Table of Contents

List of Figures

CHAPTER 1

1 Introduction

1.1 Motivation

Controlling nonlinear and constrained systems has always been more engrossing area for control designers rather than controlling linear systems. This is because every process in the universe or every practical system has some nonlinearities and complexities within it and control techniques for linear systems are unable to sufficiently quench the thirst of control designers to work for practical systems. Within the framework of control designs, *Model Predictive Control* (*MPC*) scheme is one of the most popularly known and widely used. It has revolutionized the process industries because of its abilities to control processes with constraints, nonlinearities and significant dead time or non-minimum phase behavior. However MPC requires an explicit model of the plant to do its marvelous control efforts. This restricts the application of MPC to nonlinear and complex systems or with partially known processes where deriving a mathematical model is often not feasible. On the other hand neural networks and *fuzzy systems* have proved to be effective in the modeling of complex and nonlinear processes and have been acknowledged as universal approximators. This has led to the blending of the schemes, Fuzzy Modeling and *Predictive Control*, for controlling constrained and nonlinear systems. Being inspired by the growing research in this domain, the author is motivated to adopt Fuzzy Model based Predictive Control for controlling a class of nonlinear system i.e. Hammerstein model subjected to actuator constraints on plant input.

1.2 Hammerstein Models

Some industrial systems and processes such as ph-neutralization process, electromechanical system, heat exchanger and distillation column confront such nonlinear effects which can be efficiently modeled as a series connection of a *memory-less nonlinear static element* and a *linear dynamic* part. Since the nonlinearities in these systems are static, these nonlinearities can be significantly reduced from control problems which allow the use of uncomplicated linear algorithms rather than computationally intensive and complex nonlinear programming. This category of nonlinear models, which is composed of the interconnection of *Linear Time Invariant* (LTI) systems and *static nonlinearities* are well known as block oriented nonlinear models [22].

Three main block oriented model structures can be defined on the basis of the order of static and dynamic blocks [21]. These primary model structures are:

- Hammerstein model
- Wiener model
- Feedback block oriented model

The thesis work is based on Hammerstein model which is known as a special case of the *Nonlinear Additive AutoRegressive models with eXogenous* inputs (NAARX). The Hammerstein model can be obtained as a series combination of a memoryless static nonlinearity and linear dynamics as shown in Fig 1.1.

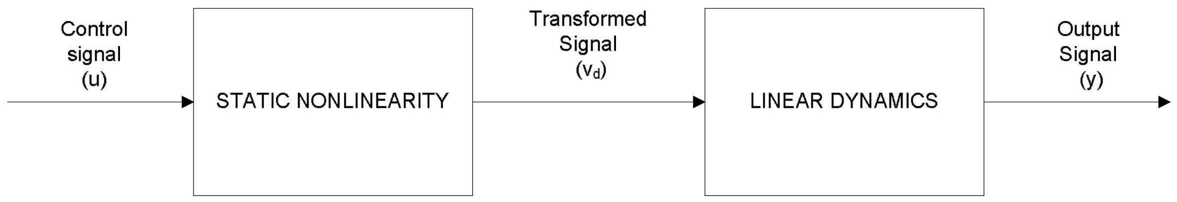

Fig 1.1: Structure of Hammerstein model

In the case of Hammerstein models, all the delayed control inputs have same static nonlinearity '*f*'.

When an input signal *u* is applied on the Hammerstein model, the *memoryless static nonlinearity* first transforms the input signal into a transformed signal v_d which is applied to the linear dynamics of the model. The output of Hammerstein model is therefore given as:

$$y(k) = \sum_{i=1}^{n_u} a_i y(k-i) + \sum_{j=1}^{n_b} b_j f(u(k-j)) + e(k) \tag{1.1}$$

For Thesis work, Fuzzy Hammerstein model is developed to get the most of Fuzzy modeling techniques. Using the 0th order TS fuzzy model, the membership grades of static nonlinearity *f* are determined whiles the parameters of linear dynamics a_i and b_j are determined using constrained recursive least squares approximation. The transformed variable $d_i$is determined using both, the estimated parameters of linear dynamics and static nonlinearity, through some assumptions and analysis which will be described later on.

1.3 Control Scheme

The proposed control strategy is to use en efficient algorithm of *Model Predictive control* which is also known as *Receding Horizon control* and implementing an inverse controller for the static nonlinearity so that an approximate linear dynamics can be obtained. First we will discuss the inverse model control then the Model predictive control will be described.

1.3.1 Inverse Model Control

Normally Fuzzy models describe the behavior of a system in forward reasoning, from cause to effect. By inverting the fuzzy model and obtaining the backward direction of reasoning from effect to cause, an inverse model is obtained. The inverse model gives the correct value for the input for the desired output of that system as depicted in Fig 1.2[21].

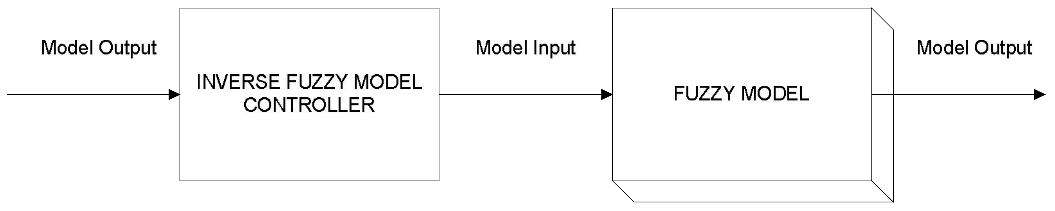

Fig 1.2: Fuzzy model inversion

Let $y= f(\mathrm{x}) = f(\mathrm{x}^*, x_i)$ denote the fuzzy model, where the input variable is split up into a vector and a scalar variable $\mathrm{x}=[\mathrm{x}^*, x_i]$. The aim of the inversion is the determination of the function: $x_i = f(\mathrm{x}^*, y)$, where y denotes the desired output of the system [21].

1.3.2 Model Predictive Control

The concept of Predictive control was first introduced by Richal, et al. (1978) and Cutler and Ramker (1980), describing Dynamic Matrix control and the Model Algorithmic Control methods, respectively. Since then a large number of publications have been written on the subject, where Clarke, at el. (1987) introducing Generalized Predictive Control and Soeterboek (1992) defining Unified Predictive Control are two of the most relevance one [43].

The phrase MPC is derived from the proposal of utilizing a precise model of the system under control. This model is used to predict the future output behavior of the plant. This capability of prediction allows resolving of optimal control problems on line, where *tracking error* which is the difference between the desired reference and the predicted output is reduced over a future horizon. The error minimization may be subjected to constraints on the manipulated inputs and outputs. The basic structure of MPC is shown in Fig 1.3.

The optimization in MPC is carried out using Receding horizon scheme in which there is only the first input of the optimal command sequence at sample instant ‘k’ which is applied to the plant. The remaining optimal inputs of the sequence are rejected. A new but similar optimal control problem is solved at sample instant k + 1. The basic principle of MPC is depicted in Fig 1.4 and Fig 1.5 where H_c represents the control horizon, H_p represents the maximum prediction horizon and k is the sample instant.

Since new measurements are obtained from the plant at each sample instant k, the receding horizon scheme enables the controller to exhibit desired feedback characteristics.

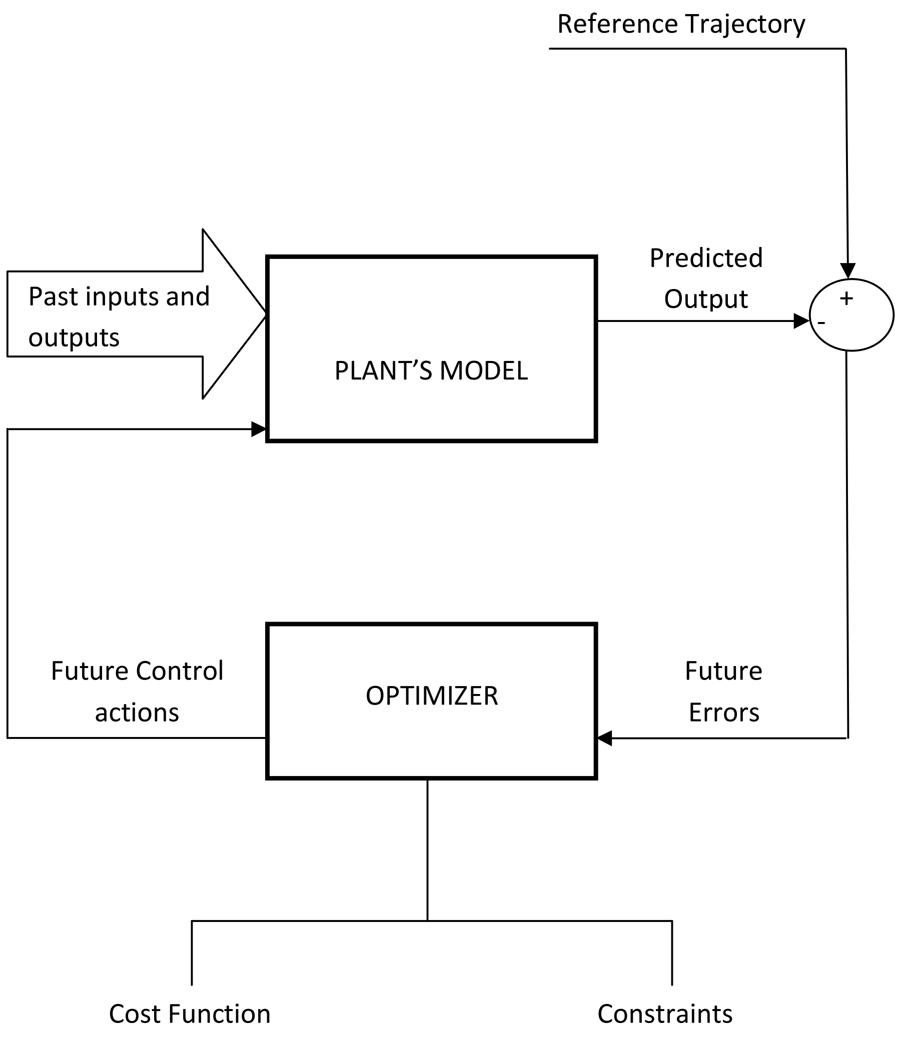

Fig 1.3: Basic structure of MPC

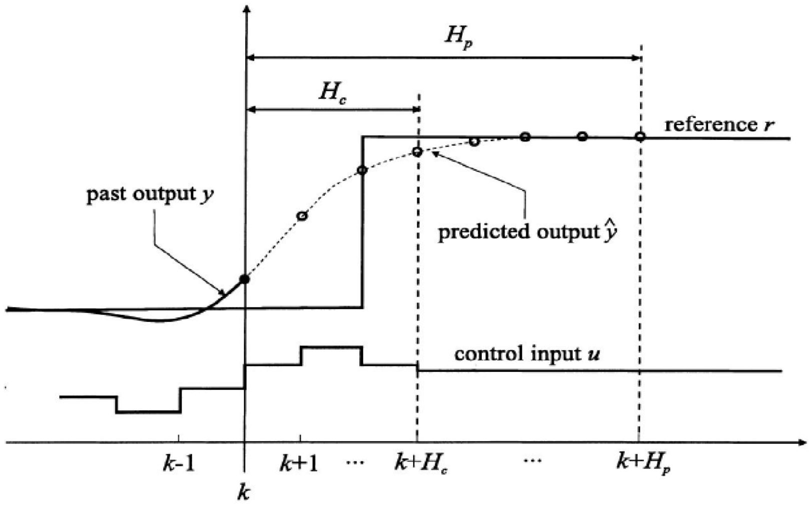

Fig 1.4: Basic principle of MPC

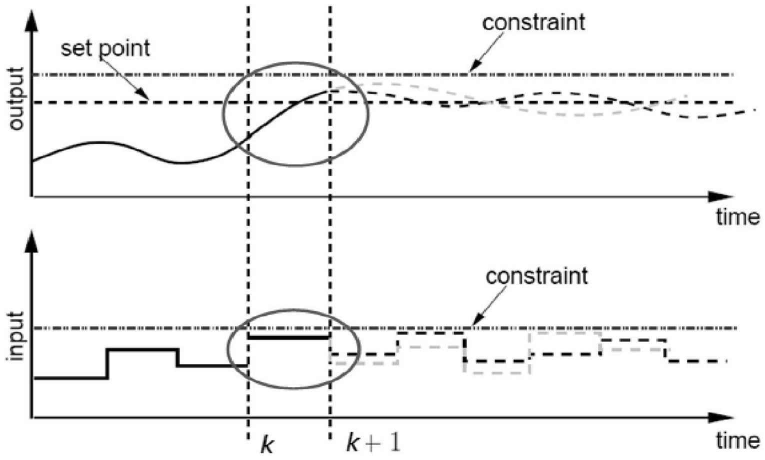

Fig 1.5: Inputs and outputs in receding horizon philosophy

1.3.2.1 Basic Terminologies related to MPC

➢ *Control Horizon*

It is the time span over which the sequence of control actions is determined. After the lapse of this time span, control action becomes constant.

➢ *Prediction Horizon*

It is the period of time which states how far the future output is desired to be predicted.

➢ *Receding horizon strategy*

It is a constantly moving away horizon such that, if for example, the control law optimizes predicted performance at sample *k* over the time span

$$k+1 \leq t \leq k+H_p \qquad \textit{where: } H_p \textit{ is prediction horizon}$$

then at sample *k*+1, the performance is optimized by the control law over the time span

$$k+2 \leq t \leq k+H_p+1 \qquad \textit{where: } H_p \textit{ is prediction horizon}$$

It is therefore apparent that the time span over which the optimization occurs is always receding. At the current sample, those points are taken into account; that were formerly beyond the time span [41].

➢ *Plant model*

The model of original plant that describes system dynamics is a paramount in predictive control. An accurate dynamic model will give consistent and precise predictions of the future [48].

➢ *Cost Function*

For taking the best decision, a criterion or standard is required to replicate the objective. The objective in general relates to an error function which is based on the difference between the desired and the actual responses. This objective function is known as the *cost function J*, and the optimal control action is determined by minimizing this cost function within the *optimization window* [48].

1.3.2.2 Main Components of MPC

There are three main components of MPC:

➢ *Explicit plant model*

The modeling mechanism in MPC design is one of the most crucial activities. The quality of the resulting controller is directly proportional to the model quality and thus the model should translate the system as precisely as possible.

- *Control problem and MPC framework*

This is a significant component of MPC which usually asks for practical experiences. There are certain times when it is tricky even to identify what should be controlled and optimized. All the basic properties and limitations of MPC are required to be known at this stage.

- *Optimization problem*

The last step is the rendition of MPC control problem to a numerical optimization problem. As described earlier, optimization in MPC is carried out using the concept of receding horizon control. Usually this is comparatively straightforward part.

Because of its optimization approach and the explicit use of a process model, MPC is able to provide a systematic approach to control the multivariable dynamical systems with constraints.

1.3.2.3 MPC versus Classical Control

In comparison to popular classical control, Predictive control is proved to be more advantageous because:

- In classical control there is no information of constraints while in predictive control, constraints are the part of design.
- In classical control set point is away from the constraints while in predictive control, set point is closer to the optimal constrained solution.
- Classical control usually generates suboptimal plant operation while predictive control offers improved plant operation.

1.3.2.4 Famous MPC algorithms

There are three broad approaches of predictive control design. Each approach utilizes a unique model structure. Formerly, Finite Impulse Response (FIR) models and step response models were preferred. FIR model or the step response model based design algorithms include Dynamic Matrix Control (DMC) (Cutler and Ramaker, 1979) and the quadratic DMC formulation of Garcia and Morshedi (1986). The FIR type models are attractive to process engineers because this model structure gives a translucent description of process time delay, response time and gain. However, these models are limited to stable plants only and often require large model orders. Depending on the choice of sampling intervals and the process dynamics, this model structure typically involves 30 to 60 impulse response coefficients.

On the other hand, Transfer function models give a more prudent description of process dynamics and can be used for both stable and unstable plants. The well-known transfer function models include those of the predictive control algorithm of Peterka (Peterka, 1984) and the Generalized Predictive Control (GPC) algorithm of Clarke and colleagues (Clarke *et al.*, 1987). One drawback of transfer function model-based predictive control is that it is often found to be less effective in handling multivariable plants.

A state-space representation of GPC was then presented in Ordys and Clarke (1993). Predictive control is now growing popularly using the state-space design methods (Ricker, 1991, Rawlings and Muske, 1993, Rawlings, 2000, Maciejowski, 2002).

Other famous MPC algorithms [42]which are being widely accepted in industries are:

- MPHC – Model based Predictive Heuristic Control which is derived from the impulse response representation of the controlled plant (Richal *et al.*, 1978),
- MAC – Model Algorithmic Control employing the plant impulse response (Rouhani and Mehra, 1982),
- GDMC – Generalized Dynamic Matrix Control which uses the step response model of plant (Xi, 1989),
- EHAC – Extended Horizon Adaptive Control which applies the variable forgetting factor (VFF) method for the identification of ARMA model of the plant (Ydstie, 1984; Ydstie *et al.*, 1985),
- EPSAC – Extended Prediction Self-Adaptive Control which is built on parallel CARMA models (de Keyser and van Cauwenberghe, 1981).

1.3.2.5 Generalized Predictive Control

The thesis makes use of Generalized Predictive Control (GPC) algorithm. It is also commonly known as *Long range predictive controller.* Clarke, Mohtadi and Toffs in 1987 proposed this algorithm for the first time. Its powerful features lie in the long range prediction of the future outputs. When the future control $u(k \pm 1)$ action is asserted at each sample time k, all the future output responses are also determined along some horizon. Because of its long-range prediction, GPC can be utilized to control complicated systems with over parameterized model or with dead-time variation.

1.3.2.6 Actuator constraints

The real world applications of feedback control are engaged with the control actuators having amplitude and rate limitations. Particular physical electromechanical devices can provide only a limited torque, force, flow capacity, stroke or linear/ angular rate. The control designs that pay no attention to these actuator limits may cause undesirable transient response, degraded closed loop performance and may even cause instability in the closed loop. For instance advanced tactical fighter aircraft with high maneuverability requirements may be subjected to pilot-induced oscillations resulting in worse flight performance or even disastrous failure due to actuator amplitude and rate saturation in the control mechanism [16]. Thus a fundamental limitation of many linear and nonlinear control design techniques is constituted by actuator saturation and has attracted the attention of many researchers.

1.4 Scope of the Work

The aspiration for this work is to study and understand the emerging approach of Fuzzy model based Predictive control for a class of nonlinear system subjected to actuator constraints. For simulation purposes, the Hammerstein model having logarithmic static nonlinearity and delayed dynamics is used and in order to incorporate system constraints, actuator saturation and backlash are introduced in the system to define a constrained nonlinear system. For controlling this system using some Model predictive algorithm, it is essential to develop the model of the plant first because MPC loses all its benefits if it is not provided with plant model. It is assumed here that the plant is unknown and input output data of the plant is made available online. Black box modeling approach is therefore used for identification. A 0th order TS Fuzzy model is used for the memoryless static nonlinearity from which membership grades for the input will be determined. Then constrained Recursive least squares approximation will be used to identify the parameters online. The approach is to first identify the parameters locally then these are subjected to linear constraints, which when fulfill iteratively, identify the parameters of complete Fuzzy Hammerstein model. Once the model is developed and system parameters are in hand, one of the most widely used algorithm of MPC, Generalized Predictive Control will be implemented using the identified model. The Hammerstein system [21] is incorporated in the MPC scheme using an adaptive inverse model controller of the static nonlinearity which when combined with the Hammerstein system produces an approximate linear dynamics by cancelling the static nonlinearity of Hammerstein system. Finally for the optimization, Mixed Integer Quadratic Programming (MIQP) is used which generates an optimized control action for the plant subjected to actuator constraints.

Using the approach of incorporating inverse controller to deal with system nonlinearity, GPC is left responsible to control only the approximate Linear Dynamics rather than the linear dynamics connected with the static nonlinearity moreover the removal of static nonlinearity results in convex optimization which makes the overall algorithm more attracting.

1.5 Thesis Organization

The thesis is organized as mentioned below:

Chapter 2 details the literature review of the cultivation in the domain of Fuzzy Model based Predictive control for Hammerstein model with actuator constraints.

Chapter 3 discusses the identification of Hammerstein model briefly using Fuzzy 0th order TS model and Recursive least squares estimation.

Chapter 4 describes the complete control architecture for controlling the Hammerstein model with constraints handling.

Chapter 5 delineates the overall performance in detail by implementing the control architecture on fuzzy Hammerstein model. The simulation results of model identification and controller will be presented and discussed.

Chapter 6 finally concludes the thesis work by outlining the origin of the proposed scheme, the novelty introduced in the basic idea, its benefits and the recommendations.

CHAPTER 2

2 Literature Review

2.1 Prologue

The chapter presents a literature review of the studies carried out for thesis work. For the sake of compactness, all of the corresponding material is not brought into the scope of the chapter however some key references will be discussed briefly. A detail list of the reference material is provided in reference section for further study.

2.2 Brief Review of the Research Work

Nowadays, the requirements for eminence of automatic control in the process industries have augmented significantly because of the enhanced complications in plants and acuter specifications of product quality. Simultaneously, the available computing power has elevated a lot. Resultantly the computers, which are computationally expensive, became germane even to rather complex problems. Model based and intelligent control techniques have developed to attain better control for such applications. Recently the amalgamation of fuzzy models as intelligent control techniques with adaptive control system design has been asserted as a novel scheme for the control of systems with significant dead time, nonlinearities and constraints. Control designers have accepted the scheme as an interesting research area and a significant number of papers have been published where the authors make use of different benefits which are extracted from different variations in the scheme.

Fuzzy models are suitable where the goals and constraints or the physical mechanisms are ambiguous. The predictive control based on the model is a common strategy for solving the control problem in time domain and is composed of three key concepts [3] of explicit use of model, computation of a sequence of future control signals and the receding horizon strategy.

The scheme can be applied for the optimal control of linear, nonlinear and multivariable processes, with non-minimal phase or with a considerable dead time in the presence of constraints [3].

J´erˆome Mendes, RuiAra´ujo, and Francisco Souza in '*Adaptive Fuzzy Generalized Predictive Control based on discrete T-S fuzzy model*' present an adaptive fuzzy predictive control based on discrete-time Takagi-Sugeno (T-S) fuzzy model. The proposed controller is based on generalized predictive control (GPC) algorithm, and a discrete time TS fuzzy model is employed to approximate the unknown non-linear process. To provide a better accuracy in identification of unknown parameters of the model, it is proposed an on-line adaptive law which ensures that the tracking error remains bounded. The stability of closed-loop control system is studied via the Lyapunov stability theory [25].

Aldo Cipriano and *Doris Saez* in '*Fuzzy Generalized Predictive control and its application to an inverted pendulum*' detail the design framework and assessment of a predictive control algorithm which depends on a fuzzy model applied to the angular stabilization of an inverted pendulum. They have favorably compared the designed algorithm with a conventional generalized predictive control [2].

In the research paper, '*Study of Fuzzy Generalized Predictive Control Algorithm on Nonlinear systems*' by *Qiang Li, BaocunQu ,ZhiqiangGe and Xisheng Zhan*, T-S model is utilized to describe a non-linear system with the identified fuzzy rules to linearize the model locally and dynamically. Then generalized predictive controller is incorporated to control nonlinear object defined by the developed fuzzy model. Simulation results are presented to claim that the algorithm has good robustness, an appreciable accuracy in identification, and nice control performance therefore this novel scheme can be used for a wide range of non-linear systems [37].

Salima Djebrani and *Foudil Abdessemed* in '*Fuzzy model predictive control for nonlinear systems*'[11] put forward the application of GPC algorithm to the class of nonlinear systems by using fuzzy dynamic models. In their work they have applied the predictive control to the obtained fuzzy model. A typical genetic algorithm is then combined with GPC algorithm for the online optimization of design parameters.

The paper '*Identification and control of nonlinear systems using Fuzzy Hammerstein model*' by *Janos Abonyi et al* discusses the identification of Fuzzy Hammerstein model using three iterative techniques. To generate an optimal control signal then, Generalized Predictive control algorithm of MPC is used for approximate linear dynamics. The approximate linear dynamics is obtained by combining the Hammerstein model with an inverse fuzzy model of memoryless static nonlinearity. The scheme is then tested on electric water heater and fine control performance is obtained [22].

Regarding the system constraints, actuator saturation is amongst the most common and noteworthy nonlinearities in a control system. In the research papers several examples are presented where neglecting the amplitude and rate saturation has led to difficulties and imperiled the overall stability of the system. One application area is flight control of high agility aircrafts. Dornheim (1992) and Murray (1999) have referred to an YF-22 aircraft crash in April 1992. The incident was due to the pilot-induced oscillations caused by amplitude rate saturation of control surfaces. This ultimately induced the effects of time-delay in the control loop. A similar example is a Gripen JAS 39 aircraft crash in August 1993 (Murray, 1999). In this case too, it was saturations performed a dangerous role [5].

H. Zabiri and Y. Samyudia in *'MPC design for constrained multivariable systems under actuator backlash'* have extended the MPC design methodology for addressing simultaneously the actuator backlash and saturation [17]. They have presented that for getting rid of actuator constraints of saturation and backlash, MPC design could be formulated as Mixed Integer Quadratic Programming problem by developing a set of logical constraints. They have presented simulation studies using the FCCU process to compare their approach with other techniques i.e. the nonlinear inverse and re-tuning strategies. The results show that MIQP-based MPC design has surpassed other compensation methods in the presence of both actuator saturation and backlash [17].

Piotr M. Marusak in his papers *'On prediction generation in efficient MPC algorithm based on Fuzzy Hammerstein model'* and '*Numerically efficient analytical MPC algorithm based on fuzzy Hammerstein model'*[29] has compared the performance of Nonlinear Model Predictive control (NMPC), Linear Model Predictive Control (LMPC) and Fuzzy Model based Predictive Control (FMPC) algorithms for the distillation column which is a significantly delayed nonlinear plant. Dynamic Matrix Control Algorithm of MPC is used in his papers. This algorithm is based on reliable quadratic programming routine [29].The three controllers are tested by varying their set-points. He has proved with the experiments that the responses obtained in the control system for different set-points with FMPC algorithm are very close to those obtained with the NMPC algorithm with full nonlinear optimization. The standard LMPC algorithm however operates almost as good as its counterparts for a certain range of these set–points but it works unacceptably worse after a certain range because of significant nonlinearity in the control plant.

The review on the literature reveals that Fuzzy model based Predictive control has truly and effectively taken the place of complex algorithms like Nonlinear Model Predictive control (NMPC) because of its abilities to control nonlinear and constraints systems efficiently. Although NMPC is also a powerful control technique but there are researches which show that Fuzzy model based predictive control has comparable performance with much reduced complexity. This is the reason why the control researchers' interest is enhancing in this field.

2.3 Chapter Summary

In this chapter some of the work in the field of fuzzy model based Predictive control and the utilization of MPC for handling actuator constraints is briefly presented. It is evident from this literature that the researches are keen to control the nonlinear, complex and constrained processes with predictive control however in order to deal with the limitation of MPC and all its corresponding algorithms like GPC, that these algorithms require an explicit model of the plant, researchers have developed fuzzy models using different techniques. When the limitations of MPC are fulfilled by amalgamating it with fuzzy models, the amalgamated FMPC gives performance comparable to NMPC.

<u>CHAPTER 3</u>

3 Identification of Fuzzy Hammerstein Model

3.1 Prologue

The chapter begins with a brief discussion on modeling approaches and then jumps to the TS fuzzy modeling. Linking the gap between TS fuzzy model and Fuzzy Hammerstein model, it describes the structure of Fuzzy Hammerstein models and discusses the identification scheme using constrained recursive least squares in detail. Utilization of different approaches for identification and modeling is presented with concrete reasons to give a thorough understanding.

3.2 Modeling Approaches

In this section we will briefly discuss the available modeling approaches. There are four famous approaches to model a system. These are:

➢ *White-box: also known as mechanistic or physical modeling*

White box modeling approach utilizes laws such as physical or chemical to derive the mathematical model of the processes. The laws may be defined by algebraic, differential or difference equations. The model developed using this approach is precise and exact in its description but the modeling methodology is monotonous and deadly time-consuming so these models are only suitable when sufficient amount of information is available about the process and the designer has enough time to work on it.

- *Black-box : also known as data-driven or inductive modeling*

Black box modeling is used when there is no information available about the system under study. Therefore unlike white box modeling there is no need of detailed bunch of knowledge rather the system is identified using the input output data of the process.

- *Gray-box : also known as semi-mechanistic or hybrid modeling*

In gray box modeling approach, it is supposed that the steady-state behavior of the system is available as *a priori* knowledge. This approach is not suitable when the process is complex in its physical description because in this case, computations are intensive and first principle steady state models can rarely be obtained.

- *Fuzzy: also known as knowledge-based or qualitative modeling*

Perhaps the most popular methodology nowadays is fuzzy modeling in which system model is defined by the qualitative knowledge and using linguistic rules described by the system expert. To some extent there is the involvement of physics related to the system because the expert defines the rules accordingly, but it is such an adolescent use that does not leave the marks of tediousness on the modeling approach. Similarly it is close to black box modeling because of the utilization of input output data of the process for identification.

3.3 TS Fuzzy Model

Fuzzy models are non-linear in their structures. These models are suitable for articulating non-linear dynamics of the systems along with uncertainties. Two famous fuzzy model structures are:

- Mamdani Fuzzy models
- T-S fuzzy models

Out of these two structures, TS fuzzy models have been widely recognized as *Universal Approximators* for nonlinear systems. The model was put forward by Takagi, Sugeno and Kang [46], [28] for producing fuzzy rules from available input output. In TS fuzzy model, mathematical model is blended with logical model such that the logical rules define fuzzy antecedents while the mathematical functions define the consequent part of fuzzy model. These antecedents of fuzzy rules split up the input space into several fuzzy regions, while the consequent functions define the behavior of system within a given region:

$$\begin{aligned} R_j : \textbf{If } m_1 \textbf{ is } A_{1,j} \textbf{ and ..... and } m_n \textbf{ is } A_{n,j} \\ \textbf{then } y = f(q_1 ....q_y) \end{aligned} \tag{3.1}$$

Where:

- $m=[m_1,....,m_n]^T$: n-dimensional vector for the antecedents of fuzzy rules
- $q=[q_1,....,q_y]^T$: y-dimensional vector for the consequents of fuzzy rules
- mϵ x and q ϵ x where x is the combination of the inputs related to the model $y=f(x)$.
- $A_{i,j}(z_i)$: the antecedent fuzzy set for the i^{th} input. These antecedent fuzzy sets defined by the fuzzy rules divide the input space into different fuzzy regions
- $f_j(q)$: consequent function defining the behavior of system within a specified region.

Generally the f_j consequent function is a polynomial of the input variables, yet it may be any randomly selected function that can suitably define system output within the domain specified by the antecedents of fuzzy rule. When $f_j(q)$ is given by a first order polynomial then:

$$\begin{aligned} f_j(\mathrm{q}) &= p_j^0 + p_j^1 q_1 + ... + p_j^m q_m \\ &= \sum_{l=0}^{m} p_j^l q_l \qquad \text{where} \quad q_0 = 1 \end{aligned} \tag{3.2}$$

The fuzzy inference system thus produced is well known as the first order Takagi-Sugeno model. If however f_j represents only a constant i.e. m=0 then the fuzzy inference system is called zero-order Takagi-Sugeno or the singleton fuzzy model.

By using the fuzzy inference derived from product-sum-gravity at a certain input, the overall output of the fuzzy model, *y*, is deduced by determining the weighted average of the consequent functions as shown in (3.3).

$$y = \frac{\sum_{j=1}^{N_r} \beta_j(\mathbf{m}) f_j(\mathbf{q})}{\sum_{j=1}^{N_r} \beta_j(\mathbf{m})} \tag{3.3}$$

Where:

- β_j: weights of membership function .

The range of the weights of membership function lies between 0 and 1. It is mathematically represented as:

$$\beta_j(m) = \prod_{i=1}^{n} A_{i,j}(m_i) \tag{3.4}$$

The antecedent membership functions are usually assumed to be triangular and are given as:

$$\begin{aligned} A_{l,i_1}(m_l) &= \frac{m_l - a_{l,i_1-1}}{a_{l,i_1} - a_{l,i_1+1}}, \qquad & a_{l,i_1-1} \le m_l < a_{l,i_1} \\ A_{l,i_1}(m_l) &= \frac{a_{l,i_1+1} - m_l}{a_{l,i_1+1} - a_{l,i_1}}, \qquad & a_{l,i_1} \le m_l < a_{l,i_1+1} \end{aligned} \tag{3.5}$$

Where:

- $a_{l,il}$: Cores of the fuzzy set. They also indicate the support and core of fuzzy set because:

$$\sup_{l,i_l} = a_{l,i_1+1} - a_{l,i_1-1}$$
$$a_{l,i_1} = core(A_{l,i_1}(m_l)) = \{m_l \mid A_{l,i_1}(m_l) = 1\}$$

3.4 Fuzzy Hammerstein Model (FHM)

A fuzzy model is used to depict the steady state behavior of nonlinear dynamical systems [21]. This model is translucent and elucidated because it is identified using the linguistic rules and input output data obtained from the process.

