inaccurate estimation of the period of disturbance. Second is the truncation error introduced in trying to pick the controller period to be an integer multiple of sample time. As a solution to the first problem, most methods in the literature propose the use of an adaptive scheme to identify with finer resolution the period of the error signal based on on-line measurements [8, 9]. To deal with the second problem, research suggests augmenting the frequency estimation scheme with an adjustment of sampling time [10]. The idea is to change the sampling time to minimize the truncation error between the actual period and that assumed in the controller. The application of these propositions is not always practical and might be restricted by many other factors such as frequency resolution bounds and hardware limitations.

In contrast to the above techniques, this thesis uses the concept of using multiple memory loops to develop a Robust RMPC to handle small mismatch in period-time in a
non-adaptive manner. This R-RMPC can be used independently or in conjunction with any of the above mentioned frequency estimation techniques to handle larger mismatches and also to render better performance.

1.3 Objective of the Thesis

As mentioned earlier, Lee and Natarajan developed the RMPC technique, which can deal effectively with periodic errors within the framework of MPC. The performance of RMPC is highly dependent upon the precise match between the actual period and that assumed by the controller. In the case of perfect period matching, the controller is able to completely reject the periodic disturbance asymptotically. However, even a slight mismatch in the period can bring the controller’s performance down significantly.

The main objective of this thesis is to develop an extension called Robust RMPC (R-RMPC) such that small changes in the period do not diminish the disturbance rejection properties of the RMPC. This robustness is achieved by employing more than one ‘memory-loop’ in RMPC. R-RMPC renders an acceptable asymptotic performance even in the presence of small period-length mismatches. The improvement is illustrated by an example of a mechanical motion tracking machine that is able to reject disturbances of uncertain period. Another objective of this thesis is to apply this robust rmpc to control a simulated moving bed chromatography system for period mismatch case. As we shall see, this objective is not achieved fully, but it has led to some valuable insights into limitations with RMPC/R-RMPC. As a remedy, we propose a heuristic solution to deal with the mismatch problem of this particular kind.
CHAPTER 2

CONTROL OF PERIODIC PROCESSES

2.1 Periodic Processes

Periodic systems are found in a large number of engineering applications. These are the processes in which the operation is periodic in nature and/or the reference trajectory/disturbance to be followed/rejected is periodic.

Periodic processes can occur either naturally or due to a periodically forced operation. For example, in a wastewater treatment plant, the feed flow rates and compositions show natural diurnal cycles [1]. In other applications, like simulated moving bed processes and pressure swing absorption columns, periodic switches are forced in order to improve the process performance. These periodic processes have been studied extensively in chemical engineering, mechanical engineering, and more recently in economic planning problems.

In the chemical process industry, a periodic forcing of a reactor feed stream can sometimes give better yields and improved overall performances than a steady state operation [11]. With mechanical systems, we often encounter the situation where the reference commands to be tracked and/or disturbances to be rejected are periodic signals. This problem often arises in rotating machinery (mostly associated with engines, electrical motors), or machines performing a same task again and again [12]. Repetitive reference signals are common in robotics and other tracking tasks where a mechanical task is carried repetitively [13].
Figure 2.1 Block diagram of processes with periodic references/trajectories
2.2 Examples of Periodic Systems

2.2.1 Continuous Processes

A continuous system subjected to periodic reference trajectories or periodic disturbances is one type of periodic system [Figure 2.1]. An example of process where reference trajectory to be tracked is periodic is in computer memory storage device. The function of the controller is to maintain the read/write head accurately on a selected track of the rotating disk [14]. This is not a difficult problem if the center of the circular tracks falls perfectly on the center of the rotating disk. In practice, however, this does not hold true. The eccentricity of tracks causes a major problem of run out spindle. This requires the read/write head to follow a significant periodic reference trajectory.

One area where the need to reject periodic disturbance is highly stressed is space flight control [10]. In space crafts, some pieces of equipment such as cyro-pump, or small imbalance in a momentum wheel create periodic disturbances. It is desired to isolate and protect other sensitive equipment from these vibrations by rejecting these periodic disturbances. Repetitive control offers one option to such problem.

Periodic operations of many chemical processes result in better performances than their steady-state counterparts. In a large-scale industrial polyamine process, it was found by performing a dynamic optimization that reaction selectivity can be improved significantly by periodically switching between high and low levels of the feed [Figure 2.2] [15]. This periodic switching of the feed rate causes the reactor temperature to change periodically. However, the benefits of the periodic operation are lost if the reactor temperature is not maintained constant. This mandates the need of a good periodic control of the reactor temperature.

2.2.2 Batch processes

In a batch process, the control problem is usually given as a tracking problem for time-varying reference trajectories defined over a finite interval. Compared to continuous processes, batch process operations are by nature much more dynamic involving several
Figure 2.2 Polyamine process and optimal periodic trajectories of operation

Figure 2.3 Different phases of batch processes and its repetitive nature
transitions that cover large operating envelopes [Figure 2.3]. Since in many batch processes same runs are carried out repeatedly, same time varying trajectories are used batch after batch. Therefore, batch processes can also be considered as periodic processes. One good example is the rapid thermal processing (RTP) of chemical vapor deposition in wafer [16]. The ultimate control objective is to control the deposition of the thickness at the end of each RTP cycle. Once the run is finished, the terminal deposition thickness is measured, and the control profile for next cycle is updated by using the terminal thickness error to achieve the desired thickness.

2.2.3 Hybrid process

Periodic behavior in a process can also arise due to periodic discrete events occurring in an otherwise continuous process [Figure 2.4]. Simulated moving bed process, which performs chromatographic separation on a continuous basis, falls into this category. In batch chromatography, the separation is achieved based on the difference in the affinity of solutes to be supplied. This principle can be exploited further by moving both the solid phase and the solid phase counter-currently in order to maximize the average driving force. To avoid operational problems associated with moving the solid phase, in SMB the countercurrent flow of the solid phase is simulated by switching of inlet and product ports periodically [Figure 2.5]. This discrete cyclic switching of ports leads to periodic profiles of product concentrations at steady state [2].

2.3 Shortcomings of Conventional Controllers Applied on Periodic Processes

Despite their potential benefits, there are some added difficulties with periodic processes. They are considerably more difficult to operate than a steady state process. Moreover, the cyclic operation of a unit may affect the whole plant’s reliability. To realize the benefits of cyclic operations in many cases, control problems associated with the periodic operation must be addressed properly. Periodic processes are more complex in nature and pose special control challenges [11].
Figure 2.4 A continuous process with underlying discrete periodic events

Figure 2.5 Simulated moving bed
First, we will see how a PID controller and a MPC controller each perform on periodic systems. These two control methodologies have already proven to be very successful in practice.

2.3.1 PID Controller with Periodic Processes

PID control continues to be one of the most popular control techniques due to its simplicity and usefulness. It is well known that for linear time-invariant systems the integral action in the PID controller is able to give offset-free rejection of step disturbances. This zero steady state error is achieved by summing the output error over the sampling time index and giving a control signal proportional to this cumulative error. The integral action causes the controller output to change as long as an error persists in the controlled output.

However, for periodic systems error at steady state is dynamic because it is periodic in nature. The PID controller is not able to remove the dynamic periodic error completely. The periodic error information is not accumulated over the run index and cyclic error persists at the steady state [5] [Figure 2.6].

2.3.2 MPC with Periodic Processes

Model predictive control is a generic term for a group of related algorithms that make an explicit use of a process model to calculate control moves minimizing the objective function [17]. The main ideas of MPC are [Figure 2.7, 2.8]

1. Using an explicit dynamic model of plant to predict the effect of future moves on manipulated variables.
2. Calculating these moves such that they minimize a specific performance criterion while satisfying given operational constraints.
3. Solving this (quadratic) optimization problem in receding horizon manner, using the most recent measurements from the plant to update the prediction.

Model predictive control is especially useful for applications involving constraints on manipulated and/or controlled variables. MPC has been successfully applied in
Figure 2.6 PID controller with periodic trajectory tracking

Figure 2.7 Basic structure of MPC
Figure 2.8 Receding horizon MPC

Figure 2.9 MPC with periodic trajectory tracking
petroleum refineries and extended to numerous other application areas including those found in chemicals, food processing, automotive and aerospace industries. The reason for its popularity is that it addresses the key practical issues often encountered in process control problems including multivariable interactions, constraints, and potentially process nonlinearity all in a single systematic framework [3, 18, 19]. In spite of its advantages and predictive nature, the standard MPC is not able to reject periodic disturbances without offset [Figure 2.9]. The reason is that MPC assumes that error at current sample time will repeat itself at all subsequent sample times. Hence, it does not take into account the periodicity of disturbance and as a result cyclic error persists.

2.4 Challenges of Controlling Periodic Processes

Periodic processes can be thought of as two dimensional (2-D) systems, in which information is passed both along the finite ‘time’ axis and the infinite ‘run’ axis. This makes the design, analysis, and synthesis of controller more difficult [20].

