DISSERTATION

PERFECT TRACKING FOR NON-MINIMUM PHASE SYSTEMS WITH APPLICATIONS TO BIOFUELS FROM MICROALGAE

Submitted by

Michael R. Buehner

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

for the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer 2010

COLORADO STATE UNIVERSITY

July 12, 2010

WE HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER OUR SUPERVISION BY MICHAEL R. BUEHNER ENTITLED PERFECT TRACKING FOR NON-MINIMUM PHASE SYSTEMS WITH APPLICATIONS TO BIOFUELS FROM MICROALGAE BE ACCEPTED AS FULFILLING IN PART REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY.

Committee on Graduate Work

Edwin K.P. Chong

Louis L. Scharf

Charles W. Anderson

Adviser: Peter M. Young

Department Head: Anthony A. Maciejewski

ABSTRACT OF DISSERTATION

PERFECT TRACKING FOR NON-MINIMUM PHASE SYSTEMS WITH APPLICATIONS TO BIOFUELS FROM MICROALGAE

In a causal setting, a closed-loop control system receives reference inputs (with no a priori knowledge) that it must track. For this setting, controllers are designed that provide both stability and performance (e.g., to meet tracking and disturbance rejection requirements). Often, feedback controllers are designed to satisfy weighted optimization criteria (e.g., weighted tracking error) that are later validated using test signals such as step responses and frequency sweeps. Feedforward controllers may be used to improve the response to measurable external disturbances (e.g., reference inputs). In this way, they can improve the closed-loop response; however, these approaches do not directly specify the closed-loop response.

Two controller architectures are developed that allow for directly designing the nominal closed-loop response of non-minimum phase systems. These architectures classify both the signals that may be perfectly tracked by a non-minimum phase plant and the control signals that provide this perfect tracking. For these architectures, perfect tracking means that the feedback error is zero (for all time) in the nominal case (i.e., the plant model is exact) when there are no external disturbances. For the controllers presented here, parts of the feedforward controllers are based on the plant model, while a separate piece is designed to provide the desired level of performance. One of the potential limitations to these designs is that the actual performance will depend upon the quality of the model used. Robustness tools are developed that may be used to determine the expected performance for a given level

of model uncertainty. These robustness tools may also be used to design the piece of the feedforward controller that provides performance. There is a tradeoff between model uncertainty and achievable performance. In general, more model uncertainty will result in less achievable performance.

Another way to approach the issue of performance is to consider that a good model must either be known a priori or learned via adaptation. In the cases where a good model is difficult to determine a priori, adaptation may be used to improve the models in the feedforward controllers, which will, in turn, improve the performance of the overall control system. We show how adaptive feedforward architectures can improve performance for systems where the model is of limited accuracy.

An example application of growing microalgae for biofuel production is presented. Microalgae have the potential to produce enough biofuels to meet the current US fuel demands; however, progress has been limited (in some part) due to a lack of appropriate models and controllers. In the work presented here, models are developed that may be used to monitor the productivity of microalgae inside a photobioreactor and to develop control algorithms. We use experimental data from a functional prototype photobioreactor to validate these models and to demonstrate the advantages of the advanced controller architectures developed here.

Michael R. Buehner
Department of Electrical and Computer Engineering
Colorado State University
Fort Collins, Colorado 80523
Summer 2010

ACKNOWLEDGMENTS

This has been an interesting and challenging journey over the past six years, and I have many people to thank. First of all, I would like to thank my adviser Peter Young for sharing his brilliance with me and for his patience and guidance over the years. This would not have been possible without him.

I would like to thank my masters adviser Louis Scharf for providing me with a solid understanding of signals and systems, guidance as an instructor, and for opening up my mind to the world of research. He has also always helped me out when I needed it (including serving on my committee at the last second).

I would like to thank my other committee members Chuck Anderson and Edwin Chong for all that they have taught me and for their input into my research. Also, I would like to thank Doug Hittle for being a committee member and collaborator for the first part of this project.

I have had the chance to work some amazing graduate students throughout the years. The list would be too long to include everyone, but I would particularly like to thank Keith and Dave for their interesting and stimulating conversions.

I spent about two years working at industry jobs, which I greatly enjoyed. I would like to thank Dave Noeldner, Brian Gutcher, Kevin Campbell, Kurt Kastein,

Al Poppelmann, Li Du, Jongseung Park, and Andrei Vityaev from my time at LSI Corp. Also, I would like to thank Kamran Shahroudi, Jack Schade, and the rest of my coworkers at Woodward Governor.

A part of the research was funded by the NSF under grant ECS-0245291. Funding was also provided by the Department of Electrical and Computer Engineering at CSU. I would like to thank the ECE Department chair Tony Maciejewski for allowing me to teach classes and for providing me with teaching assistantships. Part of this research was performed in conjunction with Solix Biofuels, who also funded a piece of the project. I would like to thank Bryan Willson, Mark Machacek, Steve Bunch, and Guy Babbitt for their collaboration during my time at Solix.

I would like to thank my mom, Mary Ann, and Mike for their support.

Last, but most certainly not least, I would like to thank all of my friends who have become my surrogate family over the years. The list has gotten to be too long to list everyone, but I would particularly like to thank Nate and Patti; Truby and Becky; Trav, Jami, and Gitana; Moergen; Funk and Donna; Ehren; Andrea; Jared and Jen; Jimbo; Jeremy and Sheri; Jeff and Kim; Neil and Jaime; Ken and Gina; Das and Spirit; Ani and Kirsten, and all of their families for being there all of these years. I would also like to thank my long distance friends that I meet up with whenever life allows. This list is also too long to list everyone, but I would particularly like to thank Thompson, Johnny, Tommy, Rich, Tammy, Jay, Christine, Matt, Katie, Brian, Tiffany, Harrison, Dave, and Kari.

DEDICATION

In loving memory of my father, Pat Buehner.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATION AND SYMBOLS

$\mathbb{R}$ and $\mathbb{C}$	fields of real and complex numbers
$\in$	belongs to
$\subset$	subset
$\cup$	union
$\cap$	intersection
$\square$	end of proof
$\diamond$	end of remark
$\overline{\alpha}$	complex conjugate of $\alpha \in \mathbb{C}$
$\|\alpha\|$	absolute value of $\alpha \in \mathbb{C}$
$\mathrm{Re}(\alpha)$	real part of $\alpha \in \mathbb{C}$
$R^{n\times m}$	real matrix with n rows and m columns
R^n	real column vector with n elements
I_n	n x n identity matrix
$C^{n\times m}$	complex matrix with n rows and m columns
C^n	complex column vector with n elements
a_{ij}	an element in the ith row and jth column in the matrix A
$diag(a_1, \ldots, a_n)$	an n x n diagonal matrix with a_i as its ith diagonal element
A^T and A^*	transpose and complex conjugate transpose of A
A^{-1}	inverse of A such that $AA^{-1} = A^{-1}A = I$
$A^\dagger$	Moore-Penrose pseudoinverse of A (non-square or singular)

$\lambda(A)$	eigenvalue of A
$\lambda_i(A)$	an eigenvalue value of A
$\rho(A)$	spectral radius of A $(= \max_i \lvert\lambda_i(A)\rvert)$
$\sigma(A)$	singular value of A
$\sigma_i(A)$	ith singular value of A
$\overline{\sigma}(A)$ and $\underline{\sigma}(A)$	the largest and smallest singular values of A
$det(A)$	determinant of A $(= \prod_i \lambda_i(A))$
$tr(A)$	trace of A $(= \sum_i \lambda_i(A))$
$\forall\ x$	for all values of x
sup	supremum, the least upper bound
inf	infimum, the greatest lower bound
$\mathcal{L}_\infty(j\mathbb{R})$	space of functions bounded on $\mathrm{Re}(s) = 0$ including at ∞
$\mathcal{L}_2(j\mathbb{R})$	space of square integrable functions on $\mathrm{Re}(s) = 0$
$\mathcal{H}_\infty$	subset of $\mathcal{L}_\infty(j\mathbb{R})$ with functions analytic in $\mathrm{Re}(s) \geq 0$
$\mathcal{H}_2$	subset of $\mathcal{L}_2(j\mathbb{R})$ with functions analytic in $\mathrm{Re}(s) \geq 0$
$\mathcal{RM}$	space of real-rational proper transfer matrices
$\mathcal{RH}_\infty$	subset of $\mathcal{RM}$ with no poles in $\mathrm{Re}(s) \geq 0$
G	plant model
K	feedback controller, in whatever configuration
L	loop transfer function $L = GK$
S	sensitivity function, $S = (I + L)^{-1}$
T	complementary sensitivity function $T = I - S = L(I + L)^{-1}$
Δ	uncertainty
$\lVert\Delta\rVert_\infty$	$\mathcal{H}_\infty$ norm of a system: $\lVert\Delta\rVert_\infty = \sup_\omega \overline{\sigma}(\Delta(j\omega))$

LIST OF ABBREVIATIONS

FF	Feedforward
DTC	Dead-Time Compensator
SP	Smith Predictor
BIBO	Bounded-Input Bounded-Output (stability for linear systems)
LTI	Linear Time-Invariant
NLTV	Non-Linear Time-Varying
RHP	Right Half Plane
LHP	Left Half Plane
NID	Non-invertible / Invertible Decomposition
AMD	All-Pass / Minimum-Phase Decomposition
MPC	Model Predictive Control
τ	Time Constant (seconds)
TSFFC	Two Stage Feedforward Control
DFFPC	Dual Feedforward Predictive Control
DFFSP	Dual Feedforward Smith Predictor
NS	Nominal Stability
NP	Nominal Performance
RS	Robust Stability
RP	Robust Performance

MIAC	Model Identification Adaptive Control
RL	Reinforcement Learning
SARSA	State Action Reward State Action
NREL	National Renewable Energy Lab (in Golden, Colorado)
PBR	Photobioreactor
CSI	Control-Structure Interaction

Chapter 1

Introduction

In many control applications, feedback controllers are designed to give both good reference tracking and good disturbance rejection. In general, reference tracking leads to large initial error signals, whereas disturbances generally produce smaller error signals. Since linear feedback controllers perform better on smaller error signals, it is desirable to eliminate the large tracking errors seen in reference tracking. In the work presented here, a two-stage feedforward (FF) plus feedback architecture is developed that addresses the large error signal seen during reference tracking for non-minimum phase systems. In the work presented here, a controller architecture is given that has the following *perfect tracking control* property:

Perfect Tracking Control: When the model of the plant is perfect and there are no external disturbances, signals generated by a pair of feedforward controllers will be tracked with zero feedback error for all time.

The idea of perfect tracking control is an interesting and important topic in control, since it addresses the questions of:

- What trajectory can the plant actually follow?
- What control signal will drive the plant along this trajectory?

Essentially, this characterizes the class of signals that the closed-loop system can

attain. The same conditions for perfect tracking hold for both stable and unstable systems; however, it is the non-minimum phase components that create additional constraints on what may be perfectly tracked. This idea has been primarily investigated for minimum-phase applications.