Here we will discuss the Fuzzy Hammerstein model which is a straightforward form of general fuzzy model NARX (Nonlinear AutoRegressive with eXogenousiput) structure given as:

$$\hat{y}(k) = f(y(k-1)),....,y(k-n_a),u(k-n_d),....,u(k-n_b-n_d)) \tag{3.6}$$

Where:

- n_a: maximum lag for output
- n_b: maximum lag for input
- n_d: delay or dead time
- f : fuzzy model mapping

NARX model is commonly utilized for several methods of nonlinear identification [38]. Like other modeling schemes, some drawbacks are also associated with NARX modeling. For instance these models are not suitable for complex, high-order dynamic processes because of their exponentially increasing memory and the requirements of information. Furthermore, these models do not consider the fact that usually the industrial processes are slightly excited around the nominal operation point due to which transient data is not comprised of satisfactory non-linear information about the process nonlinearity. These limitations ask for incorporating the whole body of *a priori* knowledge in the development of dynamic fuzzy models by constraining its parameters. Yet another option is to decrease the impacts of over-parameterization to restrict the structure of NARX model by implementing *Nonlinear Additive AutoRegressive with eXogenous input* (NAARX) models which are obtained by employing model building methods. An exclusive case of NAARX model is the simplified non-linear block-oriented structure of Hammerstein model. It was discussed in chapter 1 that nonlinear effects present in most practical processes can be efficiently modeled using the Hammerstein models where there is the series combination of a nonlinear static element and Linear Time invariant dynamic part.

These models have been recognized as appropriate for grey-box and fuzzy modeling approaches.

3.4.1 Structure of FHM

As discussed earlier, in Hammerstein models memoryless static nonlinearity is connected in series with linear time invariant dynamics. If we parameterize the static nonlinearity separately, $f(.)$ can be framed as a set of functions:

$$v_h = f_h(u_h) \qquad \text{for } h=1,\ldots,H_c.$$

Zero order Takagi Sugeno fuzzy models are used to represent f_h (.) a as a set of rules:

$$R^h_{i_1,\ldots,i_{H_c}} : \textbf{If } u_1 \textit{ is } A_{1,i_1} \textbf{ and,.......,and } u_{H_c} is A_{n,i_{H_c}} \textbf{ then } v_h = d^h_{i_1,\ldots,i_{H_c}} \tag{3.7}$$

Where A_{1,i_1} (u_j) is the i_j^{th} antecedent fuzzy set for the j^{th} input and $d^h_{i_1,\ldots,i_{H_c}}$ is the consequent parameter (a real number).

Because in spite of the fact that the first order TS fuzzy models are more efficient than the zero order models yet there are many issues at the identification and control applications of the first order TS models that are not found in zero order TS models. In the following three such issues will be described briefly:

- *Interpretation of model*

Membership functions possess critical importance for the performance of a first or higher order TS model hence the consequent local models cannot be inferred locally.

- *Control applications based on extracted linear models*

When the controller is developed by the linearization of the fuzzy model, the applications of first order TS fuzzy models become tricky. The issue is that the locally interpreted interpolated model differs from the model attained by linearization and this difference has to be considered cautiously at the controller designing stage.

- *Control applications based on model inversion*

Contrary to zero order TS fuzzy models, the first or higher order TS fuzzy models are not inverted analytically. Moreover, it is difficult to guarantee monotonic behavior of these models required for inversion. This can however be easily obtained for zero order TS models.

Because of these reasons and the simplicity associated with zero order TS models, it was decided to choose this model.

Thus the memoryless static nonlinearity of the Hammerstein model is developed as a fuzzy model. Given an input vector, **u**, the output of the fuzzy model of static nonlinearity, v_h, is computed as the weighted average of rule consequents:

$$v_h = \frac{\sum_{i=1}^{E_1} \cdots \sum_{i_{H_c}=1}^{E_{H_c}} \beta_{i_1,\dots,i_{H_c}}(\mathbf{u}) d^h_{i_1,\dots,i_{H_c}}}{\sum_{i=1}^{E_1} \cdots \sum_{i_{H_c}=1}^{E_{H_c}} \beta_{i_1,\dots,i_{H_c}}(\mathbf{u})} \tag{3.8}$$

Where:

- E_j: the number of fuzzy sets in the j^{th} input domain.

The weight, $0 \le \beta_{i_1,\dots,i_{H_c}}(\mathbf{u}) \le 1$, is the overall truth value of the $(i_1,.........,i_{H_c})^{th}$ rule calculated as:

$$\beta_{i_1,\dots,i_{H_c}}(\mathbf{u}) = \prod_{j=1}^{H_c} A_{j,i_j}(u_j) \tag{3.9}$$

Here triangular membership functions with unity partition are used as antecedents. These membership functions are pictorially represented as shown Fig 3.1 and mathematically given as:

$$
\begin{aligned}
A_{j,i_j}(u_j) &= \frac{u_j - a_{j,i_j-1}}{a_{j,i_j} - a_{j,i_j+1}} , && a_{j,i_j-1} \le u_j \le a_{j,i_j} \\
A_{j,i_j}(u_j) &= \frac{a_{j,i_j+1} - u_j}{a_{j,i_j+1} - a_{j,i_j}} , && a_{j,i_j} \le u_j \le a_{j,i_j+1} \\
A_{j,i_j}(u_j) &= 0 , && \text{otherwise}
\end{aligned}
\tag{3.10}
$$

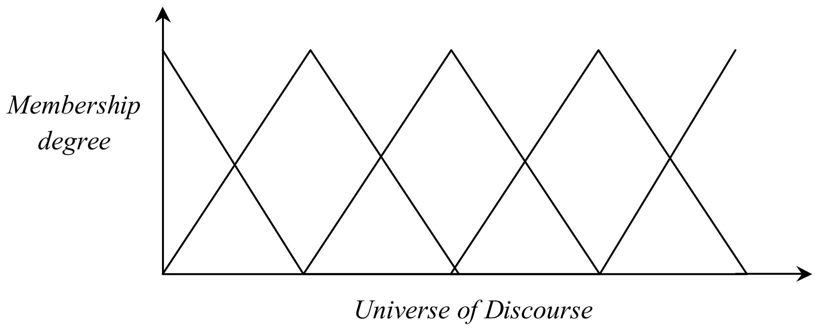

Fig 3.1: Triangular Fuzzy sets

The cores of adjacent fuzzy sets define their support. This mechanism assures that the sum of the membership functions equals one and assists in obtaining elucidated, grid-type partitioning rule-base. The consequent estimation method is, however, independent of the shape and type of membership functions. Because the product operator is utilized for the 'and' connective as shown in (3.9), the overall truth values of the rules satisfy that:

$$\sum_{i=1}^{M_1} \cdots \sum_{i_{H_c}=1}^{M_{H_c}} \beta_{i_1,\dots,i_{H_c}}(\mathbf{u}) = 1 \tag{3.11}$$

This helps in simplifying (3.8) as:

$$v_h = \sum_{i=1}^{M_1} \cdots \sum_{i_{H_c}=1}^{M_{H_c}} \beta_{i_1,\dots,i_{H_c}}(\mathbf{u}) d^h_{i_1,\dots,i_{H_c}} \tag{3.12}$$

After the static nonlinearity in the series connection of Hammerstein model, there is a linear dynamics represented by ARX type model. The combination of NARX type static nonlinearity and ARX type linear dynamics results in NAARX representation of the SISO Hammerstein model given by:

$$\begin{aligned} y(t+1) &= \sum_{i=1}^{n_a} a_i y(t-i+1) + \sum_{i=1}^{n_b} b_i f(u(t-i-n_d+1)) \\ y(t+1) &= \sum_{i=1}^{n_a} a_i y(t-i+1) + \sum_{i=1}^{n_b} b_i v_{t-i-n_d+1} \end{aligned} \tag{3.13}$$

Substituting the value of v_h from (3.12) we get:

$$\begin{aligned} y(t+1) &= \sum_{i=1}^{n_a} a_i y(t-i+1) + \sum_{i=1}^{n_b} b_i \sum_{j=1}^{N_R} \beta_j (u(t-i-n_d+1)) d_j \\ &= \sum_{i=1}^{n_a} a_i y(t-i+1) + \sum_{j=1}^{N_R} \sum_{i=1}^{n_b} b_i d_j \beta_j (u(t-i-n_d+1)) \end{aligned} \tag{3.14}$$

Where n_a ,n_b and n_d are as defined above.

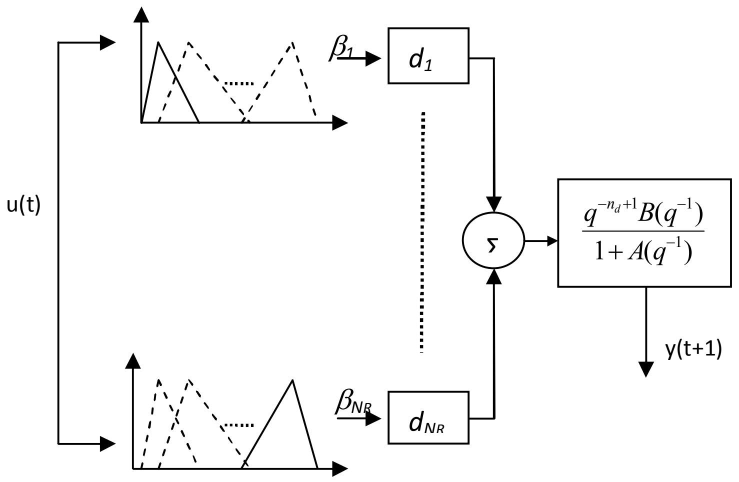

Fig 3.2: Basic structure of Fuzzy Hammerstein model

In continuation of (3.14), parameters a_i and b_i of linear dynamic model will be remarked as “linear parameters” while parameters d_j, associated with the fuzzy model, will be remarked as “nonlinear parameters”. The general structure of the model thus obtained is shown in Fig 3, where q represents the shift operator, *i.e.*, $u(t)q^{-1}=u(t-1)$.

Since the static nonlinearity determines the static gain of Hammerstein model, from (3.14) it is clear that the FH model is over-parameterized. Therefore to cope up this over parameterization, either the gain of linear dynamics or one of its parameters can be chosen freely. The common choices are:

- Select b_1=1or
- Assign unit static gain to the linear model.

If we opt to assign unity gain to the linear dynamics as in (3.15) then the fuzzy model alone exhibits the steady-state behavior, $y_s=f(u_s)$, where u_s and y_s represent the corresponding steady-state input-output data pair.

$$\frac{\sum_{i=1}^{n_b} b_i}{1-\sum_{i=1}^{n_a} a_i} = 1 \tag{3.15}$$

Furthermore if we introduce the parameterization $b_i^j = b_i d_j$, the FH model in (3.14) containing the nonlinear parameter d_j becomes linear in its new parameter b_i^j and the fuzzy model is reduced to be generalized FH (gFH) model

$$y(t+1) = \sum_{i=1}^{n_a} a_i y(t-i+1) + \sum_{j=1}^{N_R} \sum_{i=1}^{n_b} b_i^j \beta_j (u(t-i-n_d+1)) \tag{3.16}$$

Considering the given assumption:

$$b_i^j = b_i d_j \tag{3.17}$$

It is obvious that:

$$\frac{b_i^j}{b_k^j} = \frac{b_i}{b_k}$$

Therefore the following is valid:

$$\frac{b_i^j b_k^l}{b_k^j b_i^l} = 1 \tag{3.18}$$

The validity of this fact in (3.18) assures that the gFH model is identical to the original Fuzzy Hammerstein model. If this condition is not satisfied the gFH model will have different dynamic behavior in each operating region defined by different fuzzy sets.

3.4.2 Identification of FHM

It is always an exigent task to identify a block-oriented model. Due to the conduct and structure of these models, however, the model identification can be abridged provided the non-linear steady-state data, which is independent of the system dynamics, is available [23].

Many identification algorithms have been proposed for the estimation of the parameters of Hammerstein models however two methods are used most commonly:

- Parametric estimation of parameters
- Non-parametric estimation of parameters

In *parametric estimation*, methods like batch least squares, recursive least squares and gradient method are most commonly used while in *nonparametric estimation*, Bayesian regression is usually used which portrays the unknown model map as multidimensional stochastic process which encapsulates the available prior information statistically.

The purpose of the nonparametric method is to relax the assumptions on the form of the nonlinear function and to let the training data choose which characteristic suits them best. Moreover, the nonlinearity is considered as a continuous or a measurable function. If so, the non-linear element is defined by an approximation of a truncated series or an orthogonal function, however, the selection of type and the length of series are not easy.

The least square parameter estimation algorithm was first utilized by Gauss in 1795. It identifies the unknown parameters by employing such techniques in which the squared error between the ideal and actual parameter estimation is minimized by fitting the measurement to the underlying governing equations.

In our work, we will use parametric method of identification where RLS algorithm will be used to identify the parameters of steady-state nonlinearity and simultaneously the linear dynamic of the Fuzzy Hammerstein model. This gives rise to a nonlinear identification problem. To make the algorithm easily interpretable, the identification problem is however converted to a constrained recursive linear least-squares estimation of the parameters of FH model [22].

There are certain reasons which led to the selection of RLS algorithm these are some benefits associated with the scheme which are described below:

Benefits of RLS

- It can be applied in the real time, because it is not necessary to use all input- output data pairs to estimate parameters, and method uses earlier estimated parameters as initial conditions or to specify previous estimates (using non recursive method we need to recalculate parameters from all data) if plant conditions changes.
- The method does not require to have input output data which cover all possible input output set.
- The method works faster because it does not use operations with matrices.
- The method converges faster if forgetting factor techniques is used.
- The method is simple for the implementation.
- The algorithm is able to learn very good policies using only a small number of samples compared to conventional learning approaches, such as Q-learning [48].
- The algorithm requires little or no modification to adapt it to various situations.

The key behind RLS algorithm is to determine the parameter update at time instant *k* by introducing a correction term to the previous parameter estimate once the new information is available. Such formulation reduces the computational load, making the RLS scheme quite attractive in past decades for the applications of on-line parameter estimation.

For *fuzzy Hammerstein* model parameters identification, RLS algorithm seeks for the best estimates of the model parameters a_i, b_i and ultimately d_i, assuming that other parts of the model (the number of fuzzy sets, the centers of membership functions) have been selected appropriately in advance. The parameters of the *fuzzy Hammerstein* model are identified from both the process input-output data for linear dynamics and the linguistic rules for static nonlinearity. Since the recursive least squares is linear so we will now make use of the assumption in (3.18) to linearize the model and submit the model to RLS algorithm for parameters estimation.

Using this method a generalized structure of the FH model is identified which is linear in all its parameters and possesses different dynamic behavior in different operating regions which are defined by the rules of fuzzy model. As discussed in previous section, to approach FH, the parameters of gFH model are to be forecasted onto a set of linear constraints. This forecast assures that the steady state behavior of the resulting FH is similar to the steady state behavior of the previously identified gFH model. It also ensures that the parameters of linear dynamic model are optimal.

3.4.3 Identification based on a Generalized Fuzzy Hammerstein Model (gFH)

After defining the identification scheme, now we will present all the mathematical details related to the identification based on Generalized Fuzzy Hammerstein model. The generalized fuzzy Hammerstein model can be considered as a linear MISO system with N_r inputs $\beta_1(u(k)), \ldots\ldots, \beta_{Nr}(u(k))$as can be seen from Fig 3.3.

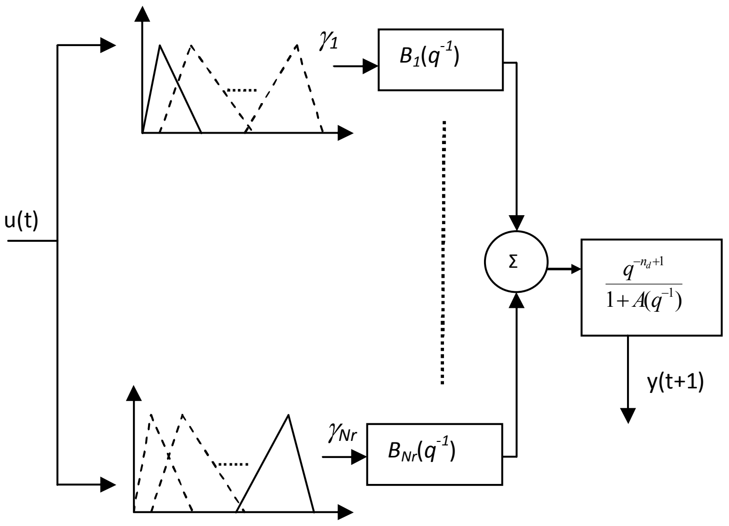

Fig 3.3: Structure of generalized Fuzzy Hammerstein model for SISO system

Therefore the parameters of Fuzzy Hammerstein model, a_i and b_i^j , are found out by using a standard linear recursive least squares algorithm.

Let the regressor vector is defined as:

$$\begin{aligned}\varphi(t-1) = [&y(t-1),....,y(t-n_a),\\ &\beta_1(u(t-n_d-1)),....,\beta_{N_R}(u(t-n_d-1)),.....,\\ &\beta_1(u(t-n_d-n_b)),....,\beta_{N_R}(u(t-n_d-n_b))]^T\end{aligned} \tag{3.19}$$

and the parameter vector is defined as:

$$\theta(t-1) = [a_1,........,a_{n_a},b_1^1,..........,b_{n_b}^{N_R}] \tag{3.20}$$

Then the linear unconstrained recursive least squares (RLS) estimate of $\theta$is:

$$\theta(t) = \theta(t-1) + \frac{P(t-2)\varphi(t-1)[y(t)-\varphi^T(k-1)\theta]}{\lambda+\varphi^T(k-1)P(t-2)\varphi(t-1)} \tag{3.21}$$

$$P(t-1) = \frac{1}{\lambda}[P(t-2) - \frac{P(t-2)\varphi(t-1)\phi^T(t-1)P(t-2)}{\lambda+\varphi^T(t-1)P(t-2)\varphi(t-1)}] \tag{3.22}$$

Where

- λ : Forgetting factor($0.9 \le \lambda \le 1$)
- P : The covariance matrix

The condition of convergence for recursive least squares algorithm are:

- $\emptyset$ is bounded
- The process is persistently excited, that is:

$$\lambda I < \frac{1}{S^\Delta}\sum_{i=j+1}^{j+S^\Delta}\phi(i)\phi^T(i) < \delta I \tag{3.23}$$

This condition has to be considered while designing the identification signal. For this condition, the assumption of persistent excitation should be considered as well as the designed input sequence should produce training data that have sufficient information about the process dynamics in the whole operating range.

To acquire a Hammerstein model, the identified parameters should satisfy the constraints defined in (3.18). But these constraints cannot be fulfilled automatically in the presence of noise in measurement and model plant mismatch. This requires the parameters of gFH model to be constrained. To get rid of this issue, a nice solution is to introduce linear equality constraints defined by (3.24).

$$M\theta^{c} = c \tag{3.24}$$

By this strategy the optimal constrained solution θ^{c} can be determined by the optimal projection of unconstrained parameters obtained by (3.21). These constrained parameters θ^{c} are determined as:

$$\theta^{c} = \theta - PE^{T}[(EPE^{T})^{-1}(E\theta - c)] \tag{3.25}$$

In spite of the simplicity in solution, it cannot be used directly because the constraints defined in (3.18) are nonlinear. In this scenario, the optimal projection problem can be solved as an iterative algorithm which linearizes the nonlinear constraints at each iteration. The modified algorithm can be used to identify the fuzzy Hammerstein models but its convergence is not guaranteed [21]. For assured convergence therefore we need linear constraints in place of the corresponding nonlinear constraints. These linear constraints can be obtained by incorporating following assumptions:

- The parameters of gFH model can be projected to obtain the parameters of FH model
- The linear part of the fuzzy Hammerstein model, obtained after the projection of gFH model, should have unity gain.
- The unconstrained and projected gFH models should be identical in their steady state behavior which is given as:

$$y_{ss} = \frac{1}{1 - \sum_{i=1}^{H_c} a_i} \sum_{j=1}^{N_R} \sum_{i=1}^{n_b} b_i^j \beta_j(u_{ss}) \tag{3.26}$$

Extracting Linear Equality Constraints

The *nonlinear parameter*, *d_j*, of the fuzzy Hammerstein model containing a linear dynamics of unity gain can be obtained by taking the stead state outputs at the cores of the antecedent fuzzy sets of static nonlinearity therefore:

$$d_j = \frac{\sum_{i=1}^{n_b} b_i^j}{1 - \sum_{i=1}^{n_a} a_i} \qquad j = 1, \ldots.., N_R \tag{3.27}$$

From (3.27), we get:

$$d_j (1 - \sum_{i=1}^{n_a} a_i) = \sum_{i=1}^{n_b} b_i^j \qquad j = 1, \ldots.., N_R \tag{3.28}$$

Similarly consider the nonlinearity constraint in (3.18)

$$\frac{b_i^j b_k^l}{b_k^j b_i^l} = 1$$

Substituting the value of (3.17)in (3.18):

$$\frac{b_i^j b_k d_l}{b_k d_j b_i^l} = 1$$

$$b_i^j b_k d_l = b_k d_j b_i^l$$

$$b_i^j b_k d_l - b_k d_j b_i^l = 0$$

Finally

$$b_i^j d_l - d_j b_i^l = 0 \qquad j = 1, \ldots\ldots.., N_R \tag{3.29}$$

Equations(3.28) and (3.29) define the linear equality constraint for the constrained recursive least squares.

3.5 Identification Algorithm

Algorithm 3.1 below summarizes the overall identification procedure:

Step 1: At first gFH model is identified using (3.21)and (3.22).

Step 2: Linear constraints are computed as(3.28) and (3.29); and these are formulated in matrices form as in (3.24) where θ^{c} defines the vector of parameters and E and c are adjusted according to the constraints equations.

Step 3: The constrained solution is computed using (3.25) and the parameters of FH model are thus produced.

3.6 Chapter Summary

A thorough study of Fuzzy Hammerstein model is presented in the chapter. The Hammerstein model has a block oriented structure. The part of static nonlinearity is modeled using zero order TS fuzzy model and then recursive least squares estimation approach is used to determine the linear and nonlinear parameters of the generalized model. Because of the nonlinear parameter, there are certain constraints which are linearized and iteratively applied to determine the overall fuzzy Hammerstein model.

CHAPTER 4

4 Control Scheme

4.1 Prologue

The chapter expounds the overall block diagram of the control scheme and then dissects the overall methodology to illustrate different nodes in the scheme.

4.2 The outline of control scheme

The overall control scheme is composed of an adaptive inverse controller combined with generalized predictive control. The adaptive inverse controller is used to eliminate the effect of static nonlinearity of the plant while generalized predictive control is implemented to control the delayed plant in the presence of constraints imposed by actuator limitations. As discussed in chapter 3, online input output data of the unknown plant will be used to identify the model parameters adaptively so that any variation in plant parameters can be detected and submitted for online control. The scheme is depicted in Fig 4.1.

4.3 Brief Description of the overall Block Diagram

As shown in Fig 4.1, a filtered reference signal generates the desired set point for GPC. Since GPC requires an explicit plant model to predict the output so an online iterative parameter estimation scheme is continuously updating plant’s parameters.

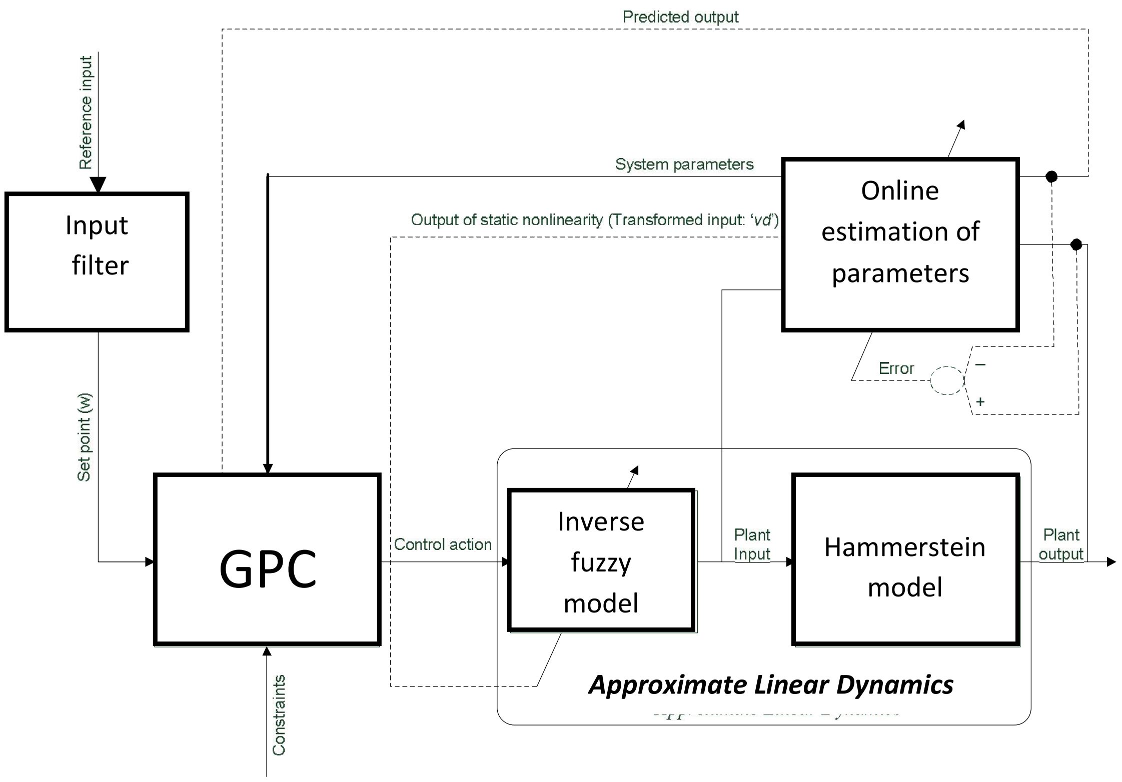

Fig 4.1: Overall block diagram of the proposed control scheme

Furthermore for the reduction of cost function, GPC requires predicted output of the plant. This predicted output is also generated from the iterative parameter estimation block. Simultaneously the block is generating a set of transformed variables (i.e. output of the static nonlinearity as consequents of fuzzy model).

The block for online iterative estimation of parameters, as shown in Fig 4.1, is adaptive by making use of the error between plant's original output and its predicted output, produced by the same block.

The iteratively updated set of transformed variables is an input for the adaptive inverse model controller which generates the input signal for the plant. Since the plant contains static nonlinearity, the plant input, generated by the inverse fuzzy model of static nonlinearity, will cancel out its effect. Hence it can be argued that since static nonlinearity is being cancelled out approximately, the GPC scheme is controlling an approximate linear dynamic plant.

The strategy further gives benefit in optimization where the cancellation of nonlinearity causes convex optimization. When actuator limitations are considered then a constrained control is required for which constraints are also incorporated in GPC algorithm. One important note is that in spite of the static nonlinearity, adaptive inverse controller is used. This is because it is advantageous in two domains, firstly a better approximate of transformed variable can be obtained and secondly the scheme will remain equally workable for all sort of static nonlinearities i.e. it can be used for different nonlinearities and different linear dynamics.

4.4 Dissection of the scheme

This section dissects the overall control scheme and defines each node of the block diagram given in Fig 4.1 in detail.

4.4.1 Input Filter

Input filter is also commonly known as reference model which quantifies the desired performance. The designer must decide what to choose for the reference model that quantifies the desired performance. The specified characteristics should be desirable as well as reasonable. Not only this, but, if an input filter is not used, sharp transitions of the input signal may cause serious damage to the control system

The performance of the overall system is computed with respect to the input filter by generating an error (e_p) between set point (w) and the plant's output (y).

$$e_p = w - y$$

4.4.2 Online Iterative estimation of parameters

This topic has been covered in detail in chapter 3. Briefly commenting, an adaptive scheme is used for online estimation of the parameters of Fuzzy Hammerstein model using constrained recursive least squares approximation. The adaptive mechanism learns on the basis of error between plant's original output and its predicted output.

4.5 Adaptive Fuzzy Control

There are two general approaches to adaptive control, the first of which is depicted in Fig 4.2. In this approach the “adaptation mechanism” observes the signals from the control system and adapts the parameters of the controller to maintain performance even if there are changes in the plant. Sometimes, the desired performance is characterized with a “reference model,” and the controller then seeks to make the closed-loop system behave as the reference model would even if the plant changes. This is called “model reference adaptive control”.

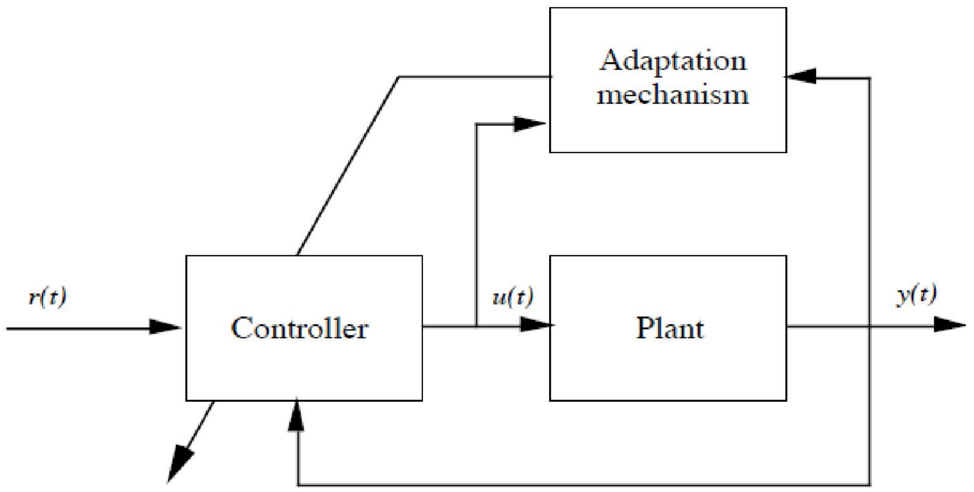

Fig 4.2: Direct Control approach of Inverse Controller designing

In the second general approach to adaptive control, which is shown in Fig 4.3, we use an on-line system identification method to estimate the parameters of the plant and a “controller designer” module to subsequently specify the parameters of the controller. If the plant parameters change, the identifier will provide estimates of these and the controller designer will subsequently tune the controller. It is inherently assumed that we are certain that the estimated plant parameters are equivalent to the actual ones at all times (this is called the “certainty equivalence principle”). Then if the controller designer can specify a controller for each set of plant parameter estimates, it will succeed in controlling the plant. The overall approach is called “indirect adaptive control” since we tune the controller indirectly by first estimating the plant parameters (as opposed to direct adaptive control, where the controller parameters are estimated directly without first identifying the plant parameters).

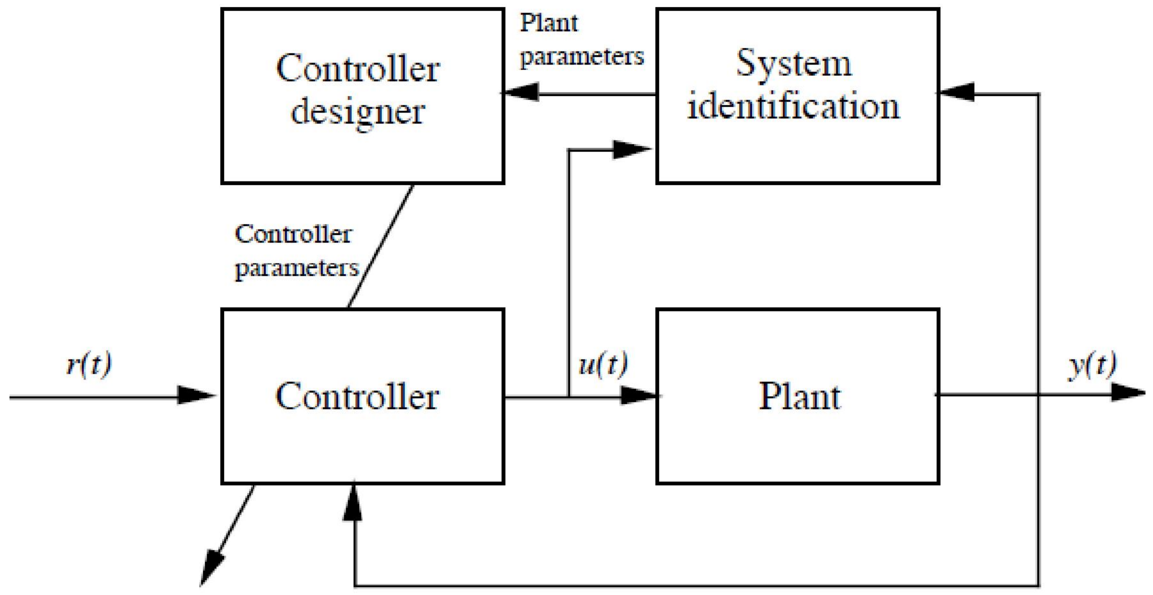

Fig 4.3: Indirect Control approach for Inverse Controller designing

4.6 Adaptive Fuzzy Model Inverse Controller

The simplest way to control a process, when an inverse model is available, is to use this inverse model in an open loop configuration.