To follow a periodic trajectory, one might try to attempt an open loop type control based on model inversion. But this would require the model accuracy to be unrealistically high. Furthermore, because the inverted model would contain a differentiator, the model inversion approach would be highly sensitive to the high frequency component in correction error [21].

2.5 Learning Controller Applied to Periodic Processes

Learning control can be viewed as an iterative way of finding the model inverse solution using input/output data. Hence, this method too will be sensitive to high frequency components in the error signal. The sensitivity to high frequency error can be adjusted through some tuning parameters without compromising the asymptotic performance, however. Because the inverse solution is found through on-line data rather than a predetermined model, it is significantly more robust to model errors than the open-loop model inversion approach.
Iterative Learning Control (ILC) and Repetitive Control (RC) are the two methodologies that deal with periodic processes. These two methodologies have some similarities and some differences which are discussed later in detail. However, one important feature that they share is that both are based on learning from previous trials to update the input profile such that the error in the next trial decreases.

The learning mechanisms are often associated with repeating actions of some sort. Movements of robot arm, read/write head of hard-disc drive, and mechanical systems with revolving mechanisms inside are examples of such periodic systems. The central idea is that, instead of teaching the controller, a suitable “learning” algorithm is implemented to let the controller achieve the desired control action automatically on its own. This is the principle idea behind repetitive control: repeat the same type of action and by repetition acquire the desired control action [13].

2.5.1. Iterative Learning Control (ILC)

ILC is a branch of learning control, primarily focused on batch processes that are operated repetitively. One such common operational scenario is found in the control of wafer temperature distribution in rapid thermal processing unit in semiconductor manufacturing.

The ILC approach observes an error generated on a given trial and then uses this information to make adjustments to the input for next trial to reduce the error in next run [Figure 2.10]. This differentiates ILC from conventional approaches where error information is not fed back from trial to trial. Refinement of input bias signals based on the general concept of ILC can potentially enhance the performance of tracking control systems significantly [22-24].

First order ILC algorithm updates the input trajectory according to \( u_{k+1} = u_k + H e_k \). In the above, \( H \) is called the “learning filter”, an operator that maps the error signal \( e_k \) to from the input adjustment \( u_{k+1} - u_k \) for the next run. The objective of ILC can roughly be viewed as design of this “learning” operator \( H \).
Figure 2.10 Basic structure of learning controller
2.5.2 Repetitive Control (RC)

Another operational scenario in learning control is tracking of the periodic signal when the process operates continuously in time. The goal of Repetitive Control is to ensure tracking of periodic reference trajectories or rejection of cyclic disturbances in continuous systems.

Repetitive control was first introduced in the control of proton synchrotron magnetic supply, where in order to obtain the desired proton acceleration pattern it was necessary to control the current supply in a specific curve with a very high precision requirement. Since protons turn around in the synchrotron, the reference commands are periodic [25]. Therefore, if one can implement a mechanism that has the ability of self-correction based on previous experience, then after a few trials, the system tracks this periodic signal.

2.6 Difference between ILC and RC

One major difference between iterative learning control and repetitive control is the fact that iterative learning control assumes a fixed initial condition for the system at the beginning of each period, whereas repetitive control assumes the system to have an initial condition at the beginning of a period that is the result of input actions in the previous periods. Thus iterative learning control considers motions such as a repeated pick and place operation of a robot, whereas RC considers continuous cyclic operations such as those occurring in rotating equipment.

Because in RC there is no returning of the system to the same initial condition at the start of next period, the transients can propagate across periods. Also, changes in control actions made near the end of one period strongly influence the error at the start of the next. This makes the stability issues of ILC and RC very different [23, 26].

The absence of resetting of the initial state in RC also makes it hard to update input from the error at the corresponding time in previous period when the true period is not an exact integer of sample times. When period is forced on the process, one may be able to have an integer number of samples per periods, but when it is determined by an
external periodic disturbance, this become less likely. Moreover, if the true period is not perfectly known or changes somewhat with time, RC will fail to keep in phase with the true period.

Both of these problems arise in continuous periodic processes when the period of the controller and the period of plant do not match for some reason.

2.7 Principle of RC

2.7.1 Internal Model Control

In servo system design, the internal model principle by Francis and Wonham play an important role [27]. The internal model principle states that the controlled output tracks a class of reference commands without a steady state error only if the generator for references is included in the stable closed-loop system. For example, no steady state error occurs for step reference commands in a stable feedback system, which has an integrator 1/s (generator of step function) in the loop.

For periodic signals this means to achieve asymptotic tracking/rejection, the controller needs to contain poles at the frequency of the periodic signal and its entire nonzero harmonics. This can easily be achieved by inclusion of the so called periodic signal generator, which is discussed next.

2.7.2 Periodic Signal Generator

Any periodic signal of a known period can be generated by a free system including a time lag element corresponding to the period with an appropriate initial function [28]. This periodic signal generator, also known as “memory loop”, can be realized by using a delay element as shown in Figure 2.11.

Now, according to the internal model principle, if we include a model of the exogenous signal generator in the loop and stabilize the unity feedback system, then the closed loop system [Figure 2.12] asymptotically tracks the signal.
Figure 2.11 Block diagram of a memory loop/periodic signal generator

Figure 2.12 Block diagram of a repetitive controller
2.8 Rejection Restriction in RC

Unfortunately, the direct implementation of this memory loop is often unstable. Using small gain theorem, it has been proved that for strictly proper plants it is impossible to construct a repetitive controller that exponentially stabilizes the system [28]. The theorem shows the impossibility of achieving the asymptotic rejection of general periodic inputs. This occurs because the stability condition previously shown is highly restrictive. This restriction comes from the apparently unrealistic over-specification of tracking high frequency signals.

Various modifications of the scheme were proposed to improve the situation. These typically amount to relaxation of the requirement for the asymptotic rejection of higher harmonics i.e. reducing the loop gain of the repetitive compensator for high frequencies range. One way to handle this is to introduce a low-pass filter. Because this tradeoff relationship between stability and tracking is frequency dependent, it is desirable and possible to take the filter in such a way that it is close to one in a low frequency range where tracking is important and that it is less than one (preferably close to zero) in the higher frequency range so as to improve on the stability condition [28].

This leads to the idea of modified repetitive control in which the delay element is preceded by a suitable proper function q(s) to deactivate the learning at high frequencies. The modified RC has decreased ability to track signals in the high frequency band, but it has the advantage of wider applicability. In reality this is not a drawback, as we do not need tracking in a very high frequency anyways.

2.9 Digital Repetitive Controller

The motivation of considering discrete time domain analysis and synthesis of RC is the ease of digital implementation of RC over its analog counterpart. Therefore, when implementing the RC in a real plant, it is quite natural to resort to digital control. With digital memory the repetitive compensator \((1/(1-z^{-N}))\) no longer remains an infinite dimensional, but merely a finite dimensional discrete system.
In digital RC, the unrealistic stability requirement for the controlled plant does not appear because discrete time analysis naturally limits the highest frequency component to be tracked [14]. Like other systems in discrete domain, for a discrete time repetitive controller, zero error may be obtained asymptotically for all the harmonics from the fundamental up to the Nyquist frequency.
CHAPTER 3

REPEATITIVE MODEL PREDICTIVE CONTROL

3.1 Shortcoming of RC

Repetitive controllers face a barrier in wider application to process control problems due to their inability to take into account certain characteristics of chemical processes. In RC the focus is on linear time invariant systems, while most chemical processes exhibit nonlinear dynamics. These systems must also operate within some physical and safety constraints. Furthermore, MIMO systems, non-square systems, and systems with large model errors cannot be easily handled with frequency domain based RC. Likewise, in many control situations, the trajectory to be tracked is not pre-defined but must be determined on-line based on some performance optimization criterion.

3.2 Advantages of Combining Repetitive Control with Model Predictive Control

These limitations can be easily overcome by embedding the concept of RC in MPC. Through its evolution in the last two decades, MPC is now regarded by many as the standard advanced control method for industrial processes. MPC has many attractive features, such as easy accommodation of conflicting control requirements of a multivariable system within an optimization criterion (including objective function and constraints) [18].

One shortcoming of MPC is that when applied on a periodic process it duplicates the same control error in repeated trials. It does not exploit the repetitive nature of a trajectory or disturbance.

For batch processes, Lee et al. [21, 24] have proposed a new control technique, called Batch Model Predictive Control (BMPC), which modifies the traditional MPC to account for error in previous runs besides responding to new disturbances as they occur during a run.
A counterpart of BMPC for continuous processes, termed Repetitive Model Predictive Control, has been developed by Lee and Seshatre [5, 29]. It combines the best of both RC and MPC as one integrated algorithm. The RMPC with its state space framework is more appealing than RC because it allows the user to exploit well established results in state-space theory [19].

