University of Strathclyde Glasgow

Nonlinear Predictive Restricted Structure Control

Sultan Alotaibi

Wind Energy and Control Centre

Department of Electronic and Electrical Engineering

University of Strathclyde

Thesis submitted in fulfilment of the requirements for the degree of PhD

March 2021

Declaration

Signed: Sultan Alotaibi

Date: 12/02/2021

Preface

I want to express my thanks to my supervisor, Professor Michael Grimble, for his continuous guidance and patience during research years and in the thesis progression, and to Dr Reza Katebi, for his advice. I also would like to thank Dr Pawel Majecki for his assistance during algorithms implementation.

And a special thanks to my family for their support during years of hard work that brought me to this point.

Table of Contents

List of Tables

List of Figures

Nomenclature

ARMAX	Autoregressive Moving Average with Exogenous Input
CARMA	Controlled Autoregressive Moving Average
CSTR	Continuous Stirred Tank Reactors
DC	Direct Current
DMC	Dynamic Matrix Control
DOF	Degree of Freedom
EGR	Exhaust Gas Recirculation
ELC	Extended Linear Complementarity
EMF	Electro Magnetic Field
ETB	Electronic Throttle Body
ETC	Electronic Throttle Control
GMV	Generalized Minimum Variance
GPC	Generalised Predictive Control
GPFC	Generalised Predictive Functional Control
HA	Hybrid Automata
HS	Hybrid System
IAE	Integral of Absolute Error
IC	Internal Combustion
IMC	Internal Model Control
ISE	Integral of Squared Error
ITAE	Integral of Time multiply Absolute Error
LC	Linear Complementarity
LH	Limp Home
LHA	Linear Hybrid Automata

LPV	Linear Parameter Varying
LQR	Linear Quadratic Regulator
LQG	Linear Quadratic Gaussian
LTI	Linear Time-Invariant
LTV	Linear Time-Varying
MBPC	Model-Based Predictive Control
MIMO	Multiple Input Multiple Output
MIQP	Mixed-Integer Quadratic Programming
MLD	Mixed Logical Dynamical
MMPS	Max-Min Plus Scaling
MPC	Model Predictive Control
MR	Magneto-Rheological
MV	Minimum Variance
NGMV	Nonlinear Generalized Minimum Variance
NGPC	Nonlinear Generalised Predictive Control
NGPFC	Nonlinear Generalised Predictive Functional Control
NP	Nonlinear Programming
NPGMV	Nonlinear Predictive Generalized Minimum Variance
PFC	Predictive Functional Control
PI	Proportional Integral controller
PID	Proportional Integral Derivative controller
PWA	Piecewise Affine
PWM	Pulse Width Modulation
QFT	Quantitative Feedback Theory
qLPV	quasi Linear Parameter Varying
QP	Quadratic Programming

QTP	Quadruple Tank Process
RMSE	Root Mean Square Error
RS	Restricted Structure
SIMO	Single Input Multiple Output
SD	State Dependant
SI	Spark Ignition
TPS	Throttle Position Sensors
TVKF	Time-Varying *Kalman* Filter
VCT	Variable Camshaft Timing
VGT	Variable Geometry Turbocharger
VVT	Variable Valve Timing

Abstract

This thesis defines new developments in predictive restricted structure control for industrial applications. It begins by describing the augmented system for both *state-space* and polynomial model descriptions. These descriptions can contain the plant model, the disturbance model, and any additional essential model subsystems. It then describes the predictive restricted structure control solution for both linear and nonlinear systems in *state-space* form. The solution utilizes the recent development in nonlinear predictive generalized minimum variance by adding a general operator subsystem that defines nonlinear system along with the linear or the linear parameter varying output subsystem.

The next contribution is the polynomial predictive restricted structure control algorithm for polynomial linear parameter varying model that may result from nonlinear equations or experimental *data-driven* model identification. This algorithm utilizes the generalised predictive control method to approximate and control nonlinear systems in the linear parameter varying system *input-output* transfer operator matrices. The solution is simple in unconstrained and constrained optimization solutions and required a small computing capacity.

Four examples have been chosen to test the algorithms for different nonlinear characteristics. In the first three examples, state-space versions of the algorithm for the linear, the quasi-linear parameter varying and the state-dependent were employed to control the quadruple tank process, electronic throttle body, and the continuous stirred tank reactors. In the last example, the polynomial linear parameter varying restricted structure controller is used to control automotive variable camshaft timing system.

Chapter 1 Introduction

Classical *PID* control is very useful in the industry. It is applied successfully in various manufacturing sectors and is the most applied controller in industry. The survey in [1] for the industrial controllers' position has estimated that 97% of controllers used in the typical chemical plant are *PID*. The *PID* controller is generally used in its basic structure for ease of implementation in both the hardware and software. The controller structure simplicity and effectiveness are essential properties in manufacturing. This *structure-property* occasionally is more important than the adoption of other advanced control approaches.

The *PID* controller, even its digital version, is founded on *age-old* concepts and determined by the restrictions brought down by technologies of the time when it was implemented [2]. Therefore, the *PID* controller may be suitable for some enhancements. The *Ziegler-Nichols PID* tuning methods and *Smith* predictor concept were developed in the late 1950s [3]–[6]. Yet, if the dynamics of the systems are slightly complex, then *PID* control may deliver weak performance, and a *higher-order* controller may be necessary to deal with this and also the rigid multivariable system dynamics. This case is similar to when the *PID* controller performance drops for plants with *non-minimum* phase behaviour. A predictive feature is sometimes needed for the *PID* controller to cope with *non-minimum* phase and long *dead-time* behaviours [7].

The *PID* controller's derivative part can be deemed the predictive part, but this prediction property is not suitable once treating *non-minimum* phase systems. Hence, the *PI* control part is preserved, and prediction is carried out by implementing a process internal model simulation within the controller [8], where the *IMC* technique can be regarded as an enhancement of the *Smith* predictor [9], [10]. This controller expansion seems reasonable since it provides more design freedom by extending the concept of *PID* control.

For example, one idea for a simple scalar problem is to include other *frequency-domain* based terms in a *PID* structure as a double integrator or a time constant term. Of course, this is somewhat arbitrary, and a better scientific basis should be proposed to accomplish this type of enhancement. Another suggestion is to include certain selected functions that, in some way, can cover frequency responses ranges that might be required. When the functions selection method is considered, the ideas of *Richalet* related to *PFC* come to mind [11], [12], and the idea of utilizing functions as a ground for projecting the control signal or the manipulated variable is used in [13].

The controller retains the usual feedback structure form in this thesis and will be specified in terms of *frequency-sensitive* functions multiplied by gains that need to be calculated. When determining how to calculate the unknown gain vectors, it may be remembered that *GPC* is also very successful in the industry in its various *MBPC* forms.

Therefore, it is proposed that the gains be computed to minimize a *GPC* style *cost-function* so that optimal control is achieved but using the controller of *RS* introduced in [14]. This approach is not the same as the control strategy of *Richalet* that introduced the functions differently [11]. However, *PFC* does, of course, have its advantages, demonstrated in industry, such as simplicity [15].

The proposed controller will, therefore, have a traditional feedback control structure with functions that could, for instance, match the *PID* type terms. The background controller is very likely to be of *higher-order* to gain improved performance and to be able to handle multivariable systems in a more natural manner. The gains will be computed instantly and are *time-varying* in *receding-horizon* optimal control spirit. However, most of the structure of the controller will be fixed as in classical control so that the output of any integral terms offsetting disturbances can, for example, be monitored.

Several efforts have, of course, been made to blend both the *PID* benefits and predictive control. In [16], the design in a rather unusual method modified the *GPC* performance index by incorporating *PID* terms. One of the advantages of using the *GPC* method to computes the controller gains is to consider future variation in reference signal or any feedforward disturbance variations. As in predictive control, the system model will be needed to predict forward, and this, of course, means that the method does require system model information. Either *state-space* or polynomial based models could be used, therefore in the following Chapters, both the *state-space* and the polynomial descriptions are given.

1.1 Predictive PID Design

Many researchers adopted advanced control methods as *GPC* to reproduce *PID* controllers or other techniques to restrict the optimal controllers' structure to relate to a *PID* controller. The work in [10] brought in a *PID* controller based on the *IMC* concept for a *first-order* process, and the work in [17] broadened *IMC* based *PID* controller to address the *second-order* process.

The primary restriction of these designs was only on tuning methods which were calculated for systems without delay. [18], [19] shows that *IMC* conduct as *PID* controllers for most models in industrial applications. [20] employed *least-squares* algorithms to calculate the nearest comparable *PID* controller to an *IMC* design using the frequency response method. But, the design was weak when implemented on unstable or systems with time delay.

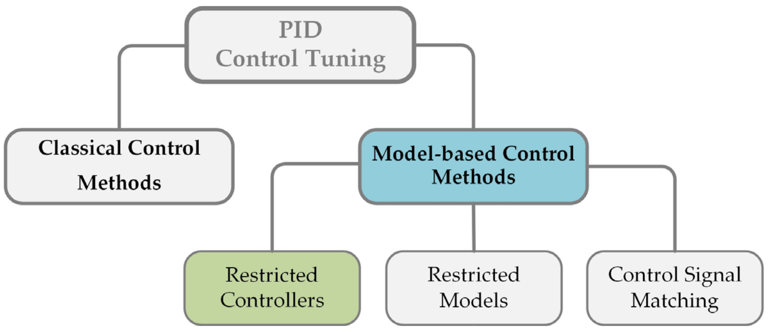

Figure 1-1: Predictive PID control methods

Three *MBPC* methods have been adopted for predictive *PID* control, as shown in Figure 1-1, and can be summarised as follow:

Restricted Models Methods: This approach applies the *MBPC* methodology but brings reductions on the plant model, and then the *MPC* law appears as *PID* control law. The approach is built based on the work introduced in [21].

Control Signal Matching Methods: This technique also employs complete *MBPC* for the solution with an altered *PID* control law with one coefficient set to produce future control increments. The *PID* controller gains are then selected to provide a tight match among predictive *PID* and *MPC* signals [22].

Restricted Controllers Methods: In this approach, the *PID* controller structure is chosen, and gains are determined to minimize a *GPC* cost. It works for any order or *MIMO* plant, so there is no need to assume a *low-order* plant or approximate the control signal [14].

RS control's research area is essential and *up-and-coming* since it acts as a bridge between modern control theory and classical methods. This thesis describes the work commenced to consider the possibility of progressing the *RS* control method toward multivariable and nonlinear systems since most systems are inherently nonlinear and always of interest to engineers and mathematicians.

1.2 Optimal Nonlinear Control

The *MV* control is an optimal control strategy established in [23], with the objective to minimize the *k-steps-ahead* output variance of the stochastic system given the information available in the current time instant, and the optimal *cost-function* involves the conditional expectation of the squared output of the system to obtain its variance. For the optimal control calculation, the *MV* is a *model-based* optimal scheme that uses the *CARMA* model to generate *k-steps-ahead* predictions of the process using the *Diophantine equation* [24]. The *MV* scheme inherently provides *dead-time* compensation and has properly lowered process output variability while considering measurable disturbances within the optimal control solution, handling the *MIMO* systems, and showing slight computational complexity. However, it is inappropriate for *non-minimum* phase systems where the design results in excessive control action and is shown to deliver a *sub-optimal* solution in the case of saturation constraints.

The initial idea of the *GMV* control was to develop controllers using a blend of optimal control theory commenced on *MV* control and the *well-known IMC* technique of [25] and inspired by the famous *Smith* predictor structures. The *GMV* design [26] is an extension that includes the weighted control action into the initial *MV cost-function*, and this provides better results in reducing control action variability. This upgrade aided the *GMV* in managing *non-minimum* phase systems, although it has exhibited poor stability behaviour for some *non-minimum* phase systems [27].

The *NGMV* controller aims to minimize a very similar *cost-function* to the *GMV*. The process model utilized to produce these signals within the *cost-function* can contain a general nonlinear operator, as described in [25]. The idea is that the plant model can be decomposed to delay terms set, a primarily stable general nonlinear subsystem, and a linear subsystem characterized in a linear *state-space* or polynomial matrix descriptions, which can contain unstable modes. The system output signal in the *cost-function* is generalized to a penalty on the error signal, and stabilizing control law is obtained by selecting a set of weightings. Compared to the error weighting, the control weighting can be nonlinear, which adds to nonlinear design flexibility.

Although the implemented *one-step-ahead* optimal control law in the *GMV* scheme may produce an internally stabilizing control law, that still not certain for particular control weighting selections, and again the controller fails for unstable and *non-minimum* phase systems, mainly for systems with badly identified delays. Further improvement was produced by [28] and [29] has led to the *GPC,* which has come to be one of the very admired *MPC* methods in industry and been effectively applied in many industrial sectors [30] due to its great performance and specific degree of robustness. The *GPC* was initially derived in a polynomial form. The *state-space* version was then obtained by [31], where the *GPC* optimal quadratic *cost-function* is solved within a specified prediction horizon based on the *receding-horizon* principle.

The *NGPC* builds upon the *MPC* idea well applied in the industry with proven performance and profitability advances. The predictive controller benefits by utilizing future reference signal or setpoint information to minimise a *multi-step* *cost-function*. This method becomes very useful in dealing with processes that comprise *time-varying* parameters, long *dead-times* or multivariable interactions. This controller utilizes nonlinear models to generate predictions and solve its optimal *cost-function* considering nonlinear input, output and state constraints. Therefore, the use of *NGPC* usually involves a *real-time open-loop* optimal control problem solution. The model fidelity and the states' estimation obtained through the *Kalman filter* are crucial aspects and necessary for generating the predictions. In this regard, the *LPV* approach can often improve system approximation and allow a broader range of systems to be adopted by the algorithm.

The *NPGMV* development was based on the *GPC* structure and shared some of the *GPC* algorithms characteristics. It shares similar stability assumptions and uses the same subsystem decomposition with the primary *NGMV* controller. Being a predictive control approach, it differs in that it utilizes a *multi-step cost-function* that includes future tracking error and control signal weightings in a *GPC* problem form [32]. Both the *NGMV* and *NGPC* controllers are special cases of *NPGMV,* and if the system is linear and the control cost term tends to zero, then the controller returns to those for a *GPC* controller.

The adaptation of *NGPC* and *NPGMV* to *LPV* systems has a very similar plant description. For *time-varying* systems and via proper subsystem representation, the goal is to provide a *model-based* controller with a *fixed-structure*. This system description decomposed in a general operator subsystem that can represent the hard nonlinearities and an *LPV* subsystem to capture other dynamics.

1.3 RS Control

This approach is different from restricted models and control signal matching methods which may be either unrealistic or involve approximations. Although order reduction of plant model became common lately, *reduced-order* models designed controllers can result in unstable *closed-loop* if utilized for the *larger-scale* system [33]. In *RS* control, the *PID* structure is often chosen, and the gains are determined to minimize a *GPC* cost. It works for any order or *MIMO* plant, so it is not like restricted models methods, and hence there is no need to assume a *low-order* plant. The *RS* solution doesn't intend to get close to *GPC* unrestricted control solution because approximating a control signal is not reliable or robust since it neglects some *closed-loop* properties [34]. The *RS* controller generates the best gains for the *PID* chosen in terms of the *cost-function* selected, and this can be any predictive type *cost-function* and not only *GPC*. Performance comparable to the *GPC* controller was developed by an optimal predictive *PID* control algorithm design [14] with a *single-DOF* control structure. The target was to create *PID* controllers tuning method via an online optimization that can function as a conventional *PID* controller or multivariable *GPC* controller. The *RS* controller approach introduced here is linked to the solutions above in the specific situation in which the functions are *PID* motivated functions, and if the controller for a scalar system is more general like:

$$C_0(z) = \frac{c_{0_n} + c_{1_n} z + \cdots + c_{n_n} z^{-n}}{c_{0_d} + c_{1_d} z + \cdots + c_{m_d} z^{-m}}$$

Then the results can apply in finding numerator and denominator coefficients instead of *PID* gains. However, the properties of the controller sought here are more related to extending the excellent properties of *PID* or classical control.

Since one of the strong motivating factors is that classical controllers of *low-order* and particularly *PID* controllers show natural robustness features when there are significant nonlinearities and uncertainties in the system. In this work, the problem's predictive aspects provide an easy method of calculating controller gains. This method is useful due to the added complexity introduced when using a wider set of controller functions that possibly substantially extend the *PID* functions.

Noting that the *cost-function* in [14] was also restricted to being of traditional *GPC* form, and in this case, the *black-box* input subsystem was not present. Thus, the control strategy explored in this thesis provides a framework with a very general system description that may employ a *black-box* model of the plant, and this can be practical for real applications where access to system information is limited for various reasons. The idea here is to formulate a *RS* controller to accommodate a wide class of nonlinear systems without undergoing major modifications within its core architecture for different system characteristics. To be practical, this controller should use a model of the process where this is available or represent the process using a *black-box* model when the physical structure is unknown, or even a polynomial *LPV* model results from an experimental *data-driven LPV* identification. The methods considered in this thesis are briefly described in the next sections.

1.3.1 State-Space RS Control

For this method, a predictive *RS* control solution is given to control *discrete-time state-space MIMO* systems. The plant model is represented by two subsystems of different types that offer modelling flexibility. The input subsystem includes an operator of a general linear form where no structural information is needed or available except the subsystem's output can be calculated for a given input. The output subsystem is a *state-space* model that links the output to the input subsystem and can be linear, *SD*, *LPV*, or *qLPV*. The *PWA* system can also be turned into the equivalent *SD* system by inserting a custom logic state to describe the intersection's switching sections.

The controller structure is given in *RS* form, which involves a selected linear *transfer-functions* set, and gains set obtained to minimise a *GPC* or *NPGMV cost-index*. In the simplest case, the terms of the functions can be chosen as in the *PID* control. The linear version and the equivalent *cost-index* of the *RS* control are given in Chapter 4. The nonlinear *RS* control for *LPV state-space* systems and the extension for a subclass of *HS* are given in Chapter 5.

1.3.2 Polynomial RS Control

For this method, a predictive *RS* control solution is given for *discrete-time MIMO* systems. The system is characterised by an *LPV* subsystem that may be unstable and represented in polynomial matrix form. The *pre-specified low-order RS* controller is parameterised in terms of discrete transfer operators set chosen by the designer and a set of optimised parameters or gains. The controller within the feedback loop can have a general parameter varying form or just a *PID*. The predictive control *multi-step cost-index* minimised includes terms in the magnitude, gains rate of change, and other terms. The resulting simple *RS* controller is covered in Chapter 6 and has the control capabilities that only the *long-range* predictive control algorithm normally provides.

1.4 Research Goals and Objectives

The main objectives as they evolved through this work can be summarised as:

- Develop the predictive *RS* solutions group for different nonlinear systems classes defined in the *state-space* representation.
- Develop the predictive *RS* solution for nonlinear systems defined in the *LPV* polynomial representation.
- Demonstrate the benefits of the solutions on application examples chosen from process control and automotive type areas. Implement and test the applicability and performance of both linear and nonlinear predictive *RS* control algorithms in simulation.

1.5 Thesis Contributions

The contributions from this work can be outlined as follows:

- Different structures were explored regarding the improvement of the *RS* algorithm's implementation and implemented successfully in the simulation.
- The *qLPV* and *PWA* systems adaptation for the *state-space RS* algorithm was derived mathematically and implemented in simulation
- The polynomial *LPV* adaptation for the polynomial *RS* algorithm was derived mathematically and implemented in simulation.
- Four *RS* solutions for the linear, *qLPV*, *PWA* and polynomial *LPV* application were designed, and suggestion to enhance the robustness is outlined in Section 8.2.

1.6 Thesis Organization

Chapter 1: Provides an introduction survey on predictive *RS* control methods and identifies the place for this work within this framework.

Chapter 2: Introduces the augmented system description within the *state-space* framework and provides some of the algorithm's leading design aspects.

Chapter 3: Introduces the augmented system description within a polynomial framework and provides some of the algorithm's leading design aspects.

Chapter 4: Describes the linear *RS* algorithm's derivation for linear *state-space* and unstructured subsystems. Additionally, it gives suggestions to enhance its computational implementation aspects like execution speed.

Chapter 5: Describes the nonlinear *RS* algorithm's derivation and its adaptation to *LPV* systems in the *state-space* framework. Moreover, a solution extension toward an *HS* class is provided, where the *PWA* system is converted into its equivalent *SD* system representation.

Chapter 6: Defines the polynomial *LPV RS* algorithm's derivation for nonlinear systems represented in the *LPV* system *input-output* transfer operator matrices.

Chapter 7: Implements and assesses the performance of the predictive version of the *RS* algorithm on four examples: the quadruple tank process, electronic throttle body, continuous stirred tank reactors and variable camshaft timing.

Chapter 8: Provides a conclusion on predictive *RS* control methods and suggest some future work and recommendations.

Appendix: Provides the augmented system description derivation and the *Matlab* and *Simulink* code developed for applications in Chapter 7.

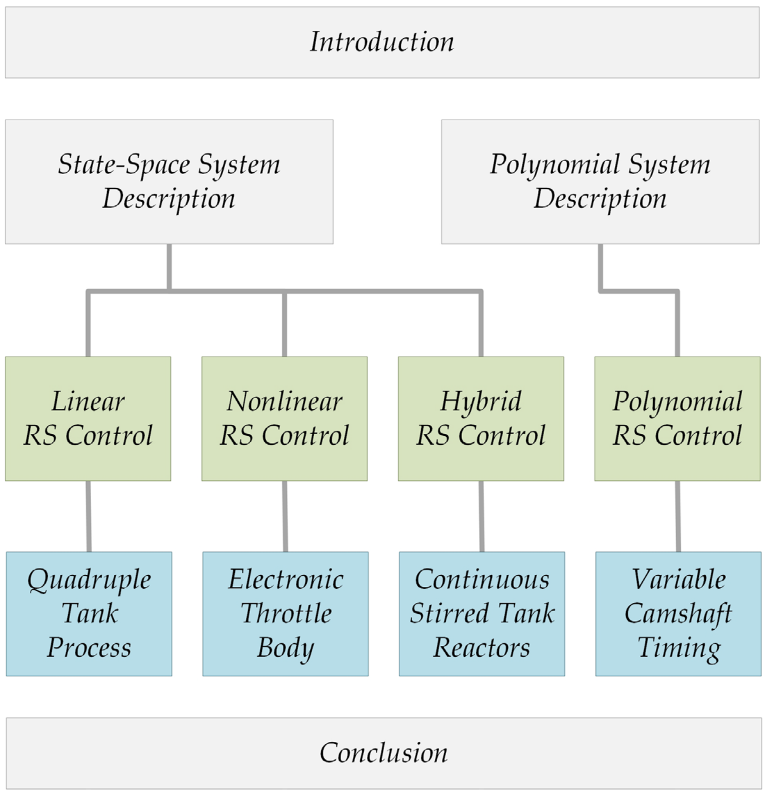

Figure 1-2: Thesis flow overview

The diagram in Figure 1-2 gives a brief overview of how the thesis flows.

1.7 List of Publications

- S. Alotaibi, M. Grimble, 2019, Nonlinear Optimal Generalized Predictive Functional Control Applied to Quasi-LPV Model of Automotive Electronic Throttle, The 15th IEEE International Conference on Control and Automation, Edinburgh, UK.
- S. Alotaibi, M. Grimble, 2019, Optimal Generalized Predictive Functional Control with Applications to Extended PID Control for Quadruple Tank, The 25th IEEE International Conference on Automation and Computing, Lancaster, UK.
- M. Grimble, S. Alotaibi and P. Majecki, 2021, Restricted Structure Polynomial Systems Approach to LPV Generalized Predictive Control, The 7th IFAC Conference on Nonlinear Model Predictive Control, Bratislava, Slovakia.
- S. Alotaibi, M. Grimble and L. Cavanini, 2021, Nonlinear Optimal Generalized Predictive Functional Control of Piecewise Affine Systems, The 29th IEEE Mediterranean Conference on Control and Automation, Puglia, Italy.

Chapter 2 State-Space System Description

This Chapter defines the system description for which the *RS* controllers have been developed. It begins with the linear *state-space* system description and then its nonlinear version. The *state-space* approach adaptation to *NPGMV* algorithm for *state-space* systems, introduced by [35], [36], is used as a basis for the method described in the following sections:

Section 2.1: Introduction to *state-space* prediction model; *vector-matrix* notation; linear prediction equations; *predictor-corrector Kalman filter*.

Section 2.2: Introduction to *qLPV/SD* subsystem models; *state-space* prediction models; *vector-matrix* notation; *time-varying* prediction equations; *TVKF*.

The subsystems illustrated in Figure 2-1 contain their equivalent mathematical representations, which can vary depending on the problem, i.e., *transfer-functions*, *state-space* models etc. For this work, both model descriptions are used. The input plant subsystem W_{1k} is a general linear operator form where no structure is assumed. The mathematical representations of these various subsystems W_r, W_d, W_0, W_{1k} and P_c represent input weighting, disturbance, linear plant, plant input operator and error weighting subsystems, respectively, are described in the next section.

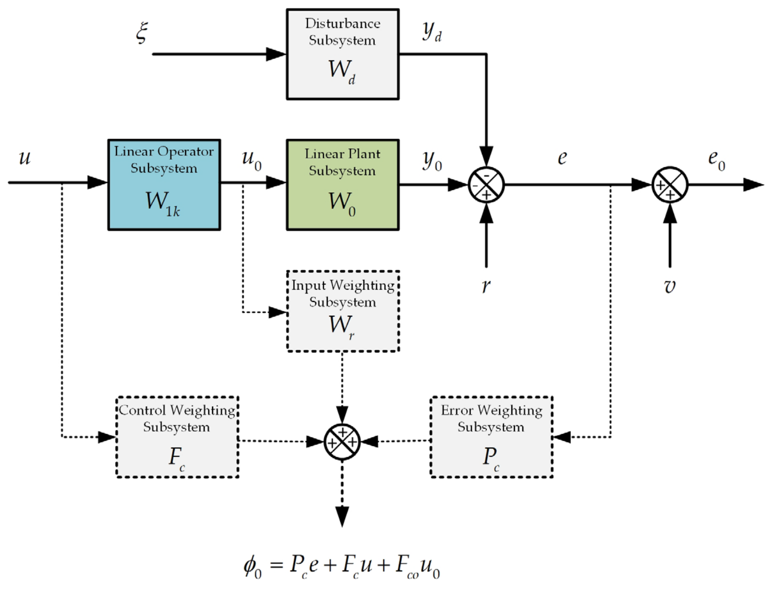
ξ
Disturbance Subsystem
W_d
y_d
u
Linear Operator Subsystem
W_{1k}
u_0
Linear Plant Subsystem
W_0
y_0
e
e_0
r
v
Input Weighting Subsystem
W_r
Control Weighting Subsystem
F_c
Error Weighting Subsystem
P_c
$\phi_0 = P_c e + F_c u + F_{co} u_0$

Figure 2-1: System description subsystems layout

Input Weighting:

$$y_r(t) = W_r u_0(t)$$

Output Disturbance:

$$y_d(t) = W_d \xi(t)$$

Plant Subsystem:

$$y_0(t) = W_0 u_0(t)$$

Error Weighting:

$$e_p(t) = P_c e(t)$$

It is useful in some problems to have an alternative mechanism to introduce a control costing or a dynamic weighting on the input $u_0(t-k)$ signal. If a dynamic costing on the input signal is applied, it can be treated as part of the system model.

The disturbance elements are described by an *LTI* model excited by *white-noise,* and a deterministic output disturbance part symbolized $d(t)$. Furthermore, the reference $r(t)$ is a deterministic signal. Hence, both signals are expected given during the predictive control horizon to be used. The feedback system depicted in Figure 2-2 consists of a linear plant model, reference, measurement noise and disturbance signals, and the measurement *white-noise* signal $v(t)$, which is expected to have a fixed covariance matrix $R_f = R_f^T \geq 0$. And this is fine since there is no generality loss if the *white-noise* $\xi(t)$ is assumed to have an identity covariance matrix. Both noise signals $v(t)$ and $\xi(t)$ are independent Gaussian *zero-mean white-noise* vectors. There is no necessity to identify the noise sources distribution as the particular system form conducts a prediction equation, which just depends on the disturbance model.

The plant first "input" subsystem (2.1) may also be introduced if needed to give more modelling flexibility and can be a linear operator structure where there is no structure assumed:

$$(W_1 u)(t) = z^{-k} (W_{1k} u)(t) \tag{2.1}$$

where z^{-k} indicates common delay components *diagonal-matrix* in all output signals lines with $k > 0$. The first subsystem W_{1k} output is $u_0(t) = (W_{1k}u)(t)$, and W_1 is anticipated as a stable subsystem. However, the other linear "output" subsystem in (2.2) can include any unstable modes and will be described in more detail in the next.

$$W_0 = z^{-k} W_{0k} \tag{2.2}$$

The first "input" subsystem is defined $W_{1k} = I$ if the first linear subsystem component is not needed and not in use.

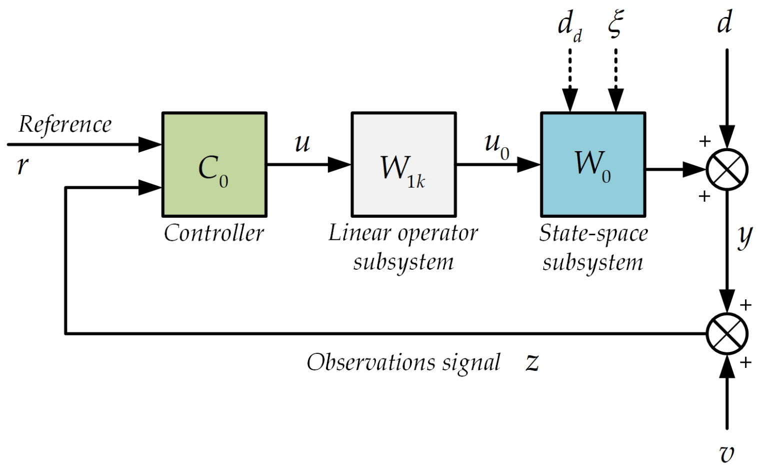

Figure 2-2: RS control of unstructured and state-space subsystems

The generalization to various delays in various signal lines can make the solution difficult. Still, the work in [37] simplifies the solution. Finally, the dynamic *cost-function* of the weighted error $e_p(t) = P_c(z^{-1})e(t)$ is stable.

2.1 Linear State-Space Subsystem

The initial specified *LTI* subsystems are the *state-space* model of the disturbance W_d and the plant W_0 subsystems. Consider first, this *state-space* subsystem illustrated in Figure 2-3 and connected to the system measured outputs. The augmented model comprises the error dynamic weighting $P_c(z^{-1})$ and a disturbance model.

The structure here is quite general, and the way a common industrial problem is expressed in this augmented form is explained in Appendix: A-1 Augmented System Matrices.

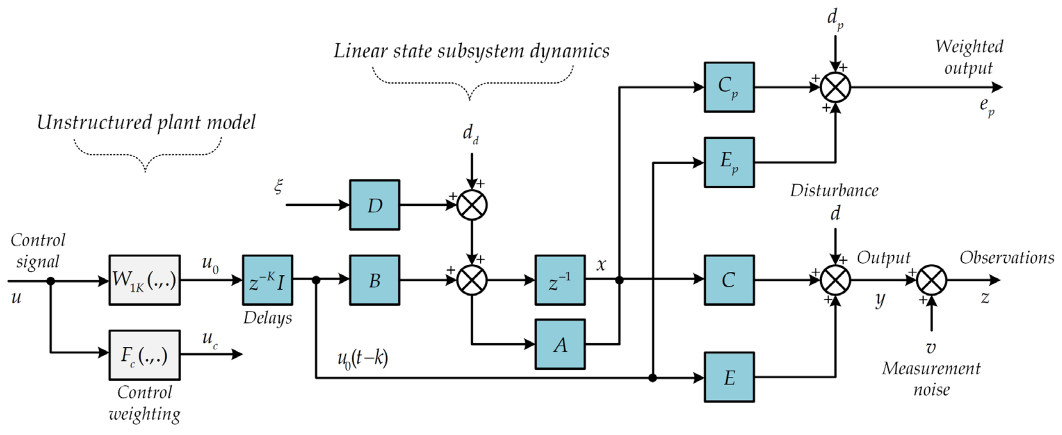

Figure 2-3: General and linear state-space subsystems

The model of the augmented linear subsystem assumed a *stabilizable, detectable* and given in the *state-space* form as:

$$x(t+1)=A\ x(t)+Bu_0(t-k)+D\xi(t)+d_d(t) \tag{2.3}$$

$$y(t)=d(t)+C\,x(t)+E\,u_0(t-k) \tag{2.4}$$

$$z(t)=d(t)+C\,x(t)+E\,u_0(t-k)+v(t) \tag{2.5}$$

The through term matrix E here can be specified to be full rank if the explicit delay totals *k-steps*. The weighted error that will be minimized is:

$$e_p(t)=d_p(t)+C_p\,x(t)+E_p\,u_0(t-k) \tag{2.6}$$

where matrices A,B,C,D,E,C_p,E_p are fixed, and the linear subsystem *delay-free* plant transfer can be given as:

$$W_{0k}=E+C\Phi B \tag{2.7}$$

where

$$\Phi=(zI-A)^{-1}$$

Signal Descriptions: The plant model input channels are expected to comprise a *k-steps transport-delay,* and the vectors signals are given as follows:

$x(t)$	Plant subsystem along with disturbance model n system states
$u_0(t)$	Linear subsystem m input signals.
$u(t)$	Input subsystem m control signals.
$y(t)$	Measured r plant output signals.
$z(t)$	Measured r plant output with measurement noise.
$r(t)$	Plant r setpoint signals.
$e_p(t)$	m *inferred* controlled weighted error signals.
$d(t)$	Measured r output disturbance signal.

2.1.1 State-Space Prediction Models

It is essential to have the output prediction to build the control algorithm in Chapter 4. Hence, the states and output future values, at time *t*, are found via a recursive *state-equation* as:

$$\begin{aligned}
&x(t+1)= \\
&Ax(t)+Bu_0(t-k)+D\xi(t)+d_d(t)
\end{aligned}$$

$$\begin{aligned}
&x(t+2)= \\
&A\left(Ax(t)+Bu(t)+D\xi(t)+d_d(t)\right)+Bu_0(t+1-k)+D\xi(t+1)+d_d(t+1)= \\
&A^2x(t)+ABu(t)+Bu_0(t+1-k)+AD\xi(t)+D\xi(t+1)+Ad_d(t)+d_d(t+1)
\end{aligned}$$

$$\begin{aligned}
&x(t+3)= \\
&Ax(t+2)+Bu_0(t+2\text{-}k)+D\xi(t+2)+d_d(t+2)= \\
&A^3x(t)+A2Bu_0(t-k)+ABu_0(t+1-k)+Bu_0(t+2-k)+ \\
&A^2D\xi(t)+AD\xi(t+1)+D\xi(t+2)+A^2d_d(t)+Ad_d(t+1)+d_d(t+2)
\end{aligned}$$

The previous result can be generalized to acquire the state in $t+i$ future times as in (2.8), where $i>0$.

$$x(t+i)=A^i x(t)+\sum_{j=1}^{i} A^{i-j}\left(B u_0(t+j-1-k)+D\xi(t+j-1)+d_d(t+j-1)\right) \quad (2.8)$$

Noting that, future state's calculation needs the *state-vector* and future inputs at time *t*, and the future states time can be found by applying time shifting in (2.8) using the explicit *transport-delay k-steps* as:

$$\begin{aligned} x(t+i+k)=&A^i\, x(t+k)+\\ &\sum_{j=1}^{i} A^{i-j}\left(B u_0(t+j-1)+D\xi(t+j+k-1)+d_d(t+j+k-1)\right) \end{aligned} \quad (2.9)$$

The future times controlled weighted error $e_p(t)$ has the formation in (2.10) for $i\geq 1$ and can comprise any stable *cost-function* dynamic weighting.

$$\begin{aligned} e_p(t+i+k)=&d_p(t+i+k)+C_p\, A^i x(t+k)+\\ &\sum_{j=1}^{i} C_p\, A^{i-j}\left(B u_0(t+j-1)+D\xi(t+j+k-1)+d_d(t+j+k-1)\right)+\\ &E_p\, u_0(t+i) \end{aligned} \quad (2.10)$$

The deterministic disturbance signal terms can be collected as:

$$d_{pd}(t+i+k)=d_p(t+i+k)+\sum_{j=1}^{i} C_p\, A^{i-j}\, d_d(t+j+k-1) \quad (2.11)$$

Noting (2.10) the weighted error $e_p(t)$ is:

$$\begin{aligned} e_p(t+i+k)=&d_{pd}(t+i+k)+C_p\, A^i\, x(t+k)+\\ &\sum_{j=1}^{i} C_p\, A^{i-j}\left(B u_0(t+j-1)+D\xi(t+j+k-1)\right)+E_p\, u_0(t+i) \end{aligned} \quad (2.12)$$

2.1.2 Vector and Matrix Notation

For controls in the period $\tau \in [t, t+N]$, the computed weighted errors for $N > 0$ can be accumulated in the $N+1$ vector notation (2.13) as:

$$\begin{bmatrix} e_p(t+k) \\ e_p(t+1+k) \\ e_p(t+2+k) \\ \vdots \\ e_p(t+N+k) \end{bmatrix} =$$

$$\begin{bmatrix} d_{pd}(t+k) \\ d_{pd}(t+1+k) \\ d_{pd}(t+2+k) \\ \vdots \\ d_{pd}(t+N+k) \end{bmatrix} + \begin{bmatrix} C_p I \\ C_p A \\ C_p A^2 \\ \vdots \\ C_p A^N \end{bmatrix} x(t+k) +$$

$$\begin{bmatrix} 0 & 0 & \cdots & 0 \\ C_p B & 0 & \ddots & 0 \\ C_p AB & C_p B & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ C_p A^{N-1} B & C_p A^{N-2} B & \cdots & C_p B \end{bmatrix} \begin{bmatrix} u_0(t) \\ u_0(t+1) \\ \vdots \\ \vdots \\ u_0(t+N-1) \end{bmatrix} + \begin{bmatrix} E_p u_0(t) \\ E_p u_0(t+1) \\ E_p u_0(t+2) \\ \vdots \\ E_p u_0(t+N) \end{bmatrix} +$$

$$\begin{bmatrix} 0 & 0 & \cdots & 0 \\ C_p D & 0 & \ddots & 0 \\ C_p AD & C_p D & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ C_p A^{N-1} D & C_p A^{N-2} D & \cdots & C_p D \end{bmatrix} \begin{bmatrix} \xi(t+k) \\ \xi(t+1+k) \\ \vdots \\ \vdots \\ \xi(t+N-1+k) \end{bmatrix} \tag{2.13}$$

And (2.13) is given as:

$$E_{P_{t+k,N}} = D_{P_{t+k,N}} + C_{PN} A_N x(t+k) + (C_{PN} B_N + E_{PN}) U^0_{t,N} + C_{PN} D_N W_{t+k,N} \tag{2.14}$$

The next *block-matrices* and vectors can be specified for a more typical case for $N > 0$ like in (2.15).

$$A_N = \begin{bmatrix} I \\ A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \qquad B_N = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ B & 0 & \cdots & \vdots & 0 \\ \vdots & B & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & 0 \\ A^{N-1}B & A^{N-2}B & \cdots & B & 0 \end{bmatrix}$$

$$D_N = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ D & 0 & \ddots & 0 \\ AD & D & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ A^{N-1}D & A^{N-2}D & \cdots & D \end{bmatrix}, D_{P_{t,N}} = \begin{bmatrix} d_{pd}(t) \\ d_{pd}(t+1) \\ d_{pd}(t+2) \\ \vdots \\ d_{pd}(t+N) \end{bmatrix}, \tag{2.15}$$

$$W_{t,N} = \begin{bmatrix} \xi(t) \\ \xi(t+1) \\ \vdots \\ \xi(t+N-1) \end{bmatrix}, \qquad U^0_{t,N} = \begin{bmatrix} u_0(t) \\ u_0(t+1) \\ \vdots \\ u_0(t+N) \end{bmatrix}$$

$$C_{PN} = diag\{C_p, C_p, \ldots, C_p\}\, and\, E_{PN} = diag\{E_p, E_p, \ldots, E_p\}, square\ (N+1)$$

The expression (2.15) can be defined as $A_N = I, B_N = D_N = 0, C_{PN} = C_p$ and $E_{PN} = E_p$ for the particular situation of *cost-function* with *single-step* at $N = 0$.

The transfer $W_{t,N}$ is the future *white-noise* inputs vector and $U^0_{t,N}$ is the future input signals block vector. The tracking error $E_{P_{t+k,N}}$ that incorporates dynamic costing weighting *k-steps-ahead* can, therefore, given using (2.14) in the vector:

$$E_{P_{t+k,N}} = D_{P_{t+k,N}} + C_{PN} A_N x(t+k) + V_{PN} U^0_{t,N} + C_{PN}\, D_N\, W_{t+k,N} \tag{2.16}$$

The weighted error to be minimized in the solution considered later is expected to be in the control signal's equivalent dimension. The matrix V_{PN} used in (2.16) is defined below and is therefore of a *lower-triangular* and square block as in (2.17) for $N > 0$.

$$V_{PN} = C_{PN} B_N + E_{PN}$$
$$= \begin{bmatrix} E_p & 0 & \cdots & 0 & 0 \\ C_p B & E_p & \cdots & \vdots & 0 \\ \vdots & C_p B & \ddots & \vdots & \vdots \\ C_p A^{N-2} B & \vdots & \ddots & E_p & 0 \\ C_p A^{N-1} B & C_p A^{N-2} B & \cdots & C_p B & E_p \end{bmatrix} \tag{2.17}$$

For the particular issue of *single-stage cost-function*, at horizon $N = 0$ and noting (2.13), the matrix (2.17) is given $V_{PN} = E_p$, where E_p implies the through term among input signal $u_0(t-k)$ and weighted output.

2.1.3 Linear Prediction Equations

The output signal *i-steps-ahead* prediction can be determined by remarking the previous result (2.10) and believing for the current that the control action future values are established and known.

Thus, the predicted weighted signal to be minimized is given in (2.18) utilizing (2.12) and this if $\hat{e}_p(t+i+k|t) = E\{\hat{e}_p(t+i+k)|t\}$.

$$\begin{aligned} \hat{e}_p(t+i+k\,|\,t) = {} & d_{pd}(t+i+k) + C_p\, A^i\, \hat{x}(t+k|t) + \\ & \sum_{j=1}^{i} C_p A^{i-j} B u_0(t+j-1) + E_p u_0(t+i) \end{aligned} \tag{2.18}$$

where $\hat{x}(t+k|t)$ is a *Kalman filter least-squares* state estimate outcome. By adding together the outcomes for when $N > 0$ then the predicted errors vector $\hat{E}_{P_{t+k,N}}$ is acquired in the next block matrix given in (2.19).

$$\begin{bmatrix} \hat{e}_p(t+k|t) \\ \hat{e}_p(t+1+k|t) \\ \hat{e}_p(t+2+k|t) \\ \vdots \\ \hat{e}_p(t+N+k|t) \end{bmatrix} = \underbrace{\begin{bmatrix} d_{pd}(t+k) \\ d_{pd}(t+1+k) \\ d_{pd}(t+2+k) \\ \vdots \\ d_{pd}(t+N+k) \end{bmatrix}}_{D_{P_{t+k,N}}} + \underbrace{\begin{bmatrix} C_p I \\ C_p A \\ C_p A^2 \\ \vdots \\ C_p A^N \end{bmatrix}}_{C_{PN} A_N} \hat{x}(t+k|t) +$$

$$\underbrace{\begin{bmatrix} E_p & 0 & \cdots & 0 & 0 \\ C_p B & E_p & \cdots & \vdots & 0 \\ \vdots & C_p B & \ddots & \vdots & \vdots \\ C_p A^{N-2} B & \vdots & \ddots & E_p & 0 \\ C_p A^{N-1} B & C_p A^{N-2} B & \cdots & C_p B & E_p \end{bmatrix}}_{V_{PN} = C_{PN} B_N + E_{PN}} \underbrace{\begin{bmatrix} u_0(t) \\ u_0(t+1) \\ \vdots \\ \vdots \\ u_0(t+N) \end{bmatrix}}_{U^0_{t,N}} \tag{2.19}$$

The vector of $N+1$ *step-ahead* prediction (2.19) is given as in (2.20):

$$\hat{E}_{P_{t+k,N}} = D_{P_{t+k,N}} + C_{PN}\, A_N\, \hat{x}(t+k|t) + V_{PN} U^0_{t,N} \tag{2.20}$$

And the prediction error of the output $\tilde{E}_{P_{t+k,N}} = E_{P_{t+k,N}} - \hat{E}_{P_{t+k,N}}$ is:

$$= C_{PN} A_N x(t+k) + V_{PN} U^0_{t,N} + C_{PN} D_N W_{t+k,N} - (C_{PN} A_N \hat{x}(t+k|t) + V_{PN} U^0_{t,N}) \tag{2.21}$$

Thereafter, the *inferred* output estimation error:

$$\tilde{E}_{P_{t+k,N}} = C_{PN}\, A_N\, \tilde{x}(t+k\,|\,t) + C_{PN}\, D_N\, W_{t+k,N} \tag{2.22}$$

where $\tilde{x}(t+k|t) = x(t+k) - \hat{x}(t+k|t)$ is a *k-steps-ahead state-estimation* error. Note for later use that if plant and model mismatch is ignored, the *state-estimation* error doesn't depend on a selection of control action. The orthogonality of the optimal $\hat{x}(t+k|t)$ and $\tilde{x}(t+k|t)$, and the expectation of the control action future values result (believed given in obtaining prediction equation) and the *white-noise* exciting signals are null. We conclude that the predicted signals $\hat{E}_{P_{t+k,N}}$ vector in (2.20) and the prediction error $\tilde{E}_{P_{t+k,N}}$ are orthogonal.

2.1.4 Kalman Filter Predictor-Corrector Form

The state estimation $\hat{x}(t+k\,|\,t)$ can be generated for *k-steps-ahead* in a practical computation approach employing a *Kalman filter* [38]. In this case, the filter states quantity are not grown by the explicit delays *k* size, and the *Kalman filter* algorithm is summarised briefly as:

$$\begin{aligned}&(Predictor)\\ &\hat{x}(t+1|t)=A\,\hat{x}(t|t)+B\,u_0(t-k)+d_d(t)\\ &(Corrector)\\ &\hat{x}(t+1|t+1)=\hat{x}(t+1|t)+K_f\left(z(t+1)-\hat{z}(t+1|t)\right)\end{aligned} \tag{2.23}$$

where K_f is the *Kalman filter* gain and:

$$\hat{z}(t+1|t)=d(t+1)+C\hat{x}(t+1|t)+Eu_0(t+1-k) \tag{2.24}$$

The desired prediction equation follows:

$$\hat{x}(t+k|t)=A^k\,\hat{x}(t|t)+T_0(k,z^{-1})B\,u_0(t) \tag{2.25}$$

$T_0(k,z^{-1})$ is a finite impulse response block, $T_0(0,z^{-1})=I$ and for $k\geq 1$

$$T_0(k,z^{-1})=(I-A^k z^{-k})(zI-A)^{-1}=z^{-1}(I+z^{-1}A+z^{-2}A^2+\ldots+z^{-k+1}A^{k-1}) \tag{2.26}$$

From equations in (2.23) and (2.24), the optimal estimate can now be given in the next expression:

$$\begin{aligned}\hat{x}(t+1|t+1)=A\hat{x}(t|t)+Bu_0(t-k)+&\\ K_f\left(z(t+1)-d(t+1)-CA\hat{x}(t|t)-CBu_0(t-k)-Eu_0(t+1-k)\right)&\end{aligned}$$

$$\begin{aligned}\hat{x}(t|t)&=\left(I-z^{-1}(I-K_fC)A\right)^{-1}\left(\left(Bz^{-1}-K_fCBz^{-1}-K_fE\right)u_0(t-k)+K_f\left(z(t)-d(t)\right)\right)\\ &=\left(I-z^{-1}(I-K_fC)A\right)^{-1}\left(K_f\left(z(t)-d(t)\right)-z^{-k}\left(K_fE+(K_fC-I)Bz^{-1}\right)u_0(t)\right)\end{aligned}$$

The previous expression can, thus, be given in a more brief form:

$$\hat{x}(t|t) = T_{f_1}(z^{-1})\Big(z(t) - d(t)\Big) + T_{f_2}(z^{-1})u_0(t) \tag{2.27}$$

The linear transfer operators:

$$T_{f_1}(z^{-1}) = \Big(I - z^{-1}\big(I - K_f C\big)A\Big)^{-1} K_f \tag{2.28}$$

$$T_{f_2}(z^{-1}) = -\Big(I - z^{-1}\big(I - K_f C\big)A\Big)^{-1} z^{-k}\Big(K_f E + \big(K_f C - I\big)Bz^{-1}\Big) \tag{2.29}$$

Unbiased estimates-property: To have an unbiased *Kalman filter*:

$$T_{f_1}(z^{-1})\Big(C\Phi(z^{-1})B + E\Big)z^{-k} + T_{f_2}(z^{-1}) = \Phi(z^{-1})Bz^{-k} \tag{2.30}$$

Identity: This practical outcome in (2.30) may easily be verified using (2.28) and (2.29), as:

$$\begin{aligned}
&T_{f_1}(z^{-1})\Big(C\Phi(z^{-1})B + E\Big)z^{-k} + T_{f_2}(z^{-1}) \\
&= \Big(I - z^{-1}(I - K_f C)A\Big)^{-1}\Big[K_f\Big(C\Phi(z^{-1})B + E\Big) - \Big(K_f E + (K_f C - I)Bz^{-1}\Big)\Big]z^{-k} \\
&= \Big(I - z^{-1}(I - K_f C)A\Big)^{-1}\Big[K_f C z^{-1}\Big(I - z^{-1}A\Big)^{-1} - \Big(K_f C - I\Big)z^{-1}\Big]Bz^{-k} \\
&= \Big(I - z^{-1}(I - K_f C)A\Big)^{-1}\Big[K_f C - \Big(K_f C - I\Big)\Big(I - z^{-1}A\Big)\Big]\Big(I - z^{-1}A\Big)^{-1} Bz^{-k-1} \\
&= \Big(I - z^{-1}A\Big)^{-1} Bz^{-k-1}
\end{aligned}$$

Or

$$T_{f_1}(z^{-1})\Big(C\Phi(z^{-1})B + E\Big)z^{-k} + T_{f_2}(z^{-1}) = \Phi(z^{-1})Bz^{-k} \tag{2.31}$$

2.2 qLPV/SD State-Space Subsystem

LPV systems fall into the broader range of *gain-scheduling* methods, but they differ in the way they represent and capture a system's dynamics. Therefore, they are not only static approximations around some operating points. The general formulation of an *LPV* system is given in (2.32).

$$\begin{aligned}\dot{x}(k+1)&=A\big(\rho(k)\big)x(k)+B\big(\rho(k)\big)u(k)\\ y(k)&=C\big(\rho(k)\big)x(k)+E\big(\rho(k)\big)u(k)\end{aligned} \tag{2.32}$$

where x, u, y are the state, input and output system vectors individually. These systems are linear, choosing their matrices for a *LTI* system but are nonlinear as matrices of the system are *SD*. Varying parameters may also represent *Inter-state* dependencies to capture the system's dynamics. These parameters are generally considered bounded within a set of values $\Delta\rho\left\{\rho : R \geq 0 \rightarrow \Delta\rho\right\}$ in the literature and used to describe the following:

- An exogenous measured *time-varying* parameter (*LPV*).
- An endogenous system states (*qLPV*).

LPV systems were primarily formed within the framework of *gain-scheduling* control design for representing nonlinear systems [39]. The *LPV* approach brought design a step closer to a nonlinear system design to improve the *gain-scheduling* method. *Gain-scheduling* entails switching via interpolation among linear controllers set obtained from a system linearisation at multiple points across the operating range [39]. Exogenous vs endogenous distinction becomes clear when classifying *LPV* systems formulations into general *LPV* or *qLPV*.

The *qLPV* systems accommodate *state-variables* as the varying parameters and provide an excellent approximation to the original nonlinear response. An important observation in the understanding of *LPV* systems is that their structure resembles that of *LTV* systems. This structure is handy in predictive control as it is useful in increasing future states' fidelity and outputs generation.

The feedback system illustrated in Figure 2-4 includes a *SD* or *qLPV* model, collectively including measurement noise, disturbance and reference signals. The measurement *white-noise* signal $v(t)$ is expected, as in the previous section, to have a fixed covariance matrix $R_f = R_f^T \geq 0$. The measurement noise signals $v(t)$ and $\xi(t)$ are independent Gaussian *zero-mean white-noise* signals vectors.

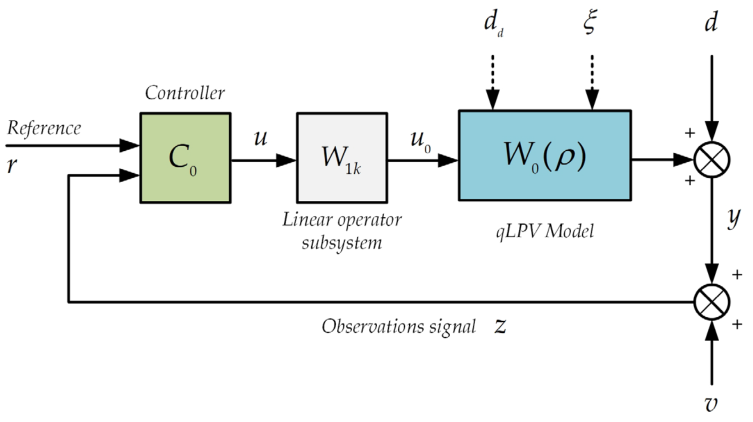

Figure 2-4: Nonlinear control for qLPV systems

The *SD* or *qLPV* plant model can now be described, and this comprises the plant, disturbance, error weighting, and control weighting subsystems. Further details on how the augmented system is generated for the nonlinear plant are given in Appendix: A-1 Augmented System Matrices.

The augmented system here is said to be pointwise *stabilizable* and *detectable* and given the next form:

$$x(t+1)=A_t x(t)+B_t u_0(t-k)+D_t \xi(t)+d_d(t) \tag{2.33}$$

$$y(t)=d(t)+C_t x(t)+E_t u_0(t-k) \tag{2.34}$$

$$z(t)=d(t)+C_t x(t)+E_t u_0(t-k)+v(t) \tag{2.35}$$

$$e_p(t)=d_p(t)+C_{p_t} x(t)+E_{p_t} u_0(t-k) \tag{2.36}$$

The state matrices A_t, B_t, C_t, D_t and E_t illustrated in Figure 2-5 can be *SD, LPV, qLPV* or just *LTV*. Different delays in various channels can be included. However, this complicates the analysis, so the delay terms are expected identical in all input channels (delay length k).

The matrix E_{p_t} is expected full rank, while the *cost-function* to be specified will comprise a *full-rank* dynamic control weighting term. The input subsystem W_{1k} can be removed by setting $W_{1k}=I$ in this case so that $u_0(t)=u(t)$ for the analysis in Chapter 5.