If an ideal model of the process is available i.e. the model is equal to the process, and both model and controller (inverse model) are input output stable, the control is perfect and input output stable (Economou, *et al.*, 198l). This situation of perfect control is impossible to achieve because an exact inversion of the process can only be found in special situations, and the model is never equal to the process, resulting in model plant mismatches. Moreover the variables of the process can be subjected to level and rate constraints and disturbances acting on the process are present and not taking into account in the controller. Further when the system has a delay of *d* steps, the inverse must be done for *d* steps ahead. All these problems must be overcome, in order to apply inverse model control in practice [43].

Consider the general form of a multi-input multi-output (MIMO) model

$$\mathbf{x}(k+1) = \mathbf{f}\,(\mathbf{x}(k),\mathbf{u}(k))$$

$$\mathbf{y}(k) = \mathbf{g}(\mathbf{x}(k)) \qquad (4.1)$$

Where

- $u(k) \in \mathcal{U} \subset R^m$ are the control actions
- $y(k) \in \mathcal{Y} \subset R^p$ are the outputs of the system
- $x(k) \in \mathcal{X} \subset R^n$ are the states and k denotes the time sample

The domains of u, y and x are respectively given by ***u***, ***y*** and ***x*** the orders of control actions, outputs and states are denoted by m, p, n ε N, respectively. The function **f** relates the states at time k+1 with the states and the control action at time k, and the function **g** relates the outputs with the states at time k. for the structure in (4.1), any MIMO model can be decomposed in p MISO models without lack of generality. The MISO fuzzy system, shown in Fig 4.2, with n states is given as:

$$\hat{y}(k+\mathbf{1}) = \mathbf{f}(\mathbf{x}(k)) \qquad (4.2)$$

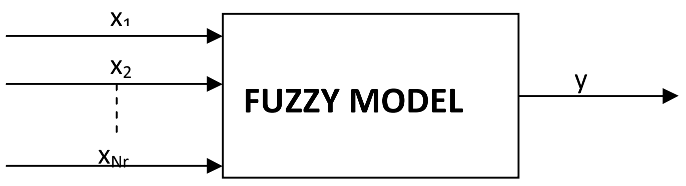

Fig 4.4: General MISO Fuzzy model

There are two ways of inverting this fuzzy model:

- Global inversion

➢ Partial inversion

In ***Global inversion*** of the model, where all states become outputs of the inverted model and the output of the original model becomes the state of the inverted model as depicted in Fig 4.5. Thus this inversion computes all the state variables when the original output is given. The solution of this inversion is normally not unique and it is given by a family of solutions [43].

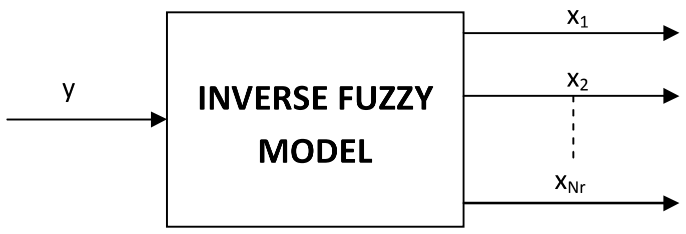

Fig 4.5: Global inversion of Fuzzy model

In ***Partial inversion***, only one of the states of the original model turns out to be an output of the inverted model and the other states, accompanied by the original output, are the inputs of the inverted model as depicted in Fig 4.6.

Partial inversion usually has a unique solution which is a big advantage compared to a global inversion [43]. In partial inversion, inverted state is known as controllable input for the original system. The states that are not inverted are called non-controllable, albeit these are the inputs of the system. This method cannot be directly applied if several control actions are needed at the same time however it can produce several input actions independently, by applying partial inversion to every state corresponding to an input of the system.

Partial inversion is applicable only if the inverse of the fuzzy model exists and if this inversion is unique also. If the inversion is not unique, some supplementary criteria must be included to attain the best solution. When the inverted model is utilized as a controller, these criteria of existence and uniqueness must be taken care to determine the best control action.

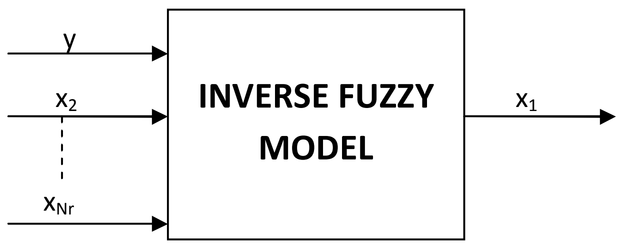

Fig 4.6: Partial inversion of Fuzzy model

A fuzzy model is invertible if the model is represented by a function

$$y = f\left(u, x_2, \ldots\ldots, x_n\right) \tag{4.3}$$

and the inverted function exists, such that

$$u = f^{-1}\left(y, x_2, \ldots\ldots, x_n\right) \tag{4.4}$$

This statement implies that the function describing the original fuzzy model must be strictly monotone with respect to *u* [43]. The discussion of the invertibility conditions for the singleton fuzzy model will be presented in Section 4.6.1.1.

4.6.1 A useful tool for the formulation of inverse model

Since model based control is penetrating in industrial applications, the triumphant application of fuzzy models requires that it should be able to generate such elements which could be incorporated in model-based controllers, such as fuzzy model inversion and fuzzy model linearization schemes. Input reduction of fuzzy models is a useful tool for extracting knowledge from fuzzy models for the easy formulation of inverse fuzzy model. As it is shown in Fig 4.7, when some of the input variables of the model are fixed, the fuzzy model can be transformed into a smaller model that contains fewer input variables.

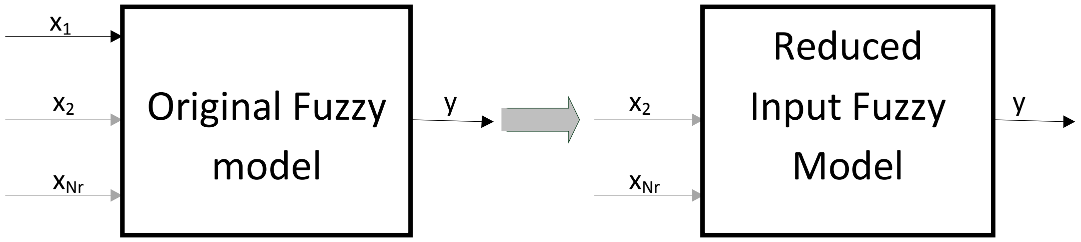

Fig 4.7: Reduced input Fuzzy model from original Fuzzy model using partial reduction of the input variables

TS Fuzzy model reduction

If the fuzzy model has Ruspini partition, the rules of the original fuzzy model can be written as [21]:

$$\begin{aligned}&\textbf{if } z_1 \textbf{ is } A_{i,j} \textbf{ and........and } z_n \textbf{ is } A_{n,j}\\ &\textbf{then } y = f_j(q)\end{aligned} \tag{4.5}$$

Equation (4.5) can also be written as:

$$\begin{aligned}&\textbf{if } z_1 \textbf{ is } A_{i,j} \textbf{ and........and } z_s \textbf{ is } A_{s,j}\\ &\textbf{then (if } z_{s+1} \textbf{ is } A_{s+1,j} \textbf{ and........and } z_n \textbf{ is } A_{n,j} \textbf{ then } y = f_j(q)\textbf{)}\end{aligned} \tag{4.6}$$

At given $\mathbf{z_\alpha=[\ z_1,......,z_s]}$, the model can be reduced by using partial input reduction. The simplified models have rules like:

$$\begin{aligned}&R_j = \textbf{if } z_{s+1} \textbf{ is } A_{s+1,j} \textbf{ and........and } z_n \textbf{ is } A_{n,j}\\ &\textbf{then } y = f_j(q)\end{aligned} \tag{4.7}$$

where the consequent functions of the resulted fuzzy model are calculated as:

$$\tilde{f}_j(\mathbf{q}) = \sum_{j=1}^{N_r}[(\prod_{\substack{l=1\\ l\neq i}}^{n} A_{l,j}(z_l))f_j(\mathbf{q})] \tag{4.8}$$

For first order TS fuzzy model, this results in**:**

$$\tilde{f}_j(\text{q}) = \sum_{j=1}^{m} \tilde{p}_j^l q_l \qquad (4.9)$$

Where

- q_o=1
- $\tilde{p}_j^l = \sum_{j=1}^{N_r} [(\prod_{\substack{l=1 \\ l \neq i}}^{n} A_{l,j}(z_l)).p_j^l]$

This equation suggests that the TS fuzzy model is a special case of a hierarchical local model network [33], when the subnetworks are fuzzy models that can be obtained by the partial input reduction [43].

4.6.1.1 Methods of Inversion

Following are the two widely used methods of inversion:

- By using input-output data
- By using original model

These two methods are detailed below:

4.6.1.1.1 By using input-output data

Using input output data for the identification of inverse model may be the most discerning approach to inverse modeling; it tries to correspond the data in an inverse function f^1 (Batur, et al., 1993). Two key approaches are perceived in this approach

- Direct inverse learning
- Specialized inverse learning

***Direct Inverse Learning**:* In this method the process is stimulated by a training signal and the fuzzy system renovate the input signal from the given output signal as shown in Fig 4.8.

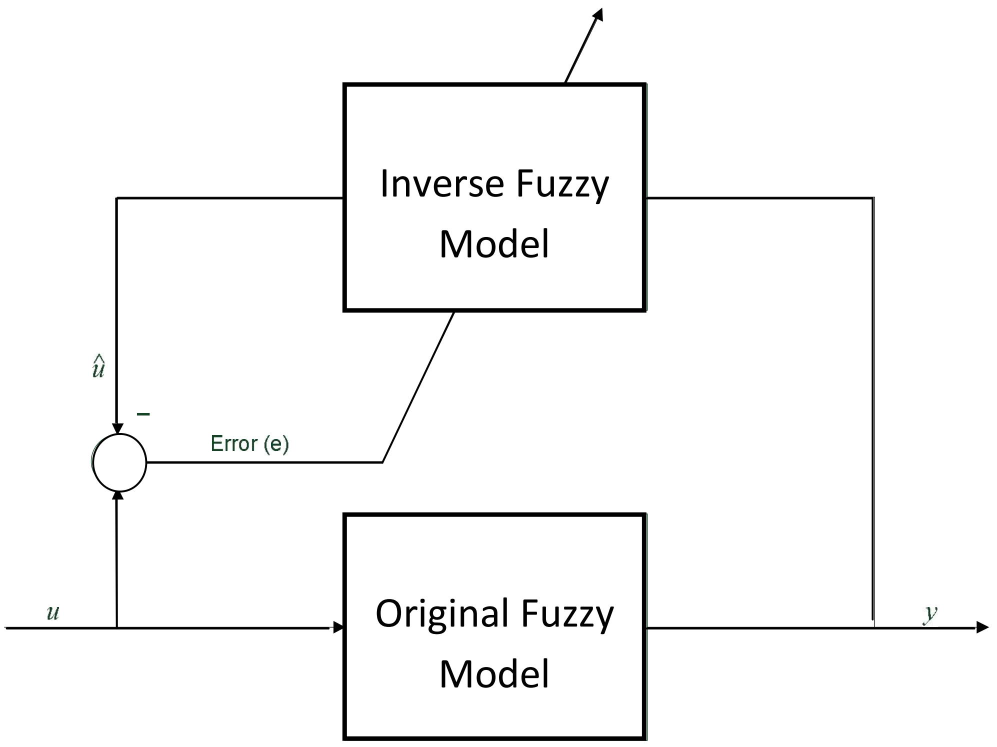

Fig 4.8: Direct Inverse Learning scheme

Various identification algorithms can be used to obtain the inverse model. If the approach of least squares estimation is employed, the identification algorithm plots y to the mean value of all the inputs u. This methodology leads to a consequential inverse model. Other methods, such as the one proposed by (Nelles and Fischer, 1996) does not result in a unique solution for u moreover the dynamics of the system can be a many-to-one mapping.

It is often tricky to obtain a suitable training signal for direct inverse learning. This is because the inverse model is assumed to perform over a wide range of input amplitudes on y and for a huge bandwidth. However, as the stimulation of the system is commenced as the activation of u, a persistent excitation of y cannot be ensured.

Specialized Inverse Learning*:* In this scheme, the inverse model is cascaded with the forward plant model or the process as shown in Fig 4.9. The parameters of inverse model P^{-1} are adapted by minimizing the deviation between the output y and the reference r.

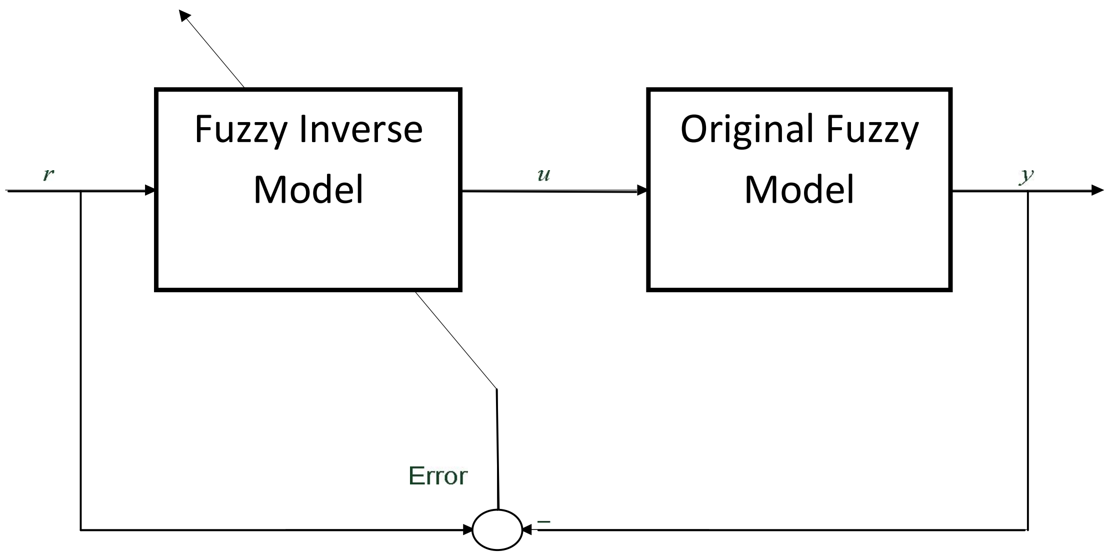

Fig 4.9: Specialized Inverse Learning scheme

Hence the adaptation is a goal-oriented technique [43] and the process is automatically excited with the correct signal if a typical reference trajectory must be followed. This shows that the drawbacks of direct inverse learning can be permeated by specialized inverse learning. But it is still difficult to use this inverse model in control scheme due to the model plant mismatch and disturbance influences. Therefore the best solution seems to be to invert a fuzzy model exactly by using some analytical method.

4.6.1.1.2 Inversion of the original model

In this method plant model is directly available rather than its input-output data which is directly used to develop the inverse.

4.6.1.2 T-S Fuzzy Model inversion

In two exclusive cases, TS fuzzy model can be inverted analytically.

- If $m = 0$ or TS model if of 0th order (fuzzy model is the member of Singleton Fuzzy model family).
- If the TS fuzzy model is affine in the relevant input i.e $m \neq 0$ and $x \notin z$.

For 0th order fuzzy model, Babuska *et al* [38] and Barayani *et al* [35] proposed methods for the exact inversion of a singleton fuzzy model. For affine TS model, Fischer's method [19],[34] can be used for inversion.

For other than these two cases, the fuzzy model cannot be inverted analytically yet it is possible to compute the inverse numerically. For numerical inverse of the fuzzy model, the inversion has to be expressed as a nonlinear optimization problem.

The approaches of analytical inversion will now be discussed. All methods are derived by the conversion of general MISO fuzzy model, $y = f(\mathbf{x})$, into SISO model $y=\tilde{f}(x)$ with the use of partial input reduction [21] proposed in the Section 4.6.1.

For a given input vector

$$\mathrm{x}=[\mathrm{x}^*, x_i],\ \mathrm{z}_\alpha = [\mathrm{x}^*] \text{ and } \mathrm{z}_\beta = [x_i],$$

MISO fuzzy model is given as:

$$y = f(\mathrm{x}^*, x_i)$$

when this MISO model is transformed into SISO model by using input reduction, following model is obtained:

$$y=\tilde{f}(x_i)$$

where $\tilde{f}$ is determined by partial input reduction. The fuzzy rules of reduced model are given as:

$$\tilde{R}\text{: }\mathbf{If}\ x\ is\ A_j(x_i)\ \mathbf{then}\ \hat{y}=\tilde{f}(\mathbf{q})$$

Thus the inversion of a general MISO fuzzy model is reformulated to the inversion of partially defuzzified SISO fuzzy model. If an invertible function is represented by this simplified fuzzy model, the inverse fuzzy model can be represented by a unique mapping from the input to the output as mentioned below:

$$x_i = \tilde{f}^{-1}(y) \tag{4.10}$$

4.6.1.2.1 Inversion of Singleton Fuzzy model

The partially reduced inverted model of singleton fuzzy model formulated as:

$$\tilde{R} : If\ x\ is\ A_j(x_i)\ then\ y = \tilde{p}_j^0 \tag{4.11}$$

When the original fuzzy model is first order TS type, still the singleton fuzzy model results by using partial input reduction. This can occur if $z_n = x_i$ and $x_i \notin \mathbf{q}$ [21].

$$\tilde{p}_j^0 = \sum_{k=0}^{m} \{ \sum_{j=1}^{N_r} [(\prod_{\substack{l=1 \\ l \neq i}}^{n} A_{l,j}(z_l)).p_j^l].q_k \} \tag{4.12}$$

The rules of the inverse fuzzy model have the following general form:

$$\tilde{R}_j^{(-1)} : If\ y\ is\ \tilde{A}_j\ then\ x_i = \tilde{a}_j \tag{4.13}$$

Where $\tilde{\boldsymbol{A}}_{\mathbf{1},\boldsymbol{j}}$ represents the antecedent membership functions of the inverse model defined on the domain of the output of the forward fuzzy model. These antecedent sets are shown in Fig 4.10 where $P_j(y) = \tilde{\boldsymbol{A}}_{\mathbf{1},\boldsymbol{j}}(y)$ and $p_j = \tilde{\boldsymbol{p}}_{\boldsymbol{j}}^{\mathbf{0}}$ for $1 \le j \le N_R$.

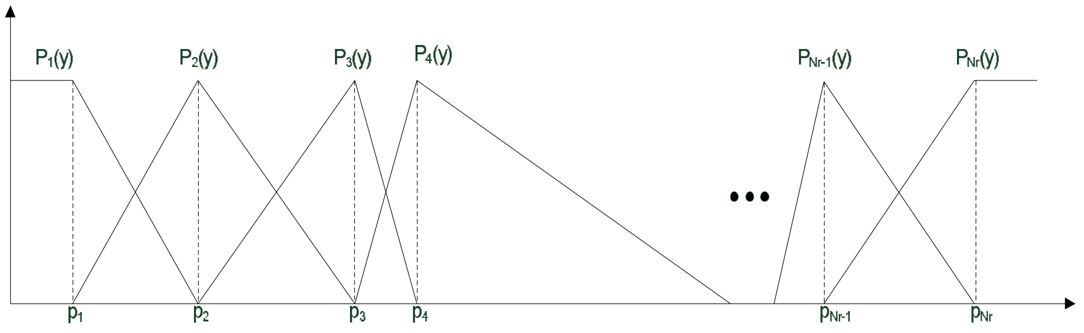

Fig 4.10: Antecedent Fuzzy sets for Fuzzy inverse model

$$\tilde{A}_{1,j}(y) = \frac{y - \tilde{p}_{j-1}^0}{\tilde{p}_j^0 - \tilde{p}_{j-1}^0} \qquad \text{for} \qquad \tilde{\boldsymbol{p}}_{\boldsymbol{j-1}}^{\mathbf{0}} \le \mathbf{y} < \tilde{\boldsymbol{p}}_{\boldsymbol{j}}^{\mathbf{0}}$$

$$\tilde{A}_{1,j}(y) = \frac{\tilde{p}_{j+1}^0 - y}{\tilde{p}_{j+1}^0 - \tilde{p}_j^0} \qquad \text{for} \qquad \tilde{\boldsymbol{p}}_{\boldsymbol{j}}^{\mathbf{0}} \le y < \tilde{\boldsymbol{p}}_{\boldsymbol{j+1}}^{\mathbf{0}}$$

$\tilde{A}_j(y) = 0$ otherwise

An interpolation method must be applied to obtain a_j, the centres of the consequent fuzzy sets of inverse model because the consequents for inverse model, $\tilde{\boldsymbol{p}}_j^{\mathbf{0}}$, are singletons. This interpolation is carried out by the fuzzy sets $\tilde{\boldsymbol{A}}_{\boldsymbol{j}}$ shown in Fig 4.11. If the desired reference $r(k+1)$is within the range of the reached states from the actual state $x(k)$, i.e.$\tilde{\boldsymbol{p}}_1^{\mathbf{0}} \leq r(k+1) \leq \tilde{\boldsymbol{p}}_{N_r}^{\mathbf{0}}$then the serial connection of the inverted and the forward fuzzy model produces an identity mapping. The whole method is shown in Fig 4.11.

Fig 4.10 has the conventions similar to Fig 4.11. From Fig 4.10, it is clear that the inversion of reference $r(k+1)$ is given by one and only one point. If desired output is unable to be reached from the current state in one time step, i.e. $r(k+1) < \tilde{\boldsymbol{p}}_1^{\mathbf{0}}$ or $r(k+1) > \tilde{\boldsymbol{p}}_{N_r}^{\mathbf{0}}$, the control action is still able to generate mapping with minimal error. Whwn $r(k+1) > \tilde{\boldsymbol{p}}_{N_r}^{\mathbf{0}}$,$\boldsymbol{\mu}_{\tilde{\boldsymbol{p}}_{N_r}^{\mathbf{0}}}(r(k+1))=1$, and the control input is $a(k)$ = core (A_{Nr}). The degree of fulfillment of control action is given by:

$$\boldsymbol{\mu}_{A_{N_r}}(r(k+1)) = 1$$

which yields the model output

$$y(k+1) = \tilde{\boldsymbol{p}}_{N_r}^{\mathbf{0}}$$

As $\tilde{\boldsymbol{p}}_{N_r}^{\mathbf{0}} > \tilde{\boldsymbol{p}}_j^{\mathbf{0}}$, when $1 \leq j \leq N_r-1$, the difference $| r(k+1) - y(k+1) |$ is the minimum possible. Parallel for $r < \tilde{\boldsymbol{p}}_1^{\mathbf{0}}$, the control action $a = core\ (A_1)$ yields the output

$$y(k+1)=\tilde{\boldsymbol{p}}_1^{\mathbf{0}}$$

This gives the best control action, since $\tilde{\boldsymbol{p}}_1^{\mathbf{0}} < \tilde{\boldsymbol{p}}_j^{\mathbf{0}}$, when $2 \leq j \leq N_R$.

The discussion shows that Mamdani inference is basically used to generate the inverse TS model using the fuzzy sets developed from the singleton consequents of forward model. These fuzzy sets are used as the antecedents of Mamdani model and the fuzzy sets of the inputs of forward model are used as the consequents of Mamdani model.

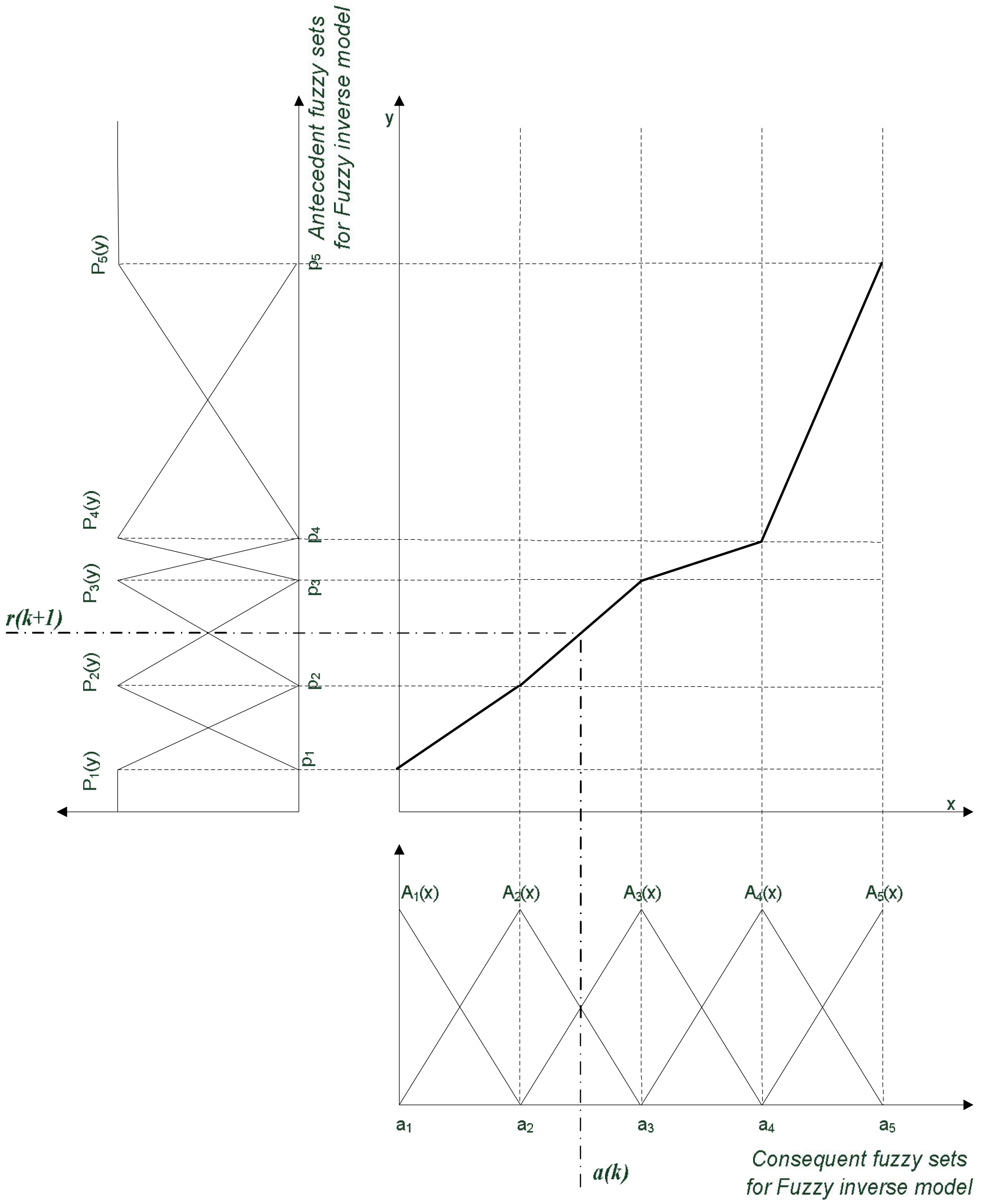
Antecedent fuzzy sets for Fuzzy inverse model
y
P5(y)
p5
P4(y)
p4
P3(y)
p3
r(k+1)
P2(y)
p2
P1(y)
p1
x
A1(x)
A2(x)
A3(x)
A4(x)
A5(x)
a1
a2
a3
a4
a5
a(k)
Consequent fuzzy sets for Fuzzy inverse model

Fig 4.11: Inversion of SISO Fuzzy inverse model

4.6.1.3 Adaptive Inversion of fuzzy model

As described above, an adaptive fuzzy inverse controller is required for the control scheme that is under the study scope. The presented method is a standard algorithm which uses Mamdani model to develop the inverse controller. Mamdani method is well known for capturing expert knowledge and depicts the expertise in more innate and more human-like manner. However, Mamdani-type fuzzy inference involves a substantial computational burden, hampering its utilization in adaptive control. So for an adaptive fuzzy inverse controller, the better choice is to use TS model because it is computationally more effective and works well with optimization and adaptive techniques. This makes it very attractive in control problems.

The structure of adaptive inverse controller is that of Direct Inverse controller however there is no impact of its drawbacks on control scheme because:

- Non uniqueness of solution is removed by using Recursive least squares estimation which yields a unique solution.
- The inverse controller is implemented in a model predictive control scheme which excites the inverse controller by itself through an optimized control action hence appropriate perturbation signal is automatically available.

In the presented work, the adaptive Inverse controller is further equipped with feedback. The overall block diagram of the inverse controller is shown Fig 4.12:

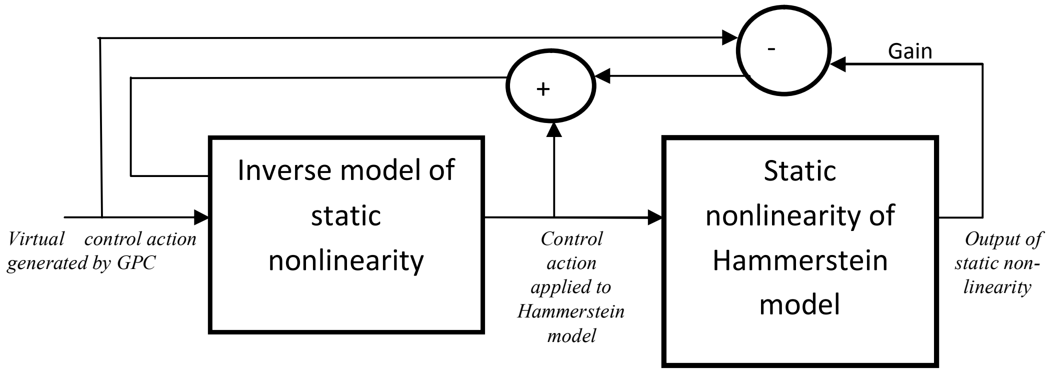
-
Gain
+
Inverse model of static nonlinearity
Static nonlinearity of Hammerstein model
Virtual control action generated by GPC
Control action applied to Hammerstein model
Output of static non-linearity

Fig 4.12: Block Diagram of Adaptive Inversion technique

The scheme depicted in Fig 4.12 makes use of TS model combined with TAN's work [45] in which he utilizes feedback in inverse modeling technique. Dexter's work is blended with adaptive TS model in this control strategy to effectively cancel out the effects of static nonlinearity. The performance of the controller will be shown in chapter 5. The flow of work for the adaptive inverse fuzzy controller is described in the flow diagram shown in Fig 4.13.

Online estimation of fuzzy model consequents has been discussed in chapter 3. These consequents are used as centers for developing the antecedent fuzzy sets for the inverse model controller. The method is detailed above in Section 4.6.1.2. Recursive least squares algorithm is used to estimate the consequents for the fuzzy inverse model recursively. The obtained output is then compared with the actual input of the plant. The RLS algorithm minimizes the error i.e. the difference between actual plant input and the defuzzified output of the inverse model. This error adapts the inverse model controller and a suitable plant input is designed.

4.7 Generalized Predictive Control

In spite of the fact that self-tuning and adaptive control has developed a lot over the past decades in terms of theoretical understanding as well as practical applications; there is still no general purpose algorithm which can ensure stable control for the majority of real world processes [9]. A general purpose algorithm, on the other hand, must be pertinent to:

- *Plant with non-minimum phase:* This is a general problem. Continuous time transfer functions usually tend to exhibit discrete-time zeros outside the unit circle when sampled at very fast rate.
- *Plant with variable or unknown dead-time:* Minimum-variance self-tuners and the like schemes are highly sensitive to the theories set for the dead-time moreover the methods attempting to approximate the dead-time using operating data tend to be highly complex and lack robustness.