3.3 Time Domain Interpretation of RC

RC is a frequency domain technique and to combine it with MPC-a time domain approach- it needs to be interpreted in time-domain. The foundation of RC is the periodic signal generator inside it. The function of this generator or memory loop is to store the error of previous trials for input calculation of future trials. The figure clearly illustrates that this memory loop acts like an integrator but instead of adding errors at consecutive sample times, it aggregates the cyclic error after each period [Figure 3.1]. This is the main idea behind the development of RMPC [5].

3.4 Development of RMPC

3.4.1 General Formulation of Control Problem

The dynamics of the process to be controlled can be described by a linear time discrete state space model

\[
x_k(t+1) = A(t)x_k(t) + B(t)u_k(t)
\]

\[
y_k(t) = C(t)x_k(t), \quad t = 0, \ldots, N - 1
\]

Where \(x\) is state vector, \(y\) is the output vector and \(u\) is input vector. \(k\) is the run index and \(t\) is the time index inside a run. Each period is divided into \(N\) equally spaced sample times.

In the standard formulation of RMPC, it is assumed that the period of disturbance /reference trajectory is constant and known perfectly. Furthermore, there is no mismatch between the period of controller and the periodic signal to be rejected /tracked.
Figure 3.1 Time domain interpretation of period-wise integration in repetitive control
Because the process is periodic and continuous, the state transition from one period to
other is described by the following equation:

\[ x_{k+1}(0) = x_k(N) = A(N-1)x_k (N-1) + Bu(N-1)u_k (N-1) \]  

(3.2)

Once again, this is the main difference between a periodic continuous process and a batch
process where the state is reset at the end of each run.

For a given reference trajectory \( r(t) \), the objective can be defined as finding \( u_k(t) \) to
\[
\min \sum_{t=0}^{N-1} \left\| y_k(t) - r(t) \right\|_{Q(t)} \quad \text{as } k \to \infty
\]  

(3.3)

Where \( \left\| \cdot \right\| \) denotes the weighted 2-norm of a vector. For a positive semi-definite matrix \( Q \),
the above objective leads to perfect asymptotic tracking if the objective function reaches
zero as \( k \to \infty \). Constraints on input and/or output can be considered while solving this
QP, which will turn it into a constrained optimization problem.

3.4.2 Lifting and Augmenting

The main idea behind RMPC is to store the periodic error for one trial and use it
for input calculation at the beginning of next run. To achieve this purpose, the underlying
time-variant system is transformed into a run-to-run, time-invariant model by “lifting”
and augmenting. This linear time-invariant model is then used for the formulation of a
linear optimal control problem, which takes the periodic characteristics of error into
account. From this run-to-run formulation a real-time structure for calculating optimal
moves at each sample time within a given run can also be developed. A detailed
description can be found in reference [5].

3.4.3 Run-to-Run Formulation

Recall the fact that RC removes periodic error by embedding an integrator over
the period. To develop a linear time-invariant control model for formulation of run-to-run
optimal control problem, the variables for whole run at all sample times are grouped into
augmented vectors as
Further, the system matrices A, B, and C at all sample times within one run are lifted leading to a compact representation. The lifted form of the system can be written as

\[ x_{k+1}(0) = \Phi x_k(0) + \Gamma u_k \]
\[ y_k = \Pi x_k(0) + Gu_k \]  

The matrices \( \Phi, \Gamma, \Pi, \) and \( G \) represent the state transition, (reversed) controllability, observability, and impulse response matrices respectively. They are defined for time interval from \( t = 0 \) to \( N \) for the time-varying linear system [equation (3.1)].

\[ \Phi = A(N-1)A(N-2)\ldots A(0) \]
\[ \Gamma = \begin{bmatrix} A(N-1)\ldots A(1)B(0) & A(N-1)\ldots A(2)B(1) & \ldots & B(N-1) \end{bmatrix} \]

\[ \Pi = \begin{bmatrix} C(0) \\ C(1)A(0) \\ \vdots \\ (C(N-1)A(N-2)\ldots A(0)) \end{bmatrix} \]

\[ G = \begin{bmatrix} 0 & 0 & \ldots & 0 & 0 \\ C(1)B(0) & 0 & \ldots & 0 & 0 \\ C(2)B(1)B(0) & C(2)B(1) & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (C(N-1)A(N-2)\ldots A(1)B(0)) & \ldots & \ldots & C(N-1)B(N-2) & 0 \end{bmatrix} \]

With the definition of \( \Delta \) as differencing of a variable with respect to run index

\[ \Delta u_k = u_k - u_{k-1} \quad \text{and} \quad \Delta u_k(t) = u_k(t) - u_{k-1}(t) \]  

The lifted system defined by equations (3.1) and (3.2) can be written as

\[ \Delta x_{k+1}(0) = \Phi \Delta x_k(0) + \Gamma \Delta u_k \]
\[ \Delta y_k = \Pi \Delta x_k(0) + G \Delta u_k \]  

Note that this system (3.8) includes the run-to-run transition error as the difference of the initial state of the next run \( x_{k+1}(0) \). The error trajectory \( e_k \) is defined as the difference of the run-invariant output vector \( y_k \) in the run \( k \) and the run-invariant output reference \( r \)

\[ e_k = y_k - r \]  

(3.9)
Substituting this error formulation into equation (3.8) and augmenting the state with the error trajectory, the above can be written as

\[
\begin{bmatrix}
\Delta x_{k+1}(0) \\
e_k
\end{bmatrix}
= \begin{bmatrix}
\Phi & 0 \\
\Pi & I
\end{bmatrix}
\begin{bmatrix}
\Delta x_k(0) \\
e_{k-1}
\end{bmatrix}
+ \begin{bmatrix}
\Gamma \\
G
\end{bmatrix}
\Delta u_k
\]

\[(3.10)\]

\[e_k = \begin{bmatrix}
\Pi & I
\end{bmatrix}
\begin{bmatrix}
\Delta x_k(0) \\
e_{k-1}
\end{bmatrix}
+ G\Delta u_k\]

These equations represent the periodic system resulting from augmentation of the run-to-run transition model with the periodic model error. In compact form, it can be represented as

\[z_{k+1} = \bar{\Phi}z_k + \bar{\Gamma}\Delta u_k\]

\[e_k = \bar{\Pi}z_k + G\Delta u_k\]

This representation includes the transition of periodic error from one run to another and therefore suitable for derivation of the run-to-run repetitive controller.

3.4.3.1 Unconstrained Run-to-Run RMPC

To achieve a good asymptotic performance and a reasonable transient performance, the controller design objective can be chosen as

\[
\min_{\Delta u_k} \sum_{k=1}^{p} \left\{ \| e_k \|_Q^2 + \| \Delta u_k \|_R^2 \right\}
\]

\[(3.12)\]

Term \( \Delta u_k \) is chosen over \( u_k \) to ensure that controller continues to take action as long as error is present in the controlled variable. In other words, using \( u \) in difference form serves to achieve the desired integral action over run index. \( p \) denotes the size of prediction horizon. In case \( p \) is set to \( \infty \), the formulation is referred to as ‘infinite horizon problem,’ while for a finite \( p \) it is called ‘finite horizon problem.’ \( Q \) and \( R \) are tuning parameters that change the rate of convergence but do not affect the asymptotic behavior.

For control formulation given by (3.11) and (3.12), the optimal control problem can be written as

\[
\min_{\Delta u_k} \sum_{k=1}^{\infty} \left\{ \| z_k \|_Q^2 + \| \Delta u_k \|_R^2 \right\}
\]

\[(3.13)\]
Where

\[
\bar{Q} = \begin{bmatrix}
\bar{P}^T Q \bar{P} & \bar{P}^T Q G \\
G^T Q \bar{P} & G^T Q G + R
\end{bmatrix}
\]  

(3.14)

In absence of constraints, this is equivalent to a standard infinite horizon LQ problem which can be easily worked out by solving an algebraic Riccati equation. The optimal state feedback control law can be found in the form of

\[
\Delta u_k = \bar{L}_e z_k = \bar{L}_i \Delta x_k(0) + \bar{L}_2 e_{k-1}
\]  

(3.15)

3.4.3.2 Constrained Run-to-Run RMPC

In presence of constraints, the dynamic programming based approach that leads to the algebraic Riccati equation is not feasible. Instead, at the beginning of each run the following finite horizon problem is solved.

\[
\begin{align*}
\min_{A_u_i} & \left\{ \sum_{i=k}^{k+p-1} \|e_i\|_Q^2 + \|\Delta u_i\|_R^2 \right\} \\
\text{such that} & \quad C \Delta u_i \leq c
\end{align*}
\]  

(3.16)

(3.17)

The hard constraints can be turned into soft constraints by incorporating a penalty term (or slack variable). This measure is taken because overly stringent hard constraints can lead to infeasibility. Slack variable is given a proper weighting in the objective function to make sure that soft constraints are not violated unnecessarily.