In the work presented here, the characterization is extended to all linear time-invariant (LTI) systems (of a specific form that is defined in Chapter 3) through the use of a specific controller architecture. In some of the published work, multi-rate digital techniques are used that guarantee zero tracking error at the sample points for a specific class of signals. In the work presented here, an architecture is developed that can guarantee perfect tracking using either continuous-time or discrete-time controllers. In the discrete-time case, multi-rate techniques are not required to maintain the perfect tracking property. For the controller architecture developed here, the perfect tracking property holds for both minimum-phase and non-minimum phase systems (i.e., systems with time delays and right half plane zeros) that are not causally invertible. This is a feature not present in current feedforward plus feedback controllers.

Many process control applications have non-minimum phase components that are difficult to control using traditional feedback techniques [1]. Specifically, they tend to have very long time delays that result from transporting gases and fluids over long distances. These types of delays result in feedback controllers that are very sluggish, which is required to maintain closed-loop stability. A specific example of this is growing microalgae for biofuels in a large flat panel photobioreactor (PBR). In order to get an increase in biomass production, a CO_2 rich input gas stream is delivered from a central source. In practice, there can be a long delay between the CO_2 source (e.g., a power plant) and the individual flat panels (e.g., the PBR may be located on property adjacent to the power plant). This may result in significant gas transport delay (e.g., on the order of minutes).

The methodologies that are specific to controlling systems with dead times are known as *dead-time compensators* (DTCs) [1]. Smith predictors (SP) and model predictive control (MPC) are two common methods that may be used as DTCs [1]. Both of these methods use a prediction of the dead time inside the feedback loop to predict what the future output of the plant will be. Then, a feedback controller corrects the control signal based on the difference between the predicted and actual outputs. Feedforward enhancements have been added to each of these methods for disturbance rejection, but these feedforward enhancements are not used to improve reference tracking. Instead, improved reference tracking has been primarily focused on filtering the reference signal so that the feedback controller can track the filtered reference with a smaller tracking error. In this case, the reference filter determines the class of signals that the feedback controller will attempt to track. In the work presented here, two new methods are presented that can provide perfect tracking for a certain class of signals that are dependent upon the non-minimum phase components of the plant. In the first method, a prediction of the path that the closed-loop system will follow is given in one of the feedforward paths, and a second feedforward controller provides the control signal that will drive the system along the predicted closed-loop path. In this case, the path that the closed-loop system will follow is determined by both a design parameter and the non-minimum phase components of the plant, which specifies the class of signals that the feedforward controller can perfectly track when there are no modeling errors or external disturbances. In the second method, an augmentation is made to a standard Smith predictor to provide perfect tracking. In this case, the prediction of the non-minimum phase components appears in the feedback loop, but the design parameter remains in the first feedforward path. This method provides an analogous result to the other architecture for stable systems; however, this method is unsuitable for unstable systems, which makes it less general than the first method. For both of these methods, the perfect tracking is obtained by

sharing signals between the two feedforward controllers. While components of both of these methods appear in the literature, there is no general theory that presents them as a whole.

Adaption may be used to cope with manufacturing variations in parts, change in performance due to wear-and-tear over time, and biological changes in living organisms. The last case is specifically true for the microalgae production example given earlier. One of the big advantages to the proposed structure is that it is easier to stably adapt feedforward components than it is components inside the feedback loop. This is due to the fact that closed-loop stability is not affected by adapting the feedforward components, since they are outside of the feedback loop. Instead, the only requirement is that the feedforward components themselves be bounded to preserve bounded-input bounded-output (BIBO) stability for the overall system. For certain adaptation schemes, this structure will allow for a larger stable adaptation range due to the reduced stability requirements. This type of adaptation is particularly useful for applications that need to adapt to changing plant conditions.

1.1 Primary motivation: Biofuels from Microalgae

Microalgae can convert carbon dioxide (CO_2) into storage lipids that can be refined into biofuels. More CO_2 is being produced now than in previous years, and there is limited supply of fossil fuels in the world. Under the right conditions, microalgae will utilize the excess CO_2 being produced by human activity to produce lipids that may help supplement the limited fuel supply. Some of the control objectives are to:

- Maximize biomass production.
- Maximize CO_2 uptake efficiency.
- Maximize storage lipid production (i.e., the lipids that are favorable for biofuel production).

The focus of the work presented here is on improving biomass production and CO_2 uptake efficiency by regulating key variables (e.g., pH, dissolved CO_2, dissolved O_2, and temperature). While the use of models and controllers to improve lipid production are not addressed in this work, increased lipid production was achieved through the use of an appropriate microalgae strain [2].

Biofuels have gone through two generations to get to the so called "Third Generation" of biofuels that currently exist, namely *biofuels from microalgae*. In the first generation, ethanol from food crops (such as corn) was used as a fuel supplement. While these are able to produce fuel, they deplete the land of vital nutrients, cut into the food supply, and do not have the potential to replace U.S. fossil fuel demand. In fact, it would take roughly 1.54 billion hectares, which is roughly 846% of the existing cropping land in the U.S., to meet 50% of the U.S. oil demand [2]. In the second generation of biofuels, non-food plants such as Jatropha were used to produce biofuels. The advantages to plants such as Jatropha are that they do not interfere with the food supply and they have a higher oil yield per acre. For Jatropha, it would take 140 million hectares, which is about 77% of the existing U.S. cropping land, to meet 50% of the U.S. fuel demands [2]. While this is a significant improvement over corn, it still requires a large portion of the existing U.S. cropping land.

A program of research sponsored by the National Renewable Energy Laboratory (NREL) from 1978-1996 estimated that microalgae could produce lipids at the rate of 7,000 - 15,000 gallons/acre/year [3]. This is almost 35X the current productivity of lipids from Jatropha, at roughly 200-250 gallons/acre/year [2], and almost 350X the current productivity of lipids from corn, at roughly 20-25 gallons/acre/year [2]. This means that microalgae could meet 50% of all transport fuel needs of the United States while using only 2 to 4.5 million hectares, which translates to about 1.1% to 2.5% of the existing US cropping area [2]. While this is encouraging, there have been numerous problems with scaling up to a large production facility [4]. The primary

challenge has been to design economical photobioreactors (PBRs) that are able to utilize intense light and maintain appropriate gas concentrations at a commercial scale (c.f., [2; 4; 5; 6; 7; 8; 9; 10]).

Various PBR configurations (e.g., bubble columns, raceway ponds, and flat panel reactors) have been studied in [9]. Each configuration has its advantages and disadvantages. The trade-offs are designing reactors that are affordable to purchase and operate at a commercial scale, while efficiently utilizing high intensity light, and efficiently removing dissolved oxygen produced during photosynthesis. For example, bubbling a gas stream through the media containing the microalgae is very efficient at removing dissolved oxygen, which is required for sustained growth, but is costly to operate on a continuous basis. Therefore, the gas stream may be bubbled through the media intermittently to get a good balance between reduced operating costs and adequate dissolved oxygen removal.

System models provide a valuable tool for evaluating the different reactors. In particular, it is important to develop models that are valid for both small scale reactors and larger scale production reactors. In the work presented here, this is achieved by defining quantities in terms of densities and efficiencies as opposed to overall yields. By doing this, the comparisons of performance at various scales can be a one-to-one comparison (e.g., the performance of a $300L$ reactor is considered to be the same performance of a $6000L$ reactor if they both yield the same biomass density in g/L. However, the overall yield will be the biomass density scaled by the size of the reactor.). The majority of the existing models are for tubular reactors. A variety of linear and nonlinear models have been developed to address this for tubular reactors (c.f., [5; 6; 7; 8; 10; 11; 12; 13; 14; 15; 16]). However, there have been significantly less publications on flat panel reactors, which is the style of reactor addressed in the work presented here (see later for reasons).

Model based controls have been developed in [10; 17; 18; 19; 20; 21]. Most of these papers model growth as a function of light using a *Monod kinetics* model. The Monod model describes how bacteria go through exponential, linear, and decaying growth phases as they consume their nutrients. This idea is extended to how microalgae grow in a light limited closed PBR. This is an entirely empirical model that works well for some situations, but lacks physical intuition of growth in a light limited environment. In the work presented here, a scalable model is developed that addresses these phases by considering the physics of resource limited microalgae growth inside a vertical flat panel PBR. The microalgae growth rate is modeled as a function of incident light. When the culture goes above a critical density, the growth changes from exponential to linear, because the microalgae become light limited (i.e., only a fraction of the microalgae in the reactor receive light and grow, while the remaining microalgae stay in the dark and respirate, which results in a loss of biomass). As biomass continues to increase, a larger amount of microalgae do not receive light and respirations losses will become greater. Our results regarding this were published in [22] and will be revisited in Chapter 10.

According to the Center for the Study of Carbon Dioxide and Global Change [23], aquatic plants, such as microalgae, are very sensitive to the amount of dissolved CO_2 available for photosynthesis. In particular, an elevated CO_2 concentration can lead to a dramatic increase in biomass. For most experiments and PBR setups, the amount of dissolved CO_2 available for photosynthesis is regulated by bubbling through a CO_2 enriched gas stream (e.g., air mixed with pure CO_2 to achieve a desired ppm CO_2). In [24], the green microalgae *Scenedesmus obliquus* was grown at the CO_2 concentration of ambient air (about 387 ppm CO_2) and elevated CO_2 (about 100,000 ppm CO_2). After growing for five days, the microalgae exposed to the elevated CO_2 concentration accumulated about 300% more biomass at a low light intensity and 600% more biomass at a higher light intensity than the microalgae exposed to

ambient air levels of CO_2. In a similar study in [25], a freshwater microalgae from the genus *Chlorella* was grown at elevated CO_2 concentrations. Relative to ambient air concentrations, the microalgae growth rates were 200% more at 100,000 ppm CO_2, 170% more at 200,000 ppm CO_2, 125% more at 300,000 ppm CO_2, and 40% more at 500,000 ppm CO_2. These results show that both a significant increase in biomass accumulation may be achieved by proper CO_2 regulation, and there is a desired CO_2 concentration for maximal biomass production. For verification of the model in the work presented here, the microalgae *Nannochloropsis oculata* is used.

In order to improve biomass production, an overall model is developed that relates the interactions between the three subsystems, namely the incident light subsystem, the microalgae subsystem, and the media subsystem that consists of the surrounding media that provides CO_2 and nutrients for growth and receives dissolved O_2 from photosynthesis. Developing these subsystems is required to understand the interactions between:

- A CO_2 enriched gas stream and the microalgae media.
- Dissolved gases (i.e., CO_2 and O_2) in the media and the microalgae.
- Incident light intensity and microalgae photosynthesis.

Control Objective: The primary control objective is:

- To supply CO_2 to the microalgae as it is being consumed by photosynthesis to maintain a desired pH in the media.