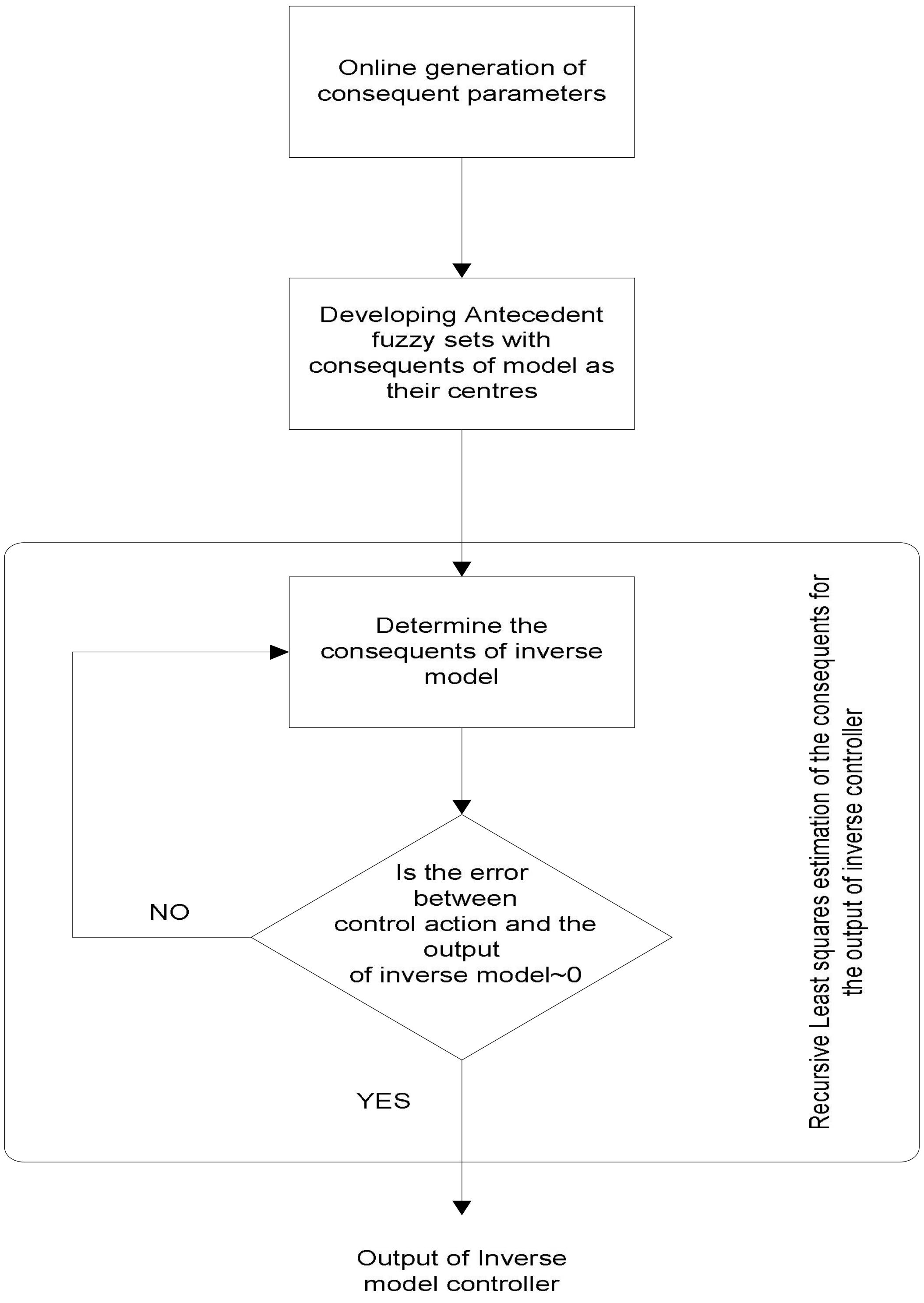
Online generation of consequent parameters
Developing Antecedent fuzzy sets with consequents of model as their centres
Determine the consequents of inverse model
Is the error between control action and the output of inverse model~0
NO
YES
Recursive Least squares estimation of the consequents for the output of inverse controller
Output of Inverse model controller

Fig 4.13: Flow diagram for the design of Adaptive inverse controller

- *Plant with certain instabilities:* For systems such as flexible spacecraft or robots, there are usually badly-damped poles. Furthermore an open-loop unstable plant is usually common in practical applications such as helicopter.
- *Plant with unknown order:* the set forth schemes of pole-placement and LQG self-tuners execute badly if in the system identification, plant's order is overestimated because of pole-zero cancellations. Special precautions need to be taken for stable control in this situation [9].

Generalized Predictive Control or GPC seems to surmount all of these problems in a single algorithm. It can provide stable control of processes with variable parameters, varying dead-time, and with instantaneously changing model order provided that the input output data are adequately rich that can permit satisfactory plant identification. Hence it is suited to high performance applications such as the control of flexible systems.

GPC has several ideas similar to other predictive controllers because it is developed upon the matching concepts but it has some dissimilarity as well. In the absence of constraints, it provides an analytical solution. Moreover it can handle unstable and non-minimum phase plants and integrates the concept of control horizon and that of the weighting of control increments in cost function. Usually the choices available for GPC lead to a larger variety of control objectives in comparison to other approaches; however some these may be the limiting cases for GPC.

The primary inspiration of GPC is to compute a sequence of future control actions such that it can minimize a cost function defined over a prediction horizon. It is required to optimize the performance index. This index is an expectation of the quadratic function measuring the error between the reference sequence and the predicted system output over some prediction horizon added with a quadratic function measuring the control effort. A rudimentary GPC scheme is depicted in Fig 4.14. It comprises of the plant to be controlled, a linear model of the plant, and an algorithm for the minimization of Cost Function (CFM) that establishes the required input to generate desired performance for the plant. The CFM block is contained within GPC algorithm [42].

The flow of GPC scheme however begins with the reference signal, *r,* which is submitted to the input filter or the reference model. This model generates a tracking reference signal, *w,* which is utilized as an input to the algorithm for cost function minimization. The output of CFM algorithm is used as an input to the plant.

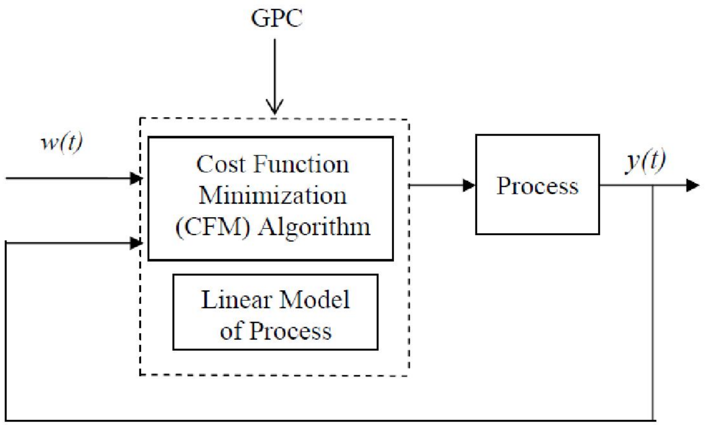

Fig 4.14: Basic structure of GPC

CFM algorithm calculates the next control action, $u(k+1)$, from predictions of response determined by the model of the plant. After the minimization of cost function, this input is forwarded to the plant.

4.7.1 The CARIMA model

As discussed earlier, every algorithm of MPC requires an explicit model. For GPC algorithm, the most widely accepted is the Controlled Auto-Regressive and Integrated Moving-Average (CARIMA) model. In the following, CARIMA model development will be described.

When regulation is considered about a particular operating point, even a nonlinear plant generally admits a locally-linearized model [9]:

$$\overline{A}(q^{-1})y(t) = \overline{B}(q^{-1})u(t-1) + z(t) \tag{4.14}$$

Where:

- *u(t)* is the control input

- y(t) is the measured variable or output
- *z(t)* is a disturbance term.

$\bar{\boldsymbol{A}}$ and $\overline{\boldsymbol{B}}$ are polynomials which are given as:

$$\bar{A}(q^{-1}) = 1 + \bar{a}_1 q^{-1} + ... + \bar{a}_{na} q^{-na}$$
$$\bar{B}(q^{-1}) = \bar{b}_0 + \bar{b}_1 q^{-1} + ... + \bar{b}_{nb} q^{-nb}$$

It is obvious that for the plants having non-zero dead-time, the leading elements of the polynomial $B(q^{-1})$, are zero.

Usually *z(t)* is considered to be of moving average form:

$$z(t) = \bar{D}(q^{-1})\xi(t) \tag{4.15}$$

Where

- $\bar{D}(q^{-1}) = 1 + \bar{d}_1 q^{-1} + ... + \bar{d}_{nd} q^{-nd}$

n_a, n_b and n_d are the orders of polynomial $\bar{A}$, $\bar{B}$ and $\bar{D}$ respectively.

In (4.15), $\xi(t)$ is an uncorrelated random sequence. By substituting (4.15) into (4.14), *Controlled Auto-Regressive and Moving- Average* (CARMA) model is obtained [9]:

$$\bar{A}(q^{-1})y(t) = \bar{B}(q^{-1})u(t-1) + \bar{D}(q^{-1})\xi(t) \tag{4.16}$$

This model is however inappropriate for many industrial applications in which non-stationary disturbances are incorporated.

Two principal disturbances are usually encountered in practice:

- Random steps that occur at random times (For example, degradation in the quality of material)
- Brownian motion (Usually present in plants depending on the energy balance).

For both these cases, appropriate disturbance model is:

$$z(t) = \frac{\bar{D}(q^{-1})\xi(t)}{\bar{\Delta}} \tag{4.17}$$

Where $\bar{\Delta}$ is the difference operator ($1\text{-}q^{-1}$)

When the model in (4.14) is coupled with (4.17), a new model is produced which is well known as CARIMA model:

$$\bar{A}(q^{-1})y(t) = \bar{B}(q^{-1})u(t-1) + \frac{\bar{D}(q^{-1})\xi(t)}{\bar{\Delta}} \tag{4.18}$$

For simplicity in calculation we suppose that:

$$\bar{D}(q^{-1}) = 1$$

Therefore the simplified CARIMA model is given as:

$$\bar{A}(q^{-1})y(t) = \bar{B}(q^{-1})u(t-1) + \frac{\xi(t)}{\bar{\Delta}} \tag{4.19}$$

4.7.2 Unconstrained GPC

4.7.2.1 Using Recursive Diophantine equation

Consider the polynomial identity given (4.19) to derive a predictor which is *j*-step ahead of the predicted output $\hat{y}(t+j)$ developed on the basis of CARIMA model:

$$1 = M_j(q^{-1})\bar{A}\bar{\Delta} + q^{-j}N_j(q^{-1}) \tag{4.20}$$

Where M_j and N_j are polynomials which are uniquely defined using $\bar{\boldsymbol{A}}(j)$ and the interval of prediction *j*. CARIMA model results in (4.21) when it is multiplied with $M_j\Delta q^j$.

$$M_j\bar{A}\bar{\Delta}y(t+j) = M_j\bar{B}\bar{\Delta}u(t+j-1) + M_j\xi(t+j) \tag{4.21}$$

By substituting the value of $M_j\bar{\boldsymbol{A}}\bar{\Delta}$ from (4.20), we get:

$$y(t+j) = M_j\bar{B}\bar{\Delta}u(t+j-1) + N_j y(t) + M_j\xi(t+j) \tag{4.22}$$

Since the degree of $M_j(q^{-1})$ is j-1, all the noise elements are in future. This reveals that with measured output data up to time t and any given $u(t + i)$for i>l the optimal predictor is clearly (4.23):

$$y(t+j\,|\,t) = \bar{G}\bar{\Delta}u(t+j-1) + N_j y(t) \tag{4.23}$$

Where

$$\bar{G}_j\left(q^{-1}\right) = M_j\bar{B} \tag{4.24}$$

From (4.23) and (4.24), it is clear that:

$$\bar{G}_j(q^{-1}) = \frac{(q^{-1})\left[1 - q^{-j}N_j(q^{-1})\right]}{A(q^{-1})} \tag{4.25}$$

This implies that one way of computing $\bar{G}_j$ is to consider the z-transform of the step response of plant and employ the first j terms (Clarke and Zhang, 1985).

In GPC, the overall set of predictions is considered for which j runs from a minimum prediction horizon up to the maximum prediction horizon in contrary to GMV self-tuning controllers where only one prediction i.e. $\hat{y}(t + s|t)$ is considered.

For the prediction interval that is smaller than system delay ($j < s$), the prediction process $\hat{y}(t + s|t)$ *is* completely dependent on the data that is available, but when the prediction interval is greater than or equal to the system delay ($j \geq s$), the designer needs to make assumptions about future control actions. These assumptions are the keystones of the GPC strategy.

Recursion of Diophantine Equation

Different strategies have been proposed by researchers for the implementation of long-range or *j* steps ahead predictions. De Keyser and Van Cauwenberghe (1982, 1983) and Mosca *et al.,* (1984) suggest to use a bank of self-tuning predictors for each horizon *j*. Yet another approach is to resolve the polynomial identity given in (4.20) recursively for the considered range of *j* to determine M_j and N_j numerically. The drawback of these approaches is that these are *computationally expensive*. On the other hand, recursion of Diophantine equation to determine $M_{j+1}$and $N_{j+1}$from given the values of M_j and N_j is remarked as easier and more efficient [41].

For simplicity of notation, it is assumed that M = M_j, $R = M_{j+1}$, $N = N_j$, $S = N_{j+1}$. Consider two Diophantine equations in which, $\bar{A}\Delta$ is defined by $\breve{A}$.

$$1 = M\breve{A} + q^{-j}N \tag{4.26}$$

$$1 = R\breve{A} + q^{-(j+1)}S \tag{4.27}$$

Subtracting (4.26) from (4.27) results in:

$$0 = \breve{A}(R - M) + q^{-j}(q^{-1}S - N) \tag{4.28}$$

The degree of polynomial *R-M* is *j* and it may be split up into two parts:

$$R - M = \breve{R} + r_j q^{-j}$$

Substituting the split polynomial in (4.28), we get:

$$\breve{A}\breve{R} + q^{-j}(q^{-1}S - F + \breve{A}r_j) \tag{4.29}$$

From (4.29), it is clear that if $\breve{R}$=0 then S = $q(\breve{A}r_j - F)$.

Since the leading element of $\breve{A}$ is unity we have:

$$r_j = f_0 \tag{4.30}$$

$$S_i = f_{i+1} - \breve{a}_{i+1}r_j \tag{4.31}$$

For i=0 for the degree of $S(q^{-1})$ and

$$R(q^{-1}) = M(q^{-1}) + q^{-i}r_j \tag{4.32}$$

$$\overline{G}_{i+1} = \overline{B}(q^{-1})R(q^{-1}) \tag{4.33}$$

Therefore when plant polynomials, $\bar{A}(q^{-1})$ and $\bar{B}$ (q^{-1}),and one solution of $M_j(q^{-a})$ and $N_j(q^{-1})$ are available then (4.30) and (4.31) can be used to obtain $N_{j+1}(q^{-1})$ and (4.32) to obtain $M_{j+1}(q^{-1})$ and so on, with reduced computational effort. For the initialization of the iterations note that for $j = 1$:

$$1 = M_1 \breve{A} + q^{-1}N_1$$

and because the leading element of $\breve{A}$ is 1 so for the first iteration:

$$M_1 = 1 \text{ and } N_1 = q(1 - \breve{A})$$

The drawn in calculations using recursive Diophantine equations are clearly straightforward and easier than those required for single predictor (for each output horizon).

4.7.2.2 Using Toeplitz Hankel matrices

After Clarke's proposal for generalized predictive control (GPC), several customized methods have been exploited to improve the control performance or cope with the heavy on-line computation problem. Some authors have used the process model directly to change the form of control law [41]. Using these algorithms, there is no necessity of solving Diophantine equation. On the other hand, the formula of parameters solution is made simpler by the modified generalized predictive control [41]. Furthermore the complication of equation solution is trimmed down by a modified performance algorithm. Yet another approach is the introduction of Toeplitz Hankel to predict the system future action and to split the input increments and the predictive output error variables. By employing linear relationship between the future output error and input increment based on the anticipated Toeplitz predictive equation, the online computation burden is reduced greatly by keeping away from the recursive solution of Diophantine equation [41].

In the thesis, Toeplitz Hankel matrices are used to formulate GPC mechanism. At first a brief review of Toeplitz Hankel matrices is presented in detail.

Toeplitz Hankel matrices are used to simplify much of the algebra related to MPC. They are simple to define. Let's consider a standard polynomial $v(z)$, where

$$v(q) = v_0 + v_1(q^{-1}) + ... + v_{v0}(q^{-v0}) \tag{4.34}$$

Now Toeplitz matrices $\acute{\Gamma}_v$, and C_v for $v(z)$ are defined from the following stripped matrix:

$$\Gamma_v = \frac{C_v}{M_v} = \left[\begin{array}{ccccc} v_0 & 0 & 0 & 0 & 0 \\ v_1 & v_0 & 0 & 0 & 0 \\ v_2 & v_1 & v_0 & 0 & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{v0} & v_{v0-1} & v_{v0-2} & \cdots & v_0 \\ \hline 0 & v_{v0} & v_{v0-1} & v_{v0-2} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & v_0 \end{array}\right] \tag{4.35}$$

and the Hankel matrix H_v is defined as:

$$H_v = \begin{bmatrix} v_1 & v_2 & v_3 & \cdots & v_{v0-1} & v_1 \\ v_2 & v_3 & v_4 & \cdots & v_{v0} & 0 \\ v_3 & v_4 & v_5 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{v0-1} & v_{v0} & 0 & 0 & \vdots & 0 \\ v_{v0} & 0 & 0 & 0 & \vdots & 0 \\ 0 & 0 & 0 & 0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{4.36}$$

The dimensions of $\acute{\Gamma}_v$ and H_v are as dictated by the context but following points should be taken care of:

- $\acute{\Gamma}_v$ can have any number of columns but it should have $v_0$more rows than the columns.
- C_v should be a square matrix with any dimension.
- H_v should have at least $v_0$non-zero rows and columns. More zero rows and/or columns can however be added without changing its operation.

From (4.35) and (4.36), it can be summarized that the elements of Toeplitz Hankel matrices can be defined as:

$$\begin{aligned} \acute{\Gamma}_v(i,j) &= C_v(i,j) = v_{i-j} \\ H_v(i,j) &= v_{i+j-1} \end{aligned}$$

assuming that $v_i = 0, \begin{cases} i < 0 \\ i > v_0 \end{cases}$

Polynomial Multiplication of Toeplitz Hankel Matrices

$\acute{\Gamma}_v$ is a trimmed matrix extracted from Toeplitz matrix C_v. It is trimmed such that it is tall and thin variant of the Toeplitz matrix. This matrix is used for changing polynomial convolution into a matrix-vector multiplication. Define:

$$\begin{aligned} e(q) &= e_0 + e_1(q^{-1}) + \ldots\ldots + e_{e0}(q^{-e0}) \\ f^{\circ}(q) &= v(q)e(q) = f^{\circ}{}_0 + f^{\circ}{}_1(q^{-1}) + \ldots\ldots + f^{\circ}{}_{e0+v0}(q^{-e0-v0}) \end{aligned} \tag{4.37}$$

It is obvious that the coefficients of $\mathring{f}(q)$ are given by:

$$\begin{bmatrix} f_0^o \\ f_1^o \\ f_2^o \\ \vdots \\ f_{v0}^o \\ f_{v0+1}^o \\ \vdots \\ f_{v+e0}^o \end{bmatrix} = \begin{bmatrix} v_0 & 0 & 0 & 0 & 0 \\ v_1 & v_0 & 0 & 0 & 0 \\ v_2 & v_1 & v_0 & 0 & : \\ : & : & : & : & : \\ v_{v0} & v_{v0-1} & v_{v0-2} & .. & v_0 \\ \hline 0 & v_{v0} & v_{v0-1} & .. & \\ : & : & : & : & : \\ : & : & : & : & : \\ : & : & : & : & : \\ 0 & 0 & 0 & .. & v_0 \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ e_2 \\ \vdots \\ e_{e0} \end{bmatrix} \tag{4.38}$$

In a more compact form (4.38) can be represented as:

$$\boldsymbol{f^o} = \boldsymbol{\acute{\Gamma}_v} + \mathbf{e}$$

Inversion of Toeplitz matrix

It is interesting to note that inversion for Toeplitz matrices has a physical meaning. It is given as:

$$[C_v]^{-1} = C_{\frac{1}{v}}$$

So if $\bar{\Delta} = 1-q^{-1}$ and $\frac{1}{\bar{\Delta}} = 1+q^{-1}+q^{-2}+\cdots$, then:

$$C_{\bar{\Delta}} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{4.39}$$

And its inverse is given as shown in (4.40):

$$[C_{\bar{\Delta}}]^{-1} = C_{\frac{1}{\bar{\Delta}}} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix} \tag{4.40}$$

Thus a relationship can be developed between quite complex matrix or vector multiplications and equivalent operations in transfer functions. For example:

$$\mathbf{f}^o = [C_v]^{-1} e \Rightarrow f^o(q) = \frac{e(q)}{v(q)} \tag{4.41}$$

In (4.41), it is assumed that the matrix dimensions are always endured to fit the context.

Commutativity of Toeplitz matrices

If multiplication of a polynomial function is commutative then its Toeplitz operation is also commutative [41].

For example:

$$C_n C_m \mathbf{a} = C_m C_n \mathbf{a} \qquad \text{etc.}$$

4.7.2.3 Prediction with transfer function models – using Toeplitz Hankel Matrices

The first step is to introduce offset free tracking in the model. It is a usual practice to incorporate integral action in the model of MPC and use the *augmented model* [41] in control scheme. The basic aim of introducing integral action in an MPC control law is to achieve a consistent and correct estimate of the expected steady-state value of the input such that offset free tracking of the response can be achieved.

An augmented form of (4.16) is developed by relating the outputs to the control increments such that:

$$\bar{\Delta} u_t = u_t - u_{t-1}$$

This is done by multiplying (4.16) by $\overline{\Delta}$. Thus

$$(\overline{A}(q^{-1})\overline{\Delta}(q))y(t) = \overline{B}(q^{-1})\overline{\Delta}u_t + \xi(t) \tag{4.42}$$

Thus the consistent estimates of the states required to give offset free tracking in the steady-state are $y = r$, $\bar{\Delta}u_t = 0$.

Note also that the only unknown in (4.42) is $\xi(t)$, which is zero mean noise hence it can be assumed to be equal to zero for future predictions and will not affect them.

After implementing the augmented form of CARIMA model for offset free prediction in the steady state and removing the effects of $\xi(t)$, the system is now described by:

$$\begin{aligned} \overline{\overline{A}}(q^{-1}) &= 1 + \overline{\overline{a}}_1 q^{-1} + \ldots\ldots + \overline{\overline{a}}_{n+1} q^{-n-1} \\ \overline{B}(q^{-1}) &= \overline{b}_1 q^{-1} + \overline{a}_1 q^{-1} + \ldots\ldots + \overline{b}_n q^{-n} \end{aligned} \tag{4.43}$$

Where $\overline{\overline{A}} = \overline{A}(q^{-1})\overline{\Delta}(q)$

j step ahead prediction

In this case, *j* step ahead predictions are computed by recursive use of (4.42)after deriving it for the future predicted output and incorporating (4.43) in it such that:

$$\overline{\overline{A}}(q^{-1})y(t) = \overline{B}(q^{-1})\overline{\Delta}u_t \tag{4.44}$$

This appears to be simple and computationally very efficient. However, the algebra can be cumbersome and hence is not the simplest procedure.

This is the time to incorporate Toeplitz Hankel matrices in the system to ease the computations. This gives an efficient, compact, insightful and nice approach to the solution. The flow of the procedure is detailed below:

Step 1: Prepare the predicted output equations for next j predictions

Equation (4.44) is expanded using (4.43) as shown below:

$$
\begin{aligned}
&(1+\bar{\bar{a}}_1 q^{-1}+......+\bar{\bar{a}}_{n+1} q^{-n-1})y(t)=(\bar{b}_1 q^{-1}+\bar{b}_2 q^{-2}+......+\bar{b}_n q^{-n})\bar{\Delta} u_t \\
&y(t)+\bar{\bar{a}}_1 y(t-1)+.....+\bar{\bar{a}}_n y(t-n)=\bar{b}_1\bar{\Delta} u(t-1)+\bar{b}_2\bar{\Delta} u(t-2)+....+\bar{b}_n\bar{\Delta} u(t-n) \\
&y(t)+\bar{\bar{a}}_1 y(t-1)+.....+\bar{\bar{a}}_n y(t-n)=\bar{b}_1\bar{\Delta} u(t-1)+\bar{b}_2\bar{\Delta} u(t-2)+....+\bar{b}_n\bar{\Delta} u(t-n)
\end{aligned}
$$

Moving one step ahead

$$
y(t+1)+\bar{\bar{a}}_1 y(t)+\cdots+\bar{\bar{a}}_n y(t-(n+1))=\bar{b}_1\bar{\Delta} u(t)+\bar{b}_2\bar{\Delta} u(t-1)+\cdots+\bar{b}_n\bar{\Delta} u(t-(n+1))
$$

Similarly after *j* steps ahead

$$
\begin{aligned}
y(t+j)+\bar{\bar{a}}_1 y(t+(j-1))+.....+\bar{\bar{a}}_n y(t-(n+j)) &= \bar{b}_1\bar{\Delta} u(t+(j-1))+\bar{b}_2\bar{\Delta} u(t+(j-1)) \\
&\quad +....+\bar{b}_n\bar{\Delta} u(t-(n+j))
\end{aligned}
\tag{4.45}
$$

Step 2: Prediction Equations in matrices form

Transforming the prediction equations in matrices form, we get:

$$
\begin{aligned}
&\begin{bmatrix} 1 & \bar{\bar{a}}_1 & \cdots & \bar{\bar{a}}_n \\ \bar{\bar{a}}_1 & 1 & \cdots & 0 \\ \bar{\bar{a}}_2 & \bar{\bar{a}}_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}
\begin{bmatrix} y(t+1) \\ y(t+2) \\ \vdots \\ y(t+j) \end{bmatrix}
+
\begin{bmatrix} \bar{\bar{a}}_1 & \bar{\bar{a}}_2 & \cdots & \bar{\bar{a}}_{n+1} \\ \bar{\bar{a}}_2 & \bar{\bar{a}}_3 & \cdots & 0 \\ \bar{\bar{a}}_3 & \bar{\bar{a}}_4 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}
\begin{bmatrix} y(t) \\ y(t-1) \\ \vdots \\ y(t-(n-j-1) \end{bmatrix} \\
&=
\begin{bmatrix} \bar{b}_1 & 0 & \cdots & 0 \\ \bar{b}_2 & \bar{b}_2 & \cdots & 0 \\ \bar{b}_2 & \bar{b}_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}
\begin{bmatrix} \bar{\Delta} u(t) \\ \bar{\Delta} u(t+1) \\ \vdots \\ \bar{\Delta} u(t+(j-1)) \end{bmatrix}
+
\begin{bmatrix} \bar{b}_2 & \bar{b}_3 & \cdots & \bar{b}_3 \\ \bar{b}_3 & \bar{b}_4 & \cdots & 0 \\ \bar{b}_4 & \bar{b}_5 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}
\begin{bmatrix} \bar{\Delta} u(t-1) \\ \bar{\Delta} u(t-2) \\ \vdots \\ \bar{\Delta} u(t-n+1) \end{bmatrix}
\end{aligned}
\tag{4.46}
$$

Step 3: Representation in terms of Toeplitz Hankel notation

Equation (4.46) can be written in compact form using Toeplitz Hankel thus:

$$
C_{\bar{\bar{A}}}\underset{\rightarrow}{y}_t + H_{\bar{\bar{A}}}\underset{\leftarrow}{y}_t = C_{q\bar{B}}\bar{\Delta}\underset{\rightarrow}{u}_t + H_{q\bar{B}}\bar{\Delta}\underset{\leftarrow}{u}_t
\tag{4.47}
$$

Step 4: Determining the output predictions

Finally the output predictions are determined from (4.47) as shown below:

$$\underset{\rightarrow}{y}_t = (C_{\bar{\bar{A}}})^{-1}(C_{q\bar{B}}\overline{\Delta}\underset{\rightarrow}{u}_t + H_{q\bar{B}}\overline{\Delta}\underset{\leftarrow}{u}_t - H_{\bar{\bar{A}}}\underset{\leftarrow}{y}_t) \tag{4.48}$$

In a more compact form, (4.48) can be written as:

$$\underset{\rightarrow}{y}_t = (\tilde{H}\overline{\Delta}\underset{\rightarrow}{u}_t + \tilde{\tilde{P}}\overline{\Delta}\underset{\leftarrow}{u}_t - \tilde{Q}\underset{\leftarrow}{y}_t) \tag{4.49}$$

where

$$\tilde{H} = (C_{\bar{\bar{A}}})^{-1}C_{q\bar{B}}$$
$$\tilde{\tilde{P}} = (C_{\bar{\bar{A}}})^{-1}H_{q\bar{B}}$$
$$\tilde{Q} = (C_{\bar{\bar{A}}})^{-1}H_{\bar{\bar{A}}}$$

If matrix algebra is unsupported in some application then, recursion of $\tilde{H}$, $\tilde{\tilde{P}}$, and $\tilde{Q}$ polynomials can be used instead. It will have same computational complexity as that of the matrix approach given earlier. However approach of recursion is not recommended if matrix algebra is supported by the application [41].

4.7.2.4 Performance Index and Optimization

The purpose of the predictive control system is to bring the predicted output as close as possible to the set point signal, where it is supposed that the set point signal remains unchanged within the optimization window. This objective is converted into a design framework to determine the best control parameter vector $\Delta\boldsymbol{u}$ (where $\Delta\boldsymbol{u}$ = [$\Delta u_1 \Delta u_2 \Delta u_3 .... \Delta u_{Hc}$]) such that an error function between the set point and the predicted output is minimized. This design is usually a 2-norm measure of predicted performance. A typical performance index, usually known as the objective function or the cost function is given in (4.50).

$$J = \sum_{i=H_{p1}}^{H_{p2}} \| \mathbf{r}_{t+i} - \hat{\mathbf{y}}_{t+i} \|_2^2 + \lambda \sum_{i=0}^{H_c - 1} \| \Delta \mathbf{u}_{t+i} \|_2^2$$
$$J = \sum_{i=H_{p1}}^{H_{p2}} \| \mathbf{e}_{t+i} \|_2^2 + \lambda \sum_{i=0}^{H_c - 1} \| \Delta \mathbf{u}_{t+i} \|_2^2 \qquad (4.50)$$

where

$$\Delta \mathbf{u}_{t+i|t} = 0 \; for \; i \geq H_c \qquad (4.51)$$

which states that the squares of the predicted tracking errors are added from an initial prediction horizon H_{p1} to maximum prediction horizon H_{p2} and the squares of the control changes are added over the control horizon H_c with some weighting factor λ. Both these sums are added to define the cost function to be minimized.

In the objective function, given in (4.50), the first term relates to the objective of minimizing the errors between the predicted output and the set point signal and the second term reveals the importance given to the size of ΔU when the objective function J is made to be as small as possible. Minimum performance index (J = 0) implies offset free tracking because J = 0 only if both the errors of reference following and control action change are nullified.

i.e. $e(j) = w(t+j) - \hat{y}(t+j|t)$ and $\Delta u_k = 0$

This ensures that the predicted output is following the set point exactly and there is no change in control action.