In the above constrained horizon formulation, the performance and computation effort required depends on the size of prediction horizon. It is difficult to assess beforehand how large a prediction horizon should be used. This problem can be alleviated by transforming the infinite horizon problem into an equivalent finite horizon problem under some specific assumptions and adding a terminal weight matrix. This can be done by considering the new QP:

\[
\begin{align*}
\min_{A_u_i} & \left\{ \sum_{i=k}^{k+p-1} \|e_i\|_Q^2 + \|\Delta u_i\|_R^2 + \|z_{k+p}\|_M^2 \right\} \\
\text{such that} & \quad C \Delta u_i \leq c
\end{align*}
\]  

(3.18)

(3.19)
and subject to some additional artificial constraints.

The crucial step in this approach is choice of terminal matrix $M$ and selection of additional constraints. Various methods for choosing these conditions can be found in work [30-32].
CHAPTER 4

PERIOD MISMATCH PROBLEM IN REPETITIVE CONTROL

RC has proven very useful for asymptotic tracking and rejection of periodic signals. It has been successfully applied to areas such as mechanical manipulators [33], computer disk drives [34] and spindle run out compensation [35]. Repetitive control offers the potential to completely cancel the effect of a periodic disturbance by taking its periodic nature into account. However, to be able to reject disturbances completely RC needs to know the exact period of reference/disturbance. At first, one might think that a slight mismatch in period will not cause a major effect on the performance. But, as we shall see, even a small mismatch in the period diminishes the controller tracking/rejection performance significantly.

4.1 Sensitivity analysis

The performance of the RC system depends on how precise the match is between the repetitive controller’s signal generator period and the actual signal period. The most common approach for rejection of a periodic disturbance is based on the internal model principle, which states that a model of the disturbance should be included in the feedback. The internal model for any periodic signal with a known period can be generated by a free dynamic system that has a positive feedback around a pure time delay.

The underlying property used is that a linear feedback system has perfect rejection at some frequency if the controller gain is infinite at that frequency. The repetitive controller generates a comb filter shaped closed loop sensitivity function [Figure 4.1]. The notches in the comb filter provide substantial sensitivity reduction but they are extremely narrow. The error magnitude at the Fourier harmonic frequencies of the periodic signal is proportional to
where \( T_p \) is the signal period and \( \hat{T}_p \) is the period used in the repetitive controller and hence \( \epsilon \) is the period mismatch ratio. As we know, for standard RMPC when there is no period mismatch i.e. \( \hat{T}_p = T_p \), the error is zero for first harmonic \( (n = 1) \). Even a slight mismatch of 1% shoots up the error magnitude ratio to 0.1 for the first harmonic. The error is even larger for higher harmonics [6].

4.2 Causes of Mismatch

4.2.1 Frequency Not Known Exactly

In the problem of active noise and vibration control, the noise source oftentimes has periodic components due to some rotating machinery generating the undesired signal.
and the frequency of these disturbances is usually not known a priori. Examples of such noises are engine noise in a turboprop aircraft, engine noise in automobiles and ventilation noise in HVAC.

4.2.2 Varying Frequency

In some situations the period of the disturbance signal is uncertain and slowly changing. For example, the periodic disturbance may be caused by an imbalance in the rotating machinery due to variations in the motor speed. Such situation requires an RC which can self-tune the respective controller signal generator period to match the external signal closely.

4.2.3 Non-divisibility of Period

Typical repetitive controller assumes that the period of the disturbance can be represented as some integer multiple of the sample time of the digital control system. But, in general, period will not be an exact integer multiple except by chance. This causes serious issues because repetitive control during the present time step wants to examine the information from the previous period of disturbance at an appropriate corresponding time. As data is only available at discrete sample times, unless the period is an integer number of time, this issue can significantly degrade the performance.

4.3 Current Approaches for Handling Period Mismatch

4.3.1 For Frequency Not Known Exactly and/or Changing

Several adaptive algorithms have been previously developed that can achieve good disturbance rejection with uncertainty in the frequency. A sampled data recursive algorithm with a resolution finer than the sampling period is applied to fine tune the period [8, 9]. However, these algorithms have a major disadvantage: the accuracy of period identification greatly depends upon the sampling interval. Computational issues
apart, sampling interval limits the resolution of the estimated period and hence puts a bound on period matching precision.

Manayathara et al. [36] designed a discrete time repetitive controller for the continuous steel casting process that is used for the rejection of periodic load disturbances with unknown period. A discrete, recursive scheme is applied to identify the period setting of the repetitive control algorithm.

4.3.2 For Period Non-integer Multiple of the Sampling Time

There are various possible approaches to the problem when the period of disturbance can not be represented as an integer multiple of sample-time.

4.3.2.1 Changing the Sampling Rate

Adopting the sampling period according to the signal period time seems like the simplest way; however, a good control algorithm should be able to cater itself to the hardware instead of the other way around. Besides, in practice it not usually easy to arbitrarily change the sampling period without causing serious implementation issues. Still, the main attractiveness of this scheme is that by adjusting the sample time the period mismatch may be reduced significantly if not completely eliminated.

In literature, discrete time self-tuning RCs have been proposed, which adapt the sample interval to a given periodic signal, based on the period identification scheme [6, 8]. The fine adaptation of the controller sampling interval attempts to make the identified signal period an exact integer multiple of the controller’s sampling time and renders a superior performance than that of conventional fixed sampling interval repetitive controllers. Consider the case where the actual period of disturbance is 4.95 minutes while the sampling is done at every minute. In this case, changing the sampling time to 0.99 minutes makes the period an exact multiple of sampling time.

4.3.2.2 Extending the Learning Period

Another possible way to withstand a period mismatch is to extend the learning period from $T_{actual}$ to $T_{extended}$, which is the smallest common multiple of $T_{actual}$ and $T_{sampling}$.
[37]. For example, if the actual period is 4.75 minutes and sampling time is 1 minute, the learning period can be chosen as 19 minutes. Though this makes the controller learn more slowly, the controller is able to reject the disturbance of period 4.75 completely. A potential problem of this approach is that the new learning period could be intolerably long. For example, consider a case when actual period is 4.95 and sampling time is 1 minute. Then, the extended period would have to be 99 minutes, which makes the controller extremely slow in learning.

4.3.2.3. Introducing Virtual Sampling

One more complex way of addressing this problem is to generate virtual sampling instants from the actual ones, using some interpolation techniques. When interpolating the accumulated signal term in a recursive repetitive control law, the dependence on error gets spread over a larger set of time steps as one goes back further in time. To manage this matter, one can use a higher order repetitive control law that picks out the error or interpolated errors of interest for a number of repetitions back in time, and then by applying the recursive accumulation term with associated interpolation [10].

4.4 Need for Robust RMPC

The above developments show the most common causes of mismatch between the controller’s assumed period and the actual period of disturbance/reference. The mismatch has substantial adverse effect on the controller’s performance.

One way this performance can be improved is by reducing the inaccuracy in the estimation of frequency with better period estimation, though the period identification schemes have their limitations. On the other hand, errors associated with truncation and round-off error resulting from forcing the period to be an integer multiple of sampling time, are difficult to avoid.

Because the basis principle in RMPC is taken from RC, RMPC also suffers from the same performance degradation when there is a mismatch between the controller period and actual period of disturbance.
Therefore, a “Robust” RMPC is desired, which is robust with respect to small period mismatches between the controller and the reference/disturbance signal. The aim is to extend the standard RMPC in such a manner that it neither requires an exact period estimation nor demands the change in sampling time for small period mismatches, while being able to sustain its performance.
CHAPTER 5

DEVELOPMENT OF ROBUST RMPC

RMPC equips MPC with the additional capability to handle periodic processes by integrating the principle of RC into it. The performance of RMPC, like of RC, depends strongly on the match between controller period and actual period of disturbance. Even a small period mismatch can diminish RMPC tracking/rejection properties significantly. This is not surprising given that RMPC is built by borrowing the fundamental principle of RC.

5.1 Robust Repetitive Control

To deal with small period uncertainties, Steinbuch [7] has developed a robust repetitive control technique based on the idea of using “multiple memory loops.” The idea is to modify the frequency dynamics of RC and increase its robustness to small period changes, by using more than one memory loop. The function of multiple memory loops can be roughly viewed as storing errors for more than one previous trial to make more “robust” decisions. This is different from the standard RC where only last run error is used for the input update [Figure 5.1, Figure 5.2]. In case of period mismatch, input update based on multiple memory loops can lead to more robust control than that based on a single memory loop, if properly applied.

A very similar approach has been used by Singh and Vadali in robust time delay control [38] to minimize the residual vibration of structures or lightly damped servomechanisms. They use multiple time delays in conjunction with a proportional part to cancel the dynamics of the system in a robust fashion.