One of the biggest difficulties in maintaining the proper CO_2 concentration at a large scale is dealing with the long transportation delay between when the CO_2 concentration in the input gas stream is changed and when the CO_2 reaches the media. To further complicate this, dissolved CO_2 is not measured directly due to the high cost of an online dissolved CO_2 sensor. Instead, dissolved CO_2 is inferred from a

less expensive pH sensor. Under a relatively constant pressure and temperature, pH will correlate well with dissolved CO_2. As CO_2 dissolves into the water, it combines with water molecules to form negatively charged hydroxide ions (OH^-) and positively charged hydrogen ions (H^+) that determine the pH. This reaction may take a few seconds, which adds another delay from when a CO_2 concentration is commanded and when it is finally sensed as a change in pH. A photo of the actual PBR used for these experiments is shown in Figure 1.1.

Figure 1.1: Photobioreactor Test Bed.

1.1.1 Microalgae Contributions

The main contributions to the area of biofuels from microalgae are:

- Develop physically based models that are independent of scale that may be used to measure reactor performance.

- Utilize the developed models for feedforward CO_2 delivery to maintain a desired dissolved CO_2 concentration in the media.

1.2 Deeper Motivation: Improving Tracking Performance

Feedforward control may be used to either improve the tracking performance or disturbance rejection of traditional closed-loop feedback control systems. For improved tracking of reference changes (e.g., a step input), a two degree-of-freedom (2-DOF) controller is used where the feedforward controller filters the reference input to provide a reference trajectory that the feedback controller can track [26; 1]. The motivation is that the filtered reference signal will produce smaller error signals, which are more appropriate for the feedback controller. For improved disturbance rejection, a feedforward controller uses measurable and predictable disturbances to counteract the effect that these disturbances will have on the closed-loop system. In these cases, the output of the feedforward controller is added to the feedback control signal to provide an overall control effort. In the work presented here, these two ideas are combined to improve the tracking performance of the overall control system. This idea parallels some of the concepts from neuromuscular actuation systems.

In humans, the neuromuscular system is the servomechanism portion that includes sensory and motor neurons at the spinal cord level and their associated muscles, joints, and receptors in the periphery [27]. When this system is used to command muscles (e.g., to command motion of a limb), it is referred to as the neuromuscular actuation system. This system naturally describes a combination of the two feedforward control methodologies described above, since it describes how the brain decides upon an action, takes the action, and then corrects for errors along the way. As an example, consider the neuromuscular actuation system (NMAS) used to reach for a cup. In this case, the following happens:

1. The human brain calculates the desired path (feedforward prediction).

2. A ballistic response is implemented where the hand extends towards the cup (feedforward control signal).

3. Small corrections to the desired path are made based on the "observed" position of the hand (feedback control on small error signals).

This type of phenomena is well documented in the literature (e.g., [27; 28; 29; 30; 31; 32; 33]). Each step of the NMAS aligns well with the previously discussed controllers. For example, the filtered reference output of a 2-DOF feedforward controller is a prediction of a path that the closed-loop can track with no error, which is similar to the feedforward prediction in a NMAS. A feedforward controller provides a large control signal, which is similar to the ballistic response in a NMAS. When a system being controlled is stable minimum-phase, these techniques work well by themselves to improve tracking since they can essentially invert the plant dynamics. In both cases, the feedback controller is used to correct the control output based on the "observed" and "desired" outputs, which is one of the main arguments for using feedback control. When the system is non-minimum phase, both of the feedforward methods break down. This is due to the fact that non-minimum phase systems do not have stable causal inverses. However, the benefits from each of the two feedforward controllers may be combined to provide perfect tracking on non-minimum phase systems, which is the focus of the work presented here. In particular, a non-minimum phase plant may be factored into a minimum-phase piece that does have a stable causal inverse and a non-minimum phase piece that does not have a stable causal inverse. The feedforward prediction addresses the non-minimum phase part of the plant and the second feedforward controller provides a large control signal, which is the ballistic response that addresses the stable causal invertible piece of the plant. In the nominal case with no external disturbances, the plant output will perfectly track

the feedforward prediction. In order to achieve this, the feedforward prediction and feedforward ballistic response systems must share information between them, which means that neither method by itself may achieve the perfect tracking property.

To continue with the previous cup example, assume that the person repeatedly reaches for the same cup. As this happens, the brain gets better at both calculating the fastest path to the cup (i.e., the first feedforward controller) and implementing that path (i.e., the second feedforward controller), which reduces the need for corrective actions (i.e., the feedback controller). This idea is extended to adaptive learning where the two feedforward controllers are adapted to improve the overall performance of the closed-loop system, while a fixed feedback controller is used to correct for differences.

As an extension to the previous example, consider what happens when the cup is placed on a table that is tilted at a 45^{o} angle so that the cup is sliding in the direction of gravity. Now when the person tries to grab the cup, the first feedforward control response needs to predict where the cup will be by the time the hand arrives at the cup location and the second feedforward controller implements the appropriate action. Now, the feedback controller is correcting for errors along the predicted path and not the current error between the cup's position (reference position) and the hands position (measured position). In this case, the feedforward controllers are working together to pick the best path, and the feedback controller is only used to correct for small differences between the planned path and actual path, which motivates the need for communication between the two controllers.

The presented controller architecture greatly simplifies closed-loop stability analysis by moving the adaptive and nonlinear parts of the control system out of the feedback loop while still utilizing both the strength of the feedback controller (i.e., correcting on small error signals), and the power of nonlinear adaptation for large variations.

1.2.1 Control Theory Contributions

The contributions to control theory involve providing a new general architecture that is applicable to non-minimum phase systems. These contributions include:

- Predictive feedforward structure that uses two feedforward controllers to get perfect tracking on a certain class of signals for LTI systems (of a specific form).
- Addition of a feedforward component to a 2-DOF plus Smith predictor structure to get perfect tracking on a certain class of signals for stable LTI systems (of a specific form).
- Robustness analysis and design tools for the presented structures.
- Stable adaptation methods for the proposed structure that will improve performance.
- An improved stability bound for adaptation via echo state networks based on robust control theory.

1.3 Relation to Existing Approaches

The idea of modeling a neuromuscular actuation system has its roots in [27]. Since that time, the idea has been extended to feedback control systems in various forms [28; 29; 31; 32; 33]. In each of these methods, the feedforward controller is used to provide either the ballistic response or the feedforward prediction and the feedback controller provides the other. In these cases, the feedback controller is active during reference tracking, which means that the perfect tracking property (i.e., zero error during reference tracking when there are no modeling errors or external disturbances) described previously does not hold. In the work presented here, a 2-DOF architecture is developed that is able to provide both the feedforward prediction and the ballistic responses. This results in a controller architecture that is able to perfectly track

reference inputs for LTI systems (of a specific form). In [28], the ideas of using a feedforward ballistic controller and the use of Smith predictors are discussed separately; however, the combination of the two is never discussed, and the idea of perfect tracking is never explored. In the controller architectures presented here, the ideas of [27] are interpreted differently from the current approaches. Specifically, the nominal reference tracking requirements are eliminated from the feedback controller design, since reference tracking is done completely by the feedforward controller (when there are no modeling errors or external disturbances). In optimal control, the goal of the feedback controller design is to minimize the effects of disturbance inputs on controlled outputs. In robust control, the feedback controller is designed to provide optimal control in the presence of model uncertainty. In a traditional optimal control design, the *reference tracking problem* is posed as a disturbance on the feedback error, and the *disturbance rejection problem* is posed as a disturbance on the output. This means that the feedback controller is designed to minimize the effects of both reference inputs and disturbance inputs. By pulling the reference tracking requirements out of the feedback design and putting them into the feedforward design, the feedback controller may be designed solely as a disturbance rejection problem, which utilizes the strength of the current robust and optimal feedback controller design methodologies.

The idea of perfect tracking has been explored in the applications of seeking control in a hard disk drive [34; 35], voltage control of an inverter [35], and a magnetic levitation system [36]. In these applications, multi-rate digital controllers are used to guarantee that the plant output matches a predicted output at the sample times of the slowest sampling rate in the system. These methods use a controller in the feedback loop to provide the ballistic response, which is a different approach than the one considered here. Also, these methods restrict the perfect tracking property to discrete-time systems, whereas the architecture presented here may be implemented using either continuous-time or discrete-time systems. This generalization to both

continuous-time and discrete-time systems results from the unique controller architecture that is used here.

The use of multiple feedforward controllers was explored in [37], where a data preprocessor was used to feed both a pre-filter and a feedforward controller on a selective catalytic reduction (SCR) catalytic converter. In this application, the feedforward controller was designed based on the known dynamics of the system, and the pre-filter provided a prediction of the expected trajectory. The difference between the work presented here, and this method, is that the data preprocessor in this method used the measured output as an input, which created a feedback loop. Also, the controller for this application did not have the perfect tracking property that we are able to achieve with the controller architecture presented here.

The use of adaptive feedforward control to improve reference tracking in a robot motion planning application was explored in [38]. In this method, the controller was decomposed into a path planning stage that determine the speed and shape of the path, a feedforward controller that provided the reference path, and a cascaded feedback controller that provided the ballistic response and was used for position, speed, and current control based on the reference path. In order to improve the tracking performance of the overall system, the feedforward controller was adapted to adjust the reference path to the path that the robot was actually taking. By adapting this piece, it allowed the feedback controller to provide control signals that the plant was actually responding to, but it did not adapt the ballistic response, since this piece was in the feedback loop. In the work presented here, we also adapt the feedforward controllers to calculate a reference signal that the plant will actually follow; however, we are also able to adapt the ballistic response without affecting closed-loop stability, since the ballistic response is not in the feedback loop.

A similar controller architecture was developed (in parallel to the work presented here) that appeared in [39]. This method characterizes the signals that may be

perfectly tracked for minimum-phase systems. Through an example, they showed how their architecture could characterize the signals that may be perfectly tracked for a system with a right-half plane zero (i.e., a system with a specific non-minimum phase component). However, due to restrictions in their problem formulation, perfect tracking of the class of LTI systems considered here is not possible. In the work presented here, this limitation is overcome by specifying a new control architecture that explicitly addresses the non-minimum phase components of a system. Also, adaptation was never discussed in [39], which is addressed in the work presented here.

1.4 Objectives

Perfect tracking control has been considered by various researchers. Most of this work has been focused on essentially inverting minimum-phase plants and on using multi-rate digital controllers. For the multi-rate digital controllers, the methods are focused on modifications made to an architecture that first appeared in [34], which has the limitations that parts of the structure (i.e., the part of the controller that provides the ballistic response) cannot be adapted without taking extra steps to ensure closed-loop stability, a fast-rate digital controller is required, and the robust analysis and design tools have not been fully developed for this method. These issues are addressed in the presented work by:

- Developing a two-stage feedforward plus feedback controller architecture that improves closed-loop tracking performance (in particular for non-minimum phase systems).
- Developing robustness analysis tools for quantifying the performance gain.
- Developing robust design methodologies for synthesizing the feedforward and feedback controllers.