To calculate the optimal control sequence that will minimize J, the objective function is expanded and its first derivative is determined with respect to the control sequence. The derivative is then equated with zero because the necessary condition for minimum J *is* that its first derivative, with respect to the function to be optimized, should be zero. This methodology generates an optimized sequence of control actions. The procedure is described below:

First of all consider a suitable cost function such as given in (4.50):

$$J = \sum_{i=H_{p1}}^{H_{p2}} \| \mathbf{r}_{t+i} - \hat{\mathbf{y}}_{t+i} \|_2^2 + \lambda \sum_{i=0}^{H_c-1} \|\Delta \mathbf{u}_{t+i} \|_2^2$$

Now we'll work over the minimization of cost function. For convenience it is assumed that there is just one step ahead prediction. Moreover future and past sequences will be indicated by lower arrows hence (4.50)can be written as:

$$J = \| \underrightarrow{\mathbf{r}} - \underrightarrow{\hat{\mathbf{y}}} \|_2^2 + \lambda \, \| \Delta \underrightarrow{\mathbf{u}} \|_2^2$$

It may also be written as:

$$J = (\underrightarrow{\mathbf{r}} - \underrightarrow{\hat{\mathbf{y}}})^T (\underrightarrow{\mathbf{r}} - \underrightarrow{\hat{\mathbf{y}}}) + \lambda (\Delta \underrightarrow{\mathbf{u}})^T (\Delta \underrightarrow{\mathbf{u}})$$

Now solving the cost function expression

$$\begin{aligned}
&J = (\underrightarrow{\mathbf{r}}^T - \underrightarrow{\hat{\mathbf{y}}}^T)(\underrightarrow{\mathbf{r}} - \underrightarrow{\hat{\mathbf{y}}}) + \lambda (\mathbf{\Delta} \underrightarrow{\mathbf{u}}^T . \mathbf{\Delta} \underrightarrow{\mathbf{u}}) \\
&J = (\underrightarrow{\mathbf{r}}^T \underrightarrow{\mathbf{r}} - \underrightarrow{\mathbf{r}}^T \underrightarrow{\hat{\mathbf{y}}} - \underrightarrow{\hat{\mathbf{y}}}^T \underrightarrow{\mathbf{r}} + \underrightarrow{\hat{\mathbf{y}}}^T \underrightarrow{\hat{\mathbf{y}}}) + \lambda (\mathbf{\Delta} \underrightarrow{\mathbf{u}}^T . \mathbf{\Delta} \underrightarrow{\mathbf{u}}) \\
&\because a^T b = b^T a \\
&J = (\underrightarrow{\mathbf{r}}^T \underrightarrow{\mathbf{r}} - \underrightarrow{\mathbf{r}}^T \underrightarrow{\hat{\mathbf{y}}} - \underrightarrow{\mathbf{r}}^T \underrightarrow{\hat{\mathbf{y}}} + \underrightarrow{\hat{\mathbf{y}}}^T \underrightarrow{\hat{\mathbf{y}}}) + \lambda (\mathbf{\Delta} \underrightarrow{\mathbf{u}}^T . \mathbf{\Delta} \underrightarrow{\mathbf{u}}) \\
&J = (\underrightarrow{\mathbf{r}}^T \underrightarrow{\mathbf{r}} - 2 \underrightarrow{\mathbf{r}}^T \underrightarrow{\hat{\mathbf{y}}} + \underrightarrow{\hat{\mathbf{y}}}^T \underrightarrow{\hat{\mathbf{y}}}) + \lambda (\mathbf{\Delta} \underrightarrow{\mathbf{u}}^T . \mathbf{\Delta} \underrightarrow{\mathbf{u}})
\end{aligned}$$

$$\therefore \hat{y} = \underrightarrow{y}_t = \tilde{H} \overline{\Delta} \underrightarrow{u}_t + \tilde{\tilde{P}} \overline{\Delta} \underleftarrow{u}_t - \tilde{Q} \underleftarrow{y}_t$$

Substituting the value of predicted output $\hat{y}$ we get:

$$\begin{aligned}
J = {} & (\underrightarrow{\mathbf{r}}^T \underrightarrow{\mathbf{r}} - 2 \underrightarrow{\mathbf{r}}^T (\tilde{H} \mathbf{\Delta} \underrightarrow{\mathbf{u}}_{\mathbf{t}} + \tilde{\tilde{P}} \mathbf{\Delta} \underleftarrow{\mathbf{u}}_{\mathbf{t}} - \tilde{Q} \underleftarrow{\mathbf{y}}_{\mathbf{t}}) + (\tilde{H} \mathbf{\Delta} \underrightarrow{\mathbf{u}}_{\mathbf{t}} + \tilde{\tilde{P}} \mathbf{\Delta} \underleftarrow{\mathbf{u}}_{\mathbf{t}} - \tilde{Q} \underleftarrow{\mathbf{y}}_{\mathbf{t}})^T (\tilde{H} \mathbf{\Delta} \underrightarrow{\mathbf{u}}_{\mathbf{t}} + \tilde{\tilde{P}} \mathbf{\Delta} \underleftarrow{\mathbf{u}}_{\mathbf{t}} - \tilde{Q} \underleftarrow{\mathbf{y}}_{\mathbf{t}})) \\
& + \lambda (\mathbf{\Delta} \underrightarrow{\mathbf{u}}^T . \mathbf{\Delta} \underrightarrow{\mathbf{u}}) \\
J = {} & (\underrightarrow{\mathbf{r}}^T \underrightarrow{\mathbf{r}} - 2 \underrightarrow{\mathbf{r}}^T \tilde{H} \mathbf{\Delta} \underrightarrow{\mathbf{u}}_{\mathbf{t}} - 2 \underrightarrow{\mathbf{r}}^T \tilde{\tilde{P}} \mathbf{\Delta} \underleftarrow{\mathbf{u}}_{\mathbf{t}} + 2 \underrightarrow{\mathbf{r}}^T \tilde{Q} \underleftarrow{\mathbf{y}}_{\mathbf{t}} - (\mathbf{\Delta} \underrightarrow{\mathbf{u}}_{\mathbf{t}}{}^T \tilde{H}^T + \mathbf{\Delta} \underleftarrow{\mathbf{u}}_{\mathbf{t}}{}^T \tilde{\tilde{P}}^T - \underleftarrow{\mathbf{y}}_{\mathbf{t}}{}^T \tilde{Q}^T) \underrightarrow{\mathbf{r}} \\
& + (\mathbf{\Delta} \underrightarrow{\mathbf{u}}_{\mathbf{t}}{}^T \tilde{H}^T + \mathbf{\Delta} \underleftarrow{\mathbf{u}}_{\mathbf{t}}{}^T \tilde{\tilde{P}}^T - \underleftarrow{\mathbf{y}}_{\mathbf{t}}{}^T \tilde{Q}^T)(\tilde{H} \mathbf{\Delta} \underrightarrow{\mathbf{u}}_{\mathbf{t}} + \tilde{\tilde{P}} \mathbf{\Delta} \underleftarrow{\mathbf{u}}_{\mathbf{t}} - \tilde{Q} \underleftarrow{\mathbf{y}}_{\mathbf{t}})) + \lambda (\mathbf{\Delta} \underrightarrow{\mathbf{u}}^T . \mathbf{\Delta} \underrightarrow{\mathbf{u}})
\end{aligned}$$

Neglecting the terms independent of future control actions, we get:

$$J=-2\underset{\rightarrow}{\mathbf{r}}^T\tilde{H}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}+(\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T+\mathbf{\Delta}\underset{\leftarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{\tilde{P}}^T-\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}^T\tilde{Q}^T)(\tilde{H}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}+\tilde{\tilde{P}}\mathbf{\Delta}\underset{\leftarrow}{\mathbf{u}}_{\mathbf{t}}-\tilde{Q}\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}))+\lambda(\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}^T.\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}})$$

$$J=-2\underset{\rightarrow}{\mathbf{r}}^T\tilde{H}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}+(\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T\tilde{H}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}+\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T\tilde{\tilde{P}}\overline{\mathbf{\Delta}}\underset{\leftarrow}{\mathbf{u}}-\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T\tilde{Q}\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}+\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{\tilde{P}}^T\tilde{H}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}-\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}^T\tilde{Q}^T\tilde{H}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}))$$
$$+\lambda(\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}^T.\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}})$$

$$J=-2\underset{\rightarrow}{\mathbf{r}}^T\tilde{H}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}+2\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^{\mathbf{T}}\tilde{H}^T\tilde{\tilde{P}}\mathbf{\Delta}\underset{\leftarrow}{\mathbf{u}}-2\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T\tilde{Q}\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}+(\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T(\tilde{H}^T\tilde{H}+\lambda I)\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}})$$

$$J=\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T(\tilde{H}^T\tilde{H}+\lambda I)\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}-2\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T\underset{\rightarrow}{\mathbf{r}}+2\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T\tilde{\tilde{P}}\overline{\mathbf{\Delta}}\underset{\leftarrow}{\mathbf{u}}-2\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T\tilde{Q}\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}$$

Finally the cost function to be minimized w.r.t the future control action is:

$$J=\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T(\tilde{H}^T\tilde{H}+\lambda I)\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}+2\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underset{\leftarrow}{\mathbf{u}}-\tilde{Q}\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}-\underset{\rightarrow}{\mathbf{r}}) \tag{4.52}$$

Now we have to determine:

$$\min(J)=\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T(\tilde{H}^T\tilde{H}+\lambda I)\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}+2\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underset{\leftarrow}{\mathbf{u}}-\tilde{Q}\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}-\underset{\rightarrow}{\mathbf{r}}) \tag{4.53}$$

It may also be written as:

$$\min(J)=\Delta\underset{\rightarrow}{u}_t^T\mathbf{S}\,\Delta\underset{\rightarrow}{u}+2\Delta\underset{\rightarrow}{u}_t^T\mathbf{f} \tag{4.54}$$

where:

$$\mathbf{S}=\tilde{H}^T\tilde{H}+\lambda I$$
$$\mathbf{f}=\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underset{\leftarrow}{\mathbf{u}}-\tilde{Q}\underset{\leftarrow}{\mathbf{y}}_{\mathbf{t}}-\underset{\rightarrow}{\mathbf{r}})$$

Since we desire min(*J*) = 0, (4.54) can therefore be written in standard form (using the fact that $a^Tb=b^Ta$) as:

$$\min(J)=\frac{1}{2}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}}^T\mathbf{S}\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}+\mathbf{f}^T\mathbf{\Delta}\underset{\rightarrow}{\mathbf{u}}_{\mathbf{t}} \tag{4.55}$$

Now (4.55) can be minimized by taking the first derivative of min(*J*) and equating it with zero. The mathematical flow is given below:

Starting from (4.53):

$$\min(J) = \mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}(\tilde{H}^T\tilde{H}+\lambda I)\mathbf{\Delta}\underrightarrow{\mathbf{u}} + 2\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{Q}\underleftarrow{\mathbf{y}}_{\mathbf{t}} - \underrightarrow{\mathbf{r}})$$

$$\frac{\delta J}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}} = \frac{\delta(\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}(\tilde{H}^T\tilde{H}+\lambda I)\mathbf{\Delta}\underrightarrow{\mathbf{u}} + 2\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{Q}\underleftarrow{\mathbf{y}}_{\mathbf{t}} - \underrightarrow{\mathbf{r}}))}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}}$$

$$\frac{\delta J}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}} = \frac{\delta(\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}(\tilde{H}^T\tilde{H}+\lambda I)\mathbf{\Delta}\underrightarrow{\mathbf{u}}}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}} + \frac{2\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{Q}\underleftarrow{\mathbf{y}}_{\mathbf{t}} - \underrightarrow{\mathbf{r}}))}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}}$$

$$\frac{\delta J}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}} = ((\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}(\tilde{H}^T\tilde{H}+\lambda I) + \mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}(\tilde{H}^T\tilde{H}+\lambda I)^T) + 2(\mathbf{\Delta}\underleftarrow{\mathbf{u}}^{T}\tilde{\tilde{P}}^T\tilde{H} - \underleftarrow{\hat{\mathbf{y}}}_{\mathbf{t}}^{T}\tilde{Q}^T\tilde{H} - \underrightarrow{\mathbf{r}}^T\tilde{H}))$$

$$\therefore (\tilde{H}^T\tilde{H}+\lambda I)^T = (\tilde{H}^T\tilde{H}+\lambda I)$$

$$\frac{\delta J}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}} = (2\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}^{T}(\tilde{H}^T\tilde{H}+\lambda I)) + 2(\tilde{H}^T\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{H}^T\tilde{Q}\underleftarrow{\hat{\mathbf{y}}}_{\mathbf{t}} - \tilde{H}^T\underrightarrow{\mathbf{r}}))$$

$$\frac{\delta J}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}} = (2(\tilde{H}^T\tilde{H}+\lambda I)^T\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}) + 2(\tilde{H}^T\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{H}^T\tilde{Q}\underleftarrow{\hat{\mathbf{y}}}_{\mathbf{t}} - \tilde{H}^T\underrightarrow{\mathbf{r}}))$$

For the future control sequence with minimum cost, the necessary condition is that the ration between the partial derivative of the cost function and the partial derivative of the change in future control action should be minimum i.e. zero.

In the following this concept is derived mathematically:

$$\frac{\delta J}{\delta\mathbf{\Delta}\underrightarrow{\mathbf{u}}} = 0$$

$$\Rightarrow (2(\tilde{H}^T\tilde{H}+\lambda I)^T\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}) + 2(\tilde{H}^T\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{H}^T\tilde{Q}\underleftarrow{\hat{\mathbf{y}}}_{\mathbf{t}} - \tilde{H}^T\underrightarrow{\mathbf{r}})) = 0$$

$$\Rightarrow (2(\tilde{H}^T\tilde{H}+\lambda I)^T\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}}) - 2\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{Q}\underleftarrow{\hat{\mathbf{y}}}_{\mathbf{t}} - \underrightarrow{\mathbf{r}})) = 0$$

$$\Rightarrow (\tilde{H}^T\tilde{H}+\lambda I)^T\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}} = \tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{Q}\underleftarrow{\hat{\mathbf{y}}}_{\mathbf{t}} - \underrightarrow{\mathbf{r}})$$

Finally the optimized sequence of control action is:

$$\mathbf{\Delta}\underrightarrow{\mathbf{u}}_{\mathbf{t}} = (\tilde{H}^T\tilde{H}+\lambda I)^{-1}\tilde{H}^T(\tilde{\tilde{P}}\mathbf{\Delta}\underleftarrow{\mathbf{u}} - \tilde{Q}\underleftarrow{\hat{\mathbf{y}}}_{\mathbf{t}} - \underrightarrow{\mathbf{r}}) \tag{4.56}$$

From optimized future control sequence, $\Delta\boldsymbol{u}$, the first element, Δu_k, is implemented as a control action and the optimization is repeated and updated at each sampling instant.

Limitations on the future control trajectory

It is mandatory to take care of the restriction (4.51) which is placed on the class of future input trajectories. This limitation has noteworthy consequences on the structure of the prediction equations. The Toeplitz matrix $\tilde{H}$ multiplies upon the sequence of future control actions Δ**u**→. Originally *H* is defined as square matrix and it is multiplied with all the future sequence of control moves. Thus, the prediction equations presume that:

$$\tilde{H}\Delta\underset{\rightarrow}{\mathbf{u}} = \left[\begin{array}{cccccc|cc} h_0 & 0 & 0 & \cdots & 0 & & \cdots & 0 \\ h_1 & h_0 & 0 & \vdots & \vdots & & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ h_{(H_{p2}-H_{p1})-1} & h_{(H_{p2}-H_{p1})-2} & h_{(H_{p2}-H_{p1})-3} & \cdots & h_{(H_{p2}-H_{p1})-H_c} & & \cdots & h_0 \end{array}\right] \left[\begin{array}{c} \Delta u_{k|k} \\ \Delta u_{k+1|k} \\ \vdots \\ \Delta u_{k+(H_c-1)|k} \\ \hline \vdots \\ \Delta u_{k+(H_{p2}-H_{p1}-1)|k} \end{array}\right] \tag{4.57}$$

However, the limitation in (4.51), that is $\Delta u_{k+Hc+i|k}= 0,\ i \geq 0$, implies that:

$$\tilde{H}_{th}\Delta\underset{\rightarrow}{\mathbf{u}} = \begin{bmatrix} h_0 & 0 & \cdots & \cdots & \cdots & 0 \\ h_1 & h_0 & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{(H_{p2}-H_{p1})-1} & h_{(H_{p2}-H_{p1})-2} & h_{(H_{p2}-H_{p1})-3} & \cdots & \cdots & h_{(H_{p2}-H_{p1})-H_c} \end{bmatrix} \begin{bmatrix} \Delta u_{k|k} \\ \Delta u_{k+1|k} \\ \vdots \\ \Delta u_{k+(H_c-1)|k} \end{bmatrix} \tag{4.58}$$

Where $\tilde{H}_{th}$ constitutes only the first H_c columns of $\tilde{H}$.

Receding Horizon Control

Optimization in MPC algorithms follows the receding horizon strategy. In this strategy, following tasks are performed at each sample instant *t*:

- The set-point sequence $w(t + j)$ is calculated for the future.
- The predicted model of plant is employed to generate a set of predicted outputs

$\hat{y}(t+j|t)$ with corresponding reference tracking errors ($e(t+j) = w(t+j)-\hat{y}(t+j|t)$) by taking care that $\hat{y}(t+j|t)$ for $j > k_d$ (where k_d is plant's dead time) depends in part on future control signals $u(t + j)$.

- A suitable quadratic function of the future errors and control actions is to be minimized, by assuming that after some "control horizon" further increments in control action are zero to provide a proposed sequence of future controls $u(t + j)$;
- The first member of the sequence Δu_k is maintained and the appropriate data vectors are shifted one step ahead so that the calculations can be repeated at the next sample instant.

4.7.2.5 Unconstrained GPC algorithm for transfer function models

Finally the formulation of unconstrained GPC algorithm is summarized step wise.

Step 1: First of all, the cost function is formulated in vector form as shown in (4.50).

Step 2: Future predicted outputs are then substituted from (4.49) into (4.50).

To cope up the limitation of future control trajectory defined in (4.51), $\widetilde{H}$is developed as tall and thin matrix.

Step 3: The required minimization is defined after expanding(4.50) to obtain (4.53)

Step 4: The *performance index* is always positive because it is quadratic function hence a unique minimum which can be located by setting the first derivative of (4.53)to zero.

Step 5: Finally the GPC control law is obtained by the extracting the first element of control sequence obtained in (4.56).

4.7.3 Constrained GPC

Practical systems usually have certain constraints associated with them. The problem of constraints handling is *systematically* handled in MPC due to its feature of optimization [48]. The GPC algorithms described above are however unable to control the plant in the absence of constraints. Different approaches are later on presented to control a constrained problem using GPC. These include the works of Clarke *et al* [9], Rossitor [31] and Miguel Martinez *et al* [41].

In the following, constrained GPC algorithm will be discussed after a brief review to system constraints.

4.7.3.1 Frequently encountered operational constraints

Practical applications usually encounter three major categories of constraints:

- Constraints on change in input control action
- Constraints in amplitude
- Constraints on output

Constraints Formulation

In this section we will discuss how to formulate the three major types of constraints mentioned above. After formulating the constraints, the next phase is to convert them into linear inequalities and associate them to the predictive control problem. The basic idea is to parameterize the constrained variables using the sequence of future control actions $\Delta \mathbf{u_t}$. The constraints are expressed in a set of linear equations based on the parameter vector, $\Delta \mathbf{u_t}$, and are considered in each moving horizon window because of the utilization of receding horizon strategy for optimization in MPC. This ultimately constitutes a powerful methodology which allows adapting the constraints in each optimization window. The general form for representing inequality constraints is:

$$M\Delta\mathbf{u} \leq \gamma \tag{4.59}$$

Where:

M: Matrix indicating constraints. Its dimension is (*no. of constraints* x *no. of future control actions*)

γ : Vector indicating constraints values and limits

4.7.3.1.1 Constraints on change in control action

The rate of change of control variables usually encounters hard constraints. Assume that for a SISO system, the upper limit is Δu_{max} and the lower limit is Δu_{min}.

$$\Delta u_{\min} \leq \Delta u_k \leq \Delta u_{\max}$$

The rate of change constraints can be used to impose directional movement constraints on the control variables; for example if $u(k)$ can only increase not decrease then possibly we select $0 \leq \Delta u_k \leq \Delta u_{max}$. The rate constraints can be used to cope with the cases where the rate of change of control amplitude is restricted or limited in value.

Given that the input increments are predicted to be zero beyond the horizon H_c, one can check the constraints up to then, using the following:

$$\begin{bmatrix} \Delta u_{\min} \\ \Delta u_{\min} \\ \vdots \\ \Delta u_{\min} \end{bmatrix} \leq \begin{bmatrix} \Delta u_k \\ \Delta u_{k-1} \\ \vdots \\ \Delta u_{k+H_c-1} \end{bmatrix} \leq \begin{bmatrix} \Delta u_{\max} \\ \Delta u_{\max} \\ \vdots \\ \Delta u_{\max} \end{bmatrix} \tag{4.60}$$

Equation (4.60) can be represented as a single set of linear inequalities to give a more predictable form i.e.:

$$\begin{bmatrix} I \\ -I \end{bmatrix} \Delta \underset{\rightarrow}{u} - \begin{bmatrix} \mathbf{\Delta u}_{\max} \\ \mathbf{\Delta u}_{\min} \end{bmatrix} \leq 0 \tag{4.61}$$

Where

- I represents a suitable dimension identity matrix.

4.7.3.1.2 Constraints on amplitude of control action

For the formulation of input constraints, the future input sequence must first be expressed in terms of future control moves $\Delta \underset{\rightarrow}{u}$ such that:

$$\begin{aligned} \underset{\rightarrow}{\mathbf{u}} &= C_{I/\Delta} \Delta \underset{\rightarrow}{\mathbf{u}} + \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix} \mathbf{u}_{k-1} \\ \underset{\rightarrow}{\mathbf{u}} &= C_{I/\Delta} \Delta \underset{\rightarrow}{\mathbf{u}} + L \mathbf{u}_{k-1} \end{aligned} \tag{4.62}$$

Where

- $L = \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix}$

As discussed earlier $C_{I/\Delta}$ is a Toeplitz matrix which depends upon $1/(1-q^{-1})$ therefore it is lower triangular and contains identity matrices to fill the lower triangular part. The upper and lower limits for the inputs are defined as:

$$u^{min} \leq u_k \leq u^{max}$$

In matrices form, it may be written as:

$$\begin{bmatrix} u_{\min} \\ u_{\min} \\ \vdots \\ u_{\min} \end{bmatrix} \leq \begin{bmatrix} u_k \\ u_{k-1} \\ \vdots \\ u_{k+H_c-1} \end{bmatrix} \leq \begin{bmatrix} u_{\max} \\ u_{\max} \\ \vdots \\ u_{\max} \end{bmatrix} \tag{4.63}$$

The constraints can be satisfied over the control horizon H_c only but this will be sufficient for the predictions thereafter because for the predictions, it is assumed that $\mathbf{u}_{k+Hc+i} = \mathbf{u}_{k+Hc-1} \forall i \geq 0$

Hence by substituting the relation from (4.62), we get:

$$\mathbf{u}_{\min} \leq C_{I/\Delta} \Delta \underset{\rightarrow}{u} + Lu_{k-1} \leq \mathbf{u}_{\max} \tag{4.64}$$

Typically (4.64) can be written as:

$$\begin{bmatrix} C_{I/\Delta} \\ -C_{I/\Delta} \end{bmatrix} \Delta \underset{\rightarrow}{u} - \begin{bmatrix} u_{\max} - Lu_{k-1} \\ -u_{\min} - Lu_{k-1} \end{bmatrix} \leq 0 \tag{4.65}$$

These constraints are usually experienced. For example it cannot be expected that a valve can open more than 100 percent nor a voltage level can be more or less than the specified range of device operation to which it has to be applied. These are defined as the physical hard constraints.

It should be concentrated that u_k is the incremental variable not the actual one. The original physical control variable is equal to the incremental variable u added with the steady state value $\mathrm{u_{ss}}$.

4.7.3.1.3 Constraints on output

Constraints can also be specified on the input to limit its operating range. These constraints are defined as:

$$y^{min} \leq y(t) \leq y^{max}$$

Output constraints are from the category of 'soft constraints' because a slack variable $s_{vs}>0$ is cascaded with the constraints such that:

$$y^{min} - s_{vs} \leq y(t) \leq y^{max} + s_{vs}$$

When output constraints are enforced, they usually bring about severe changes in the control as well as the incremental control variables. Therefore these variables can defy the constraints implemented upon them which give rise to the issue of conflicts in constraints. When the constraints on the control variables are mandatory for plant operation, the output constraints are slowed down by choosing a larger slack variable s_{vs}.

The linear inequalities related to output constraints are given as:

$$\begin{bmatrix} \tilde{H} \\ -\tilde{H} \end{bmatrix} \Delta \underrightarrow{u} - \begin{bmatrix} y_{\max} - \tilde{Q}\underleftarrow{\hat{y}} - \tilde{\tilde{P}}\Delta \underleftarrow{u} \\ -y_{\min} - \tilde{Q}\underleftarrow{\hat{y}} - \tilde{\tilde{P}}\Delta \underleftarrow{u} \end{bmatrix} \leq 0 \qquad (4.66)$$

The output constraints are influential up to the prediction horizon $H_{p2.}$

4.7.3.2 Optimization in the presence of constraints

Constraints may be defined as equality constraints and inequality constraints. In the following section, these constraints will be described in detail.

4.7.3.2.1 Constrained Minimization for equality constraints

The easiest challenge for quadratic programming is to determine the constrained minimum of a positive definite quadratic function with linear equality constraints. Linear equality constraints are defined by a hyper plane the level surfaces of positive definite quadratic functions are hyper ellipsoids. Instinctively the minimum constrained solution lies at the point of tangency between the boundary of the viable set and the hyper ellipsoid that is being minimized.

Lagrange Multipliers

For the minimization of objective functions which are subjected to equality constraints, Lagrange expression is considered:

$$J = \frac{1}{2}x^T\mathbf{S}x + x^T\mathbf{f} + \lambda^T(\mathbf{M}x - \gamma) \tag{4.67}$$

where S, f and M are matrices as defined above in the discussion while x and λrepresent the variables. Equation (4.67) is minimized by utilizing the necessary condition given in (4.68) and (4.69). The cost function has to be minimized w.r.t x as well as w.r.t λ.

$$\frac{\delta J}{\delta x} = \mathbf{S}x + \mathbf{f} + M^T\lambda = 0 \tag{4.68}$$

$$\frac{\delta J}{\delta \lambda} = Mx - \lambda = 0 \tag{4.69}$$

The components of vector λ are known as Lagrange multipliers.

The optimal variables λ and x are found after constrained optimization through the set of linear equations defined by (4.68) and (4.69) where:

$$\lambda = -(M\mathbf{S}^{-1}M^T)^{-1}(\gamma + M\mathbf{S}^{-1}\mathbf{f}) \tag{4.70}$$

$$x = -\mathbf{S}^{-1}(M^T\lambda + \mathbf{f}) \tag{4.71}$$

Substituting the value of (4.70) in (4.71) we get:

$$x = -S^{-1}f - S^{-1}M^T\lambda \tag{4.72}$$

It may also be written as:

$$x = x^o - S^{-1}M^T\lambda \tag{4.73}$$

In (4.73), The global optimal solution is represented by x^o that will determine minimum of the original cost function without constraints while the second term is an adjustment term due to equality constraints. Equality constraints should always be less in number or equal to the number of decision variables (*i.e. x*) [41].

4.7.3.2.2 Constrained Minimization for inequality constraints

When inequality constraints are minimized, the no. of constraints could be more than the number of decision variables. The inequality constraints $Mx \leq \lambda$ are comprised of active constraints and inactive constraints. An inequality is said to be active if $M_ix=\gamma_i$ and inactive if $M_ix<\gamma_i$. Kuhn –Trucker conditions are introduced to define the active and inactive constraints in terms of Lagrange multipliers.

Kuhn Tucker (KT) conditions

The essential conditions for the optimization problem of inequality constraints are:

$$\begin{aligned} &Sx + f + M^T\lambda = 0 \\ &Mx - \gamma \leq 0 \\ &\lambda^T(Mx - \gamma) = 0 \\ &\lambda \geq 0 \end{aligned} \tag{4.74}$$

where the vector λ contains the Lagrange multipliers. These *KT* conditions can be simplified by expressing them as the set of active constraints. If we mention the set of active constraints as C_{act} then the essential conditions are accordingly transformed as:

$$\mathbf{S}x + \mathbf{f} + \sum_{i \in C_{act}} \lambda_i M_i^T = 0$$

$$M_i x - \gamma_i = 0 \qquad i \in C_{act} \tag{4.75}$$

$$M_i x - \gamma_i < 0 \qquad i \notin C_{act} \tag{4.76}$$

$$\lambda_i \geq 0 \qquad i \in C_{act} \tag{4.77}$$

$$\lambda_i = 0 \qquad i \notin C_{act} \tag{4.78}$$

where: M_i is the i^{th} row of M matrix. Equation (4.75) states that for the i^{th} row,

- $M_i x$-γ_i=0 implies that it is an equality constraint which indicates that it is an active constraint. . For an active constraint, the Lagrange multiplier is positive.
- $M_i x$-γ_i<0 implies that the constraint is fulfilled; therefore it is an inactive constraint. The Lagrange multiplier is zero if the constraint is inactive.

Obviously if the designer knows the active set, the problem of inequality constraints can be changed by problem containing only the equality constraints.

Assume that M_{act} and λ_{act} are known then the closed form of the optimum solution with inequality constraints would be given as:

$$\lambda_{act} = -(M_{act}\mathbf{S}^{-1}M_{act}^T)^{-1}(\gamma_{act} + M_{act}\mathbf{S}^{-1}\mathbf{f}) \tag{4.79}$$

$$x = -\mathbf{S}^{-1}(\mathbf{f} + M_{act}^T\lambda_{act}) \tag{4.80}$$

4.7.3.2.3 Algorithm for constrained optimization in Predictive control…. Optimizing the change in future control actions

After defining in detail all the steps required in optimization, in the following a generalized algorithm presented. The algorithm outlines the procedure of constrained predictive control for the optimization of the rate of future control actions.

Step 1: First of all, operational limits are defined for the plant.

Step 2: Minimum and maximum parameter limits are then defined by incorporating steady state information

Step 3: Using the minimum and maximum limits, linear inequality constraints are defined.

Step 4: Finally the constrained optimization problem is solved using a quadratic programming procedure at every sampling instance to obtain the optimal solution of the sequence of future control actions.

4.8 The Proposed Control Scheme

Now the block diagram given in Fig 4.1 will be discussed in detail with respect to the proposed scheme. First of all the nonlinear system is identified as Hammerstein model. For this identification *Fuzzy Hammerstein* model is used with *Constrained Recursive Least Squares algorithm* so that both the linear and nonlinear parameters of the Hammerstein model can be identified. For simulation purpose, first ordered, delayed SISO systems are used but the delay is not being identified; however it is shown that GPC algorithm is able to control the delayed plant effectively without having any knowledge about the delay interval.

Side by side of the online identification, the parameters of linear dynamics are being applied to linear GPC algorithm without the information of delay. The general cost function has to be modified to adjust the inverse controller before the actual plant (as shown in Fig 4.1). In the following it is discussed in detail.

The general control law is given as:

$$J = \| \underset{\rightarrow}{\mathbf{r}} - \underset{\rightarrow}{\hat{\mathbf{y}}} \|_2^2 + \lambda \| \Delta \underset{\rightarrow}{\mathbf{u}} \|_2^2 \tag{4.81}$$

But in case of a Hammerstein model as given in Fig 1.1, we first need to cancel out the effect of static nonlinearity because linear GPC algorithm can control linear plants. This indicates that rather than u, the GPC algorithm has to produce v_d (as mentioned in Fig 1.1). This requires a modification in the control law.

From the Hammerstein model it is known that:

$$\Delta\mathbf{v} = K(u)\Delta\mathbf{u} \tag{4.82}$$

The control weighting coefficient λ is usually expressed as the product of a scaled control weighting coefficient λ' and square of the process (static nonlinearity) gain [22].

$$\lambda = \lambda'(K(u))^2 \tag{4.83}$$

From (4.82) and (4.83), we get:

$$\Rightarrow \lambda = \lambda'\left(\frac{\Delta\mathbf{v}}{\Delta\mathbf{u}}\right)^2$$
$$\Rightarrow \lambda\Delta\mathbf{u}^2 = \lambda'\Delta\mathbf{v}^2$$

Hence the Control law can be modified as follows:

$$J = \| \underset{\rightarrow}{\underline{\mathbf{r}}} - \underset{\rightarrow}{\hat{\underline{\mathbf{y}}}} \|_2^2 + \lambda \| \Delta\underset{\rightarrow}{\underline{\mathbf{u}}} \|_2^2$$

$$\Rightarrow J = (\underset{\rightarrow}{\underline{\mathbf{r}}} - \underset{\rightarrow}{\hat{\underline{\mathbf{y}}}})^2 + \lambda(\Delta\underset{\rightarrow}{\underline{\mathbf{u}}})^2$$

$$\Rightarrow J = (\underset{\rightarrow}{\underline{\mathbf{r}}} - \underset{\rightarrow}{\hat{\underline{\mathbf{y}}}})^2 + \lambda'(\Delta\underset{\rightarrow}{\underline{\mathbf{v}}})^2$$

The GPC algorithm is then developed by using the modified control law. Now to introduce the constraints on the control action, the constraints have to be mapped over the virtual control action rather than actual control action to be applied to plant (as defined in section 4.6.3.1.1 and 4.6.3.1.2) by using (4.84) and (4.85).

$$\Delta v(k) = f\left(\Delta\left(u(k)\right)\right) \tag{4.84}$$

$$\Delta v = K(u)\Delta u \tag{4.85}$$

Typically, the constraints are defined on the minimum and maximum of $u(k)$ and $\Delta u(k)$. However, these constraints have to be transformed into the domains of $v(k)$ and $\Delta v(k)$ through the nonlinear, but static relationship between $u(k)$ and $v(k)$. Keeping in view

(4.84) and (4.85), (4.59) will be modified as:

$$M\Delta\mathbf{v} \le \tilde{\gamma} \tag{4.86}$$

Thus we get:

$$\begin{bmatrix} C_I/_\Delta \\ -C_I/_\Delta \\ I \\ -I \end{bmatrix} \Delta v - \begin{bmatrix} v_{\max} - Lv_{k-1} \\ -v_{\min} - Lv_{k-1} \\ \Delta\mathbf{v}_{\max} \\ \Delta\mathbf{v}_{\min} \end{bmatrix} \le 0$$

where the elements of v_{max} can be calculated as $f(u_{max})$ and v_{min} by $f(u_{min})$ using the identified steady-state nonlinearity. The calculation of the elements of $\Delta\mathbf{v}_{\max}$ and $\Delta\mathbf{v}_{\min}$ is more difficult since they should be written as $\Delta\mathbf{v}_{\max} = v_{max}\mathrm{K}$ and $\Delta\mathbf{v}_{\min} = v_{min}\mathrm{K}$ where 'K' is a vector containing the gain of the model over the control horizon. Abonyi suggests that if the solution does not converge with this strategy then such a suboptimal solution can be guaranteed from the worst case analysis but it leads to a restrictive procedure. Hence Abonyi introduced an iterative algorithm that adaptively changes the constraints mapping. The convergence of this iterative procedure is guaranteed if the adaptive positive scalar is unity because in this case the linear inequality constraints mapping will converge to the worst case analysis given (4.86) which is proven to generate a feasible control solution.