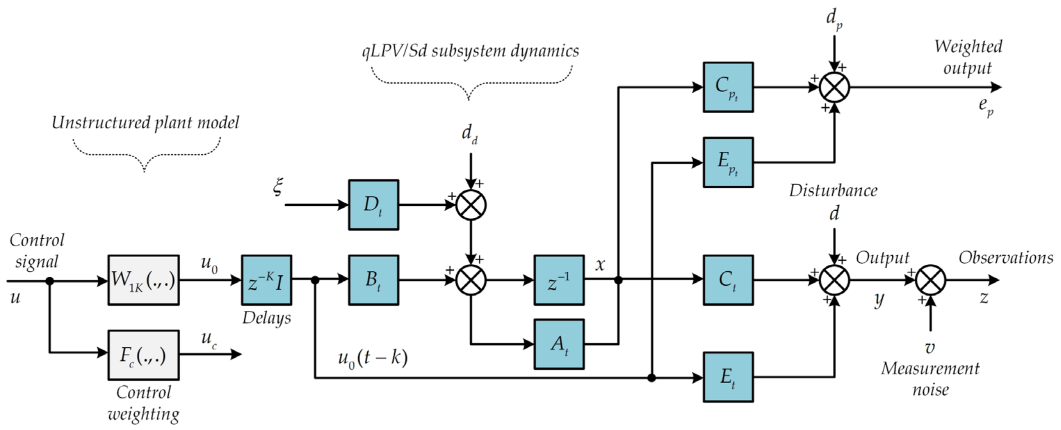

Figure 2-5: Unstructured model and qLPV/SD state-space subsystem

2.2.1 State-Space Prediction Models

For the system matrices future values to be estimated, the deterministic disturbance part is expected given. The next *state-equation* is used to recursively acquire the future state values:

$$x(t+1) = A_t x(t) + B_t u_0(t-k) + D_t \xi(t) + d_d(t)$$

The *state-vector* in *i-steps-ahead* expression is acquired as:

$$\begin{aligned} x(t+i) = {} & A_{t+i-1}A_{t+i-2}\dots A_t x(t) + \\ & \sum_{j=1}^{i} A_{t+i-1}A_{t+i-2}\dots A_{t+j}\left(B_{t+j-1}u_0(t+j-1-k) + D_{t+j-1}\xi(t+j-1) + d_d(t+j-1)\right) \end{aligned}$$

Moreover, it may be arranged more concisely using the next notation:

$$\begin{aligned} x(t+i) = {} & A_t^i x(t) + \\ & \sum_{j=1}^{i} A_{t+j}^{i-j}\left(B_{t+j-1}u_0(t+j-1-k) + D_{t+j-1}\xi(t+j-1)\right) + d_{dd}(t+i-1) \end{aligned} \quad (2.37)$$

where

$$\begin{array}{ll} A_{t+m}^{i\text{-}m} = A_{t+i-1}A_{t+i-2}\dots A_{t+m} & \mathit{for}\ i > m \\ A_t^i = A_{t+i-1}A_{t+i-2}\dots A_t & \mathit{for}\ i > 0 \\ A_{t+m}^0 = I & \mathit{for}\ i = m \\ A_t^0 = I & \mathit{for}\ i = 0 \end{array}$$

The predicted measured disturbance in (2.37) is described as in (2.38).

$$\left\{ \begin{array}{ll} d_{dd}(t+i-1) = \sum_{j=1}^{i} A_{t+j}^{i-j} d_d(t+j-1) & \mathit{for} \quad i > 0 \\ d_{dd}(t-1) = 0 & \mathit{for} \quad i = 0 \end{array} \right\} \quad (2.38)$$

The controlled weighted error signal $e_p(t)$ follows similarly at future times as in (2.39) for $i \geq 0$.

$$
\begin{aligned}
e_p(t+i) &= d_p(t+i) + C_{p_{t+i}} x(t+i) + E_{p_{t+i}} u_0(t+i-k) \\
&= d_{pd}(t+i) + C_{p_{t+i}} A_t^i x(t) + C_{p_{t+i}} \sum_{j=1}^{i} \left(A_{t+j}^{i-j} B_{t+j-1} u_0(t+j-1-k) \right) + \\
& C_{p_{t+i}} \sum_{j=1}^{i} \left(A_{t+j}^{i-j} D_{t+j-1} \xi(t+j-1) \right) + E_{p_{t+i}} u_0(t+i-k)
\end{aligned}
\tag{2.39}
$$

where the deterministic signals in (2.39) are merged as in (2.40)

$$
d_{pd}(t+i) = d_p(t+i) + C_{p_{t+i}} d_{dd}(t+i-1) \tag{2.40}
$$

2.2.2 Vector and Matrix Notation

The weighted error prediction is necessary for the next control solution in Chapter 5. Therefore, the state *i-steps* prediction is achieved by applying a time shifting using the explicit *k-steps transport-delay*:

$$
\begin{aligned}
& x(t+i+k) = A_{t+k}^i x(t+k) + \\
& \sum_{j=1}^{i} A_{t+k+j}^{i-j} \left(B_{t+k+j-1} u_0(t+j-1) + D_{t+j-1} \xi(t+j+k-1) \right) + d_{dd}(t+k+i-1)
\end{aligned}
\tag{2.41}
$$

where

$$
d_{dd}(t+k+i-1) = \sum_{j=1}^{i} A_{t+k+j}^{i-j} d_d(t+k+j-1) \tag{2.42}
$$

The control action future variations are calculated at every iteration; however, future variations of the parameters have to be approximated. Gathering the deterministic disturbance signal components together in $d_{pd}(t+i+k)$

$$
d_{pd}(t+i+k) = d_p(t+i+k) + C_{p_{t+i+k}} d_{dd}(t+k+i-1) \tag{2.43}
$$

Then the weighted error $e_p(t)$ prediction is acquired as in (2.44)

$$
\begin{aligned}
& e_p(t+i+k)=d_{pd}(t+i+k)+C_{p_{t+i+k}}A_t^i x(t+k)+ \\
& \sum_{j=1}^{i} C_{p_{t+i+k}} A_{t+j}^{i-j}\Big(B_{t+j-1}u_0(t+j-1)+D_{t+j-1}\xi(t+j+k-1)\Big)+ \\
& E_{p_{t+i+k}}u_0(t+i)
\end{aligned}
\tag{2.44}
$$

Introducing in *vector-matrix* notation as in (2.45)

$$
\overbrace{\begin{bmatrix} e_p(t+k) \\ e_p(t+1+k) \\ e_p(t+2+k) \\ \vdots \\ e_p(t+N+k) \end{bmatrix}}^{E_{P_{t+k,N}}}
=
\overbrace{\begin{bmatrix} C_{pt}I \\ C_{pt}A_t \\ C_{pt}A_t^2 \\ \vdots \\ C_{pt}A_t^N \end{bmatrix}}^{C_{PN}A_N}
x(t+k)+
$$

$$
\overbrace{\begin{bmatrix}
E_{pt} & 0 & \cdots & 0 & 0 \\
C_{pt}B_t & E_{pt} & \ddots & 0 & 0 \\
\vdots & C_{pt}B_t & \ddots & \vdots & 0 \\
C_{pt}A_t^{N-1}B_t & \vdots & \ddots & E_{pt} & \vdots \\
C_{pt}A_t^{N-1}B_t & C_{pt}A_t^{N-2}B_t & \cdots & C_{pt}B_t & E_{pt}
\end{bmatrix}}^{V_{PN}}
\overbrace{\begin{bmatrix} u(t) \\ u(t+1) \\ \vdots \\ \vdots \\ u(t+N-1) \end{bmatrix}}^{U_{t,N}}
+
$$

$$
\overbrace{\begin{bmatrix}
0 & 0 & \cdots & 0 \\
C_{pt}D_t & 0 & \ddots & 0 \\
C_{pt}A_tD_t & C_{pt}D_t & \ddots & \vdots \\
\vdots & \vdots & \ddots & 0 \\
C_{pt}A_t^{N-1}D_t & C_{pt}A_t^{N-2}D_t & \cdots & C_{pt}D_t
\end{bmatrix}}^{C_{PN}D_N}
\overbrace{\begin{bmatrix} \xi(t+k) \\ \xi(t+1+k) \\ \vdots \\ \vdots \\ \xi(t+N-1+k) \end{bmatrix}}^{W_{t+k,N}}
\tag{2.45}
$$

2.2.3 Time-Varying Prediction Equations

The prediction of error signal *i-steps-ahead* is determined by using the previous results and assuming the current and future control actions are given. Thus, the predicted weighted signal, to be minimized by utilizing (2.39), is given in (2.46) if $\hat{e}_p(t+i+k|t)=E\{\hat{e}_p(t+i+k)|t\}$.

$$e_p(t+i+k)=d_{pd}(t+i+k)+C_{p_{t+i+k}}A_t^i\hat{x}(t+k)+ \\ \sum_{j=1}^{i}C_{p_{t+i+k}}A_{t+j}^{i-j}\left(B_{t+j-1}u_0(t+j-1)+D_{t+j-1}\xi(t+j+k-1)\right)+E_{p_{t+i+k}}u_0(t+i) \tag{2.46}$$

where $\hat{x}(t+k\,|\,t)$ is a *TVKF least-squares* state estimate outcome. This and the *qLPV* model are utilized, and the delays are adjusted for input channels [38]. Finally, weighted error signal estimation is accumulated in a vector structure of size $N+1$ as given in (2.47) or (2.48).

$$\hat{E}_{P_{t+k,N}}=D_{P_{t+k,N}}+\overbrace{C_{P_{t+k,N}}A_{t+k,N}}^{C_{PN}A_N}\hat{x}(t+k|t)+\overbrace{(C_{P_{t+k,N}}B_{t+k,N}+E_{P_{t+k,N}})}^{V_{PN}}U_{t,N}^0 \tag{2.47}$$

$$\hat{E}_{P_{t+k,N}}=\overbrace{D_{P_{t+k,N}}+C_{PN}A_N\hat{x}(t+k|t)}^{\tilde{D}_{P_{t+k,N}}}+V_{PN}U_{t,N}^0 \tag{2.48}$$

2.2.4 Time-Varying Kalman Filter

To estimate the *qLPV* model states, a *Kalman* predictor is necessary. The *i-steps* predicted states and weighted errors are developed from *time-varying* A_t, B_t, C_t and D_t matrices and a *time-varying* P_t error covariance matrix, so a derivation of a *TVKF* Kf_t gain factor [40]. Remarking that the stochastic disturbance is expected *zero-mean*. The *TVKF* expression is summarised in (2.49):

$$\begin{aligned}&(Predictor)\\&\hat{x}(t+1|t)=A_t\hat{x}(t|t)+B_t u_0(t-k)+d_d(t)\\&(Corrector)\\&\hat{x}(t+1|t+1)=\hat{x}(t+1|t)+Kf_{t+1}\left[z(t+1)-\hat{z}(t+1\,|\,t)\right]\end{aligned} \tag{2.49}$$

where

$$\hat{z}(t+1|t)=d(t+1)+C_t\hat{x}(t+1|t)+E_t u_0(t+1-k) \tag{2.50}$$

The *Kalman filter* gain and *Riccati* expressions are given in (2.51) and (2.52) for a system that has a plant and measurement noise covariance's Q_t and R_t:

$$Kf_{t+1} = P_{t+1|t} C_{t+1}^T \left[C_{t+1} P_{t+1|t} C_{t+1}^T + R_{t+1} \right]^{-1} \tag{2.51}$$

$$\begin{aligned} &(\textit{Priori covariance}) \\ &P_{t+1|t} = A_t P_{t|t} A_t^T + D_t Q_t D_t^T \\ &(\textit{Posteriori covariance}) \\ &P_{t+1|t+1} = P_{t+1|t} - Kf_{t+1|t} C_{t+1} P_{t+1|t} \end{aligned} \tag{2.52}$$

The controlled system is believed c*ontrollable* and *observable* from system noise inputs. Likewise, the tracking error *k-steps-ahead* in (2.44) can suitably be given as in (2.53).

$$E_{P_{t+k,N}} = D_{P_{t+k,N}} + C_{PN} A_N x(t+k) + V_{PN} U_{t,N}^0 + C_{PN} D_N W_{t+k,N} \tag{2.53}$$

The prediction error (2.54) is given based on (2.47) and (2.53):

$$\tilde{E}_{P_{t+k,N}} = E_{P_{t+k,N}} - \hat{E}_{P_{t+k,N}} = C_{PN}\, A_N\, \tilde{x}(t+k|t) + C_{PN}\, D_N\, W_{t+k,N} \tag{2.54}$$

where the state estimation error (2.55) doesn't depend on the control signal:

$$\tilde{x}(t+k|t) = x(t+k) - \hat{x}(t+k|t) \tag{2.55}$$

2.3 Chapter Summary

This Chapter summarised the *discrete-time state-space* system description used in both Chapter 4 and Chapter 5 to obtain the linear and nonlinear *RS* control solutions. The *state-space* representations of *discrete-time* systems are necessary for the *RS* controller and discussed in Section 2.1 for the linear case and Section 2.2 for the nonlinear case. An introduction to the *state-space* prediction models, *vector-matrix* notation; prediction equations; *Kalman filter* for both cases has been given.

Chapter 3 Polynomial System Description

This Chapter briefly describes the system description for which the polynomial *LPV RS* controllers has been developed. The prior work [37] on the polynomial approach is used here as a basis for the system description across the following sections. However, it is significantly different from when plant and controllers were understood to be *LTI*. The system model here is represented by a *LPV* plant and disturbance model in the polynomial matrix form, which is unusual and may have advantages over *state-space* models for some applications.

3.1 Introduction

In real life, many systems to be controlled are nonlinear. One technique that can be used to cope with nonlinear systems is *gain-scheduling* control. The classical *gain-scheduling* approach is carried out by designing local linear controllers based on linearising the nonlinear system at numerous operating points. Then an overall nonlinear controller is found by interpolating or scheduling among the designed local operation points. Still, the controlled system may have a poor response or turn out to be unstable when operating points are not at equilibrium [41].

To surmount the limitations of the method of classical *gain-scheduling*, the gains scheduling controller design based on *LPV* systems was introduced [42], [43]. The *LPV* system has a linear system form. It can be in a *state-space* description or system transfer operator matrices in *input-output* representation that depends on the measured scheduling parameters' static functions.

Compared to classical *gain-scheduling* methods, the *gain-scheduling* controller approach based on *LPV* systems can guarantee stability and optimal performance over the system's complete range of operation. Although there are attractive features of *LPV* systems, it has to be mentioned that the control system *closed-loop* performance depends on the *LPV* model quality being utilized for controller synthesis. Assuming that the plant's nonlinear dynamic equations are available, there are several methods to construct *LPV* models from them. The nonlinear terms can be made linear in parameters by linearisation, replacing them with functions using varying linear parameters. However, the models may end up with too many scheduling parameters with these techniques and then became too complicated and unsuitable for *LPV* controller design [44].

3.1.1 LPV Model Identification

Instead of forming *LPV* models from nonlinear equations, experimental system identification, as explained in Figure 3-1, can be considered an alternative. In system identification, the objective is to estimate model dynamics precisely from measured *input-output* data. The *LTI* systems identification framework is now well established and well known [45], while the identification of *LPV* models is still in development [46]–[50], and several problems and challenges have not yet been solved [51].

Hence, most *LPV input-output* models share the same disadvantage as they are not in a structure ready to be used by advanced controller synthesis methods. Therefore, the identified models have to be converted to *state-space* forms. How the *LPV input-output* systems can be effectively grasped in *state-space* descriptions is a frequently cited problem. This problem arises from the point that in *LPV* literature, *LPV* systems modelling and *discrete-time* identification is commonly realized via *input-output* model forms.

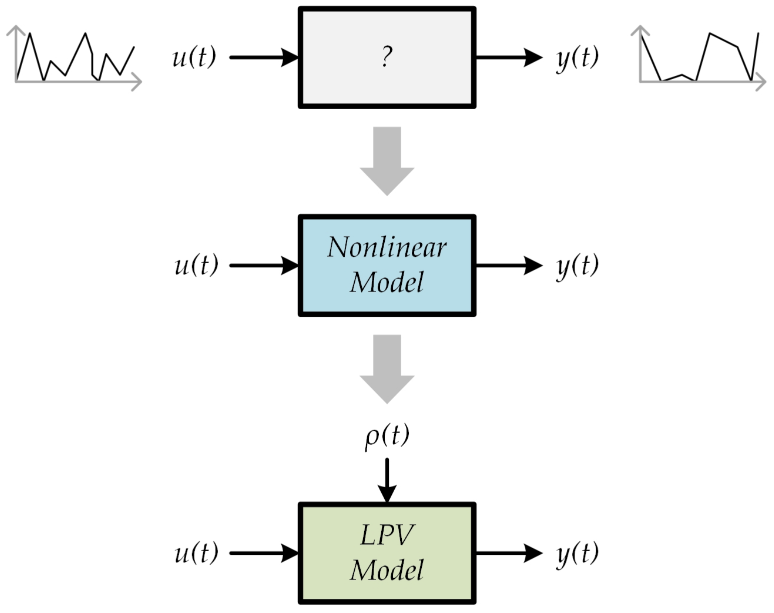

Figure 3-1: LPV model identification

The *LPV input-output* models are typically needed to be converted in equivalent *state-space* representation to be employed in control design. Such conversion is problematic, anticipated due to the dynamic dependence phenomenon addressed in [52]. The equivalence between *LPV state-space* and *LPV input-output* representations can be obtained by allowing a dynamic mapping among the system matrices and scheduling parameters. Still, this increases the complexity of the resulting *state-space* models. This complexity suggests direct use of the *LPV* polynomial models considered later.

The *LPV input-output* models system identification becomes an active research area [53], [54]. During *closed-loop* operation, identification of a plant is inspired by the necessity of plant changes observation without opening the loop [55]–[57], especially if the controlled plant was *open-loop* unstable.

3.1.2 Generalized Predictive Control

The *GPC* approach initiated by [28] is a very common *MPC* approach that has been effectively employed in different industrial applications [30]. Based on experience, it provides decent performance and a reasonable robustness level. The stochastic *GPC* controller is linked to the famous *MV* controller, defined in the *non-adaptive* version [23], which shaped the foundation of the *well-known self-tuning* controller [58], an edition of which was also studied by [59]. The *MV* controller is attained by minimizing (3.1) for a provided linear *input-output* model.

$$J = E\left\{ (y_{t+1} - r_{t+1})^2 \right\} \tag{3.1}$$

This control scheme succeeds for minimum phase systems such as stable plant zeros systems and deteriorates from requiring extreme control action for *non-minimum* phase plants so it can minimize output variance. An alternative proposed to get equivalent schemes to operate for *non-minimum* phase systems is to alter marginally (3.1) by inserting a penalty on the control action and the output as in (3.2).

$$J = E\left\{ (y_{t+1} - r_{t+1})^2 + \lambda u_t^2 \right\} \tag{3.2}$$

This new criterion was termed *GMV* control [26] and carried out an optimal control law *one-step-ahead*. This scheme delivers an internally stabilizing control law. Yet, it was not assured for certain selections of λ; it failed for certain *non-minimum* phase and unstable systems, especially for systems that have a weakly defined delay. A further upgrade was given by [28] and [29] and preceded to the *GPC,* which minimizes (3.3).

$$J = E\left\{ \sum_{j=N_1}^{N_2} \left[y_{t+j} - r_{t+j} \right]^2 + \lambda \sum_{j=1}^{N_u} \left[\Delta u_{t+j-1} \right]^2 \right\}$$
$$s.t. \Delta u_{t+i} = 0, \quad i = N_u, \dots N_2 \tag{3.3}$$

This minimization brings out $\Delta u_t, \Delta u_{t+1}, \dots \Delta u_{t+N_u-1}$ and just Δu_t is employed. At the time $t+1$, a different solution for the minimization problem is carried out. The control is seen as *receding-horizon,* and the optimization is executed on a number of $N_2 - N_1 + 1$ future outputs by a number of N_u future incremental control actions with an assumption that the delay is admitted in the window between N_1 and N_2. Mostly N_1 is selected so that the delay is the lower value of the delay approximate [60].

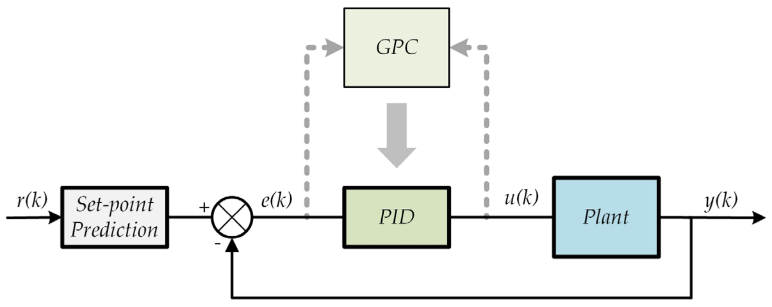

Figure 3-2: GPC background calculation

Many researchers adopted *GPC* to reproduce *PID* controllers, as illustrated in Figure 3-2. Some techniques restrict the optimal controllers' structure to being a *PID* controller. In [10], a *PID* controller based on *IMC* was used for the *first-order* process, and also in [17], it broadened for an *IMC* based *PID* controller to address the *second-order* process. The primary restriction of these designs was only on tuning methods which were calculated towards systems without delay.

[18], [19] have demonstrated that *IMC* conducts as *PID* controllers for most models in industrial applications. In [20], *least-squares* algorithms were used to calculate the nearest comparable *PID* controller to an *IMC* design following the frequency response approach. Still, the design was weak when implemented on unstable systems or with time delays. In the *RS* Control method [14], the *PID* controller's structure is decided, and gains are found to minimize *GPC* cost.

In the next sections, a brief description will be provided for the polynomial system description upon which the polynomial *LPV RS* controllers have been developed in Chapter 6.

3.1.3 The System Model

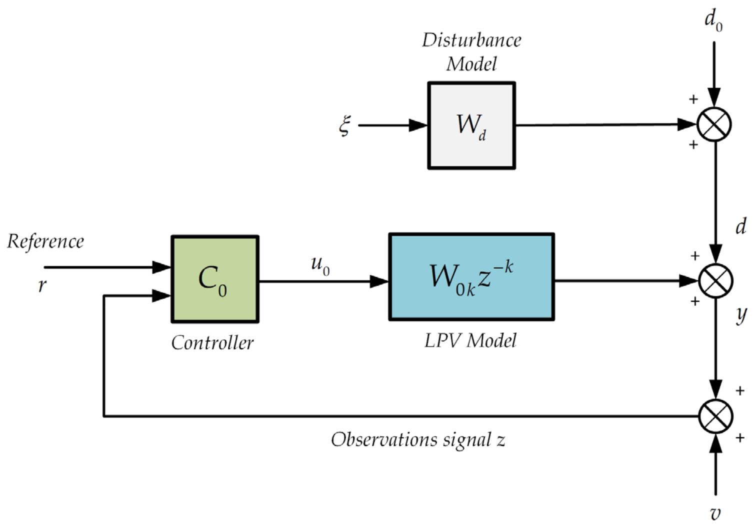

Figure 3-3: Feedback control system for polynomial LPV model

The feedback system is illustrated in Figure 3-3 and comprises the nonlinear plant, the measurement noise, the disturbance and the reference models. The system model is denoted W_{0k} and represents the *LPV* model, which may have unstable modes in the system.

Noting that, generalization to various delays in various lines is similar to that illustrated in [37]. The disturbance model is selected to be *LPV*, which is not limiting, as in several applications, the disturbance model is approximated as *LTI*. The measurement noise $v(t)$ and stochastic disturbance $\xi(t)$ are a vector of independent *zero-mean white-noise* signals with a fixed covariance matrix $R_f = R_f^T \geq 0$. Because of the problem formation, which includes a prediction equation that depends only on the *LPV* stochastic disturbance model, there is no generality loss if the disturbance *white-noise* driving source $\xi(t)$ has an identity covariance matrix.

3.2 LPV Polynomial Matrix Models

LPV system models that take account of parameter variations or variations with states and inputs are more common for models represented in a *state-equation* form. For example, there is quite a body of work in the solution of *SD* modelling and the use of iteratively computed *steady-state Riccati* equation solutions. There is less published on *ARMAX* modelling where the parameters vary with external influences such as wind speed, altitude or engine speed. However, since engineers often analyze systems' behaviour in *steady-state* conditions, where a given nonlinear operating point applies, it is common to use *frequency-domain* analysis methods. The use of what might be termed a *transfer-operator* model description which is a function of such external variables, is very natural for many application areas.

The system output $y(t)$ includes disturbance term $d_0(t)$ that is known, and hence to generate the prediction equations, consider the output without this term:

$$y_d(t) = y(t) - d_0(t) \tag{3.4}$$

The $r \times m$ *MIMO LPV* system model in a polynomial matrix form is given in the next, and the model explaining the *LPV* model of the plant may be stated in an *LPV CARMA* model as:

$$A\left(z^{-1}, \rho_t\right) y_d(t) = B_{0k}\left(z^{-1}, \rho_t\right) u_0(t-k) + C_d\left(z^{-1}, \rho_t\right) \xi(t) \tag{3.5}$$

The *parameter-vector* ρ_t, for simplicity $\rho_t = \rho(t)$, in this model represents a set of *time-varying* parameters. The model input channels comprise a *k-steps* $k \geq 0$ *transport-delay* that may be written as $B_0(z^{-1}, \rho_t) = B_{0k}(z^{-1}, \rho_t) z^{-k}$.

The *LPV* system *delay-free* plant *transfer-operator* and a model of the disturbance are specified in the left coprime *unit-delay* operator as:

$$\left[W_{0k}\left(z^{-1},\rho_t\right) \quad W_d\left(z^{-1},\rho_t\right)\right] = A^{-1}\left(z^{-1},\rho_t\right)\left[B_{0k}\left(z^{-1},\rho_t\right) \quad C_d\left(z^{-1},\rho_t\right)\right] \tag{3.6}$$

For *LTI* models, the coprime model prevents one from obtaining a *reduced-order* model with unstable pole/zero cancellations, and a similar sense is adopted for this factorization [51], [61].

Weighted Output: For later use in the definition of the *cost-function*, a dynamic *cost-function* weighting model, which has to be stable, is created in the left coprime polynomial matrix in (3.7).

$$P_C\left(z^{-1},\rho_t\right) = P_{c_d}^{-1}\left(z^{-1},\rho_t\right)P_{c_n}\left(z^{-1},\rho_t\right) \tag{3.7}$$

The weighted output without the known disturbance using the error weighting may be given as:

$$\begin{aligned} y_p(t) &= P_c\left(z^{-1},\rho_t\right)y_d(t) \\ &= P_c\left(z^{-1},\rho_t\right)\left\{A^{-1}\left(z^{-1},\rho_t\right)\left[B_{0k}\left(z^{-1},\rho_t\right)u_0(t-k) + C_d\left(z^{-1},\rho_t\right)\xi(t)\right]\right\} \end{aligned} \tag{3.8}$$

Sometimes, the above polynomial matrices arguments are omitted to simplify notation.

Disturbance Model: The stochastic part of the disturbance model is usually *LTI,* and the power spectrum $f = W_d\xi + v$ of both merged noise signal and the stochastic disturbance is processed as:

$$\Phi_{ff} = \Phi_{dd} + \Phi_{vv} = W_d W_d^* + R_f$$

The W_d adjoint notation indicates $W_d^*(z^{-1},\rho_t) = W_d^T(z,\rho_t)$, and only here, z is the *z-domain* complex number (elsewhere z^{-1} represents *unit-delay* operator).

The generalized spectral factor Y_f satisfies $Y_f Y_f^* = \Phi_{ff}$, where $Y_f = A^{-1}D_f$. The system models and factorization are expected to be as that D_f^{-1} is strictly stable transfer operator matrix [62], [63] and satisfies:

$$D_f D_f^* = C_d C_d^* + AR_f A^* \tag{3.9}$$

If the stochastic disturbance model is *LPV*, this is similar to a *LTV* system and the factorization in (3.9) applies, but the factor D_f is *time-varying*.

Innovations Signal: Usually, in real applications, the disturbance models are estimated by *LTI* systems excited by *white-noise*. The signal: $f = W_d\xi + v$ can now be modelled in an innovations signal style $f(t) = Y_f\,\varepsilon(t)$, where $Y_f = A^{-1}D_f$ is specified via the *spectral-factorization* (3.9) and $\varepsilon(t)$ implies a *zero-mean white-noise* signal with an identity covariance matrix [32].

Therefore, the measured signals are given in *LPV* system model terms, using (3.5) and an innovations signal as in (3.10).

$$\begin{aligned} z(t) &= y(t) + v(t) \\ &= d_0(t) + A^{-1}\left(z^{-1},\rho_t\right)B_{0k}\left(z^{-1},\rho_t\right)u_0(t-k) + Y_f\left(z^{-1},\rho_t\right)\varepsilon(t) \end{aligned} \tag{3.10}$$

Let the signals weighted by the error weighting *spectral-factor* $P_c(z^{-1},\rho_t)$ which is needed in the *cost-function*, be defined as::

$$\begin{aligned} y_p(t) &= P_c\left(z^{-1},\rho_t\right)\left(y(t) - d_0(t)\right) \\ v_p(t) &= P_c\left(z^{-1},\rho_t\right)v(t) \\ d_p(t) &= P_c\left(z^{-1},\rho_t\right)d_0(t) \end{aligned} \tag{3.11}$$

Identifying the error weighted spectral factor right coprime model as:

$$P_c\left(z^{-1},\rho_t\right)Y_f\left(z^{-1},\rho_t\right) = D_{fp}\left(z^{-1},\rho_t\right)A_f^{-1}\left(z^{-1},\rho_t\right) \tag{3.12}$$

Then the weighted, measured signal is:

$$z_p(t)=P_c\left(z^{-1},\rho\right)z(t)=y_p(t)+d_p(t)+v_p(t) \tag{3.13}$$

From (3.10), this is written as:

$$\begin{aligned} z_p(t)=d_p(t)+P_c\left(z^{-1},\rho_t\right)W_{0k}\left(z^{-1},\rho_t\right)u_0(t-k)+ \\ D_{fp}\left(z^{-1},\rho\right)A_f^{-1}\left(z^{-1},\rho_t\right)\varepsilon(t) \end{aligned} \tag{3.14}$$

Optimal Predictor: A *least-squares* predictor for *LPV* systems may be used to compute the inferred output y at times: $t+k+1$, $t+k+2$ .... Thus, the *cost-function* to be minimized and utilized to define the *least-squares* predictor is:

$$J=E\left\{\tilde{y}_p\left(t+j|t\right)^2\right\} \tag{3.15}$$

where the estimation error:

$$\tilde{y}_p\left(t+j|t\right)=y_p\left(t+j\right)-\hat{y}_p\left(t+j|t\right) \tag{3.16}$$

Also, $\hat{y}_p(t+j|t)$ describes $y_p(t)$ predicted value at a time *j-steps-ahead*. And to produce the prediction algorithm, a *Diophantine equation* solution is needed for the *LPV* solution (E_j,H_j), with E_j of smallest degree $\deg\left(E_j(z^{-1},\rho_t)\right)<j+k$.

First *Diophantine Equation*:

$$E_j\left(z^{-1},\rho_t\right)A_f\left(z^{-1},\rho_t\right)+z^{-j-k}H_j\left(z^{-1},\rho_t\right)=D_{fp}\left(z^{-1},\rho_t\right) \tag{3.17}$$

Moreover, (3.17) maybe written as:

$$E_j\left(z^{-1},\rho_t\right)+z^{-j-k}H_j\left(z^{-1},\rho_t\right)A_f^{-1}\left(z^{-1},\rho_t\right)=D_{fp}\left(z^{-1},\rho_t\right)A_f^{-1}\left(z^{-1},\rho_t\right) \tag{3.18}$$

Prediction Equation: The weighted observations signal (3.14) is expanded using (3.18) as in (3.19).

$$\begin{aligned} z_p(t) &= d_p(t) + P_c\left(z^{-1},\rho_t\right)W_{0k}\left(z^{-1},\rho_t\right)u_0(t-k) + \\ &\left[E_j\left(z^{-1},\rho_t\right) + z^{-j-k}H_j\left(z^{-1},\rho_t\right)A_f^{-1}\left(z^{-1},\rho_t\right)\right]\varepsilon(t) \end{aligned} \tag{3.19}$$

Substituting $\varepsilon(t) = Y_f^{-1}z(t) - D_f^{-1}B_{0k}u_0(t-k)$ from (3.10):

$$\begin{aligned} &z_p(t) = d_p(t) + P_c\left(z^{-1},\rho_t\right)W_{0k}\left(z^{-1},\rho_t\right)u_0(t-k) + E_j\left(z^{-1},\rho_t\right)\varepsilon(t) + \\ &z^{-j-k}H_j\left(z^{-1},\rho_t\right)A_f^{-1}\left(z^{-1},\rho_t\right)\left(Y_f^{-1}\left(z^{-1},\rho_t\right)z(t) - D_f^{-1}\left(z^{-1},\rho_t\right)B_{0k}\left(z^{-1},\rho_t\right)u_0(t-k)\right) \end{aligned}$$

Recall $A_f^{-1}Y_f^{-1} = D_{fp}^{-1}P_c$ from (3.12), the *weighted observations* signal becomes:

$$\begin{aligned} &z_p(t) = d_p(t) + E_j\left(z^{-1},\rho_t\right)\varepsilon(t) + \\ &z^{-j-k}H_j\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right)\left(z_p(t) - d_p(t)\right) + \\ &\left(P_c\left(z^{-1},\rho_t\right)Y_f\left(z^{-1},\rho_t\right)A_f\left(z^{-1},\rho_t\right) - z^{-j-k}H_j\left(z^{-1},\rho_t\right)\right)\left(A_f^{-1}\left(z^{-1},\rho_t\right)\right. \\ &\left.D_f^{-1}\left(z^{-1},\rho_t\right)B_{0k}\left(z^{-1},\rho_t\right)u_0(t-k)\right) \end{aligned} \tag{3.20}$$

Weighted Output Signal: The weighted output signal $y_p(t) = z_p(t) - v_p(t)$, and hence from (3.20), noting (3.12) $P_c\,Y_f A_f = D_{fp}$ and (3.17) then (3.21) gives the weighted output as:

$$\begin{aligned} &y_p(t) = d_p(t) + E_j\left(z^{-1},\rho_t\right)\varepsilon(t) - \\ &v_p(t) + z^{-j-k}H_j\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right)\left(z_p(t) - d_p(t)\right) + \\ &E_j\left(z^{-1},\rho_t\right)D_f^{-1}\left(z^{-1},\rho_t\right)B_{0k}\left(z^{-1},\rho_t\right)u_0(t-k) \end{aligned} \tag{3.21}$$

Weighted Output Future Values: The weighted output signal $j+k$ *steps-ahead* is given in (3.22).

$$\begin{aligned} &y_p(t+j+k) = \\ &E_j\left(z^{-1},\rho_{t+j+k}\right)\varepsilon(t+j+k) - v_p(t+j+k) + \\ &d_p(t+j+k) + H_j\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right)\left(z_p(t) - d_p(t)\right) + \\ &E_j\left(z^{-1},\rho_{t+j+k}\right)D_f^{-1}\left(z^{-1},\rho_{t+j+k}\right)B_{0k}\left(z^{-1},\rho_{t+j+k}\right)u_0(t+j) \end{aligned} \tag{3.22}$$

To further simplify (3.22), define (recalling D_f^{-1} is expected stable) the right coprime model:

$$B_{1k}\left(z^{-1},\rho_t\right)D_{f1}^{-1}\left(z^{-1},\rho_t\right)=D_f^{-1}\left(z^{-1},\rho_t\right)B_{0k}\left(z^{-1},\rho_t\right) \tag{3.23}$$

Letting,

$$u_f(t)=D_{f1}^{-1}(z^{-1},\rho_t)u_0(t)$$

Then (3.22) maybe written as:

$$\begin{aligned}y_p(t+j+k)=&\left(E_j\left(z^{-1},\rho_{t+j+k}\right)\varepsilon(t+j+k)-v_p(t+j+k)\right)+\\&\Big[d_p(t+j+k)+H_j\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right)\left(z_p(t)-d_p(t)\right)+\\&E_j\left(z^{-1},\rho_{t+j+k}\right)B_{1k}\left(z^{-1},\rho_{t+j+k}\right)u_f(t+j)\Big]\end{aligned} \tag{3.24}$$

where $j+k-1$ is the polynomial matrix E_j maximum degree and hence, the noise parts in $E_j\varepsilon(t+j+k)$ comprising future times $\varepsilon(t+j+k),...,\varepsilon(t+1)$. Thus, $E_j\varepsilon(t+j+k)$ denotes future *white-noise* signal parts weighted sum.

3.3 Prediction Equations

At the time $t+j+k$, provided measurements till instant *t*, the predictor can first be developed for situations when the measurement noise $v(t)$ is assumed zero. Until instant *t*, the measurements are given and the utilized predictor control inputs future values $u_0(t),..,u_0(t+j)$ are also expected known at instant *t*. Both the future control and future known disturbance are independent of the future stochastic disturbance and noise series. It concludes the square $[\cdot]$ and round $(\cdot)$ bracketed terms value in (3.24) are expected zero.

Provided that the *cost-function cross-terms* are null, then the optimal predictor to minimize the *cost-function* (3.15) can develop from (3.24) as:

$$\begin{aligned}\hat{y}_p(t+j+k\,|\,t) = d_p(t+j+k) + H_j\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right)\left(z_p(t)-d_p(t)\right)+ \\ E_j\left(z^{-1},\rho_{t+j+k}\right)B_{1k}\left(z^{-1},\rho_{t+j+k}\right)u_f(t+j)\end{aligned} \tag{3.25}$$

If the $v_p(t+j+k)=P_c(z^{-1},\rho_{t+j+k})v(t+j+k)$ measurement noise is *non-zero*. In this case, assume the weighting $P_c(z^{-1},\rho_t)$ is a constant (typical in *GPC*) or $j+k-1$ degree polynomial matrix. Consequently, $v_p(t+j+k)$ only depends on future measurement *white-noise* signal terms, and its expected value and the square bracketed in (3.24) need again to be zero.

The optimal predictor is thus offered by (3.25) and the prediction error:

$$\tilde{y}_p(t+j+k|t)=E_j\left(z^{-1},\rho_{t+j+k}\right)\varepsilon(t+j+k)-v_p(t+j+k) \tag{3.26}$$

Second *Diophantine Equation*: The next *Diophantine equation* can be formed to split up the term $E_j(z^{-1},\rho_t)B_{1k}(z^{-1},\rho_t)$ to a component with step delay $j+1$ and a component that depends on $D_{f1}(z^{-1},\rho_t)$ (mind $u_f(t)=D_{f1}^{-1}(z^{-1},\rho_t)u_0(t)$). So, create a *Diophantine equation* that has the solution (G_j,S_j) of the smallest degree G_j for $j\geq 0$:

$$\begin{aligned}&G_j\left(z^{-1},\rho_t\right)D_{f1}\left(z^{-1},\rho_t\right)+z^{-j-1}S_j\left(z^{-1},\rho_t\right)\\ &=E_j\left(z^{-1},\rho_t\right)B_{1k}\left(z^{-1},\rho_t\right)\end{aligned} \tag{3.27}$$

where $\deg\left(G_j(z^{-1},\rho_t)\right)=j$.

Weighted Predicted Output: The prediction equation (3.25) is acquired (for $j\geq 0$) as given in (3.28).

$$\begin{aligned}\hat{y}_p(t+j+k|t) = d_p(t+j+k) + H_j\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right)\left(z_p(t)-d_p(t)\right)+\\ G_j\left(z^{-1},\rho_{t+j+k}\right)u_0(t+j)+S_j\left(z^{-1},\rho_{t+k-1}\right)u_f(t-1)\end{aligned} \tag{3.28}$$

The degree of $G_j(z^{-1},\rho_t)$ is *j*, and the third term (3.28) involves future inputs $\{u_0(t+j),...,u_0(t)\}$. The signal $f_j(t+k)$ in (3.29) can be described in terms of known inputs and outputs:

$$\begin{aligned}f_j(t+k) = d_p(t+j+k) + S_j\left(z^{-1},\rho_{t+k-1}\right)u_f(t-1)+\\ H_j\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right)\left(z_p(t)-d_p(t)\right)\end{aligned} \tag{3.29}$$

The predicted weighted output (3.28) can, therefore, be given, for $j \geq 0$:

$$\hat{y}_p(t+j+k \mid t) = f_j(t+k) + G_j\left(z^{-1},\rho_{t+j+k}\right)u_0(t+j) \tag{3.30}$$

The signal $f_j(t+k)$ represents the weighted output $y_p(t+j+k)$ *free-response* prediction, assuming that the inputs $u_0(t+i)$ for $j \geq 0$ are zero. That is, the signal $f_j(t+k)$ represents the predictor's response, assuming that all future inputs are null.

Coefficients of the Polynomial Matrix: From (3.12), (3.17), (3.23) and (3.27), the polynomial matrix components:

$$\begin{aligned}G_j\left(z^{-1},\rho_t\right) = P_c\left(z^{-1},\rho_t\right)A^{-1}\left(z^{-1},\rho_t\right)B_{0k}\left(z^{-1},\rho_t\right)-\\ z^{-j}\left(z^{-k}H_j\left(z^{-1},\rho_t\right)A_f^{-1}\left(z^{-1},\rho_t\right)B_{1k}\left(z^{-1},\rho_t\right)+\right.\\ \left.z^{-1}S_j\left(z^{-1},\rho_t\right)\right)D_{f1}^{-1}\left(z^{-1},\rho_t\right)\end{aligned} \tag{3.31}$$

From the above, the polynomial matrix $G_j(z^{-1},\rho_t)$ of the degree j, therefore, includes the first $j+1$ parameters g_j of the weighted linear plant transfer:

$$P_c(z^{-1},\rho_t)W_{0k}(z^{-1},\rho_t)$$

Furthermore, it may be arranged as:

$$G_j\left(z^{-1},\rho_t\right)=g_{0,t}+g_{1,t}z^{-1}+\cdots+g_{j,t}z^{-j}$$

where $G_0(z^{-1},\rho_t)=g_{0,t}$.

3.4 Vector and Matrix Notation

The future weighted outputs in (3.30) can be obtained as in the utilized form in (3.32). These next weighted future outputs are required later for inputs in the period $\tau\in[t,t+N]$, where $N\geq 0$.

$$\begin{bmatrix}\hat{y}_p(t+k|t)\\ \hat{y}_p(t+1+k|t)\\ \vdots\\ \vdots\\ \hat{y}_p(t+N+k|t)\end{bmatrix}=$$
$$\begin{bmatrix}f_0(t+k)\\ f_1(t+k)\\ \vdots\\ \vdots\\ f_N(t+k)\end{bmatrix}+\begin{bmatrix}g_{0,t+k} & 0 & \cdots & 0\\ g_{1,t+1+k} & g_{0,t+1+k} & 0 & 0\\ \vdots & g_{1,t+2+k} & g_{0,t+2+k} & \vdots\\ \vdots & \vdots & \ddots & \vdots\\ g_{N,t+N+k} & g_{N-1,\,t+N+k} & \cdots & g_{0,\,t+N+k}\end{bmatrix}\begin{bmatrix}u_0(t)\\ u_0(t+1)\\ \vdots\\ \vdots\\ u_0(t+N)\end{bmatrix} \tag{3.32}$$

Introducing terms explanation for matrices in (3.32), the vector shape of the predicted future weighted outputs is:

$$\hat{Y}_{P_{t+k,N}}=F_{P_{t+k,N}}+G_{P_{t+k,N}}U^0_{t,N} \tag{3.33}$$

The vector of free response predictions using (3.29) may now be identified as in (3.34).

$$F_{P_{t+k,N}} = \overbrace{\begin{bmatrix} d_p(t+k) \\ d_p(t+1+k) \\ \vdots \\ d_p(t+N+k) \end{bmatrix}}^{P_{NP}\left(z^{-1},\rho_t\right)} + \overbrace{\begin{bmatrix} S_0\left(z^{-1},\rho_{t+k\text{-}1}\right) \\ S_1\left(z^{-1},\rho_{t+k\text{-}1}\right) \\ \vdots \\ S_N\left(z^{-1},\rho_{t+k\text{-}1}\right) \end{bmatrix}}^{S_{NZ}\left(z^{-1},\rho_t\right)} u_f(t-1) + \overbrace{\begin{bmatrix} H_0\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right) \\ H_1(z^{-1},p_t)D_{fp}^{-1}\left(z^{-1},\rho_t\right) \\ \vdots \\ H_N\left(z^{-1},\rho_t\right)D_{fp}^{-1}\left(z^{-1},\rho_t\right) \end{bmatrix}}^{H_{NZ}\left(z^{-1},\rho_t\right)} z_p(t) \tag{3.34}$$

The prediction error vector form:

$$\tilde{y}_p(t+j+k|t) = E_j\left(z^{-1},\rho_{t+j+k}\right)\varepsilon(t+j+k) - v_p(t+j+k)$$

maybe given, remembering $\deg\left(E_j(z^{-1},\rho_t)\right) < j+k$, as:

$$\tilde{Y}_{P_{t+k,N}} = \begin{bmatrix} e_0\varepsilon(t+k)+\ldots+e_{k-1}\varepsilon(t+1)-v_p(t+k) \\ e_0\varepsilon(t+1+k)+\ldots+e_k\varepsilon(t+1)-v_p(t+1+k) \\ \vdots \\ e_0\varepsilon(t+N+k)+\ldots+e_{N+k-1}\varepsilon(t+1)-v_p(t+N+k) \end{bmatrix} \tag{3.35}$$

Knowledge of Future Set Point: Future deviations of the reference signal $r(t)$ are known across the defined N steps future horizon in many applications. As the weighted error, the weighted reference $r_p(t) = P_c(z^{-1},\rho_t)r(t)$ is believed to include stable weighting. The future weighted reference, weighted output, and control vectors can be given as:

$$R_{P_{t,N}} = \begin{bmatrix} r_p(t) \\ r_p(t+1) \\ \vdots \\ r_p(t+N) \end{bmatrix}, Y_{P_{t,N}} = \begin{bmatrix} y_p(t) \\ y_p(t+1) \\ \vdots \\ y_p(t+N) \end{bmatrix}, U_{t,N}^0 = \begin{bmatrix} u_0(t) \\ u_0(t+1) \\ \vdots \\ u_0(t+N) \end{bmatrix} \tag{3.36}$$

The future weighted outputs *k-steps-ahead* are given in the next vector form:

$$Y_{P_{t+k,N}} = \hat{Y}_{P_{t+k,N}} + \tilde{Y}_{P_{t+k,N}}$$

And the future tracking error is given as:

$$E_{P_{t+k,N}} = R_{P_{t+k,N}} - Y_{P_{t+k,N}} = R_{P_{t+k,N}} - \left(\hat{Y}_{P_{t+k,N}} + \tilde{Y}_{P_{t+k,N}}\right) \tag{3.37}$$

Noting that the predicted signals vector $\hat{Y}_{P_{t+k,N}}$ in (3.37) and the prediction error $\tilde{Y}_{P_{t,N}}$ are orthogonal.

Future Predicted Errors: From (3.33),

$$\hat{E}_{P_{t+k,N}} = R_{P_{t+k,N}} - \hat{Y}_{P_{t+k,N}} = (R_{P_{t+k,N}} - F_{P_{t+k,N}}) - G_{P_{t+k,N}} U^{0}_{t,N} \tag{3.38}$$

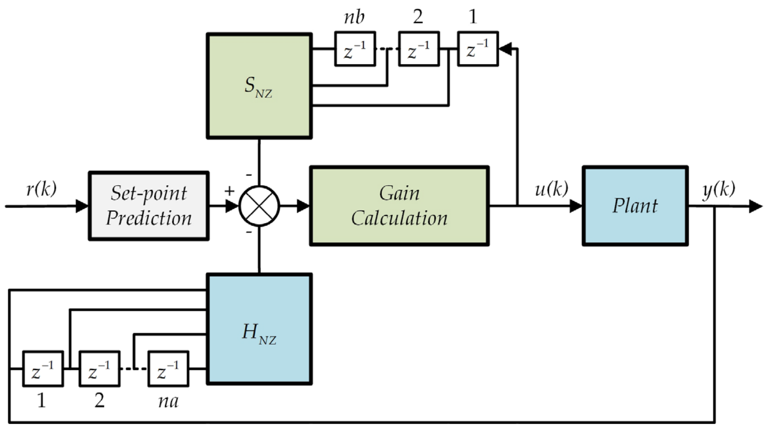

Figure 3-4: Polynomial RS control principle

Finally, the previous system description is summarised, as shown in Figure 3-4, and the aim in Chapter 6 is to complete the *RS* gain calculation.

3.5 Chapter Summary

This Chapter summarises the polynomial system description that will be used in Chapter 6. The polynomial *LPV* representation of a *discrete-time LPV* system is essential for developing *RS* controller solution. First, in Section 3.1, an introduction was provided to the previous related work. Then, other sections introduced the *LPV* polynomial matrix models, the prediction equations and concluded with the vector and matrix notation.

Chapter 4 Linear RS Control

4.1 Introduction

In this Chapter, we introduce the linear *RS* controller solution for the *state-space* system description given in Section 2.1 from Chapter 2. The *RS* controller and future controls vector parameterization is defined in Section 4.2. The linear *RS* optimization is introduced in Section 4.3 with an equivalent cost optimization problem that leads to modified *RS* controller *cost-function* and a note on the *cost-function* tuning variables. The *RS* solution for the system with an unstructured subsystem is given in Section 4.4. This solution involves introducing the *RS* problem, the *RS* control signal, optimal control signal representation, the current state estimate utilization, optimal predictive control, and a discussion on *RS* control implementation.

4.2 Linear RS Control

A direct derivation of the *RS* controller is given next, where for this section, the first unstructured subsystem block is replaced by setting $W_{1k} = I$. Noting, the *GPC* performance index that stimulates the minimization of *RS* criterion includes a dynamic error weighting is offered by (4.1).

$$J = E\left\{\sum_{j=0}^{N} e_p(t+j+k)^T e_p(t+j+k) + \lambda_j^2 \left(u_0(t+j)^T u_0(t+j)\right) | t\right\} \tag{4.1}$$

where $E\{\cdot | t\}$ entails conditional expectation on the measurements till instant t and λ_j is the weighting factor on the control action. The future optimal control action is to be determined during $\tau \in [t, t+N]$. The *state-space* models produce the signals r_p and y_p which include any dynamic weighting $P_c(z^{-1})$ like a *low-pass* filter to prevent *low-frequency* disturbances. Using the above definition, the *GPC* criterion in *vector-matrix* form is:

$$J = E\left\{J_t\right\} = E\left\{E_{P_{t+k,N}}^T E_{P_{t+k,N}} + U_{t,N}^{0^T} \Lambda_N^2 U_{t,N}^0 | t\right\}$$

The *RS cost-function* needed here can be specified to have an equal form but with a minor improvement. Therefore, controller gains limiting term is inserted in the *cost-index* to penalize the high gain, and the vector form of the *RS multi-step cost-function* is described as:

$$J = E\left\{J_t\right\} = E\left\{E_{P_{t+k,N}}^T E_{P_{t+k,N}} + U_{t,N}^{0^T} \Lambda_N^2 U_{t,N}^0 + k_c^T \Lambda_K^2 k_c \; | \; t\right\} \tag{4.2}$$

The future inputs u_0 *cost-weightings* on are $\Lambda_N^2 = diag\{\lambda_0^2, \lambda_1^2, ..., \lambda_N^2\}$ and on the controller gains $\Lambda_K^2 = diag\{\rho_0^2, \rho_1^2, ..., \rho_N^2\}$. An optimal state estimator is needed if the states are not available. The *cost-function* is given in optimal state estimate and state estimation error terms if a *Kalman filter* is used for *state-estimation* and prediction, hence as of (4.2):

$$J = E\left\{(\hat{E}_{P_{t+k,N}} + \tilde{E}_{P_{t+k,N}})^T (\hat{E}_{P_{t+k,N}} + \tilde{E}_{P_{t+k,N}}) + U_{t,N}^{0^T} \Lambda_N^2 U_{t,N}^0 + k_c^T \Lambda_K^2 k_c \; | \; t\right\} \tag{4.3}$$

The *cost-index* terms are reduced, observing that the optimal estimate $\hat{E}_{P_{t+k,N}}$ is orthogonal to estimation error $\tilde{E}_{P_{t+k,N}}$. The *cost-function* vector/matrix form is:

$$J = \hat{E}^T_{P_{t+k,N}} \hat{E}_{P_{t+k,N}} + U^{0^T}_{t,N} \Lambda^2_N U^0_{t,N} + k^T_c \Lambda^2_K k_c + J_0 \tag{4.4}$$

where both of the terms $\tilde{E}_{P_{t+k,N}} = C_{PN} A_N x(t+k|t) + C_{PN} D_N W_{t+k,N}$ along with the cost term in $J_0(t) = E\{\tilde{E}^T_{P_{t+k,N}} \tilde{E}_{P_{t+k,N}} |t\}$ doesn't depend on the control action. Observing (2.20) the *state-estimates* vector is given as:

$$\begin{aligned} \hat{E}_{P_{t+k,N}} &= D_{P_{t+k,N}} + C_{PN} A_N \hat{x}(t+k|t) + V_{PN} U^0_{t,N} \\ &= \tilde{D}_{P_{t+k,N}} + V_{PN} U^0_{t,N} \end{aligned} \tag{4.5}$$

where

$$\tilde{D}_{P_{t+k,N}} = D_{P_{t+k,N}} + C_{PN} A_N \hat{x}(t+k|t) \tag{4.6}$$

Now the state estimate may be written as $\hat{x}(t+k \mid t) = A^k \hat{x}(t \mid t) + T_0(k, z^{-1}) B u_0(t)$ and (2.25) reveals that $\hat{x}(t+k \mid t)$ depends only on the control action past values. The *multi-step cost-function* (4.4) is extended as:

$$\begin{aligned} J &= (\tilde{D}_{P_{t+k,N}} + V_{PN} U^0_{t,N})^T (\tilde{D}_{P_{t+k,N}} + V_{PN} U^0_{t,N}) + U^{0^T}_{t,N} \Lambda^2_N U^0_{t,N} + k^T_c \Lambda^2_K k_c + J_0 \\ &= \tilde{D}^T_{P_{t+k,N}} \tilde{D}_{P_{t+k,N}} + U^{0^T}_{t,N} V^T_{PN} \tilde{D}_{P_{t+k,N}} + \tilde{D}^T_{P_{t+k,N}} V_{PN} U^0_{t,N} \\ &+ U^{0^T}_{t,N} \left(V^T_{PN} V_{PN} + \Lambda^2_N \right) U^0_{t,N} + k^T_c \Lambda^2_K k_c + J_0 \end{aligned} \tag{4.7}$$

Remarks: Before executing the optimal control optimization computing, the controller structure requires to be defined to have a desired, probably *low-order* structure. This approach is unlike an unconstrained *MBPC* solution in which the future controls vector is computed, and the predicted control at only the instant *t* is employed.