An issue with model based controllers, such as the one presented here, is that they require an accurate plant model. This model can either be known a priori or learned as the system runs. In the latter case, adaptation schemes are used to improve the plant models. The current use of adaptive control can improve reference tracking and disturbance rejection performance of closed-loop systems, but these methods either require a large amount of computational overhead for stability analysis, or are restricted to a class of plants and specific control law. These issues result from the adaptive and predictive parts of the controller being inside the feedback loop. Since most available optimization routines have at least polynomial complexity (e.g., $O(n^4)$), scaling a plant up by a factor of 10 can cause the numerical complexity of the optimization routine to increase by a factor of 10000 or more, which makes solving the optimization problem unrealistic. These issues are addressed in the presented work by:.

- Only adapting the feedforward part of the new controller architecture.
- Developing an improved stability bound for echo state networks that will be used for adaptive control.
- Demonstrating the control algorithms on set of illustrative examples.

By restricting the adaptation to the feedforward components of the controller architecture, we only need to perform a one-time stability check a priori, because we are not adapting elements inside the feedback loop. The effectiveness of the proposed method will be verified on illustrative examples in Chapter 9, where the current architecture will be compared against some traditional feedback methods to demonstrate improvements in performance, robustness, and ease of design.

In the final part of the presented work, modeling and control of a PBR growing microalgae for biofuels is developed. Here, models are used to study the photosynthetic efficiency of microalgae inside a flat panel PBR. These models, along with the

controller architecture developed earlier, are used to characterize the attainable pH regulation, which promotes enhanced microalgae growth. This will be done by using a combination of experimental verification on a physical PBR and simulation on verified models. For this part of the dissertation, the objectives are to:

- Develop and validate a model for growing microalgae in a flat panel PBR using experimental data.
- Use the model to determine the photosynthetic efficiency of microalgae.
- Demonstrate the achievable pH regulation.

1.5 Overview of Dissertation

The remainder of this dissertation provides the mathematical framework for achieving perfect tracking that encapsulates the limitations of non-minimum phase systems. Chapter 2 provides an overview of known results in the controls community. Chapter 3 frames the limitations imposed by the non-minimum phase components of a physical plant and provides an overview of two new controller architectures that may achieve perfect tracking for both minimum-phase and non-minimum phase systems. Chapters 4 and 5 develop the two controller architectures in detail and provide robustness tools for controller analysis and synthesis. For both of these architectures, there is a common piece to the controller design. Chapter 6 provides methodologies for designing this common piece of the controller.

The two developed architectures contain feedforward controllers that are based on plant models. Since these controllers are not in the feedback loop, they may be easily adapted without affecting closed-loop stability. Chapter 7 provides methods for adapting and augmenting these controllers based on system identification and reinforcement learning, respectively. In the work presented here, reinforcement learning via echo state networks is considered. Echo state networks were originally defined

in terms of two necessary conditions on a recurrently connected layer in the ESN. In [40], we both redefined these conditions into necessary and sufficient conditions and provided a less restrictive sufficient condition. This material is presented in Chapter 8.

Chapter 9 provides illustrative examples that demonstrate the two architectures and the various adaptation schemes. The purpose of this chapter is to provide insight on how to use the methods developed.

Chapters 10 and 11 are focused on a specific application of growing microalgae for biofuel production. Chapter 10 develops a scalable model of a PBR. This model is a physically-based model that is validated on using experimental data from an actual PBR. Chapter 11 uses this model to develop controllers that regulate pH, which will promote increases biomass production.

Chapter 12 concludes the presented work and provides future directions.

Chapter 2

Background

This chapter contains the preliminary material required for the controller development in later chapters.

2.1 Polynomials and Transfer Functions

Most of the controller development will be focused on controlling continuous-time plants that are modeled as *linear time-invariant* (LTI) real rational transfer functions. A continuous-time transfer function may be expressed as a ratio of polynomials such as

$$G(s) = \frac{B(s)}{A(s)} = \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \ldots a_1 s + a_0}, \tag{2.1}$$

where *roots* of the polynomial $B(s)$ are the *zeros* of the transfer function $G(s)$, and the *roots* of the polynomial $A(s)$ are the *poles* of the transfer function $G(s)$. Here, $s \in \mathbb{C}$; however the coefficients of the $B(s)$ and $A(s)$ are real valued (i.e., $b_i \in \mathbb{R}$ for $i = 1 \ldots m$ and $a_j \in \mathbb{R}$ for $j = 1 \ldots n$). When s is evaluated at $s = j\omega$, the result, namely $G(j\omega)$, is referred to as the *complex frequency response* of the system. Systems defined this way are known to have Hermitian symmetry (i.e., $G(j\omega) = \overline{G(-j\omega)}$) and their poles and zeros will appear as complex conjugate pairs (e.g., if $s_1 = -3 + j5$ is pole (or zero) of $G(s)$, then $\bar{s}_1 = -3 - j5$ will also be a pole (or zero) of $G(s)$).

2.1.1 Relative Degree of a Transfer Function

The *relative degree* of a transfer function is given by "number of poles" minus the "number of zeros", or $n-m$ from eqn (2.1). For a system to be realized in hardware, it must have a non-negative relative degree (i.e., $n \geq m$ in eqn (2.1)), which is referred to as a proper system. If $n > m$ (i.e., a positive relative degree), this is referred to as a strictly proper system, and if $n = m$, this is referred to as a biproper system.

The set of all real-rational proper transfer matrices is denoted $\mathcal{RM}$. This is the set of matrices whose entries are transfer functions of the form in eqn 2.1 with a finite number of real coefficients (i.e., every matrix entry is a transfer function with $m \leq n < \infty$ and b_k and a_i are real for $k = 1, \ldots, m$ and $i = 1, \ldots, n$, respectively).

2.1.2 Pole and Zero Locations

In the work presented here, causal continuous-time LTI systems will be classified as stable, unstable, minimum-phase, and non-minimum-phase based on the location of their poles and zeros. A general system may be decomposed into its stable, unstable, minimum-phase, and non-minimum-phase components as

$$G(s) = \frac{N(s)}{D(s)} = \frac{N_{nmp}(s)N_{mp}(s)}{D_u(s)D_s(s)}. \tag{2.2}$$

Here, the roots of the numerator polynomial $N_{mp}(s)$ are the *minimum phase zeros* of $G(s)$ with $\mathrm{Re}(s) < 0$ and the roots of the numerator polynomial $N_{nmp}(s)$ are the *non-minimum phase zeros* of $G(s)$ with $\mathrm{Re}(s) \geq 0$. Similarly, the roots of the denominator polynomial $D_s(s)$ are the *stable poles* of $G(s)$ with $\mathrm{Re}(s) < 0$ and the roots of the denominator polynomial $D_u(s)$ are the *unstable poles* of $G(s)$ with $\mathrm{Re}(s) \geq 0$. If there are no poles or zeros in the half plane, then the polynomial is equal to one (e.g., if there are no unstable poles in $G(s)$, then $D_u(s) = 1$). For the scenarios considered here, stable means that the system is *bounded input-bounded output* (BIBO) stable, which results from all of the poles being in the open left-half plane (LHP). In some contexts,

the term *marginal* is used to describe poles and zeros on the $j\omega$-axis (i.e., $\mathrm{Re}(s) = 0$); however, in this context, poles and zeros on the $j\omega$-axis will be considered unstable and non-minimum phase, respectively. It should be noted that these definitions hold for causal systems only.

2.1.3 Discrete Time Systems

Discrete-time transfer functions, such as $G(z)$, are described by a ratio of polynomials in $z \in \mathbb{C}$. For causal discrete-time systems, minimum-phase zeros and stable poles lie within the open unit circle (i.e., $|z| < 1$) and non-minimum phase zeros and unstable poles lie on or outside the closed unit circle in the complex plane (i.e., $|z| \geq 1$). The discrete-time frequency response is the discrete-time transfer function, namely $G(z)$ evaluated on the unit circle (i.e., $z = e^{j\omega T_s}$), where T_s is the sample period of the discrete-time system. This discrete-time complex frequency response is written as $G(e^{j\omega T_s})$.

2.2 Zero-Order Hold Equivalent Systems

In practice, the final controller will be implemented in discrete-time. For the model-based designs in the work presented here, a *zero-order hold* (ZOH) equivalent discrete-time representation of the plant is used. Figure 2.1 illustrates the equivalence between the continuous-time and discrete-time systems.

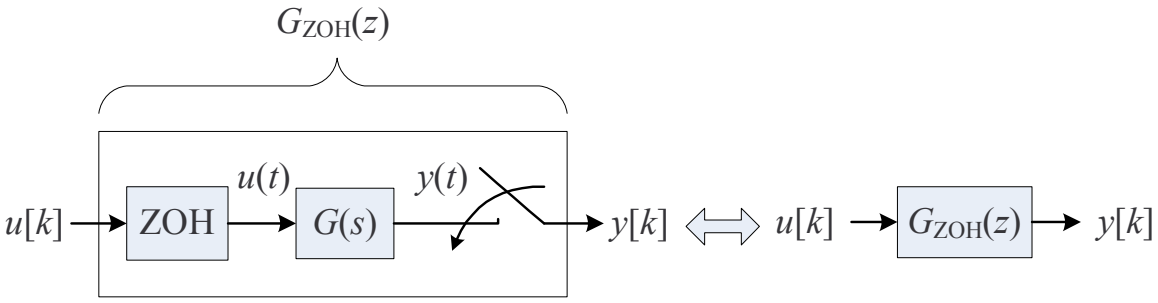

Figure 2.1: Zero-Order Hold Equivalent Block Diagrams

In this setting, a discrete-time control input $u[k]$ is passed through a zero-order hold digital-to-analog converter at rate $F_s = 1/T_s$ to produce the continuous-time control input $u(t)$. This continuous-time control input is applied to the analog plant $G(s)$ to produce the continuous-time output $y(t)$. If this output is sampled in phase at the same rate of the digital system (i.e., sampled at rate F_s), then the outputs from the two block diagrams in Figure 2.1 are equal. For more information, see [41].

2.3 All-Pass Filters

An all-pass filter is a filter that has a unity magnitude for all frequencies, but may change the phase relationship between frequencies. An all-pass filter is created by placing the poles and zeros in the transfer function such that for every stable pole in the left half plane, there is a mirror image non-minimum phase zero in the right half plane (i.e., the position of the zero is the pole location reflected across the $j\omega$ axis). A first order all-pass filter is give by

$$H_{ap}(s) = \frac{s - \bar{\alpha}}{s + \alpha}, \tag{2.3}$$

where $\alpha \in \mathbb{C}$. In order to guarantee that the transfer function is causal and stable, the restriction $\text{Re}\{\alpha\} > 0$ is imposed.