However in the proposed scheme, the static gain $K(u(k))$ is determined online by making use of optimized $\Delta v(k)$, and the corresponding inverted $\Delta u(k)$ (i.e$u(k)$-$u(k$-1)). Utilizing this value of K($u(k)$) as positive for $\Delta\mathbf{v}_{\max}$ and negative for $\Delta\mathbf{v}_{\min}$, an optimal solution is always converged. The simulation results are shown in chapter 5.

The nonlinear parameter that is related to the static nonlinearity i.e. basically the consequent of fuzzy model is applied to the inverse model controller. The fuzzy sets of this inverse model have their cores at these consequents. This is done to ensure that the controller is adaptive with respect to the varying system. The inverse controller is designed using direct adaptive approach. The structure of inverse model is 0th order TS fuzzy model with feedback to ensure nullification of the static nonlinearity. The performance of inverse controller is also shown in chapter 5.

4.9 Chapter Summary

The chapter provides the details of the overall control methodology. Starting from the block diagram of the overall system, every node of the control scheme is dissected to get the in-depth knowledge of the control scheme. Fuzzy inverse model and Generalized Predictive control is discussed multi-dimensionally to provide a good insight of the technology.

CHAPTER 5

5 Simulation Results

5.1 Prologue

In the following chapter, the performance of the controller will be discussed in detail. Simulation results will be presented to show both the open and closed loop responses. The controller performance simulations will comprise of plant's response, the error of plant's response in reference following, the performance of inverse controller and the performance of GPC algorithm.

5.2 Performance of Control Scheme

In order to test the performance of the proposed control scheme, we considered three Hammerstein models (A, B and C), all having different nonlinearities and linear dynamics with respect to each other. For simplicity, first order SISO systems are used for simulation. The Fuzzy Hammerstein model as well as the adaptive inverse fuzzy Hammerstein model utilize 0^{th} order TS model. Both the fuzzy models are operated with six fuzzy sets for the input. The control parameters include those of:

- The Generalized Predictive controller i.e.
    - Control Horizon,
    - Prediction Horizon and
    - Control Weighting factor
- The adaptive inverse fuzzy model controller with feedback i.e.
    - The error gain (ξ)
- The supports of the fuzzy model and the inverse fuzzy model also affect the control performance

- The recursive least squares algorithm performance vary with the variation in
    - The initialization of Covariance matrix
    - The initial guess about the parameters to be converged
    - Forgetting factor (λ)

Furthermore, a test is conducted by varying the system parameters online and controller performance and its robustness is observed and the need of retuning in this scenario is investigated. The simulation results show that the scheme is able to control all these systems efficiently without having any prior knowledge about the system's static nonlinearity. Step and impulse response of the Open loop, uncontrolled systems A, B and C are first conducted to check how does the system responses for a sudden change in reference and for the effect of disturbance respectively. Then the controller is implemented on the system and the overall controlled output, the error in reference following, the change in control action generated by constrained GPC algorithm and the cancellation of static nonlinearity by Adaptive Fuzzy Inverse model controller is presented to show the performance of controller.

The GPC algorithm is kept bound to generate a change in virtual control action within ±0.1 units. The constraints for the system and the overall control methodology have already been discussed in chapter 4.

5.2.1 System A: Logarithmic Nonlinearity _ τ =120 in Linear Dynamics

Consider a Hammerstein model having logarithmic nonlinearity and a delayed dynamics with τ=120. A delay of 10s is also considered. The overall Hammerstein model is:

$$\frac{y}{u} = \frac{1}{3.43}\ln(30u+1)e^{-10s}\left(\frac{1}{120s+1}\right)$$

In order to observe the characteristics of the open loop system, we assume for a while that the Hammerstein model of the system is known and test its performance by the unit and impulse response.

The open loop response of the plant with unit step input is shown in Fig 5.1. The unit step response shows that the uncontrolled system offers a delay of approximately 10s. Rise time is approximately 288s. Settling time of the response approximately 508s and there is no overshoot in the system.

Then the uncontrolled system was tested with a train of pulses having amplitude of approximately 0.7 units. The response is shown in Fig 5.2.When the system is excited with a pulse train of 0.7 units in amplitude, the response of the system indicates 0.2 units increase in the output with no offset and overshoot. This indicates that the output of the plant is varying with varying set point and is not following it.

In order to observe the impact of disturbance on the system, a unit impulse is applied to it and the impulse response was observed as shown in Fig 5.3. It is evident here that with unit impulse, the disturbance amplitude will be approximately 0.8% only and it will completely settle down within 600s.

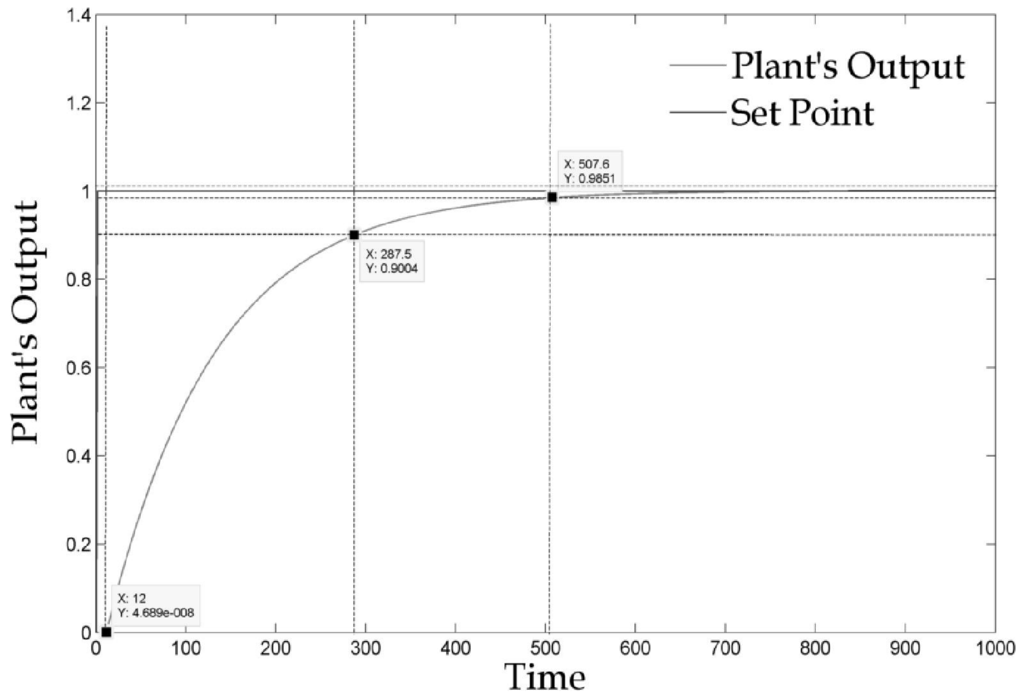

Fig 5.1: Step Response of the uncontrolled System A

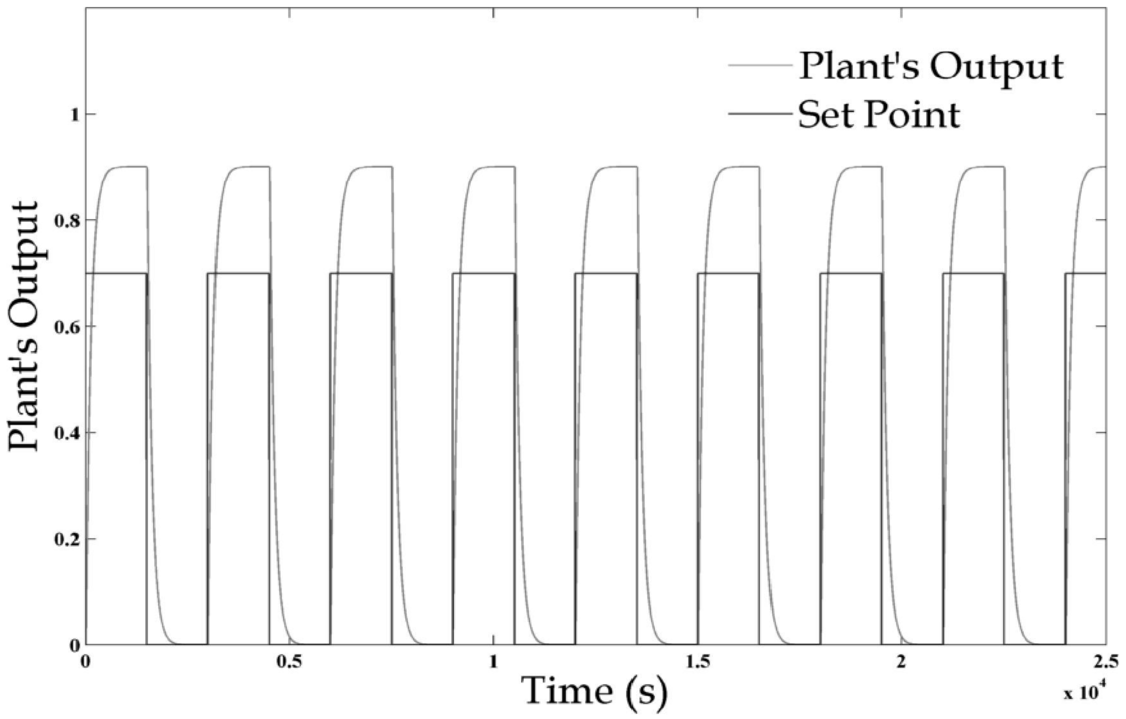

Fig 5.2: Uncontrolled behavior of system A

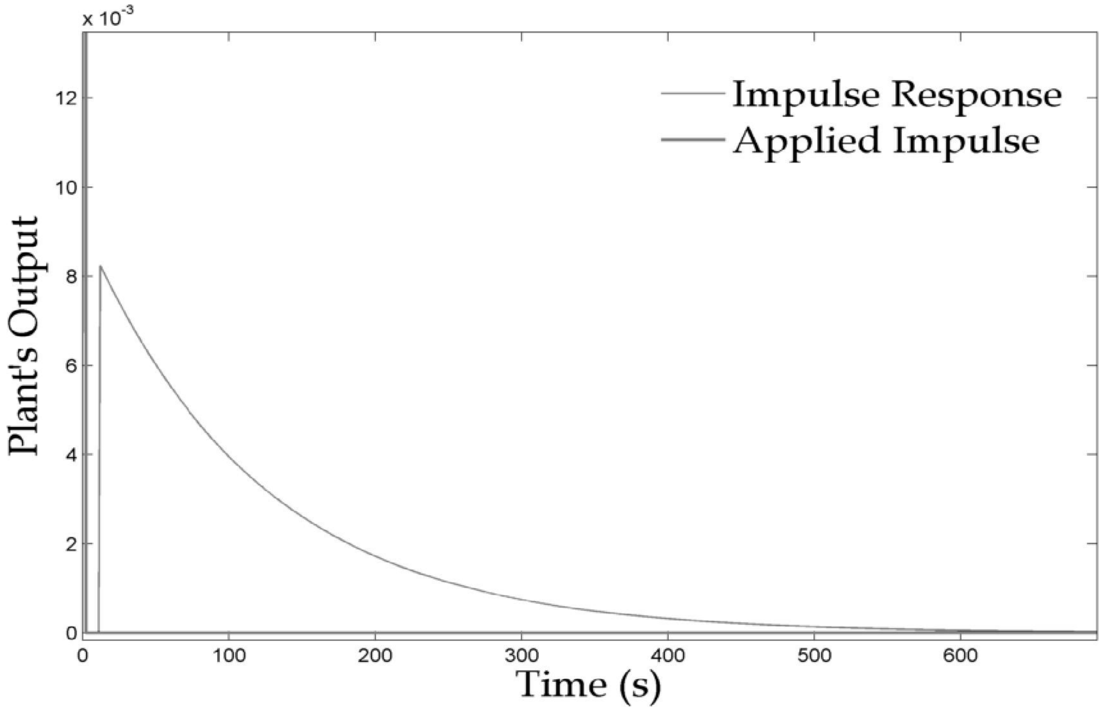

Fig 5.3: Impulse Response of Uncontrolled System A

Now the task is to control the system. The above analysis of the uncontrolled system (by assuming that the system is known) indicates that the main problem with this system is the static nonlinearity due to which the plant's response does not follow the reference. If the nonlinearity is removed from the system, it will work fine. Other performance parameters can be varied as per desire by utilizing the Predictive controller.

The only available information about the system is that it possesses a Hammerstein model. Using this information, the system is first identified as a fuzzy Hammerstein model. Resultantly, the parameters of linear dynamics and the fuzzy consequents of static nonlinearity are identified online. For System A:

The identified parameters of Linear Dynamics (Discrete time model with sampling time of 1s) are:

$$\frac{0.0040\mathrm{z} + 0.0040}{\mathrm{z} - 0.992}$$

And the parameters of nonlinear dynamics are identified as:

[0.358 0.7050 0.7003 1.3024 1.0936 1.2436].

The parameters of linear dynamics are applied to the Linear GPC scheme. The control scheme generates a virtual control action v i.e. applied to the adaptive inverse fuzzy Hammerstein model of the static nonlinearity. This controller makes fuzzy sets of the virtual control action with the cores centered at the values of fuzzy consequents that are identified online by the online identification scheme. This inverse controller is provided with the feedback of actual control action to generate a suitable control action for the actual Hammerstein model of the plant. The error generated by feedback signal is scaled according to the system under observation hence the error gain ξ is an important control parameter.

Consequently the inverse model cancels out the effect of static nonlinearity hence the linear GPC algorithm is able to control the Hammerstein model just by the tuning of certain control parameters i.e. the control effort λ, Control Horizon H_c, Prediction horizon

H_p and Error gain ξ.

With λ =0.4, H_c =2, H_p =15 and ξ=0.6, the system produces the best control results. Fig 5.4 shows the corresponding simulation results. The pulse train response shows that initially the controller learns but later on it starts to follow the set point effectively.

Another aspect was observed when the system was tested with a changed set of parameters (H_c=2, H_p=4, λ=0.04, ξ=0.8). For short span of time, there seems no disturbance in response as shown in Fig 5.5 but Fig 5.6 shows that although the plant is well controlled but when the simulation ran for longer time we observed that the response decreases and then regains the desired amplitude.

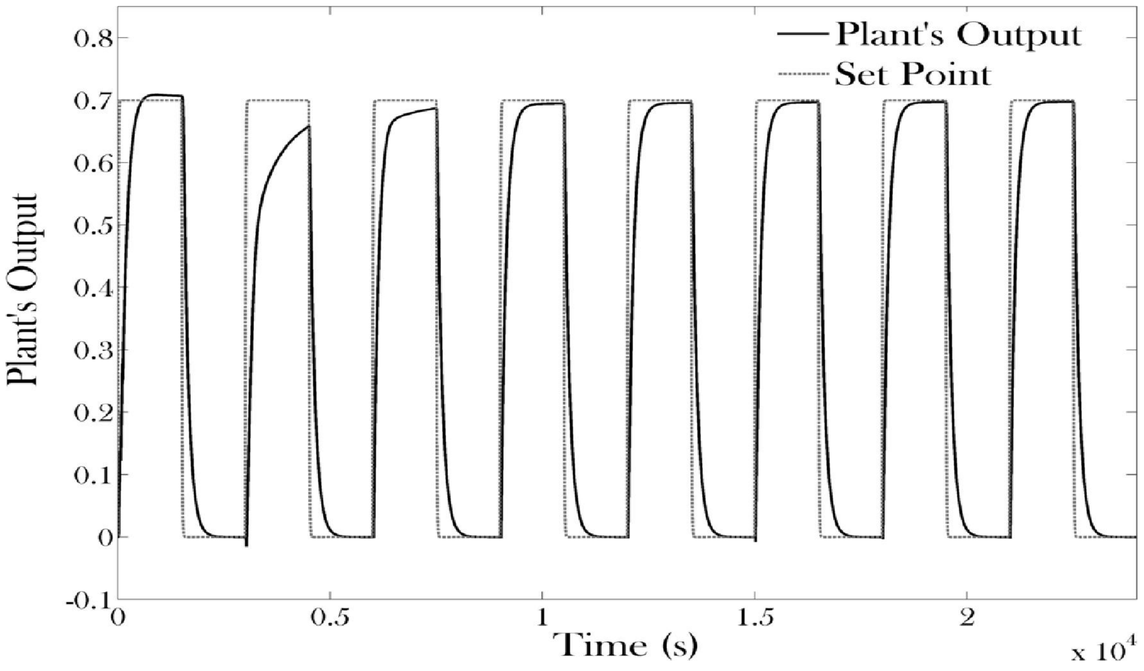

Fig 5.4: Plant's Response for System A

The possible reason is this delay in regaining the desired amplitude is the feedback in inverse controller. This is found to be a flaw of introducing feedback with error gain in the adaptive controller. However, with smaller values of error gain and suitable control parameters for predictive controller, this problem may be minimized or even solved completely as observed in the previous case.

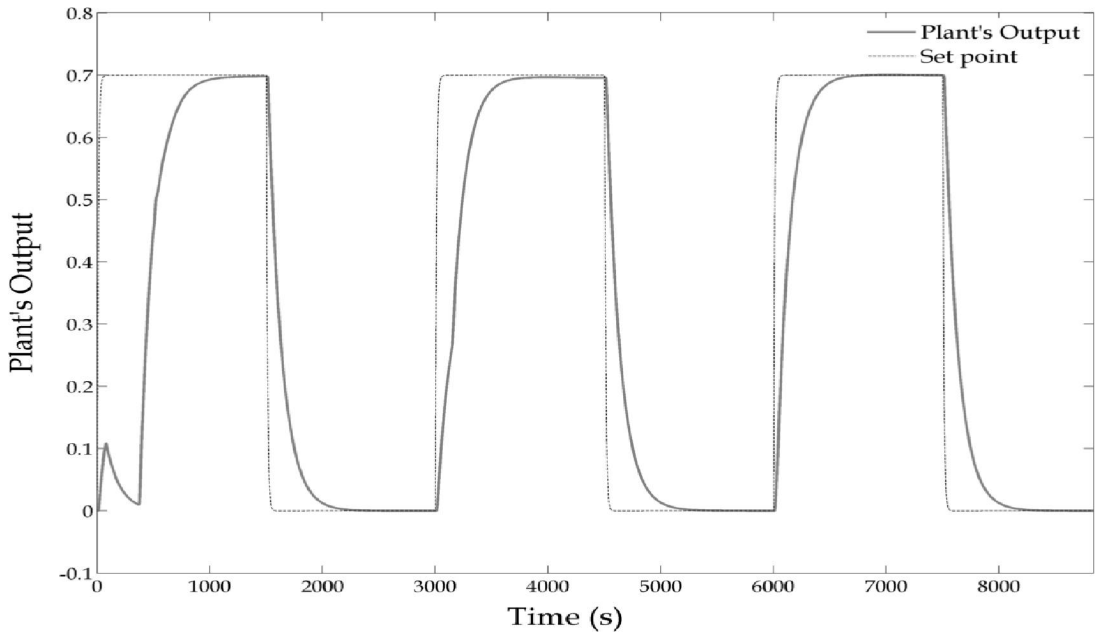

Fig 5.5: Plant's response with different control parameters

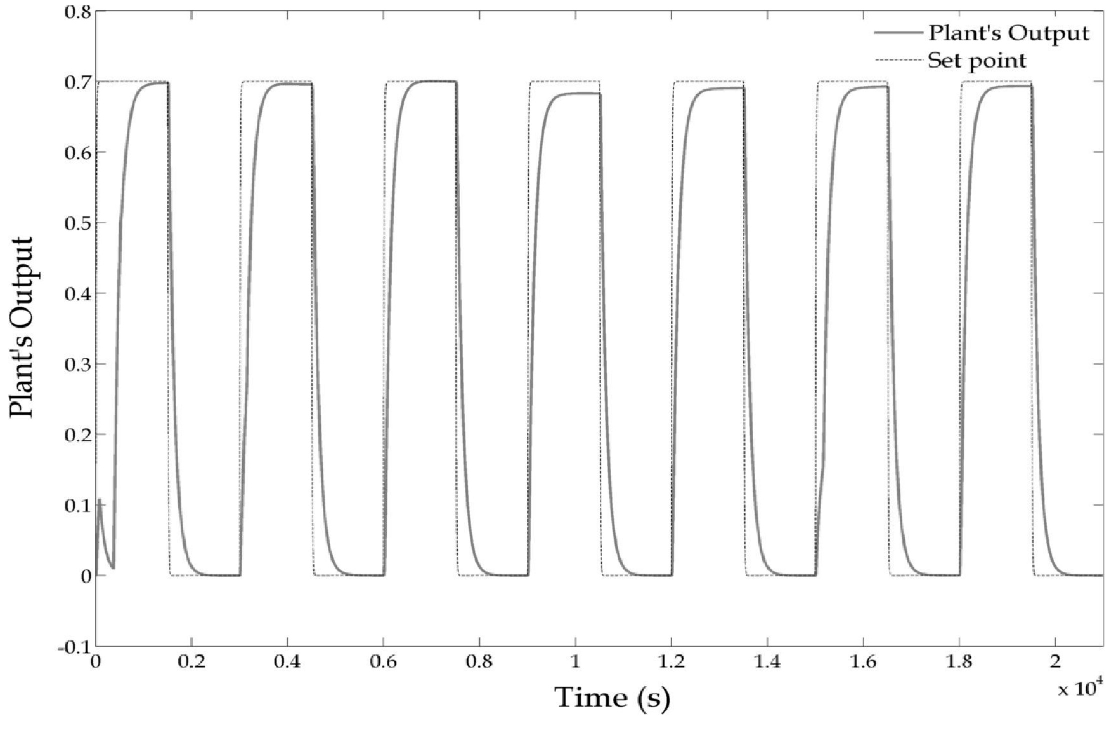

Fig 5.6: Simulation for longer time

The error plot of set point v/s plant's output is given in Fig 5.7. A significant error is observed initially but later on the system is tuned to follow the set point nicely. Initial transients are however observed at each transient mostly due to the system dynamics i.e. delayed and little bit due to the selection of control parameters that may be retuned to get the desired performance.

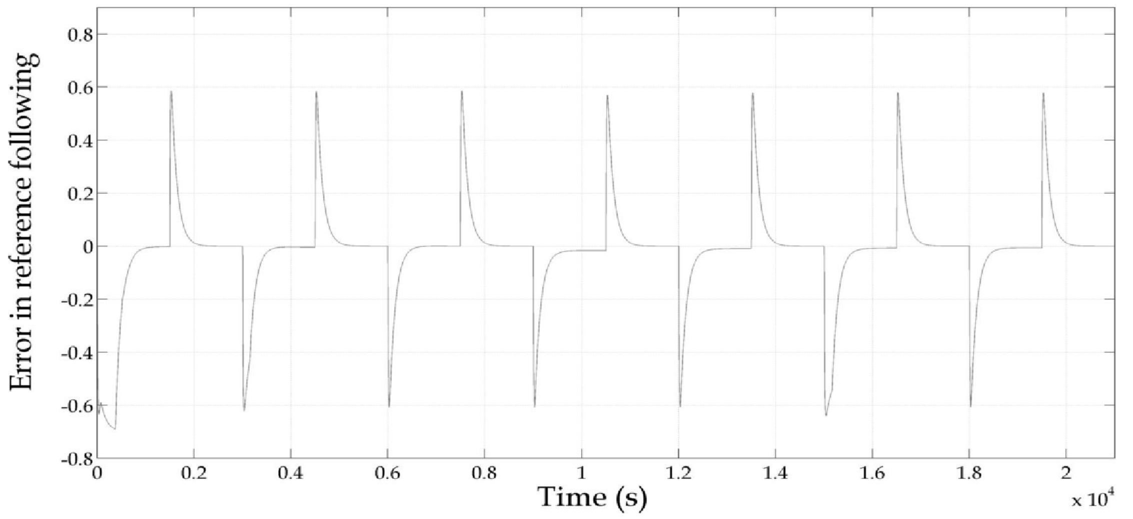

Fig 5.7: Error between Set point and Plant's output

The virtual control action produced by GPC algorithm and the control action to be applied to linear dynamics after the cancellation of static nonlinearity is shown in Fig 5.8. This indicates the fine performance of adaptive TS inverse fuzzy model controller.

The GPC algorithm generates the desired virtual control action by making use of its output obtained after convex optimization of the change in control action. The desired constraints for this change in control action are fulfilled by the algorithm and the performance of the algorithm is shown in Fig 5.9. Further desired control performances can be obtained by varying the control parameters of the Generalized Predictive control algorithm.

It is claimed that the proposed scheme is capable enough to control variety of Hammerstein models having different nonlinearities and linear dynamics. In order to prove this, two more systems are controlled using the same scheme as described below.

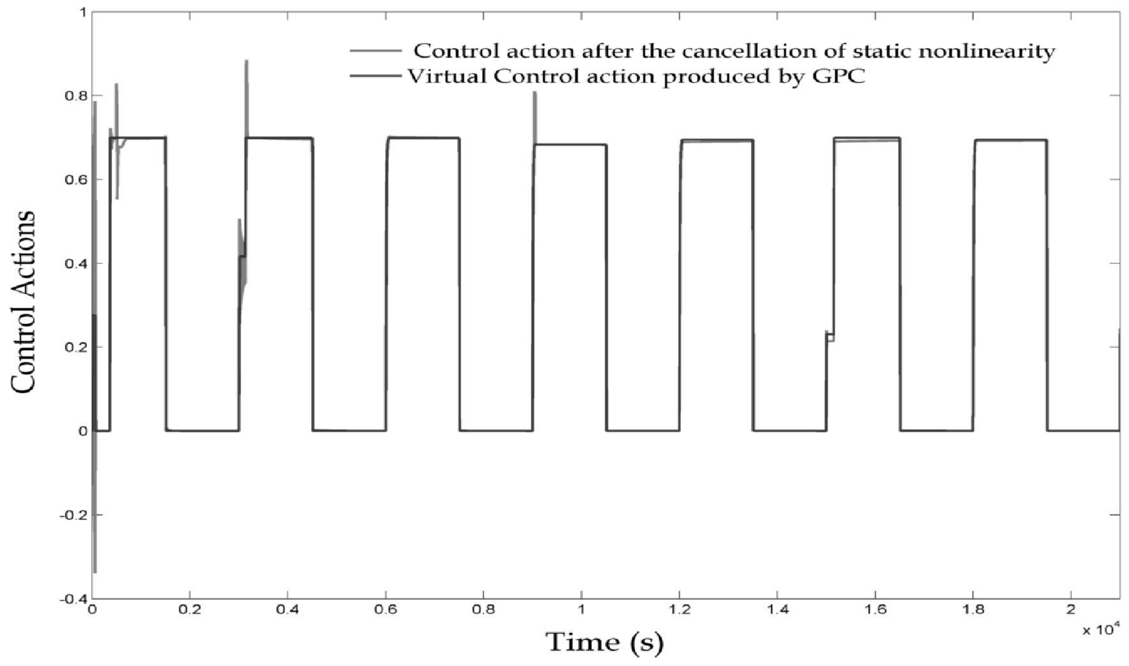

Fig 5.8: Performance of Inverse Controller

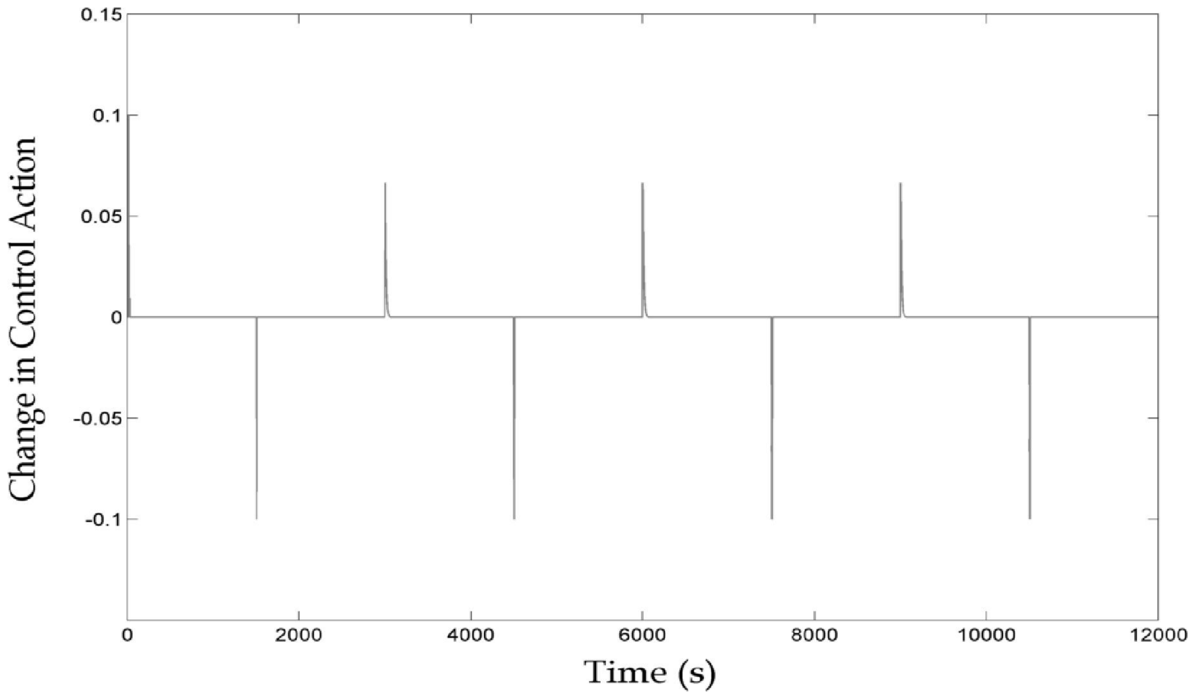

Fig 5.9: Change in Virtual Control action produced by GPC algorithm

5.2.2 System B: Exponential Nonlinearity _ τ =50 in Linear Dynamics

Consider another Hammerstein model having exponential nonlinearity and linear dynamics with τ=50. System delay is kept constant. The overall Hammerstein model is:

$$\frac{y}{u} = u * e(0.9u)e^{-10s}\left(\frac{1}{50s+1}\right)$$

In order to observe the characteristics of the open loop system, once again we assume for a while that the Hammerstein model of the system is known and test its performance by the unit and impulse response.

The open loop response of the plant with unit step input is shown in Fig 5.10. The unit step response shows that the uncontrolled system does not follow the set point rather it is 0.6 units larger than the set point.

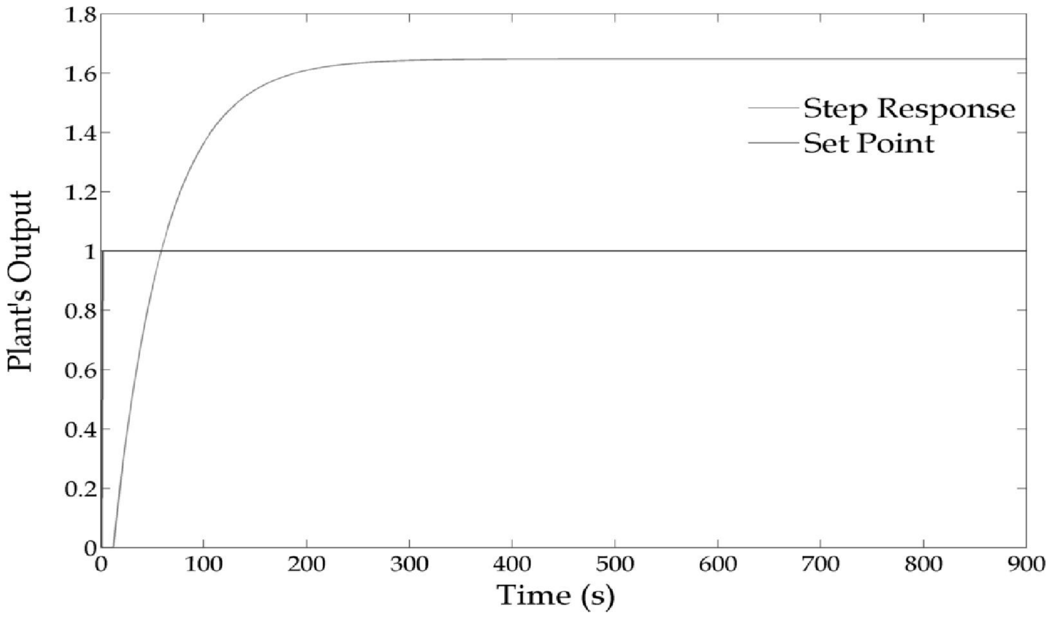

Fig 5.10: Step Response of Uncontrolled System B

Then the uncontrolled system was tested with a train of pulses having amplitude of approximately 0.7 units. The response is shown in Fig 5.11. When the system is excited with a pulse train of 0.7 units amplitude, the response of the system indicates 0.3 units increase in the output with no offset and overshoot.