4.2.1 Parameterizing the RS Controller

The *RS* controller structure will be characterized in the next, where only gains are to be calculated. The main functions or weightings on which the controller is defined are chosen beforehand. This definition can be the proportional, integrator, and derivative functions in the *PID* control in the simplest case. A number of N_e *frequency-sensitive* linear dynamic functions are picked for their unique frequency response characteristics, and this is rather like choosing the basis for space to form the control signal as:

$$u(t)=L_u(z^{-1})\sum_{j=1}^{N_e} f_j(z^{-1})\,k_j(t)\,L_e(z^{-1})\,e_0(t) \tag{4.8}$$

where the weighted error $e_L(t)=L_e(z^{-1})e_0(t)$ denotes error *frequency-weighting* and $L_u(z^{-1})$ denotes the system inputs *frequency-weighting*. The functions $f_j(z^{-1})$ and gains $k_j(t)$ in *diagonal-matrix* forms are:

$$f_j(z^{-1})=diag\left\{f_{11}^j(z^{-1}),\quad f_{22}^j(z^{-1}),\quad \cdots\quad ,f_{pp}^j(z^{-1})\right\} \tag{4.9}$$

$$k_j=diag\left\{k_{11}^j,\quad k_{22}^j,\quad \cdots\quad ,k_{pp}^j\right\} \tag{4.10}$$

The weightings $L_u(z^{-1})$ and $L_e(z^{-1})$ in (4.8) may not be essential and maybe just assigned to the identity. As in the *McFarlane* and *Glover loop-shaping* design procedure [64], they may be chosen for *MIMO* systems where specific sensitivity minimization *loop-shaping* is mandatory.

The suggested *RS* controller control action is produced by adding the outcome of unique vector functions. For example, this could be the sum of proportional, integral with filtered derivative functions in every channel as in (4.11).

$$u(t)=L_u(z^{-1})\Big(f_1(z^{-1})k_1e_L(t)+f_2(z^{-1})k_2e_L(t)+\cdots+f_{N_e}(z^{-1})k_{N_e}e_L(t)\Big) \qquad (4.11)$$

The functions $f_j(z^{-1})$ and the gains k_j denote diagonal weighting functions and controller gains, respectively. The functions $f_j(z^{-1})$ represent *frequency-sensitive* weighting functions that are designer *pre-specified* functions, and the N_e gain terms k_j are the multivariable controller gain vectors set.

Example: *PI* and Filtered Derivative Controller

The first term in each channel $f_1(z^{-1})k_1e_L(t)$ is selected as a proportional term, and the second term $f_2(z^{-1})k_2e_L(t)$ embodies integral term and so on. Thus, the functions are chosen for a scalar system as $f_1(z^{-1})=1, f_2(z^{-1})=1/(1-z^{-1})$ and $f_3(z^{-1})=(1-z^{-1})/(1-\alpha z^{-1})$, and the *RS* optimal control is:

$$u(t)=k_1\,e(t)+\frac{1}{(1-z^{-1})}k_2\,e(t)+\frac{(1-z^{-1})}{(1-\alpha z^{-1})}k_3e(t)$$

For the multivariable system, each term in the calculation of the control signal in (4.8) has the formation $f_j(z^{-1})k_je_L(t)$. As of (4.9) and (4.10) the j^{th} function term that may be present in each channel can be given in the *diagonal-matrix* form:

$$f_j(z^{-1})k_j=diag\left\{f_{11}^j(z^{-1})k_{11}^j,\quad f_{22}^j(z^{-1})k_{22}^j,\quad \cdots\quad ,f_{pp}^j(z^{-1})k_{pp}^j\right\} \qquad (4.12)$$

For more significant generality p channels are thought, but in practice, the system may be square, so that $p=r=m$. Also, note that the weighted errors vector $e_L(t)=L_e(z^{-1})e_0(t)$ can be given for each channel in terms of scalar signals $e_L^T(t)=[e_1^L(t)\quad e_2^L(t)\quad \cdots\quad e_p^L(t)]$. Therefore, for all channels, the j^{th} functional term multiplied by the error is:

$$f_j(z^{-1})k_j e_L(t) = \begin{bmatrix} f_{11}^j(z^{-1})k_{11}^j e_1^L(t) \\ f_{22}^j(z^{-1})k_{22}^j e_2^L(t) \\ \vdots \\ f_{pp}^j(z^{-1})k_{pp}^j e_p^L(t) \end{bmatrix}$$
$$= diag\left\{ f_{11}^j(z^{-1})e_1^L(t),\ \ f_{22}^j(z^{-1})e_2^L(t),\ \ \cdots\ \ , f_{pp}^j(z^{-1})e_p^L(t) \right\} \begin{bmatrix} k_{11}^j \\ k_{22}^j \\ \vdots \\ k_{pp}^j \end{bmatrix} \tag{4.13}$$

Note that the *gain-vector* in (4.13) includes the scalar gains for the j^{th} function element in each channel. They may represent the integral gain terms in each channel, for example. Hence based on the parameterized controller, the control signal is:

$$u(t) = L_u(z^{-1}) \sum_{j=1}^{N_e} \left\{ f_j(z^{-1})k_j e_L(t) \right\}$$
$$= L_u(z^{-1}) \sum_{j=1}^{N_e} \left\{ diag\left\{ f_{11}^j(z^{-1})e_1^L(t),\ \ f_{22}^j(z^{-1})e_2^L(t),\ \ \cdots\ \ , f_{pp}^j(z^{-1})e_p^L(t) \right\} \begin{bmatrix} k_{11}^j \\ k_{22}^j \\ \vdots \\ k_{pp}^j \end{bmatrix} \right\}$$

$$u(t) = L_u(z^{-1}) \begin{bmatrix} \sum_{j=1}^{N_e} f_{11}^j(z^{-1})e_1^L(t)k_{11}^j \\ \sum_{j=1}^{N_e} f_{22}^j(z^{-1})e_2^L(t)k_{22}^j \\ \vdots \\ \sum_{j=1}^{N_e} f_{pp}^j(z^{-1})e_p^L(t)k_{pp}^j \end{bmatrix} \tag{4.14}$$

The idea of extending the capabilities of *PID* control by adding additional terms is not new. However, the range of possibilities is extensive, and the only way to easily assess potential benefits is to either try on a real system or utilize a simulation. There is then the problem of choosing gains associated with each function, where the predictive control capability is useful. It doesn't provide a rapid means of computing the gains only, but it also provides a *cost-function* or benchmarking index by which different solutions can be compared.

4.2.2 Parameterized Controller

The approach in (4.14) delivers a possible controller parameterization, although it is desirable to have an alternative structure for the gains optimization. Thus, a matrix expression is essential to allow the gains collection and be stored in an easy to process vector form. Therefore, for a row $s \in [1, p]$, the *function-vector* signal identified:

$$e_{fs}(t) = \left[f_{ss}^{1}(z^{-1}) e_{s}^{L}(t) \quad f_{ss}^{2}(z^{-1}) e_{s}^{L}(t) \quad \cdots \quad f_{ss}^{N_e}(z^{-1}) e_{s}^{L}(t) \right] \tag{4.15}$$

Furthermore, the *gain-vector* for the s^{th} channel can be given as:

$$k_{cs} = \begin{bmatrix} k_{ss}^{1} \\ k_{ss}^{2} \\ \vdots \\ k_{ss}^{N_e} \end{bmatrix}, \quad s \in [1, p] \tag{4.16}$$

The s^{th} row term $\sum_{j=1}^{N_e} f_{ss}^{j}(z^{-1}) e_{s}^{L}(t) k_{ss}^{j}$ in (4.14) may, therefore, be written as:

$$\begin{aligned} \sum_{j=1}^{N_e} f_{ss}^{j}(z^{-1}) e_{s}^{L}(t) k_{ss}^{j} &= e_{fs}(t) k_{cs} \\ &= \left[f_{ss}^{1}(z^{-1}) e_{s}^{L}(t) \quad f_{ss}^{2}(z^{-1}) e_{s}^{L}(t) \quad \cdots \quad f_{ss}^{N_e}(z^{-1}) e_{s}^{L}(t) \right] \begin{bmatrix} k_{ss}^{1} \\ k_{ss}^{2} \\ \vdots \\ k_{ss}^{N_e} \end{bmatrix} \end{aligned} \tag{4.17}$$

The above links to the scalar gains vector multiplied by the related function in channel s. Hence, larger augmented matrices are utilized to accumulate these signals and gains.

Total Error Vector: The signals given in (4.15) are accumulated in the following *diagonal-matrix* as:

$$e_f(t)=diag\left\{e_{f1}(t) \quad e_{f2}(t) \quad \cdots \quad e_{fp}(t)\right\} \tag{4.18}$$

Alternatively, after substituting from (4.15):

$$e_f(t) =$$
$$\begin{bmatrix} \left[f_{11}^{1}(z^{-1})e_1^{L}(t) \quad f_{11}^{2}(z^{-1})e_1^{L}(t) \quad \cdots \quad f_{11}^{N_e}(z^{-1})e_1^{L}(t) \right] & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \left[f_{pp}^{1}(z^{-1})e_p^{L}(t) \quad f_{pp}^{2}(z^{-1})e_p^{L}(t) \quad \cdots \quad f_{pp}^{N_e}(z^{-1})e_p^{L}(t) \right] \end{bmatrix}$$

Example: *PI* Controller for 2×2 Systems

For 2×2 square systems, if, for example, *PI* control is used in each channel, then the matrix is given as:

$$e_f(t)=\begin{bmatrix} \left[f_{11}^{1}(z^{-1})e_1(t) \quad f_{11}^{2}(z^{-1})e_1(t) \right] & 0 \\ 0 & \left[f_{22}^{1}(z^{-1})e_2(t) \quad f_{22}^{2}(z^{-1})e_2(t) \right] \end{bmatrix}$$

The various channel gains may be accumulated together in a vector when considering the general case.

Total Gain Vector: Also, let the gain vector that contains the total gains formed from the channel gains in (4.16) be defined as:

$$k_c=\begin{bmatrix} k_{c1} \\ k_{c2} \\ \vdots \\ k_{cp} \end{bmatrix} \tag{4.19}$$

This vector includes all the scalar gain terms in the function gain vectors in (4.16) for each $s \in [1, p]$. It is the full vector of gains needed to implement the controller ordered in terms of the channel numbers. It follows from (4.18) and (4.19) that the matrix in (4.14) above may be given as:

$$e_f(t)k_c = \begin{bmatrix} e_{f1}(t)k_{c1} \\ e_{f2}(t)k_{c2} \\ \vdots \\ e_{fp}(t)k_{cp} \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{N_e} f_{11}^j(z^{-1})e_1^L(t)k_{11}^j \\ \sum_{j=1}^{N_e} f_{22}^j(z^{-1})e_2^L(t)k_{22}^j \\ \vdots \\ \sum_{j=1}^{N_e} f_{pp}^j(z^{-1})e_p^L(t)k_{pp}^j \end{bmatrix}$$

Example: *PI* Controller for 2×2 Systems

The *2-square* system with *PI* control is considered for the example above:

$$\begin{aligned} e_f(t)k_c &= \begin{bmatrix} \begin{bmatrix} f_{11}^1(z^{-1})e_1(t) & f_{11}^2(z^{-1})e_1(t) \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} f_{22}^1(z^{-1})e_2(t) & f_{22}^2(z^{-1})e_2(t) \end{bmatrix} \end{bmatrix} \begin{bmatrix} k_{11}^1 \\ k_{11}^2 \\ k_{22}^1 \\ k_{22}^2 \end{bmatrix} \\ &= \begin{bmatrix} \left(f_{11}^1(z^{-1})k_{11}^1 + f_{11}^2(z^{-1})k_{11}^2\right)e_1(t) \\ \left(f_{22}^1(z^{-1})k_{22}^1 + f_{22}^2(z^{-1})k_{22}^2\right)e_2(t) \end{bmatrix} \end{aligned}$$

The control signal to be implemented (4.14) above can be given in terms of these matrices as:

$$u(t) = L_u(z^{-1}) \begin{bmatrix} e_{f1}(t)k_{c1} \\ e_{f2}(t)k_{c2} \\ \vdots \\ e_{fp}(t)k_{cp} \end{bmatrix} = L_u(z^{-1}) \begin{bmatrix} \sum_{j=1}^{N_e} f_{11}^j(z^{-1})e_1^L(t)k_{11}^j \\ \sum_{j=1}^{N_e} f_{22}^j(z^{-1})e_2^L(t)k_{22}^j \\ \vdots \\ \sum_{j=1}^{N_e} f_{pp}^j(z^{-1})e_p^L(t)k_{pp}^j \end{bmatrix}$$

$$u(t) = \left(L_u(z^{-1})e_f(t)\right) k_c \tag{4.20}$$

The Functional Controller Gains: To summarise, in terms of the gain vector and *pre-specified* functional controller, the expression for the control action:

$$\begin{aligned} u(t) &= L_u(z^{-1})\sum_{j=1}^{N_e}\left\{f_j(z^{-1})k_j e_L(t)\right\} \\ &= L_u(z^{-1}) diag\left\{e_{f1}(t) \quad e_{f2}(t) \quad \cdots \quad e_{fp}(t)\right\}\begin{bmatrix} k_{c1} \\ k_{c2} \\ \vdots \\ k_{cp}\end{bmatrix} \\ &= L_u(z^{-1})e_f(t)k_c \end{aligned} \tag{4.21}$$

Total Gain Vector: The *gain-vector* k_c that must be computed comprises, for each channel, the functional controller gains, and given in the $p \times N_e$ rows as:

$$\begin{aligned} k_c &= \begin{bmatrix} k_{c1} \\ k_{c2} \\ \vdots \\ k_{cp}\end{bmatrix} \\ &= [k_{11}^1 \quad k_{11}^2 \quad \cdots \quad k_{11}^{N_e} \quad k_{22}^1 \quad k_{22}^2 \quad \cdots \quad k_{22}^{N_e} \quad \cdots \quad k_{pp}^1 \quad k_{pp}^2 \quad \cdots \quad k_{pp}^{N_e}]^T \\ &= [\underbrace{k_{11}^1 \quad k_{11}^2 \quad \cdots \quad k_{11}^{N_e}}_{channel\,1\,gains} \quad \underbrace{k_{22}^1 \quad k_{22}^2 \quad \cdots \quad k_{22}^{N_e}}_{channel\,2\,gains} \quad \cdots \quad \underbrace{k_{pp}^1 \quad k_{pp}^2 \quad \cdots \quad k_{pp}^{N_e}}_{channel\,p\,gains}]^T \end{aligned} \tag{4.22}$$

Structure: The structure of the optimal gain matrix is functional; therefore, for a scalar system, the gain vector simply includes the gains related to each function chosen to parameterize the controller. For a multivariable system, the controller is not diagonal since pre and *post-compensation* weightings are not diagonal. The controller gain vector is partitioned, where the first block represents the gains for channel one and the second for channel two and so on.

4.2.3 Future Controls Vector Parameterizing

A predictive *control-based* computation provides the controller gains in (4.22) simply. This approach is not the typical *GPC* technique to predictive control, as it will be expected that the predicted controls are computed using the controller structure (4.11) defined above.

Vector of Future Controls: The future controls vector $U_{t,N}$ are offered as:

$$U_{t,N} = \begin{bmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+N) \end{bmatrix} = \begin{bmatrix} L_u(z^{-1})e_f(t) \\ L_u(z^{-1})e_f(t+1) \\ \vdots \\ L_u(z^{-1})e_f(t+N) \end{bmatrix} k_c(t) \tag{4.23}$$

The matrix (4.23) is referred to U_{fe} and defined as:

$$U_{fe}(t)=\begin{bmatrix} L_u(z^{-1})e_f(t) \\ L_u(z^{-1})e_f(t+1) \\ \vdots \\ L_u(z^{-1})e_f(t+N) \end{bmatrix} \tag{4.24}$$

This matrix has $(N+1)\times m$ rows and $p\times N_e$ columns, and that the i^{th} block row in (4.24) is:

$$L_u(z^{-1})e_f(t+i) = L_u(z^{-1})diag\left\{e_{f1}(t+i) \quad e_{f2}(t+i) \quad \cdots \quad e_{fp}(t+i)\right\}$$

Furthermore, this is an easily computed signal given the predicted values of the error and the *i-step-ahead* control:

$$\begin{aligned} u(t+i) &= L_u(z^{-1})e_f(t+i) \\ &= L_u(z^{-1})diag\left\{e_{f1}(t+i) \quad e_{f2}(t+i) \quad \cdots \quad e_{fp}(t+i)\right\}k_c \end{aligned}$$

Parameterised *RS* Controller: The future controls vector can, therefore, be expressed in the parameterised *RS* controller terms as:

$$U_{t,N} = U_{fe}(t)k_c(t) \tag{4.25}$$

where the vector of control gains:

$$k_c = \begin{bmatrix} k_{11}^1 & k_{11}^2 & \cdots & k_{11}^{N_e} & k_{22}^1 & k_{22}^2 & \cdots & k_{22}^{N_e} & \cdots & k_{pp}^1 & k_{pp}^2 & \cdots & k_{pp}^{N_e} \end{bmatrix}^T \tag{4.26}$$

Assumption: Recall a *GPC* optimal control signal at instant *t* is formed using *receding-horizon* theory [65], where the first component in future controls vector $U_{t,N}^0$ is given as the optimal control. The optimal control is processed for a whole horizon, and only the first value at the instant *t* is applied. The same idea employed here for *RS* control is that k_c , and at the instant *t,* can be utilised to calculate the optimal control for instant *t*. In the same *receding-horizon* principle and the following sample instant, the steps are restarted, and the $k_c(t)$ fresh value can be produced and used in (4.21).

4.3 Linear RS Optimisation

Using the above controller parameterisation, the *cost-function* (4.7) develops to:

$$\begin{aligned} J = {} & \tilde{D}_{P_{t+k,N}}^T \tilde{D}_{P_{t+k,N}} + U_{t,N}^{0T} V_{PN}^T \tilde{D}_{P_{t+k,N}} + \\ & \tilde{D}_{P_{t+k,N}}^T V_{PN} U_{t,N}^0 + U_{t,N}^{0T}\left(V_{PN}^T V_{PN} + \Lambda_N^2\right)U_{t,N}^0 + k_c^T \Lambda_K^2 k_c + J_0 \end{aligned}$$

or

$$\begin{aligned} J = {} & \tilde{D}_{P_{t+k,N}}^T \tilde{D}_{P_{t+k,N}} + k_c^T U_{fe}^T V_{PN}^T \tilde{D}_{P_{t+k,N}} + \\ & \tilde{D}_{P_{t+k,N}}^T V_{PN} U_{fe} k_c + k_c^T\left(U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) U_{fe} + \Lambda_K^2\right) k_c + J_0 \end{aligned} \tag{4.27}$$

If the signals are deterministic, then the cost term minimising approach is nearly the same as if conditional *cost-function* is employed. The *cost-function* gradient is set to zero to achieve the future optimal control signals vector using perturbation and gradient computation [38], remarking that the term in J_0 doesn't depend on control action, and the optimal control gain signals vector becomes to:

$$\begin{aligned} k_c(t) &= -\left(U_{fe}^T(V_{PN}^T V_{PN} + \Lambda_N^2)U_{fe} + \Lambda_K^2\right)^{-1} U_{fe}^T V_{PN}^T \tilde{D}_{P_{t+k,N}} \\ &= -X_N^{-1} U_{fe}^T V_{PN}^T \left(D_{P_{t+k,N}} + C_{PN} A_N \hat{x}(t+k|t)\right) \end{aligned} \tag{4.28}$$

and the *time-varying* matrix:

$$X_N = U_{fe}^T(V_{PN}^T V_{PN} + \Lambda_N^2)U_{fe} + \Lambda_K^2 \tag{4.29}$$

Minimum Cost: Substituting in (4.27) for the gain in (4.28) and simplifying the *minimum-cost*:

$$J_{\min} = \tilde{D}_{P_{t+k,N}}^T \left(I - V_{PN} U_{fe} X_N^{-1} U_{fe}^T V_{PN}^T\right) \tilde{D}_{P_{t+k,N}} + J_0 \tag{4.30}$$

where $\tilde{D}_{P_{t+k,N}}$ followed from (4.6)

Sub-optimality: The *minimum-cost* (4.30) may be linked to the *minimum-cost* for conventional *GPC* optimal control to minimise in (4.1) when the gain weighting Λ_K^2 tends to zero so that $X_N \to U_{fe}^T(V_{PN}^T V_{PN} + \Lambda_N^2)U_{fe}$. Assuming the matrix X_N has an inverse in this limiting case, the minimal cost in the *RS* control problem:

$$J_{\min} \to \tilde{D}_{Pt+k,N}^T \left(I - V_{PN} U_{fe} \left(U_{fe}^T \begin{pmatrix} V_{PN}^T V_{PN} \\ +\Lambda_N^2 \end{pmatrix} U_{fe} \right)^{-1} U_{fe}^T V_{PN}^T \right) \tilde{D}_{Pt+k,N} + J_0 \tag{4.31}$$

Furthermore, this may be compared with the *minimal-cost* in the *GPC* problem:

$$J_{\min}^{GPC} = \tilde{D}_{Pt+k,N}^{T}\left(I - V_{PN}\left(V_{PN}^{T}V_{PN} + \Lambda_N^2\right)^{-1}V_{PN}^{T}\right)\tilde{D}_{Pt+k,N} + J_0 \qquad (4.32)$$

The two costs (4.31) and (4.32) approach the same value when U_{fe} is square and full rank as the gain weighting Λ_K^2 tends to zero, but this is an exceptional case.

Example: *PI* Controller for 2×2 Systems

Consider again the case of just two functions in the controller and a *2-square* system. Then from (4.24),

$$U_{fe}(t) = \begin{bmatrix} L_u(z^{-1})e_f(t) \\ L_u(z^{-1})e_f(t+1) \\ \vdots \\ L_u(z^{-1})e_f(t+N) \end{bmatrix}$$

$$e_f(t) = \begin{bmatrix} \left[f_{11}^1(z^{-1})e_1(t) \quad f_{11}^2(z^{-1})e_1(t)\right] & 0 \\ 0 & \left[f_{22}^1(z^{-1})e_2(t) \quad f_{22}^2(z^{-1})e_2(t)\right] \end{bmatrix}$$

If the cost has $N = 1$, the matrix $U_{fe}(t)$ is square, and if it is full rank, then the cost (4.31) and (4.32) becomes identical.

General conclusions cannot be drawn from a specific example, and several assumptions were made. Still, the minimum *RS* Controller cost will increase relative to the *GPC* solution if the number of functions employed is too small and do not have the desired rank conditions and if the number of steps in the *cost-index* is too small.

Theorem I: Linear *RS* Control

For the linear system and its assumptions given in Chapter 2, wherein this case the subsystem $W_{1k} = I$. The optimal *RS* control is expected to minimise the *RS* controller *cost-index* (4.2):

$$J{=}E\left\{E^T_{Pt+k,N}E_{Pt+k,N} + U^{0T}_{t,N}\Lambda^2_N U^0_{t,N} + k^T_c\Lambda^2_K k_c \mid t\right\} \tag{4.33}$$

The linear subsystem model $L_e(z^{-1})W_0(z^{-1})L_u(z^{-1})$ can be *pre-post-compensated*, and the *RS* controller can be executed as:

$$u(t){=}L_u(z^{-1})\sum_{j=1}^{N_e} f_j(z^{-1})k_j L_e(z^{-1})e_0(t) \tag{4.34}$$

where the j^{th} function term and gain in each channel may be represented in the *diagonal-matrix* forms:

$$f_j(z^{-1}) = diag\left\{f^j_{11}(z^{-1}),\quad f^j_{22}(z^{-1}),\quad \cdots\quad, f^j_{pp}(z^{-1})\right\}$$
$$k_j = diag\left\{k^j_{11},\quad k^j_{22},\quad \cdots\quad, k^j_{pp}\right\}$$

$$f_j(z^{-1})k_j = diag\left\{f^j_{11}(z^{-1})k^j_{11}\quad f^j_{22}(z^{-1})k^j_{22}\quad \cdots\quad f^j_{pp}(z^{-1})k^j_{pp}\right\} \tag{4.35}$$

Here the functions are *pre-specified*, and gains are selected to minimise (4.33). To compute the optimal gains, introduce the *block-diagonal-matrix* of signals:

$$e_f(t) = diag\left\{e_{f1}(t)\quad e_{f2}(t)\quad \cdots\quad e_{fp}(t)\right\}$$

where for each row $s \in [1, p]$ the signal:

$$e_{fs}(t) = \left[f^1_{ss}(z^{-1})e^L_s(t)\quad f^2_{ss}(z^{-1})e^L_s(t)\quad \cdots\quad f^{N_e}_{ss}(z^{-1})e^L_s(t)\right]$$

And the weighted error channel signal $e_L(t){=}L_e(z^{-1})e_0(t)$. The total gain vector to be computed:

$$k_c(t) = \begin{bmatrix} k_{c1}^T & k_{c2}^T & \cdots & k_{cp}^T \end{bmatrix}$$
$$= \begin{bmatrix} k_{11}^1 & k_{11}^2 & \cdots & k_{11}^{N_e} & k_{22}^1 & k_{22}^2 & \cdots & k_{22}^{N_e} & \cdots & k_{pp}^1 & k_{pp}^2 & \cdots & k_{pp}^{N_e} \end{bmatrix}^T$$

Gain Vector: The *RS* optimal control gains vector becomes as in (4.36), using the *receding-horizon* philosophy.

$$k_c(t) = -X_N^{-1} P_{CN} \left(D_{Pt+k,N} + C_{PN} A_N \hat{x}(t+k|t) \right) \tag{4.36}$$

where the *non-singular* matrix $X_N = U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) U_{fe} + \Lambda_K^2$ and the matrix $P_{CN} = U_{fe}^T V_{PN}^T$.

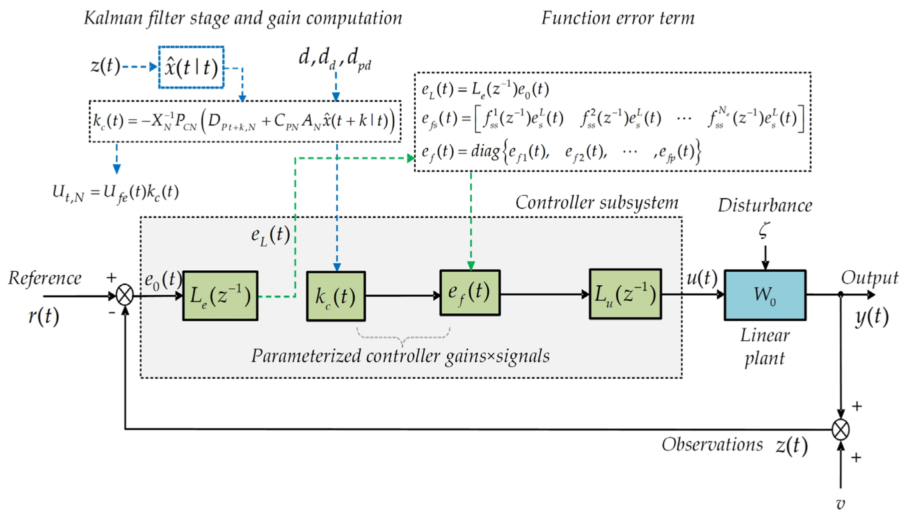
Kalman filter stage and gain computation
Function error term
$z(t)$
$\hat{x}(t|t)$
d, d_d, d_{pd}
$k_c(t) = -X_N^{-1}P_{CN}\left(D_{Pt+k,N} + C_{PN}A_N\hat{x}(t+k|t)\right)$
$e_L(t) = L_e(z^{-1})e_0(t)$
$e_{fs}(t) = \left[f_{ss}^1(z^{-1})e_s^L(t) \quad f_{ss}^2(z^{-1})e_s^L(t) \quad \cdots \quad f_{ss}^{N_e}(z^{-1})e_s^L(t)\right]$
$e_f(t) = diag\{e_{f1}(t), \quad e_{f2}(t), \quad \cdots \quad, e_{fp}(t)\}$
$U_{t,N} = U_{fe}(t)k_c(t)$
Controller subsystem
Disturbance
ζ
$e_L(t)$
Reference
$r(t)$
$e_0(t)$
$L_e(z^{-1})$
$k_c(t)$
$e_f(t)$
$L_u(z^{-1})$
$u(t)$
W_0
Output
$y(t)$
Linear plant
Parameterized controller gains×signals
Observations $z(t)$
v

Figure 4-1: State-space RS controller diagram

Optimal Feedback Control: The optimal feedback control, illustrated in Figure 4-1, may then be found as:

$$u(t) = L_u(z^{-1}) \sum_{j=1}^{N_e} f_j(z^{-1}) k_j L_e(z^{-1}) e_0(t) = \left(L_u(z^{-1}) e_f(t) \right) k_c(t) \tag{4.37}$$

The parameterised future controls vector is found as:

$$U_{t,N} = U_{fe}^T(t) k_c(t) \tag{4.38}$$

where

$$U_{fe}^{T}(t) = \left[\left(L_u(z^{-1})e_f(t)\right)^T \quad \left(L_u(z^{-1})e_f(t+1)\right)^T \quad \cdots \quad \left(L_u(z^{-1})e_f(t+N)\right)^T \right]^T \tag{4.39}$$

Solution: The proof develops by gathering the results of the previous sections.

Control Solution Comments: Notice that the *GPC* control denominator matrix is *full-rank* due to system descriptions and the cost. The representation for the *gain-vector* here appears slightly like the typical *GPC* solution form. However, the denominator matrix in (4.36) includes signal terms and can also be a considerably smaller dimension than typically in *GPC* control. This matrix size depends on the parameterised controller unknown gains number, which is generally lesser than the *GPC* control horizon design in some cases. The signals will stay at constant values when the system achieves a *steady-state,* and it follows the subsequent controller gains will develop to a stable value. Though, suppose the plant is disturbed by significant disturbances or reference variations. In that case, the denominator matrix's signals will be changing, which will result quite easily in *time-varying* controller gains.

An exciting feature of the parameterised controller is that each of the function blocks' output provides a useful measure of the system responses in frequency ranges depending on function choice. The outputs of these functions will probably not change as much as the control action, which is, of course, dependent upon the *time-varying* computed gain vector. It is worth considering the unusual case when the external inputs to the system are absent, and the integral term is therefore zero. If the weighting Λ_K^2 were not present on the controller's size gains, the denominator matrix in the above expression would tend to zero, and the gains would become very large.

However, in this situation, the numerator terms in the gain expression would also tend to zero. Thus, even without the precaution of including a gain costing term, the computed controller gains should not become unbounded. It seems to be a valuable practical asset to have gains magnitudes penalising capability. However, the primary purpose is to ensure this denominator matrix can not be singular (assuming *non-zero* costing on each of the controller gains).

Further Remarks on the Solution: Note from (4.38) the optimisation process involves the vector of future predicted controls. Now the parameterised *RS* controller is formed to have a traditional cascade structure, but the retrieved gains minimise the predictive control *cost-function* in (4.2). The solution (4.36) relies on X_N^{-1} and therefore depends upon the $U_{fe}(t)$ which is a possibly *non-square* matrix. That is:

$$U_{fe}(t)=\begin{bmatrix} L_u(z^{-1})diag\left\{e_{f1}(t) \quad e_{f2}(t) \quad \cdots \quad e_{fp}(t)\right\} \\ L_u(z^{-1})diag\left\{e_{f1}(t+1) \quad e_{f2}(t+1) \quad \cdots \quad e_{fp}(t+1)\right\} \\ \vdots \\ L_u(z^{-1})diag\left\{e_{f1}(t+N) \quad e_{f2}(t+N) \quad \cdots \quad e_{fp}(t+N)\right\} \end{bmatrix} \tag{4.40}$$

where

$$e_{fs}(t)=\left[f_{ss}^{1}(z^{-1})e_s^L(t) \quad f_{ss}^{2}(z^{-1})e_s^L(t) \quad \cdots \quad f_{ss}^{N_e}(z^{-1})e_s^L(t)\right] \tag{4.41}$$

Notice that the matrix U_{fe} in (4.23) is $(N+1)\times m$ rows by $p\times N_e$ columns. Sure, there will be many additional rows in practice than columns for a realistic prediction horizon, as the N_e functions number will be considerably fewer than the prediction horizon. Also, note that V_{PN}^T has an *upper-triangular* block matrix structure as in (4.42).

$$V_{PN}^T = B_N^T C_{PN}^T + E_{PN}^T = \begin{bmatrix} E_p^T & B^T C_p^T & \cdots & B^T A^{TN-2} C_p^T & B^T A^{TN-1} C_p^T \\ 0 & E_p^T & B^T C_p^T & 0 & B^T A^{TN-2} C_p^T \\ \vdots & 0 & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & E_p^T & B^T C_p^T \\ 0 & 0 & \cdots & 0 & E_p^T \end{bmatrix} \tag{4.42}$$

and $V_{PN} = E_p$, in the particular case of $N = 0$.

4.3.1 Optimisation Problem Equivalent Cost

A particular *cost-minimisation* control problem can be connected to the previous problem, and this is required to stimulate the linear control problem created next. Hence, a constant real symmetric matrix which is *positive-definite* (4.29) can be factorised into the form:

$$Y^T Y = X_N = U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) U_{fe} + \Lambda_K^2 \tag{4.43}$$

Then observe noting (4.25), the *cost-function* is given in (4.44) by working the squares in (4.27).

$$\begin{aligned} J &= \tilde{D}_{P_{t+k,N}}^T \tilde{D}_{P_{t+k,N}} + U_{t,N}^{0T} V_{PN}^T \tilde{D}_{P_{t+k,N}} + \tilde{D}_{P_{t+k,N}}^T V_{PN} U_{t,N}^0 \\ &\quad + k_c^T \left(U_{fe}^T \left(V_{PN}^T V_{PN} + \Lambda_N^2 \right) U_{fe} + \Lambda_K^2 \right) k_c + J_0(t) \\ &= \tilde{D}_{P_{t+k,N}}^T \tilde{D}_{P_{t+k,N}} + k_c^T U_{fe}^T V_{PN}^T \tilde{D}_{P_{t+k,N}} + \tilde{D}_{P_{t+k,N}}^T V_{PN} U_{fe} k_c + k_c^T Y^T Y k_c + J_0(t) \\ &= \left(\tilde{D}_{P_{t+k,N}}^T V_{PN} U_{fe} Y^{-1} + k_c^T Y^T \right) \left(Y^{-T} U_{fe}^T V_{PN}^T \tilde{D}_{P_{t+k,N}} + Y k_c \right) \\ &\quad + \left(\tilde{D}_{P_{t+k,N}}^T (I - V_{PN} U_{fe} Y^{-1} Y^{-T} U_{fe}^T V_{PN}^T) \tilde{D}_{P_{t+k,N}} + J_0(t) \right) \end{aligned} \tag{4.44}$$

Consequently, the *cost-function* is given in a comparable structure as:

$$J = \hat{\Phi}_{P_{t+k,N}}^T \hat{\Phi}_{P_{t+k,N}} + J_{10}(t) \tag{4.45}$$

where

$$J_{10}(t)=\left(\tilde{D}_{t+k,N}^{T}(I-V_{PN}U_{fe}Y^{-1}Y^{-T}U_{fe}^{T}V_{PN}^{T})\tilde{D}_{t+k,N}+J_{0}(t)\right) \tag{4.46}$$

The control action independent terms are given as $J_{10}(t)=J_{1}(t)+J_{0}(t)$. where,

$$J_{1}(t)=\tilde{D}_{t+k,N}^{T}(I-V_{PN}U_{fe}Y^{-1}Y^{-T}U_{fe}^{T}V_{PN}^{T})\tilde{D}_{t+k,N} \tag{4.47}$$

From (4.44), it is likewise helpful to identify the signal:

$$\begin{aligned}\hat{\Phi}_{P_{t+k,N}} &= Y^{-T}U_{fe}^{T}V_{PN}^{T}\tilde{D}_{P_{t+k,N}}+Y\,k_{c}\\ &= Y^{-T}U_{fe}^{T}V_{PN}^{T}\left(D_{P_{t+k,N}}+C_{PN}A_{N}\hat{x}(t+k|t)\right)+Y\,k_{c}(t)\end{aligned} \tag{4.48}$$

Remarks:

- Noting the term $J_{10}(t)$ in (4.45) is a control action independent, and setting to zero, the first squared term, returns the optimal control.
- Additionally, the *minimum-cost* $J_{10}(t)$ depends on the term $J_{1}(t)$ that depends on $U_{fe}(t)$ which depends on controller parameterisation.
- The optimal control that minimises the squared term (4.48) is obtained by setting $\hat{\Phi}_{P_{t+k,N}}$ to zero.

Restricted Structure Control:

$$\begin{aligned}k_{c}(t) &= -\left(Y^{T}Y\right)^{-1}U_{fe}^{T}V_{PN}^{T}\left(D_{P_{t+k,N}}+C_{PN}A_{N}\hat{x}(t+k\,|\,t)\right)\\ &= -X_{N}^{-1}U_{fe}^{T}V_{PN}^{T}\left(D_{P_{t+k,N}}+C_{PN}A_{N}\hat{x}(t+k\,|\,t)\right)\end{aligned} \tag{4.49}$$

The solution (4.49) is the same optimal control (4.28). Hence, the optimal *RS* controller for the previous linear system is identical to the controller that minimises the *Euclidean* norm signal $\hat{\Phi}_{P_{t+k,N}}$ given in (4.48).

4.3.2 Modified RS Controller Cost-Function

The previous argument stimulates the description of a slightly different *multi-step MV* cost problem with a similar result for the required optimal controller for the linear problem. Consider a different signal to be minimised, including the addition of weighted error and input as:

$$\phi(t+k)=P_c e(t+k)+F_{c_0} u_0(t)$$

Thence, this signal future value vector for a *multi-step cost-index* is written as:

$$\Phi_{P_{t+k,N}} = P_{CN} E_{P_{t+k,N}} + F_{CN}^0 U_{t,N}^0 + F_{CN}^1 k_c \tag{4.50}$$

Inspired by the *RS* controller weightings above, set the *cost-function* weightings to have the unique *time-varying* matrix forms:

$$\begin{aligned} P_{CN} &= U_{fe}^T V_{PN}^T \\ F_{CN}^0 &= U_{fe}^T \Lambda_N^2, \; F_{CN}^1 = \Lambda_K^2 \end{aligned} \tag{4.51}$$

This option is supported by the conclusions obtained below that are collected in **Theorem II** that follows.

Multi-Step Cost-Function Definition: Define new *MV multi-step cost-function*, utilising the signals vector:

$$\tilde{J}=E\{\tilde{J}_t\} = E\{\Phi_{P_{t+k,N}}^T \Phi_{P_{t+k,N}} |t\} \tag{4.52}$$

Predicting forward *k-steps*:

$$\Phi_{Pt+k,N} = P_{CN} E_{Pt+k,N} + F_{CN}^0 U_{t,N}^0 + F_{CN}^1 k_c \tag{4.53}$$

Count the signal $\Phi_{P_{t+k,N}}$ and replace $E_{P_{t+k,N}} = \hat{E}_{P_{t+k,N}} + \tilde{E}_{P_{t+k,N}}$ for the vector of errors from (4.53) as:

$$\begin{aligned}\Phi_{P_{t+k,N}} &= P_{CN} E_{P_{t+k,N}} + F_{CN}^{0} U_{t,N}^{0} + F_{CN}^{1}\, k_c \\ &= P_{CN} (\hat{E}_{P_{t+k,N}} + \tilde{E}_{P_{t+k,N}}) + F_{CN}^{0} U_{t,N}^{0} + F_{CN}^{1}\, k_c \\ &= P_{CN} \hat{E}_{P_{t+k,N}} + F_{CN}^{0} U_{t,N}^{0} + F_{CN}^{1}\, k_c + P_{CN} \tilde{E}_{P_{t+k,N}}\end{aligned} \tag{4.54}$$

The above representation is written in an estimate and estimation error vectors terms as:

$$\Phi_{P_{t+k,N}} = \hat{\Phi}_{P_{t+k,N}} + \tilde{\Phi}_{P_{t+k,N}} \tag{4.55}$$

The predicted signal:

$$\hat{\Phi}_{P_{t+k,N}} = P_{CN} \hat{E}_{P_{t+k,N}} + F_{CN}^{0} U_{t,N}^{0} + F_{CN}^{1}\, k_c \tag{4.56}$$

And the prediction error:

$$\tilde{\Phi}_{P_{t+k,N}} = P_{CN} \tilde{E}_{P_{t+k,N}} \tag{4.57}$$

Multi-Step Performance Index Simplification: Replace for (4.55) in the given performance index in (4.52) as:

$$\tilde{J} = E\{\tilde{J}_t\} = E\{\Phi_{P_{t+k,N}}^{T} \Phi_{P_{t+k,N}} | t\} = E\{(\hat{\Phi}_{P_{t+k,N}} + \tilde{\Phi}_{P_{t+k,N}})^{T} (\hat{\Phi}_{P_{t+k,N}} + \tilde{\Phi}_{P_{t+k,N}}) | t\}$$

And evoke that both the optimal estimate $\hat{E}_{P_{t+k,N}}$ and estimation error $\tilde{E}_{P_{t+k,N}}$ are orthogonal to simplify the performance index terms in (4.52).

Hence, find:

$$\tilde{J} = E\{\hat{\Phi}_{P_{t+k,N}}^{T} \hat{\Phi}_{P_{t+k,N}} | t\} + E\{\hat{\Phi}_{P_{t+k,N}}^{T} \tilde{\Phi}_{P_{t+k,N}} | t\} + E\{\tilde{\Phi}_{P_{t+k,N}}^{T} \hat{\Phi}_{P_{t+k,N}} | t\} + E\{\tilde{\Phi}_{P_{t+k,N}}^{T} \tilde{\Phi}_{P_{t+k,N}} | t\}$$

$$\tilde{J} = \hat{\Phi}_{P_{t+k,N}}^{T} \hat{\Phi}_{P_{t+k,N}} + E\{\tilde{\Phi}_{P_{t+k,N}}^{T} \tilde{\Phi}_{P_{t+k,N}} \mid t\} \tag{4.58}$$

Therefore, the *cost-function* is given as:

$$\tilde{J}(t) = \hat{\Phi}_{P_{t+k,N}}^{T} \hat{\Phi}_{P_{t+k,N}} + \tilde{J}_1(t) \tag{4.59}$$

The *cost-term* that doesn't depend on control action is given by (4.59) as:

$$\tilde{J}_1(t) = E\{\tilde{\Phi}^T_{P_{t+k,N}} \tilde{\Phi}_{P_{t+k,N}} \mid t\} = E\{\tilde{E}^T_{P_{t+k,N}} P^T_{CN} P_{CN} \tilde{E}_{P_{t+k,N}} \mid t\} \tag{4.60}$$

The vector of predicted signals $\hat{\Phi}_{P_{t+k,N}}$ concise form by replacing for $\hat{E}_{P_{t+k,N}}$ from (2.20) and (4.43), noting $F^0_{CN} = U^T_{fe}\Lambda^2_N$ and $F^1_{CN} = \Lambda^2_K$. Thus follow as:

$$\begin{aligned}\hat{\Phi}_{P_{t+k,N}} &= P_{CN}\hat{E}_{P_{t+k,N}} + F^0_{CN}U^0_{t,N} + F^1_{CN}\,k_c \\ &= P_{CN}(D_{P_{t+k,N}} + C_{PN}A_N\hat{x}(t+k|t) + V_{PN}U^0_{t,N}) + F^0_{CN}U^0_{t,N} + F^1_{CN}\,k_c \\ &= P_{CN}D_{P_{t+k,N}} + P_{CN}C_{PN}A_N\hat{x}(t+k|t) + U^T_{fe}V^T_{PN}V_{PN}U^0_{t,N} + F^0_{CN}U^0_{t,N} + F^1_{CN}\,k_c\end{aligned}$$

But $U_{t,N} = U_{fe}(t)k_c(t)$ and hence:

$$\begin{aligned}\hat{\Phi}_{P_{t+k,N}} &= P_{CN}D_{P_{t+k,N}} + P_{CN}C_{PN}A_N\hat{x}(t+k \mid t) + U^T_{fe}V^T_{PN}V_{PN}U^0_{t,N} + F^0_{CN}U_{fe}k_c + \Lambda^2_K\,k_c \\ &= P_{CN}D_{P_{t+k,N}} + P_{CN}C_{PN}A_N\hat{x}(t+k|t) + U^T_{fe}V^T_{PN}V_{PN}U_{fe}k_c + U^T_{fe}\Lambda^2_N U_{fe}k_c + \Lambda^2_K\,k_c \\ &= P_{CN}D_{P_{t+k,N}} + P_{CN}C_{PN}A_N\hat{x}(t+k|t) + \left(U^T_{fe}(V^T_{PN}V_{PN} + \Lambda^2_N)U_{fe} + \Lambda^2_K\right)k_c\end{aligned}$$

Thence, from (4.43), the predicted signal:

$$\hat{\Phi}_{P_{t+k,N}} = P_{CN}D_{P_{t+k,N}} + P_{CN}C_{PN}A_N\hat{x}(t+k|t) + X_N k_c(t) \tag{4.61}$$

From the related case in earlier sections, the *multi-step MV* optimal predictive control adjusts to zero $\hat{\Phi}_{P_{t+k,N}} = 0$ the first squared term in (4.59), and the optimal control, as a result, follows from setting (4.61) to zero, providing:

$$k_c(t) = -X^{-1}_N P_{CN}\left(D_{P_{t+k,N}} + C_{PN}A_N\hat{x}(t+k|t)\right)$$

This expression is equivalent to the *RS* control gains vector, as briefed in the theorem below.

Theorem II: Equivalent Cost-Minimisation Problem

The system and its expectations presented in Chapter 2. The input subsystem is set $W_{1k}=I$, and the optimal *RS* controls vector is given by (4.28). Thus, to minimise the *RS cost-index* (4.2), define the second *cost-index* to have a *multi-step MV* type:

$$\tilde{J}(t)=E\{\Phi^T_{P_{t+k,N}}\Phi_{P_{t+k,N}}\mid t\} \tag{4.62}$$

where

$$\Phi_{P_{t+k,N}}=P_{CN}E_{P_{t+k,N}}+F^0_{CN}U^0_{t,N}+F^1_{CN}k_c \tag{4.63}$$

And the *cost-function* weightings $P_{CN}=U^T_{fe}V^T_{PN}$, $F^0_{CN}=U^T_{fe}\Lambda^2_N$ and $F^1_{CN}=\Lambda^2_K$. Define $V_{PN}=C_{PN}B_N+E_{PN}$ and $X_N=U^T_{fe}(V^T_{PN}V_{PN}+\Lambda^2_N)U_{fe}+\Lambda^2_K$ then, the optimal control gains vector:

$$\begin{aligned} k_c(t)&=-X_N^{-1}P_{CN}\left(D_{P_{t+k,N}}+C_{PN}A_N\hat{x}(t+k|t)\right)\\ &=-X_N^{-1}U^T_{fe}V^T_{PN}\left(D_{P_{t+k,N}}+C_{PN}A_N\hat{x}(t+k|t)\right)\end{aligned} \tag{4.64}$$

The main result is that this gain vector form is comparable to **Theorem I** or the *RS* controller in (4.28), and the future controls vector, needed in the prediction equations, is provided as:

$$\begin{aligned} U_{t,N}&=U_{fe}(t)\,k_c(t)\\ U_{t,N}&=-U_{fe}\,X_N^{-1}U^T_{fe}V^T_{PN}(D_{Pt+k,N}+C_{PN}A_N\hat{x}(t+k\mid t))\end{aligned} \tag{4.65}$$

Solution: The proof develops by gathering the results of the previous sections.

4.4 RS Unstructured Subsystem

A related problem is analyzed here that illustrates the connection with traditional *transport-delay* compensation schemes. The unstructured input subsystem is included here, which means the model for W_{1k} need not be known, but the output of W_{1k} can be determined for a given input. Recall that the real input to the system is the control signal $u(t)$, illustrated in Figure 2-2, instead of the input to the unstructured subsystem $u_0(t)$. The *cost-function* for the problem may comprise an added control signal costing term.

If the *k-steps* is the size of the smallest delay in every plant output channel, this means that the output is affected by the control signal at instant *t* at least *k-steps* later. The costing of the control signal then becomes:

$$\left(F_c u\right)(t)=z^{-k}\left(F_{ck} u\right)(t) \tag{4.66}$$

The control weighting operator F_{ck} has an inverse and assumed *full-rank*. So, a new signal is considered to minimise its variance. In parallel to the earlier *RS* problem, a *multi-step cost-index* are specified related to *cost-function* in (4.52).

Extended Multi-Step Cost-Index:

$$J_p=E\{\Phi^{0^T}_{P_{t+k,N}}\Phi^{0}_{P_{t+k,N}}\mid t\} \tag{4.67}$$

The signal $\Phi^{0}_{P_{t+k,N}}$ is an extension of (4.63), defined to comprise the costing term of the future control signal:

$$\begin{aligned}\Phi^{0}_{P_{t+k,N}} &= P_{CN}E_{P_{t+k,N}} + F^{0}_{CN}U_{t,N} + F^{1}_{CN}k_c + U^{T}_{fe}F_{ck,N}U_{t,N} \\ &= U^{T}_{fe}V^{T}_{PN}E_{P_{t+k,N}} + \left(U^{T}_{fe}\Lambda^{2}_{N}U_{fe} + \Lambda^{2}_{K} + U^{T}_{fe}F_{ck,N}U_{fe}\right)k_c\end{aligned} \tag{4.68}$$

where the function $F_{ck,N}U_{t,N}$ will usually be specified to have a straightforward diagonal structure:

$$(F_{ck,N}U_{t,N}) = diag\left\{\left(F_{ck}u\right)(t), \left(F_{ck}u\right)(t+1), ..., \left(F_{ck}u\right)(t+N)\right\} \tag{4.69}$$

Also, note that the vector of inputs:

$$U^{0}_{t,N} = (W_{1k,N}U_{t,N})$$

where $W_{1k,N}$ likewise has a *block-diagonal-matrix* form:

$$\begin{aligned}(W_{1k,N}\,U_{t,N}) &= diag\{W_{1k}, W_{1k}, \dots, W_{1k}\}U_{t,N} \\ &= [(W_{1k}u)(t)^{T}, ..., (W_{1k}u)(t+N)^{T}]^{T}\end{aligned} \tag{4.70}$$

4.4.1 The RS Problem Solution

This problem's solution develops from comparable steps to those given in Section 4.2.3 and is presented briefly here.

Notice from (4.53) that $\Phi^{0}_{P_{t,N}} = \Phi_{P_{t,N}} + z^{-k}(F_{ck,N}U_{t,N})$ and $\Phi^{0}_{P_{t+k,N}} = \hat{\Phi}^{0}_{P_{t+k,N}} + \tilde{\Phi}^{0}_{P_{t+k,N}}$ where

$$\begin{aligned}\hat{\Phi}^{0}_{P_{t+k,N}} &= \hat{\Phi}_{P_{t+k,N}} + (F_{ck,N}U_{t,N}) \\ &= P_{CN}\hat{E}_{P_{t+k,N}} + F^{0}_{CN}\,U^{0}_{t,N} + F^{1}_{CN}\,k_{c} + U^{T}_{fe}F_{ck,N}U_{t,N}\end{aligned} \tag{4.71}$$

The error of the estimation:

$$\tilde{\Phi}^{0}_{P_{t+k,N}} = \tilde{\Phi}_{P_{t+k,N}} = U^{T}_{fe}V^{T}_{PN}\tilde{E}_{P_{t+k,N}} \tag{4.72}$$

The future predicted error values in the signal $\hat{\Phi}^{0}_{P_{t+k,N}}$ contains the estimated weighted errors vector $\hat{E}_{P_{t+k,N}}$ and are orthogonal to $\tilde{E}_{P_{t+k,N}}$.

Also, note that the estimation error is *zero-mean* and its product value with any known signal is anticipated null, and so the *cost-function* is given as:

$$\tilde{J}(t) = \hat{\Phi}^{0^T}_{P_{t+k,N}} \hat{\Phi}^{0}_{P_{t+k,N}} + \tilde{J}_1(t) \tag{4.73}$$

where the optimal control delivers $\hat{\Phi}^{0}_{P_{t+k,N}} = 0$

Optimality Condition: The optimality condition that governs optimal solution, consequently, is:

$$\begin{aligned} &P_{CN}\hat{E}_{P_{t+k,N}} + F^1_{CN}\, k_c + (U^T_{fe}F_{ck,N} + F^0_{CN}W_{1k,N})U_{t,N} = 0 \\ &P_{CN}\hat{E}_{P_{t+k,N}} + \left(\Lambda^2_K + U^T_{fe}(F_{ck,N} + \Lambda^2_N W_{1k,N})U_{fe}(t)\right)k_c(t) = 0 \end{aligned} \tag{4.74}$$

4.4.2 The RS Control Signal

To minimise the *cost-index* (4.73), the future optimal control signals vector from the optimality condition in (4.74) satisfies:

$$k_c(t) = -\left(\Lambda^2_K + U^T_{fe}(F_{ck,N} + \Lambda^2_N W_{1k,N})U_{fe}(t)\right)^{-1} P_{CN}\hat{E}_{P_{t+k,N}} \tag{4.75}$$

A different solution of (4.74), in a further easier form for application, gives:

$$k_c(t) = -\Lambda^{-2}_K\left(P_{CN}\hat{E}_{P_{t+k,N}} + U^T_{fe}(F_{ck,N} + \Lambda^2_N W_{1k,N})U_{fe}(t)k_c(t)\right) \tag{4.76}$$

Simplifying the Expression for the Gain: Equation (4.76), for future predicted error signals vector, is substituted to obtain a useful expression for the *gain-vector*. Noting the expression (2.20) for $\hat{E}_{P_{t+k,N}}$ then (4.71) is given as:

$$\begin{aligned}\hat{\Phi}^0_{P_{t+k,N}} &= P_{CN}\hat{E}_{P_{t+k,N}} + F^0_{CN}U^0_{t,N} + F^1_{CN}k_c + U^T_{fe}F_{ck,N}U_{t,N} \\ &= P_{CN}\left(D_{P_{t+k,N}} + C_{PN}A_N\hat{x}(t+k|t) + V_{PN}U^0_{t,N}\right) \\ &+ F^0_{CN}U^0_{t,N} + F^1_{CN}k_c + U^T_{fe}F_{ck,N}U_{t,N}\end{aligned} \tag{4.77}$$

The optimality condition becomes:

$$P_{CN}\left(D_{P_{t+k,N}} + C_{PN}A_N\hat{x}(t+k|t) + V_{PN}U^0_{t,N}\right) + F^0_{CN}U^0_{t,N} + F^1_{CN}k_c + U^T_{fe}F_{ck,N}U_{t,N} = 0$$

or

$$P_{CN}\left(D_{P_{t+k,N}} + C_{PN}A_N\hat{x}(t+k|t)\right) + \left(U^T_{fe}F_{ck,N} + (P_{CN}V_{PN} + F^0_{CN})W_{1k,N}\right)U_{t,N} + F^1_{CN}k_c = 0$$

Recall $P_{CN} = U^T_{fe}V^T_{PN}, F^0_{CN} = U^T_{fe}\Lambda^2_N, F^1_{CN} = \Lambda^2_K$ and hence:

$$\begin{aligned}&P_{CN}\left(D_{P_{t+k,N}} + C_{PN}A_N\hat{x}(t+k|t)\right) \\ &+U^T_{fe}\left(F_{ck,N} + (V^T_{PN}V_{PN} + \Lambda^2_N)W_{1k,N}\right)U_{fe}k_c + \Lambda^2_K k_c = 0\end{aligned} \tag{4.78}$$

To simplify this equation, write $P_{CN} = U^T_{fe}V^T_{PN}$ and introduce the matrix:

$$C_\phi = P_{CN}C_{PN}A_N = U^T_{fe}V^T_{PN}C_{PN}A_N \tag{4.79}$$

Optimality Condition: The condition for optimality is similar to (4.61) but with added control weighting $F_{ck,N}$.

$$\begin{aligned}&U^T_{fe}V^T_{PN}\left(D_{P_{t+k,N}} + C_{PN}A_N\hat{x}(t+k|t)\right) + \\ &U^T_{fe}(V^T_{PN}V_{PN} + \Lambda^2_N)W_{1k,N}U_{fe}k_c + \left(U^T_{fe}F_{ck,N}U_{fe} + \Lambda^2_K\right)k_c = 0\end{aligned}$$

or

$$\begin{aligned}&\left(P_{CN}D_{P_{t+k,N}} + C_\phi\hat{x}(t+k|t)\right) + \\ &U^T_{fe}(V^T_{PN}V_{PN} + \Lambda^2_N)W_{1k,N}U_{fe}k_c + \left(U^T_{fe}F_{ck,N}U_{fe} + \Lambda^2_K\right)k_c = 0\end{aligned} \tag{4.80}$$

Optimal Control: The two solutions option for future optimal controls vector, in predicted future state estimate terms, therefore, becomes:

$$\begin{aligned} k_c(t) = -\left(U_{fe}^T F_{ck,N} U_{fe} + \Lambda_K^2\right)^{-1} \Big(& P_{CN} D_{P_{t+k,N}} + C_\phi \hat{x}(t+k|t) \\ & + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) W_{1k,N} U_{fe} k_c(t)\Big) \end{aligned} \tag{4.81}$$

or

$$\begin{aligned} k_c(t) = -\left(U_{fe}^T ((V_{PN}^T V_{PN} + \Lambda_N^2) W_{1k,N} + F_{ck,N}) U_{fe} + \Lambda_K^2\right)^{-1} \Big(& P_{CN} D_{P_{t+k,N}} \\ & + C_\phi \hat{x}(t+k|t)\Big) \end{aligned} \tag{4.82}$$

Solution Remarks:

- The control law (4.81) and (4.82) comprises an internal process model, and the applied control law utilizes a *receding-horizon* idea as in the previous *RS* solution.
- Noting $X_N = U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) U_{fe} + \Lambda_K^2$ in the gain (4.82) turns out to be equal to the *RS* controller (4.28), when $W_{1k,N} = I$ and for a few cases as the control signal's costing term $F_{ck,N} \to 0$.