The schematic of a generalized repetitive controller (for three memory loops) is shown in Figure 5.3. For a generalized period repetitive controller with multiple periodic signal generators, the transfer function from e to z is
Figure 5.1 Block diagram of basic structure of a repetitive controller

Figure 5.2 Standard single memory loop

Figure 5.3 Multiple memory-loops for generalized repetitive controller
\[
\frac{z}{e} = G(s) = \frac{H(s)}{1 - H(s)}
\]  
(5.1)

\[
H(s) = \sum_{i=1}^{N} W_i e^{-i\omega T_p}
\]  
(5.2)

For standard RC, with only one memory loop [Figure 5.2], the corresponding transfer function is

\[
G(s) = \frac{e^{-i\omega T_p}}{1 - e^{-i\omega T_p}}
\]  
(5.3)

\[
H(s) = e^{-i\omega T_p}
\]  
(5.4)

In single memory loop RC, at harmonic frequencies \(\omega = n2\pi / T_p\), \(H(s)\) is equal to one, making the gain of the transfer function infinite. This is the desired trait of controller gain, which enables RC to reject a periodic disturbance and its entire harmonics.

Again, for the generalized repetitive controller we would like the gain \(G(s)\) to be infinite at harmonics, i.e. \(H(s) = 1\) for \(s = jn2\pi / T_p\). This means

\[
H(jn2\pi / T_p) = \sum_{i=1}^{N} W_i e^{-jn2\pi} = 1
\]  
(5.5)

\[
\sum_{i=1}^{N} W_i = 1
\]  
(5.6)

Now, to improve this generalized RC robustness with respect to period mismatch, additional requirements can be imposed to reduce variation in \(H(s)\) for small changes in period around the harmonics. Mathematically speaking, we desire

\[
\frac{\partial H(s)}{\partial T_p} \bigg|_{s=jn2\pi / T_p} = 0
\]  
(5.7)

which gives

\[
\sum_{i=1}^{N} W_i i = 0
\]  
(5.8)

The sensitivity to period variation can be further reduced by setting higher derivatives of \(H(s)\) with respect to period equal to zero. For an \(N\) delay-line generalized RC structure, this can be done for up to \((N - 1)^{th}\) derivatives, resulting in equations:
\[ \sum_{i=1}^{N} W_i i^{(N-1)} = 0 \]  \hfill (5.9)

The above equations (5.6), (5.8) and (5.9) are used to calculate the proper weights in a generalized RC. The weight selection process can be easily illustrated with an example of two memory-loops RC. From equations (5.6), (5.8) \( W_1 + W_2 = 1, W_1 + 2W_2 = 0 \); hence the solution is \( W_1 = 2, W_2 = -1 \).

The frequency response of generalized RC [Figure 5.5] is plotted and compared with standard RC [Figure 5.4]. With use of multiple memory loops, the gain of a standard controller widens around the harmonics. Hence, for slight mismatch in period, the gain of a periodic generator does not fall down as sharply, and the robust controller can reject periodic disturbances better than standard RC. It is important to note that robustness of multiple memory loops RC increases with the number of delay lines used [Figure 5.6].

5.2 Robust Repetitive Model Predictive Controller Construction

Using the analogy between RC and RMPC, one can try to construct a robust RMPC technique by properly extending the robust RC approach to RMPC methodology. The basic assumption in this development is that the period of disturbance is constant, and there is a slight mismatch between the controller and the disturbance period.

Because MPC is a time domain approach, the first step in developing RMPC from RC was to translate the principle of period signal generator- a frequency domain based argument - into time domain. The insight that RC acts as a period-wise integrator was later imported into MPC to construct RMPC which is able to reject/track the periodic signals.

Along the same line, before developing Robust RMPC, we try to transfer the principle used in Robust RC approach to time domain and then combine it with MPC.
Figure 5.4 Frequency response of single memory loop repetitive controller

Figure 5.5 Frequency response of multiple memory loop repetitive controller
standard repetitive controller (N=1); generalized repetitive controller (N=2,3)
5.2.1 Interpretation of Multiple Memory Loops in RMPC Framework

In discrete time, integration of output error can be realized by the following algorithm $e'_t = e_t + e_{t-1}'$, where $e'_t$ the accumulated error is used to derive (integral) control action and $e_t$ is the error at $t$ sampling point.

As we know, to reject periodic disturbance one needs to include an integrator that integrate over runs rather than over time. In discrete time this can easily be achieved by replacing $e'_t$ by $e'_k$ (“lifted” error vector for the whole run) and integrating this lifted error at the beginning of each run $k$.

$$e'_k = e'_k + e'_{k-1}$$

$$e'_k = e'_k + z^{-N} e'_k$$

$$e'_k = \frac{e'_k}{(1 - z^{-N})} \quad (5.10)$$

Figure 5.6 Change in periodic signal generator magnitude with respect to mismatch standard repetitive controller (N=1); generalized repetitive controller (N=2, 3)
Here $N$ is the length of period in terms of sampling time.

The above algorithm can be represented described using the block diagram in Figure 5.7.

\[ \frac{1}{1 - z^{-N}} \]

\[ e_k \rightarrow \frac{1}{1 - z^{-N}} \rightarrow e_k' \rightarrow C \rightarrow u_k \]

Figure 5.7 Block diagram of a period-wise integrator

For linear systems, blocks 1 and 2 can be switched without changing the input-output relation.

\[ e_k \rightarrow \frac{1}{1 - z^{-N}} \rightarrow \Delta u_k' \rightarrow u_k \]

\[ C \rightarrow \frac{1}{1 - z^{-N}} \]

Figure 5.8 Alternate representation of a periodic integrator

This dashed box in new structure indicates that by using single run error $e_k$ to determine calculate $\Delta u_{k+1}$, one can generate the same integral action as using integral period error $e_k'$ to calculate $u_{k+1}$. This understanding can be used to explain how RMPC achieves periodic integral action by penalizing $\Delta u$ instead of $u$ when minimizing periodic error $e$ in objective function. Keeping this in mind, we now try to develop a Robust RMPC based on double memory loops robust RC.
For two delay lines, the generalized RC memory loop has the following structure [Figure 5.9]

![Block diagram of a double memory loop](image)

Figure 5.9 Block diagram of a double memory loop \( W_1 = 2, W_2 = -1 \)

In discrete time, delay \( e^{-sT} \) can be replaced by equivalent \( z^{-N} \) and the simplification of the double memory loop structure in Figure 5.9 in RC structure gives relationship between period error and input as shown in Figure 5.10.

![Discrete time equivalent of a double memory loop structure](image)

Figure 5.10 Discrete time equivalent of a double memory loop structure

Now, without changing the input-output relationship between \( u_k \) and \( e_k \), the controller C is positioned immediately after the period error \( e_k \) in above process structure.

![Alternative representation of double memory loop repetitive control](image)

Figure 5.11 Alternative representation of double memory loop repetitive control
Comparison of this structure with arrangement in Figure 5.8 used for RMPC interpretation, suggests that one should duplicate the robustness of RC in RMPC by penalizing $\Delta^2 u$ instead of $\Delta u$ in the control quadratic objective function. The new objective function can be written as

$$
\min_{\Delta^2 u} \sum_{k=1}^{N_u} \left( \|e_k\|_Q^2 + \|\Delta^2 u_k\|_R^2 \right)
$$

(5.11)

In accordance with the new objective function, equations (3.11) are differenced producing

$$
\Delta z_{k+1} = \Phi \Delta z_k + \Gamma \Delta^2 u_k
$$

$$
e_k = e_{k-1} + \Pi \Delta z_k + G \Delta^2 u_k
$$

(5.12)

After augmentation, the system can be written as

$$
\begin{bmatrix}
\Delta z_{k+1} \\
e_k
\end{bmatrix} = \begin{bmatrix}
\Phi & 0 \\
\Pi & I
\end{bmatrix} 
\begin{bmatrix}
\Delta z_k \\
e_{k-1}
\end{bmatrix} + \begin{bmatrix}
\Gamma \\
G
\end{bmatrix} \Delta^2 u_k
$$

(5.13)

$$
e_k = \begin{bmatrix}
\Pi \\
G
\end{bmatrix} \begin{bmatrix}
\Delta z_k \\
e_{k-1}
\end{bmatrix} + G \Delta^2 u_k
$$

System (5.13) can be expressed compactly as

$$
\zeta_{k+1} = \Phi \zeta_k + \Gamma \Delta^2 u_k
$$

$$
e_k = \Pi \zeta_k + G \Delta^2 u_k
$$

(5.14)

This control problem formulation is similar to the one developed in standard RMPC (3.11), except that it contains $\Delta^2 u_k$ term in place of $\Delta u_k$, and $\Phi, \Gamma, \Pi$ have changed to $\Phi, \Gamma, \Pi$.

This is the Robust RMPC structure based on generalized RC with two memory loops. Similar analysis can be carried out if one wishes to develop Robust RMPC based on a higher number of memory loops. In RMPC, one can have added robustness by penalizing the higher degree of differencing. For example, robustness of RC with three memory loops can be attained in RMPC by penalizing $\Delta^3 u_k$ in the objective function.