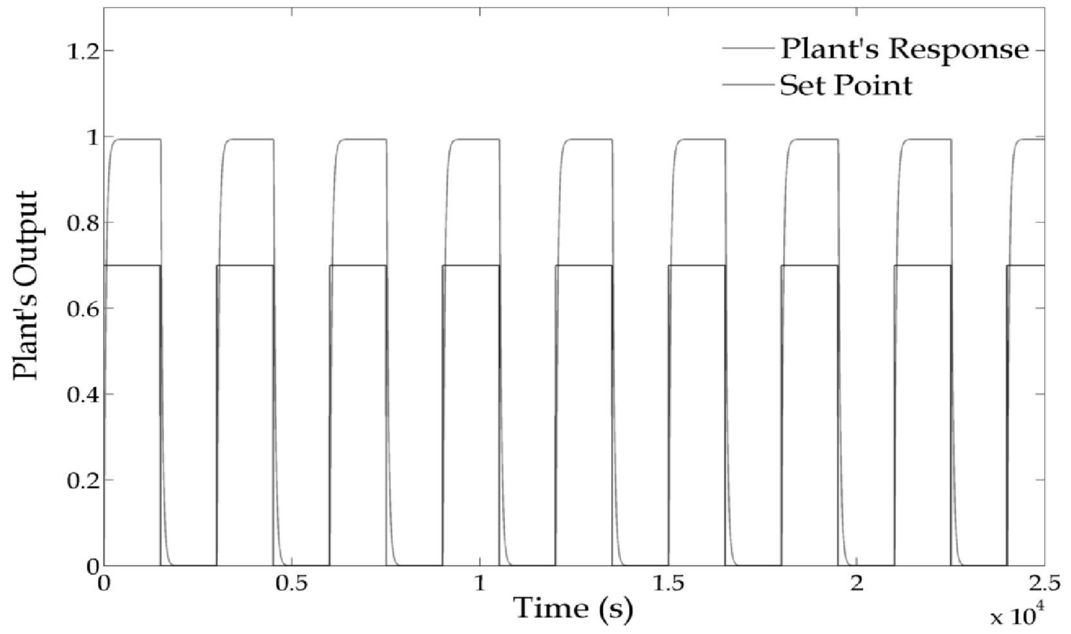

Fig 5.11: Open loop response of System B for train of pulses

In order to observe the impact of disturbance on the system, a unit impulse is applied to it and the impulse response was observed as shown in Fig 5.12. It is evident from Fig 5.12 that with unit impulse, the disturbance amplitude will be approximately 3% only and it will completely settle down within 300s.

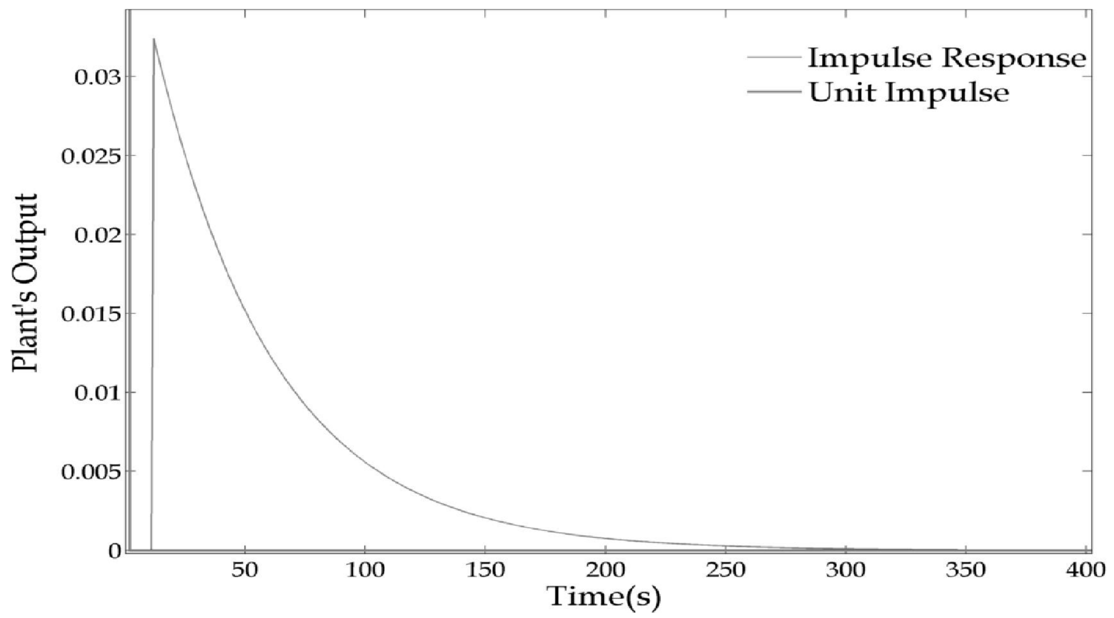

Fig 5.12: Open loop Impulse Response for System

Now the task is to control the system. Like System A, the above analysis of the uncontrolled system (by assuming that the system is known) indicates that the main problem with this system is the static nonlinearity due to which the plant's response does not follow the reference. If the nonlinearity is removed from the system, it will work fine. Other performance parameters can be varied as per desired; by utilizing the Predictive controller.

The only available information about the system is that it possesses a Hammerstein model. Using this information, the system is first identified as a fuzzy Hammerstein model. Resultantly, the parameters of linear dynamics and the fuzzy consequents of static nonlinearity are identified online. For System B:

The identified parameters of Linear Dynamics (Discrete time model with sampling time of 1s) are:

$$\frac{0.0099\mathrm{z} + 0.0099}{\mathrm{z} - 0.9803}$$

And the parameters of nonlinear dynamics are:

[0.0014 0.3922 0.7717 0.5076 0.3576 0.5076]

The parameters of linear dynamics are applied to the Linear GPC scheme. The control scheme generates a virtual control action v i.e. applied to the adaptive inverse fuzzy Hammerstein model of the static nonlinearity. This controller makes fuzzy sets of the virtual control action with the cores centered at the values of fuzzy consequents that are identified online by the online identification scheme. This inverse controller is provided with the feedback of actual control action to generate a suitable control action u for the actual Hammerstein model of the plant. Consequently the inverse model cancels out the effect of static nonlinearity hence the linear GPC algorithm is able to control the Hammerstein model just by the tuning of certain control parameters i.e. the control effort λ, Control Horizon H_c, Prediction horizon H_p and Error gain ξ.

With control effort λ=0.1, Control Horizon H_c= 1, Prediction horizon H_p =5 and the Error gain (ξ= 0.7), the controller controlled the system efficiently. Simulation results with these parameters are given in Fig 5.13.

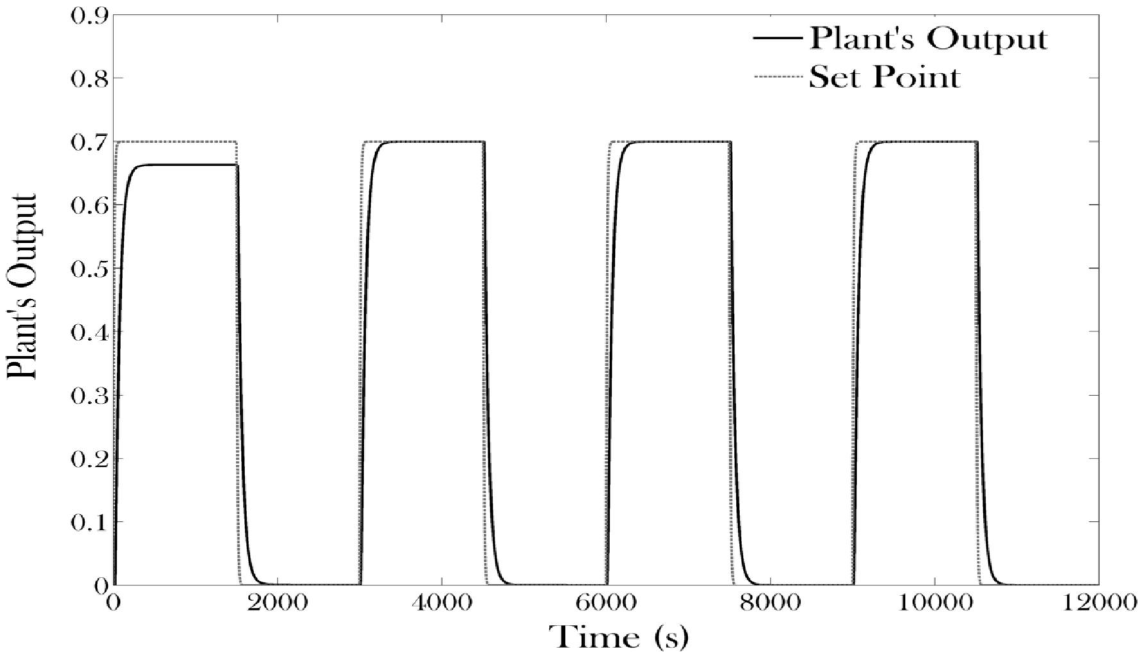

Fig 5.13: Plant's Response for System B

The error plot of set point v/s plant’s output is given in Fig 5.14. A significant error is observed initially but later on the system is tuned to follow the set point nicely. Initial transients are however observed at each transient mostly due to the system dynamics i.e. delayed and little bit due to the selection of control parameters that may be retuned to get the desired performance.

The virtual control action produced by GPC algorithm and the control action to be applied to linear dynamics after the cancellation of static nonlinearity is shown in Fig 5.15. This indicates the fine performance of adaptive TS inverse fuzzy model controller.

The GPC algorithm generates the desired virtual control action by making use of its output obtained after convex optimization of the change in control action. The desired constraints for this change in control action are fulfilled by the algorithm and the performance of the algorithm is shown in Fig 5.16.

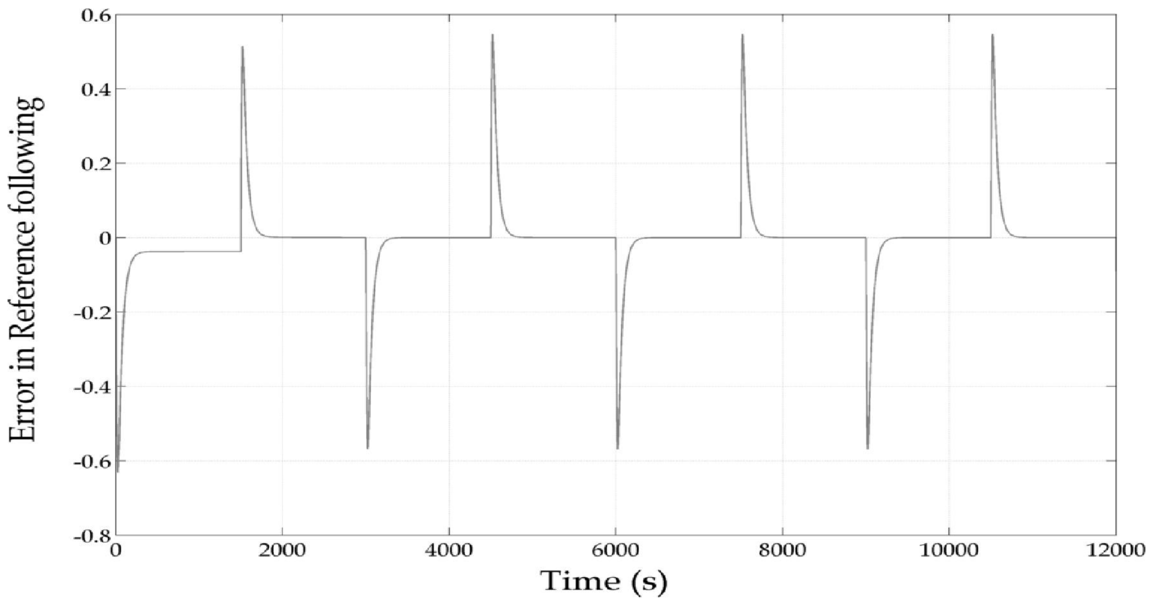

Fig 5.14: Error between Set point and Plant's output

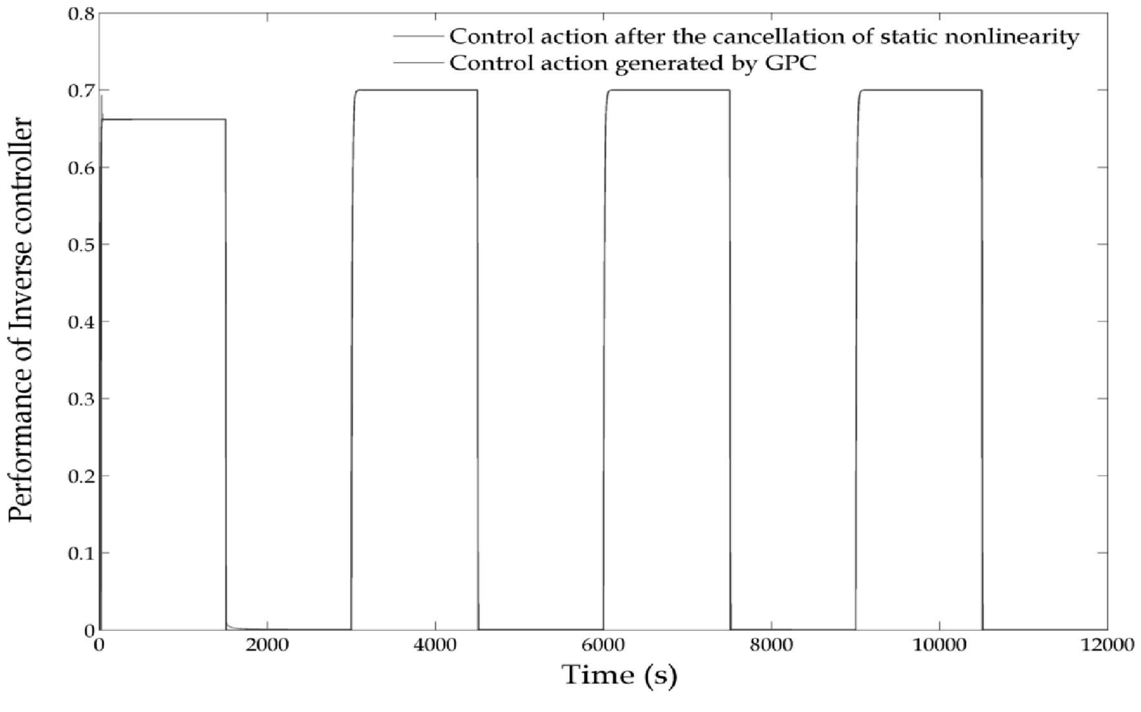

Fig 5.15: Performance of Inverse Controller

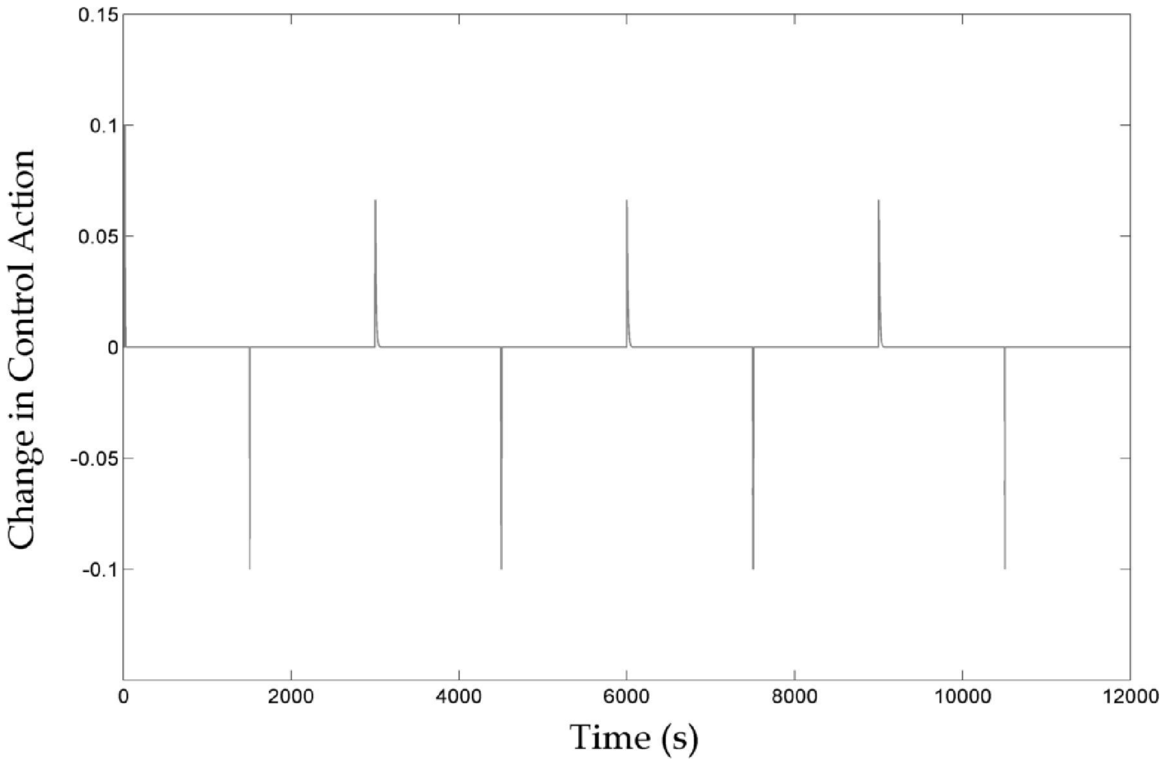

Fig 5.16: Change in control action produced by GPC algorithm

5.2.3 System C: Squared Nonlinearity _ τ =60 in Linear Dynamics

Consider another Hammerstein model with squared nonlinearity and linear dynamics with τ=60. System delay is kept constant. The overall Hammerstein model is:

$$\frac{y}{u} = (u^2 + 0.5u)e^{-10s}\left(\frac{1}{60s+1}\right)$$

In order to observe the characteristics of the uncontrolled system, we assume for a while that the Hammerstein model of the system is known and test its performance by the unit and impulse response.

The open loop response of the plant with unit step input is shown in Fig 5.17. The unit step response shows that the uncontrolled system does not follow the set point rather it is 0.5 units larger than the set point.

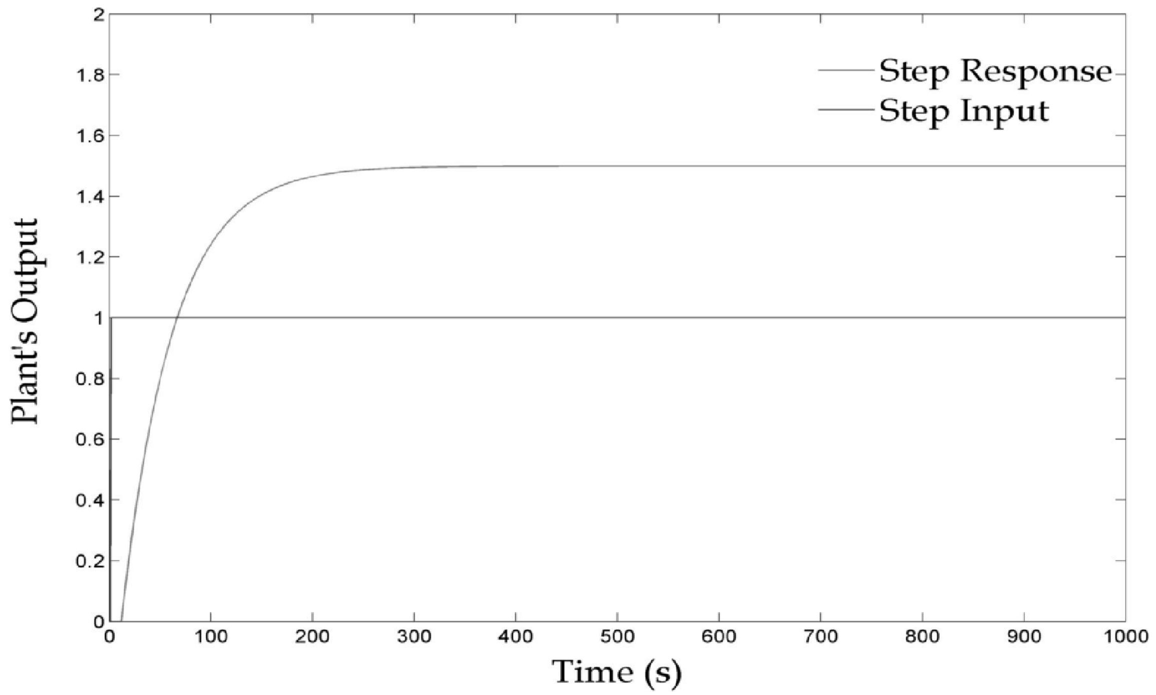

Fig 5.17: Step Response of Open loop System C

Then the uncontrolled system was tested with a train of pulses having amplitude of approximately 0.7units. The response is shown in Fig 5.18. When the system is excited with a pulse train of 0.7units amplitude, the response of the system indicates 0.05 units increase in the output with no offset and overshoot.

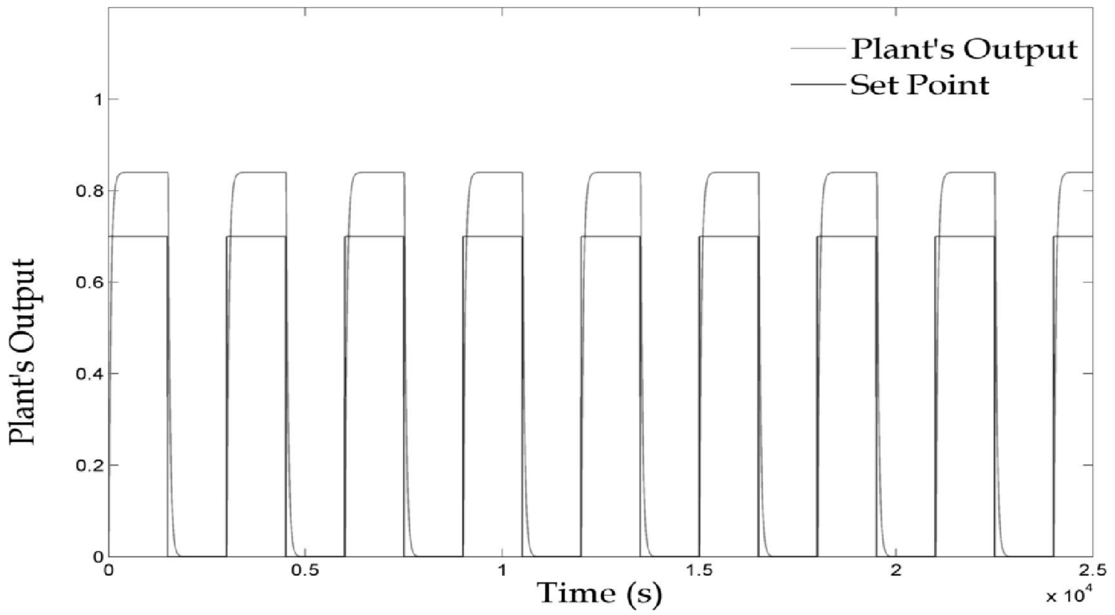

Fig 5.18: Uncontrolled Plant's response for pulse train Input

In order to observe the impact of disturbance on the system, a unit impulse is applied to it and the impulse response was observed as shown in Fig 5.19. The response shows that disturbance amplitude will be approximately 3% only and it will completely settle down within 300s.

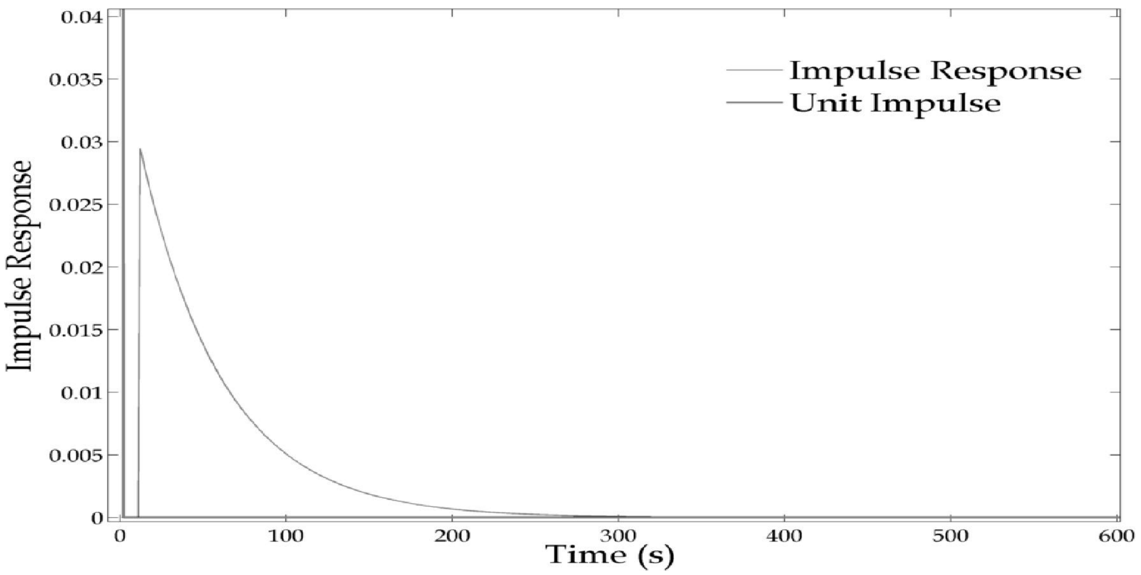

Fig 5.19: Impulse response of Uncontrolled System C

Now the task is to control the system. Like System A, the above analysis of the uncontrolled system (by assuming that the system is known) indicates that the main problem with this system is the static nonlinearity due to which the plant's response does not follow the reference. If the nonlinearity is removed from the system, it will work fine. Other performance parameters can be varied as per desired; by utilizing the Predictive controller.

The only available information about the system is that it possesses a Hammerstein model. Using this information, the system is first identified as a fuzzy Hammerstein model. Resultantly, the parameters of linear dynamics and the fuzzy consequents of static nonlinearity are identified online. For System C:

The identified parameters of Linear Dynamics (Discrete time model with sampling time of 1s) are:

$$\frac{0.0083 + 0.0083}{\mathrm{z} - 0.9835}$$

And the parameters of nonlinear dynamics are:

[0.0022 0.2433 0.4914 0.8299 0.4551 0.6051]

The parameters of linear dynamics are applied to the Linear GPC scheme. The control scheme generates a virtual control action v i.e. applied to the adaptive inverse fuzzy Hammerstein model of the static nonlinearity. This controller makes fuzzy sets of the virtual control action with the cores centered at the values of fuzzy consequents that are identified online by the online identification scheme. This inverse controller is provided with the feedback of actual control action to generate a suitable control action u for the actual Hammerstein model of the plant. Consequently the inverse model cancels out the effect of static nonlinearity hence the linear GPC algorithm is able to control the Hammerstein model just by the tuning of certain control parameters i.e. the control effort λ, Control Horizon H_c, Prediction horizon H_p and Error gain ξ.

Using the control effort (λ) of 0.1, unity Control Horizon (H_c), double Prediction horizon (H_p=2) and error gain (ξ=0.7), the system performs well. Fig **5.20** shows the corresponding simulation results. The pulse train response follows the set point effectively from the very beginning.

The error plot of set point v/s plant's output is given in Fig 5.21. A significant error is observed initially but later on the system is tuned to follow the set point nicely. Initial transients are however observed at each transient mostly due to the system dynamics i.e. delayed and little bit due to the selection of control parameters that may be retuned to get the desired performance.

The virtual control action produced by GPC algorithm and the control action to be applied to linear dynamics after the cancellation of static nonlinearity is shown in Fig 5.22. This indicates the fine performance of adaptive TS inverse fuzzy model controller.

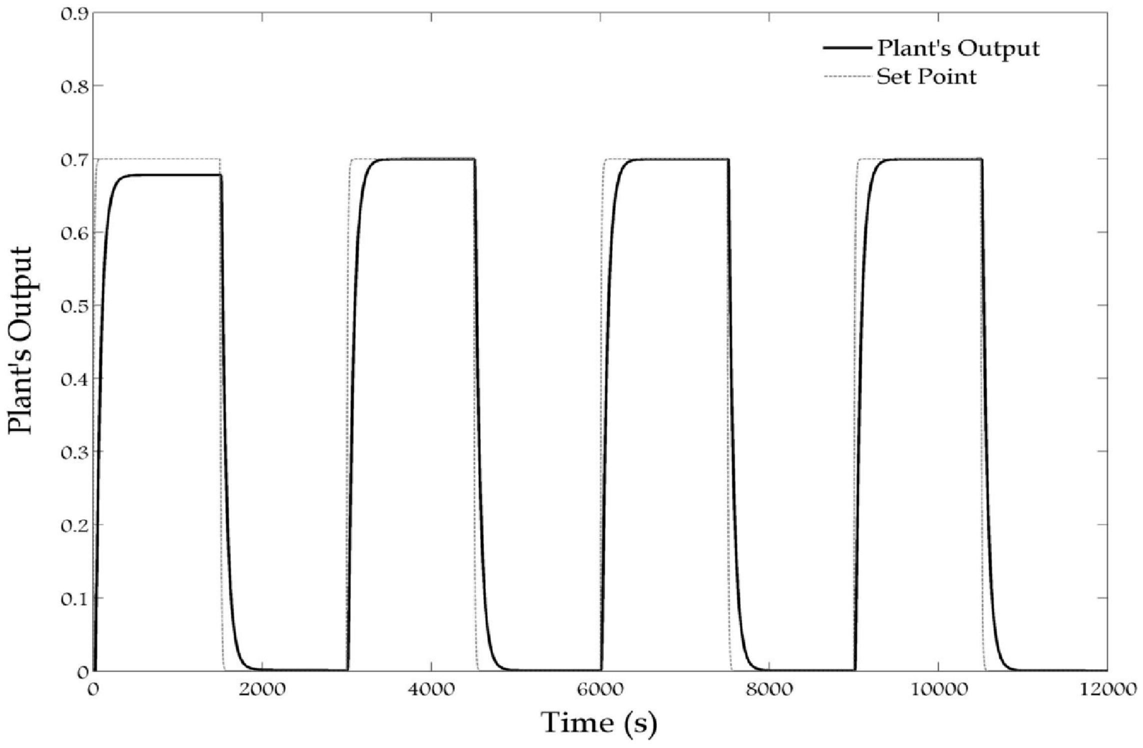

Fig 5.20: Plant's Response for System C

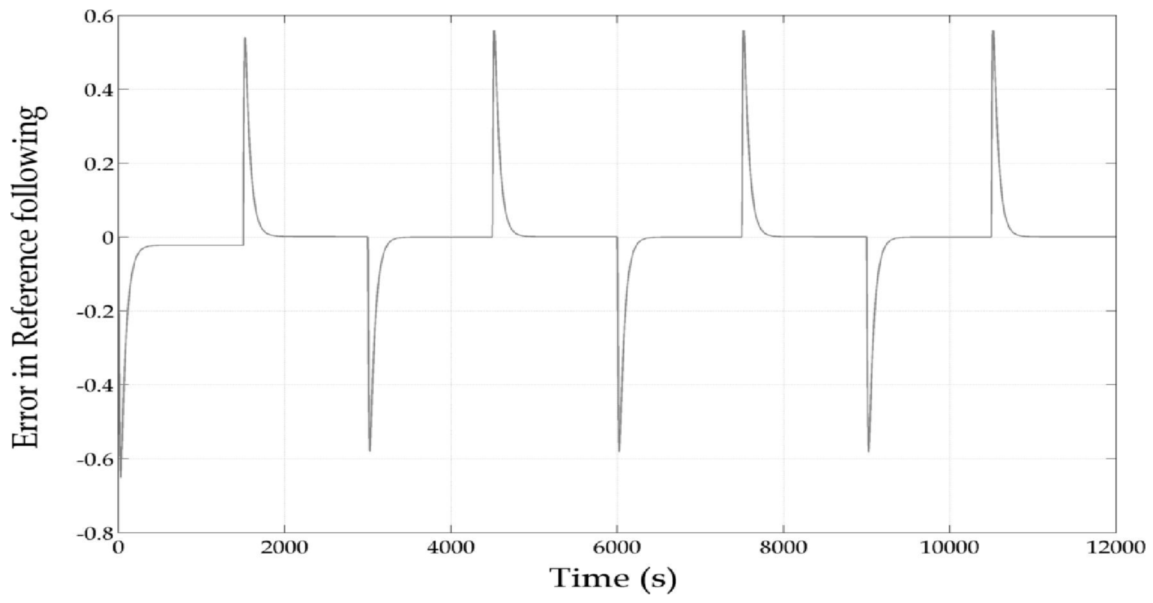

Fig 5.21: Error between Set point and Plants response

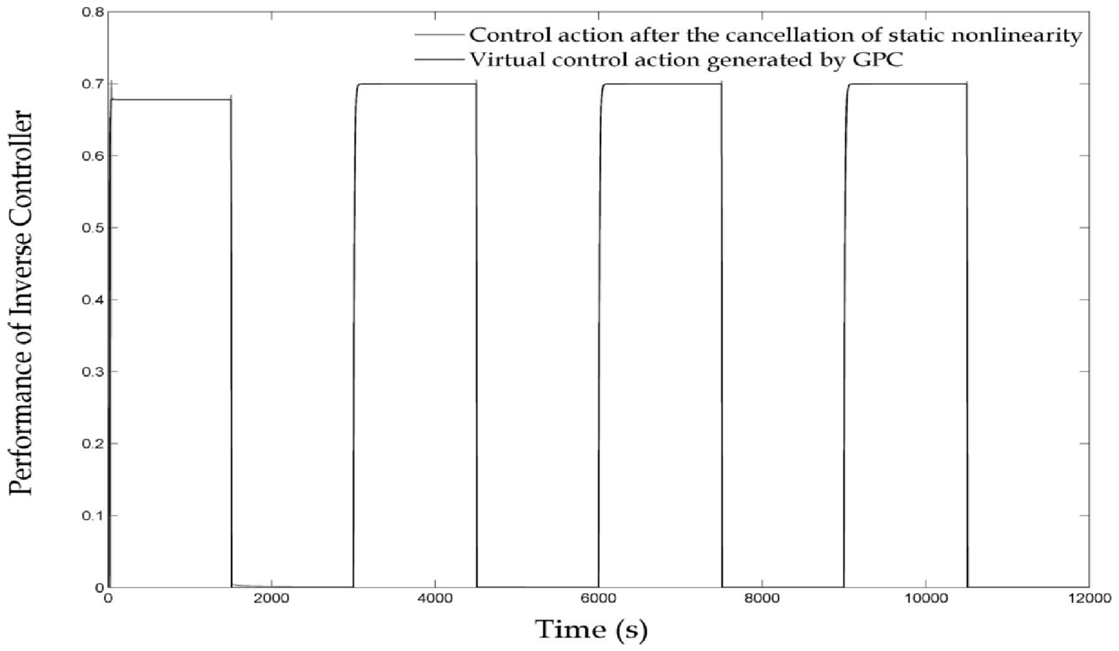

Fig 5.22: Control actions

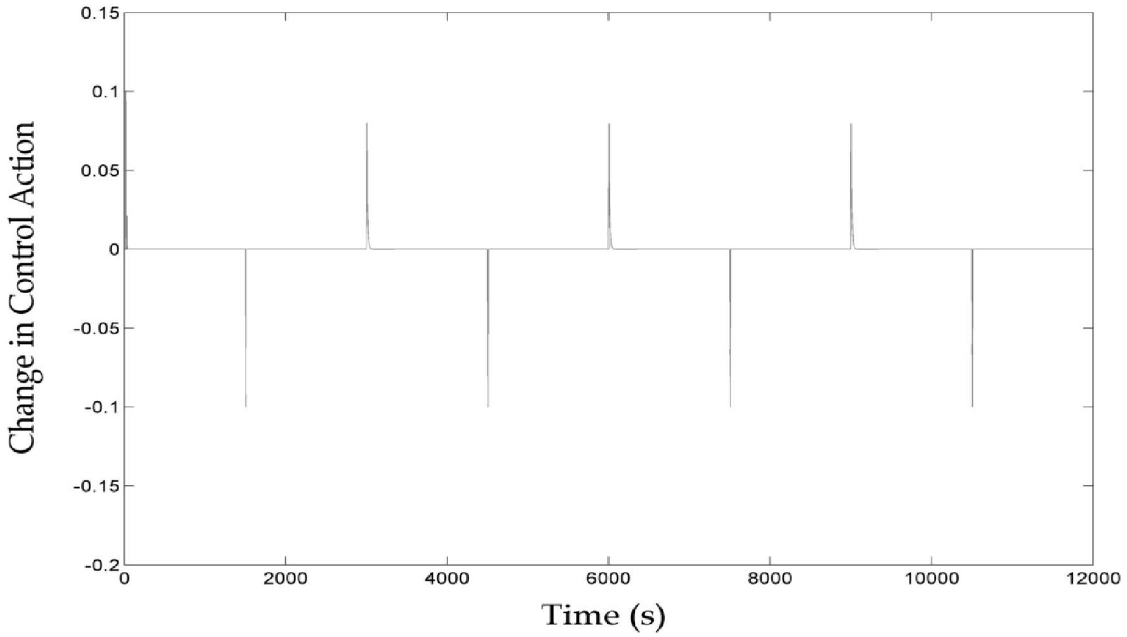

Fig 5.23: Change in control action

The GPC algorithm generates the desired virtual control action by making use of its output obtained after convex optimization of the change in control action. The desired constraints for this change in control action are fulfilled by the algorithm and the performance of the algorithm is shown in Fig 5.23.