2.3.1 Phase Interpolation

For the discussion here, we assume that $\alpha \in \mathbb{R}$. Now, the magnitude is given by

$$|H_{ap}(j\omega)| = \left|\frac{j\omega - \alpha}{j\omega + \alpha}\right| = \sqrt{\frac{\omega^2 + \alpha^2}{\omega^2 + \alpha^2}} = 1, \tag{2.4}$$

which motivates the name all-pass. The phase response (in radians) is given by

$$\begin{aligned} \theta_{ap} = \angle H_{ap}(j\omega) &= \pi + \tan^{-1}\left(\frac{\omega}{-\alpha}\right) - \tan^{-1}\left(\frac{\omega}{\alpha}\right) \\ &= \pi - 2\tan^{-1}\left(\frac{\omega}{\alpha}\right), \end{aligned} \tag{2.5}$$

which can take on frequencies $0 < \theta_{ap} < \pi$. For phase interpolation, we need to be able to interpolate $-\pi < \theta_{ap} \leq \pi$. For phases $-\pi < \theta_{ap} < 0$, the all-pass filter $H_{ap}(j\omega) = \frac{-(j\omega-\alpha)}{j\omega+\alpha}$ may be used, which has phase response $\theta_{ap} = \angle H_{ap}(j\omega) = -2\tan^{-1}\left(\frac{\omega}{\alpha}\right)$. This leaves two special cases, namely $\theta_{ap} = 0$ and $\theta_{ap} = \pi$, which may be obtained by setting $H_{ap}(j\omega) = 1$ and $H_{ap}(j\omega) = -1$, respectively.

Let $G(j\omega)$ be a transfer function and let $|G(j\omega_0)|$ be the magnitude of $G(j\omega)$ evaluated at $\omega = \omega_0$. We say that an all-pass filter interpolates the frequency ω_0 of $G(j\omega)$ if there exists a all-pass filter such that

$$|H_{ap}(j\omega)||G(j\omega)| = |G(j\omega)|, \tag{2.6}$$

and

$$H_{ap}(j\omega_0)G(j\omega_0) = |G(j\omega_0)|. \tag{2.7}$$

In polar form, $G(j\omega_0) = |G(j\omega_0)|\angle G(j\omega_0)$, where $\angle G(j\omega_0) = 2\pi n + \theta_G$ (n is an integer and $-\pi < \theta_G \leq \pi$). Based on the developed equations, the relationship

$$H_{ap}(j\omega_0)G(j\omega_0) = |G(j\omega_0)|\angle(2\pi n + \theta_G + \theta_{ap}) \tag{2.8}$$

The all-pass filter design constraint becomes $\theta_G + \theta_{ap} = 2\pi$ and $\alpha > 0$ may be solved by picking the appropriate transfer function. Let $\theta_{ap} = \pi - \theta_G$ and choose the correct all-pass filter from the following options:

$$\begin{aligned}
\theta_{ap} = \pi \quad &\longrightarrow \quad H(s) = -1 \\
0 < \theta_{ap} < \pi \quad &\longrightarrow \quad H(s) = \frac{s-\alpha}{s+\alpha}, \quad \alpha = \frac{\omega_0}{\tan\left(\frac{\pi-\theta_{ap}}{2}\right)} \\
\theta_{ap} = 0 \quad &\longrightarrow \quad H(s) = 1 \\
-\pi < \theta_{ap} < 0 \quad &\longrightarrow \quad H(s) = \frac{-(s-\alpha)}{s+\alpha}, \quad \alpha = \frac{\omega_0}{\tan\left(\frac{-\theta_{ap}}{2}\right)}
\end{aligned} \tag{2.9}$$

2.4 Norms for Signals and Systems

The robust and optimal controller analysis and synthesis considered here will be quantified in terms of the "size" of signals and systems. One method for achieving this is the use of signal and system norms. There are a variety of norms that are commonly used to quantify controls systems; however, we will restrict the discussion here to the norms that are used in the work presented here. For a more general review of signal and systems norms, see [42; 43].

For signal norms, we will only utilize the 2-norm, which quantifies the energy in a signal. The 2-norm of the signal $w(t)$ is given by

$$\|w(t)\|_2 = \left(\int_{-\infty}^{\infty} \sum_i |w_i(t)|^2 dt \right)^{1/2}. \tag{2.10}$$

For system norms, we focus on the $\mathcal{H}_2$ and $\mathcal{H}_\infty$ norms of a system $G(j\omega)$. The $\mathcal{H}_2$ norm of $G(j\omega)$ is given by

$$\|G(j\omega)\|_2 = \left(\frac{1}{2\pi} \int_{-\infty}^{\infty} \sum_i \sigma_i^2(G(j\omega)) \right)^{1/2}, \tag{2.11}$$

which is finite if the transfer function $G(s)$ is strictly proper. The $\mathcal{H}_\infty$ norm of $G(j\omega)$ is given by

$$\|G(j\omega)\|_\infty = \sup_\omega \overline{\sigma}(G(j\omega)), \tag{2.12}$$

where $\overline{\sigma}(G(j\omega))$ is the maximum singular value of the matrix $G(j\omega)$.

In the optimal control settings considered later, controllers will be designed that attempt to minimize the $\mathcal{H}_2$ norm and the $\mathcal{H}_\infty$ norm of the final system (i.e., the system that contains the plant and controller). These controllers are known as the $\mathcal{H}_2$ optimal and $\mathcal{H}_\infty$ optimal controllers, respectively.

2.5 Nominal Feedback Control

A traditional single degree-of-freedom linear time-invariant controller architecture is shown in Figure 2.2.

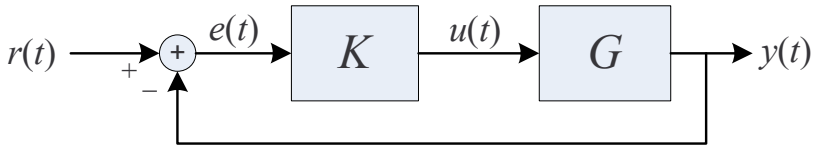

Figure 2.2: Nominal Feedback Controller Architecture

In this setting, the feedback controller adjusts the control signal $u(t)$ based on the error signal $e(t)$. If the controller is tracking the output perfectly (i.e., $y(t) = r(t)$), then the error is zero (i.e., $e(t) = 0$). The ability of the a closed-loop system to track reference inputs is given by its *sensitivity function*

$$S(s) = \frac{1}{1 + G(s)K(s)}, \tag{2.13}$$

which is the complex gain from commanded reference $r(t)$ to tracking error $e(t)$ (i.e., $S(s) = \frac{E(s)}{R(s)}$). At frequencies where $S(s)$ is small, the system is not very sensitive and the output is tracking the reference input well (i.e., the error $e(t)$ is small). The complement of $S(s)$ is known as the *complementary sensitivity function* $T(s)$ and is given by

$$T(s) = 1 - S(s) = \frac{G(s)K(s)}{1 + G(s)K(s)}, \tag{2.14}$$

For the feedback system in Figure 2.2, the complementary sensitivity function is also the closed-loop transfer function from commanded reference $r(t)$ to plant output $y(t)$ and is labeled $M(s)$ (i.e., $M(s) = T(s) = \frac{Y(s)}{R(s)}$).

The feedback loop in Figure 2.2 is said have *nominal stability* (or internal stability) if all of the poles of the closed-loop system satisfy $\mathrm{Re}(s) < 0$ (i.e., they are in the open left-half plane) and there are no pole/zero cancelations in $\mathrm{Re}(s) \geq 0$ when forming the

loop gain $L(s) = G(s)K(s)$ (i.e., there are no unstable pole/zero cancelations). There are standard ways (e.g., the Nyquist criterion) for determining internal stability of the feedback structure in Figure 2.2 (c.f., [44; 42]).

For the discussion here [42], *nominal performance* is achieved if the plot of $|S(j\omega)|$ lies under a desired curve. This may be expressed as

$$|S(j\omega)| < |W_1(j\omega)|^{-1} \quad \forall\omega \tag{2.15}$$

or in other words

$$\|W_1(j\omega)S(j\omega)\|_\infty < 1. \tag{2.16}$$

At frequencies where $|W_1(j\omega)|$ is large, the tracking error is required to be small and the feedback controller is providing good tracking performance. At frequencies where $|W_1(j\omega)|$ is small, the tracking error requirements are relaxed. This means that the controller is not required to provide good performance in these frequency ranges, which is often required for closed-loop stability. For many applications, performance is required at lower frequencies (e.g., steady state tracking is a performance requirement at DC). Most physical systems are strictly proper, which means that the interconnect $G(s)K(s)$ will be a strictly proper transfer function. Therefore, $G(j\omega)K(j\omega) \to 0$ as $\omega \to \infty$, which means that $S(j\omega) \to 1$ at higher frequencies. As a result, it is difficult to require much performance at higher frequencies.

2.6 SISO Robust Analysis

In this section, we build off the ideas of nominal stability (NS) and nominal performance (NP) from the previous section to develop robust stability (RS) and robust performance (RP) criteria that may be used to get a better understanding of how a controller will perform on an actual system. For ease of reading, the transfer function

dependence on s will be dropped for this section, but the dependence is still implied (e.g., the notation G will be used in place of $G(s)$, but the two functions are identical).

In practice, the finite dimensional LTI plant model G will never fully "capture" all of the dynamics of an actual plant. This is due to the fact that the actual plant is infinite dimension and may contain nonlinearities. However, lower order models are usually able to capture most of the actual plant dynamics, which is enough to synthesize controllers that will be effective on the actual plant. For robustness analysis, we quantify a set of plants, labeled $\tilde{G}$, that contains the nominal plant G along with a description of the un-modeled dynamics of the actual plant. For the cases considered here, the set of plants $\tilde{G}$ is defined in terms of the nominal plant G and a perturbation, labeled Δ. Two common perturbed plant descriptions are additive (i.e., $\tilde{G} = G + \Delta$) and multiplicative (i.e., $\tilde{G} = (1 + \Delta)G$); however other configurations do exist. For more information, see [42; 43; 45].

For robust stability, stability must be guaranteed for every plant in the set $\tilde{G}$. The tests for robust stability are specific to the formulation of the set of plants $\tilde{G}$. Nominal stability is required for robust stability; however, the robust stability criteria discussed next do not explicitly check for nominal stability. Therefore, nominal stability must be checked separately before applying the robust stability tests.

For the discussion here, robust performance follows a similar path to nominal performance in that it requires both robust stability and some weighted performance criteria for all plants in the set $\tilde{G}$. Two methods for forming $\tilde{G}$ and their robust stability and robust performance criteria are discussed next. These methods are taken from [42].

2.6.1 Additive Uncertainty

One method for robust analysis is to consider a perturbed plant with additive uncertainty. The set of additively perturbed transfer functions may be defined as

$$\tilde{G} = \{G + W_2\Delta\}. \tag{2.17}$$

Here, W_2 is a fixed stable transfer function, and an allowable Δ is any stable transfer function satisfying $\|\Delta\|_\infty < 1$. A block diagram for a standard feedback controller on a plant with additive uncertainty is shown in Figure 2.3.