### 5.2.2 Unconstrained Run-to-Run Robust RMPC
Similar to Run-to-Run RMPC formulation, optimal control problem for Robust RMPC can be written as

\[
\min_{\Delta^2 u_k} \sum_{k=0}^{\infty} \left( \left\| \begin{bmatrix} \xi_k \\ \Delta^2 u_k \end{bmatrix} \right\|_Q^2 \right)
\]

(5.15)

Where

\[
\bar{Q} = \begin{bmatrix}
\Pi^T Q \Pi & \Pi^T Q G \\
G^T Q \Pi & G^T Q G + R
\end{bmatrix}
\]

(5.16)

In absence of constraints, this infinite horizon LQ regulation problem can be easily solved with Algebraic Riccatti equation. The input update law is then given by

\[
\Delta^2 u_k = \bar{L}_2 \Delta \xi_k + \bar{L}_1 \Delta^2 x_k (0) + \bar{L}_2 \Delta e_{k-1} + \bar{L}_3 e_{k-1}
\]

(5.17)

The above control law shows how in Robust RMPC the error information of not only last trial but even one before that are stored and used in updating the control input for the next run..

5.2.3 Constrained Run-to-Run Robust RMPC

In presence of constraints, it is not feasible with dynamic programming approach to derive an analytical control update law as in the unconstrained case. The resulting problem becomes constrained least square problem and must be solved numerically. Efficient off-the-shelf solvers are available for this quadratic programming (QP) problem. In addition, the stability and performance properties of the finite receding horizon control can be ensured using various methods, details of which can be found in standard model predictive control literature [39].

5.2.4 Real-time Control for Robust RMPC

Next, we try to develop a real time control algorithm that utilizes the feedback information as it becomes available within each run, instead of waiting until the end of
run to update the system. To achieve this objective, we try to convert period-to-period system description into an equivalent, periodically time varying system description.

For this purpose, we use the following definition of error at time $t$.

$$ e_k(t) = y_k(t) - r \quad \text{assuming} \quad \Delta x_j(j) = 0, \quad j \geq t $$  \hspace{1cm} (5.18)

The equation (3.1) is differenced and written as,

$$ \Delta x_k(t+1) = A(t)\Delta x_k(t) + B(t)\Delta u_k(t) $$

$$ y_k(t) = y_{k-1}(t) + C(t)\Delta x_k(t) $$  \hspace{1cm} (5.19)

Using the definition (5.18) of error at time $t$,

$$ e_k(t+1) = e_{k-1} + \begin{bmatrix}
C(0)\Delta x_k(0) \\
\vdots \\
C(t-1)\Delta x_k(t-1) \\
C(t)\Delta x_k(t) \\
0 \\
\vdots \\
0
\end{bmatrix} $$  \hspace{1cm} (5.20)

This gives,

$$ e_k(t) = e_{k-1}(t) + C(t)\Delta x_k(t) $$

$$ \begin{bmatrix}
0 \\
\vdots \\
0
\end{bmatrix} $$

with $C(t)$ \hspace{1cm} (5.21)

Now, the following periodically time varying input/output system description results:
\[
\begin{bmatrix}
\Delta x_k(t+1) \\
\mathbf{e}_k(t+1)
\end{bmatrix}
= \begin{bmatrix} A(t) & 0 \\ C(t) & I \end{bmatrix}
\begin{bmatrix}
\Delta x_k(t) \\
\mathbf{e}_k(t)
\end{bmatrix}
+ \begin{bmatrix} B(t) \\ 0 \end{bmatrix}
\Delta u_k(t)
\] (5.22)

\[
e_k(t) = \begin{bmatrix} C(t) & H(t) \end{bmatrix}
\begin{bmatrix}
\Delta x_k(N) \\
\mathbf{e}_k(t)
\end{bmatrix},
\]

where \( H(t) = \begin{bmatrix} 0 \\
\vdots \\
1 \\
\vdots \\
0 \end{bmatrix} \leftarrow t^{th} \) row element

The above system is differenced once again, which gives
\[
\begin{bmatrix}
\Delta^2 x_k(t+1) \\
\Delta \mathbf{e}_k(t+1)
\end{bmatrix}
= \begin{bmatrix} A(t) & 0 \\ C(t) & I \end{bmatrix}
\begin{bmatrix}
\Delta^2 x_k(t) \\
\Delta \mathbf{e}_k(t)
\end{bmatrix}
+ \begin{bmatrix} B(t) \\ 0 \end{bmatrix}
\Delta^2 u_k(t)
\]

\[
e_k(t) = e_{k-1}(t) + \begin{bmatrix} C(t) & H(t) \end{bmatrix}
\begin{bmatrix}
\Delta^2 x_k(t) \\
\Delta \mathbf{e}_k(t)
\end{bmatrix},
\] (5.23)

\[
\begin{bmatrix}
\Delta^2 x_{k+1}(0) \\
\Delta \mathbf{e}_{k+1}(0)
\end{bmatrix}
= \begin{bmatrix} \Delta^2 x_k(N) \\
\Delta \mathbf{e}_k(N) \end{bmatrix}
\]

In run \( k \),
\[
e_{k-1}(0) = e_{k-1}(1) = \ldots = e_{k-1}(N)
\]

In other words, \( e_{k-1}(t+1) = e_{k-1}(t) \)

Using this, system (5.23) can be augmented as
\[
\begin{bmatrix}
\Delta^2 x_k(t+1) \\
\Delta \mathbf{e}_k(t+1) \\
e_{k-1}(t+1)
\end{bmatrix}
= \begin{bmatrix} A(t) & 0 & 0 \\ C(t) & I & 0 \\ 0 & 0 & I \end{bmatrix}
\begin{bmatrix}
\Delta^2 x_k(t) \\
\Delta \mathbf{e}_k(t) \\
e_{k-1}(t)
\end{bmatrix}
+ \begin{bmatrix} B(t) \\ 0 \\ 0 \end{bmatrix}
\Delta^2 u_k(t)
\] (5.24)

\[
e_k(t) = \begin{bmatrix} C(t) & H(t) & H(t) \end{bmatrix}
\begin{bmatrix}
\Delta^2 x_k(t) \\
\Delta \mathbf{e}_k(t) \\
e_{k-1}(t)
\end{bmatrix},
\]

Similarly, at the end of run \( k \) (and for run \( k+1 \))
\[
e_k(0) = e_k(1) = \ldots = e_k(N)
\]

which gives \( e_k(0) = e_k(N) \)
\[ e_k(0) = e_k(N) - e_{k-1}(N) + e_{k-1}(N) \]
\[ = \Delta e_k(N) + e_{k-1}(N) \]

with above relation the state transition from run \( k \) to \( k+1 \) can be written as
\[
\begin{bmatrix}
\Delta^2 x_{k+1}(0) \\
\Delta e_{k+1}(0) \\
e_k(0)
\end{bmatrix}
= \begin{bmatrix}
\Delta^2 x_k(N) \\
\Delta e_k(N) \\
e_{k-1}(N) + \Delta e_k(N)
\end{bmatrix}
= \begin{bmatrix}
I & 0 & 0 \\
0 & I & 0 \\
0 & I & I
\end{bmatrix}
\begin{bmatrix}
\Delta^2 x_k(N) \\
\Delta e_k(N) \\
e_{k-1}(N) + \Delta e_k(N)
\end{bmatrix}
\]

(5.25)

The system given by equations (5.24) and (5.25) can be denoted as
\[ \xi_k(t+1) = \hat{A}(t)\xi_k(t) + \hat{B}(t)\Delta u_k(t) \]
\[ e_k(t) = \hat{C}(t)\xi_k(t) \]
\[ \xi_{k+1}(0) = \Omega\xi_k(N) \]

(5.26)

The optimal objective for the above representation can be written as
\[
\min_{\Delta u_k(t)} \left( \sum_{k=1}^{N-1} \left( \|\xi_k(t)\|_{Q(t)}^2 + \|\Delta^2 u_k(t)\|_{R(t)}^2 \right) \right)
\]

(5.27)

where \( Q(t) = \hat{C}^T(t)Q(t)\hat{C}(t) \)

To solve above problem, first the problem is lifted in the form of equation (3.5)
\[ \xi_{k+1}(0) = \Phi\xi_k(0) + \Gamma \Delta^2 u_k \]
\[ e_k = \Pi \xi_k(0) + \hat{G} \Delta^2 u_k \]

(5.28)

where \( \Phi, \Gamma, \Pi \) and \( \hat{G} \) are defined as equation (3.6) for periodic system \( \hat{A}(t), \hat{B}(t), \hat{C}(t) \).

In lifted form, objective function can be written as
\[
\min_{\Delta u_k(t)} \sum_{i=k+1}^\infty \left( \|e_i\|_{Q(t)}^2 + \|\Delta^2 u_i\|_{R(t)}^2 \right)
\]

(5.29)

or
\[
\min_{\Delta u_k(t)} \sum_{i=k+1}^\infty \left( \|\xi_i(0)\|_{Q(t)}^2 \right)
\]

(5.30)

where \( \hat{Q} = \left[ \begin{array}{cc}
\Pi^T Q \Pi & \Pi^T Q \hat{G} \\
\hat{G}^T Q \Pi & \hat{G}^T Q \hat{G} + R
\end{array} \right] \)

(5.31)

In absence of constraints, solving the algebraic Riccati equation for LQ problem in equation (5.30) gives the optimal value if the infinite horizon cost as \( \xi_{k+1}(0)^T \hat{M}_{LQ} \xi_{k+1}(0) \)
where $\hat{\mathbf{M}}_{LQ}$ is the solution of the algebraic Riccati equation.