5.2.4 Online variation in system parameters

In order to further test the controller, we started the control simulation with plant model:

$$\frac{y}{u} = u * e(0.9u)e^{-10s}\left(\frac{1}{120s+1}\right)$$

After 5535s, the linear dynamics was changed as follows:

$$\frac{y}{u} = u * e(0.9u)e^{-10s}\left(\frac{1}{60s+1}\right)$$

At 10640s, the linear dynamics was further changed as follows:

$$\frac{y}{u} = u * e(0.9u)e^{-10s}\left(\frac{1}{20s+1}\right)$$

Then at 16040s, the static nonlinearity was changed to be cubic and the overall Hammerstein model becomes:

$$\frac{y}{u} = (u^3 + 0.5u) * e^{-10s}\left(\frac{1}{20s+1}\right)$$

The variation in plant's response with this variation in system parameters is shown in Fig 5.24. It is observed that when the parameter of linear dynamics (i.e. τ) is changed, the controller continues to perform fine after a little transient depending upon the variation in system parameter without any variation in tuning parameters. At 5536s when τ was reduced from 120s to 60 s, there was a little spike in response with amplitude of about 1.2ms. After this little transient, the plant continues to follow the reference. When the dynamics is further changed (τ reduced from 60s to 20s), the response shows transients as mentioned clearly in Fig 5.24 but settles down to follow the set point. But when the type of static nonlinearity is changed online, the response disturbs i.e. it needs retuning of the controller parameters to perform smoothly. This is obvious because change of static nonlinearity reflects change in model structure. GPC controllers are claimed to offer efficient performance with varying linear dynamics and this is evident from the plants' response in Fig 5.24.

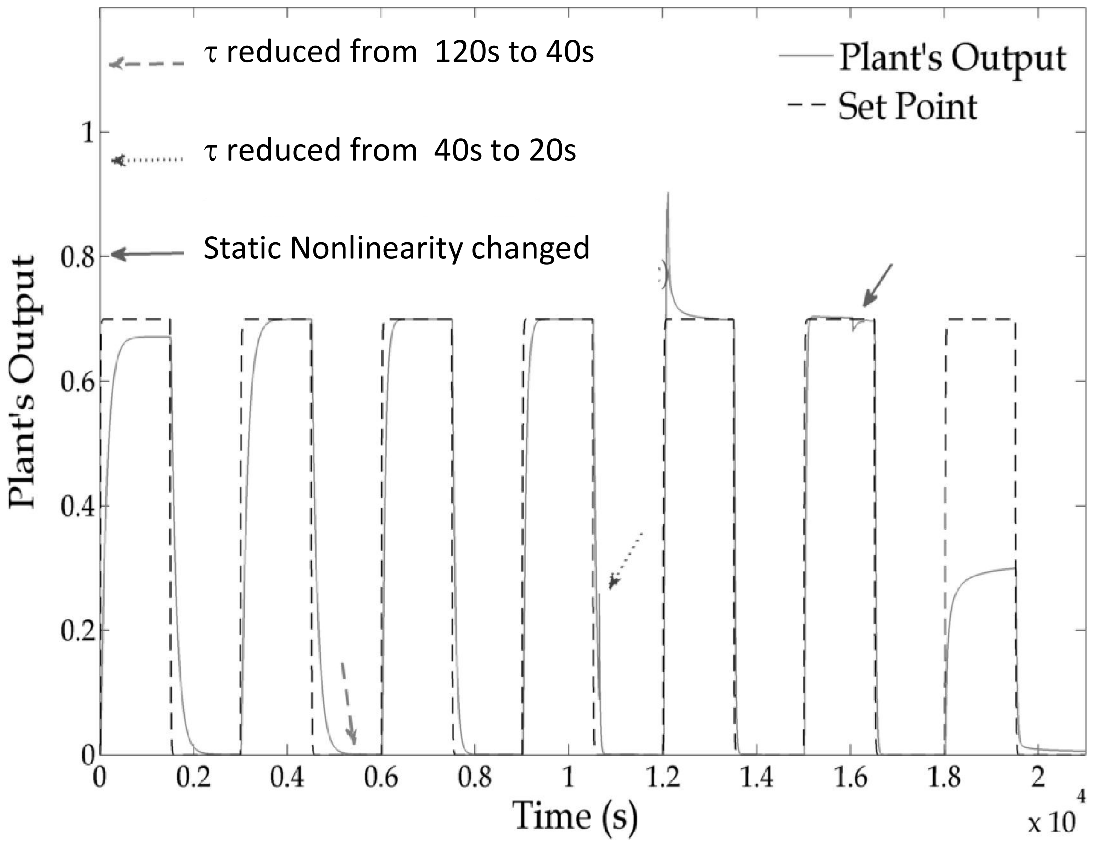

Fig 5.24: Plant's Response with varying dynamics

5.3 Chapter Summary

Control of three different nonlinear systems with fuzzy Hammerstein models is presented in this chapter. The overall control strategy has performed satisfactorily with these systems. Furthermore the system dynamics is varied online and the possibility of retuning was investigated. It was observed that with the variation in system parameter values of linear dynamics, no retuning is required but when the structure of nonlinearity is changed, the controller needs to be retuned for efficient and effective control.

CHAPTER 6

6 Past, Present and Future of the Proposed Control Scheme

6.1 Prologue

In this chapter we will discuss the basic idea, from which the proposed scheme was derived i.e. the past of the scheme, the modifications in the original work and its benefits i.e. the present of the scheme and will propose some future modifications that can amplify the credibility of the control scheme.

6.2 Past of the Proposed Control Scheme

The original scheme of *Fuzzy Hammerstein model based Generalized Predictive Control* dates back to 2000 when Janos Abonyi proposed to identify and control the nonlinear system using Hammerstein Model. In his scheme, he first identified the nonlinear system using Fuzzy Hammerstein model (discussed in chapter 3). In this scheme he utilizes RLS algorithm with the consideration of constraints to identify the system properly. Once the model is identified, he implements GPC algorithm, developed using Diophantine equation to generate a virtual control action. In order to control the actual plant that is identified as fuzzy Hammerstein model however, he makes use of an adaptive inverse model. The inverse model is developed using the famous Mamdani structure. This generates the actual control action for the plant (the Hammerstein model). In order to achieve guaranteed convergence of control action however, double layered convex optimization using Quadratic programming is implemented. The flow of work is similar to the proposed scheme and has already been discussed in chapter 4. Here the components of Abonyi’s scheme are presented in Fig 6.1.

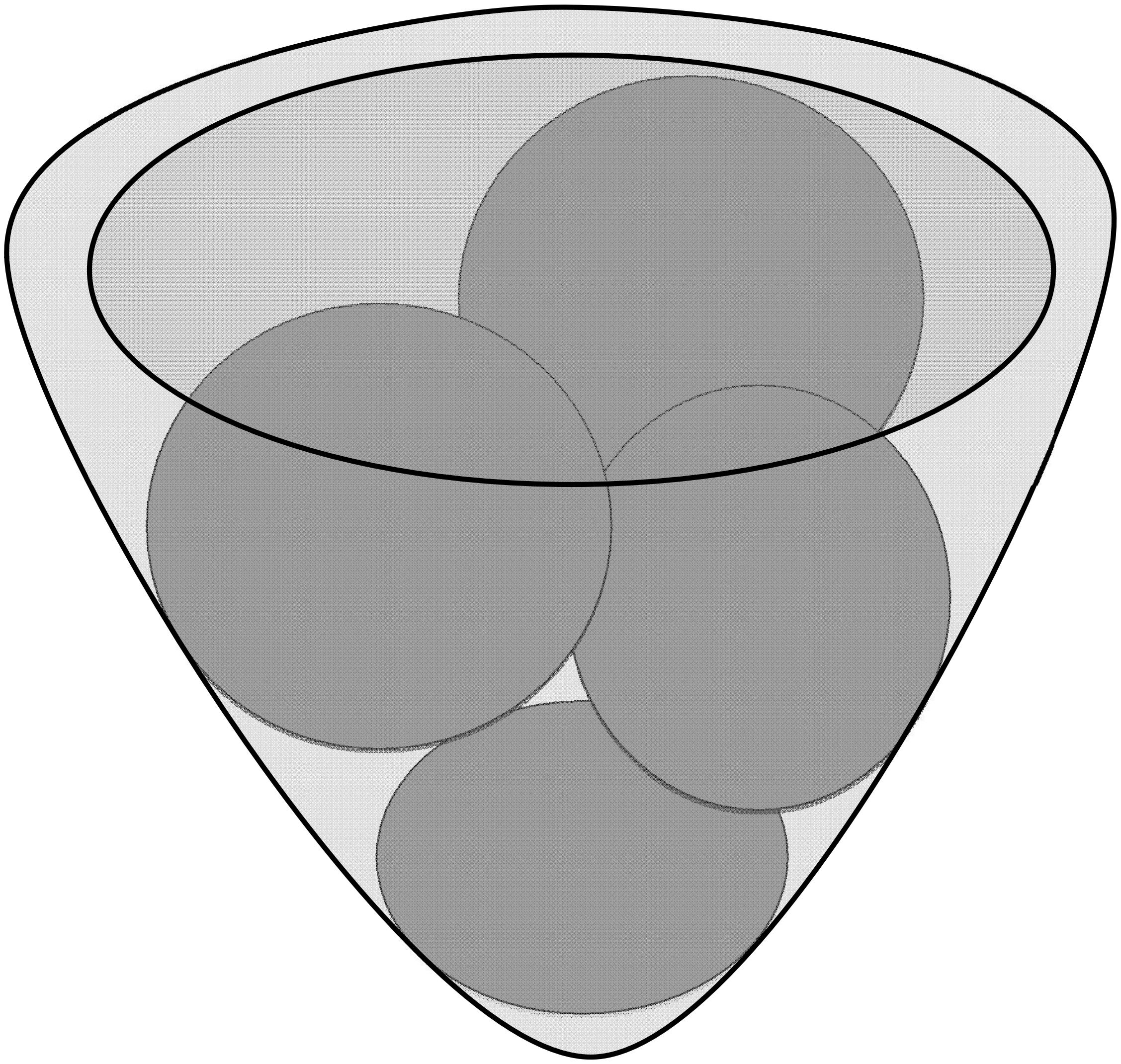

Fig 6.1: Components of Abonyi's control scheme

Abonyi utilized his scheme to identify and control an Electrical Water Heater’s Hammerstein model and produced satisfactory results. Later on F. Jurado and M.Ortega tested the same scheme to control Solid Fuel Cell and got the desired results [13].

6.3 Present of the Proposed Control Scheme

The combination suggested by Abonyi offers a simple and nice scheme to control the class of nonlinear systems but the proposed scheme makes use of such algorithms and structures which makes the computational load high, losing its utilization. Furthermore he has mostly emphasized on system identification but some more consequences can be obtained from the scheme which is individually offered by linear GPC algorithm. So in order to relax the computational burden to enhance the usability of the proposed scheme and to unveil further consequences of the scheme, some modifications were carried out on Abonyi's working. The proposed control has already discussed in detail and the results and consequences have been discussed, here the components of the scheme and its core benefits will be outlined.

In the proposed scheme, the identification is carried out as it is done by Abonyi. The Generalized Predictive Control algorithm is developed using Toeplitz Hankel matrices rather than more computationally heavy Diophantine equations [41]. Ying Xu *et al* proposed the algorithm and proved its efficiency over the Diophantine equation based algorithm [51]. It is simple to use and implement; and reduces the computational load. Furthermore Abonyi's inverse model was adaptive but made use of Mamdani structure that utilizes the concept of interpolation. On the other hand TS fuzzy model based controller are computationally less complex and offer efficient working in adaptive control. Keeping in view that the nonlinear plant is unknown and the desired control action is also unknown, *Direct Control* approach of inverse model controller as proposed by TAN *et al* [45] is blended with zeroth order TS fuzzy inverse model controller. Resultantly, the scheme performs well in nullifying the effect of static nonlinearity and can control Hammerstein model having variety of static nonlinearities.

In order to further reduce the computational burden, the effort was carried out to reduce the convex optimization to single layer. The combination of the above two modifications and the selection of gain for control weighting factor in control law, made this task to be accomplished as well.

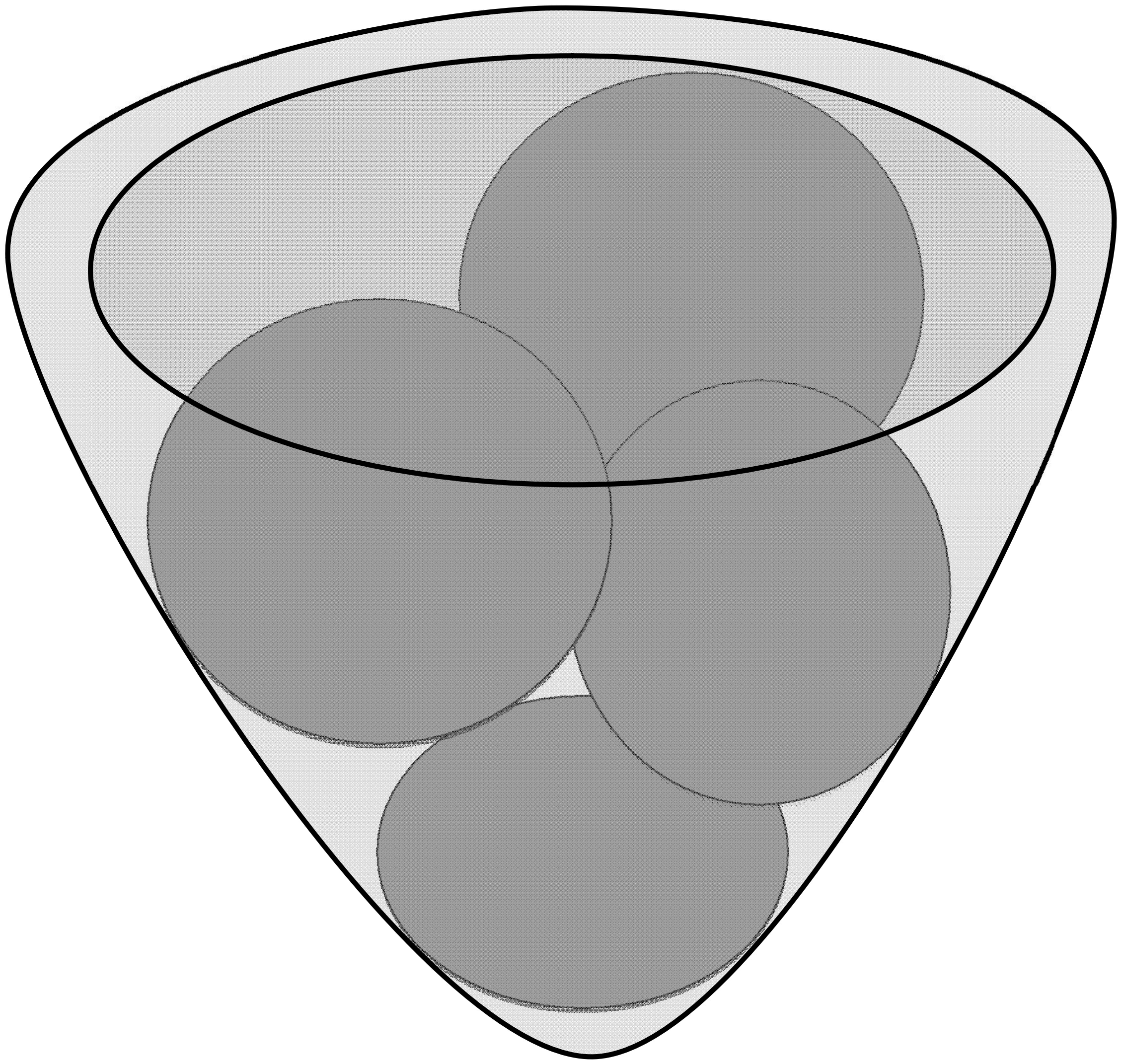

Fig 6.2: Components of the Proposed control scheme

Furthermore, as discussed earlier, all the system under test are considered to be delayed while no identification was carried out for this delay. This is done to test the capability of GPC algorithm to control delayed system without having any knowledge about the system delay.

Another capability of GPC algorithm that is to control the system under parameter variations was also observed in Section 5.2.4. The parameters of the model varied significantly but the control scheme was capable enough to control it. However when the system is itself changed by changing the static nonlinearity, the control rightly goes out.

6.4 Future of the Control Scheme

Control schemes never freeze at a point. These schemes are being improved and will remain in the running state. Several modifications have been observed from simple PID controllers till the Model Predictive approach and several are in queue. So does the proposed scheme, although it has been expanded from the original approach still the only field of Adaptive control is wide enough that can motivate designers to further improve the scheme. In fact there are some improvements which were desired to be implemented in this scheme but due to the lack of timings and resources, could not be worked out. These are listed below so that interested candidates can have a guide to further modify and improve the usability of the scheme.

- As discussed in chapter 5, the scheme is tested for simple 1st order SISO models; it was desired to check the control efficiency for second order systems at least or for any practical system such as electrical water heater etc.
- The control parameters of GPC algorithm can be made adaptive on the basis of desired performance; this will further automate the scheme.
- The error gain parameter in the feedback of inverse model controller can be made adaptive as per the system and desired performance.
- The scheme can be tested on hardware using DSPs or FPGAs to implement the controller on any practical system
- In the proposed controller, the static nonlinearity is being nullified using the inverse model controller. An effort can be carried out to remove the effect of nonlinearities by making use of constraints definition however it will require that the static nonlinearity should be known so it is more useful for actuator saturation and backlash constraints.

6.5 Conclusion

In this chapter the outline of the basic inspiration of proposed scheme, its current scenario, performance capabilities and future improvements are discussed. This final and brief description will pave the ways of improvements and research in the scheme.

References

[1] Aqsa Raees and Dr. Muhammad Bilal Kadri, "Fuzzy Hammerstein model based Generalized Predictive Control for Ball and Beam system," *in Proc. of 8th International Conference on Emerging Technologies* (*ICET*), Islamabad, Pakistan, Oct. 8-10, 2012.

[2] Aqsa Raees and Dr. Muhammad Bilal Kadri, "Fuzzy model based Predictive Control for Hammerstein models," *in Proc. of 1st International AESS-IEEE Conference on European Satellite and Telecommunication* (*ESTEL*),Italy, Oct. 2-5,2012.

[3] Aldo Cipriano and Doris Saez, "Fuzzy Generalized Predictive Control and its Application to an Inverted Pendulum", *in Proc of IEEE IECON 22nd International Conference on Industrial Electronics, Control, and Instrumentation,* vol 3 , pgs 1966-1971,1996.

[4] Alexandru Bara, Sanda Dale, "Fuzzy Model Based Predictive Control of Nonlinear Processes." *IEEE Transaction* , pgs 4, 2008.

[5] B.Koubstiyskis, J. A. Rossitor, J. R. Gossner,"Improved algorithm for multivariable stable generalised predictive control", *in IEE Proc. Of Control Theory Applications*, vol 144(4), 1997.

[6] Bak, M.,*Control of systems with constraints*, PhD thesis, Department of Automation, Technical University of Denmark, pgs 194, 2000.

[7] Bemporad, A., Manfred Morari,"Control of Systems Integrating Logic, Dynamics, and Constraints", *Automatica*, vol 35, pgs 407-427,1999.

[8] Crawshaw, S., *Control of systems with actuator nonlinearities*, Dissertion for PhD, Department of Engineering, University of Cambridge, pgs 177, 2000.

[9] Du Feng, Du Wencai and Lei Zhi, "New Smith Predictor and Generalized

Predictive Control for Wireless Networked Control Systems," *in Proc. International MultiConference of Engineers and Computer Scientists*. vol. 2, Hong Kong, March 2009.

[10] D. W. Clarke, C. Mohtadi, P. S. Tuffs,"Generalized Predictive Control-Part I. The Basic Algorithm", *Automatica*, vol23(2), pgs 137-148, 1987.

[11] D. W. Clarke, C. Mohtadi, P. S. Tuffs, "Generalized Predictive Control Part II. Extensions and Interpretations", *Automatica,* vol23(2), pgs 149-160, 1987.

[12] F. Abdessemed, S. Djbrani, "Fuzzy Model Based Predictive Control of Nonlinear Systems", *in Proc of 12th IEEE international Conference on Electronics, Circuits and Systems,* 2005.

[13] F.Giri, F.Z.Chaoui, M.Haloua, Y.Rochdi and A.Naitali, "Hammerstein model identification", in *Proc. 10th Mediterranean Conf. on Control and Automation*, Portugal, July 9-12, 2002.

[14] F. Jurado, M. Ortega, "Fuzzy model based Predictive control of a Solid oxide Fuel cell*", in Proc of 10th IEEE Conf on Emerging Technologies and Factory Automation*, vol 2, pg 987-993,2005.

[15] Ganesh U.L, Dr. M.Karishna, Dr.S.A.Hariparsad and S.Karishna Rau (2011) , "Review on models for Generalized predictive control," CSCP, vol 2, Available :http://www.airccj.org

[16] Giovanini, Leonardo L., "Model Predictive Control with amplitude and rate actuator saturation." *ISA Transactions Journal*, vol 42(2), pgs 227-240, 2003.

[17] Grigoriadis, KarolosM., *Actuator saturation control*, New York, Basel, Marcel Dekker, 2002.

[18] H. Zabiri, Y. Sanyudia, "MPC design for constrained multivariable systems under actuator backlash", in S.L. Shah and J.F. Mac Gregor (Eds.), *Dynamics and Control of Process Systems 2004*, Elsevier Press 2005.

[19] Huaguang, Z, "Mutivariable Fuzzy Generalized Predictive Control", *in Proc of an inter. Jour on Cybernatics and System*,pgs 69-99, 2002.

[20] Isermann R., M. Fischer,"Robusthybrid control based on inverse fuzzy process models", *in Proc. of the Fifth IEEE International Conference on Fuzzy Systems*, vol2, pgs 1210-1216, New York,1996.

[21] Ivan Goethals, Kristian Pelckmans, Johan A.K.Suykens and Bart De Moore, "Identification of MIMO Hammerstein models using least squares support vector machines", *Automatica 41*, pp 1263-1272, 2005.

[22] Janos Abonyi, *Fuzzy model identification for control*, USA, Birkhauser, 2002, Ch3, 4 and 5.

[23] J. Abonyi, R. Babuˇska, M. Ayala Botto, F. Szeifert and L. Nagy, "Identification and control of nonlinear systems using Fuzzy Hammerstein model," *CiteSeer--Industrial and Engineering Chemistry Research,* vol. 2,pgs 4302-4314, 2000.

[24] J. Abonyi, R. Babuˇska, F. Szeifert, L. Nagy, and H.B. Verbruggen, *Design and application of block–oriented fuzzy models – fuzzy Hammerstein model*, in Y. Suzuki, S. Ovaska, and T. Furuhashi, editors, *Soft Computing in Industrial Applications*. Springer, London, UK, September 2000.

[25] J. M. Maciejowski, *Predictive Control with Constraints*, Britain, Prentice Hall, Pearson Education, 2002.

[26] J´erˆome Mendes, RuiAra´ujo, and Francisco Souza, "Adaptive Fuzzy Generalized Predictive Control Based on Discrete-Time T-S Fuzzy Model", *Emerging Technologies and Factory Automation (ETFA),* pgs 1-8, 2010.

[27] K.J. Li and G.P. Liu, "A simplified GPC algorithm of networked control systems", in *Proc. 2007 IEEE Int. Conf. Networking, Sensing and Control*, pgs 58-63, London, UK, April 2007.

[28] K.S.Holkar and L.M.Waghmare (2010), "An overview of model predictive

control”, *International Journal* of *Control and Automation*, vol. 3,Available: http://www.sersc.org

[29] M. Sugeno and G. T. Kang, "Fuzzy modeling and control of multilayer incinerator." *Fuzzy Sets and Systems*, vol 18, pgs 329-346, 1986.

[30] Marusak, P. M., *Arificial Intelligence and Soft Computing*, pgs 136-143, Springer Berlin Heidelberg, Poland, 2010.

[31] Marusak, P. M, "Numerically efficient analytical MPC algorithm based on fuzzy Hammerstein model", *in Proc of 10th international conference on adaptive and natural computing algorithms*, vol (2), pgs 177-185, 2011.

[32] Miguel Martnez, Juan S. Senent& Xavier Blasco, "Generalized Predictive Control Using Genetic Algorithms (GAGPC)." *CiteSeerX__Engineering Applications of Artificial Intelligence*, vol 11(3), pp. 355-367, 1998.

[33] Mohd. Faudzi, Ahmad Athif, *“Implementation of generalized predictive control (GPC) for a real-time process control using Labview,”* MS thesis, Faculty of Electrical Engineering; Electrical-Mechatronics and Automatic Control, University Technology Malaysia, 2006.

[34] Murray-Smith, Ruderick, “*A local model network approach to nonlinear modeling”*, PhDthesis,Computer Science Department, University of Strathclyde, 1994.

[35] O. Nelles, M. Fischer, "Predictive control based on local linear fuzzy models", *International Journal of Systems sciences*, vol 29, pgs679-697, 1998.

[36] P. Barayani, I.M.Bavelaar, R. Babuska, L.C.Koczy, A. Titli and H.B.Verbruggen, "A method to invert a linguistic fuzzy model." *International Journal of Systems sciences*, vol 29(7), pgs 711-721, 1998.

[37] Qiang Li, B. Q., Zhiqiang Ge1and Xisheng Zhan, “Study of Fuzzy Generalized Predictive Control Algorithm On Nonlinear Systems”, in *1st International*

Conference on Innovative Computing, Information and Control (*ICICIC*), 2006.

[38] Qi_an Li, Shu_quing Wang, Jia Chu, "Direct algorithm for coefficients of Generalized Predictive Controller", *Information and Control*, vol 35(3), pgs319-323, 2006.

[39] Robert Babuska, *Fuzzy Modeling for Control*, Boston, Kluwer Academic Publishers, 1998.

[40] R. Babuska, J.M. Sousa and H.B. Verbruggen, *Inverse Fuzzy model based Predictive Control*, Springer, Heidelberg, 1998.

[41] R. Liutkevicius, "Fuzzy Hammersetin model for nonlinear plant", *Nonlinear Analysis: Modeling and Control*, vol 13, pgs201-212, 2008.

[42] Rossitor, *Model based Predictive Control __ A practical approach*, USA, CRC Press, Florida, 2004.

[43] Sadhana Chidrawar, B. Patre, "Generalized Predictive Control and Neural Generalized Predictive Control." *Leonardo Journal of Sciences*, vol 13, pgs 133-152, 2008.

[44] Sousa, J. Costa, *A Fuzzy approach to model based control*, Portugal,1998.

[45] Steven W. Su *et al*, “Nonparametric Hammerstein Model based Model Predictive Control for Heart Rate Regulation”, in *Proc. 29th Annual Int Conf.* , pp. 2984–2987, France, 2007.

[46] TAN, Wein Wan and Lo, Chang How, “Development of Feedback error learning strategies for training Neurofuzzy controllers online”, *in IEEE Transcation,* vol 1, pgs 7803-7293, 2001.

[47] T.Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control", *IEEE Transactions on Systems, Man and Cybernatics*, vol 15(1), pgs 116-132, 1985.

[48] T.T.C. Tsang and D.W. Clarke, "Generalized Predictive control with input constraints." *IEEE Proceedings,* vol135, 1998.

[49] Wang, L., *Model Predictive control system design and implementation using MATLAB*, Springer, 2009.

[50] Yaohua Hu and SuwuXu (2009), "Generalized Predictive controller Design for ship track keeping*". International Journal* on *Computer Science and Network Security*, vol. 9, Available: http://www.paper.ijcsns.org

[51] Yijing Wang, Z. Z., "A new Fast Algorithm of Generalized Predictive Control", *Pattern recognition and Artificial Intelligence*, vol 15(3), pgs 295-298, 2002.

[52] YingXu_Bin Liu and Kang-ling Fang, "Generalized Predictive Control based on Toeplitz equation", *in Proc. of the 7th World Congress on Intelligent Control and Automation*, Chongqing China, 2008.

[53] Yuanyu Jin, Xing-yuanGu, "Modified Generalized Predictive Control Algorithm", *in Jour of Information and Control*, vol 19(3), pgs 8-14, 1990.

[54] Zhiyu Xi and Tim Hesketh, "Ball and Beam system---Nonlinear MPC using Hammerstein model", in *Second IEEE Conf. on Industrial Electronics and Applications*, 2007.