4.4.3 Optimal Predictive Control Signal

The optimal control expression utilizing the present state estimate may now be derived. The expressions (4.81) and (4.82) can be reformed more by replacing for the optimal predicted state expression in (2.25). That is, from the optimality condition (4.80):

$$\begin{aligned} &\left(P_{CN} D_{P_{t+k,N}} + C_\phi \left(A^k \hat{x}(t|t) + T_0(k,z^{-1}) B u_0(t)\right)\right) \\ &+ U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) W_{1k,N} U_{fe} k_c + \left(U_{fe}^T F_{ck,N} U_{fe} + \Lambda_K^2\right) k_c = 0 \end{aligned} \tag{4.83}$$

It is helpful to distinguish the controls vector to be employed at instant *t* and future controls vector. The control at instant *t* is calculated for $N > 0$ using current and future controls vector by defining a matrix C_{I0}=$[I \quad 0]$=$[I, 0, ..., 0]$. This procedure allows the control at instant *t* to be retrieved from current and future controls vector as $u(t) = C_{I0} U_{t,N} = [I, 0, ..., 0] U_{t,N}$.

Also, note $u_0(t) = W_{1k} u(t) = C_{I0} W_{1k,N} U_{fe} k_c(t)$ and the term $\tilde{D}^0_{P_{t+k,N}}$ is defined as:

$$\tilde{D}^0_{P_{t+k,N}} = P_{CN} D_{P_{t+k,N}} + C_\phi A^k \hat{x}(t \mid t) \tag{4.84}$$

Thence, the optimality condition as:

$$\begin{aligned} &\left(\tilde{D}^0_{P_{t+k,N}} + C_\phi T_0(k, z^{-1}) B C_{I0} W_{1k,N} U_{fe} k_c\right) + \\ &U^T_{fe}(V^T_{PN} V_{PN} + \Lambda^2_N) W_{1k,N} U_{fe} k_c + \left(U^T_{fe} F_{ck,N} U_{fe} + \Lambda^2_K\right) k_c = 0 \end{aligned} \tag{4.85}$$

Optimal Gain Vector: The optimal gain vector follows from (4.85) as:

$$\begin{aligned} k_c(t) = -&\left(U^T_{fe}\left(\left(V^T_{PN} V_{PN} + \Lambda^2_N\right) W_{1k,N} + F_{ck,N}\right) U_{fe} + \Lambda^2_K\right)^{-1} \left(\tilde{D}^0_{P_{t+k,N}}\right. \\ &\left. + C_f T_0(k, z^{-1}) B C_{I0} W_{1k,N} U_{fe} k_c(t)\right) \end{aligned} \tag{4.86}$$

An alternative expression for (4.85) is developed from optimality condition as:

$$\begin{aligned} &k_c(t) = \\ &-\left(U^T_{fe} F_{ck,N} U_{fe} + \Lambda^2_K\right)^{-1} \left(\left(\tilde{D}^0_{P_{t+k,N}} + C_f T_0(k, z^{-1}) B \left(C_{I0} W_{1k,N} U_{fe}\right)(t) k_c(t)\right)\right. \\ &\left. + U^T_{fe}\left(V^T_{PN} V_{PN} + \Lambda^2_N\right) W_{1k,N} U_{fe} k_c(t)\right) \end{aligned} \tag{4.87}$$

where $C_{I0} W_{1k,N} = [W_{1k}, 0, ..., 0]$. To simplify this expression let,

$$X_F = U^T_{fe} F_{ck,N} U_{fe} + \Lambda^2_K \tag{4.88}$$

Optimal Control: Two possible expressions for future optimal controls vector, in current state estimate terms, from (4.86):

$$\begin{aligned} k_c(t) = &-\left(U_{fe}^T\left(\left(V_{PN}^T V_{PN} + \Lambda_N^2\right)W_{1k,N} + F_{ck,N}\right)U_{fe} + \Lambda_K^2\right)^{-1}\left(\tilde{D}^0_{P_{t+k,N}}\right. \\ &\left. + C_f T_0(k,z^{-1})BC_{I0}W_{1k,N}U_{fe}k_c(t)\right) \end{aligned} \tag{4.89}$$

and from (4.87) (in a friendly implementation form):

$$\begin{aligned} k_c(t) &= -X_F^{-1}\left(P_{CN}\tilde{D}^0_{P_{t+k,N}} + (C_f T_0(k,z^{-1})BC_{I0}\right. \\ &\quad \left. + U_{fe}^T(V_{PN}^T V_{PN} + \Lambda_N^2))W_{1k,N}U_{fe}k_c(t)\right) \\ &= -X_F^{-1}\left(P_{CN}D_{P_{t+k,N}} + P_{CN}C_{PN}A_N A^k \hat{x}(t\,|\,t)\right. \\ &\quad \left. + \left(C_f T_0(k,z^{-1})BC_{I0} + U_{fe}^T\left(V_{PN}^T V_{PN} + \Lambda_N^2\right)\right)W_{1k,N}U_{fe}k_c(t)\right) \end{aligned} \tag{4.90}$$

Theorem III: Optimal *RS* Control Law

Consider plant linear part, disturbance and error weighting subsystem. Add in augmented *state-space* form (2.3), (2.4) to input from the stable first subsystem W_{1k} of the plant. Specify for $N > 0$, the predictive control *multi-step cost-function* to be minimised and comprises future cost terms sum in vector form as:

$$J_p = E\{\Phi^{0^T}_{P_{t+k,N}} \Phi^{0}_{P_{t+k,N}} \mid t\} \tag{4.91}$$

where from (4.92), signal $\Phi^{0}_{P_{t+k,N}}$ depends on future error, control signal costing, and input terms:

$$\Phi^{0}_{P_{t+k,N}} = P_{CN} E_{P_{t+k,N}} + F^{0}_{CN} U_{t,N} + F^{1}_{CN} k_c + U^{T}_{fe} F_{ck,N} U_{t,N} \tag{4.92}$$

The *cost-function* weightings of the error and *control-input* are established in the *RS* problem (4.1), and they decide the *cost-index* block matrix forms:

$$P_{CN} = U^{T}_{fe} V^{T}_{PN},\ F^{0}_{CN} = U^{T}_{fe} \Lambda^{2}_{N},\ V_{PN} = C_{PN} B_{N} + E_{PN}$$
$$X_{N} = U^{T}_{fe} (V^{T}_{PN} V_{PN} + \Lambda^{2}_{N}) U_{fe} + \Lambda^{2}_{K},\ C_{\phi} = P_{CN} C_{PN} A_{N}$$

The parameterised controller gain weighting $F^{1}_{CN} = \Lambda^{2}_{K}$ is also included, and the control signal *cost-function* weighting $F_{ck,N}$ is diagonal control weighting (4.69) dependent upon $(F_c u)(t) = F_{ck} u(t-k)$, where *k* is explicit *transport-delay*.

The vector of optimal gain to minimise the variance (4.91), in the predicted state terms, is:

$$\begin{aligned} k_c = -\left(U^{T}_{fe} F_{ck,N} U_{fe} + \Lambda^{2}_{K}\right)^{-1} \Big(\left(P_{CN} D_{P_{t+k,N}} + C_f \hat{x}(t+k \mid t)\right) \\ + U^{T}_{fe}\left(V^{T}_{PN} V_{PN} + \Lambda^{2}_{N}\right) W_{1k,N} U_{fe} k_c\Big) \end{aligned} \tag{4.93}$$

The equivalent expressions, in the current state estimate terms:

$$\begin{aligned} k_c = -X_F^{-1}\Big(P_{CN}D_{P_{t+k,N}} + C_\phi A^k \hat{x}(t \mid t) \\ +\Big(C_\phi T_0(k,z^{-1})BC_{I0} + U_{fe}^T\big(V_{PN}^T V_{PN} + \Lambda_N^2\big)\Big)W_{1k,N}U_{fe}k_c\Big) \end{aligned} \tag{4.94}$$

where the finite impulse response term:

$$T_0(k,z^{-1}) = (I - A^k z^{-k})(zI - A)^{-1}$$

And the future controls vector are determined as $U_{t,N} = U_{fe}(t)k_c(t)$.

Solution: The optimal control proof for the case when the input subsystem is used was provided prior to the theorem, which outlines the main findings.

Remarks:

- The two *NPGMV* control signal's styles have an advantage for the two similar structures for applying the controller. The first is illustrated in Figure 4-2 and is in the predicted state terms.
- The controller comprises a *Kalman* predictor part, and it is essential to say, the *Kalman filter* order just depends on the plant *delay-free* linear subsystems. Therefore, the estimator order doesn't grow with channel delays.

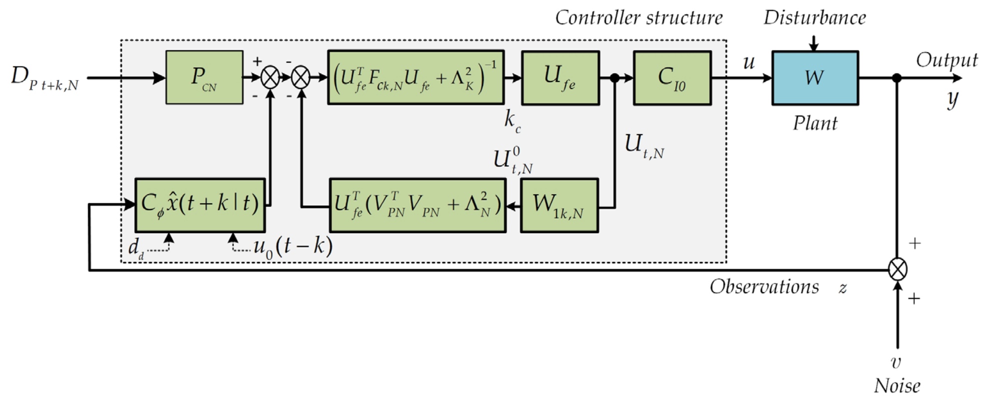

Figure 4-2: Kalman predictor and the RS control signal generation

4.4.4 RS Implementation

To calculate the future controls vector for $t > 0$, define the matrix:

$$C_{0I} = \begin{bmatrix} 0 & I_N \end{bmatrix} \tag{4.95}$$

where

$$U_{t,N}^f = C_{0I} U_{t,N} = \begin{bmatrix} 0 & I_N \end{bmatrix} \begin{bmatrix} u(t) \\ \vdots \\ u(t+N) \end{bmatrix} = \begin{bmatrix} u(t+1) \\ \vdots \\ u(t+N) \end{bmatrix} \tag{4.96}$$

Since the diagonal block formation of the control signal costing $F_{ck,N}$, observe that $C_{I0} F_{ck,N}^{-1} = [F_{ck}^{-1}, 0, \ldots, 0] = F_{ck}^{-1} C_{I0}$. The optimal control at instant t is processed, utilizing (4.90) as:

$$\begin{gathered} k_c = -X_F^{-1} \Big(P_{CN} D_{Pt+k,N} + C_\phi A^k \hat{x}(t \mid t) + \\ \Big(C_f T_0(k, z^{-1}) B C_{I0} + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) \Big) W_{1k,N} U_{fe} k_c \Big) \end{gathered} \tag{4.97}$$

4.4.5 Control Signal Alternative Expression

Now obtain an alternative representation for the *NPGMV* optimal control, which motivates the controller's implementation illustrated in Figure 4-3.

Using (4.94) and (2.25), (2.27) as:

$$\begin{aligned} k_c = -X_F^{-1}\Big(& P_{CN}D_{P_{t+k,N}} + C_\phi A^k \hat{x}(t\,|\,t) \\ &+ \big(C_\phi T_0(k,z^{-1})BC_{I0} + U_{fe}^T\big(V_{PN}^T V_{PN} + \Lambda_N^2\big)\big)W_{1k,N}U_{fe}k_c\Big) \end{aligned}$$

or

$$\begin{aligned} k_c = -X_F^{-1}\Big(& P_{CN}D_{P_{t+k,N}} + C_\phi A^k T_{f1}(z^{-1})\big(z(t)-d(t)\big) + C_\phi A^k T_{f2}(z^{-1})u_0(t) \\ &+ \big(C_\phi T_0(k,z^{-1})BC_{I0} + U_{fe}^T\big(V_{PN}^T V_{PN} + \Lambda_N^2\big)\big)W_{1k,N}U_{fe}k_c\Big) \end{aligned} \tag{4.98}$$

This equation needs the *Kalman filter* two paths to be split up as in Figure 4-3, involving computation of the separate transfers.

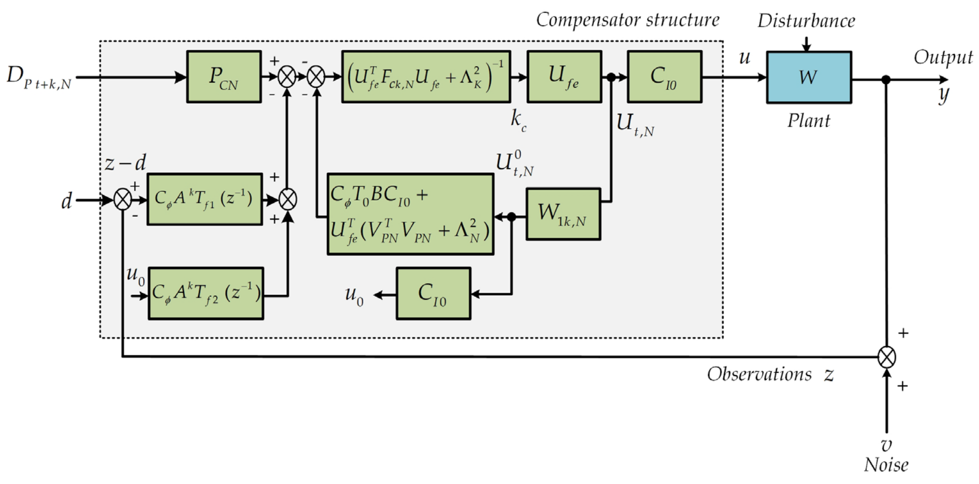

Figure 4-3: Signal generation and the RS controller modules

4.5 Chapter Summary

This Chapter summarises the linear *RS* control solution for the *discrete-time state-space* system description given in Chapter 2. The Chapter started by introducing and defining the linear *RS* controller parameterization in Section 4.2. The *RS* optimisation problem and equivalent *cost-function* were described in Section 4.3, and this section concluded with a brief on the *cost-function* tuning variable. The second unstructured subsystem is presented in Section 4.4 by introducing the *RS* problem solution and showing how the *RS* control signal is generated. This section also provides a discussion on the *RS* implementation issues. In the next Chapter, the nonlinear *RS* control solution will be introduced for *qLPV* systems and a class of *HS*.

Chapter 5 Nonlinear RS Control

In this Chapter, we describe the nonlinear *RS* controller solution for the system described in the *state-space* description given in Section 2.2 from Chapter 2. The *RS* controller solution is also provided for a class of *HS*. An introduction to nonlinear *MPC* is given in Section 5.1. The nonlinear *RS* solution is given in Section 5.2, and details are provided for the gains vector, the nonlinear *RS* controller and the *RS* control problem. Section 5.5 introduces a hybrid *RS* solution and gives detailed information on *PWA*, *SD*, and *LHA*. Sections 5.5.1 and 5.5.3. assigns the conditions on which the *SD* system can be utilized for the nonlinear *RS* solution presented in Section 5.2.

5.1 Introduction

This section offers a short introduction to nonlinear *MPC* theory. It sets the basis on which the *LPV* predictive paradigm will be analysed in the next for the nonlinear *RS* controller *state-space* formulation.

Early developments on the concept of predictive control are traced back to the innovative approach on optimal *state-feedback* control by *Kalman* and following methods of *LQR* and *LQG* controllers. In all these methods, a *cost-function* was involved in which output errors and control actions were penalised. The plant optimal control inputs were computed for the minimisation of that function. This framework allowed dealing with both tracking and regulating processes whilst control effort was maintained within a specified acceptable range at the same time.

Various practical extensions to these schemes were addressed over the years, like the addition of an integral action and direct control of plant outputs by modifying the *cost-function*. However, practical limitations in real applications motivated a predictive control methodology to address the following issues [66]:

- Input, state and output constraints.
- Process nonlinearities.
- Model uncertainty.

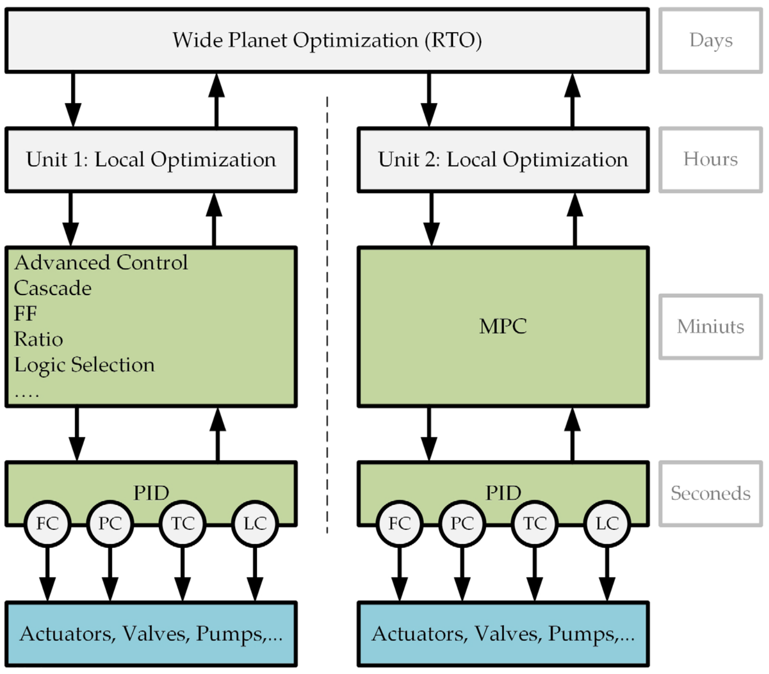
Wide Planet Optimization (RTO)
Days
Unit 1: Local Optimization
Unit 2: Local Optimization
Hours
Advanced Control
Cascade
FF
Ratio
Logic Selection
....
MPC
Miniuts
PID
FC
PC
TC
LC
PID
FC
PC
TC
LC
Seconeds
Actuators, Valves, Pumps,...
Actuators, Valves, Pumps,...

Figure 5-1: Process hierarchical structure of MPC and classical control

In early process applications, *MPC* strategy started as the supervisory control module. The *MPC* was located at the *higher-level* of the hierarchical structure as in Figure 5-1. Within this framework, the *MPC* controller functioned more like an optimiser that determined the optimal setpoints for the *low-level* control loops (such as *PID*, *lead-lag* controllers). This setpoint computation was often based on plant economics and other slowly varying systems, whereas control at *low-level* dealt with faster dynamic control loops. Initial *MPC* practical developments methodology were done by groups led by [67], [68]. And the first methodology involved impulse response models identified from *open-loop* tests to generate predictions. It considers the input and output constraints into the calculated optimal solution using the iterative algorithm, the mathematical dual of identification.

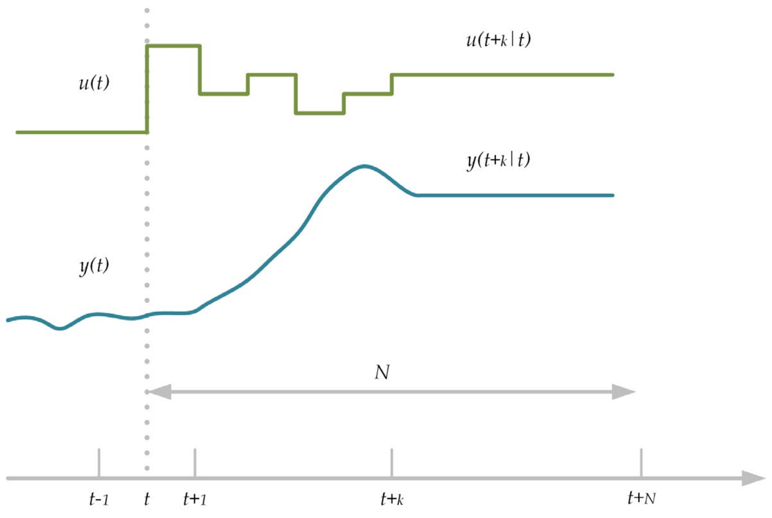

Figure 5-2: The strategy of the MPC

The second approach involved linear step response models for the same purpose and utilized a *least-squares cost-function* to be solved within a finite horizon. Both schemes could be employed for either scaler or *MIMO* systems. They could provide both trackings of a future trajectory whilst regulating the control cost by applying sufficient weighting on control signal moves. The second, however, in its initial development, did not consider constraints in its *least-squares* solution, and further development by *Shell* led to reformulating the *DMC* problem as a *QP* problem to deal with constraints explicitly [69].

To summarise, subject to input, state and output constraints, *MPC* philosophy involves solving an online *open-loop* finite horizon quadratic optimal control problem.

At each iteration *k* of the algorithm, current inputs sequence and measured variables are used within the internal discrete mathematical process model for system response future times prediction. This prediction horizon is computed over a specified time window N_p discrete steps and used within the quadratic *MPC cost-function* [70].

The latter is then minimised for the optimal controls vector that achieves this objective subject to constraints. The length of the optimal controls vector is defined as the control horizon N_u. The *receding-horizon* principle is then employed, according to which just the first element of the calculated optimal controls vector is utilized to control the plant. These steps are restarted in the next iteration for the updated state, as depicted in Figure 5-2.

The solution of this dynamic optimisation problem at each iteration can be considered an indirect implementation of feedback. It is a useful feature because it can compensate for uncertainties due to model mismatch, disturbances, or noise present in the process.

The linear *MPC* algorithm has been found to perform adequately, mostly in relatively simple and slowly varying processes and handle both soft and hard constraints where these apply. However, with the increasing complexity of processes and constraints set by more demanding designs and regulations, the control solution must operate the system in a broad conditions range. This task, coupled with the inherent nonlinear characteristics of most natural processes, motivates the development of more generic control schemes that can accommodate these requirements.

5.1.1 Nonlinear Predictive Control

Linear *MPC* uses linear models to generate predictions and handle constraints. In contrast, nonlinear *MPC* algorithms are about using nonlinear models and constraints in the optimal solution. The process of developing an adequate nonlinear model for control design is an obstacle, given that the fundamental strength of nonlinear approaches lies in the model fidelity. These models can be obtained either via first principle modelling or other methods like *black-box* identification [71].

Frequently, a combination of both yields efficient models for this purpose, known as *grey-box* modelling. Some methods use neural networks, fuzzy logic [72] etc. Nonlinear *MPC* may include offline optimisation like neighbouring extremals in which the optimal control problem full solution is derived offline. In the explicit schemes, the sequence of calculations are precomputed across the operating range, and the appropriate controller is chosen based upon the current value of the system state. However, this method involves computational complexity when the system has an extended operating range or many states.

5.1.2 Implicit MPC

In this category of implicit *MPC* schemes, an *open-loop* future control signal trajectory is used rather than a feedback control law. Thus, online optimization is executed, and the optimization problem is resolved at every instant, given measured or estimated state or output. Then for constrained systems, the optimal control is calculated utilizing a *QP* solver. The process output becomes an implicit nonlinear function of a controller's current state and the additional variables, for instance, the current reference values [73]. The constrained *MPC* techniques solve a *QP* instantly to find out the optimal control. Referable to the calculations complication, implicit *MPC* imposes extra challenges when applied to rapid systems or small sampling times.

5.1.3 Explicit MPC

The fast advancement in *multi-parametric* programming techniques led to an optimization solution for some problems in an explicit manner. A control law was generated as a parametric function [74]. The explicit *MPC* algorithms complete most of the calculations offline. The optimization problem is worked out parametrically for all sets of variables, which fulfil the design constraints as presented in [75].

The main reward of the implicit schemes is abstract simplicity and how the constraints are contained. Still, the implicit approach demands a considerable measure of calculations that need to be executed online. Consequently, most utilized applications are somewhat slow dynamics because the optimization problem solution is needed at each sampling time.

The explicit *MPC* problems result in online *MPC* control laws, and there is no need for optimization problem solutions at every time step. Instead, there is a performance index optimisation, constraints vector and parameters vector. The variables optimization solution is then retrieved as a parametric function and the parameters space domains in which these functions are valid. In general, the solution found by *multi-parametric QP*, a linear *MPC* controller gains set, is *pre-calculated* for various operating points set.

The optimal control problem is consequently solved only once, and the results are stored for later online use. *Gain-scheduling* or switching is applied to choose the nearest gains in proportion to the current operating point. The explicit *MPC* algorithm has various crucial features:

- The online processing time is low and of microseconds, if needed, and always desirable for rapid systems.
- Implementation online is almost as simple as for classical controllers.
- The *real-time* code is straightforward, short and efficient.

However, explicit *MPC* becomes less desirable when the number of regions rises, as excessively memory usage is needed. It seems more desirable for systems of *low-order* when states are restricted to about ten or less [75].

5.2 Nonlinear RS Control

The *RS* controller optimization is suitably established, but the method defined in the Chapter is new. While no approximation is needed, as in the optimization technique that appears in [76], [77], the controller is formed and specified in a structure in which *pre-specified* functions multiplied by gains found through an optimization.

These functions could be the proportional, integrator, and derivative functions as in *PID* control; however, they could be far more general. Thus, a sum of N_e dynamic functions can be chosen for its required unique frequency response property.

The control signal is recognized as follow:

$$\begin{aligned} u(t) &= \sum_{j=1}^{N_e} f_j\left(z^{-1}, k_j(t)\right) e_0(t) \\ &= f_1\left(z^{-1}, k_1(t)\right) e_0(t) + f_2\left(z^{-1}, k_2(t)\right) e_0(t) + \cdots + f_{N_e}\left(z^{-1}, k_{N_e}(t)\right) e_0(t) \end{aligned} \tag{5.1}$$

where $f_j(z^{-1}, k_j(t))$ and $k_j(t)$ are the controller structure functions and gains, individually. The functions are *frequency-sensitive* dynamic terms selected by the control designer. The gains are *time-varying* gain vectors set, and in the simple case of a scalar *PID*, there are $N_e = 3$ function block terms. Note that a more controller general structure requires past control terms on the right side of the equation (5.1), which is an essential extension.

***MIMO RS* Control:** Specific matrix terms are essential for the *MIMO* case. The functions $f_j\left(z^{-1}, k_j(t)\right)$ and gains $k_j(t)$ are specified in the next matrix:

$$f_j\left(z^{-1}, k_j(t)\right) = \begin{bmatrix} f_{11}^j(z^{-1}) k_{11}^j(t) & \cdots & f_{1r}^j(z^{-1}) k_{1r}^j(t) \\ \vdots & f_{22}^j(z^{-1}) k_{22}^j(t) & \vdots \\ & & \ddots \\ f_{m1}^j(z^{-1}) k_{m1}^j(t) & \cdots & f_{mr}^j(z^{-1}) k_{mr}^j(t) \end{bmatrix} \tag{5.2}$$

where *r* is the control input signal channels as:

$$e_0(t) = \left[e_{01}^T(t) \quad e_{02}^T(t) \quad \cdots \quad e_{0r}^T(t) \right]^T \tag{5.3}$$

The terms in the *RS* control (5.1) have the form $f_j(z^{-1}, k_j(t)) e_0(t)$. As of (5.2) and (5.3), the impact of the j^{th} function term in every channel is described as:

$$f_j(z^{-1}, k_j(t)) e_0(t) = \begin{bmatrix} f_{11}^j(z^{-1}) k_{11}^j e_{01}(t) + f_{12}^j(z^{-1}) k_{12}^j e_{02}(t) + \ldots + f_{1r}^j(z^{-1}) k_{1r}^j e_{0r}(t) \\ f_{21}^j(z^{-1}) k_{21}^j e_{01}(t) + f_{22}^j(z^{-1}) k_{22}^j e_{02}(t) + \ldots + f_{2r}^j(z^{-1}) k_{2r}^j e_{0r}(t) \\ \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ f_{m1}^j(z^{-1}) k_{m1}^j e_{01}(t) + f_{m2}^j(z^{-1}) k_{m2}^j e_{02}(t) + \ldots + f_{mr}^j(z^{-1}) k_{mr}^j e_{0r}(t) \end{bmatrix} \tag{5.4}$$

Control Signal: The parameterized controller control signal follows from (5.1) and (5.5).

$$
\begin{aligned}
u(t) &= \sum_{j=1}^{N_e} f_j(z^{-1}, k_j(t))\, e_0(t) \\
&= f_1\left(z^{-1}, k_1(t)\right)e_0(t) + f_2\left(z^{-1}, k_2(t)\right)e_0(t) + \cdots + f_{N_e}\left(z^{-1}, k_{N_e}(t)\right)e_0(t)
\end{aligned}
$$

$$
u(t) = \begin{bmatrix}
\sum_{j=1}^{N_e} \left\{ f_{11}^j(z^{-1})k_{11}^j\, e_{01}(t) + f_{12}^j(z^{-1})k_{12}^j\, e_{02}(t) + \ldots + f_{1r}^j(z^{-1})k_{1r}^j\, e_{0r}(t) \right\} \\
\sum_{j=1}^{N_e} \left\{ f_{21}^j(z^{-1})k_{21}^j e_{01}(t) + f_{22}^j(z^{-1})k_{22}^j e_{02}(t) + \ldots + f_{2r}^j(z^{-1})k_{2r}^j e_{0r}(t) \right\} \\
\vdots \qquad \vdots \qquad \vdots \\
\sum_{j=1}^{N_e} \left\{ f_{m1}^j(z^{-1})k_{m1}^j e_{01}(t) + f_{m2}^j(z^{-1})k_{m2}^j e_{02}(t) + \ldots + f_{mr}^j(z^{-1})k_{mr}^j e_{0r}(t) \right\}
\end{bmatrix} \tag{5.5}
$$

5.2.1 Vector of Gains

The above results for the multivariable case appear rather complicated, and to reorganize the representation for the *RS* controller (5.5), the following functions vectors row and gains vectors column are specified.

Allow the leading row in (5.5) be reworked utilizing:

$$
\begin{aligned}
f_e^{11} &= \left[f_{11}^1(z^{-1})e_{01}(t) \quad f_{11}^2(z^{-1})e_{01}(t) \quad \cdots \quad f_{11}^{N_e}(z^{-1})e_{01}(t) \right] \\
f_e^{12} &= \left[f_{12}^1(z^{-1})e_{02}(t) \quad f_{12}^2(z^{-1})e_{02}(t) \quad \cdots \quad f_{12}^{N_e}(z^{-1})e_{02}(t) \right] \cdots \\
f_e^{1r} &= \left[f_{1r}^1(z^{-1})e_{0r}(t) \quad f_{1r}^2(z^{-1})e_{0r}(t) \quad \cdots \quad f_{1r}^{N_e}(z^{-1})e_{0r}(t) \right]
\end{aligned}
$$

and the second row:

$$
\begin{aligned}
f_e^{21} &= \left[f_{21}^1(z^{-1})e_{01}(t) \quad f_{21}^2(z^{-1})e_{01}(t) \quad \cdots \quad f_{21}^{N_e}(z^{-1})e_{01}(t) \right] \cdots \\
f_e^{2r} &= \left[f_{2r}^1(z^{-1})e_{0r}(t) \quad f_{2r}^2(z^{-1})e_{0r}(t) \quad \cdots \quad f_{2r}^{N_e}(z^{-1})e_{0r}(t) \right]
\end{aligned}
$$

and the final term:

$$f_e^{mr} = \left[f_{mr}^{1}(z^{-1})e_{0r}(t) \quad f_{mr}^{2}(z^{-1})e_{0r}(t) \quad \cdots \quad f_{mr}^{N_e}(z^{-1})e_{0r}(t) \right]$$

Define the *block-diagonal-matrix*:

$$F_e(t) = diag\left\{ e_{f1}(t) \quad e_{f2}(t) \quad \cdots \quad e_{fm}(t) \right\} \tag{5.6}$$

$$= diag\left\{ [f_e^{11} \quad f_e^{12} \quad \cdots \quad f_e^{1r}], \ [f_e^{21} \quad f_e^{22} \quad \cdots \quad f_e^{2r}], \ \cdots \ ,[f_e^{m1} \quad f_e^{m2} \quad \cdots \quad f_e^{mr}] \right\}$$

where $e_{f_s}(t) = \left[f_e^{s1} \quad f_e^{s2} \quad \cdots \quad f_e^{sr} \right]$ for $s \in \left[1, m\right]$.

The following gains in every channel related to the separate inputs are labelled to match corresponding first row functions, second row, etc.:

$$\begin{gathered}
k_c^{11} = \begin{bmatrix} k_{11}^{1} \\ k_{11}^{2} \\ \vdots \\ k_{11}^{N_e} \end{bmatrix}, \ k_c^{12} = \begin{bmatrix} k_{12}^{1} \\ k_{12}^{2} \\ \vdots \\ k_{12}^{N_e} \end{bmatrix}, \ \dots \ , \ k_c^{1r} = \begin{bmatrix} k_{1r}^{1} \\ k_{1r}^{2} \\ \vdots \\ k_{1r}^{N_e} \end{bmatrix}, \\
k_c^{21} = \begin{bmatrix} k_{21}^{1} \\ k_{21}^{2} \\ \vdots \\ k_{21}^{N_e} \end{bmatrix}, \ k_c^{22} = \begin{bmatrix} k_{22}^{1} \\ k_{22}^{2} \\ \vdots \\ k_{22}^{N_e} \end{bmatrix}, \ \dots \ , \ k_c^{2r} = \begin{bmatrix} k_{2r}^{1} \\ k_{2r}^{2} \\ \vdots \\ k_{2r}^{N_e} \end{bmatrix}, \cdots, k_c^{mr} = \begin{bmatrix} k_{mr}^{1} \\ k_{mr}^{2} \\ \vdots \\ k_{mr}^{N_e} \end{bmatrix}
\end{gathered} \tag{5.7}$$

Total Gain Vector: The structure of the total gain vector is:

$$\begin{aligned}
k_c &= \left[k_{c1}^T \quad k_{c2}^T \quad \cdots \quad k_{cm}^T \right] \\
&= \left[\underbrace{k_c^{11T} \ k_c^{12T} \cdots k_c^{1rT}}_{channel\#1 gains} \quad \underbrace{k_c^{21T} \ k_c^{22T} \cdots k_c^{2rT}}_{channel\#2 gains} \quad \cdots \quad \underbrace{k_c^{m1T} \ k_c^{m2T} \cdots k_c^{mrT}}_{channel\#m gains} \right]^T
\end{aligned} \tag{5.8}$$

Control Signal: The *RS* controller control signal can be processed as in (5.9)

$$u(t) = F_e(t)\,k_c(t) = \begin{bmatrix} e_{f1}(t)k_{c1}(t) \\ e_{f2}(t)k_{c2}(t) \\ \vdots \\ e_{fm}(t)k_{cm}(t) \end{bmatrix} \tag{5.9}$$

5.2.2 Nonlinear RS Controller Form

The *RS* controller gains, which might embody the three gains vector as in a *PID* controller, can be divided into two elements. If the gains are split into a fixed part $\bar{k}_c$ and a *time-varying* part $\tilde{k}_c(t)$ giving:

$$k_c(t) = \bar{k}_c + \tilde{k}_c(t) \tag{5.10}$$

This design creates two interesting cases:

- Letting $\bar{k}_c = 0$ is the absolute gain condition, and the controller full gain vector $k_c(t) = \tilde{k}_c(t)$ is calculated for criterion minimization.
- The *so-called* gain deviation case arises when assigning a constant gain vector $\bar{k}_c \neq 0$, where $\tilde{k}_c(t)$ is calculated for criterion minimization.

The nonlinear *RS* controller is given, utilizing (5.10), as:

$$\begin{aligned} u(t) &= F_e(t)\,k_c(t) = F_e(t)\,\bar{k}_c + F_e(t)\,\tilde{k}_c(t) \\ &= \sum_{j=1}^{N_e}\left\{f_j(z^{-1})\bar{k}_j e_0(t)\right\} + \sum_{j=1}^{N_e}\left\{f_j(z^{-1})\tilde{k}_j e_0(t)\right\} \end{aligned} \tag{5.11}$$

Implementing, say, a *PID* controller in this parallel arrangement involves the first element, which could be a fixed *PID* and another element that has the gains of *time-varying* kind. If there is variation in the reference or disturbance signals, then the optimal *time-varying* gains will vary correspondingly. It is not adaptive in the typical sense, of course, but it does vary gains according to measured signal changes.

The first condition is, of course, when the *PID* controller gains are minimized wholly. The second condition might be applied if a present *PID* controller is available, and this identifies the base gain term $\overline{k}_c$. The calculated gain hence is a variation around the fixed *PID* values. This last matter is understood as when utilizing two *PID* controllers in parallel, where one has fixed gains, and the other has *time-varying* gain. These two possible cases will be considered in the following, but note they do not provide the even optimal solution since the *cost-function* is different in the two cases.

5.2.3 Nonlinear RS Control Problem

For the next analysis, just the *qLPV* subsystem is used from subsystems shown in Figure 2-5, and the input subsystem is ignored by defining $W_{1k} = I$. With dynamic error weighting included, the performance index of the *GPC* [28], [78] that motivates the proposed *RS* criterion to be minimised is given as:

$$J = E\left\{ \sum_{j=0}^{N} \mathrm{e}_p(t+j+k)^T \mathrm{e}_p(t+j+k) + \lambda_j^2 \left(u_0(t+j)^T u_0(t+j) \right) | t \right\} \tag{5.12}$$

where $E\{\cdot \mid t\}$ is the conditional expectation on the measurements till time t and λ is scalar *weighting-factor* on control action. The *state-space* models producing the signal e_p comprises dynamic weighting *cost-function*.

The *RS* criterion to be minimized has a similar method, but with added terms, for gains optimization. A controller gain variation restriction term is added into the criterion so that large gain variations are costed—besides, the possibility to restrict the gain rate of change in the same manner.

The *cost-function* of the *RS* controller may, consequently, be expressed as:

$$J = E\left\{J_t\right\} = E\left\{E^T_{P_{t+k,N}} E_{P_{t+k,N}} + U^{0T}_{t,N}\Lambda^2_N U^0_{t,N} + \tilde{k}^T_c \Lambda^2_k \tilde{k}_c + \Delta\tilde{k}^T_c \Lambda^2_d \Delta\tilde{k}_c \mid t\right\} \tag{5.13}$$

where the weightings $\Lambda^2_k > 0$ and $\Lambda^2_d \geq 0$, and the incremental gain change:

$$\Delta\tilde{k}_c(t) = \tilde{k}_c(t) - \tilde{k}_c(t-1) = k_c(t) - k_c(t-1) \tag{5.14}$$

As noted above, the fixed and the variation method of executing the controller advantage to distinct solutions as the *cost-index* (5.13) hinges on $\tilde{k}_c(t)$, which is part or either all gains vector optimized.

Criterion Terms: A review of the weighting terms in the criterion is given here. Two weightings involved in the costing of the controller gains are significant in this *RS* solution.

- $\Lambda^2_N = diag\left\{\lambda^2_0, \lambda^2_1, ..., \lambda^2_N\right\}$ are the control inputs u_0 static weightings.
- $\Lambda^2_k = diag\left\{\rho^2_0, \rho^2_1, ..., \rho^2_N\right\}$ are the controller gains changes weightings.
- $\Lambda^2_d = diag\left\{\gamma^2_0, \gamma^2_1, ..., \gamma^2_N\right\}$ are the controller gains deviations increments weightings.

5.3 The RS Controller Solution

The *cost-function* is introduced in the state estimate and state estimation error terms, and so an optimal state estimator is usually essential. For the *qLPV state-estimation* and prediction, the future weighted errors vector can be swapped by terms of orthogonal predicted errors and estimation error; using a *TVKF*, the expression in (5.13) becomes:

$$J=E\left\{(\hat{E}_{P_{t+k,N}}+\tilde{E}_{P_{t+k,N}})^T(\hat{E}_{P_{t+k,N}}+\tilde{E}_{P_{t+k,N}})+U_{t,N}^{0T}\Lambda_N^2U_{t,N}^0+\tilde{k}_c^T\Lambda_k^2\tilde{k}_c+\Delta\tilde{k}_c^T\Lambda_d^2\Delta\tilde{k}_c|t\right\} \quad (5.15)$$

The *cost-index* terms can here be made simpler by reaffirming optimal estimate $\hat{E}_{Pt+k,N}$ and the estimation error $\tilde{E}_{Pt+k,N}$ orthogonality again as:

$$J=\hat{E}_{P_{t+k,N}}^T\hat{E}_{P_{t+k,N}}+U_{t,N}^{0T}\Lambda_N^2U_{t,N}^0+k_c^T\Lambda_K^2k_c+\Delta\tilde{k}_c^T\Lambda_d^2\Delta\tilde{k}_c+J_0 \quad (5.16)$$

where both the term:

$$\tilde{E}_{P_{t+k,N}}=C_{PN}A_Nx(t+k\,|\,t)+C_{PN}D_NW_{t+k,N}$$

The cost term $J_0(t)=E\{\tilde{E}_{P_{t+k,N}}^T\tilde{E}_{P_{t+k,N}}|t\}$ doesn't depend on the control action.

Noting (2.48) the *state-estimates* vector is given as:

$$\hat{E}_{P_{t+k,N}}=\tilde{D}_{P_{t+k,N}}+V_{PN}U_{t,N}^0 \quad (5.17)$$

Noting that $\hat{x}(t+k\,|\,t)$ just depends on the control signal past values. The *multi-step cost-function* (5.16) is developed as:

$$\begin{aligned}J=&\tilde{D}_{P_{t+k,N}}^T\tilde{D}_{P_{t+k,N}}+U_{t,N}^{0T}V_{PN}^T\tilde{D}_{P_{t+k,N}}+\tilde{D}_{P_{t+k,N}}^TV_{PN}U_{t,N}^0+\\&U_{t,N}^{0T}\left(V_{PN}^TV_{PN}+\Lambda_N^2\right)U_{t,N}^0+k_c^T\Lambda_K^2k_c+\Delta\tilde{k}_c^T\Lambda_d^2\Delta\tilde{k}_c+J_0\end{aligned} \quad (5.18)$$

The controller structure is assigned the preferred structure before conduction the optimal control optimization calculation, where $U_{t,N} = U_{fe}(t)k_c(t)$. From (5.10), (5.14), noting $W_{1k} = I$, so that $U^0_{t,N} = U_{t,N}$ and substituting in (5.18) gives:

$$\begin{aligned} J = {} & \tilde{D}^T_{P_{t+k,N}} \tilde{D}_{P_{t+k,N}} + k_c^T(t)U_{fe}^T V_{PN}^T \tilde{D}_{P_{t+k,N}} + \\ & \tilde{D}^T_{P_{t+k,N}} V_{PN} U_{fe} k_c(t) + k_c^T \left(U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) U_{fe} + \Lambda_K^2 \right) k_c(t) - \\ & k_c^T(t)\Lambda_k^2 \bar{k}_c - \bar{k}_c^T \Lambda_k^2 k_c(t) - k_c^T(t-1)\Lambda_d^2 k_c(t) - \\ & k_c^T(t)\Lambda_d^2 k_c(t-1) + k_c^T(t)\left(\Lambda_k^2 + \Lambda_d^2\right)k_c(t) + \bar{k}_c^T \Lambda_k^2 \bar{k}_c + k_c^T(t-1)\Lambda_d^2 k_c(t-1) + J_0 \end{aligned}$$

(5.19)

5.4 Nonlinear RS Optimisation

The *cost-function* (5.19) can be simplified by defining and substituting for expressions (5.20), (5.21) and (5.22).

$$X_N = U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) U_{fe} + \Lambda_K^2 + \Lambda_D^2 \quad (5.20)$$

$$P_{CN} = U_{fe}^T V_{PN}^T \quad (5.21)$$

$$C_\phi = P_{CN} C_{PN} A_N \quad (5.22)$$

Moreover, the following *cost-function* can be obtained:

$$\begin{aligned} J = {} & \tilde{D}^T_{P_{t+k,N}} \tilde{D}_{P_{t+k,N}} + k_c^T(t) P_{CN} \tilde{D}_{P_{t+k,N}} \\ & + \tilde{D}^T_{P_{t+k,N}} P_{CN}^T k_c(t) - \left(\bar{k}_c^T \Lambda_k^2 - k_c^T(t-1)\Lambda_d^2 \right) k_c(t) \\ & - k_c^T(t)\left(\Lambda_k^2 \bar{k}_c + \Lambda_d^2 k_c(t-1) \right) + k_c^T(t) X_N k_c(t) + \bar{J}_0 \end{aligned} \quad (5.23)$$

The *cost-function* (5.23) can be given in concise form as in (5.25) by defining the signal $\varphi(t)$:

$$\varphi(t) = -\Lambda_k^2 \overline{k}_c - \Lambda_d^2 k_c(t-1) \tag{5.24}$$

$$\begin{aligned} J = {} & \tilde{D}_{P_{t+k,N}}^T \tilde{D}_{P_{t+k,N}} + k_c^T(t) P_{CN} \tilde{D}_{P_{t+k,N}} + \tilde{D}_{P_{t+k,N}}^T P_{CN}^T k_c(t) \\ & + \varphi^T(t) k_c(t) + k_c^T(t)\varphi(t) + k_c^T(t) X_N k_c(t) + \overline{J}_0 \end{aligned} \tag{5.25}$$

where $\overline{J}_0 = \overline{k}_c^T \Lambda_k^2 \overline{k}_c + k_c^T(t-1)\Lambda_d^2 k_c(t-1) + J_0$.

This cost term minimising approach is nearly the same as if conditional *cost-function* is employed if the signals are deterministic. The *cost-function* gradient is set to zero to achieve the future optimal control signals vector. From a rather evident perturbation and gradient computation [38], and remarking that the J_0 term doesn't depend on the control signal, and the optimal control gain signals vector is:

$$k_c(t) = -\left(U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2) U_{fe} + \Lambda_K^2 + \Lambda_D^2\right)^{-1} \left(P_{CN} \tilde{D}_{P_{t+k,N}} + \varphi(t)\right) \tag{5.26}$$

Define $\tilde{D}_{P_{t+k,N}}^0 = P_{CN} \tilde{D}_{P_{t+k,N}}$ then (5.26) can be presented as:

$$k_c(t) = -X_N^{-1}\left(\tilde{D}_{P_{t+k,N}}^0 + \varphi(t)\right) \tag{5.27}$$

where $\tilde{D}_{P_{t+k,N}}^0 = P_{CN} D_{P_{t+k,N}} + C_\phi \hat{x}(t+k|t)$

Asymptotic Behaviour: Noting (5.20) and (5.24) if $\Lambda_d^2 \to \infty \times I$ the restricting gain $k_c(t) = k_c(t-1)$ along with the values of the gain turns out to be constant. If $\Lambda_k^2 \to \infty \times I$ the restricting gain $k_c(t) = \overline{k}_c$ and gains turn out to be the same as the *PID* constant primary gain values.

5.5 Hybrid RS Control

The work in [79], [80] utilized *NGMV* to control *PWA* converted into nonlinear *SD* systems through creating related binary functions to demonstrate the switching surfaces crossing conditions. The lead of *SD* systems upon *PWA* systems is that *SD* systems design is a lot simpler, as the switching conditions together with state, input constraints are all contained in the same system model. This section will follow the same approach and use the nonlinear *RS* solution given in Section 5.2 to control *SD* systems by firstly obtaining the *PWA* in the *SD* form, and then the *RS* algorithm can be employed.

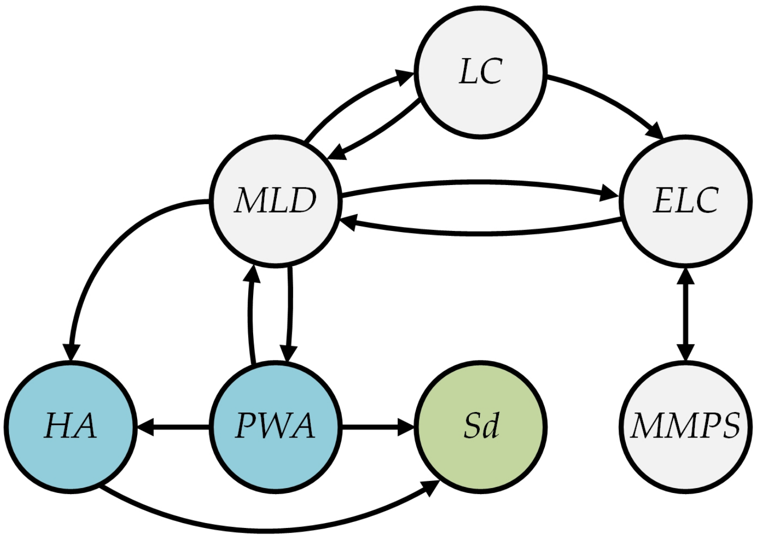

Figure 5-3: Links between the classes of HS

HS comprise together continuous and discrete elements. Most recently, *HS* has obtained consideration from the computer science community and control society. Since there are no exclusive workable analyzing methods for general *HS*, some researchers devoted their work to particular subclasses of *HS* that have established analysis and control design techniques. *LC* systems [81], [82], *MLD* systems [83], *first-order* linear *HS* with saturation [84], *PWA* systems [85], *MMPS* systems [86], and others are subclasses examples. Besides, computer scientists have also suggested several models, including *HA* [87], which are possibly the most dominant model. Equivalence conversion among the subclasses *MMPS*, *ELC*, *LC*, *MLD* and *PWA* systems was given in moderate conditions [88]. Recently, equivalence relations between *PWA* systems and *SD* systems were introduced in [79] and was extended by [89] to show the relation between *SD* models and *LHA*, as illustrated in Figure 5-3.

Each subclass has its advantages over the others. These equivalence relations are important because they allow the relocating of theoretical assets and tools from one class to another. To analyse an *HS* that fits any of these classes, one can select the most useful hybrid modelling structure. Control algorithms formed for *PWA* systems are frequently created utilizing optimal control or *MPC* methods. The former *MPC* algorithm for *HS*, established for the *MLD* system equivalent to the *PWA* system, is given in [83]. However, the algorithm implementation needs high computing capacity, mainly caused by the *NP-hard MIQP* online problem solution needed to be completed at every time instant, which is considered a downside.

5.5.1 Systems Equivalence: PWA and SD

The *discrete-time PWA* systems described in [90] is very common compared to other major *PWA* descriptions since disturbances and *time-delays* are contained easily. The *SD* structure expansion for *PWA* systems in the system model characteristics was inspired because some *HS* control problem solutions are not practical. Also, *SD* systems are simpler to design and identify than other *HS* methods and comprises states, input constraints, and switching conditions.

The *SD* system is important also as parametric model uncertainty arises [91], or in nonlinear system approximation through the *SD* system cases, especially when approximation by *LTI* model is not proper or enough. The benefits of this method are:

- *SD* system requires less logical supervision than other *HS* controllers.
- System time delay and disturbances can be easily modelled and added in the plant model.
- Other nonlinearities or uncertainties modes can be augmented simply in *SD* systems.

5.5.2 PWA Systems

The *state-space* description of the delayed *discrete-time PWA* system is:

$$\begin{aligned} x(t+1) &= A_i x(t) + B_i u(t-k) + D_i \xi(t) + f_i \\ y(t) &= C_i x(t) + E_i u(t-k) + g_i \end{aligned} \tag{5.28}$$

where the state is $x \in \mathbb{R}^n$, the input is $u \in \mathbb{R}^m$, the output is $y \in \mathbb{R}^p$, $d \in \mathbb{R}^n$ is the disturbance, and common delay elements magnitude is k.

Each affine subsystem $\left(A_i, B_i, C_i, D_i, E_i\right),\ i = 1,\ldots,s$ is specified on a polyhedron cell $\Omega_i \subset \mathbb{R}^n \times \mathbb{R}^m$.

Additionally, to make the explanation more accessible, the polyhedral cells sets are specified by matrices G_{ix}, h_{ix}, G_{iu} and h_{iu} as:

$$\Omega_i = \left\{ \begin{bmatrix} x \\ u \end{bmatrix} | \, G_{ix} x \le h_{ix} \wedge G_{iu} u \le h_{iu} \right\} \tag{5.29}$$

The cells in (5.29), satisfy $\Omega_i \cap \Omega_j = \varnothing$, $\forall i \ne j$ and the union describes the states and inputs admissible set $\Omega = \cup_{i=1}^{s} \Omega_i$.

5.5.3 SD Systems

A *SD* system entails a *time-varying* state equation and depends on states, inputs, other varying parameters, or even a command request. The *SD* system can have this structure:

$$\begin{aligned} x_{sd}(t+1) &= A\left(x(t),u(t)\right)x(t)+B\left(x(t),u(t)\right)u(t-k)+D\left(x(t),u(t)\right)\xi(t) \\ y_{sd}(t) &= C\left(x(t),u(t)\right)x(t)+E\left(x(t),u(t)\right)u(t-k) \end{aligned} \tag{5.30}$$

Definition: The *well-posed PWA* model (5.28) is given in the *SD* model (5.30) for the feasible states and inputs polyhedral partition set $\Omega=\cup_{i=1}^{s}\Omega_i$. Its related system parameters $A_i,B_i,C_i,D_i,E_i,\ i=1,\ldots,s$ in (5.28). There is a system combination $A(x,u),B(x,u),C(x,u),D(x,u),E(x,u)$ of (5.30) for all *PWA* system trajectories $x(t),\ u(t),\ y(t)$ in (5.28) to fulfil the *SD* model (5.30).