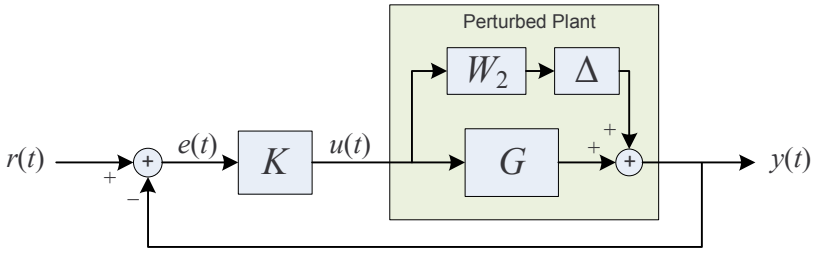

Figure 2.3: Traditional LTI Feedback Architecture with Additive Uncertainty

The perturbed sensitivity function for a plant with additive uncertainty is given by

$$\begin{aligned} \tilde{S} &= \frac{1}{1 + (G + \Delta W_2)K} \\ &= \frac{1/(1+GK)}{(1 + GK + \Delta W_2 K)/(1+GK)} \qquad (2.18) \\ &= \frac{S}{1 + \Delta W_2 K S} \qquad (2.19) \end{aligned}$$

For robust stability, the closed-loop poles of perturbed system must have $\mathrm{Re}(s) < 0$ (i.e., the closed-loop poles must in the left half plane). This is addressed in the next theorem that uses the methods provided in [42].

Theorem 1 (DFT [42])**.** *(Additive uncertainty model) Assuming that K provides nominal (closed-loop) stability for the block diagram in Figure 2.2, K provides robust stability for the diagram in Figure 2.3 iff*

$$\|W_2 K S\|_\infty < 1.$$

Proof. ($\Leftarrow$) Assume that $\|W_2KS\|_\infty < 1$. Construct the Nyquist plot of $1 + L = 1 + GK$, indenting the contour to the left around poles on the imaginary axis. Since nominal stability is assumed, we know that there are no unstable pole cancelations when forming $L = GK$, the Nyquist plot of $1 + L$ does not pass through the origin, and its number of counterclockwise encirclements of the origin equals the number of poles of L in $\mathrm{Re}(s) \geq 0$.

Fix an allowable Δ. Construct the Nyquist plot of $1 + \tilde{G}K = 1 + (G + \Delta W_2)K = 1 + GK + \Delta W_2 K$. No additional contour indentations are needed since $W_2\Delta$ introduces no additional imaginary axis poles. We have to show that the Nyquist plot of $1 + GK + \Delta W_2 K$ does not pass through the origin and its counterclockwise encirclements of the origin equals the number of poles of $G + W_2\Delta$ in $\mathrm{Re}(s) \geq 0$ plus the number of poles K in $\mathrm{Re}(s) \geq 0$. This is equivalent to requiring that the Nyquist plot of $1 + GK + \Delta W_2 K$ does not pass through the origin and encircles it exactly as many times as the Nyquist plot of $1 + L$. In other words, the perturbation does not change the number of origin encirclements. To see this, observe from the derivation in eqn (2.18) that

$$1 + (G + \Delta W_2)K = (1 + L)(1 + \Delta W_2 KS) \tag{2.20}$$

Since $\|\Delta W_2 KS\|_\infty \leq \|W_2 KS\|_\infty < 1$, the point $1 + \Delta W_2 KS$ always lies in some closed disk with center 1, radius < 1, for points s on the Nyquist contour. Therefore, the disk does *not* include the origin. Thus, as s goes around Nyquist contour, the net change in the angle of $1 + (G + \Delta W_2)K$ equals the net change in the angle of $1 + L$, which means that the perturbation does not change the number of origin encirclements. This gives the desired result.

($\Rightarrow$) Suppose that $\|W_2 KS\|_\infty \geq 1$. We will construct an allowable Δ that destabilizes the feedback system. When K is strictly proper (and S is proper), KS is strictly proper, which means that at some frequency ω_0,

$$|W_2(j\omega_0)K(j\omega_0)S(j\omega_0)| = 1. \tag{2.21}$$

It was shown in Section 2.3.1 that a stable all-pass filter may be constructed such that

$$\Delta(j\omega_0)W_2(j\omega_0)K(j\omega_0)S(j\omega_0) = -|W_2(j\omega_0)K(j\omega_0)S(j\omega_0)| = -1. \tag{2.22}$$

This Δ is allowable and

$$1 + \Delta(j\omega_0)W_2(j\omega_0)K(j\omega_0)S(j\omega_0) = 0. \tag{2.23}$$

From eqn (2.20), the Nyquist plot of $1 + (G + \Delta W_2)K$ passes through the origin, so the perturbed feedback system is not internally stable. It should be noted that when K is biproper, a limiting argument may be used to arrive at the same result. □

Robust performance requires both robust stability and that the weighted perturbed sensitivity function be less than one for all $\|\Delta\|_\infty \leq 1$ (i.e., $\|W_1\tilde{S}\|_\infty < 1 \quad \forall \quad \|\Delta\|_\infty \leq 1$). This is stated more precisely as

$$\|W_2KS\|_\infty < 1 \quad \text{and} \quad \left\| \frac{W_1S}{1 + \Delta W_2KS} \right\|_\infty < 1, \quad \forall \text{ allowable } \Delta \tag{2.24}$$

With a slight abuse of notation, these criteria may be expressed as a condition on the function

$$s \mapsto |W_1(s)S(s)| + |W_2(s)K(s)S(s)|, \tag{2.25}$$

which is denoted $|W_1S| + |W_2KS|$. A necessary and sufficient condition for robust performance, which is proven next, is given by

$$\||W_1S| + |W_2KS|\|_\infty < 1, \tag{2.26}$$

or equivalently,

$$|W_1(j\omega)S(j\omega)| + |W_2(j\omega)K(j\omega)S(j\omega)| < 1 \quad \forall\omega \cup \infty. \tag{2.27}$$

This may be further expressed as

$$\begin{aligned} & |W_1(j\omega)S(j\omega)| < 1 - |W_2(j\omega)K(j\omega)S(j\omega)| \quad \forall\omega && (2.28)\\ \iff \quad & \frac{|W_1(j\omega)S(j\omega)|}{1 - |W_2(j\omega)K(j\omega)S(j\omega)|} < 1 \quad \forall\omega && (2.29)\\ \iff \quad & \left\| \frac{W_1 S}{1 - |W_2 K S|} \right\|_\infty < 1. && (2.30) \end{aligned}$$

Note that $1 - |W_2(j\omega)K(j\omega)S(j\omega)| > 0 \ \forall\omega$ is satisfied by the robust stability requirement. Therefore, the inequality in eqn (2.29) holds.

Theorem 2 (DFT [42])**.** *(Additive uncertainty model) A necessary and sufficient condition for robust performance is*

$$\| |W_1 S| + |W_2 K S| \|_\infty < 1, \tag{2.31}$$

Proof. ($\Leftarrow$) Assume eqn (2.31), or equivalently,

$$\|W_2 K S\|_\infty < 1 \quad \text{and} \quad \left\| \frac{W_1 S}{1 - |W_2 K S|} \right\|_\infty < 1 \tag{2.32}$$

Fix Δ and assume that each of the transfer functions are evaluated at an arbitrary point $j\omega$. Then,

$$1 = |1 + \Delta W_2 K S - \Delta W_2 K S| \leq |1 + \Delta W_2 K S| + |W_2 K S|$$

and therefore

$$1 - |W_2 K S| \leq |1 + \Delta W_2 K S|.$$

This implies that

$$\left\| \frac{W_1 S}{1 - |W_2 K S|} \right\|_\infty \geq \left\| \frac{W_1 S}{1 + \Delta W_2 K S} \right\|_\infty .$$

This along with eqn (2.32) yields

$$\left\| \frac{W_1 S}{1 + \Delta W_2 K S} \right\|_\infty < 1. \tag{2.33}$$

($\Rightarrow$) Assume that

$$\|W_2 K S\|_\infty < 1 \quad \text{and} \quad \left\| \frac{W_1 S}{1 + \Delta W_2 K S} \right\|_\infty < 1, \quad \forall \text{ allowable } \Delta. \tag{2.34}$$

Pick a frequency ω_0 where

$$\frac{|W_1 S|}{1 - |W_2 K S|} \tag{2.35}$$

is maximum. Now pick any Δ such that

$$1 - |W_2 K S| = |1 + \Delta W_2 K S|.$$

This amounts to choosing an allowable Δ that satisfies

$$\Delta(j\omega_0) = -\angle W_2(j\omega_0) K(j\omega_0) S(j\omega_0) \tag{2.36}$$

One way to achieve this is to choose an all-pass Δ using the methods in Section 2.3.1. Now,

$$\begin{aligned} \left\| \frac{W_1 S}{1 - |W_2 K S|} \right\|_\infty &= \frac{|W_1 S|}{1 - |W_2 K S|} \quad \text{at } \omega_0 \\ &= \frac{|W_1 S|}{1 + \Delta W_2 K S} \quad \text{at } \omega_0 \\ &\leq \left\| \frac{W_1 S}{1 + \Delta W_2 K S} \right\|_\infty . \end{aligned} \tag{2.37}$$

This finishes the necessity part of the proof. □

2.6.2 Multiplicative Uncertainty

Another method for robust analysis is to consider a perturbed plant with multiplicative uncertainty. The set of perturbed transfer functions with multiplicative uncertainty may be defined as

$$\tilde{G} = \{(1 + W_2\Delta)G : \Delta \text{ is "allowable"}\}. \tag{2.38}$$

As before, W_2 is a fixed stable transfer function, and Δ is variable stable transfer function satisfying $\|\Delta\|_\infty \leq 1$. In the case of multiplicative uncertainty, a Δ is said to be "allowable" if it satisfies this norm bound and if there are no unstable pole/zero cancelations when forming $\tilde{G}$. A block diagram for a standard feedback controller on a plant with additive uncertainty is shown in Figure 2.4.

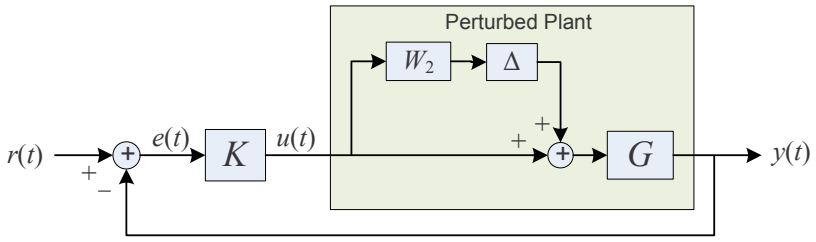

Figure 2.4: Traditional LTI Feedback Architecture with Multiplicative Uncertainty

The perturbed sensitivity function for a plant with multiplicative uncertainty is given by

$$\begin{aligned} \tilde{S} &= \frac{1}{1 + (1 + \Delta W_2)GK} \\ &= \frac{1/(1 + GK)}{(1 + GK + \Delta W_2 GK)/(1 + GK)} \\ &= \frac{S}{1 + \Delta W_2 T} \end{aligned} \tag{2.39}$$

For robust stability, the closed-loop poles of perturbed system must have $\mathrm{Re}(s) < 0$ (i.e., the closed-loop poles must in the left half plane). This is addressed in the next theorem that appears in [42].