Using the infinite horizon cost, the original optimization problem (5.27) can be written as

$$
\min_{\Delta^2 u_k(t)} \left( \sum_{r=0}^{N-1} \left( \|\xi_k(t)\|_{Q_r(t)}^2 + \|\Delta^2 u_k(t)\|_{R_r(t)}^2 \right) \right) + \xi_{k+1}(0)^T \hat{\mathbf{M}}_{LQ} \xi_k(0) 
$$

(5.32)

The solution of the above minimization can be calculated by iterating on the corresponding Riccati difference equation. This gives the periodic feedback policy in the form of

$$
\Delta^2 u_k(t) = -L(t)\xi_k(t) 
$$

(5.33)

This optimal gain $L$ depends on time index within a run, but does not change from one run to another.

5.3 Results Comparison between RMPC and Robust RMPC

The performance of Robust RMPC is compared with that of standard RMPC for a case when the period of controller is fixed (5 sec) but the period of disturbance is slightly changed. An RMPC system and an R-RMPC system (based on double memory loops) are applied to suppress a periodic disturbance. It is evident from the plot that Robust RMPC performs better than the standard RMPC by rejecting a disturbance of a slightly different period from the assumed one [Figure 5.12, Figure 5.13].

To measure the degree of robustness with respect to period mismatch, the mismatch is increased from 0.01 to 0.05 to 0.1 sec. The results show that with the increasing mismatch, the performance degrades, though R-RMPC’s performance degrades much more gracefully than the standard RMPC’s performance [Figure 5.14]. It is clear that with an increased number of memory loops used to develop R-RMPC, the resulting controller becomes more robust to period mismatch [Table 5.1]. However, this robustness is achieved at the price of increased computational load.
Figure 5.12 Performance of standard RMPC with and without period mismatch

Figure 5.13 Performance of Robust RMPC with and without period mismatch
Table 5.1 Steady state error with RMPC and R-RMPC

<table>
<thead>
<tr>
<th>Controller Period</th>
<th>Disturbance Period</th>
<th>Steady State Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>RMPC (2 delay lines)</td>
</tr>
<tr>
<td>5.00</td>
<td>5.00</td>
<td>6.9E-15</td>
</tr>
<tr>
<td>5.00</td>
<td>4.99</td>
<td>4.3E-3</td>
</tr>
<tr>
<td>5.00</td>
<td>4.95</td>
<td>9.3E-3</td>
</tr>
<tr>
<td>5.00</td>
<td>4.90</td>
<td>9.9E-3</td>
</tr>
</tbody>
</table>

Figure 5.14 Performance of RMPC and R-RMPC as a function of period mismatch
5.4 Application of Robust RMPC to Mechanical Motion Tracking Control

The usefulness of the proposed R-RMPC algorithm is shown here by applying it to a mechanical motion tracking control problem. The system consists of an air bearing supported XY stage driven by DC servo motors through ball screws. For testing purpose only the X axis motion would be considered.

One of the many useful applications of this scheme could be tracking control in a compact disk drive, which follows a similar mechanism.

The plant model used for controller design is:

\[
G_p(z^{-1}) = \frac{0.8273z^{-1} - 0.3252z^{-2} - 0.0432z^{-3} - 0.0005z^{-4}}{1 - 0.1644z^{-1} - 0.3333z^{-2} - 0.0433z^{-3} - 0.0005z^{-4}}
\] (5.34)

First, a simulation is performed for the case of no period mismatch between the disturbance period and the controller period. A sinusoidal signal wave of 20 sample intervals long (100 ms) is added as a disturbance to the output. For the no mismatch case, the RMPC’s performance is excellent as the steady state error is negligibly small (in the order of \(10^{-14}\)). Now, to simulate a period mismatch case, the period of disturbance wave is changed slightly from 20 to 20.3 sample intervals long, but the period of RMPC is kept fixed at 20.

![Figure 5.15 RMPC vs. Robust RMPC with mechanical motion tracking control](image.png)
The root mean square for this case is $5.3 \times 10^{-3}$. With Robust RMPC based on double memory loops, the root mean square for the same mismatch drops to $7 \times 10^{-4}$, which is an order of magnitude less than the error obtained with the standard RMPC. Further reduction in asymptotic error can be achieved by using higher order robust RMPC. For example, 3 and 4\textsuperscript{th} order robust RMPC are able to bring down the error to $7 \times 10^{-5}$ and $1 \times 10^{-5}$ respectively. Figure 5.15 clearly illustrates the benefits of using R- RMPC over the standard RMPC for cases when the disturbance frequency cannot be measured precisely and/or cannot be implemented exactly in the repetitive controller. By using multiple memory loops, better performance can be achieved for small changes in the period length.
Periodic processes are systems that either operate periodically [type 1] or their reference/disturbance is periodic in nature [type 2]. Application of RC is limited to the latter class of systems viz. tracking/rejection of periodic signals. RMPC/R-RMPC does not make a distinction between these two cases and extends RC to the MPC framework to deal with both types of periodic systems. However, in presence of a period mismatch, the RMPC/R-RMPC approach can give very different results depending on the type of system.

For further discussion, we categorize the periodic process in two classes according to the following definitions.

Class I: Systems that are intrinsically periodic due to their periodic operation mode. These systems are periodically time varying but invariant from run to run. The simulated moving bed processes and periodically operated polyamine reactors are two examples of this kind.

Class II: Systems that are time-invariant in nature but a periodic controller is needed to reject/follow the periodic signals e.g. mechanical motion tracking control that requires suppression of periodic noises.

6.1 Period Mismatch in Class I Systems

In the preceding chapter, we presented the results for a mechanical motion tracking device which is a Class II system. The period mismatch resulted in poor tracking performance but did not cause instability. Moreover, it was shown how this poor performance could be improved by applying R-RMPC based on multiple memory loops.

Now, to see RMPC performance with Class I system in presence of a period mismatch, we apply it to a simulated moving bed process. Details of SMB and results of RMPC with SMB (for no mismatch case) are available in [29].
First a nonlinear model of SMB is linearized along the periodic reference trajectories to give a linear, periodically varying description of plant. This PTV model is now used for controller design. For evaluation purpose, the controller is forced to follow a constant trajectory with no constraint assumption. When no mismatch exists between controller period and SMB, RMPC is able to asymptotically follow the given constant trajectories. To simulate a period mismatch case, an RMPC with an assumed period of eight minutes is applied on an SMB of 7.95 minute switching time. The measurements are taken at interval of one minute, and the control calculation is performed every eight minutes – as in run to run control. With this scheme, the controller quickly goes unstable after a few runs.

To understand this behavior in details, a simple example of an LTI system is created. The states of system are switched periodically to simulate the typical behavior of SMB [Figure 6.1]. The period of the controller is kept constant at 15 while the plant switching period is changed from 15 to 14.99. Additional cases with smaller period mismatches are also tried. But as long as there is a mismatch between the period of controller and the plant, the system eventually goes unstable [Figure 6.2]. With a smaller mismatch, it takes only longer for the system to diverge. These results are similar to the ones obtained for the SMB process.

6.1.2 Results Discussion

As simulation results indicate, using an inexact period in the RMPC controller leads to instability in Class I systems while with Class II system it results only in poor performance.

For Class II systems, because the periodicity is caused only by exogenous signal, the system stability is not affected with inexact period in controller. For Class I systems, an explanation for their unstable behavior is proposed as following.

For these systems, after the model is put in a lifted form, the run invariant formulation is same as the standard MPC formulation. The only main difference is that RMPC/R-RMPC calculates input moves at the beginning of each run rather than at each sample time. In lifted form the model used for run-to-run prediction remain constant in
Figure 6.1 A prototypical switching system

Figure 6.2 Instability due to period mismatch in the switching system
each subsequent run. However, due to period mismatch, the actual process behaves very differently from the lifted model. This causes a plant/model mismatch that continues to vary from one run to another. For example, if the true period of a process is five and the controller uses an inaccurate period of four, the plant/model mismatch would occur in the fashion shown in Figure 6.3.

![Figure 6.3 Illustration of plant/model error due to period mismatch in type 1 systems](image)

It is obvious that plant/model mismatch due to inexact period keeps on changing from one run to another and can become extremely large in some runs, leading to the unstable behavior. The smaller the period mismatch, the more number of runs it takes to reach instability, but eventually it happens.

### 6.2 Period Mismatch vs. Other Parameters Mismatch

Above discussion does not imply that RMPC or R-RMPC cannot tolerate any mismatch between the plant and the model. But, there is a big difference between
plant/model mismatch due to an inaccurately assumed period and plant/model mismatch due to wrong model parameters in general.