To prove this and to consider the *PWA* system (5.28), the condition (5.29) is reshaped in auxiliary logic operator $\delta_i(t)\in\{0,1\}$ as:

$$\delta_i(t)=\begin{cases}1 & if\ G_{ix}x(t)\le h_{ix}\wedge G_{iu}u(t)\le h_{iu} \\ 0 & otherwise\end{cases} \tag{5.31}$$

Hence, the *well-posed* system (5.28) with its partitions in (5.29) can be provided in the form given in (5.32).

$$\begin{aligned} x_{sd}(t+1) &= \sum_{i=1}^{s}\delta_i(t)\left[A_ix(t)+B_iu(t-k)+D_i\xi(t)+f_i\right] \\ y_{sd}(t) &= \sum_{i=1}^{s}\delta_i(t)\left[C_ix(t)+E_iu(t-k)+g_i\right] \end{aligned} \tag{5.32}$$

Noting, the logic variable value in (5.31) depends on the state and input signals. Identify the function of less or equal $LE(x,m)$:

$$LE(x,m)=\begin{cases}1 & if\ x\le m\\ 0 & else\end{cases} \tag{5.33}$$

The constant $m\in\mathbb{R}^n$, consequently:

$$\delta_i(t)=\prod_j LE(G_{ix}^j x(t),h_{ix}^j)\prod_l LE(G_{iu}^l u(t),h_{iu}^l) \tag{5.34}$$

where j and l indicate j^{th} and l^{th} rows, correspondingly.

The notation in (5.30) is simplified noting $A_{sd}=A\big(x(t),u(t)\big)$ and likewise for B_{sd}, C_{sd}, D_{sd} and E_{sd} then substituting (5.34) in (5.32), the equivalence system shown in Figure 5-4 is obtained as:

$$\begin{aligned}x_{sd}(t+1)=&\overbrace{\left[\sum_{i=1}^{s}\prod_j LE(G_{ix}^j x(t),h_{ix}^j)\prod_l LE(G_{iu}^l u(t),h_{iu}^l)A_i\right]}^{A_{sd}}x_{sd}(t)\\&+\overbrace{\left[\sum_{i=1}^{s}\prod_j LE(G_{ix}^j x(t),h_{ix}^j)\prod_l LE(G_{iu}^l u(t),h_{iu}^l)B_i\right]}^{B_{sd}}u(t-k)\\&+\overbrace{\left[\sum_{i=1}^{s}\prod_j LE(G_{ix}^j x(t),h_{ix}^j)\prod_l LE(G_{iu}^l u(t),h_{iu}^l)D_i\right]}^{D_{sd}}\xi_{sd}(t)\end{aligned} \tag{5.35}$$

$$\begin{aligned}y_{sd}(t)=&\overbrace{\left[\sum_{i=1}^{s}\prod_j LE(G_{ix}^j x(t),h_{ix}^j)\prod_l LE(G_{iu}^l u(t),h_{iu}^l)C_i\right]}^{C_{sd}}x_{sd}(t)\\&+\overbrace{\left[\sum_{i=1}^{s}\prod_j LE(G_{ix}^j x(t),h_{ix}^j)\prod_l LE(G_{iu}^l u(t),h_{iu}^l)E_i\right]}^{E_{sd}}u(t-k)\end{aligned} \tag{5.36}$$

Consequently, the *PWA* system (5.28) is moved into an *SD* system (5.37) that has the type of (5.30).

$$\begin{aligned}&x_{sd}(t+1)=A_{sd}x_{sd}(t)+B_{sd}u(t-k)+D_{sd}\xi_{sd}(t), \qquad x_{sd}(t)\in R^{n_{sd}}\\&y_{sd}(t)=C_{sd}x_{sd}(t)+E_{sd}u(t-k)\end{aligned} \tag{5.37}$$

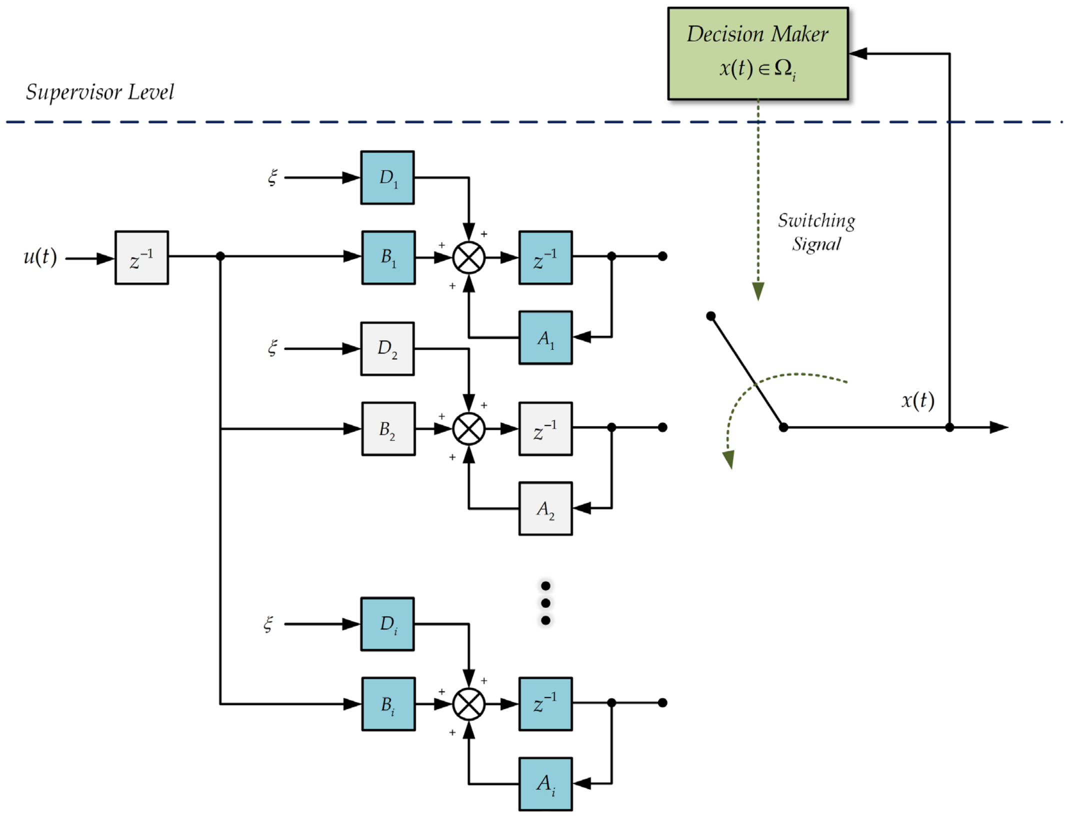

Figure 5-4: Equivalence system structure

Example: An example found in [83] is used to demonstrate the equivalences given above as:

$$x(k+1)=\begin{cases} 0.8x(k)+u(k) & when\ \ x(k)\geq 0 \\ -0.8x(k)+u(k) & when\ \ x(k)<0 \end{cases} \tag{5.38}$$

The system in (5.38) can be converted to the *SD* model:

$$\begin{aligned} x(k+1) &= GE\big(x(k),0\big)\times\big[0.8x(k)+u(k)\big] \\ &\quad + LT\big(x(k),0\big)\times\big[-0.8x(k)+u(k)\big] \end{aligned} \tag{5.39}$$

Moreover, during the optimization, the following constraints must be satisfied:

$$\begin{aligned} -10 \leq x(k) \leq 10 \\ -1 \leq u(k) \leq 1 \end{aligned} \tag{5.40}$$

5.6 Chapter Summary

This Chapter summarises the nonlinear *RS* control solution for the *discrete-time state-space* system description provided in Chapter 2. The Chapter started by giving an introduction to nonlinear *MPC* in Section 5.1. The nonlinear *RS* controller structure was defined in Section 5.2, and details were provided on the vector of gains, the *RS* controller form and the controller problem. The nonlinear controller solution is provided in Section 5.3.

Besides, Section 5.4 is concerned with the evaluation of the nonlinear *RS* optimisation procedures. Finally, the Chapter concluded with an extension for the solution towards a subclass of *HS*. The *PWA* and *SD* equivalence definition is given. This definition can be used to utilize the nonlinear solution for *PWA* systems. In the next Chapter, the *LPV* polynomial solution will be introduced for systems in polynomial form.

Chapter 6 Polynomial LPV RS Control

The polynomial *RS* controller structure that acts within the feedback loop is described in this Chapter for the polynomial system representation provided in Chapter 3. The *RS* controller parameterisation is given, and the *RS* controller for an extended *PID* is discussed in Section 6.2. The explanation of the *RS LPV* predictive control problem is provided in Section 6.3. Finally, this Chapter concludes with a brief description of the design strategy.

6.1 Introduction

A new *RS MBPC* law is described for systems described in a polynomial matrix structure with parameters that may be *time-varying* and known. The use of *LPV* polynomial system descriptions is somewhat unusual, even though the use of transfer function based models for *LTI* systems are common. In the current problem, the plant model is represented by what might be termed an *ARMAX* model. This *ARMAX* model's parameters are expected to be *LPV*, a particularly suitable representation for use with the polynomial based *RS* controllers.

The *RS* controller's idea is that designers choose a controller structure of the usual *low-order* form for commissioning engineers. It seems reasonable to use a discreet transfer function style of the controller to compensate for a plant in *LPV ARMAX* form. This style might involve a classical controller structure with a multivariable *PID* or *lead-lag* form with *time-varying* parameters. There are several reasons for using an *RS* control solution. One of the reasons is the ease of tuning or *re-tuning* by technician engineers on a plant.

With most *model-based* optimal control solutions, it is not straightforward to provide technician engineers with the ability to *re-tune* the *control-loop* without the aid of advanced control knowledge and special computing tools. However, an *RS* controller has familiar tuning knobs or parameters.

Another reason for the use of *RS* control solutions is the fact that *low-order* controllers have natural robustness. When plant model mismatch arises, a *low-order* control solution is much less likely to be destabilised than a *high-order* controller attempting rather complicated gain and phase shifts in critical frequency regions. If a *low-order* controller can provide the performance required, it is more desirable than a *high-order* control solution. In addition to the form of *RS* controller proposed, it is possible to minimise gains, which is also very desirable for many reasons, including power usage.

GPC is used to optimise the *RS* controller gains and is one of the most popular forms of *MPC* law for many industrial applications. The optimal solution is in the best gains terms, which minimise a predictive *cost-function* for selected and decided controller structure, and this is different from the conventional *MPC* approach. For example, *MPC* laws employ a natural *2-DOF* control structure, and even if the *2-DOF* structure is decided, the *RS* results are likely to be *sub-optimal* depending on the choice of structure.

However, the benefit remains in adopting the *low-order* controller since it is a robust controller, simple for implementation, and easy to be *re-tuned* by a somewhat inexperienced engineer.

6.2 RS Controller Parameterising

A 3-*DOF* controller in an *RS* form is to be used; therefore, the controller has individual feedback, feedforward and tracking terms. The use of *RS* controller structures is well established, but the approach used here is different from that in earlier work, where the controllers were understood to be *LTI* [92], [93]. The functions that are used to decide the controller formation are picked before the optimisation is performed. In a simple case, as observed in *PID*, these can be proportional, integrator and derivative functions, or the delay operator terms in a more general controller structure for a *MIMO* system. These results will be valid to a more general *LPV* multivariable controller form, where the feedback controller may be characterized in the form:

$$C_0\left(z^{-1},\rho_t\right)=\frac{C_{0_n}\left(z^{-1},\rho_t\right)}{C_{0_d}\left(z^{-1},\rho_t\right)}$$

6.2.1 Extended PID RS Controller

In the simplest *single-DOF* case, a sum of N_e *frequency-sensitive* linear dynamic functions are picked separately with a distinctive frequency response property, and the control action becomes:

$$u_0(t)=\sum_{j=1}^{N_e} f_j\left(z^{-1},k_j(t)\right)e_0(t) \tag{6.1}$$

The *RS* controller control action is generated by adding the sum of the functions vector. These functions and gains in (6.1) decide the *RS* controller dynamics, thus from (6.1):

$$u_0(t) = f_1\left(z^{-1},\, k_1(t)\right)e_0(t) + f_2\left(z^{-1},\, k_2(t)\right)e_0(t) + \cdots + f_{N_e}\left(z^{-1}, k_{N_e}(t)\right)e_0(t) \quad (6.2)$$

The designer specifies the dynamic functions $f_j(z^{-1}, k_j(t))$. The gains $k_j(t)$ are *non-dynamic time-varying* gain vectors of the *MIMO* system controller set.

6.2.2 General RS Controller Parametrising

A traditional controller for the *LPV* system can be expressed in a similar way using a multivariable *ARMAX* model. That is if a traditional controller is stated in *unit-delay* operator scalar form in (6.3).

$$\begin{aligned} u_0(t) &= C_0\left(z^{-1}, \rho_t\right)e_0(t) \\ &= \frac{C_{0_n} + C_{1_n}z^{-1} + \ldots + C_{2_n}z^{-n_m}}{I + C_{1_d}z^{-1} + C_{2_d}z^{-2} + \ldots + C_{2_d}z^{-n_d}}e_0(t) \end{aligned} \quad (6.3)$$

where the parameter matrices or functions are *LPV*. Then, the gains in the controller denominator (6.3) can likewise be optimised in this problem. After manipulation, the control signal may be rewritten in a form that is more convenient for computations:

$$\begin{aligned} u_0(t) =& \left(C_{0_n} + C_{1_n}z^{-1} + \ldots + C_{2_n}z^{-n_m}\right)e_0(t) - \\ & \left(C_{1_d}z^{-1} + C_{2_d}z^{-2} + \ldots + C_{2_d}z^{-n_d}\right)u_0(t) \end{aligned} \quad (6.4)$$

This form is also suitable for multivariable systems where the coefficients are matrices.

Controller Multivariable Parameterisation: The modified controller structure (6.4) can be introduced in (6.5) by modifying (6.1) to have a more generalisation covering the above controller parameterisation cases.

$$u_0(t) = \sum_{j=1}^{N_e} f_j\left(z^{-1}, k_j(t)\right) c_0(t) \tag{6.5}$$

where the vector $c_0(t) = [e_0^T(t) \quad -u_0^T(t)]^T$.

Moreover, this is suitable for a 3-*DOF* more general controller structure, and the different cases require different definitions for $c_0(t)$ as described below.

Single-*DOF* Extended *PID* Structure: In this simple case, the feedback controller input $c_0(t) = r(t) - y(t)$, where the optimal control is given by (6.1) or (6.2). For a *Single-DOF* case, the controller inputs number of q totals the number of measurement channels $q = r$.

General Traditional Controller Structure: In the further typical case:

$$c_0(t) = \left[e_0^T(t) \quad -u_0^T(t) \right]^T$$

The various *DOF* for the controller varies on the description of the signal e_0. For example, for the more typical controller structure and a *3-DOF* control law issue, as illustrated in Figure 6-1, the signals $e_0(t)$ and $c_0(t)$:

$$\begin{aligned} e_0(t) &= \left[-z^T(t) \quad r^T(t) \quad d_0^T(t) \right]^T \\ c_0(t) &= \left[-z^T(t) \quad r^T(t) \quad d_0^T(t) \quad -u^T(t) \right]^T \end{aligned} \tag{6.6}$$

The reference is not needed in output regulating problems or if there is no measurement of the disturbance, and the feedforward term can be omitted for 2-*DOF* control problems. For the optimisation steps that follow in the next section, it does not matter about the particular case considered since the vector $c_0(t)$ can be created for every error channel in scalar signals terms:

$$c_0(t) = \left[c_{0_1}(t) \quad c_{0_2}(t) \quad \cdots \quad c_{0_q}(t) \right]^T \in R^q \tag{6.7}$$

where q denotes the number of rows in the vector $c_0(t)$

Each term in the summation (6.5) has the functions $f_j(z^{-1},k_j(t))$, and for the multivariable case, it is expected that the function may be expanded as follows:

$$f_j\left(z^{-1},k_j(t)\right)=\begin{bmatrix} f_{11}^j(z^{-1})k_{11}^j(t) & \cdots & f_{1r}^j(z^{-1})k_{1q}^j(t) \\ \vdots & f_{22}^j(z^{-1})k_{22}^j(t) & \vdots \\ & & \ddots \\ f_{m1}^j(z^{-1})k_{m1}^j(t) & \cdots & f_{mr}^j(z^{-1})k_{mq}^j(t) \end{bmatrix} \tag{6.8}$$

$$k_j=\begin{bmatrix} k_{11}^j & k_{12}^j & \cdots & k_{1q}^j \\ k_{21}^j & k_{22}^j & & \vdots \\ \vdots & & \ddots & \\ k_{m1}^j & \cdots & & k_{mq}^j \end{bmatrix} \tag{6.9}$$

***RS* Controller Structure:** The term for the control signal, as a result of the parameterised controller (6.5), turn into:

$$u_0(t)=\begin{bmatrix} \sum\limits_{j=1}^{N_e}\left\{f_{11}^j(z^{-1})k_{11}^jc_{0_1}(t)+f_{12}^j(z^{-1})k_{12}^jc_{0_2}(t)+\ldots+f_{1q}^j(z^{-1})k_{1q}^jc_{0_q}(t)\right\} \\ \sum\limits_{j=1}^{N_e}\left\{f_{21}^j(z^{-1})k_{21}^jc_{0_1}(t)+f_{22}^j(z^{-1})k_{22}^jc_{0_2}(t)+\ldots+f_{2q}^j(z^{-1})k_{2q}^jc_{0_q}(t)\right\} \\ \vdots \\ \sum\limits_{j=1}^{N_e}\left\{f_{m1}^j(z^{-1})k_{m1}^jc_{0_1}(t)+f_{m2}^j(z^{-1})k_{m2}^jc_{0_2}(t)+\ldots+f_{mq}^j(z^{-1})k_{mq}^jc_{0_q}(t)\right\} \end{bmatrix} \tag{6.10}$$

6.2.3 Parameterized Controller

The results in (6.10) offering a reasonable controller parameterisation; however, it can be expressed in a more gains optimisation accessible form. Thus, define the function f_e^{is} and gain vector k_c^{is} to satisfy (6.11).

$$f_e^{is}k_c^{is} = [f_{is}^1(z^{-1})c_{0_s}(t)\ \ f_{is}^2(z^{-1})c_{0_s}(t)\ \ ....\ \ f_{is}^{N_e}(z^{-1})c_{0_s}(t)]\begin{bmatrix} k_{is}^1 \\ k_{is}^2 \\ \vdots \\ k_{is}^{N_e} \end{bmatrix} \tag{6.11}$$

The impact on the channel i control signal relating to the channel s controller input is given in (6.12).

$$\begin{aligned} f_e^{is}k_c^{is} &= f_{is}^1(z^{-1})k_{is}^1 c_{0_s}(t) + f_{is}^2(z^{-1})k_{is}^2 c_{0_s}(t) + \ .... + f_{is}^r(z^{-1})k_{is}^{N_e} c_{0_s}(t) \\ &= \sum_{j=1}^{N_e} \left\{ f_{is}^j(z^{-1})c_{0_s}(t)k_{is}^j \right\} \end{aligned} \tag{6.12}$$

The gain vectors k_c^{ij} are labelled corresponding to controller row number i and controller input number j. The vector size differs on functions number N_e:

$$k_c^{11} = \begin{bmatrix} k_{11}^1 \\ k_{11}^2 \\ \vdots \\ k_{11}^{N_e} \end{bmatrix},\ k_c^{12} = \begin{bmatrix} k_{12}^1 \\ k_{12}^2 \\ \vdots \\ k_{12}^{N_e} \end{bmatrix},\dots\ ,\ k_c^{1q} = \begin{bmatrix} k_{1q}^1 \\ k_{1q}^2 \\ \vdots \\ k_{1q}^{N_e} \end{bmatrix},\dots$$

$$k_c^{21} = \begin{bmatrix} k_{21}^1 \\ k_{21}^2 \\ \vdots \\ k_{21}^{N_e} \end{bmatrix},\ k_c^{22} = \begin{bmatrix} k_{22}^1 \\ k_{22}^2 \\ \vdots \\ k_{22}^{N_e} \end{bmatrix},\dots\ ,k_c^{mq} = \begin{bmatrix} k_{mq}^1 \\ k_{mq}^2 \\ \vdots \\ k_{mq}^{N_e} \end{bmatrix}$$

From (6.10) and (6.12) the parameterised control signal can be expressed as:

$$u_0(t) = \begin{bmatrix} f_e^{11}k_c^{11} + f_e^{12}k_c^{12} + \dots + f_e^{1q}k_c^{1q} \\ f_e^{21}k_c^{21} + f_e^{22}k_c^{22} + \dots + f_e^{2q}k_c^{2q} \\ \vdots \\ f_e^{m1}k_c^{m1} + f_e^{m2}k_c^{m2} + \dots + f_e^{mq}k_c^{mq} \end{bmatrix} \tag{6.13}$$

Total Gain Vector: For the gain calculation algorithm, all gains need to be stacked in one vector. The various channel gains (6.10) may be arranged to agree to the row vectors dimensions in (6.11) to achieve the total gain vector as:

$$k_c(t) = \begin{bmatrix} k_{c_1}^T & k_{c_2}^T & \cdots & k_{c_m}^T \end{bmatrix}^T \tag{6.14}$$

where the $q \times N_e$ vector,

$$k_{c_i}(t) = \begin{bmatrix} k_c^{i1^T} & k_c^{i2^T} & \cdots & k_c^{iq^T} \end{bmatrix}^T$$
$$k_c^{is} = \begin{bmatrix} k_{is}^1 & k_{is}^2 & \cdots & k_{is}^{N_e} \end{bmatrix}^T$$

Moreover, the gains vector is arranged in the controller channel inputs terms. Thus, introduce the *block-diagonal-matrix*:

$$F_e(t) = \begin{bmatrix} f_e^{11} & \cdots & f_e^{1q} & 0 & \cdots & \cdots & & & & \cdots & 0 \\ 0 & \cdots & 0 & f_e^{21} & \cdots & f_e^{2q} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & \cdots & & & & \ddots & & & \cdots & 0 \\ 0 & 0 & 0 & \cdots & & & & \ddots & & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & & \cdots & 0 & f_e^{m1} & \cdots & f_e^{mq} \end{bmatrix} \tag{6.15}$$

The parameterised control signal (6.13) may be given as:

$$u(t) = F_e(t) k_c(t) \tag{6.16}$$

The *block-diagonal-matrix* of the signals (6.15) can be represented in the form:

$$F_e(t) = diag\left\{ e_{f_1}(t) \quad e_{f_2}(t) \quad \cdots \quad e_{f_m}(t) \right\} \tag{6.17}$$

where $e_{f_i}(t) = \begin{bmatrix} f_e^{i1} & f_e^{i2} & \cdots & f_e^{iq} \end{bmatrix}, \quad i \in [1, m]$

Parameterized Control: The control signal to be employed in (6.10), in terms of the total gain vector, and (6.17), becomes:

$$\begin{aligned} u_0(t) = F_e(t)k_c(t) &= \begin{bmatrix} e_{f_1}(t)k_{c1} \\ e_{f_2}(t)k_{c2} \\ \vdots \\ e_{f_m}(t)k_{cm} \end{bmatrix} \\ &= diag\left\{ e_{f_1}(t) \quad e_{f_2}(t) \quad \cdots \quad e_{f_m}(t) \right\} \begin{bmatrix} k_{c1} \\ k_{c2} \\ \vdots \\ k_{cm} \end{bmatrix} \end{aligned} \tag{6.18}$$

Total Gain Vector: The total *gain-vector* k_c to be optimised comprises the controller gains for every channel. The *gain-vector* has $m \times q \times N_e$ rows, which may be written as:

$$\begin{aligned} k_c(t) &= \begin{bmatrix} k_{c_1}^T & k_{c_2}^T & \cdots & k_{c_m}^T \end{bmatrix}^T \\ &= \begin{bmatrix} \underbrace{k_c^{11^T} \; k_c^{12^T} \cdots k_c^{1q^T}}_{channel\, 1\, gains} & \underbrace{k_c^{21^T} \; k_c^{22^T} \cdots k_c^{2q^T}}_{channel\, 2\, gains} & \cdots & \underbrace{k_c^{m1^T} \; k_c^{m2^T} \cdots k_c^{mq^T}}_{channel\, m\, gains} \end{bmatrix}^T \end{aligned} \tag{6.19}$$

6.2.4 RS Controller Gains

It is possible to create the *RS* controller gain based on a fixed or "baseline" gain plus "deviation" terms. For the optimisation process, the gains are divided into a baseline part $\overline{k}_c$ and deviation part $\tilde{k}_c(t)$, which is a *time-varying*.

where

$$k_c(t) = \overline{k}_c + \tilde{k}_c(t) \tag{6.20}$$

The *cost-minimisation* problem to be defined includes the gain deviation $\tilde{k}_c(t)$ and yields two interesting and exceptional cases:

- The case of absolute control gain in which the total controller gains are computed for predictive control criterion minimization by adjusting $\bar{k}_c = 0$, noting $k_c = \tilde{k}_c$.
- The case of deviation in which the gain deviation $\tilde{k}_c$ is calculated for predictive control criterion minimization, noting $\bar{k}_c \neq 0$.

The absolute controller gains are minimised in the above first case, and this is more suitable for new *RS* controller designs. If there is an existing *PID* controller, the active controller gains will identify the baseline gains. The second case applies, and gain deviation around the baseline *PID* is minimised. This case is similar to when two *PID* controllers are connected in parallel, one with fixed gains and the other has *time-varying* gains.

The *RS* controller is given utilizing (6.5) and (6.20):

$$u_0(t) = \sum_{j=1}^{N_e} \left\{ f_j(z^{-1}) \bar{k}_j c_0(t) \right\} + \sum_{j=1}^{N_e} \left\{ f_j(z^{-1}) \tilde{k}_j c_0(t) \right\}$$

In the parametrisation and matrix $F_e(t)$ terms, the *RS* control law may be given as in (6.21).

$$u_0(t) = F_e(t) k_c = F_e(t) \bar{k}_c + F_e(t) \tilde{k}_c(t) \tag{6.21}$$

6.2.5 Future Controls Vector Parameterisation

The *GPC* control signal at instant *t* is formed using *receding-horizon* theory [65]. The future controls vector $U_{t,N}^0$ is processed to minimise *multi-step cost-function* over prediction horizon *N*. Then, the first element $u_0(t)$ in the vector $U_{t,N}^0$ is used at instant *t*, and other future values $u_0(t+1), u_0(t+2), \ldots$ are neglected.

At the following time instant, the optimal control is again restarted for the entire horizon, and just the first value is applied. The *RS* control problem solution involves a related process; however, the optimal control is computed using a *pre-specified* controller structure. A *receding-horizon* type of philosophy can again be invoked. Still, variances of the future tracking and control action are minimised, assuming the controller's structure and gains are fixed over the prediction time interval. It is assumed that the *RS* controller gain vector $k_c(t)$ is fixed through prediction interval [0, *N*], and the optimal gain vector is computed to minimise the criterion.

The current control $u_0(t)$ is then computed from (6.18) using the optimal gains $k_c(t)$. In the same *receding-horizon* control approach and at the following time instant, the procedure is restarted to produce an updated gain vector value and (6.18) is employed again to generate the control.

Future Controls Vector: Using (6.18) and the predicted future values of $F_e(t)$, as in (6.22) then the future controls vector $U^0_{t,N}$ using *receding-horizon* approach are given as:

$$U^0_{t,N} = \begin{bmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+N) \end{bmatrix} = \begin{bmatrix} F_e(t) \\ \hat{F}_e(t+1) \\ \vdots \\ \hat{F}_e(t+N) \end{bmatrix} k_c(t) \tag{6.22}$$

Prediction Control Matrix: The matrix (6.22) may be defined U_{fe} and expressed as in (6.23).

$$U_{fe}(t) = \left[F_e^T(t) \quad \hat{F}_e^T(t+1) \quad \cdots \quad \hat{F}_e^T(t+N) \right]^T \tag{6.23}$$

The i^{th} block row in (6.23) has this matrix in (6.24).

$$\hat{F}_e(t+i) = diag\left\{\hat{e}_{f_1}(t+i) \quad \hat{e}_{f_2}(t+i) \quad \cdots \quad \hat{e}_{f_m}(t+i)\right\} \tag{6.24}$$

Given the error predicted values, then the control action *i-step-ahead* is:

$$\begin{aligned} u_0(t+i) &= \hat{F}_e(t+i)k_c(t) \\ &= diag\left\{\hat{e}_{f_1}(t+i) \quad \hat{e}_{f_2}(t+i) \quad \cdots \quad \hat{e}_{f_m}(t+i)\right\} k_c(t) \end{aligned}$$

Parameterised Predictive Control: The future controls vector is then given in the parameterised predictive controller terms as:

$$U^0_{t,N} = U_{fe}(t)k_c(t) \tag{6.25}$$

where

$$U_{fe}(t) = \begin{bmatrix} diag\left\{e_{f_1}(t) \quad e_{f_2}(t) \quad \cdots \quad e_{f_m}(t)\right\} \\ diag\left\{\hat{e}_{f_1}(t+1) \quad \hat{e}_{f_2}(t+1) \quad \cdots \quad \hat{e}_{f_m}(t+1)\right\} \\ \vdots \\ diag\left\{\hat{e}_{f_1}(t+N) \quad \hat{e}_{f_2}(t+N) \quad \cdots \quad \hat{e}_{f_m}(t+N)\right\} \end{bmatrix}$$

and

$$\hat{e}_{f_i}(t) = \begin{bmatrix} \hat{f}_e^{i1} & \hat{f}_e^{i2} & \cdots & \hat{f}_e^{iq} \end{bmatrix}$$

The matrix U_{fe} in (6.22) is of $(N+1)\times m$ rows by $q\times m\times N_e$ columns. Usually, in practice, the number of prediction horizon steps is larger than the number of functions N_e. Therefore, there will typically be more rows than columns in the matrix U_{fe}.

6.3 Polynomial RS Control

The extended criterion to be minimised for the *RS-GPC* controller polynomial form will be derived in the next. It is founded and stimulated by the traditional *GPC* performance index in (6.26).

$$J = E\left\{\sum_{j=0}^{N}\left[e^T(t+j+k)e(t+j+k)+\lambda_j^2\left(u_0^T(t+j)u_0(t+j)\right) \mid t\right]\right\} \tag{6.26}$$

where $E\{\cdot \mid t\}$ represents the conditional expectation on measurements until the instant *t* and λ_j implied a scalar control signal weighting. The first extension is to penalise the dynamically weighted tracking error rather than merely tracking error. The models generating the weighted reference and output signals and may comprise dynamic *cost-function* weighting $P_c(z^{-1},\rho_t)$. Then the future *weighted* error signal vector:

$$e_p\left(t+j+k\right)=r_p\left(t+j+k\right)-y_p\left(t+j+k\right)=P_c\left(z^{-1},\rho\right)\left[r\left(t+j+k\right)-y\left(t+j+k\right)\right]$$

The *GPC* performance index involves a *multi-step cost-function* to determine the future optimal predictive control signal for the interval $\tau \in [t,\ t+N]$. It can be created, in a compact vector shape, using vectors in (3.36) as:

$$J = E\{J_t\} = E\left\{E_{P_{t+k,N}}^T E_{P_{t+k,N}} + U_{t,N}^{0^T}\Lambda_N^2 U_{t,N}^0 \mid t\right\} \tag{6.27}$$

The *RS-GPC cost-function*, to be introduced below, can be specified to have an analogous form but with a few improvements. Also, it is sensible to restrict the deviation in controller gains $\tilde{k}_c$ by adding terms into the *cost-index* to penalise large gain deviations, and the gains rate of change where:

$$\Delta\tilde{k}_c(t)=\tilde{k}_c(t)-\tilde{k}_c(t-1)=k_c(t)-k_c(t-1) \tag{6.28}$$

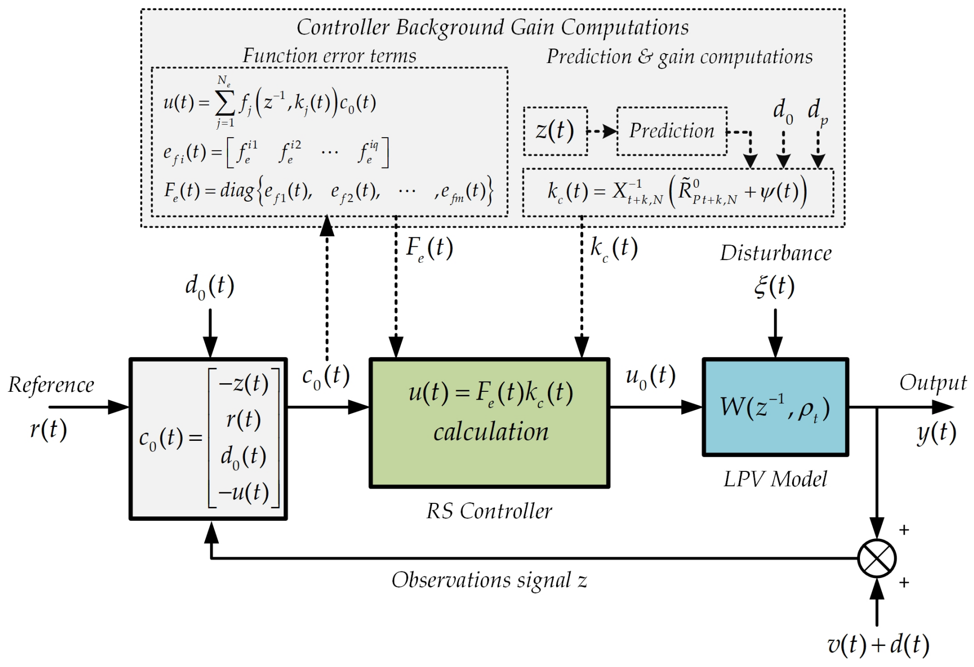
Controller Background Gain Computations
Function error terms
$u(t)=\sum_{j=1}^{N_e} f_j\left(z^{-1},k_j(t)\right)c_0(t)$
$e_{fi}(t)=\left[f_e^{i1} \quad f_e^{i2} \quad \cdots \quad f_e^{iq}\right]$
$F_e(t)=diag\left\{e_{f1}(t), \quad e_{f2}(t), \quad \cdots \quad ,e_{fm}(t)\right\}$
Prediction & gain computations
$z(t)$
Prediction
d_0 d_p
$k_c(t)=X_{t+k,N}^{-1}\left(\tilde{R}_{Pt+k,N}^0+\psi(t)\right)$
$F_e(t)$
$k_c(t)$
Disturbance
$\xi(t)$
$d_0(t)$
Reference
$r(t)$
$c_0(t)=\begin{bmatrix}-z(t)\\ r(t)\\ d_0(t)\\ -u(t)\end{bmatrix}$
$c_0(t)$
$u(t)=F_e(t)k_c(t)$ calculation
RS Controller
$u_0(t)$
$W(z^{-1},\rho_t)$
LPV Model
Output
$y(t)$
+
+
Observations signal z
$v(t)+d(t)$

Figure 6-1: 3-DOF controller structure for polynomial LPV model

RS-GPC Cost-Function: The *RS-GPC multi-step* J=$E\{J_t\}$ *cost-function vector-form* is given as:

$$J=E\left\{E_{P_{t+k,N}}^T E_{P_{t+k,N}}+U_{t,N}^{0^T}\Lambda_N^2 U_{t,N}^0+\tilde{k}_c^T(t)\Lambda_K^2\tilde{k}_c(t)+\Delta\tilde{k}_c^T(t)\Lambda_D^2\Delta\tilde{k}_c(t)\,\middle|\,t\right\} \tag{6.29}$$

Significance Cost-Weightings: The weightings of error, control, gain, and *gain-increment* are:

- $E_{P_{t+k,N}}$ are the dynamically weighted future tracking error terms.
- $\Lambda_N^2=diag\{\lambda_0^2,\lambda_1^2,...,\lambda_{N_e}^2\}$ are the future inputs u_0 *cost-weightings*.
- $\Lambda_K^2=diag\{\rho_0^2,\rho_1^2,...,\rho_{N_e}^2\}$ are the gains deviation *cost-weightings*.
- $\Lambda_D^2=diag\{\gamma_0^2,\gamma_1^2,...,\gamma_{N_e}^2\}$ are the gains increment *cost-weightings*.

where the length of the prediction horizon is *N* and N_e is utilized *RS* controller functions number.

6.3.1 RS Solution Using Output-Estimation

The *cost-function* is formed in the optimal signal estimates and estimation errors terms. The future outputs vector is decomposed in terms of predicted outputs vector and output estimation error vectors since the future predicted outputs vector $\hat{Y}_{P_{t+k,N}}$ and prediction errors $\tilde{Y}_{P_{t+k,N}}$ are orthogonal. Thus, future tracking error vector terms becomes:

$$E_{P_{t+k,N}} = R_{P_{t+k,N}} - Y_{P_{t+k,N}} = R_{P_{t+k,N}} - \hat{Y}_{P_{t+k,N}} - \tilde{Y}_{P_{t+k,N}} \tag{6.30}$$

The future reference vector values are expected to be a known signal and hence define:

$$\tilde{R}_{P_{t+k,N}} = R_{P_{t+k,N}} - F_{P_{t+k,N}} \tag{6.31}$$

Noting (3.33), then:

$$R_{P_{t+k,N}} - \hat{Y}_{P_{t+k,N}} = R_{P_{t+k,N}} - F_{P_{t+k,N}} - G_{P_{t+k,N}} U^0_{t,N} = \tilde{R}_{P_{t+k,N}} - G_{P_{t+k,N}} U^0_{t,N}$$

Moreover, the future tracking error follows as:

$$E_{P_{t+k,N}} = \hat{E}_{P_{t+k,N}} + \tilde{E}_{P_{t+k,N}} = \left(\tilde{R}_{P_{t+k,N}} - G_{P_{t+k,N}} U^0_{t,N}\right) - \tilde{Y}_{P_{t+k,N}}$$

where the predicted future tracking error follows as:

$$\hat{E}_{P_{t+k,N}} = \tilde{R}_{P_{t+k,N}} - \hat{Y}_{P_{t+k,N}} = \tilde{R}_{P_{t+k,N}} - G_{P_{t+k,N}} U^0_{t,N} \tag{6.32}$$

and the tracking estimation error:

$$\tilde{E}_{P_{t+k,N}} = -\tilde{Y}_{P_{t+k,N}} \tag{6.33}$$

These signals (6.32) and (6.33) are orthogonal because of the use of an optimal estimator.

Cost Simplification: Substituting for $E_{P_{t+k,N}}$ in (6.29):

$$J=E\left\{(\hat{E}_{P_{t+k,N}}+\tilde{E}_{P_{t+k,N}})^T(\hat{E}_{P_{t+k,N}}+\tilde{E}_{P_{t+k,N}})+U_{t,N}^{0^T}\Lambda_N^2U_{t,N}^0 \right.$$
$$\left. +\tilde{k}_c^T(t)\Lambda_K^2\tilde{k}_c(t)+\Delta\tilde{k}_c^T(t)\Lambda_D^2\Delta\tilde{k}_c(t)\middle|t\right\} \quad (6.34)$$

The *cost-index* terms can be made simpler utilizing optimal estimate $\hat{E}_{P_{t+k,N}}$ and estimation error $\tilde{E}_{P_{t+k,N}}$ orthogonality and noting (6.28):

$$J=\hat{E}_{P_{t+k,N}}^T\hat{E}_{P_{t+k,N}}+U_{t,N}^{0^T}\Lambda_N^2U_{t,N}^0+\tilde{k}_c^T(t)\Lambda_K^2\tilde{k}_c(t)+\Delta k_c^T(t)\Lambda_D^2\Delta k_c(t)+J_0(t) \quad (6.35)$$

where the last *cost-term*:

$$J_0(t)=E\left\{\tilde{E}_{P_{t+k,N}}^T\tilde{E}_{P_{t+k,N}}\mid t\right\}=E\left\{\tilde{Y}_{P_{t+k,N}}^T\tilde{Y}_{P_{t+k,N}}\mid t\right\}$$

J_0 doesn't depend on the choice of the control signal. Swap for (3.33) in (6.35):

$$J=\left(\tilde{R}_{P_{t+k,N}}-G_{P_{t+k,N}}U_{t,N}^0\right)^T\left(\tilde{R}_{P_{t+k,N}}-G_{P_{t+k,N}}U_{t,N}^0\right)$$
$$+U_{t,N}^{0^T}\Lambda_N^2U_{t,N}^0+\tilde{k}_c^T(t)\Lambda_K^2\tilde{k}_c(t)+\Delta k_c^T(t)\Lambda_D^2\Delta k_c(t)+J_0(t)$$

$$J=\tilde{R}_{P_{t+k,N}}^T\tilde{R}_{P_{t+k,N}}-U_{t,N}^{0^T}G_{P_{t+k,N}}^T\tilde{R}_{P_{t+k,N}}-\tilde{R}_{P_{t+k,N}}^TG_{P_{t+k,N}}U_{t,N}^0$$
$$+U_{t,N}^{0^T}G_{P_{t+k,N}}^TG_{P_{t+k,N}}U_{t,N}^0+U_{t,N}^{0T}\Lambda_N^2U_{t,N}^0+\tilde{k}_c^T(t)\Lambda_K^2\tilde{k}_c(t) \quad (6.36)$$
$$+\Delta k_c^T(t)\Lambda_D^2\Delta k_c(t)+J_0(t)$$

Also noting (6.28),

$$\Delta k_c^T(t)\Lambda_D^2\Delta k_c(t)=k_c^T(t)\Lambda_D^2k_c(t)-k_c^T(t-1)\Lambda_D^2k_c(t)$$
$$-k_c^T(t)\Lambda_D^2k_c(t-1)+k_c^T(t-1)\Lambda_D^2k_c(t-1)$$

RS Parameterisation: The controller is parameterised in (6.25) and note (6.20) and the next term:

$$\tilde{k}_c^T(t)\Lambda_K^2\tilde{k}_c(t)=\left(k_c^T(t)-\overline{k}_c^T\right)\Lambda_K^2\left(k_c(t)-\overline{k}_c\right)$$
$$=k_c^T(t)\Lambda_K^2k_c(t)-\overline{k}_c^T\Lambda_K^2k_c(t)-k_c^T(t)\Lambda_K^2\overline{k}_c+\overline{k}_c^T\Lambda_K^2\overline{k}_c$$

To simplify the expressions, also define $\tilde{R}^0_{P_{t+k,N}}$ as:

$$\tilde{R}^0_{P_{t+k,N}} = U^T_{fe}(t) G^T_{P_{t+k,N}} \tilde{R}_{P_{t+k,N}} = U^T_{fe}(t) G^T_{P_{t+k,N}} \left(R_{P_{t+k,N}} - F_{P_{t+k,N}} \right) \tag{6.37}$$

so that

$$U^{0^T}_{t,N} G^T_{P_{t+k,N}} \tilde{R}_{P_{t+k,N}} = k^T_c(t) U^T_{fe}(t) G^T_{P_{t+k,N}} \tilde{R}_{P_{t+k,N}} = k^T_c(t) \tilde{R}^0_{P_{t+k,N}}$$

Also, define the full rank *LPV* matrix:

$$X_{t+k,N} = U^T_{fe}(t) G^T_{P_{t+k,N}} G_{P_{t+k,N}} U_{fe}(t) + U^T_{fe}(t) \Lambda^2_N U_{fe}(t) + \Lambda^2_K + \Lambda^2_D \tag{6.38}$$

The cost statement vector/matrix shape is acquired by substituting in (6.36) as:

$$\begin{aligned} J = {} & \tilde{R}^T_{P_{t+k,N}} \tilde{R}_{P_{t+k,N}} - k^T_c(t) \left(\tilde{R}^0_{P_{t+k,N}} + \Lambda^2_K \bar{k}_c \right) - \left(\tilde{R}^{0^T}_{P_{t+k,N}} + \bar{k}^T_c \Lambda^2_K \right) k_c(t) \\ & + k^T_c(t) \left(U^T_{fe}(t) G^T_{P_{t+k,N}} G_{P_{t+k,N}} U_{fe}(t) + U^T_{fe}(t) \Lambda^2_N U_{fe}(t) + \Lambda^2_K + \Lambda^2_D \right) k_c(t) \\ & - k^T_c(t-1) \Lambda^2_D k_c(t) - k^T_c(t) \Lambda^2_D k_c(t-1) + \bar{k}^T_c \Lambda^2_K \bar{k}_c + k^T_c(t-1) \Lambda^2_D k_c(t-1) + J_0(t) \end{aligned}$$

Also, define:

$$\begin{aligned} \psi(t) &= \Lambda^2_K \bar{k}_c + \Lambda^2_D k_c(t-1) \\ \bar{J}_0 &= \bar{k}^T_c \Lambda^2_K \bar{k}_c + k^T_c(t-1) \Lambda^2_D k_c(t-1) + J_0 \end{aligned} \tag{6.39}$$

Substitute for (6.39), and the cost expression vector/matrix form is achieved as:

$$\begin{aligned} J = {} & \tilde{R}^T_{P_{t+k,N}} \tilde{R}_{P_{t+k,N}} - k^T_c(t) \left(\tilde{R}^0_{P_{t+k,N}} + \psi(t) \right) \\ & - \left(\tilde{R}^{0^T}_{P_{t+k,N}} + \psi^T(t) \right) k_c(t) + k^T_c(t) X_{t+k,N} k_c(t) + \bar{J}_0(t) \end{aligned} \tag{6.40}$$

where the $\bar{J}_0$ term doesn't depend on the control action

Optimisation: The *cost-function* minimising process for this term is analogous to the case as if the signals are deterministic. Namely, the *cost-function* gradient is set to zero to find future optimal controls vector. From a perturbation and gradient calculation [38], or by adjusting the gradient to zero provides the *RS-GPC optimal control gains* vector:

$$k_c(t) = X_{t+k,N}^{-1}\tilde{R}_{P_{t+k,N}}^{0} \tag{6.41}$$

where

$$\tilde{R}_{P_{t+k,N}}^{0} = U_{fe}^{T}(t)G_{P_{t+k,N}}^{T}\left(R_{P_{t+k,N}} - F_{P_{t+k,N}}\right) \tag{6.42}$$

$$X_{t+k,N} = U_{fe}^{T}(t)G_{P_{t+k,N}}^{T}G_{P_{t+k,N}}U_{fe}(t) + U_{fe}^{T}(t)\Lambda_N^2 U_{fe}(t) + \Lambda_K^2 \tag{6.43}$$

Asymptotic Behaviour: Notice in (6.41) that if $\Lambda_D^2 \to \infty \times I$, the restricting gain $k_c(t) = k_c(t-1)$ and the gains turn out fixed as required. Also, if $\Lambda_K^2 \to \infty \times I$, then the restricting gain $k_c = \overline{k}_c$ and the gains turn out equal to baseline gain values.

Minimum-Cost: Recalling (6.37) and substituting for the gains (6.41) in (6.40):

$$J_{\min} = \tilde{R}_{P_{t+k,N}}^{T}\tilde{R}_{P_{t+k,N}} - \left(\tilde{R}_{P_{t+k,N}}^{0^T} + \psi^T(t)\right)X_{t+k,N}^{-1}\left(\tilde{R}_{P_{t+k,N}}^{0} + \psi(t)\right) + \overline{J}_0(t) \tag{6.44}$$

6.3.2 RS Solution Summary

The *RS-GPC* solutions above are summarised below:

Theorem IV: *RS-GPC* of *LPV* Polynomial System

Consider the polynomial *LPV* system as introduced in Chapter 3.

The *RS-GPC* illustrated in Figure 6-1 is essential to minimise the *cost-index*:

$$J = E\left\{ E^T_{P_{t+k,N}} E_{P_{t+k,N}} + U^{0^T}_{t,N} \Lambda^2_N U^0_{t,N} + \tilde{k}^T_c(t)\Lambda^2_K \tilde{k}_c(t) + \Delta\tilde{k}^T_c(t)\Lambda^2_D \Delta\tilde{k}_c(t) \middle| t \right\} \tag{6.45}$$

The *RS-GPC* controller is executed as:

$$u_0(t) = \sum_{j=1}^{N_e} f_j\left(z^{-1}, k_j(t)\right) c_0(t) = F_e(t) k_c(t) \tag{6.46}$$

where $f_j(z^{-1}, k_j(t)),\ j \in [1,\ N_e]$ are designer *pre-specified* functions to define the structure of the *RS* controller, and $F_e(t)$ has a *block-diagonal-matrix* shape:

$$F_e(t) = diag\left\{ e_{f_1}(t) \quad e_{f_2}(t) \quad \cdots \quad e_{f_m}(t) \right\}$$

and for every $i = \{1, 2, \ldots, m\}$, the row vector is:

$$e_{f_i}(t) = \left[f^{i1}_e \quad f^{i2}_e \quad \cdots \quad f^{iq}_e \right]$$

The gains of the optimal feedback controller are selected to minimise (6.45).

The optimal *time-varying* gains of *RS-GPC* satisfy (6.47) by performing a type of *receding-horizon* approach.

$$k_c(t) = X^{-1}_{t+k,N}\left(\tilde{R}^0_{P_{t+k,N}} + \psi(t) \right) \tag{6.47}$$

The future optimal controls vector may be acquired as:

$$U_{t,N}^{0} = U_{fe}(t)k_c(t)$$

where

$$U_{fe}(t) = \left[F_e^T(t) \quad \hat{F}_e^T(t+1) \quad \cdots \quad \hat{F}_e^T(t+N) \right]^T$$

Solution: The proof develops by gathering the results of the previous sections.

6.3.3 Design Strategy

The most valuable benefit of the *RS* control solution is that the within the loop controller is *low-order* and is consequently easy to compute. The background processing to undertake the optimisation does not need to be at the same sample rate, and hence there are numerical efficiencies to be obtained. There is also the possibility of implementing the *RS* solution using scheduling to retrieve the gains from a *semi-automatic* process in specific applications.

A further advantage of the problem's mathematical construction is that the chosen controller in a *low-order* structure can be related to an existing classical solution implemented in a system. Say a *PID* controller already stabilises a system, the fixed set of *PID* gains can be treated as a baseline and modified using a deviation term to optimise the system. The deviation in these gains can be penalised in the predictive control *cost-function* rather than the absolute values. If the penalty on the gain deviation is large, the optimal control will revert to an existing classical solution. One of this approach's gains is that the default level of gains will ensure integral action is included in the usual way, and any state errors cannot persist whether there is a modelling error or not in the predictions. This problem arises in conventional *MPC* that has many possible solutions that are not straightforward for some problems. Therefore, the use of discrete operator *ARMAX* based plant models is particularly appropriate when the *RS* controller is chosen to have a classical form, such as a multivariable *PID* or a discrete *transfer-function* form. The opportunity to include *parameter-variations* in a structured way can be more suitable than merely using scheduled linear models.

***RS* Controller Properties:** *RS* optimal and predictive control techniques enable the structure of the controller within the loop to be fixed and contain easily adjustable tuning variables while at the same time optimising a *cost-function* based upon a model. In practice, the optimal gains remain approximately constant if disturbances and references changes are small, but they change with time when these signals are significant. For instance, if a *PID* formation is chosen, the *PID* gains vary whenever such changes arise since the background processing involves the predictive control law *cost-optimisation*. This approach offers the advantages of *MBPC*, and at the same time, classically trained engineers can retune the system using the parameters selected for the controller structure. Such changes will still result in optimal control because of predictive *cost-function* optimisation.

Robustness: *Low-order* controller is less sensitive to modelling uncertainties. For example, $H\infty$ controllers have an order equal to the total plant order plus any weightings or disturbance models. Therefore, they are generally of a *high-order* and attempt to shape the frequency responses, particularly around the unity-gain or bandwidth point, which gives excellent sensitivity functions unless the plant model is mismatched.

An equivalent *low-order* controller that might provide similar performance does not attempt the modest gain and phase changes of the $H\infty$ design and is, therefore, less sensitive to any plant model mismatch.

Implementation Benefits: *RS* controllers can be employed in a numerically efficient manner. For example, it is unnecessary to have the background processing run at the same feedback loop sample rate. The computations for the controller within the loop are like those for any classical structure. If the background processing is at a lower sample rate, considerable computational power savings may be available. It also provides a natural way to develop a scheduled control law where the gains are computed automatically by merely recording the *RS* controller gains at significant operating points and then using them within a scheduled solution.

Simple Retuning: Most advanced controllers that provide high performance are based on system models, and technicians or calibration engineers cannot quickly retune such controllers. Industries like automotive would like *model-based* control benefits, but they also need the simplicity of tuning that simple controller structures provide. For example, automating the engine calibration process allows the optimisation algorithm to compute the gains over a driving cycle, which may be applied as the basis for tuning by calibration engineers.

6.4 Chapter Summary

This Chapter summarises the nonlinear *RS* control solution for the *discrete-time* polynomial system description provided in Chapter 3. The Chapter started by giving an introduction in Section 6.1. The controller parameterising and the controller problem forming are given in Section 6.2, along with an extended *PID RS* controller is discussion. The solution of the polynomial *RS* controller is given in Section 6.3. Finally, the Chapter concluded with a strategy for the design in Section 6.3.3. In the next Chapter, four examples are used for simulation to show the *RS* controllers performance.

Chapter 7 Applications

In this Chapter, an introduction is given for dynamic weighting selection and stability analysis in Section 7.1. In Section 7.2, the *RS* controller formed in both *PI* and *PID* structure for the *QTP* regulation and tracking problems. In Section 7.3, a *qLPV* model was used to catch the *ETB* nonlinearities, and a nonlinear *RS* controller was used to form a *PI* structure. Similarly, in Section 7.4, the *PWA* model of *CSTR* was converted to an *SD* model, and a nonlinear *RS* controller was used to form a *PI* structure controller. Section 7.5 was used to provide proof for the *LPV* polynomial *RS* controller introduced in Chapter 6 and performance analysed on an automotive example of *VCT*.

7.1 Introduction

The output of the feedback system depicted in Figure 7-1 is represented as:

$$Y(z^{-1}) = y_d(z^{-1}) + \left[W_0(z^{-1})\right]U_0(z^{-1}) \tag{7.1}$$

The design aims to obtain a control system such that $Y(z^{-1})$ follow $R(z^{-1})$ with limited use of $U_0(z^{-1})$, and in this case, the design is termed as reference tracking. If the reference or a setpoint signal is zero, the control plan becomes a regulating problem. During operating conditions, the plant is disturbed by the system output disturbance, in this case $y_d(z^{-1})$, and other unwanted elements such as measurement noise, modelling errors or variations in the supply input of the system.