Theorem 3 (DFT [42]). *(Multiplicative uncertainty model) Assuming that K provides nominal (closed-loop) stability for the block diagram in Figure 2.2, K provides robust stability for the diagram in Figure 2.4 iff*

$$\|W_2T\|_\infty < 1.$$

Proof. The proof of this is very similar to the additive uncertainty proof in that it uses a Nyquist argument to provide sufficiency and constructs a destabilizing Δ to show necessity. A full proof of this appears on pages 44-45 in [42]. □

Similar to the additive uncertainty case, the robust performance condition for systems with multiplicative uncertainty is given by the following theorem.

Theorem 4 (DFT [42]). *(Multiplicative uncertainty model) A necessary and sufficient condition for robust performance is*

$$\||W_1S| + |W_2T|\|_\infty < 1, \tag{2.40}$$

Proof. The proof of this is very similar to the additive uncertainty proof. For sufficiency, you fix an allowable Δ and show the norm bound holds (across all frequency). For necessity, you fix ω and show that all allowable Δs will satisfy the norm bound (for an arbitrary frequency ω). A full proof of this appears on pages 47-48 in [42]. □

2.6.3 Delay Uncertainty

A system with a time delay may be expresses as

$$G_d = Ge^{-s\tau_d}, \tag{2.41}$$

where G is a delay free stable transfer function.

If there is a mismatch in time delay, the perturbed plant is defined as $\tilde{G}_d = Ge^{-s(\tau_d+\Delta\tau_d)}$, which results in the perturbed sensitivity function

$$\begin{aligned}\tilde{S} &= \frac{1}{1+Ge^{-s(\tau_d+\Delta\tau_d)}K} \\ &= \frac{1/(1+GK)}{(1+Ge^{-s(\tau_d+\Delta\tau_d)}K)/(1+GK)} \\ &= \frac{1+GK}{1+Ge^{-s(\tau_d+\Delta\tau_d)}K}S. \end{aligned} \tag{2.42}$$

From this, weighted performance analysis can be computed in the frequency domain by evaluating $S(j\omega)$ at individual frequencies. However, the robustness analysis from previous section and the later controller synthesis require that the uncertainty be described using rational transfer functions. In order to create this rational perturbation, we note that this time delay uncertainty may be expressed using multiplicative uncertainty. This is shown in Figure 2.5.

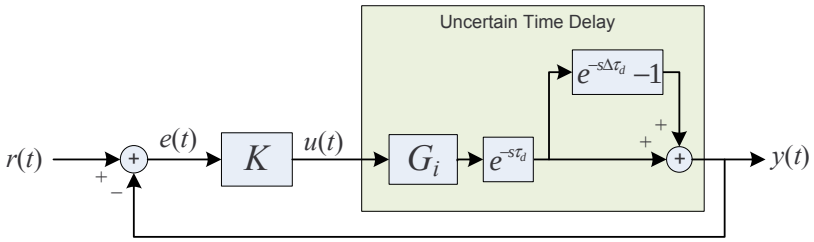

Figure 2.5: Traditional LTI Feedback Architecture with Time Delay Uncertainty

Now, a multiplicative uncertainty weight needs to be used that "covers" $e^{-s\Delta\tau_d}-1$. Many ways for covering this uncertainty were given in [46] and their trade-offs were examined. For the discussion here, we will pick our own weight based on the first order Padé approximation

$$e^{-s\Delta\tau_d} \approx \frac{1-\frac{\Delta\tau_d}{2}s}{1+\frac{\Delta\tau_d}{2}s}. \tag{2.43}$$

From this, an initial guess for the uncertainty weight is

$$W_2(s) = \frac{-\Delta\tau_d s}{1+\frac{\Delta\tau_d}{2}s}\left(\approx e^{-s\Delta\tau_d}-1\right). \tag{2.44}$$

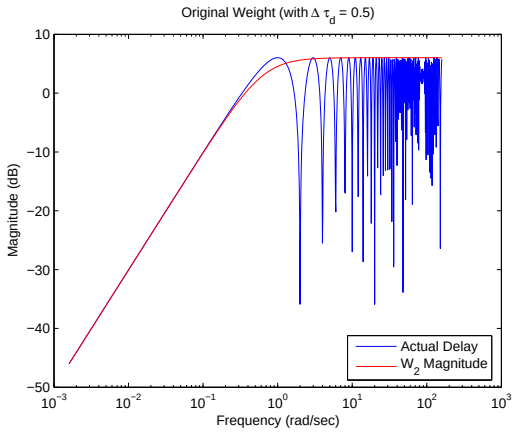

Figure 2.6: Original Guess for a $W_2(s)$

However, this weight does not "cover" $e^{-s\Delta\tau_d} - 1$. This is shown in Figure 2.6

This may be addressed by picking a weight of the form

$$W_2(s) = \frac{2\beta s}{s + \frac{2}{\alpha\Delta\tau_d}}. \tag{2.45}$$

Now, the parameters α and β may be adjusted to cover $e^{-s\Delta\tau_d} - 1$. An example set of parameters that were empirically tuned are $\alpha = 2.05$ and $\beta = 1.05$. The resulting weight is shown in Figure 2.7 and will be used in later robust controller synthesis designs.

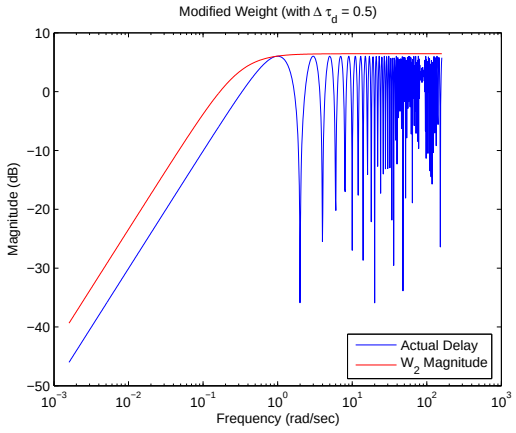

Figure 2.7: Example $W_2(s)$ that will Cover an Uncertain Delay

2.7 Complex μ Analysis and Synthesis

In the previous section, we isolated a single tracking performance control objective for a set of perturbed single-input single-output (SISO) systems. For multiple-input multiple-output (MIMO) systems (and for SISO or MIMO systems with multiple control objectives), a more general robustness framework is needed. In November 1982, two papers were published in the same issue of *IEE Proceedings, Part D* by John Doyle [47] and Michael Safonov [48] that provided a general framework to handle such systems. In Doyle's paper [47], he defined the structured singular value (SSV), which he labeled μ, and Safonov [48] defined the multivariable stability margin, which he labeled k_m. From the definition of these two terms, it was clear that $\mu = 1/k_m$ and that these two methods developed similar tools for analyzing robustness. For the discussion here, we only consider the complex SSV as it was defined by Doyle in [47]. Over a decade after Doyle's initial paper, a comprehensive and thorough treatment of the complex SSV was provided in [49]. All of the information from this section was taken from [50; 51; 52; 49; 43; 45].

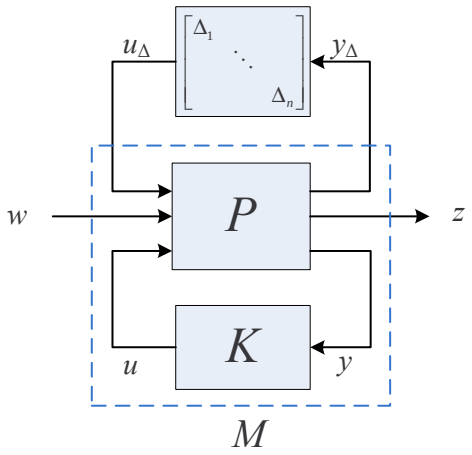

Figure 2.8: Standard Robust Controller Canonical System Interconnection

It may shown that many LTI interconnect may be rearranged into the standard canonical form shown in Figure 2.8. In this configuration, P is the generalized plant

that contains the nominal plant and all of the weighted criteria that are used to analyze and synthesize the controller K. In the previous section, the weighted criteria were the model uncertainty weight $W_2(s)$ and tracking performance weight $W_1(s)$. In a multiple objective control problem, tracking performance will not be the only weighted performance criteria. In this case, signals such as the control authority may also be weighted, and these weights are incorporated into the generalized plant model P in Figure 2.8. Δ_i for $i = 1 \ldots m$ are the block uncertainties that make up the structured perturbation $\Delta = \text{block diag}(\Delta_1, \ldots, \Delta_m)$. The set of all allowable Δ's is given by:

$$\mathbf{\Delta} = \{\Delta := \text{block diag}(\Delta_1, \ldots, \Delta_m), \Delta_i \in \mathbb{C}^{n_i \times n_i}\}, \tag{2.46}$$

where $n_i \times n_i$ are the dimensions of the perturbation Δ_i.

Using linear fractional transforms (LFTs), the interconnect shown in Figure 2.8 may be expanded as

$$\begin{aligned} M &= F_L(P, K) = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21} && (2.47) \\ z &= F_U(M, \Delta)w, && (2.48) \end{aligned}$$

where

$$F_U(M, \Delta) = M_{22} + M_{21}\Delta(I - M_{11}\Delta)^{-1}M_{12}. \tag{2.49}$$

Here, $F_L(\cdot)$ and $F_U(\cdot)$ are the lower and upper LFTs, respectively. In these definitions, the generalized open-loop plant P and the nominal closed-loop system M are partitioned as

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \tag{2.50}$$

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}, \tag{2.51}$$

$$\tag{2.52}$$

where the block partitions are obvious from the problem formulation.

For robustness analysis of a matrix with structured uncertainties such as those in eqn (2.46), we use the structured singular value μ. The structured singular value for a complex valued matrix M is given by

$$\mu(M) = \frac{1}{\min\limits_{\Delta \in \mathbf{\Delta}} \{\overline{\sigma}(\Delta) : \det(I - M\Delta) = 0\}}, \tag{2.53}$$

and if there is no $\Delta \in \mathbf{\Delta}$ that makes $I - M\Delta$ singular, we define $\mu(M) = 0$.

Note that $\min\limits_{\Delta \in \mathbf{\Delta}} \{\overline{\sigma}(\Delta) : \det(I - M\Delta) = 0\}$ is a size (measured via $\overline{\sigma}$) of the smallest "destabilizing" perturbation in the set $\mathbf{\Delta}$, i.e., one that makes $\det(I - M\Delta) = 0$. Therefore, μ may be interpreted as the reciprocal of the size of the smallest destabilizing perturbation.