For example, compare the model/plant error in Figure 6.4 for no period mismatch with period mismatch case in Figure 6.3. For period mismatch case, the plant/model error varies from one run to another; but for exact period, the plant/model error is run-invariant.

From this, one can infer that period mismatch in endogenous/internal periodic systems translates into plant/model mismatch which varies with runs and potentially grows very large causing instability. On the other hand, with no period mismatch, the RMPC is able to tolerate some degree of parameters inaccuracy.

![Figure 6.4 Illustration of plant/model error due to parameter mismatch only](image)

The conclusion is that for internally periodic systems (Class I), the period mismatch translates into significant out of phase behavior between plant and model and leads to instability issues, which will not be addressed by the aforementioned modification of R-RMPC in general. The problem with periodically varying systems is that the extent of plant/model mismatch changes from one run to another and sooner or eventually becomes too large beyond the controller’s inherent robustness to handle it.
Period mismatch can have serious consequences with a periodically operated nonlinear process, which results in a PTV system when linearized around some reference trajectory for controller design. These highly nonlinear processes are the ones most likely to benefit from the periodic operations. Additionally, if the process is only mildly nonlinear, leading to mildly periodically varying processes, the application of periodic controller would not be needed.

Fortunately, for periodically operated process the period of process is usually known and forced on the controller, which makes a mismatch between the controller period and the process period less likely. However, if a period needs to be changed somewhat from time-to-time, it may necessitate the cumbersome task of having to redesign the controller. In addition, a very small amount of mismatch may be unavoidable due to limitations in machine precisions, etc.

6.3 Simple Practical Approach to Tackle Small Period Mismatches for Class I Systems

Since RMPC/R-RMPC approach becomes unstable in presence of a period mismatch when applied on a SMB (a Class I system), a simple approach is tried here for this case. In this scheme, instead of applying the controller input moves for the assumed (and inexact) period on the plant, they are applied for the actual process period time. This can be easily done shrinking or expanding the sample interval near the end of each period to make up for the mismatch. Implementation of this scheme requires that a mechanism is available that tells when the actual period ends and thereby allows the periodic manipulation of input to fit precisely with the real period.

The above mentioned approach is applied on an SMB plant of switching time 7.95 minute whereas RMPC has a period of eight minutes. The controller calculates the input moves for eight minutes based on the model of eight-minute SMB, but implements these input moves only for 7.95 minutes by truncating the last one minute move to 0.95 minute. The process measurements are taken at one minute intervals for the first seven minutes and then at 7.95 minute (end of run). Based on these eight feedback measurements, the controller re-computes the input to be applied at the beginning of next period i.e. every 7.95 minutes.
Due to some inherent physical limitations, zero offset at steady state is not achieved here, but the closed loop is stable despite the small period mismatch.[Figure 6.5] However, as the period mismatch grows bigger, the mismatch between the plant and the model widens as the controller model is kept unchanged while the plant model changes with period.

This approach is viable only for the small period mismatch. As the period mismatch increases the model/plant mismatch may increase to a point that the scheme no longer works. For example, when the same technique is used to control an SMB of switching time of 8.5 minutes, with an RMPC of eight minute, the system becomes unstable.

![Figure 6.5 Control of SMB using heuristic approach in presence of period mismatch](image)

This improvised technique is not guaranteed to work for large mismatches, but it is stable for small period mismatches. Also, this approach is suboptimal and one needs to consider the trade-off between the effort/cost involved in redesigning the controller and the performance gain.
CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Summary of Work

Periodic systems are those in which either operating trajectories are periodic or disturbances are periodic in nature. RMPC is an MPC-based technique that is able to handle systems with periodic behavior. It exploits the periodic nature of process while keeping all the important features of a model based controller. However, as a limitation, RMPC is able to make significant run-to-run improvement only if the period of the controller matches the period of disturbance exactly. This assumption of no period mismatch is limiting in practice as it cannot always be met. For some processes, this period mismatch is inevitable due to limitations imposed by frequency estimation accuracy and sampling intervals that can be implemented.

One might think at first that a slight mismatch would not degrade the performance by much. This is not the case, however; the rate of performance loss is very sharp with respect to the mismatch. This happens because RMPC embeds the concept of repetitive controller, which generates a comb filter shaped sensitivity function to reject the periodic disturbance and its harmonics. The notches in the comb filter are extremely narrow and hence a small mismatch can reduce the controller gain sharply. For these reasons, the standard RMPC is highly sensitive to period mismatches.

The main objective of this research was to develop a Robust RMPC which is able to reject periodic disturbances or track periodic trajectories even in presence of a slight period mismatch. The developed Robust RMPC approach is based on the idea of using multiple memory loops in Robust Repetitive Control to reduce the effects of a period mismatch. It is shown that in the RMPC framework this robustness can be achieved by replacing $\Delta u$ term by its higher order difference (e.g. $\Delta^3 u$) in the control objective
function and reformulating the process description accordingly. The new structure is more robust to a period uncertainty by broadening the comb-like gain structure of the standard RMPC. With the new Robust RMPC, the performance remains acceptable up to a certain degree of period mismatch. The degree of robustness can be increased by incorporating more memory loops into the development of Robust RMPC. However, this increases the state size and would add to the computational load. The advantages of Robust RMPC are shown on a mechanical motion control device for rejecting periodic disturbances of an uncertain period. These results show the improvement achieved with robust RMPC over the standard RMPC in presence of a period mismatch.

Another objective of this research was to mitigate the effect of a period mismatch in systems that are periodic not because of periodic trajectories/references but because of their intrinsic periodic operation trajectories. These periodic operations are forced in some applications to improve the process performance.

In the development of RMPC and R-RMPC, no distinction was made between processes with internal periodic operation [Class I] and processes with exogenous periodic signals [Class II]. Class II processes arise due to periodic references/disturbances but the system itself is time invariant. On the other hand, systems in Class I are periodic due to the periodic mode of operation and are periodically time varying. With the mismatch between the periods of the controller and the process, these two classes behave very differently in terms of stability. In Class II systems, period mismatch diminishes the rejection properties of RMPC, but this can be improved upon by the use of Robust RMPC. On the other hand, in Class I systems with a mismatch present, the application of standard RMPC as well as robust RMPC produces instability.

The suggested explanation for this is that the PTV plant and model get increasingly out-of-phase leading to of a very large degree plant/model mismatch and this in-phase to out-of-phase behavior cycles. For a prototypical switching systems, representative of Class I systems, the closed loop goes unstable even with a slightest mismatch.
A simple but effective approach for these types of system is tried in which the input moves as given by the controller are truncated or lengthened to match with the exact period. This approach is stable for small period-mismatch but the performance is not optimal (achieved at the expense of the final error level reached). Further, the stability depends on the degree of period mismatch and process nature itself, and cannot be guaranteed. So, it is recommended that one redesign the controller for large mismatch cases.

7.2 Suggestions for Future Work

7.2.1 Analyzing the Effect on Non-harmonic Frequencies

It would be interesting to see of the extent to which the good performance achieved at the chosen frequency must be paid for by an amplification of errors at other frequencies. One fundamental limitation to the performance of feedback control systems is if feedback significantly attenuates the effects of disturbances in one frequency range, then it must amplify disturbances in other frequency ranges. RC often aims to eliminate all periodic disturbances of a fundamental frequency and all the harmonics. This suggests that one must pay for this elimination by amplifying the error that occurs between these evenly spaced frequencies. Also, if only one or two harmonics are contributing to the output response, there is an advantage in not using the full RC. The gain increase in the unimportant harmonics can be used to balance the trade-off in favor of other performance requirements. These questions can be answered by developing a control approach that is able to cancel errors at any selected frequencies, instead of a particular one and all its harmonics.

7.2.2 Reducing Inter-sampling Ripples

In digital RC, the tracking property can only be assured at the sampled instants, and good inter-sample behaviors are generally not guaranteed. In general, we can imagine
that precise tracking at sampled instances is accomplished at the expense of inter-sample behavior, because it is generally not taken into account in the digital design.

7.2.3 Selecting Sampling Frequency

The performance of real time RMPC/R-RMPC depends on the number of samples taken within a period. Frequent sampling improves the performance, but at the same time it increases the state size and thus requires more computation effort. One key question can be to determine how often sampling should be done.

7.2.4 Using Interpolation Schemes for Period Mismatch

For Class I systems such as SMBs, it is shown that the RMPC approach does not work even with the slightest mismatch in the period. Fortunately, in this class of systems periodic operation is often forced (regulated), and thus one has the complete control over the selection of the period and can also implement it accurately. However, redesigning the controller whenever the period has to change slightly may prove cumbersome. The approach of adjusting the final input interval is useful for a case of small period-mismatch but may not work well with large changes. It would be interesting to find some interpolation based scheme that can use information available from previous trials to handle period changes.
REFERENCES