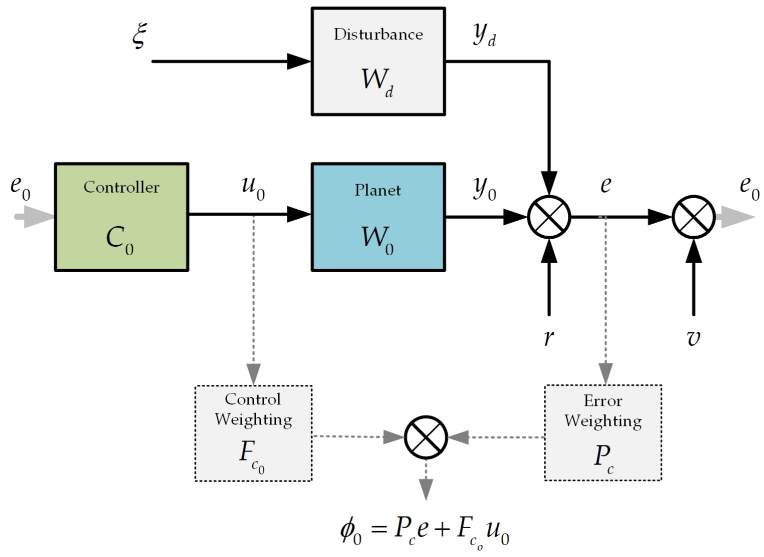

Figure 7-1: Feedback system with disturbances and measurement noise

Therefore, a decent control design is also responsible for ensuring disturbance rejection, measurement noise attenuation, besides good stability robustness. The study of control loop transfers functions is beneficial to understand the performance goals of the control design, and three sensitivity functions are essential [94]:

$$\begin{array}{lll} \textit{Sensitivity} & : & S = \dfrac{1}{1+W_0C_0} \\ \begin{array}{l}\textit{Control}\\ \textit{Sensitivity}\end{array} & : & M = C_0S = \dfrac{C_0}{1+W_0C_0} \\ \begin{array}{l}\textit{Complementary}\\ \textit{Sensitivity}\end{array} & : & T = 1-S = \dfrac{W_0C_0}{1+W_0C_0} \end{array} \tag{7.2}$$

These sensitivity functions can be utilized in the dynamic weighting selection procedure, which transfers the desired performance obligations to the optimization algorithm.

7.1.1 Dynamic Weighting Selection

The dynamic weighting incorporation in the optimization algorithm aims to achieve good stability robustness and measurement noise rejection by ensuring the system has sufficient gain at *low-frequency* and small gain at *high-frequency*. The process of *RS* controller tuning is expected to be simpler than classical controller tuning. In a simple case, the control signal weighting F_{c_0} offers an effortless approach to alter the system response speed. The error signal weighting P_c can be static or an integrator weighting if an integral action is needed in the controller design. For example, if a *PID* controller or even any other recognized classical controller does exist for a system to be controlled. In that case, the *cost-function* selection procedures can be decided by making the error weighting divided by the control weighting proportion equivalent to the existing controller [95] as starting choice. Before this result becomes acceptable, some propositions must be introduced, although it is a design launching position. Equally, having these weightings enclosed, this approach results in natural frequency response properties. Say a *PID* controller with *low-frequency* high gain and a possibility of having a small *high-frequency* gain if a filter was included. A detailed guide is given in [95] on how choosing both the control and error weightings and can be summarised as follow:

7.1.1.1 Error Weighting

A common requirement is that the controller should comprise integral action, which can be accomplished by merging the integrator action into the error weighting as:

$$P_c(z^{-1}) = \frac{P_{c_n}}{1 - z^{-1}} \tag{7.3}$$

The numerator in (7.3) can be set to a constant or selected as $(1-\alpha z^{-1})$ to have a tuning parameter $0<\alpha<1$ that determines the integral action *cut-off* point. A high gain for *low-frequency* within the feedback loop is introduced by utilizing an integral error weighting. In linear systems case, the controller poles had the error weighting poles. As the system's sensitivity function reduces when loop gains increase, the *low-frequency* disturbances will be rejected asymptotically.

7.1.1.2 Control Weighting

In a simple case, the control weighting role can be decided as a linear lead term to guarantee the controller *roll-off* at *high-frequency* and avoid the measurement noise effect augmentation. This is different from dealing with measurement noise block explicitly by including it in the system model to attenuate measurement noise. Still, if a dynamic *cost-weighting* of a control signal is utilized, there is extra control over the *roll-off*.

7.1.1.3 SD Dynamic Weighting

The *state-space* error dynamic weighting is given in Appendix: A-1 Augmented System Matrices (A.22) and (A.23):

$$\begin{aligned} x_p(t+1) &= A_p x_p(t) + B_p\left(r(t)-y(t)\right) \\ e_p(t) &= C_p x_p(t) + E_p\left(r(t)-y(t)\right) \end{aligned} \tag{7.4}$$

This weighting can be selected to be *SD* or *LPV* dynamic weighting as:

$$\begin{aligned} x_p(t+1) &= A_p(\rho) x_p(t) + B_p(\rho)\left(r(t)-y(t)\right) \\ e_p(t) &= C_p(\rho) x_p(t) + E_p(\rho)\left(r(t)-y(t)\right) \end{aligned} \tag{7.5}$$

This way of enclosing an *SD* dynamic weighting is more fitting for systems that need different treatment for different system dynamics, such as the *CSTR* process when the system switch from stable to unstable mode.

7.1.2 Design Issues

For this analysis, the next discussion assumes that the external stochastic inputs are null, and known reference and feedforward measured disturbance signals are the only inputs.

Then from (2.5), the observations:

$$\begin{aligned} z(t) &= d(t) + Cx(t) + Eu_0(t-k) + v(t) \\ &= d(t) + (E + C\Phi B)u_0(t-k) \end{aligned}$$

Thence, the expression for the optimal control gains (4.98):

$$\begin{aligned} k_c = -X_F^{-1}\Big(& P_{CN} D_{P_{t+k,N}} + C_\phi A^k T_{f_1} W_{0k} u_0(t-k) + C_\phi A^k T_{f_2}\, u_0(t) \\ &+ \left(C_\phi T_0(k,z^{-1}) BC_{I0} + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2)\right) W_{1k,N} U_{fe} k_c \Big) \end{aligned}$$

This expression corresponds to the following optimality condition:

$$\begin{aligned} X_F k_c + \Big(& P_{CN} D_{P_{t+k,N}} + C_\phi A^k T_{f_1} W_{0k} z^{-k} u_0(t) + C_\phi A^k T_{f_2}\, u_0(t) \\ &+ \left(C_\phi T_0(k,z^{-1}) BC_{I0} + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2)\right) W_{1k,N} U_{fe} k_c \Big) = 0 \end{aligned}$$

Recall (2.26), (2.30) and noting (2.7), then:

$$C_\phi T_0(k,z^{-1}) B + C_\phi A^k T_{f_2} + C_\phi A^k T_{f_1}\, z^{-k} W_{0k} = C_\phi \Phi B$$

Thence, the condition for optimality:

$$X_F^{-1} P_{CN} D_{P\,t+k,N} + \left(I + X_F^{-1}\left(C_\phi \Phi B C_{I0} + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2)\right) W_{1k,N} U_{fe}\right) k_c = 0$$

Moreover, the optimal control may be given as:

$$k_c = -\left(X_F + \left(C_\phi \Phi B C_{I0} + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2)\right) W_{1k,N} U_{fe}\right)^{-1} P_{CN} D_{P_{t+k,N}}$$

$$U_{t,N} = -U_{fe}\left(I + X_F^{-1}\left(C_\phi \Phi B C_{I0} + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2)\right) W_{1k,N} U_{fe}\right)^{-1} X_F^{-1} P_{CN} D_{P_{t+k,N}}$$

where $(W_{1k,N} U_{t,N}) = [(W_{1k}u)(t)^T, ..., (W_{1k}u)(t+N)^T]^T$.

For the *RS* predictive control, with the given and assumed plant structure, the below expression needs to have a stable inverse:

$$(I + X_F^{-1}(C_\phi \Phi B C_{I0} + U_{fe}^T (V_{PN}^T V_{PN} + \Lambda_N^2)) W_{1k,N} U_{fe})$$

where stability measure, for instance, finite gain stability, attach to subsystem W_{1k} stability assumption

7.2 Quadruple Tank Process

The industry widespread *PID* controller is employed very successfully. But, if the system dynamics are somewhat complex, then the *PID* may not provide the required performance. Thus, the adoption of a *higher-order* optimal controller is necessary to handle the dynamics of the *MIMO* system [96].

It appears reasonable to broaden the *PID* control concept by developing the controller and delivering additional design freedom. This design can be done by including certain functions selected to have frequency responses property that may be required.

7.2.1 Problem Description

In this problem, the *RS* controller structure is in the traditional feedback form and is specified as *frequency-sensitive* functions set in the *PI* and the *PID* controller. These are multiplied by gains obtained in the similar *MBPC* aspect over minimizing a *GPC cost-function* related to *PI* or *PID* type elements.

Predictive *PID* controller was presented in [22], with qualities of *MBPC*, where the controller conduct to the equal structure as a *PI* or *PID* controller for first or *second-order* systems, respectively. The predictive *PID* control algorithm in [22] with analogous *GPC* controller performance provided a *PID* controller tuning technique via online optimization. It can operate as *PID* or *GPC* controller for *MIMO* systems. These are some of the attempts formed to get in the benefits of *PID* with predictive control. The *RS* approach introduced here for the *QTP* example that has a strong interaction between different loops.

In this approach, several traditional controller structures can be utilized in the feedback loop, for instance, *lead-lag*, extended *PID* or general transfer function structure. One of the solutions vital ideas was that the system is exciting if the rank condition was assured for a matrix whose inverse was needed [14]. Excluding the particular case, when the controller gains weighting tends to zero, this assumption is not needed in this solution.

7.2.2 System Model Description

The *QTP* illustration is given in Figure 7-2 has been discussed in [97]. There are two motivating features of this process. Initially, this process has extensive interactions in a loop between $Tank^1$ and $Tank^3$, and among $Tank^2$ and $Tank^4$. This interaction is due to input from $Pump^1$ fills $Tank^1$ and $Tank^4$, besides, $Tank^3$ output fills $Tank^1$. The same interface arises between $Tank^2$ and $Tank^4$.

This interaction results in an elevated effect on appropriate control. One more attractive feature is that the *QTP* linearized model has a transmission zero adjusted to be in the left or right *half-plane* by adjusting a manual flow valve. Therefore, the *QTP* process can be selected to be a minimum phase or a *non-minimum* phase. The *QTP* mathematical model utilizing mass balances and *Bernoulli's* law are given in equations (7.6)-(7.9) as:

$$\frac{dh_1}{dt} = -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_1 k_1}{A_1}v_1 \tag{7.6}$$

$$\frac{dh_2}{dt} = -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_2 k_2}{A_2}v_2 \tag{7.7}$$

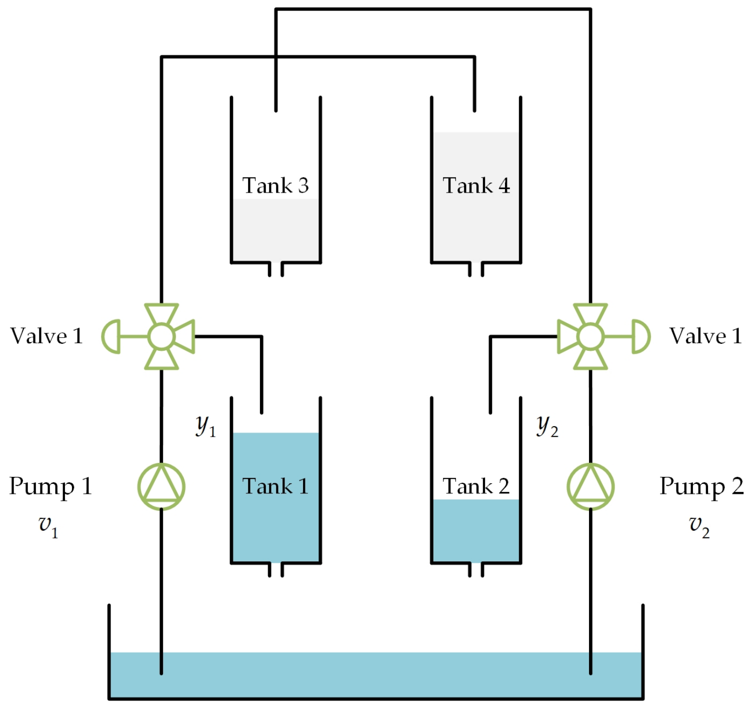

Figure 7-2: Quadruple tank process system

$$\frac{dh_3}{dt} = -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_2)k_2}{A_3}v_2 \tag{7.8}$$

$$\frac{dh_4}{dt} = -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_1)k_1}{A_4}v_1 \tag{7.9}$$

Given γ_l is the fluid flow ratio to the lower and diagonal upper tank, and A_l is the *cross-section* area, a_l is the *cross-section* of the drain hole and h_l is the liquid level, in $Tank^l$ equally. The *QTP* is a 2×2 *MIMO* system, and the goal of the control is to maintain the lower tanks liquid level setpoint tracking by adjusting liquid flow through the connected two pumps. These pumps are controlled by the voltage v_1, v_2 with the related flow $k_l v_l$. And $y_1 = k_c h_1$, $y_2 = k_c h_2$ are the outputs where k_c is the gain of the level sensor. The *QTP* process parameter values are listed in Table 7-1.

Parameter	Unit	Value
A_1, A_3	cm^2	28
A_2, A_4	cm^2	32
a_1, a_3	cm^2	0.071
a_2, a_4	cm^2	0.057
k_c	V/cm	0.5
g	cm/s^2	981

Table 7-1: QTP tank parameter value for the simulation

Both $\gamma_1, \gamma_2 \in (0,1)$ are the valve parameters that adjust the flow spreading to lower and upper diagonal tanks in the same way. The flow to $Tank^1$ is $\gamma_1 k_1 v_1$ and $(1-\gamma_1)k_1 v_1$ to $Tank^4$ and as well for $Tank^2$ and $Tank^3$ as in Figure 7-2. g is the acceleration of gravity. For $\gamma_1, \gamma_2 \in (0,1)$, there are two zeros in this system, one constantly sits in the left *half-plane,* and the other could be decided to be in the left or right *half-plane* by manipulating the valve values of γ_1, γ_2 as follow:

- $1 < \gamma_1 + \gamma_2 \leq 2 \quad QTP\ is\ minimum\ phase$
- $1 < \gamma_1 + \gamma_2 \leq 1 \quad QTP\ is\ non-minimum\ phase$
- $\gamma_1 + \gamma_2 = 1 \quad QTP\ has\ transmission\ zero\ at\ origin$

7.2.3 Simulation Results

The *QTP* control is performed for all operating modes, P^- when the system is a minimum phase system and P^+ when the system becomes a *non-minimum* phase. The selected operating modes associated parameters values are given in Table 7-2. Introduce these variables:

$$x_i = h_i - h_i^0 \tag{7.10}$$

Parameter	P^- Phase	P^+ Phase
$h_1^0, h_2^0 \quad (cm)$	(12.4, 12.7)	(12.6, 13.0)
$h_3^0, h_4^0 \quad (cm)$	(1.8, 1.4)	(4.8, 4.9)
$v_1^0, v_2^0 \quad (V)$	(3.00, 3.00)	(3.15, 3.15)
$k_1, k_2 \quad (cm^3/V_s)$	(3.33, 3.35)	(3.14, 3.29)
γ_1, γ_2	(0.70, 0.60)	(0.43, 0.34)

Table 7-2: Operating mode parameter

$$u_i = v_i - v_i^0 \tag{7.11}$$

The *multi-loop PI* gains and the *state-space* linearized *QTP* model in (7.12) and (7.13) are given in [97] for the P^- phase as, $k_p^1, t_i^1 = 3.0, 30$, $k_p^2, t_i^2 = 2.7, 40$, and for P^+ phase system as, $k_p^1, t_i^1 = 1.5, 110$ and $k_p^2, t_i^2 = -0.12, 220$ similarly.

P^-System

$$A = \begin{bmatrix} 0.9686 & 0 & 0.0790 & 0 \\ 0 & 0.9781 & 0 & 0.0637 \\ 0 & 0 & 0.9197 & 0 \\ 0 & 0 & 0 & 0.9355 \end{bmatrix}, \quad B = \begin{bmatrix} 0.1639 & 0.0038 \\ 0.0020 & 0.1242 \\ 0 & 0.0918 \\ 0.0604 & 0 \end{bmatrix} \tag{7.12}$$

$$C = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

P^+System

$$A = \begin{bmatrix} 0.9689 & 0 & 0.0491 & 0 \\ 0 & 0.9784 & 0 & 0.0346 \\ 0 & 0 & 0.95 & 0 \\ 0 & 0 & 0 & 0.965 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0949 & 0.0038 \\ 0.0019 & 0.0691 \\ 0 & 0.1512 \\ 0.1099 & 0 \end{bmatrix} \tag{7.13}$$

$$C = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

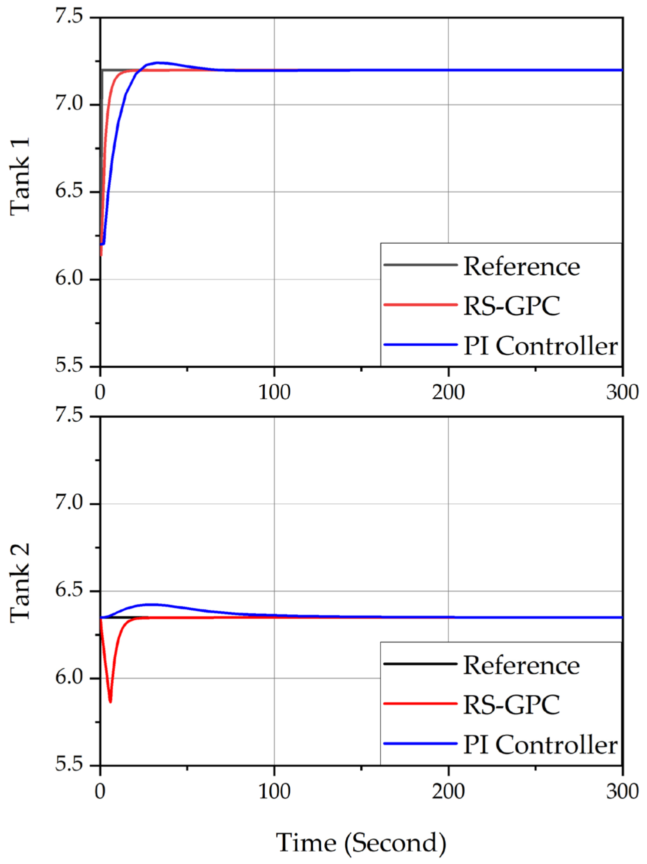

Figure 7-3: QTP minimum phase case controllers performance

7.2.3.1 Setpoint Regulation

A typical *multi-loop PI* controller has been employed for the *QTP* given modes. Overshoot has been remarked at 7.3 V with a *settling-time* of almost 60 seconds, as demonstrated in Figure 7-3 for minimum phase setup.

For the *non-minimum* phase setup, a slight inverse response mixed with a risen overshoot at 8.2 V with a *settling-time* around 1942 seconds has been noticed, as illustrated in Figure 7-4. The outcomes are marginally not the same as in [97] because simulation has been executed in the *time-domain* using the discretized *state-space* description of the model in contrast to the *frequency-domain* method used by *Johansson* [98].

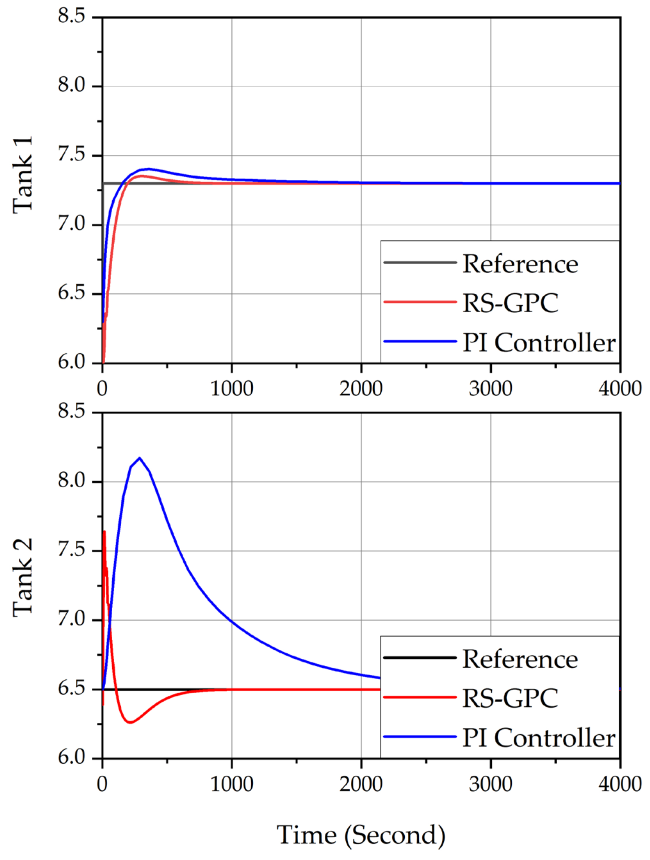

Figure 7-4: QTP non-minimum phase case controllers performance

To carry out the evaluation, initially, an *RS* controller in a *PI* structure has been employed to control the nonlinear *QTP* process model formed in regulating problem to assess the controller regulating capability.

For the minimum phase setup, the prediction horizon was selected $N = 40$, and the control horizon $N_u = 40$. Both outputs Y_1, Y_2 responses were noted without overshoot with a *settling-time* of 25 seconds, as revealed in Figure 7-3. Similarly, for the *non-minimum* phase setup, a prediction horizon of $N = 80$ and control horizon of $N_u = 20$ were chosen. Simulation reveals an inverse response down with lesser overshoot at 7.6 V for 6.5 V setpoint with *settling-time* just about 600 seconds, as illustrated in Figure 7-4.

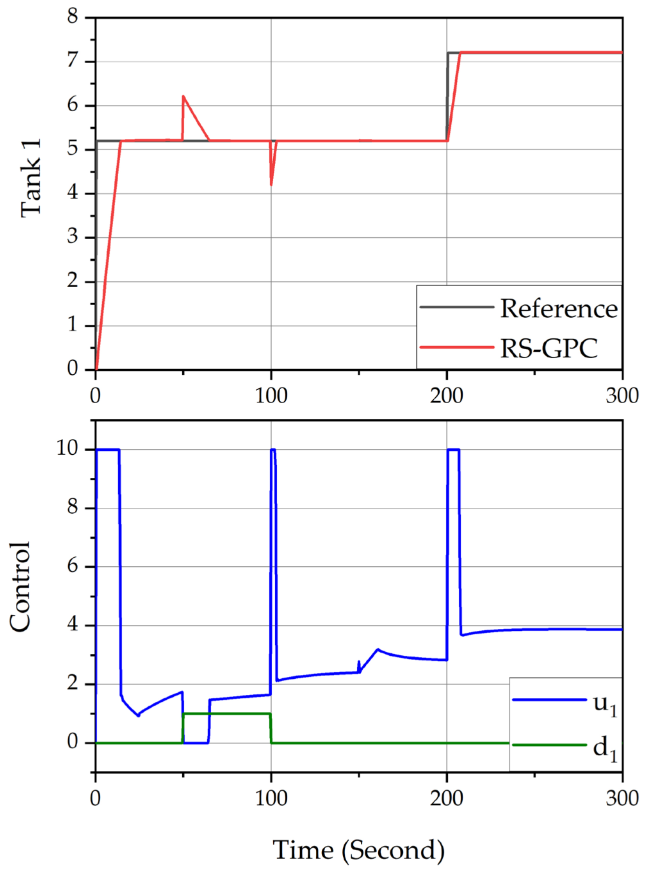

Figure 7-5: Tank¹ tracking and disturbance rejection performance

7.2.3.2 Reference Tracking

Next, for the *MIMO* system's elevated interaction between various loops, the *RS* controller has been formed in a *PID* structure for the nonlinear *QTP* to only assess the *RS* tracking feature. It should be noted here that comparing the *RS* controller performance against the fixed gain classical *PID* controller in this assessment will be unfair. The aim was to track setpoint deviation and improve the system's disturbance rejection, as displayed in Figure 7-5 and Figure 7-6. Thus, a step disturbance has been fed to the process outputs at various times. The simulation demonstrates a decent controller performance in eliminating the disturbance effects while maintaining sufficient tracking to deviation in multivariable system setpoints.

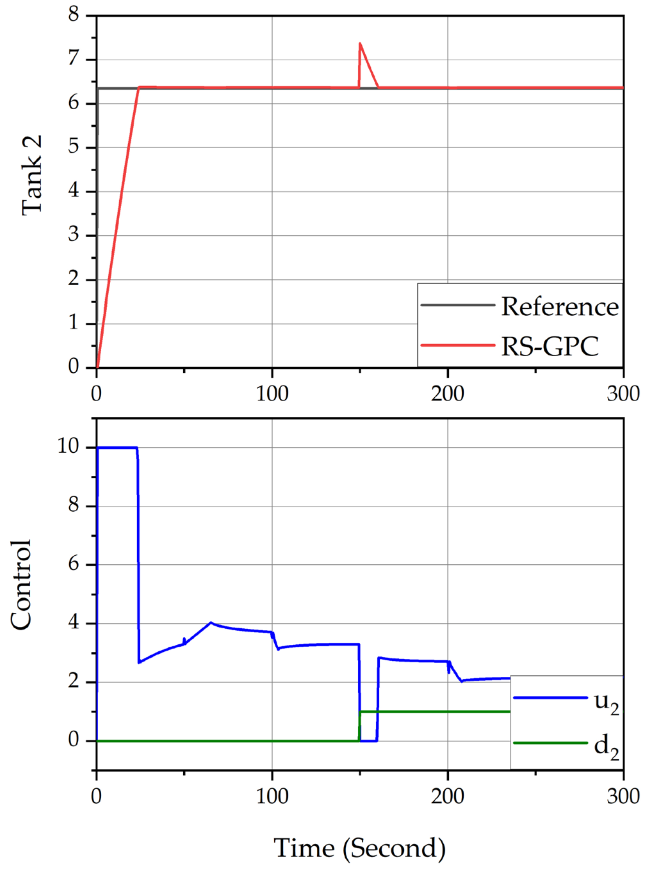

Figure 7-6: Tank² tracking and disturbance rejection performance

7.2.4 Summary

In this section, the *RS* controller has been formed to control the *QTP MIMO* process model and evaluated in both structures of *PI* and *PID* to demonstrate the benefits in performance and the flexibility of adopting the *RS* controller compared to the classical *PID*. The algorithm was utilized for both the regulating and the tracking problems, where optimized *time-varying* controller gains are revised and adjusted automatically. In this future knowledge, the *RS* controller adjusts to the reference trajectories' deviation and disturbance rejection. Performance evaluation comparison of the controller to that of a classical *PI* and *PID* controller has been carried out, and the findings show distinct and enhanced performance. The *RS* controller could be extended to control various nonlinear systems classes. In this example, the controller can be effortlessly developed to cover more of the wider *QTP* operation regions.

7.3 Electronic Throttle Body

Achieving full electronic control of the engine is crucial in satisfying automotive performance requirements and emissions legislation. Therefore, providing accurate control of the *ETB* becomes very important since it is a vital module employed in the air intake and engine torque controls, especially in *SI* engines which demands air and fuel precise control.

The *ETC* delivers the link among the *ETB* and the acceleration pedal, utilizing electrical signals to replace a mechanical link, as shown in Figure 7-7. As in Figure 7-8, a typical *ETB* involves a butterfly valve connected to an electric motor over the gears set and the *TPS*.

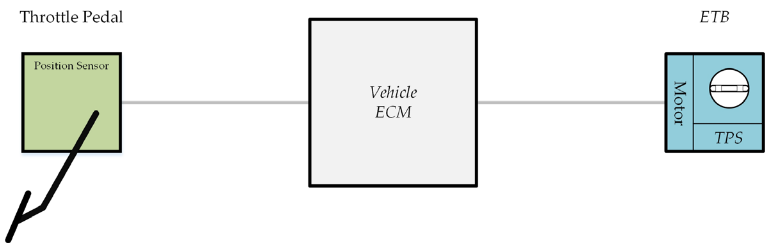

Figure 7-7: Electronic throttle control system

7.3.1 Problem Description

The automotive engineering field has shown that the *ETC* system must deliver a small *steady-state* error, quick *settling-time*, handling constraint and robustness versus alterations in the *ETB* parameters and pressure caused by production alterations and the disturbance torque [99]. This instant and precise control prerequisite must also manage the inherent nonlinearity and the *ETB* mechanical structure time inconsistency [100].

Because of its simplicity, *PID* is frequently applied for *ETB* control. Though in several situations and certain operating conditions, *PID* cannot accomplish the required performance, particularly if a low variation is needed in air flow rate due to the spring force, friction nonlinearities and the disturbance torque impacts. Therefore, it would be logical to expand *PID* control's abilities, upgrading the controller to further design freedom. A number of investigations on the *ETB* control have been made, for instance the *PID* controller with nonlinear compensator [99], [101], model approximation control [102], [103], adaptive control [104], constrained time optimal control [105], *self-learning PID* control [106], and *fractional-order fuzzy-PID* [107].

7.3.2 System Model Description

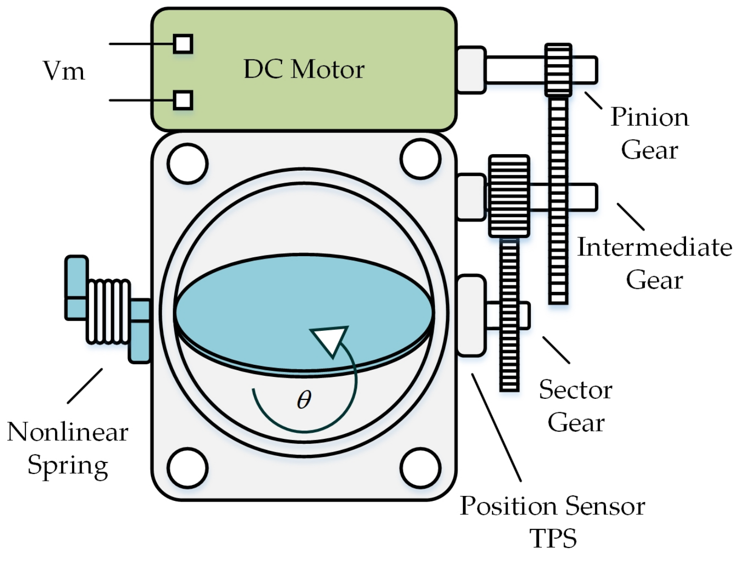

Figure 7-8: Typical electronic throttle body system

In this section, the nonlinear *ETB* model is modelled as a *qLPV* system to catch the *ETB* model nonlinearities. The valve angle and velocity are utilized as the scheduling parameter, and the disturbance torque, which depends on the airflow and other sources, was considered an exogenous input.

The *ETB DC* motor dynamics in Figure 7-9 was illustrated in [108],[109] and are characterized by (7.14).

$$V_m = iR + L\frac{di}{dt} + E_m \tag{7.14}$$

where V_m, R, i and L are motor voltage, resistance, current and inductance individually. Here $E_m = K_m\dot{\theta}_m$ is the motor back *EMF* voltage and K_m is the *EMF* coefficient. The armature lag is ignored, anticipated to the armature inductance nominal value, and similarly, the motor time constant lesser value in contrast to the sampling time T_s [110].

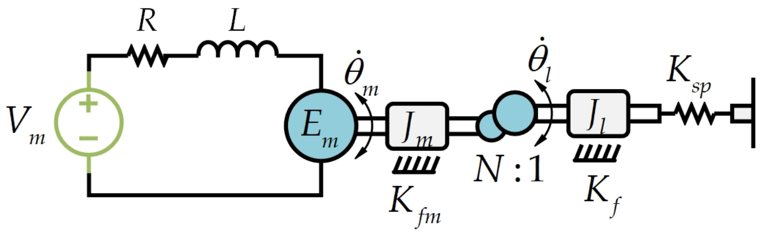

Figure 7-9: Lumped-element model of the ETB

The *DC* motor mechanical dynamics are presented in (7.15).

$$J_m\ddot{\theta}_m = T_m - K_{fm}\dot{\theta}_m - T_g \tag{7.15}$$

where J_m motor inertia, K_{fm} motor friction coefficient, $\dot{\theta}_m$ motor angular velocity, T_m motor armature torque and T_g is the reduction gears, transferred torque. The torque produced by the armature $T_m = K_a i$ is proportional linearly to the armature current and K_a is the motor torque coefficient. The armature torque T_m is characterized as in (7.16).

$$T_m = \frac{K_a}{R}(V_m - K_m\dot{\theta}_m) \tag{7.16}$$

The induced load torque T_l on the throttle valve shaft through the reduction gears:

$$T_l = NT_g \tag{7.17}$$

where N is the complex *gear-ratio,* and the throttle plate dynamics are defined in (7.18):

$$J_l\ddot{\theta}_l = T_l - \check{T}_{sp} - \check{T}_f - T_d \tag{7.18}$$

where $\check{T}_{sp}$, $\check{T}_f$ and T_d are dynamics caused and charged by the *LH* spring, friction and disturbances, correspondingly.

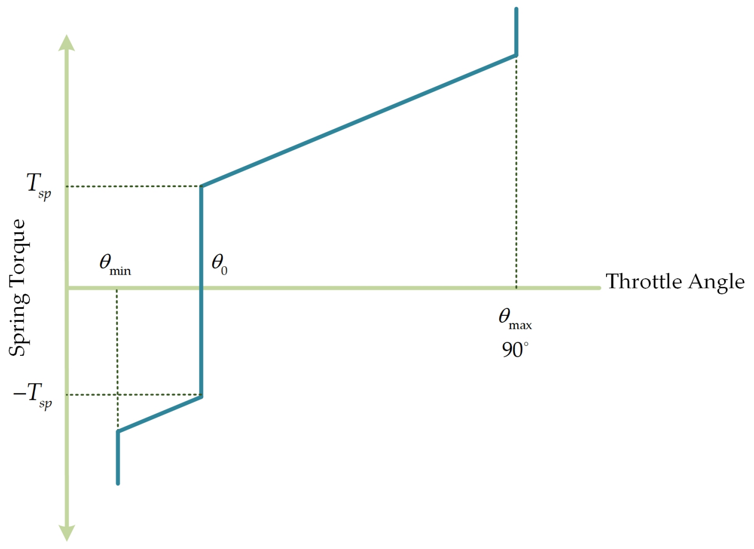

Figure 7-10: LH spring nonlinearity

There are two inbuilt springs employed in the *ETB* for the *fail-safe* mechanism. The valve plate is reverted by these springs' force to *LH* position slightly over the closed location if there is no power employed to the *ETB* actuator. This design is to permit sufficient air to be supplied to the engine when there is no applied control. This torque is subject to whether the throttle valve is in the *LH* position, moving forward or reverse. A maximum and minimum angle limits additionally constrain the valve plate movement.

These constrained limits are viewed as too stiff spring, represented with infinite force, as illustrated in Figure 7-10, as stated in (7.19).

$$\begin{aligned} \check{T}_{sp} &= K_{sp}(\theta_l - \theta_0) + T_{sp}\,\mathrm{sgn}\left(\theta_l - \theta_0\right) \\ T_{sp}\,\mathrm{sgn}\left(\theta_l - \theta_0\right) &= \begin{cases} +T_{sp} & \theta_0 < \theta_l < \theta_{\max} \\ -T_{sp} & \theta_{\min} < \theta_l < \theta_0 \end{cases} \end{aligned} \tag{7.19}$$

where T_{sp} preload torque of the spring, K_{sp} spring stiffness coefficient, θ_0 is the *LH* angle and sgn denote signum function. In friction anticipated nonlinearity, the generated force looks like it resists the throttle plates motion, for instance, linear viscous damping, which relies on nonlinear Coulomb friction, displayed in Figure 7-11, and velocity. The total produced friction torque can be described mathematically by (7.20).

$$\begin{aligned} &\check{T}_f = K_f\,\dot{\theta}_l + T_f\,\mathrm{sgn}(\dot{\theta}_l) \\ &where \quad T_f\,\mathrm{sgn}(\dot{\theta}_l) = \begin{cases} T_f & \dot{\theta}_l > 0 \\ 0 & \dot{\theta}_l = 0 \\ -T_f & \dot{\theta}_l < 0 \end{cases} \end{aligned} \tag{7.20}$$

where T_f is the Coulomb friction torque, K_{fl} viscous friction coefficient, and $\dot{\theta}_l$ is the valve angular velocity. Replacing for (7.19) and (7.20) in (7.18) returns the next throttle plate nonlinear dynamics.

$$J_l\ddot{\theta}_l = \left[NT_g - [K_{sp}(\theta_l - \theta_0) + T_{sp}\ sgn(\theta_l - \theta_0)] - [K_{fl}\dot{\theta}_l + T_f\ sgn(\dot{\theta}_l)] - T_d\right] \tag{7.21}$$

Combining (7.15), (7.16) and (7.21) then, the differential equation showing the *ETB* nonlinear model develops as in (7.22).

$$\begin{aligned} &J_t\ddot{\theta} = \\ &\left[\frac{NK_aV_m}{R} - \left(K_{ft} + \frac{N^2K_aK_m}{R}\right)\dot{\theta} - T_f\,\mathrm{sgn}(\dot{\theta}) - K_{sp}\theta - T_{sp}\,\mathrm{sgn}(\theta) - T_d\right] \end{aligned} \tag{7.22}$$

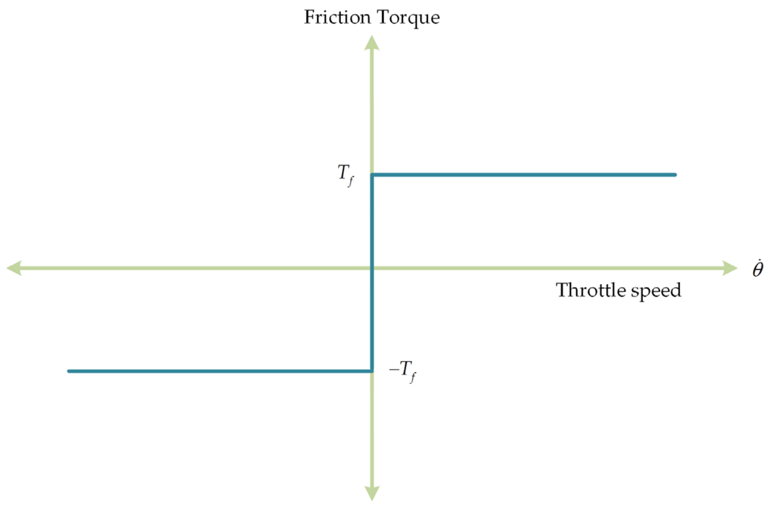

Figure 7-11: Coulomb friction nonlinearity

where

$$J_t = J_l + N^2 J_m, \quad K_{ft} = K_{fl} + N^2 K_{fm}$$
$$\theta = (\theta_l - \theta_0), \qquad \theta_l = N\theta_m \tag{7.23}$$

During engine operation, some various dynamics and disturbances disturb the *ETB* system tracking performance. These dynamics can be classed in a known dynamic, e.g. car battery voltage deviation and unmeasured dynamic, e.g. the throttle plate disturbance torque stimulated by pressure variations due to the engine *air-flow* dynamics. Hence, the *ETB* model in *qLPV state-space* is offered in (7.24).

$$\begin{aligned} \dot{x} &= A_0(\rho)x + B_0 u + D_0 T_d \\ y &= C_0 x \end{aligned} \tag{7.24}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} \tag{7.25}$$

Parameter	Value	Parameter	Value
N	20.68	K_{sp}	0.087 Nm/rad
R	1.15Ω	K_{ft}	0.0088 $Nm.s/rad$
J_t	0.0021 $kg.m^2$	T_{sp}	0.396 Nm
K_a	0.0185 Nm/A	T_f	0.284 Nm
K_m	0.0185 $V.s/rad$		

Table 7-3: Bosch ETB parameters

The throttle angle and the velocity are the *state-vector* x components. Usually, the *qLPV* system models are when any of the scheduling parameters ρ are system states [111]. Consequently, matrices A and B are typically nonlinear functions of the scheduling parameter ρ. Therefore, the components of (7.24) turn out to be as in (7.26)

$$A_0 = \begin{bmatrix} 0 & 1 \\ -\frac{K_{sp}}{J_t} - \frac{T_{sp}}{J_t}\rho_1 & -\frac{1}{J_t}\left(K_{ft} + \frac{N^2 K_a K_m}{R}\right) - \frac{T_f}{J_t}\rho_2 \end{bmatrix}$$

$$B_0 = \begin{bmatrix} 0 \\ \frac{NK_a}{J_t R} \end{bmatrix}, \qquad D_0 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \qquad C_0 = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{7.26}$$

$$\rho_1 = \frac{1}{|x_1| + \varepsilon}, \quad \rho_2 = \frac{1}{|x_2| + \varepsilon} \tag{7.27}$$

where ρ_1, ρ_2 are varying parameters obtained from measurement or estimation at every sampling instant; in a process to estimate nonlinear system behaviours. The signum function $sgn(x)$ in (7.22) can also be estimated by $x/(|x|+\varepsilon)$, where ε is a very small positive scalar [112], utilized to assure an accurate computation in case state x tends to zero. Though the estimated effects will be rounded to the closest integer number in the online computation, this estimate's impact will be eliminated.

The discrete equivalent is approximated by replacing the continuous model in (7.26) as follows:

$$x(t+1)=\begin{bmatrix} 1 & T_s \\ T_s(-\frac{K_{sp}}{J_t}-\frac{T_{sp}}{J_t}\rho_1) & 1-T_s\left(\frac{1}{J_t}(K_{ft}+\frac{N^2K_aK_m}{R})-\frac{T_f}{J_t}\rho_2\right)\end{bmatrix}x(t)$$
$$+\begin{bmatrix} 0 \\ \frac{T_sNK_a}{J_tR}\end{bmatrix}u(t-1)+\begin{bmatrix} 0 \\ -T_s\end{bmatrix}T_d$$

where a high sample rate T_s is assumed to ensure the *discrete-time* model is more appropriate.

7.3.3 Simulation Results

The *ETB* parameters in Table 7-3 were provided in [113] and utilized for the simulation to evaluate the controller's performance. The nonlinear *RS-PID* was primarily used assuming known dynamics. The controller demonstrates excellent tracking and regulation behaviours, as shown in Figure 7-12. The *RS* controller was used to generate *PID* gains, as shown in Figure 7-13.

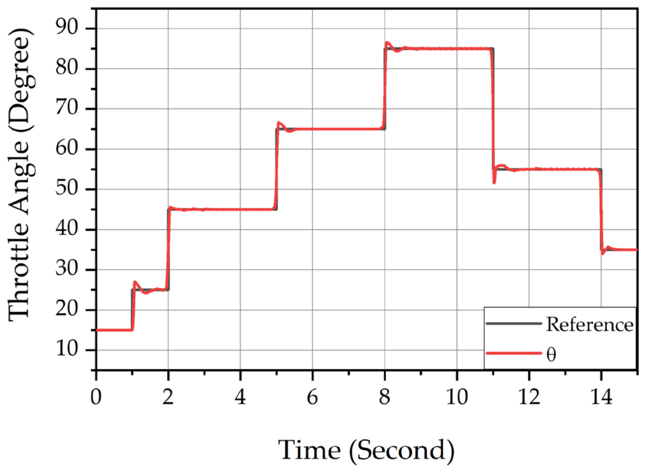

Figure 7-12: ETB nonlinear RS-PID controller

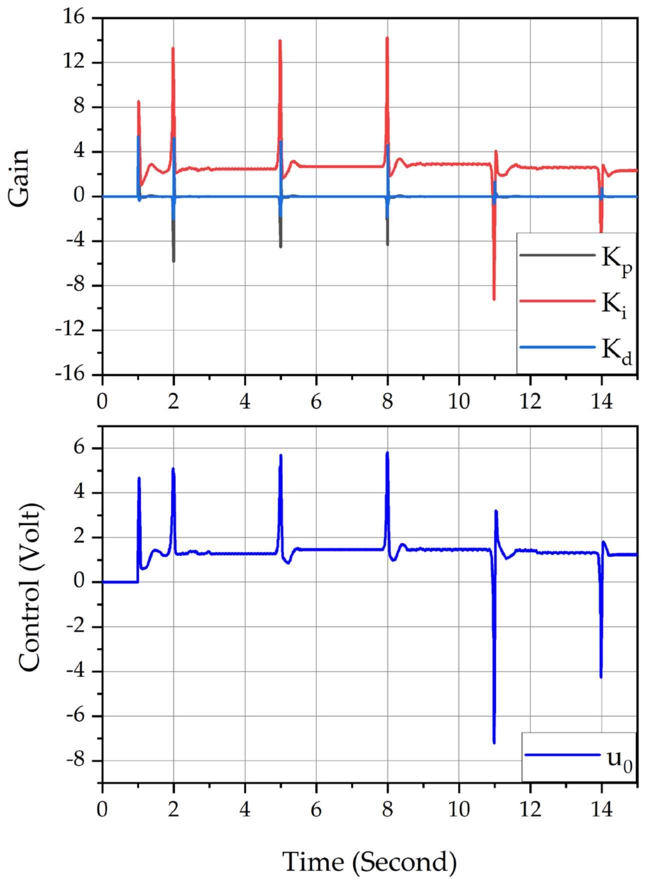

Figure 7-13: ETB nonlinear RS-PID controller gain

The 1% *settling-time* of 63 milliseconds was noticed for equally the small and big signals with a maximum of 1.7% overshoot. Likewise, a small *steady-state* error of less than 0.05 degree was noticed, as displayed in Figure 7-12. The *RS* controller has delivered a performance ahead of some automotive performance standards, such as the recommended control responses of S80 Volvo [114].

7.3.4 Performance Tuning

The dynamic weightings are employed to add a typical integral action to this controller and smooth the *ETB* performance. In this simulation section, various dynamic weighting values were employed to illustrate the tuning flexibility in *ETB* performance suitable for commissioning engineers.

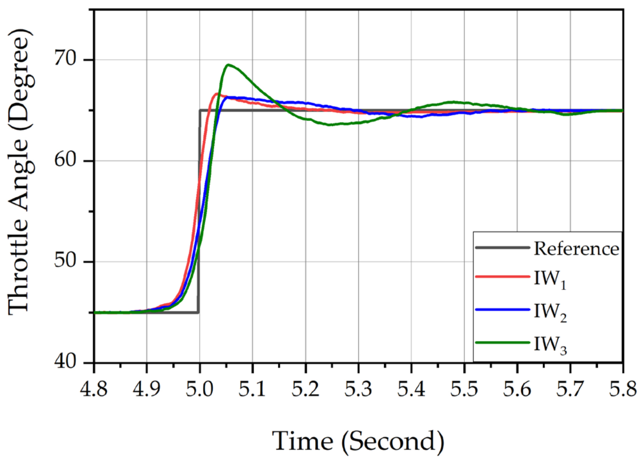

Figure 7-14: Integral dynamic weightings performance analysis

In Figure 7-14, the dynamic weighting proportional term was kept constant while testing an integral term changing performance effects.

The simulation shows a reduction in *steady-state* error as a result of the increase in an integral part. Also, a *steady-state* error of 0.001 degrees was accomplished, but with a minor overshoot of 6%. In Figure 7-15, the same practice was followed, the integral term was kept constant while testing proportional term changing effects on performance, and the impact was assessed versus the rise time. Little progress of 15 milliseconds was remarked with an increased overshoot of 15%, which a cut in an integral part could undoubtedly restrain.

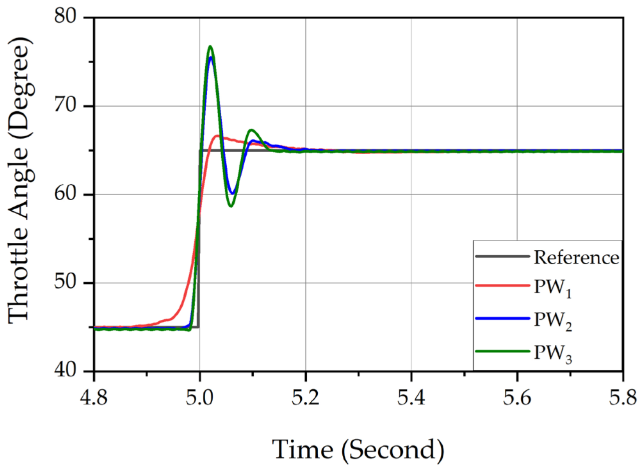

Figure 7-15: Proportional dynamic weightings performance analysis

Sports car performance requirements are separate from a car designed to be *fuel-efficient*. These outcomes demonstrate how the fast *servo-mechanism* system can be calibrated, and a new direct tuning feature utilized for commissioning engineers are offered to deliver the necessary performance. Lastly, the *RS* controller robustness was assessed against the *ETB* parameters uncertainty that may affect the parameters given in Table 7-3, and the results show robustness versus variations in *ETB* parameters.

7.3.5 Summary

In this example, a nonlinear *RS* controller is implemented based on the *ETB qLPV* model to control the *ETB* nonlinear model. The *RS* controller has been provided in a *PID* controller form and was utilized in a *reference-tracking* assessment. In this tracking assessment, optimized *time-varying* gains controller were continuously updated using future predictions. The *RS* controller can adjust to the deviations in reference trajectories or even variations in system operating regions switching caused by friction and spring nonlinearities. The performance was evaluated using some automotive performance standards, and the nonlinear *RS* controller findings demonstrated enhanced performance beyond these measures. Finally, the dynamic weighting tuning features were manipulated to enhance the *ETB* transient responses.

7.4 Continuous Stirred Tank Reactors

The *CSTR*s are prevalent in chemical and pharmaceutical systems. These systems have an extremely nonlinear behaviour and have a wide operating range. Furthermore, they may sometimes be appointed to operate in different operating regions to produce a variety of separate manufactured goods. This process's ultimate goal is to achieve flexible manufacturing and coping with market competition [115]. Hence, a crucial control goal is to reduce the product transition time by decreasing the volume of *off-specification* products manufactured throughout any transition [116].

7.4.1 Problem Description

The *CSTR* processes reveal deep nonlinear dynamics, including various *steady-state* solutions and together with stable and unstable equilibrium points. An exciting feature of the *CSTR* model is the two stable regions, at both edges of the output span, separated by an unstable region in which the *CSTR* system tuning becomes a difficult task and the system output looks repulsed. *PWA* system is an essential class of *HS* and a proper structure to describe or approximate numerous physical processes, such as the nonlinear *CSTR,* by approximating the system using multiple linearizations around different operating points. This work will follow the previous methodology provided in Section 5.5 to control *SD* systems by first obtaining the *PWA* system in an *SD* form and constructing the nonlinear *RS* controller in *PI* form, which is easy to calculate and implement. Noting that, hard constraints can be included using the *QP* solution.

7.4.2 System Model Description

A standard *two-state CSTR* system with an exothermic irreversible *first-order* reaction $A \rightarrow B$ is illustrated in Figure 7-16 and used as the *case-study* for this simulation. The C_A, T, q_c and T_{cf} are resultant concentration, reactor temperature, coolant flow rate, and coolant temperature correspondingly. The *CSTR* system output is C_A, the input is T_{cf}, and the states of the system are:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} C_A \\ T \end{bmatrix}$$

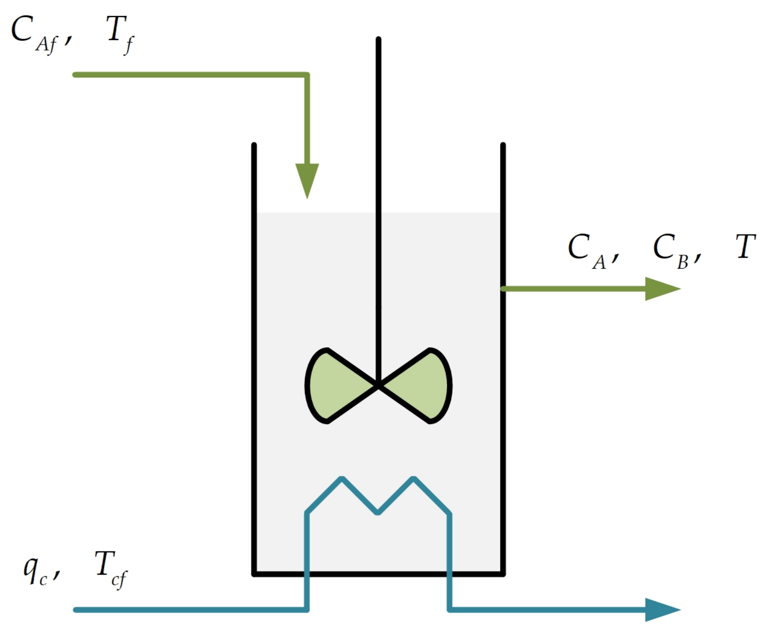

Figure 7-16: Continuous stirred tank reactors

The dynamics of the *CSTR* can be defined by the subsequent nonlinear representations [117]:

$$\begin{aligned} \frac{dx_1}{dt} &= -\theta x_1 \kappa(x_2) + q(x_{1f} - x_1) \\ \frac{dx_2}{dt} &= \beta\theta x_1 \kappa(x_2) - (q+\delta)x_2 + \delta u + q x_{2f} \\ y &= x_1 \end{aligned} \qquad (7.28)$$

where

$$\delta = 0.3, \lambda = 20, \theta = 0.072, \beta = 8, q = 1$$

$$\kappa(x_2) = \exp\left(\frac{x_2}{1+\dfrac{x_2}{\lambda}}\right), x_{1f} = 1, x_{2f} = 0$$

The *CSTR* has three *steady-states* at $u = 0$:

$$x_{s_1} = \begin{bmatrix} 0.856 \\ 0.886 \end{bmatrix}, x_{s_2} = \begin{bmatrix} 0.5528 \\ 2.7517 \end{bmatrix}, x_{s_3} = \begin{bmatrix} 0.2353 \\ 4.7050 \end{bmatrix} \tag{7.29}$$

The nonlinear system has been linearized at each *steady-state* region and then discretized by sampling time of 0.1 seconds to obtain the *CSTR* system in *PWA* model form [115] as:

$$x(k+1) = \begin{cases} A_1 x(k) + B_1 u(k) + b_1, x \in \Omega_1 \\ A_2 x(k) + B_2 u(k) + b_2, x \in \Omega_2 \\ A_3 x(k) + B_3 u(k) + b_3, x \in \Omega_3 \end{cases}$$

$$y(k) = Cx(k) \tag{7.30}$$

where

$$A_1 = \begin{bmatrix} 0.8889 & -0.0123 \\ 0.1254 & 0.9751 \end{bmatrix}, B_1 = \begin{bmatrix} -0.0002 \\ 0.0296 \end{bmatrix}, b_1 = \begin{bmatrix} 0.1060 \\ -0.0852 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 0.8241 & -0.0340 \\ 0.6365 & 1.1460 \end{bmatrix}, B_2 = \begin{bmatrix} -0.0005 \\ 0.0322 \end{bmatrix}, b_2 = \begin{bmatrix} 0.1907 \\ -0.7537 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0.6002 & -0.0463 \\ 2.4016 & 1.2430 \end{bmatrix}, B_3 = \begin{bmatrix} -0.0007 \\ 0.0338 \end{bmatrix}, b_3 = \begin{bmatrix} 0.3119 \\ -1.7083 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Using (5.35) and (5.36), the *PWA* model (7.30) can be written in *SD* form as follow:

$$x_{sd}(t+1)=\overbrace{\left\{\begin{array}{l}\delta_1\times\begin{bmatrix}0.8889 & -0.0123\\ 0.1254 & 0.9751\end{bmatrix}+\\ \delta_2\times\begin{bmatrix}0.8241 & -0.0340\\ 0.6365 & 1.1460\end{bmatrix}+\\ \delta_3\times\begin{bmatrix}0.6002 & -0.0463\\ 2.4016 & 1.2430\end{bmatrix}\end{array}\right\}}^{A_{sd}}x_{sd}(t)+\overbrace{\left\{\begin{array}{l}\delta_1\times\begin{bmatrix}-0.0002\\ 0.0296\end{bmatrix}+\\ \delta_2\times\begin{bmatrix}-0.0005\\ 0.0322\end{bmatrix}+\\ \delta_3\times\begin{bmatrix}-0.0007\\ 0.0338\end{bmatrix}\end{array}\right\}}^{B_{sd}}u(t-k)+$$

$$\overbrace{\left\{\begin{array}{l}\delta_1\times\begin{bmatrix}0.1060\\ -0.0852\end{bmatrix}+\\ \delta_2\times\begin{bmatrix}0.1907\\ -0.7537\end{bmatrix}+\\ \delta_3\times\begin{bmatrix}0.3119\\ -1.7083\end{bmatrix}\end{array}\right\}}^{b_{sd}}$$

$$y_{sd}(t)=\overbrace{\begin{bmatrix}1 & 0\end{bmatrix}}^{C_{sd}}x_{sd}(t)$$

where

$$\delta_1=\begin{cases}1 & if \quad 0.78<x_1\le 1\\ 0 & otherwise\end{cases}$$
$$\delta_2=\begin{cases}1 & if \quad 0.35<x_1\le 0.78\\ 0 & otherwise\end{cases} \tag{7.31}$$
$$\delta_3=\begin{cases}1 & if \quad 0\le x_1\le 0.35\\ 0 & otherwise\end{cases}$$

During the simulation, the following constraints must be satisfied:

$$\begin{bmatrix}0\\0\end{bmatrix}\le x_{sd}\le\begin{bmatrix}1\\6\end{bmatrix}$$
$$-2\le u\le 2 \tag{7.32}$$

7.4.3 Simulation Results

The need to build the *CSTR* flexible operating strategies and optimal switching between operating points motivates the design objective. The nonlinear plant was moved between the different *steady-states* in (7.29) by way of tracking the concentration setpoint throughout the operating regions.

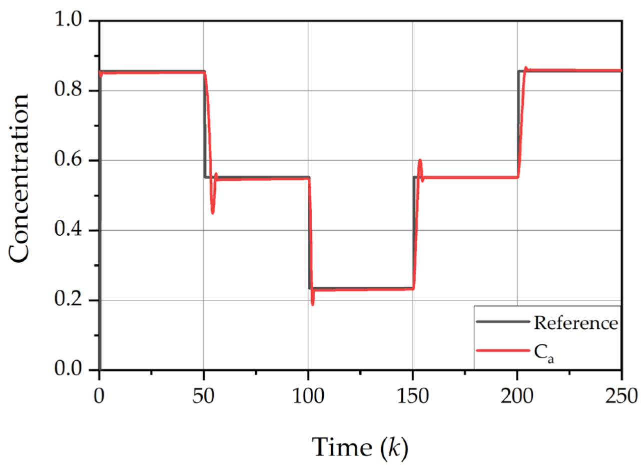

Figure 7-17: Operating points output tracking

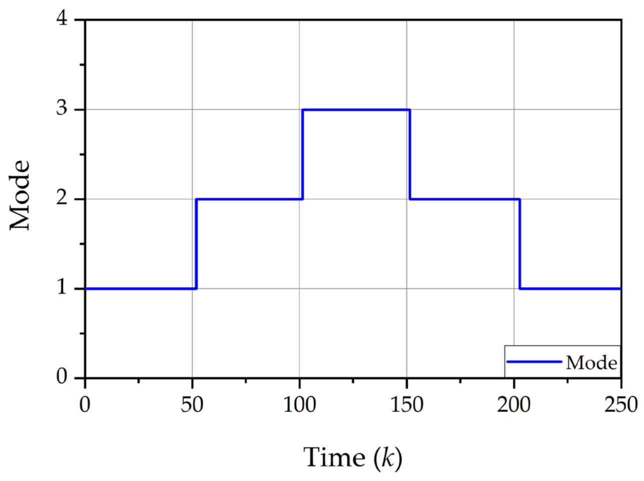

Figure 7-18: Evolution of the PWA regions

Simulation results showed adequate tracking performance in switching among operating points. The *RS* controller updates the within the loop *PI* controller's gains accordingly, as illustrated in Figure 7-17, with a slight overshoot around the unstable steady state x_{s_2}. Moreover, the controller tracking performance showed no oscillation during the *PWA* model switching from one region to another. The auxiliary logic variable in (7.31) acted as supervisor and was responsible for selecting the associated *PWA model-based* on the concentration measurement as shown in Figure 7-18 and allowed the nonlinear *RS* controller to update the within the loop *PI* controller gains accordingly, as displayed in Figure 7-19.

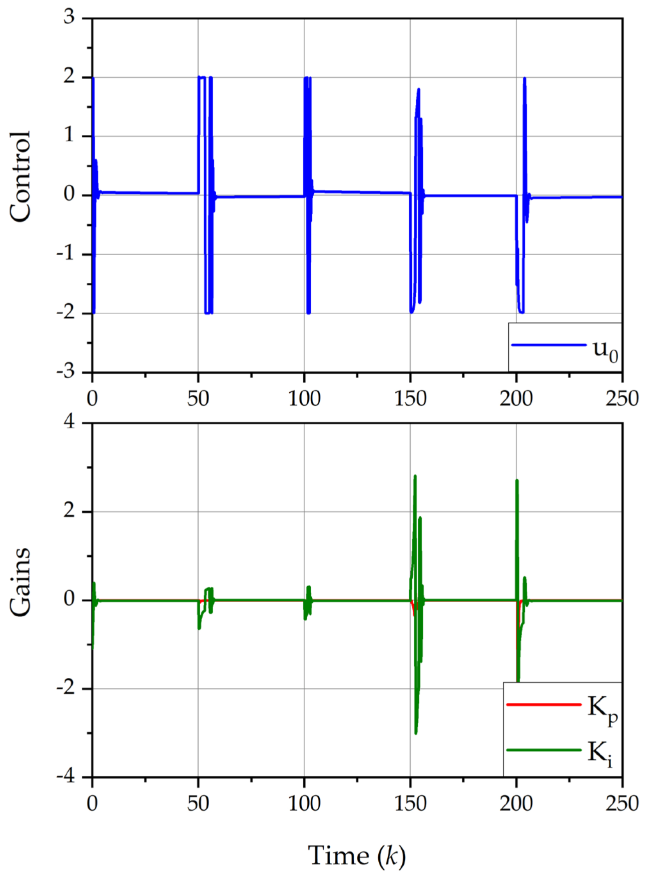

Figure 7-19: CSTR PI controller signal

7.4.4 Summary

A nonlinear *RS* controller has been designed based on a *PWA* model, whose transition dynamics depends on state or control input. This controller has been set to have a *PI* structure and was used to control a nonlinear model of *CSTR* in reference tracking, where the optimized *time-varying* controller gains were revised continuously. Utilizing future prediction, the nonlinear *RS* controller can adjust to the variations in the reference trajectory, and variations in system operating regions and satisfactory tracking performance was obtained.

7.5 Variable Camshaft Timing

The *IC* engine intake and exhaust valve timing significantly impact fuel consumption, emissions, and engine performance. In contrast, with traditional *valve-train* systems, the *VVT* design can enhance fuel consumption and decrease emissions at low engine speeds. Also, it can increase engine power and torque for high engine speeds. This approach is different from adjusting the intake and exhaust valve's timing for only one specified operational condition in the traditional design [118].

7.5.1 Problem Description

There are some parameters and modelling uncertainty in the *VVT* physical model due to variations in engine oil temperature and pressure. Sometimes, there are added problems in the system caused by the spool valve neglected unmodelled dynamics. The use of a linear *PI* controller to control the nonlinear *VVT* is not sufficient because of the low plant gain around the *steady-state* input value. Hence, the linear *PI* tracking performance declines due to the failure to provide sufficient loop gains.

The *LPV data-driven* method is attractive in cases where analytical modelling is sophisticated [53]. Since engineers often analyse systems' behaviour in *steady-state* conditions, where a given nonlinear operating point applies, it is common to use *frequency-domain* analysis methods. The use of what might be termed a *transfer-operator* model description which is a function of such external variables, is then very natural for many application areas. Hence, a *time-varying PI* is needed to achieve the desired performance.

7.5.2 System Model Description

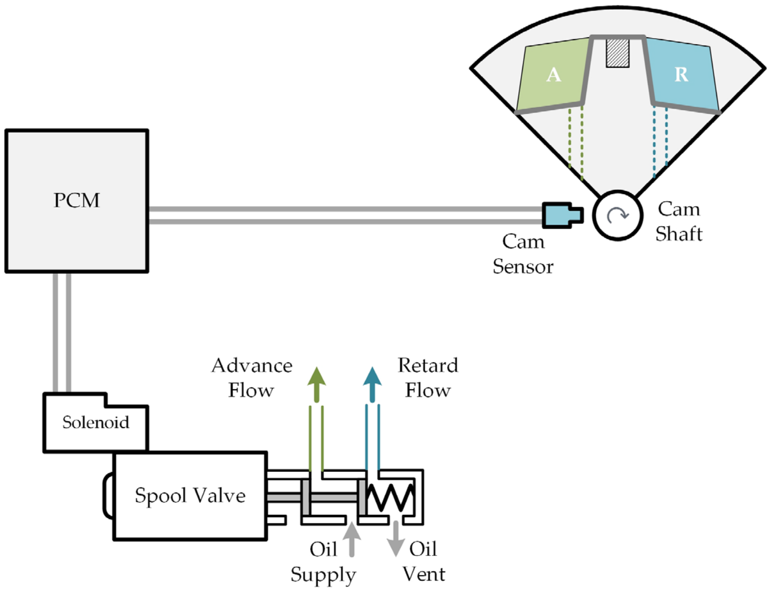

Figure 7-20: Variable camshaft timing system

The *VCT* involves a *VVT* solenoid, a solenoid interface circuit, and a hydraulic actuator attached to the camshaft with a range of ± 30 degrees, as illustrated in Figure 7-20.

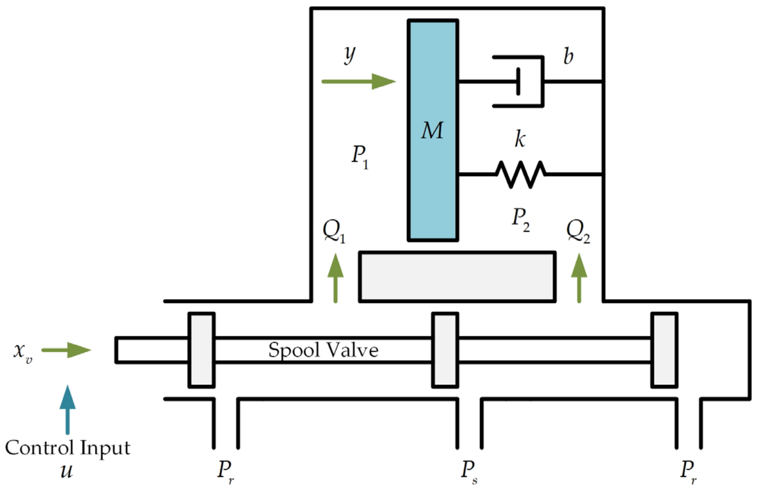

Figure 7-21: Hydraulic VVT actuator model

The *VVT* system exhibits a significant nonlinearity, and the system dynamics have been discussed in [119].

Oil Pressure	Engine Speed	ρ
310	900	0.70
	1500	0.72
	1800	0.68
414	900	0.95
	1500	0.98
	1800	0.93

Table 7-4: Time-varying parameters

The physical model of *VVT*, depicted in Figure 7-21, has a spool valve driven through a *PWM* signal to regulate the pressure P_1 and P_2 in (7.33) via the displacement x_v.

$$\ddot{y} = \frac{1}{M}\left(-b\dot{y} - ky + P_1A_1 - P_2A_2\right) \tag{7.33}$$

where *M* is the hydraulic actuator mass, *b* is the damping coefficient, *k* is the spring constant and A_1, A_2 are hydraulic actuator *left-side* and *right-side* areas correspondingly. These parameters are uncertain in the *VVT* physical model due to engine oil temperature and pressure variations. Other system uncertainties are due to spool valve neglected or unmodelled dynamics among control input and the spool position. The complete physical equations can be found in [120]. The result of *closed-loop* system identification in (7.34) employed in [121] and [122] is used in this simulation. The *VVT* system identification provided a group of linear models that approximate the *VVT* dynamics at selected engine speed and oil pressure. This identification was summarized in [123] against fixed engine speeds N_e and oil pressures P_o vectors as depicted in Table 7-4. The *transfer-functions* of an identified group of linear *VVT* systems sampled at 5 milliseconds:

$$G\left(z^{-1},\rho\right) = z^{-1}\left(\frac{0.0859\rho - 0.0609\rho z^{-1}}{1 - 1.9547z^{-1} + 0.9553z^{-2}}\right) \tag{7.34}$$

7.5.3 Simulation Results

The *RS* controller in *PI* structure performance was assessed on the *LPV* model in (7.34). A step input was used as the reference signal and varied between the Cam advance $-20^o \to 0^o$ and the Cam retard $0^\circ \to 20^\circ$ for different engine speed N_e and oil pressure P_o as illustrated in Figure 7-22, Figure 7-23 and Figure 7-24. A classical *PI* controller was also tuned for the *VVT* for comparison purposes.

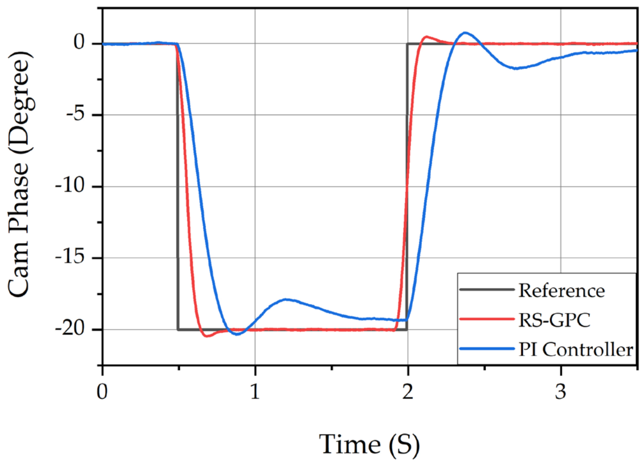

Figure 7-22: Cam advance and retard response at $P_o = 310, N_e = 900$

The *PI* tuning aimed to achieve a reasonable balance between fast response time and overshoot for selected engine speeds and oil pressures [123]. To compare over a wide range of operations, a large step size, as demonstrated in Figure 7-22 and Figure 7-23, was used, and then a small step size as displayed in Figure 7-24. These show the reference signals for both cases and both the *RS-GPC* and the classic *PI* control designs. Figure 7-22 and Figure 7-23 show how the *RS-GPC* overcome system parameters variation and adapt to the increase/decrease in *open-loop* gain. It adjusts the *PI* controller gains accordingly to both the Cam advance and retard compared to the fixed gain *PI* controller. The weak performance of classical *PI* in Figure 7-22 is expected because of the low plant gain around *steady-state* and system parameters variations at low limits.

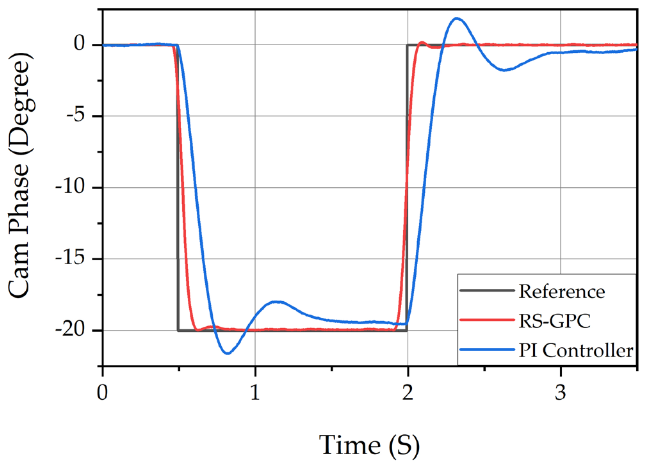

Figure 7-23: Cam advance and retard response at $P_o = 414, N_e = 1800$

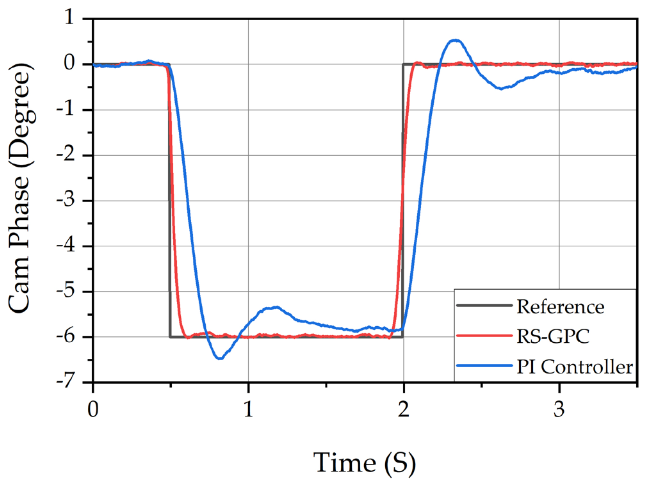

Figure 7-24: Cam advance and retard response at $P_o = 414, N_e = 1800$

It is also noticed that for the whole test trials, the *RS-GPC* controller settles faster than the classical *PI* controller. Figure 7-24 shows that a small step size 6^o for both Cam advance and retard was applied. The simulation results reveal that the *RS-GPC* controller is much responsive to the error signal variation than the classical *PI*. Note that the *PI* controller could be *gain-scheduled*, which would make for a fairer comparison; on the other hand, the *RS* approach can also be used for automatically scheduling the *PI*.

Cases	P-Index	PI	RS-GPC
$0^o \rightarrow -6^o$	ITAE	1.072	0.2588
	IAE	1.47	0.3091
	ISE	3.916	0.3333
$0^o \rightarrow -20^o$	ITAE	3.278	1.282
	IAE	4.707	1.221
	ISE	42.89	4.369

Table 7-5: PI and RS-GPC controllers evaluation criteria

The data in Table 7-5 reveals the *RS-GPC* controller performance against the classical *PI* controller for both the small and big signal cases. The improved control performance is expected as the *RS-GPC* adapt to changes in oil pressure and temperature.

7.5.4 Summary

A polynomial systems approach to the modelling and optimal predictive control of *LPV* systems, represented in polynomial model form, was considered. The controller structure involved *RS* polynomial matrix form. An advantage of the approach is that the *low-order RS* controller within the feedback loop is simple to retune by conventionally trained engineers. Simultaneously, the approach enjoys *model-based* optimal control benefits. The gains are computed online to minimize a predictive control *cost-function* that is quite general. It can include *LPV* dynamic *cost-function* error weightings and *cost-weightings* on the magnitude of the controller gains.

The *RS* controller was fixed at a *PI* structure in the example. It was used to control a *VVT* example in a *reference-tracking*, where the optimized *time-varying* controller gains were revised continuously. Using future prediction, the *RS-GPC* can adjust to variations in reference trajectory and system parameters. Although this is an unfair comparison between a very simple and a much more complicated controller, it demonstrates that when performance is paramount in some applications, the added complexity may be justified.

The *RS* approach may be applied to a system described by a *state-space* model in *LPV* form, which is very suitable for large systems and often follows the system's physical equations more clearly. However, the polynomial *LPV* model form is more natural for systems that are identified online. It is also very suitable since the *RS* controller structure is usually chosen to have a classical controller form.

7.6 Chapter Summary

In this Chapter, four examples have been provided to assess the performance of the proposed *RS* algorithm. In Section 7.2, a multivariable *QTP* was used to test the linear *RS* controller solution that has been given in Chapter 4. The *RS* controller was formed in the *PI* and *PID* structure. Both structures benefit from the proposed optimal solution to overcome the high interactions in the *QTP* loops. In Section 7.3, an *ETB* was chosen to test the nonlinear *RS* solution provided in Chapter 5. The nonlinear *ETB* model was modelled as a *qLPV* system to catch the *ETB* model nonlinearities. A section also shows the tuning flexibility in *ETB* performance that commissioning engineers may have in shaping the *ETB* performance by selecting different dynamic weighting values.

The nonlinear *RS* solution in Chapter 5 was also utilized for the *CSTR* example given in Section 7.4. The *CSTR* was identified in *PWA* systems representation, and the procedures discussed in Section 5.5 was used to convert the resultant *PWA* system to a proper *SD* structure. Then, the nonlinear *RS* controller in *PI* form was constructed as given in Section 7.4. Finally, the *LPV* polynomial *RS* solution provided in Chapter 6 was evaluated and tested on the polynomial *LPV* representation of an automotive *VCT* model. The nonlinear *RS* controller was formed in the *PI* structure, and the performance was compared and tested against the classical *PI* controller as given in Section 7.5.

Chapter 8 Conclusion

This work's main objective was to expand the *RS* solution to apply to a broad nonlinear systems class by utilizing the *LPV* modelling approach to use in nonlinear control and extend some of its features and illustrate its usefulness via industrial examples. The work described focused on design questions and mainly concerned with the various algorithmic formulations' implementation and performance aspects.

8.1 Review

Some of the essential qualities that differentiate this work from the previous approaches mentioned in Chapter 1 are its flexibility in adapting unknown system nonlinearities via the decomposition into a general linear operator considered a *black-box* in Chapter 4. The *RS* controller simply generates numerator and denominator coefficients of a general form controller and is not limited to chosen *PID* gains in terms of the *cost-function* selected, which could be a *GPC*, *NPGMV cost-function*, or any predictive type. The *RS* controller has a simplicity of concept and delivers the controller structure that may be more friendly to industry engineers. The designer can select a simple *PID* or a *low-order* transfer function structure with traditional tuning parameters. The *RS* controller will be responsible for computing the gains in background processing. These calculated gains will vary if disturbances or setpoint changes or varying, which is essential to sustain the solution optimality. It also adapts to uncertainty in the model, as if the model is different from that assumed, then the online gains will be different because the online gains depend on the measurements. The model and the real plant mismatch can be handled with the appropriate selection of error dynamic weighting terms P_c and dynamic control weighting F_{ck}.

Three application examples were investigated to prove the applicability and efficiency of the *RS* control's *state-space* design. In these applications, the *RS* controller demonstrated an improvement in performance without involving exhausting tuning procedures. In the first example in Section 7.2, the algorithm was tested on a *QTP* with high interaction between different loops. This *QTP* benchmarked the *RS* controller performance against the classical *PI* controller and assessed its disturbance rejection property.

In the second example in Section 7.3, the algorithm was assessed on *ETB* with a high nonlinearity due to *LH* spring and friction. This *ETB* was modelled in *qLPV* to capture the physical model nonlinearity. Compared to some automotive performance standards, such as the S80 Volvo recommended control responses, the *RS* controller performance was assessed for automotive performance standards.

In the third example in Section 7.4, the algorithm was assessed on identified *CSTR* in *PWA state-space*. This *PWA* model was converted to its equivalent *SD* representation. The *CSTR* model has two stable regions at both edges of the output span, divided by an unstable region. The *CSTR* was moved between the different *steady-states* by tracking the concentration setpoint throughout the operating regions to assess the optimal switching between operating points.

Finally, the fourth example in Section 7.5 was investigated to prove the *RS* control's polynomial formulation's practicality. The algorithm was tested on a nonlinear *VCT* that has been identified in *LPV* transfer function form. This *VCT* was used to evaluate the *RS* controller performance against the classical *PI* controller and assess the benefit of using the polynomial *LPV RS* controller to control automotive nonlinear systems.

The *RS* control has considerable potential based upon the following possible advantages:

- The *PID* controller is handy, and greater flexibility can be achieved by adding terms to its dynamic order.
- The controller parameterising in terms of linear dynamic functions set multiplied by unknown gains is intuitively justifiable, but a method is needed to compute the gains.
- Using predictive control as the basis of the optimisation method enables incorporating future reference and disturbance variations information into the design.
- The *RS* controller should have natural feedforward terms, provide some measure of *transport-delay* compensation, be easier to retune by classically trained engineers, and it has natural robustness properties.
- The additional *black-box* operator model that can be included results in the general linear plant description that is particularly appropriate for representing nonlinearities in actuators.
- An *LPV ARMAX* form of plant model will be suitable for many process applications where the parameters are often approximately constant for long periods and operating points slowly change.

- Engineers trained in classical methods can consider the discrete *ARMAX* model for conditions when the parameters are fixed. They can use their *frequency-domain* experience to assess likely plant behaviour in such conditions.

By blending the two most effective control techniques employed in industry, an industrial controller design method has been delivered with considerable potential. Of course, it does have a disadvantage over the usual *PID* control approach that a model is essential to the solution. There are also questions regarding the limitations imposed by *pre-specifying* a fixed controller structure.

Overall the *RS* controllers in their predictive version proved to introduce a promising solution for dealing with these particular nonlinear systems with many further improvement opportunities. The *cost-function* minimized is common and can involve *LPV* dynamic *cost-function* error weightings and *cost-weightings* on the magnitude of the controller gains and the rate of change of gains.

The *RS* controller was not established as an adaptive solution, but it has some adaptive controller features. Nevertheless, it is founded on a known linear or nonlinear model, and it does not have the unpredictability of, say, a *self-tuning* controller. From a research perspective, there are many features of this type of solution that still require assessment. However, it provides a *model-based* optimal controller from an industrial perspective, yet with traditional tuning knobs.

8.2 Future Work

In this section, two extensions to the original design are suggested. The first one is related to the *RS* control design. An online disturbance identification feature can be added by utilising tools found in recent control theory and merged into the *RS* control design. The latter suggestion is related to the process of accommodating more nonlinear system classes with different nonlinearities in the *RS* control algorithmic group.

8.2.1 Disturbance Identification

The proper disturbance rejection feature in any real system depends on the correct disturbance model. This disturbance model should have adequate flexibility to count for *low-frequency* disturbance or any pulsation peaks located possibly anywhere in the spectrum. Hence, a method that can instantly and successively construct an accurate dynamical model to track *time-varying* disturbances is desirable for the *RS* control strategy to mitigate disturbances. Throughout this thesis, the disturbance model was decided based on an intuitive way, resulting in a degraded disturbance rejection feature. Thus, an extension to the original design is suggested, which could potentially tackle this issue that may arise out of the uncertainty caused by a passive disturbance model. The disturbance model can be obtained and selected in the suggested approach, while the *RS* controller operates from an online identification process. This process can be carried out for both the *state-space* or the polynomial case by initially building a disturbance model and then acquiring meaningful physical model parameters.

8.2.2 Application Design

Two application examples are given for both the *state-space* and the polynomial description in the next to demonstrate how the *RS* algorithms can be advanced to accommodates various systems nonlinearity.

8.2.2.1 input-output qLPV identification

Usually, *input-output LPV* identification tends to be easier than the *state-space LPV* subspace identification when no first principles model is available. Still, most advanced controller design techniques are founded on *state-space* models. Thus, the *LPV input-output* representation is converted to its equivalent *state-space LPV* description, and this process is challenging and not uncomplicated. Some work has advanced the *qLPV input-output* system identification process, and the *MIMO LPV input-output* system identification toolbox starts to be available. The solution given in Chapter 6 can be utilized along with one of these toolboxes in solving some advanced control problems.

8.2.2.2 SD Constraints

The *MR semi-active* suspension has an extreme nonlinearity with hysteresis, viscous, and saturation constraint. Recent research proves that the *MR* clipped schema can have a reduced performance when the constraints are not deemed in the preferred damping force's design process. Some efforts, such as the complicated hybrid *MPC*, were used to handle the nonlinear damping force constraint. The solution given in Chapter 5 can be exploited to accelerate the hybrid *MPC* computation algorithms.

References

[1] L. Desborough and R. Miller, ‘Increasing customer value of industrial control performance monitoring - Honeywell’s experience’, in *AIChE Symposium Series*, 2002, vol. 98, no. 326, pp. 169–189.

[2] B. A. Ogunnaike and K. Mukati, ‘An alternative structure for next generation regulatory controllers: Part I: Basic theory for design, development and implementation’, *J. Process Control*, vol. 16, no. 5, pp. 499–509, 2006, doi: 10.1016/j.jprocont.2005.08.001.

[3] A. T. Bahill, ‘Simple Adaptive Smith-Predictor for Controlling Time-Delay Systems.’, *IEEE Control Syst. Mag.*, vol. 3, no. 2, pp. 16–22, May 1983, doi: 10.1109/MCS.1983.1104748.

[4] C. Meyer, D. E. Seborg, and R. K. Wood, ‘A comparison of the Smith predictor and conventional feedback control’, *Chem. Eng. Sci.*, vol. 31, no. 9, pp. 775–778, 1976, doi: 10.1016/0009-2509(76)80050-4.

[5] S. O. J. M, ‘Closer control of loops with dead-time’, *Chem. Eng. Prog.*, vol. 53, no. 5, pp. 217–219, 1957.

[6] S. O. J. M, *Feedback Control Systems*. McGraw-Hill Inc, 1958.

[7] H. Tore, ‘A Predictive PI Controller for Processes with Long Dead Times’, *IEEE Control Syst.*, vol. 12, no. 1, pp. 57–60, Feb. 1992.

[8] K. Uren and G. van, ‘Predictive PID Control of Non-Minimum Phase Systems’, in *Advances in PID Control*, InTech, 2011.

[9] K. J. Åström, *PID controllers*, 2nd ed.. Research Triangle Park, N.C.: Research Triangle Park, N.C. : International Society for Measurement and Control, 1995.

[10] D. E. Rivera, M. Morarl, and S. Skogestad, ‘Internal Model Control: Pid Controller Design’, *Ind. Eng. Chem. Process Des. Dev.*, vol. 25, no. 1, pp. 252–265, Jan. 1986, doi: 10.1021/i200032a041.

[11] J. Richalet, ‘Industrial applications of model based predictive control’, *Automatica*, vol. 29, no. 5, pp. 1251–1274, 1993.

[12] J. A. Rossiter and J. Richalet, ‘Handling constraints with predictive functional control of unstable processes’, *Proc. Am. Control Conf.*, vol. 6, pp. 4746–4751, 2002, doi: 10.1109/ACC.2002.1025409.

[13] J. Richalet, 'Commande prédictive', *Autom. ingénierie système, R7423*, pp. 1–17, Dec. 1997.

[14] D. Uduehi, A. Ordys, and M. J. Grimble, 'Multivariable PID controller deign using online generalised predictive control optimisation', *IEEE Conference on Control Applications & IEEE Conference on Computer Aided Control Systems Design*. Glasgow, 2002, doi: 10.1109/cca.2002.1040197.

[15] M. T. Khadir and J. V. Ringwood, 'Extension of first order predictive functional controllers to handle higher order internal models', *Int. J. Appl. Math. Comput. Sci.*, vol. 18, no. 2, pp. 229–239, 2008, doi: 10.2478/v10006-008-0021-z.

[16] W. Guo, W. Wang, and X. Qiu, 'An improved generalized predictive control algorithm based on PID', in *Proceedings - International Conference on Intelligent Computation Technology and Automation, ICICTA 2008*, 2008, vol. 1, pp. 299–303, doi: 10.1109/ICICTA.2008.210.

[17] I. L. Chien, 'IMC-PID controller design-an extension', *IFAC Proc. Ser.*, vol. 6, pp. 147–152, 1988, doi: 10.1016/S0005-1098(01)00170-4.

[18] M. Morari, *Robust Process Control.*, vol. 65, no. 6. Prentice Hall, 1987.

[19] M. Morari, *Advances in model-based predictive control*. Oxford University Press, 1994.

[20] Q. G. Wang, C. C. Hang, and X. P. Yang, 'Single-loop controller design via IMC principles', *Automatica*, vol. 37, no. 12, pp. 2041–2048, Dec. 2001, doi: 10.1016/S0005-1098(01)00170-4.

[21] K. K. Tan, T. H. Lee, S. N. Huang, and F. M. Leu, 'PID control design based on a GPC approach', *Ind. Eng. Chem. Res.*, vol. 41, no. 8, pp. 2013–2022, 2002, doi: 10.1021/ie010480i.

[22] M. H. Moradi, M. R. Katebi, and M. A. Johnson, 'Predictive PID control: a new algorithm', vol. 1. USA, pp. 764–769, 2001.

[23] K. J. Åström, *Introdution to stochastic control theory*. Mineola, N.Y.: Mineola, N.Y. : Dover Publications, 1979.

[24] P. Wellstead and M. Zarrop, *Self-Tuning Systems: Control and Signal Processing*. 1991.

[25] M. J. Grimble, 'GMV Control of Nonlinear Multivariable Systems', in *Optimal Control Applications and Methods*, 2004, no. August, pp. 3–5.

[26] D. W. Clarke and P. J. Gawthrop, 'Self-Tuning Control.', *Proc. Inst. Electr. Eng.*, vol. 126, no. 6, pp. 633–640, 1979, doi: 10.1049/piee.1979.0145.

[27] M. J. Grimble, 'Generalized minimum variance control law revisited', *Optim. Control Appl. Methods*, vol. 9, no. 1, pp. 63–77, Oct. 1988, doi: 10.1002/oca.4660090106.

[28] D. W. Clarke, C. Mohtadi, and P. S. Tuffs, 'Generalized predictive control-Part I. The basic algorithm', *Automatica*, vol. 23, no. 2, pp. 137–148, 1987, doi: 10.1016/0005-1098(87)90087-2.

[29] D. W. Clarke, C. Mohtadi, and P. S. Tuffs, 'Generalized Predictive Control-Part II Extensions and interpretations', *Automatica*, vol. 23, no. 2, pp. 149–160, 1987, doi: 10.1016/0005-1098(87)90088-4.

[30] D. W. Clarke, 'Application of generalized predictive control', *IFAC Proc. Ser.*, vol. 8, no. 6, pp. 1–8, Apr. 1989, doi: 10.1016/s1474-6670(17)53792-1.

[31] A. W. Ordys and D. W. Clarke, 'A state-space description for GPC controllers', *Int. J. Syst. Sci.*, vol. 24, no. 9, pp. 1727–1744, 1993, doi: 10.1080/00207729308949590.

[32] M. Grimble, 'Nonlinear generalized minimum variance feedback, feedforward and tracking control', *Automatica*, vol. 41, no. 6, pp. 957–969, 2005.

[33] T. Mitchell and M. L. Overton, 'Fixed low-order controller design and H∞ optimization for large-scale dynamical systems', *IFAC-PapersOnLine*, vol. 28, no. 14, pp. 25–30, 2015, doi: 10.1016/j.ifacol.2015.09.428.

[34] B. D. O. Anderson, 'Controller design: moving from theory to practice', *IEEE Control Syst.*, vol. 13, no. 4, pp. 16–25, Aug. 1993.

[35] M. J. Grimble, 'GMV control of non-linear continuous-time systems including common delays and state-space models', *Int. J. Control*, vol. 80, no. 1, pp. 150–165, 2007, doi: 10.1080/00207170600965497.

[36] M. J. Grimble and P. Majecki, 'State-space approach to nonlinear predictive generalised minimum variance control', *Int. J. Control*, vol. 83, no. 8, pp. 1529–1547, 2010, doi: 10.1080/00207171003736303.

[37] M. Grimble, *Industrial control systems design*. Wiley, 2001.

[38] M. Grimble and M. Johnson, *Optimal control and stochastic estimation : theory and applications*. Wiley, 1988.

[39] F. Bruzelius, ‘Linear Parameter-Varying Systems an approach to gain scheduling’, *Doktorsavhandlingar vid Chalmers Tekniska Hogskola*, no. 2076. 2004.

[40] M. Grimble and P. Majecki, ‘NGMV Control Using Unstable State-Dependent Multivariable Models’. 2016.

[41] W. J. Rugh and J. S. Shamma, ‘Research on gain scheduling’, *Automatica*, vol. 36, no. 10. pp. 1401–1425, 2000, doi: 10.1016/S0005-1098(00)00058-3.

[42] J. S. Shamma and M. Athans, ‘Guaranteed properties of gain scheduled control for linear parameter-varying plants’, *Automatica*, vol. 27, no. 3, pp. 559–564, 1991, doi: 10.1016/0005-1098(91)90116-J.

[43] A. Packard, ‘Gain scheduling via linear fractional transformations’, *Syst. Control Lett.*, vol. 22, no. 2, pp. 79–92, 1994, doi: 10.1016/0167-6911(94)90102-3.

[44] A. Kwiatkowski, *LPV Modeling and Application of LPV Controllers to SI Engines*. Verlag Dr. Hut, 2007.

[45] P. Ioannou and S. Baldi, ‘Robust adaptive control’, in *The Control Systems Handbook: Control System Advanced Methods, Second Edition*, Birkhäuser, Boston, MA, 2010, pp. 835–856.

[46] M. Nemani, R. Ravikanth, and B. A. Bamieh, ‘Identification of linear parametrically varying systems’, in *Proceedings of the IEEE Conference on Decision and Control*, 1995, vol. 3, pp. 2990–2995.

[47] B. Bamieh and L. Giarré, ‘Identification of linear parameter varying models’, *Int. J. Robust Nonlinear Control*, vol. 12, no. 9, pp. 841–853, Jul. 2002, doi: 10.1002/rnc.706.

[48] M. Butcher, A. Karimi, and R. Longchamp, ‘On the Consistency of Certain Identification Methods for Linear Parameter Varying Systems’, *IFAC Proc. Vol.*, vol. 41, no. 2, pp. 4018–4023, Jan. 2008.

[49] V. Laurain, M. Gilson, R. Tóth, and H. Garnier, ‘Refined instrumental variable methods for identification of LPV Box-Jenkins models’, *Automatica*, vol. 46, no. 6, pp. 959–967, Jun. 2010.

[50] H. Abbas and H. Werner, ‘An instrumental variable technique for open-loop and closed-loop identification of input-output LPV models’, in *2009 European Control Conference, ECC 2009*, Mar. 2014, pp. 2646–2651, doi: 10.23919/ecc.2009.7074805.

[51] R. Tóth, *Modelling and identification of linear parameter-varying systems*. Berlin: Berlin : Springer, 2010.

[52] R. Toth, H. S. Abbas, and H. Werner, 'On the state-space realization of lpv input-output models: Practical approaches', *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 1, pp. 139–153, 2012, doi: 10.1109/TCST.2011.2109040.

[53] E. Schulz, O. Janda, and M. Schultalbers, 'On LPV System Identification Algorithms for Input-Output Model Structures and their Relation to LTI System Identification∗', *IFAC-PapersOnLine*, vol. 51, no. 15, pp. 1098–1103, Jan. 2018, doi: 10.1016/j.ifacol.2018.09.047.

[54] P. Lopes dos Santos, T. P. Azevedo Perdicoúlis, C. Novara, J. A. Ramos, and D. E. Rivera, *Linear Parameter-Varying System Identification*. WORLD SCIENTIFIC, 2011.

[55] I. Gustavsson, L. Ljung, and T. Söderström, 'Identification of processes in closed loop-identifiability and accuracy aspects', *Automatica*, vol. 13, no. 1, pp. 59–75, 1977, doi: 10.1016/0005-1098(77)90009-7.

[56] H. Hjalmarsson, M. Gevers, and F. De Bruyne, 'For model-based control design, closed-loop identification gives better performance', *Automatica*, vol. 32, no. 12, pp. 1659–1673, 1996, doi: 10.1016/S0005-1098(96)80003-3.

[57] I. D. Landau, 'Identification in closed loop: A powerful design tool (better design models, simpler controllers)', *Control Eng. Pract.*, vol. 9, no. 1, pp. 51–65, 2001, doi: 10.1016/S0967-0661(00)00082-4.

[58] K. J. Åström and B. Wittenmark, 'On self tuning regulators', *Automatica*, vol. 9, no. 2, pp. 185–199, Mar. 1973, doi: 10.1016/0005-1098(73)90073-3.

[59] G. C. Goodwin, P. J. Ramadge, and P. E. Caines, 'Discrete-Time Multivariable Adaptive Control', *IEEE Trans. Automat. Contr.*, vol. 25, no. 3, pp. 449–456, Jun. 1980, doi: 10.1109/TAC.1980.1102363.

[60] R. R. Bitmead, M. Gevers, and V. Wertz, *Adaptive optimal control : the thinking man's GPC*. Prentice Hall, 1990.

[61] C. Beck, 'Coprime factors reduction methods for linear parameter varying and uncertain systems', *Syst. Control Lett.*, vol. 55, no. 3, pp. 199–213, Mar. 2006, doi: 10.1016/j.sysconle.2005.07.006.

[62] U. Shaked, 'A general transfer-function approach to the discrete-time steady-state linear quadratic gaussian stochastic control problem', *Int. J. Control*, vol. 29, no. 3, pp. 361–386, 1979, doi: 10.1080/00207177908922704.

[63] V. Kucera, *Discrete linear control: The polynomial equation approach*, vol. SMC-15, no. 4. J. Wiley, 2012.

[64] D. C. McFarlane and K. Glover, *Robust Controller Design Using Normalized Coprime Factor Plant Descriptions*, vol. 138. Berlin/Heidelberg: Springer-Verlag, 1990.

[65] W. H. Kwon and A. E. Pearson, 'A Modified Quadratic Cost Problem and Feedback Stabilization of a Linear System', *IEEE Trans. Automat. Contr.*, vol. 22, no. 5, pp. 838–842, Oct. 1977, doi: 10.1109/TAC.1977.1101619.

[66] J. Richalet, A. Rault, J. L. Testud, and J. Papon, 'Model Algorithmic Control of Industrial Processes.', 1977, pp. 103–120, doi: 10.1016/s1474-6670(17)69513-2.

[67] J. Richalet, A. Rault, J. L. Testud, and J. Papon, 'Model predictive heuristic control. Applications to industrial processes', *Automatica*, vol. 14, no. 5, pp. 413–428, 1978, doi: 10.1016/0005-1098(78)90001-8.

[68] C. R. Cutler and B. L. Ramaker, 'Dynamic Matrix Control - a Computer Control Algorithm.', *J. Environ. Sci. Heal. Part B Pestic. Food Contam. Agric. Wastes*, vol. 1, no. 17, p. 72, 1980, doi: 10.1109/JACC.1980.4232009.

[69] C. E. Garcia and A. M. Morshedi, 'Quadratic programming solution of dynamic matrix control (QDMC)', *Chem. Eng. Commun.*, vol. 46, no. 1–3, pp. 73–87, 1986, doi: 10.1080/00986448608911397.

[70] R. Findeisen and F. Allgower, 'An Introduction to Model Predictive Control', *J. Soc. Instrum. Control Eng.*, vol. 42, no. 4, pp. 310–312, 2003, doi: 10.11499/sicejl1962.42.310.

[71] J. Kocijan and R. Murray-Smith, 'Nonlinear predictive control with a Gaussian process model', in *Lecture Notes in Computer Science*, 2005, vol. 3355, pp. 185–200, doi: 10.1007/978-3-540-30560-6_8.

[72] S. J. Qin and T. A. Badgwell, 'A survey of industrial model predictive control technology', *Control Eng. Pract.*, vol. 11, no. 7, pp. 733–764, 2003, doi: 10.1016/S0967-0661(02)00186-7.

[73] A. Bemporad, M. Morari, and N. Ricker, 'Model Predictive Control Toolbox User's Guide', *The mathworks*, 2014.

[74] M. Herceg, 'Real-Time Explicit Model Predictive Control of Processes', Slovak University of Technology in Bratislava, 2009.

[75] S. Hovland, J. T. Gravdahl, and K. E. Willcox, 'Explicit model predictive control for large-scale systems via model reduction', *J. Guid. Control. Dyn.*, vol. 31, no. 4, pp. 918–926, Jul. 2008, doi: 10.2514/1.33079.

[76] M. J. Grimble, 'Restricted structure predictive optimal control', *Optim. Control Appl. Methods*, vol. 25, no. 3, pp. 107–145, 2004, doi: 10.1002/oca.741.

[77] M. J. Grimble, 'Optimal restricted structure control with prespecified gain or phase margins', *IEE Proc. Control Theory Appl.*, vol. 151, no. 3, pp. 271–277, 2004, doi: 10.1049/ip-cta:20040487.

[78] D. W. Clarke and C. Mohtadi, 'Properties of generalized predictive control', *Automatica*, vol. 25, no. 6, pp. 859–875, 1989.

[79] Y. Pang and M. J. Grimble, 'State dependent NGMV control of delayed piecewise affine systems', *Proceedings of the IEEE Conference on Decision and Control*. pp. 7192–7197, 2009, doi: 10.1109/CDC.2009.5399927.

[80] Y. Pang and M. J. Grimble, 'NGMV control of delayed piecewise affine systems', *IEEE Trans. Automat. Contr.*, vol. 55, no. 12, pp. 2817–2821, 2010, doi: 10.1109/TAC.2010.2069810.

[81] W. P. M. H. Heemels, J. M. Schumacher, and S. Weiland, 'Linear complementarity systems', *SIAM J. Appl. Math.*, vol. 60, no. 4, pp. 1234–1269, 2000, doi: 10.1137/S0036139997325199.

[82] A. J. Van Der Schaft and J. M. Schumacher, 'Complementarity modeling of hybrid systems', *IEEE Trans. Automat. Contr.*, vol. 43, no. 4, pp. 483–490, Apr. 1998, doi: 10.1109/9.664151.

[83] A. Bemporad and M. Morari, 'Control of systems integrating logic, dynamics, and constraints', *Automatica*, vol. 35, no. 3, pp. 407–427, 1999, doi: 10.1016/S0005-1098(98)00178-2.

[84] B. De Schutter, 'Optimal control of a class of linear hybrid systems with saturation', *SIAM J. Control Optim.*, vol. 39, no. 3, pp. 835–851, Jan. 2000, doi: 10.1137/S0363012999354648.

[85] E. D. Sontag, 'Nonlinear Regulation: The Piecewise Linear Approach', *IEEE Trans. Automat. Contr.*, vol. 26, no. 2, pp. 346–358, Apr. 1981.

[86] B. De Schutter and T. Van Den Boom, 'Model predictive control for max-plus-linear discrete event systems', *Automatica*, vol. 37, no. 7, pp. 1049–1056, 2001, doi: 10.1016/S0005-1098(01)00054-1.

[87] T. A. Henzinger, 'The Theory of Hybrid Automata', in *Verification of Digital and Hybrid Systems*, 2000, pp. 265–292.

[88] W. P. M. Heemels, B. De Schutter, and A. Bemporad, 'On the equivalence of classes of hybrid dynamical models', *Proc. IEEE Conf. Decis. Control*, vol. 1, pp. 364–369, 2001, doi: 10.1109/CDC.2001.980127.

[89] Y. Pang, W. L. Li, and H. Xia, 'Equivalent state-dependent models of linear hybrid automata', *Kongzhi Lilun Yu Yingyong/Control Theory Appl.*, vol. 30, no. 3, pp. 339–345, 2013, doi: 10.7641/CTA.2013.20345.

[90] Y. Pang and M. J. Grimble, 'NGMV control of delayed piecewise affine systems', *IEEE Trans. Automat. Contr.*, vol. 55, no. 12, pp. 2817–2821, 2010, doi: 10.1109/TAC.2010.2069810.

[91] F. Blanchini, 'Ultimate Boundedness Control for Uncertain Discrete-Time Systems via Set-Induced Lyapunov Functions', *IEEE Trans. Automat. Contr.*, vol. 39, no. 2, pp. 428–433, 1994, doi: 10.1109/9.272351.

[92] M. J. Grimble, P. Majecki, and L. Giovanini, 'Polynomial approach to nonlinear predictive GMV control', in *2007 European Control Conference, ECC 2007*, 2007, pp. 4546–4553, doi: 10.23919/ecc.2007.7068338.

[93] M. J. Grimble, P. Majecki, and L. Giovanini, 'Polynomial approach to nonlinear predictive GMV control', *2007 Eur. Control Conf. ECC 2007*, vol. 4, no. 3, pp. 4546–4553, 2007, doi: 10.23919/ecc.2007.7068338.

[94] M. J. Grimble and D. Biss, 'Selection of optimal control weighting functions to achieve good H-infinity robust designs', *Control. 1988. Control 88., Int. Conf.*, pp. 683–688, 1988.

[95] M. J. Grimble and P. Majecki, 'Nonlinear Control Law Design and Implementation', in *Nonlinear Industrial Control Systems*, Springer London, 2020, pp. 199–246.

[96] R. M. Miller, S. L. Shah, R. K. Wood, and E. K. Kwok, 'Predictive PID', *ISA Trans.*, vol. 38, no. 1, pp. 11–23, Jan. 1999.

[97] K. H. Johansson, 'The quadruple-tank process: A multivariable laboratory process with an adjustable zero', *IEEE Trans. Control Syst. Technol.*, vol. 8, no. 3, pp. 456–465, May 2000, doi: 10.1109/87.845876.

[98] Q. Saeed, V. Uddin, and R. Katebi, 'MIMO predictive PID control: A practical approach for quadruple tank', *J. Circuits, Syst. Comput.*, vol. 22, no. 3, p. 1350009, Mar. 2013, doi: 10.1142/S0218126613500096.

[99] J. Deur, D. Pavković, N. Perić, M. Jansz, and D. Hrovat, ‘An electronic throttle control strategy including compensation of friction and limp-home effects’, in *IEEE Transactions on Industry Applications*, 2004, vol. 40, no. 3, pp. 821–834, doi: 10.1109/TIA.2004.827441.

[100] L. Eriksson and L. Nielsen, ‘Non-linear model-based throttle control’, Mar. 2000, doi: 10.4271/2000-01-0261.

[101] D. Pavković, J. Deur, M. Jansz, and N. Perić, ‘Adaptive control of automotive electronic throttle’, *Control Eng. Pract.*, vol. 14, no. 2, pp. 121–136, Feb. 2006, doi: 10.1016/j.conengprac.2005.01.006.

[102] X. Yuan, Y. Wang, and L. Wu, ‘SVM-based approximate model control for electronic throttle valve’, *IEEE Trans. Veh. Technol.*, vol. 57, no. 5, pp. 2747–2756, Sep. 2008, doi: 10.1109/TVT.2008.917222.

[103] X. Yuan and Y. Wang, ‘A novel electronic-throttle-valve controller based on approximate model method’, *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 883–890, Mar. 2009, doi: 10.1109/TIE.2008.2004672.

[104] G. Panzani, M. Corno, and S. M. Savaresi, ‘On adaptive electronic throttle control for sport motorcycles’, *Control Eng. Pract.*, vol. 21, no. 1, pp. 42–53, Jan. 2013, doi: 10.1016/j.conengprac.2012.09.007.

[105] M. Vašak, M. Baotić, I. Petrović, and N. Perić, ‘Hybrid theory-based time-optimal control of an electronic throttle’, *IEEE Trans. Ind. Electron.*, vol. 54, no. 3, pp. 1483–1494, Jun. 2007, doi: 10.1109/TIE.2007.893060.

[106] X. Yuan and Y. Wang, ‘Neural networks based self-learning PID control of electronic throttle’, *Nonlinear Dyn.*, vol. 55, no. 4, pp. 385–393, Mar. 2009, doi: 10.1007/s11071-008-9371-1.

[107] W. Sheng and Y. Bao, ‘Fruit fly optimization algorithm based fractional order fuzzy-PID controller for electronic throttle’, *Nonlinear Dyn.*, vol. 73, no. 1–2, pp. 611–619, Jul. 2013, doi: 10.1007/s11071-013-0814-y.

[108] S. Zhang, J. J. Yang, and G. G. Zhu, ‘LPV gain-scheduling control of an electronic throttle with experimental validation’, in *Proceedings of the American Control Conference*, Jun. 2014, pp. 190–195, doi: 10.1109/ACC.2014.6859140.

[109] S. Zhang, J. J. Yang, and G. G. Zhu, ‘LPV Modeling and Mixed Constrained H2 H∞ Control of an Electronic Throttle’, *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 5, pp. 2120–2132, Oct. 2015.

[110] M. Vašak, I. Petrović, and N. Perić, ‘State estimation of an electronic throttle body’, in *Proceedings of the IEEE International Conference on Industrial Technology*, 2003, vol. 1, pp. 472–477, doi: 10.1109/icit.2003.1290368.

[111] P. Gaspar, I. Szaszi, and J. Bokor, ‘Active Suspension Design using LPV Control’, *IFAC Proc. Vol.*, vol. 37, no. 22, pp. 565–570, Apr. 2004, doi: 10.1016/s1474-6670(17)30403-2.

[112] C. Edwards, S. Spurgeon, and S. Spurgeon, *Sliding Mode Control*. CRC Press, 1998.

[113] R. N. K. Loh, T. Pornthanomwong, J. S. Pyko, A. Lee, and M. N. Karsiti, ‘Modeling, parameters identification, and control of an electronic throttle control (ETC) system’, in *2007 International Conference on Intelligent and Advanced Systems, ICIAS 2007*, Nov. 2007, pp. 1029–1035, doi: 10.1109/ICIAS.2007.4658541.

[114] M. Svensson, ‘Robust design of an electronic throttle controller’, Volvo Technological Development. Göteborg: Chalmers University of Technology, 1999.

[115] J. DU, C. SONG, and P. LI, ‘Modeling and Control of a Continuous Stirred Tank Reactor Based on a Mixed Logical Dynamical Model’, *Chinese J. Chem. Eng.*, vol. 15, no. 4, pp. 533–538, Mar. 2007.

[116] L. Özkan, M. V. Kothare, and C. Georgakis, ‘Control of a solution copolymerization reactor using multi-model predictive control’, *Chem. Eng. Sci.*, vol. 58, no. 7, pp. 1207–1221, Apr. 2003.

[117] P. B. Sistu and B. W. Bequette, ‘Nonlinear predictive control of uncertain processes: Application to a CSTR’, *AIChE J.*, vol. 37, no. 11, pp. 1711–1723, Nov. 1991, doi: 10.1002/aic.690371114.

[118] Y. Moriya, A. Watanabe, H. Uda, H. Kawamura, M. Yoshioka, and M. Adachi, ‘A newly developed intelligent variable valve timing system-continuously controlled cam phasing as applied to a new 3 liter inline 6 engine’, 1996, doi: 10.4271/960579.

[119] W. G. Wolber, ‘Sensors and Actuators.’, 1986, pp. 37–232.

[120] A. U. Genc, K. Glover, and R. Ford, ‘Nonlinear Control of Hydraulic Camshaft Actuators in Variable Cam Timing Engines’, *Cambridge Wolfson Coll. Univ. Cambridge*, 2002.

[121] Z. Ren and G. G. Zhu, 'Pseudo-random binary sequence closed-loop system identification error with integration control', *Proc. Inst. Mech. Eng. Part I J. Syst. Control Eng.*, vol. 223, no. 6, pp. 877–884, Sep. 2009, doi: 10.1243/09596518JSCE794.

[122] Z. Ren and G. G. Zhu, 'Integrated system identification and control design for an IC engine variable valve timing system', *ASME 2010 Dyn. Syst. Control Conf. DSCC2010*, vol. 1, no. 2, pp. 649–656, 2010, doi: 10.1115/DSCC2010-4038.

[123] A. White, Z. Ren, G. Zhu, and J. Choi, 'Mixed H2/H∞ observer-based LPV control of a hydraulic engine CAM phasing actuator', *ASME 2011 Dyn. Syst. Control Conf. Bath/ASME Symp. Fluid Power Motion Control. DSCC 2011*, vol. 2, no. 1, pp. 741–748, 2011, doi: 10.1115/DSCC2011-5937.

Appendices

A-1 Augmented System Matrices

Linear Subsystem

The given *state-space* system model in Chapter 2 is very common, where many different system models may be manipulated into the form given. To show how the most common problem may be written in the employed form the steps in going from the different subsystems to the augmented system are described.

Plant Model: The output *state-space* subsystem of the plant model is given in the next linear model form:

$$x_0(t+1) = A_0 x_0(t) + B_0 u_0 + D_0 \zeta(t) + G_0 d_{0d}(t) \tag{A.1}$$

$$y(t) = d_0(t) + C_0 x_0(t) + E_0 u_0(t-k) \tag{A.2}$$

where $x_0(t) \in R^{n_0}$ and $d_{0d}(t)$ is the input deterministic disturbance. The output disturbance $d_0(t) = d(t) + y_d(t)$ is represented by two components, the known deterministic disturbance component $d(t)$ and a second stochastic component $y_d(t)$.

Output Disturbance Model: The plant can contain process noise $\zeta(t)$ which can be deemed input disturbance, and likewise, an output disturbance model is used more for frequency response design shaping. This model is excited by *zero-mean white-noise* $\omega(t)$ and given in *state-equation* form as:

$$x_d(t+1) = A_d x_d(t) + B_d \omega(t), \quad x_d(t) \in R^{n_d} \tag{A.3}$$

$$y_d(t) = C_d x_d(t) \tag{A.4}$$

Error Weighting: As noted, the controlled signal involves the system weighted tracking error, and this weighting is given in *state-equation* form as:

$$x_p(t+1) = A_p x_p(t) + B_p \left(r(t) - y(t)\right), \quad x_p(t) \in R^{n_p} \tag{A.5}$$

$$e_p(t) = C_p x_p(t) + E_p \left(r(t) - y(t)\right) \tag{A.6}$$

Input Weighting: In some problems, it is beneficial to have another mechanism to introduce a control costing or a dynamic weighting on $u_0(t-k)$ input signal. If a dynamic input signal costing is to be introduced on the input, it can be treated as part of the system model, and this may require a *lead-lag* term type of frequency response, and hence the through term is needed in the subsystem model. Then the weighting:

$$x_r(t+1) = A_r x_r(t) + B_r u_0(t-k), \quad x_r(t) \in R^{n_r} \tag{A.7}$$

$$y_r(t) = C_r x_r(t) + E_r u_0(t-k) \tag{A.8}$$

Augmented System: The *state-space* model of a total $r \times m$ multivariable system illustrated in Figure 8-1 can be introduced. The augmented system *state-vector* is denoted $x(t) = [x_0^T(t) \quad x_d^T(t) \quad x_p^T(t) \quad x_r^T(t)]^T$ and combines augmented plant, disturbance, and dynamic *cost-function* weightings. The augmented system matrices are represented as A, B, C, D, E for the linear planet in (A.1) to (A.2).

$$x(t+1) = Ax(t) + Bu_0(t-k) + D\xi(t) + d_d(t) \tag{A.9}$$

$$y(t) = d(t) + Cx(t) + Eu_0(t-k) \tag{A.10}$$

$$z(t) = d(t) + Cx(t) + Eu_0(t-k) + v(t) \tag{A.11}$$

Weighted Error: The weighted error in terms of an augmented linear model is:

$$e_p(t) = d_p(t) + C_p x(t) + E_p u_0(t-k) \tag{A.12}$$

Remarks: Note that in some of the designs which follow, there is no need to assume the weighted errors number is the same as a number of control inputs, and this enables more general problems to be considered with additional *SD* weightings. If selected states are to be minimized, the mapping between states and signal to be minimized can be denoted $[C_{q0} \quad C_{qd} \quad 0 \quad 0]$. The augmented system equations that define matrices in (A.9)-(A.11) and the deterministic signal $d_d(t)$ are all derived in the next section, but the results may be summarized as:

$$A=\begin{bmatrix} A_0 & 0 & 0 & 0 \\ 0 & A_d & 0 & 0 \\ -B_pC_0 & -B_pC_d & A_p & 0 \\ 0 & 0 & 0 & A_r \end{bmatrix},\quad B=\begin{bmatrix} B_0 \\ 0 \\ -B_pE_0 \\ B_r \end{bmatrix},\quad D=\begin{bmatrix} D_0 & 0 \\ 0 & D_d \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$
$$G=\begin{bmatrix} G_0 & 0 \\ 0 & 0 \\ 0 & B_p \\ 0 & 0 \end{bmatrix},\quad C_p=\begin{bmatrix} -E_pC_0 & -E_pC_d & C_p & 0 \\ 0 & 0 & 0 & C_r \\ C_{q0} & C_{qd} & 0 & 0 \end{bmatrix},\quad E_p=\begin{bmatrix} -E_pE_0 \\ E_r \\ 0 \end{bmatrix} \tag{A.13}$$
$$C=\begin{bmatrix} C_0 & C_d & 0 & 0 \end{bmatrix},\quad E=E_0$$

Both reference $r(t)$ and disturbance $d(t)$ are deterministic signals. To simplify the analysis, define:

$$d_d(t)=G\begin{bmatrix} d_{0d}(t) \\ (r(t)-d(t)) \end{bmatrix}=\begin{bmatrix} G_0d_{0d}(t) \\ 0 \\ B_p\left(r(t)-d(t)\right) \\ 0 \end{bmatrix} \text{and } d_p(t)=\begin{bmatrix} E_p\left(r(t)-d(t)\right) \\ 0 \\ 0 \end{bmatrix} \tag{A.14}$$

The *white-noise* driving terms are merged in the vector $[\zeta^T(t) \quad \omega^T(t)]^T$.

Augmented System Matrices Definition: The relation between the augmented system in (A.9)-(A.13), to the plant, disturbance and weighting subsystems is formed below. To describe the matrices in the augmented system model, noting from (A.1)-(A.11):

$$\begin{aligned} x_p(t+1) &= A_p x_p(t) + \left(B_p r(t) - B_p y(t)\right) \\ y(t) &= d_0(t) + C_0 x_0(t) + E_0 u_0(t-k) \end{aligned}$$

Thence obtain,

$$x_p(t+1) = A_p x_p(t) + \left(B_p r(t) - B_p d(t) - B_p C_d x_d(t) - B_p C_0 x_0(t) - B_p E_0 u_0(t-k)\right)$$

Also, from equations (A.2) and (A.4),

$$\begin{aligned} e_p(t) &= C_p x_p(t) + E_p\left(r(t) - y(t)\right) \\ &= C_p x_p(t) + E_p r(t) - E_p\left(d(t) + C_d x_d(t) + C_0 x_0(t) + E_0 u_0(t-k)\right) \end{aligned} \qquad \text{(A.15)}$$

The system matrices are found from the combined *state-equation* models:

$$\begin{bmatrix} x_0(t+1) \\ x_d(t+1) \\ x_p(t+1) \\ x_r(t+1) \end{bmatrix} = \begin{bmatrix} A_0 & 0 & 0 & 0 \\ 0 & A_d & 0 & 0 \\ -B_p C_0 & -B_p C_d & A_p & 0 \\ 0 & 0 & 0 & A_r \end{bmatrix} \begin{bmatrix} x_0(t) \\ x_d(t) \\ x_p(t) \\ x_r(t) \end{bmatrix} + \begin{bmatrix} B_0 \\ 0 \\ -B_p E_0 \\ B_r \end{bmatrix} u_0(t-k) +$$

$$\begin{bmatrix} D_0 & 0 \\ 0 & D_d \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \zeta(t) \\ \omega(t) \end{bmatrix} + \begin{bmatrix} G_0 & 0 \\ 0 & 0 \\ 0 & B_p \\ 0 & 0 \end{bmatrix} \begin{bmatrix} d_{0d}(t) \\ (r(t) - d(t)) \end{bmatrix} \qquad \text{(A.16)}$$

Giving the augmented system matrices and vectors,

$$x(t+1) = Ax(t) + Bu_0(t-k) + D\xi(t) + d_d(t)$$

Following similar steps for the error equation, noting (A.15)

$$e_p(t) = \begin{bmatrix} E_p\left(r(t) - d(t)\right) \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -E_p C_0 & -E_p C_d & C_p & 0 \\ 0 & 0 & 0 & C_r \\ C_{q0} & C_{qd} & 0 & 0 \end{bmatrix} \begin{bmatrix} x_0(t) \\ x_d(t) \\ x_p(t) \\ x_r(t) \end{bmatrix} +$$

$$\begin{bmatrix} -E_p E_0 \\ E_r \\ 0 \end{bmatrix} u_0(t-k) \qquad \text{(A.17)}$$

Giving,

$$e_p(t) = d_p(t) + C_p x(t) + E_p u_0(t-k)$$

These outcomes validate the augmented matrices definition in (A.9), (A.12).

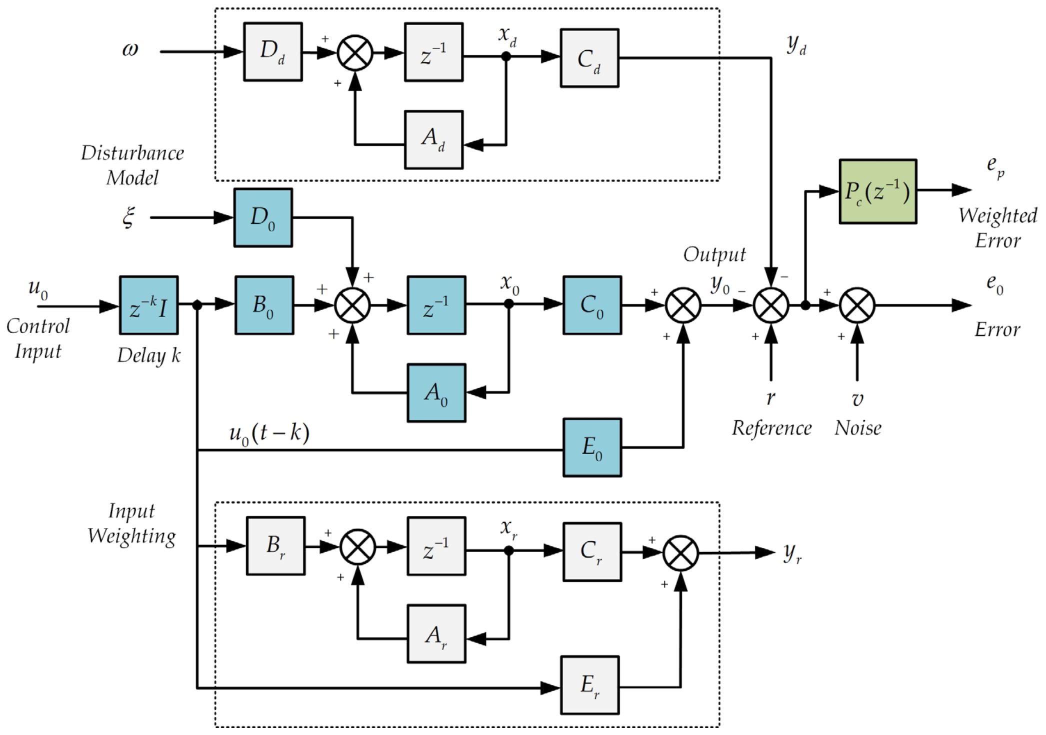

Figure 8-1: Augmented linear state-space subsystems

qLPV/SD Subsystem

The given *state-space* system model in Chapter 2 is very common, where many different system models may be manipulated into the form given. To show how the most common problem may be written in the employed form the steps in going from the different subsystems to the augmented system are described.

Plant Model: The output *state-space* subsystem of the plant model is given in the next with a minor extension to make these matrices to be functions of the system input at a time $t-k$ or even a *time-varying* set of parameters $\rho(t)$:

$$\begin{aligned} x_0(t+1) = A_0(x,\, u_0\,, \rho)x_0(t) + B_0(x,\, u_0\,, \rho)u_0(t-k) + \\ D_0(x,\, u_0\,, \rho)\zeta(t) + G_0(x, u_0\,,\, \rho)d_{0d}(t) \end{aligned} \tag{A.18}$$

$$y(t) = d_0(t) + C_0(x, u_0\,, \rho)\, x_0(t) + E_0(x, u_0\,, \rho)\, u_0(t-k) \tag{A.19}$$

where $x_0(t) \in R^{n_0}$ and $d_{0d}(t)$ is the input deterministic disturbance. The output disturbance $d_0(t) = d(t) + y_d(t)$ is represented by two components, the known deterministic disturbance component $d(t)$ and a second stochastic component $y_d(t)$.

Output Disturbance Model: The plant can contain process noise $\zeta(t)$ which can be deemed input disturbance, and likewise, an output disturbance model is used more for frequency response design shaping. This model is excited by *zero-mean white-noise* $\omega(t)$ and given in *state-equation* form as:

$$x_d(t+1) = A_d x_d(t) + B_d \omega(t), \quad x_d(t) \in R^{n_d} \tag{A.20}$$

$$y_d(t) = C_d x_d(t) \tag{A.21}$$

Error Weighting: As noted, the controlled signal involves the system weighted tracking error, and this weighting is introduced in *state-equation* as:

$$x_p(t+1) = A_p x_p(t) + B_p\left(r(t) - y(t)\right), \quad x_p(t) \in R^{n_p} \tag{A.22}$$

$$e_p(t) = C_p x_p(t) + E_p\left(r(t) - y(t)\right) \tag{A.23}$$

Augmented System: The *state-space* model of a total $r \times m$ *MIMO* system illustrated in Figure 8-2 can be introduced. The augmented system *state-vector* denoted $x(t) = [x_0^T(t) \quad x_d^T(t) \quad x_p^T(t)]^T$ combines augmented plant, disturbance, dynamic *cost-function* weightings and can be a function of the states, control or other parameters $\left(x(t), u_0(t-k), \rho(t)\right)$. The augmented system matrices will be denoted as A_t, B_t, C_t, D_t, E_t:

$$x(t+1) = A_t x(t) + B_t u_0(t-k) + D_t \xi(t) + d_d(t) \tag{A.24}$$

$$y(t) = d(t) + C_t x(t) + E_t u_0(t-k) \tag{A.25}$$

$$z(t) = d(t) + C_t x(t) + E_t u_0(t-k) + v(t) \tag{A.26}$$

Weighted Error: The weighted error in terms of an augmented *SD* model:

$$e_p(t) = d_p(t) + C_p x(t) + E_p u_0(t-k) \tag{A.27}$$

Augmented System Matrices Structure: The augmented system equations that explain the matrices in (A.24)-(A.26) and the deterministic signal $d_d(t)$ are all derived in the next section, but the results may be summarized as:

$$A = \begin{bmatrix} A_0(\rho) & 0 & 0 \\ 0 & A_d & 0 \\ -B_p C_0(\rho) & -B_p C_d & A_p \end{bmatrix}, \quad B = \begin{bmatrix} B_0(\rho) \\ 0 \\ -B_p E_0 \end{bmatrix}, \quad D_t = \begin{bmatrix} D_0(\rho) & 0 \\ 0 & D_d \\ 0 & 0 \end{bmatrix}$$
$$G = \begin{bmatrix} G_0(\rho) & 0 \\ 0 & 0 \\ 0 & B_p \end{bmatrix}, \quad \begin{matrix} C = \begin{bmatrix} C_0 & C_d & 0 \end{bmatrix}, \quad E_t = E_0, \quad E_{pt} = \begin{bmatrix} -E_p E_0 \end{bmatrix} \\ C_p = \begin{bmatrix} -E_p C_0(\rho) & -E_p C_d & C_p \end{bmatrix} \end{matrix} \tag{A.28}$$

Remarks: Subscript *t* on state matrices here is only utilized for the augmented system, and in a slight misuse of notation, it also suggests that these matrices are evaluated at time *t* so that the augmented system matrix at $t+1$ is given as A_{t+1} and so on. Both reference $r(t)$ and disturbance $d(t)$ are deterministic signals. To simplify the analysis, define:

$$d_d(t) = G_t \begin{bmatrix} d_{0d}(t) \\ (r(t)-d(t)) \end{bmatrix} \quad and \quad d_p(t) = E_p\left(r(t)-d(t)\right) \tag{A.29}$$

The *white-noise* driving terms are merged in the vector $[\zeta^T(t) \quad \omega^T(t)]^T$.

Augmented System Matrices Definition: A relation between the augmented system in (A.24)-(A.28), to the plant, disturbance and weighting subsystems is formed below. To describe the matrices in the augmented system model, noting from (A.18)-(A.26):

$$\begin{aligned} x_p(t+1) &= A_p x_p(t) + \left(B_p r(t) - B_p y(t)\right) \\ y(t) &= d_0(t) + C_0 x_0(t) + E_0 u_0(t-k) \end{aligned}$$

Thence obtain,

$$x_p(t+1) = A_p x_p(t) + \left(B_p r(t) - B_p d(t) - B_p C_d x_d(t) - B_p C_0 x_0(t) - B_p E_0 u_0(t-k)\right)$$

Also, from equations (A.19) and (A.21),

$$e_p(t) = C_p x_p(t) + E_p r(t) - E_p\left(d(t) + C_d x_d(t) + C_0 x_0(t) + E_0 u_0(t-k)\right) \tag{A.30}$$

Giving the augmented system matrices and vectors,

$$x(t+1) = A_t x(t) + B_t u_0(t-k) + D_t \xi(t) + d_d(t)$$

The system matrices are found from the combined *state-equation* models given in (A.31).

$$\underbrace{\begin{bmatrix} x_0(t+1) \\ x_d(t+1) \\ x_p(t+1) \end{bmatrix}}_{x(t+1)} = \underbrace{\begin{bmatrix} A_0(\rho) & 0 & 0 \\ 0 & A_d & 0 \\ -B_p C_0(\rho) & -B_p C_d & A_p \end{bmatrix}}_{A_t} \underbrace{\begin{bmatrix} x_0(t) \\ x_d(t) \\ x_p(t) \end{bmatrix}}_{x(t)} + \underbrace{\begin{bmatrix} B_0(\rho) \\ 0 \\ -B_p E_0 \end{bmatrix}}_{B_t} u_0(t-k) +$$
$$\underbrace{\begin{bmatrix} D_0(\rho) & 0 \\ 0 & D_d \\ 0 & 0 \end{bmatrix}}_{D_t} \begin{bmatrix} \zeta(t) \\ \omega(t) \end{bmatrix} + \underbrace{\begin{bmatrix} G_0(\rho) & 0 \\ 0 & 0 \\ 0 & B_p \end{bmatrix} \begin{bmatrix} d_{0d}(t) \\ \left(r(t)-d(t)\right) \end{bmatrix}}_{d_d(t)} \tag{A.31}$$

Following similar steps for the error equation, noting (A.30),

$$e_p(t) = \underbrace{\begin{bmatrix} E_p(r(t)-d(t)) \end{bmatrix}}_{d_p(t)} + \underbrace{\begin{bmatrix} -E_p C_0(\rho) & -E_p C_d & C_p \end{bmatrix}}_{C_{p_t}} \begin{bmatrix} x_0(t) \\ x_d(t) \\ x_p(t) \end{bmatrix} +$$
$$\underbrace{\begin{bmatrix} -E_p E_0 \end{bmatrix}}_{E_{p_t}} u_0(t-k) \tag{A.32}$$

Giving,

$$e_p(t) = d_p(t) + C_{pt} x(t) + E_{pt} u_0(t-k)$$

These outcomes validate the augmented matrices definition in (A.24), (A.27).

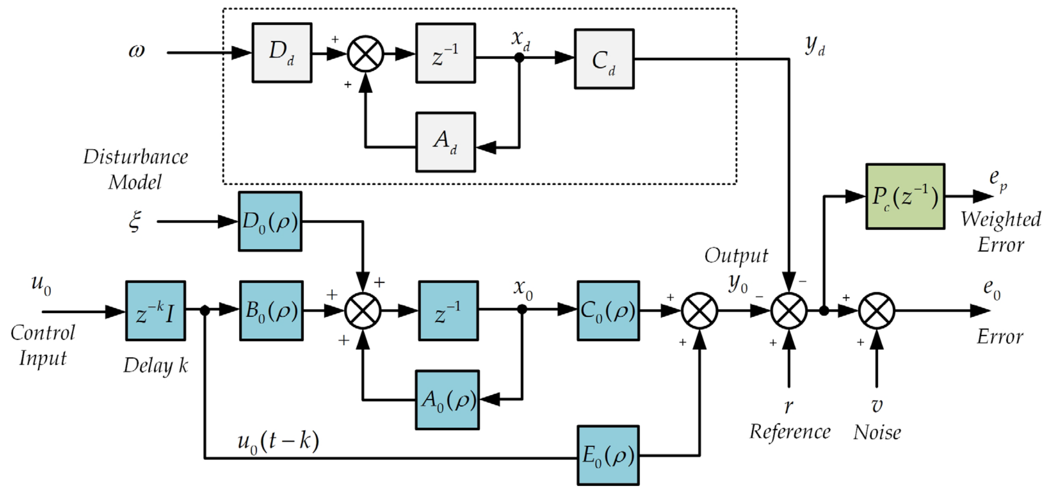

Figure 8-2: Augmented qLPV/SD state-space subsystems

A-2 Matlab Simulations Code

QTP

Main

```
clear all;clc;
global CpnAnU
global error1_t
global error2_t
error1_t=0;
error2_t=0;
CpnAnU=zeros(252,1);
addpath(genpath(pwd));
s = tf('s');
nbuffer = 41;                  % State buffer for future reference generation
Ts=0.5;                        % Sampling time
```

Model parameters

```
g = 981;
A1=28;A2=32;A3=28;A4=32;
a1=0.071;a2=0.057;a3=0.071;a4=0.057;
kc=0.5;
h1=12.4;h2=12.7;h3=1.8;h4=1.4;
k1=3.33;k2=3.35;miu1=0.7;miu2=0.6;
T1=(A1/a1)*sqrt((2*h1)/g);T2=(A2/a2)*sqrt((2*h2)/g);
T3=(A3/a3)*sqrt((2*h3)/g);T4=(A4/a4)*sqrt((2*h4)/g);
alpha1 = a1/A1;
alpha2 = a2/A2;
beta1 = miu1*k1/A1;
beta2 = miu2*k2/A2;
beta3 = (1-miu2)*k2/A3;
beta4 = (1-miu1)*k1/A4;
A=[-1/T1 0 A3/(A1*T3) 0;0 -1/T2 0 A4/(A2*T4);0 0 -1/T3 0;0 0 0 -1/T4];
B=[miu1*k1/A1 0;0 miu2*k2/A2;0 (1-miu2)*k2/A3;(1-miu1)*k1/A4 0];
C=[kc 0 0 0;0 kc 0 0];
E=zeros(2,2);
sys = ss(A,B,C,E);
sysd = c2d(sys,Ts);
[A0,B0,C0,E0] = ssdata(sysd);
```

Model dimensions

```
[nx0] = size(A0,1);                 % No of states,
[tmp,nu] = size(E0);                % No of inputs
nyc = tmp;                          % No of controlled signals
```

```
nym=nyc;                           % No of measured outputs
D0 = eye(nx0);                     % Planet stochastic disturbance
G0 = B0;                           % Planet deterministic disturbance
ng = size(D0,2);                   % No of stochastic input disturbances
nd = size(G0,2);                   % No of deterministic input disturbances
k = 0;                             % Explicit time delay [samples]
N = 40;                            % Prediction horizon
```

Initial conditions & Equilibrium points

```
ue_0 = [3;3];                           % Equilibrium input
xe_0 = [12.4;12.7;1.8;1.4];             % Equilibrium state
ye_0 = C*xe_0;                          % Equilibrium output
u_0 = [0;0];                            % Initial conditions
x_0 = [0;0;0;0];                        % Initial conditions
```

Disturbance Model

```
nxd = nym;
Ad = eye(nxd); Bd = eye(nxd);
Cd = eye(nxd);
```

Dynamic weighting

```
Kp1 = 1500;       Ki1 = 2;
Kp2 = 1500;       Ki2 = 1;
Pcc = [Kp1+Ki1/s, 0;0, Kp2+Ki2/s];
Pc = c2d(Pcc,Ts,'tustin');
[Ap,Bp,Cp,Ep] = ssdata(Pc);
nxp = size(Ap,1);
xp_0 = zeros(nxp,1);
```

Covariances for the Kalman Filter

```
QN = 0.01*diag([1 1 1 1 0 0]);   % Model noise covariance for Kalman filter
RN = 0*eye(nym);                 % Measurement noise covariance
```

GPC Controller Flags

```
        futuresp_flag = 1;                 % 1 = future reference knowledge ON
        futurectr_flag = 1;                % 0 = hold u(t-1), 1 = use U(t-1)
        bta = 0;                           % 0='u', 1='delta_u'
        Pu = [1 N];
        [Tu,Nu] = mpcprof(Pu,N,0);
        Nu=40
        TTu = kron(Tu,eye(nu));
        wR = 0.01*diag([1 1]);
        R = kron(eye((Nu)),wR);
        wQ = 10*diag([1 1]);
        Q = kron(eye((N)),wQ);
```

Integrator switch

```
nxi = bta*nu;
```

RS-GPC Parameters

```
Ac = [1 0 0 0;0 0 0 0;0 0 1 0;0 0 0 0];
Bc = [1 0;1 0;0 1;0 1];
Cc = [0 0 0 0;1 0 0 0;0 -1 0 0;0 0 0 0;0 0 1 0;0 0 0 -1];
Dc = [1 0;0 0;1 0;0 1;0 0;0 1];
L_K = 1*diag([1 1 1 1 1 1]);      % gain weighting ([P I D])
L_D = 0.0001*diag([1 1 1 1 1 1]); % gain rate of change weighting ([P I D])
% L_D = 1*ones(6,6);              % gain rate of change weighting ([P I D])
kc_bar = [0  0  0 0  0  0]';      % Single controller k=k_tilda
Ne=3;
```

Definition of Model and Control structures

```
nx = nx0+nxd+nxi+nxp;
par = struct(  'A0',A0,'B0',B0,'C0',C0,'E0',E0,'D0',D0,'G0',G0,...
               'Ad',Ad,'Bd',Bd,'Cd',Cd,'Ap',Ap,'Bp',Bp,'Cp',Cp,...
               'Ep',Ep,'nym',nym,'nyc',nyc,'nd',nd,'nu',nu,...
               'Ac',Ac,'Bc',Bc,'Cc',Cc,'Dc',Dc,'L_K',L_K,'L_D',L_D,...
               'kc_bar',kc_bar,'Ne',Ne,'nx0',nx0,'nxd',nxd,'nxp',nxp,...
               'nxi',nxi,'ng',ng,'nx',nx,'N',N,'QN',QN,'RN',RN,'Ts',Ts,...
               'k',k,'R',R,'Q',Q,'Nu',Nu,'Tu',Tu,'bta',bta,'TTu',TTu);
```

QT initialization (th0 defined above by selection)

```
t_stop =  300; t = [0:Ts:t_stop]'; th0=[0;0];
ref_Steps_1 = [0 th0(1)+5.2; 100 th0(1)+5.2; 200 th0(1)+7.2];
ref_Steps_2 = [0 th0(2)+6.35; 100 th0(2)+6.35; 200 th0(2)+6.35];
DD_In_1 =  [0 th0(1)+0; 50 th0(1)+0; 100 th0(1)+0];
DD_In_2 =  [0 th0(1)+0; 100 th0(1)+0; 200 th0(1)+0];
DD_Out_1 = [0 th0(1)+0; 50 th0(1)+1; 100 th0(1)+0];
DD_Out_2 = [0 th0(1)+0; 150 th0(1)+1; 200 th0(1)+1];
% References
reference_1 = ref_Steps_1;
reference_2 = ref_Steps_2;
open_system('RSGPC_PI')
```

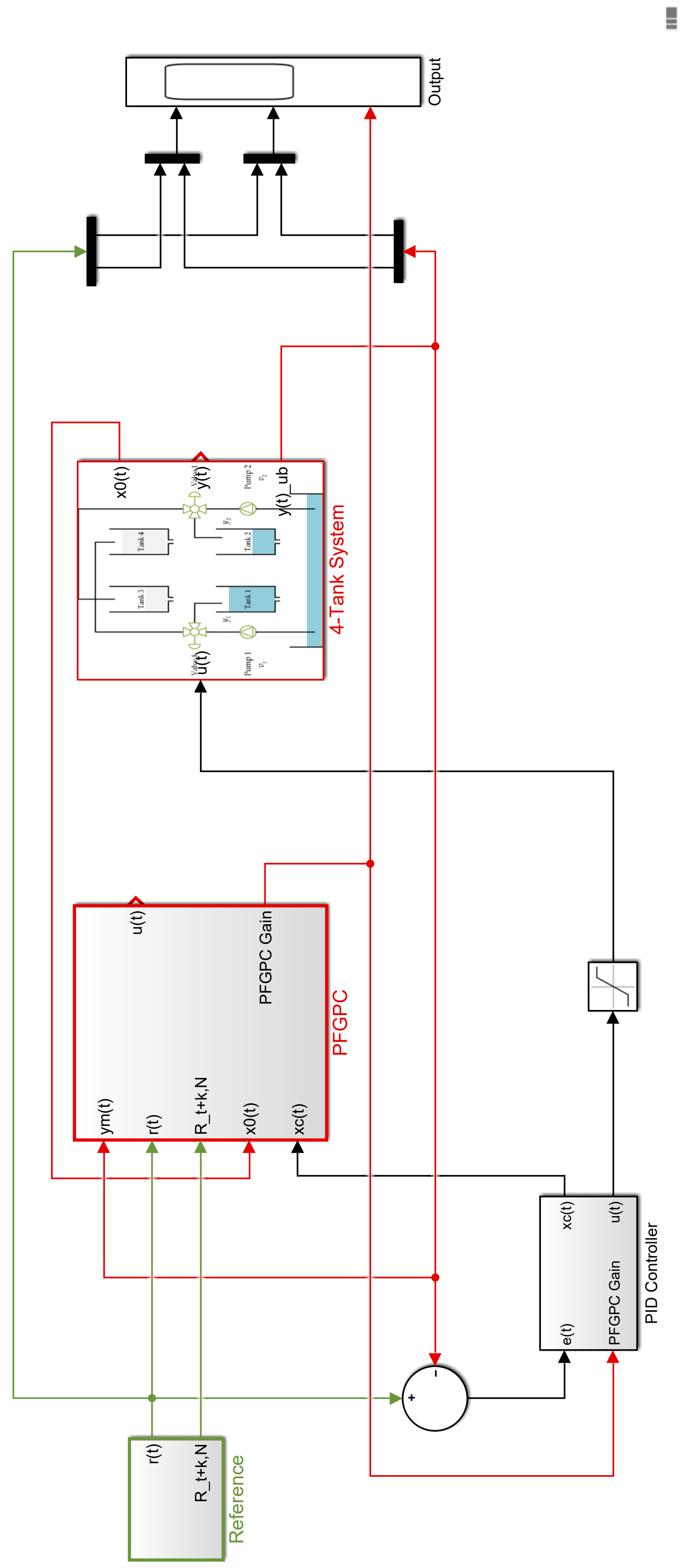

Figure 8-3: Main QTP Simulink diagram

ETB

Main

```
clear all;
clc;
addpath(genpath(pwd));
global CpnAnU
global error_t
global error_o1_t
error_o1_t=0;
error_t=0;
```

Constants

```
Ts = 10e-3;                       % [s] sample time
nbuffer = 100;                    % state buffer for future reference generation
s = tf('s');
```

Initial conditions

```
th0  = 15;                        % Limp-home (LH) position (Deg)
th0r = th0*(pi/180);              % Limp-home (LH) position (Rad)
thd0 = 0;                         % Plate speed at (LH)
x_0 = [th0*(pi/180); thd0];       % initial state: [position; velocity]
u_0 = 0;                          % initial control: torque
y_0 = 180/pi*x_0(1);
```

ETB plant parameters

```
R=1.15;             % Motor Resistance
R1=5.15
L=1.5e-3;           % Motor Inductance
Ng=20.68;           % Gear Ratio
Jt=0.0021;          % Motor throttle assembly inertia
Ka=0.0185;          % Motor torque coefficient
Km=0.0185;          % Back-EMF coefficient
Kft=0.0088;         % Viscous friction coefficient
Ksp=0.087;          % Spring stiffness coefficient
Tf=0.284;           % Friction torque
Tsp=0.396;          % spring torque
output_bias = 0;  % bias on plant output
parSYS = struct('R',R,'Ng',Ng,'Jt',Jt,'Ka',Ka,'Km',Km,'Kft',Kft,...
                'Ksp',Ksp,'Tf',Tf,'Tsp',Tsp,'th0r',th0r,'th0',th0);
```

State-dependent model dimensions

```
nx0 = 2;
nu = 1;
nym = 1;         % No of measured outputs
nyc = 1;                      % No of controlled signals
nd = 1;                       % No of input disturbances
k = 1;                        % Explicit time delay [samples]
```

Torque limits [Nm]

```
u_max = 12;
u_min = -u_max;
% Torque rate limits [Nm/s]
du_max = 12;
du_min = -du_max;
```

NGPC controller design

```
        c2dflag = 0;            % c2d flag: 0=Euler, 1=direct
        constr_flag = 0;        % Constraint handling
        freeze_flag = 0;        % 0 = full LPV, 1 = frozen model
        futuresp_flag = 1;      % 1 = future reference knowledge ON
        futurectr_flag = 1;     % 0 = hold u(t-1), 1 = use U(t-1)
        bta = 0;                % 0='u', 1='delta_u'
        N = 15;                 % prediction horizon
                                   Pu = [1 N];
                                   [Tu,Nu] = mpcprof(Pu,N,bta);
                                   Kp = 10;           Ki = 50;        Td = 0;
                   tau_d = 0.3;
        % NGPC Dynamic error weighting
            Co_CT = pid2tf(Kp,Ki,Td);
            Coo = c2d(Co_CT,Ts,'tustin'); Co1.v = 'z^-1';
            Pc_NGPC = Coo;
                                   L_U = 0.1;
        % Block-diagonal static control weight
        LN_U = kron(eye(Nu),L_U);
        % Output disturbances incl. stochastic and "robustness" states
        nxd = nym;
        Ad = eye(nxd); Bd = eye(nxd);
        Cds = zeros(nxd,nxd);      % stochastic disturbances
        Cdm = 0*eye(nxd);          % mismatch on measurements
        Cdc = 0;                   % mismatch on controlled variables
% Integrator switch
        nxi = bta*nu;
        Pc = ss(Pc_NGPC);    [Ap,Bp,Cp,Ep] = ssdata(Pc);
        nxp = size(Ap,1);
        Pcx_NGPC = ss(Ap,Bp,eye(nxp),zeros(nxp,nyc),Ts);
        xp_0 = zeros(nxp,1);
% Total number of states
        nx = nx0 + nxd + nxi + nxp;
% Covariances for the Kalman Filter
        qn_theta = 10^2;        % theta state uncertainty [rad^2]
        qn_thetadot = 10^2;     % theta_dot state uncertainty [(rad/s)^2]
        qn_omega = 10^2;
        rn_theta = 0.001^2;     % theta sensor variance [deg^2]
        QN = diag([qn_theta qn_thetadot qn_omega]);
        RN = diag([rn_theta]);
% Definition of Model and Control structures
        parNG = struct('R',R,'Jt',Jt,'Ka',Ka,'Km',Km,'Kft',Kft,...
                       'Ksp',Ksp,'Tf',Tf,'Tsp',Tsp,'th0r',th0r,...
                       'th0',th0,'Ng',Ng,'nym',nym,'nyc',nyc,'nd',nd,...
                       'nu',nu,'nx0',nx0,'nxd',nxd,'nxp',nxp,'nx',nx,...
                       'nxi',nxi,'freeze_flag',freeze_flag,...
                       'constr_flag',constr_flag,'c2dflag',c2dflag,...
                       'futuresp_flag',futuresp_flag,...
                       'futurectr_flag',futurectr_flag,'N',N,'Nu',Nu,...
                       'Tu',Tu,'QN',QN,'RN',RN,'LN_U',LN_U,'Ap',Ap,...
                       'Bp',Bp,'Cp',Cp,'Ep',Ep,'Ad',Ad,'Bd',Bd,...
                       'Cds',Cds,'Cdm',Cdm,'Cdc',Cdc,'Cd',Cds+Cdm,...
                       'u_max',u_max,'u_min',u_min,'du_max',du_max,...
                       'du_min',du_min,'Ts',Ts,'k',k,'bta',bta);
```

ETB initialization

```
t_stop =  15; t = [0:Ts:t_stop]';
ref_Steps = [0 (th0+10)*pi/180; 1 (th0+30)*pi/180; 4 (th0+50)*pi/180;...
             7 (th0+70)*pi/180; 10 (th0+40)*pi/180; 13 (th0+20)*pi/180];
OL_Steps = [0 0;4 -10;7 -20];
open_system('ETB')
% Reference
reference = ref_Steps;
% Simulation w/o Disturbance
kdist = 0;   x_00 = 0;
% Default simulation scenario
default_scenario = 'Steps';
% Default controller
default_controller = 'NGPC';
% Default model
default_model = 'qLPV';
```

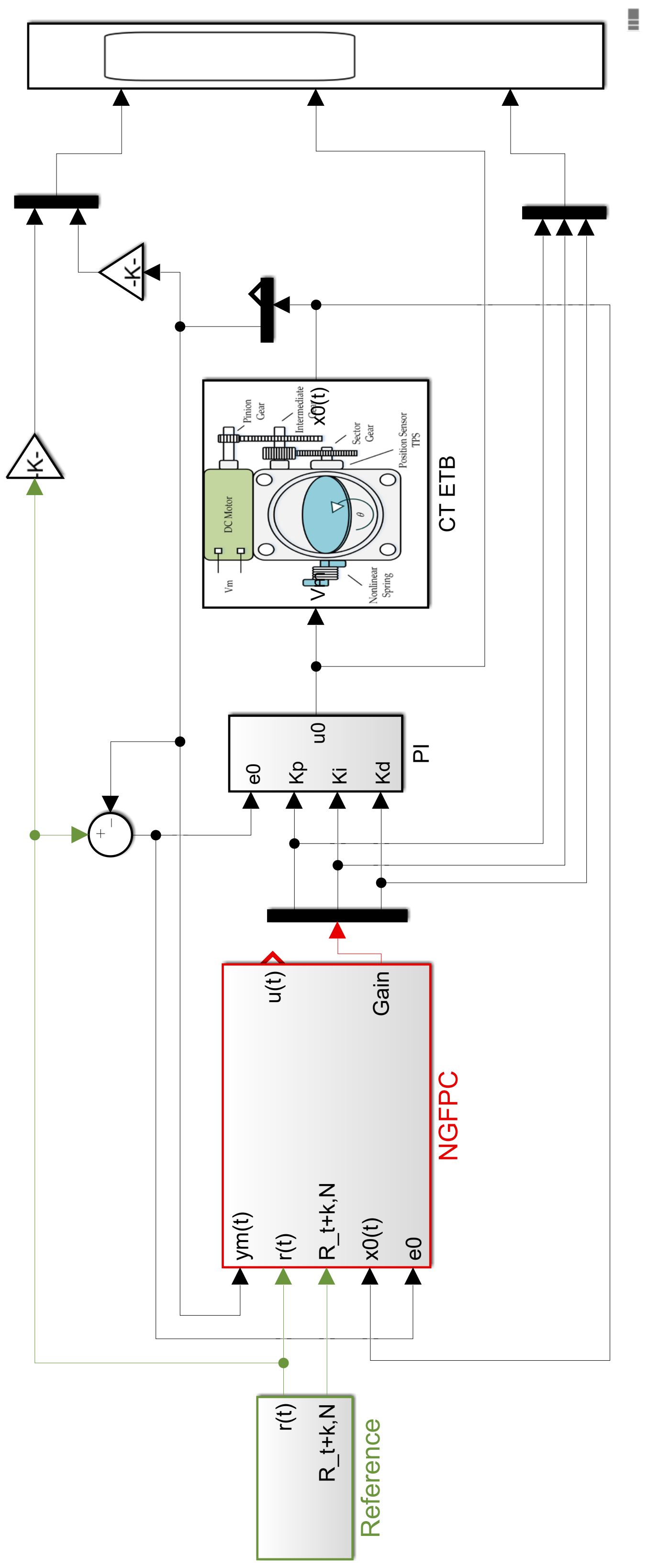

Figure 8-4: Main ETB Simulink diagram

CSTR

Main

```
clear all;
clc;
addpath(genpath(pwd));
global CpnAnU
global error_t
global error_o1_t
error_o1_t=0;
error_t=0;
```

Constants

```
Ts = 0.1;                          % [s] sample time
nbuffer = 6;                       % state buffer for future reference generation
s = tf('s');
```

Initial conditions

```
ca0  = 0.856;
ca_r = 0;
Temp0 = 0.886;
x_0 = [ca0; Temp0];  % initial state
u_0 = 0;             % initial control
y_0 = 0;
```

CSTR plant parameters

```
delta=0.3;          % Heat Transfer Coefficient
lambda=20;          % Dimensionless Activation Energy
theta=0.072;        % Damköhler number
beta=8;             % Heat of Reaction Coefficient
q=1;                %
output_bias = 0;    % bias on plant output
parSYS = struct('lambda',lambda,'delta',delta,'theta',theta,...
                'beta',beta,'q',q);
```

State-dependent model dimensions

```
nx0 = 2;            % No of states
nu = 1;             % No of control input
nym = 1;            % No of measured outputs
nyc = 1;            % No of controlled signals
nd = 1;             % No of input disturbances
k = 1;              % Explicit time delay [samples]
```

Control limits

```
u_max = 2;
u_min = -u_max;
% input rate limits
du_max = 2;
du_min = -du_max;
```

NGPC controller design

```
        c2dflag = 0;               % c2d flag: 0=Euler, 1=direct
        constr_flag = 1;           % Constraint handling
        freeze_flag = 0;           % 0 = full Sd, 1 = frozen model
        futuresp_flag = 1;         % 1 = future reference knowledge ON
        futurectr_flag = 1;        % 0 = hold u(t-1), 1 = use U(t-1)
        bta = 0;                   % 0='u', 1='delta_u'
        N = 5;                     % prediction horizon
                                   Pu = [1 N];
                                   [Tu,Nu] = mpcprof(Pu,N,bta);
                                   Kp = 100;          Ki =1;         Td = 0;
                   tau_d = 0;
        % NGPC Dynamic error weighting
            Co_CT = pid2tf(Kp,Ki,Td);
            Coo = c2d(Co_CT,Ts,'tustin'); Co1.v = 'z^-1';
            Pc_NGPC = Coo;
                                   L_U = 0.1;
% Block-diagonal static control weight
LN_U = kron(eye(Nu),L_U);
% Output disturbances incl. stochastic and "robustness" states
nxd = nym;
Ad = eye(nxd); Bd = eye(nxd);
Cds = zeros(nxd,nxd);              % stochastic disturbances
Cdm = 0*eye(nxd);                  % mismatch on measurements
Cdc = 0;                           % mismatch on controlled variables
% Integrator switch
nxi = bta*nu;
Pc = ss(Pc_NGPC);    [Ap,Bp,Cp,Ep] = ssdata(Pc);
nxp = size(Ap,1);    Pcx_NGPC = ss(Ap,Bp,eye(nxp),zeros(nxp,nyc),Ts);
xp_0 = zeros(nxp,1);
% Total number of states
nx = nx0 + nxd + nxi + nxp;
% Covariances for the Kalman Filter
qn_ca = 0.1^2;                     % ca state uncertainty [rad^2]
qn_Temp = 10^2;                    % Temp state uncertainty [(rad/s)^2]
qn_omega = 10^2;
rn_ca = 0.001^2;                   % sensor variance [deg^2]
QN = diag([qn_ca qn_Temp qn_omega]);
RN = diag([rn_ca]);
% Definition of Model and Control structures
parNG = struct('lambda',lambda,'delta',delta,'beta',beta,'q',q,...
    'ca_r',ca_r,'u_0',u_0,'x_0',x_0,'theta',theta,...
    'nym',nym,'nyc',nyc,'nd',nd,'nu',nu,'nx0',nx0,'nxd',nxd,'nxp',nxp,...
    'nx',nx,'nxi',nxi,'freeze_flag',freeze_flag,...
    'constr_flag',constr_flag,'c2dflag',c2dflag,...
```

```
                    'futuresp_flag',futuresp_flag,'futurectr_flag',futurectr_flag,...
                    'N',N,'Nu',Nu,'Tu',Tu,'QN',QN,'RN',RN,'LN_U',LN_U,...
                    'Ap',Ap,'Bp',Bp,'Cp',Cp,'Ep',Ep,...
                    'Ad',Ad,'Bd',Bd,'Cds',Cds,'Cdm',Cdm,'Cdc',Cdc,'Cd',Cds+Cdm,...
                    'u_max',u_max,'u_min',u_min,'du_max',du_max,'du_min',du_min,...
                    'Ts',Ts,'k',k,'bta',bta);
```

CSTR initialization

```
t_stop =  250; t = [0:Ts:t_stop]';
ref_Steps = [0 0.856; 50 0.5528; 100 0.2353; 150 0.5528; 200 0.856];
OL_Steps = [0 0;4 -10;7 -20];
% Reference
reference = ref_Steps;
% Simulation w/o Disturbance
kdist = 0;   x_00 = 0;
open_system('CSTR')
```

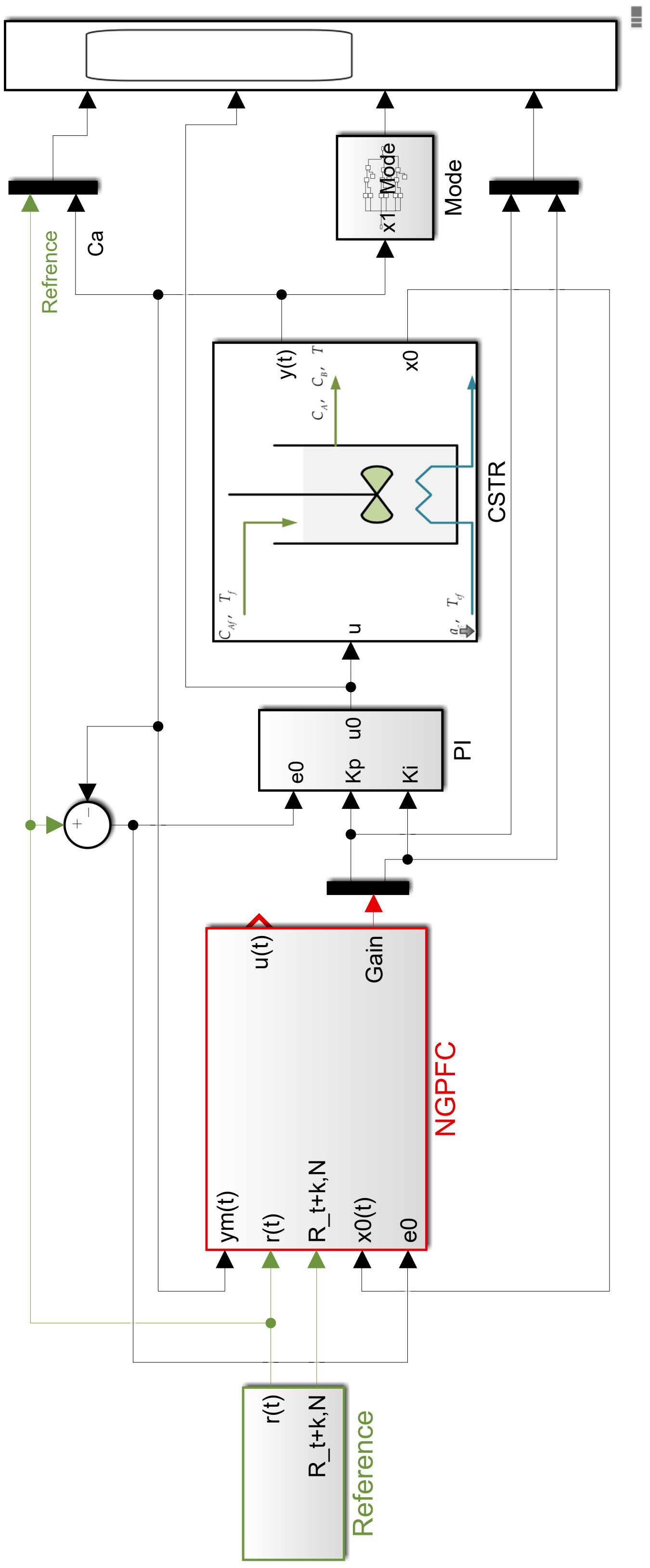

Figure 8-5: Main CSTR Simulink diagram

VCT

Main

```
%%Master script for VVT control with Polynomial GPFC Controller
clear all; close all; clc
addpath(genpath('polynomial'))
addpath(genpath('ngmv_toolbox_june_2019'))
pinit;           gensym 'z^-1';   clc
```

Initials

```
p0 = 310;                    % oil pressure [kPa]
eN0 = 900;                   % engine speed [rpm]
noVp = 2;                    % number of vaying parameters
pv = 0.7;                    % varying parameter [OL gain]
y0 = 0;                      % VVT actuator position [Deg.]
u_0 = 0;                     % VVT actuator control [V]
% pi = 0.2 + 0.1/s;          % PI for comparison
```

Main parameters

```
Ts = 0.005;                  % sample time [sec]
tol = 1e-8;                  % zeroing tolerance
zi = filt([0 1],1,Ts);
```

Polynomial System Models

```
k = 1;
num = [0.06013  -0.04263];
den = [1  -1.9547  +0.9553];
H = tf(num,den,Ts);          %VVT Open-loop transfer
set(H,'Variable','z^-1')
W0k = H/zi; W0 = H;
Nu=1;   Ny=1;
% Disturbance model Wd
numd=[0.01];
dend=[0.1 -0.1];
wd=tf(numd,dend,Ts);
set(wd,'Variable','z^-1');
Wds =wd;
Wd = Wds/zi;
% Reference model Wr
Wr = 0;
Nd = size(Wd,2);
QN = 0.1*eye(Nd);                  RN = 1e-3*eye(Ny);                  Rf = RN;
```

Dynamic weighting definition

```
Kp =1;              Ki =0;         Td = 0;           tau_d = 0;
Co_CT = pid2tf(Kp,Ki,Td);
Coo = c2d(Co_CT,Ts,'zoh'); Co1.v = 'z^-1';
%set(Coo,'Variable','z^-1')
Pc = Coo;
```

Setting

```
stop_time = 3.5;
Tf = stop_time;
tmp = [Ts,0,0.5,-6,2,0];    % CAM reference [deg]
tmp_N = [Ts,1800];          % engine speed reference [rpm]
tmp_P = [Ts,414];           % oil pressure reference[kPa]
reference = siggen(Tf,{'steps',tmp},Ts);
reference_N = siggen(Tf,{'steps',tmp_N},Ts);
reference_P = siggen(Tf,{'steps',tmp_P},Ts);
```

GPFC

```
N = 15;
lambda = 0.8; LAMBDA=lambda*eye(N+1);
C_I0 = [eye(Nu) zeros(Nu,Nu*(N))];
ref_N = ref_pred(reference,N,k,Pc);
par = struct('C_I0',C_I0,'W0k',W0k,'Wd',Wd,...
    'p0',p0,'eN0',eN0,'noVp',noVp,'pv',pv,'y0',y0,'u_0',u_0,'Ts',Ts,...
    'k',k,'Nd',Nd,'QN',QN,'Rf',Rf,'N',N,'LAMBDA',LAMBDA,'Nu',Nu,...
    'Ny',Ny,'Pc',Pc);

[H_NZ,S_NZ,GN] = ngpcpoly(W0k,Wd,k,Pc,Rf,N,Nu,Ny,Ts);
```

Simulink

```
open_system('VCT')
```

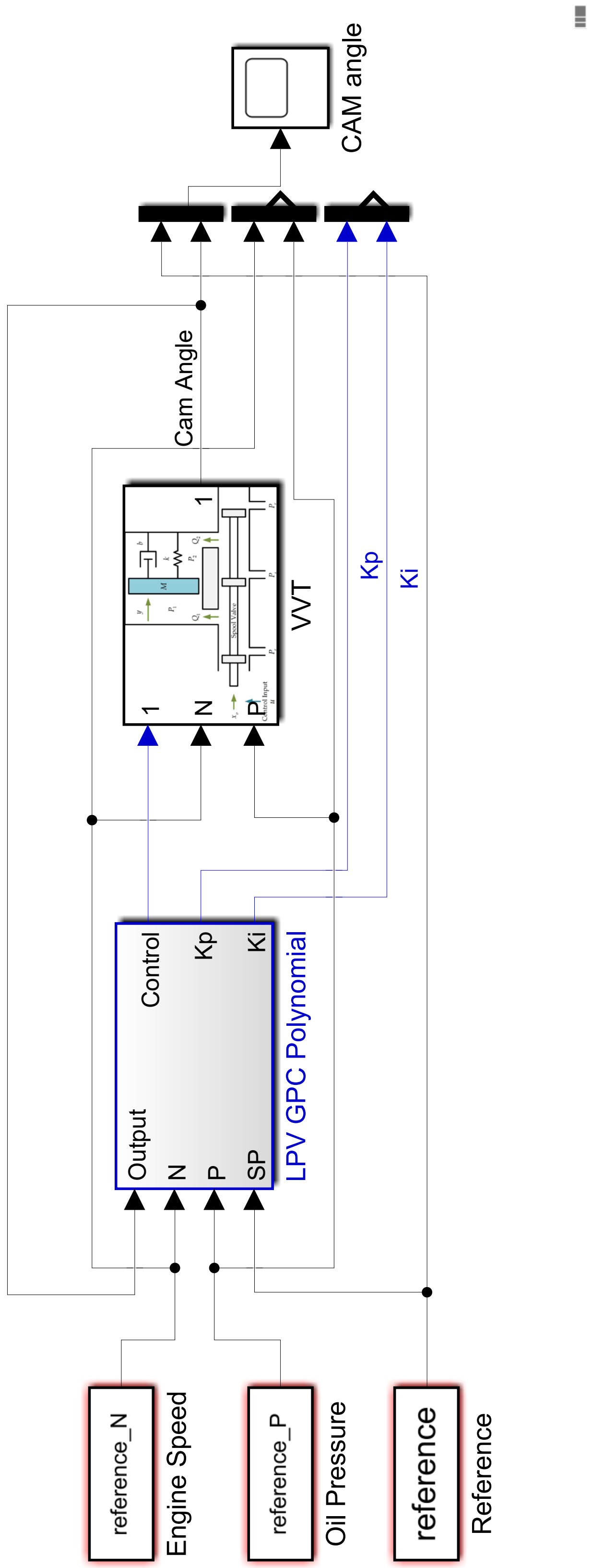

Figure 8-6: Main VCT Simulink diagram