In practice, the computation of μ is an NP-hard problem [53], which makes its exact calculation computationally very intensive. To address this, upper and lower bounds on μ have been derived that have shown to be very useful in practice (i.e., the upper and lower bounds appear to be very close to each other for many practical problems). For the robust controller analysis and synthesis considered here, we make particular use of the upper bound, which is stated next. Let $\mathbf{D}$ be the set of matrices such that $D\Delta = \Delta D, \quad \forall \ D \in \mathbf{D}$ and $\Delta \in \mathbf{\Delta}$, (i.e., the matrices D and Δ commute). In the matrix case, this is not a trivial property. For the structured perturbation set given in eqn (2.46), this set of matrices is given by

$$\mathbf{D} = \{D := \text{block diag}(d_1 I, \ldots, d_m I), d_i \neq 0\}, \tag{2.54}$$

where I denotes an identity matrix of the appropriate size (i.e., same size as the appropriate block Δ_i). Now, the (convex) μ upper bound is given by

$$\mu(M) \leq \inf_{D \in \mathbf{D}} \overline{\sigma}(DMD^{-1}), \tag{2.55}$$

which is a more tractable computation than calculating μ directly. Note that for the discussion here, that the individual Δ_i perturbations (and hence commuting $d_i I$ blocks) are assumed to be square matrices. While the theory may be extended to non-square cases, the notation becomes cumbersome. For the details on computing the μ upper bound with non-square perturbations, see [50].

To this point, μ has been defined for matrices; however the methodology may be used in the context of systems. For robustness analysis of systems, we define $M(s) \in \mathcal{RH}_\infty$ to be a known stable system that contains the generalized plant $P(s)$ connected to the (stabilizing) controller $K(s)$, and $\Delta(s) \in \mathcal{RH}_\infty$ to be an unknown, but stable, transfer function with the appropriate block structure (see eqn (2.46)). In Figure 2.8, robust performance is guaranteed if and only if $\|z(t)\|_2 \leq \|w(t)\|_2$ for all $\|\Delta(s)\|_\infty < 1$. At any given frequency ω, $M(j\omega)$ and $\Delta(j\omega)$ are matrices and μ is well understood. Using the SSV, robust performance of the system may be expressed as

$$\text{RP} \Longleftrightarrow \sup_{\omega} \mu(M(j\omega)) \leq 1. \tag{2.56}$$

It may be shown that the SISO robustness performance criteria from the previous section are a special case of this more general result.

The robust controller synthesis considered in the work presented here relies on the previously described μ framework. A procedure known as μ-synthesis is used to synthesize both feedforward and feedback controllers. μ-synthesis approximately optimizes μ via choice of K. For more details on μ-synthesis, see [50].

2.7.1 Optimal Control

For the feedforward controller designs considered in later chapters, $\mathcal{H}_2$ optimal and $\mathcal{H}_\infty$ optimal controllers are designed. For optimal control design, there are no structured uncertainties. The resulting general plant interconnect for optimal control design is shown in Figure 2.9.

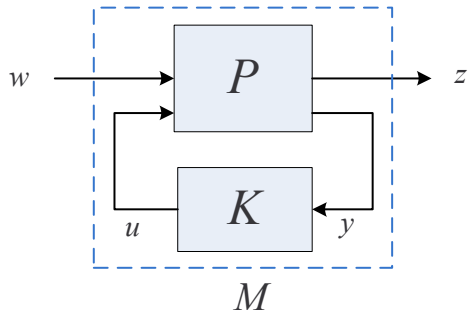

Figure 2.9: Standard Optimal Controller Canonical System Interconnection

The $\mathcal{H}_2$ optimal controller design optimizes $\|M(j\omega)\|_2$ via choice of $K(s)$. The computation of the $\mathcal{H}_2$ optimal controller involves two algebraic Riccati equations that generate a global optimum solution.

The $\mathcal{H}_\infty$ optimal controller design attempts to optimize $\|M(j\omega)\|_\infty$ via choice of $K(s)$. This method uses a decision problem (i.e., a problem with a result of "pass" or "fail") to determine if a stabilizing $K(s)$ exists with $\|M(j\omega)\|_\infty < \gamma$. Here, γ is a real positive number. Using the decision problem, a search may be performed to find the smallest γ (within a specified tolerance) for which there exists a stabilizing controller with $\|M(j\omega)\|_\infty < \gamma$. Then, a stabilizing controller $K(s)$ is calculated that satisfies $\|M(j\omega)\|_\infty = \gamma$. The result is a slightly suboptimal controller.

From the definitions of the system norms in eqns (2.11) and (2.12), it is clear that the $\mathcal{H}_2$ optimal controller minimizes all of the singular values of the closed-loop system at all frequencies, whereas the $\mathcal{H}_\infty$ optimal controller attempts to minimize the peak value of the largest singular value. From a design perspective, each method has its advantages and disadvantages. For more information on both of these design

methods, see [43; 45; 50].

2.8 Limitations on Performance

Feedback controllers are used to meet both performance requirements (such as tracking and disturbance rejection) and robustness requirements. These requirements are satisfied by appropriately placing the poles of the closed-loop transfer function

$$M(s) = \frac{G(s)K(s)}{1 + G(s)K(s)}. \tag{2.57}$$

By expressing $G(s) = \frac{N_G(s)}{D_G(s)}$ and $K(s) = \frac{N_K(s)}{D_K(s)}$, the closed-loop transfer function becomes

$$M(s) = \frac{G(s)K(s)}{1 + G(s)K(s)} = \frac{\frac{N_G(s)}{D_G(s)}\frac{N_K(s)}{D_K(s)}}{1 + \frac{N_G(s)}{D_G(s)}\frac{N_K(s)}{D_K(s)}} = \frac{N_G(s)N_K(s)}{D_G(s)D_K(s) + N_G(s)N_K(s)}. \tag{2.58}$$

Equation 2.58 illustrates some general properties about the zeros and poles of the feedback system. The closed-loop poles, namely the roots of $D_G(s)D_K(s) + N_G(s)N_K(s)$, may be placed by choosing $N_K(s)$ and $D_K(s)$. In particular, unstable poles of the plant $G(s)$ may be moved into the left-half plane, which is required for stability. However, the open-loop zeros of $G(s)K(s)$ are also zeros of the closed-loop transfer function. Left-half plane zeros in $G(s)$ may be canceled by placing poles in $K(s)$, without violating internal stability. In this case, the zeros will not appear in either the loop gain $L(s)$ or closed-loop transfer function $M(s)$. However, internal stability requires that no RHP pole/zero cancelations happen when forming $G(s)K(s)$. Therefore, the right-half plane zeros of $G(s)$ (and $K(s)$) will remain fixed in the closed-loop response. This is a fundamental property that cannot be changed by either feedforward or feedback control.

2.8.1 Right-Half Plane Zero Step Response

Stable systems with right-half plane zeros will exhibit an inverse response. For example, consider the transfer function

$$G(s) = 25\frac{-s+2}{(s+5)(s+10)}, \tag{2.59}$$

which has step response

$$s(t) = 1 - 7e^{-5t} + 6e^{-10t}. \tag{2.60}$$

For this particular example, the step response has two zeros, namely $t = 0$ and $t = \frac{\ln(6)}{5} \approx 0.3584$ seconds after the step has occurred. This may be seen by looking at the plot of the step response shown in Figure 2.10. For the plot shown here, the step occurs at $t = 1$ seconds. Therefore, the output from eqn (2.60) is delayed by one second in Figure 2.10.

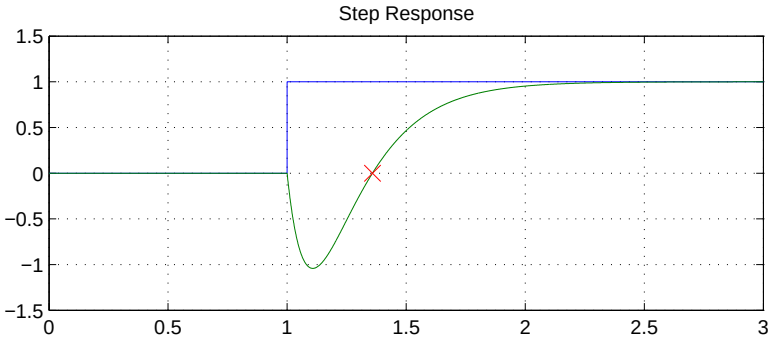

Figure 2.10: Step Response of a Stable System with One RHP Zero. The Red X Marks the Second Zero Crossing that Results from the Initial Undershoot.

This undershoot behavior seen in Figure 2.10 may be proven by the following theorem that is taken from [54].

Theorem 5. *Suppose that $G(s)$ is stable and $G(0) \neq 0$. Then its step response has an undershoot if $G(s)$ has a positive real zero.*

Proof. Let $z > 0$ be a positive real zero of $G(s)$ and $y(t)$ be the step response. Then,

$$Y(s) = G(s)\frac{1}{s} = \int_{0-}^{\infty} y(t)e^{-st}dt \tag{2.61}$$

Notice that z is in the region of convergence of the Laplace transform of $y(t)$. Then,

$$0 = G(z)\frac{1}{z} = \mathcal{L}[y(t)](z) = \int_{0-}^{\infty} y(t)e^{-zt}dt. \tag{2.62}$$

Assume $G(0) > 0$. Then, $y(\infty) = G(0) > 0$. Since $y(t)e^{-zt} > 0$ for sufficiently large t, there must exist $t_0 > 0$ such that $y(t_0)e^{-zt_0} < 0$, i.e., $y(t_0) < 0$. A similar argument applies when $G(0) < 0$.

□

2.9 Smith Predictors

Smith predictors provide a method for altering the properties in the feedback loop in order to improve performance. Originally, they were used to improve the reference tracking performance on systems with time delays [55; 1]. Since their conception, the structure and methodology for Smith predictors has matured to further improve their disturbance rejection performance and to improve their robustness properties (c.f., [1]). In this section, we will outline the various forms of a Smith predictor. In a sense, a Smith predictor acts as a cancelation controller. We will demonstrate this by analyzing the internal stability of the various structures. For this reason, many forms of the Smith predictor cannot be applied to unstable plants, since they form an unstable pole/zero cancelation in the feedback loop. There has been much work in this area and various researchers have provided their methods for compensating for this.

2.9.1 Original Smith Predictor for Time Delays

The Smith predictor was originally introduced by Otto Smith in the 1950's as a way to overcome the effects of dead-time in a stable process [55]. In general, time-delays add phase lag to systems that will reduce stability margins. In order to provide tracking for such systems, controllers (e.g., PI controllers) are made less aggressive, so they can maintain closed-loop stability. The Smith predictor seeks to remove the effect of the time-delay by predicting what the plant output will be without the time delay. In this setting, the plant model is defined as

$$\hat{G}(s) = \hat{G}_{df}(s)e^{-s\hat{\tau}_d}, \tag{2.63}$$

where $\hat{G}_{df}(s)$ is the delay free (and stable) plant model and $e^{-s\hat{\tau}_d}$ is a model of the plant time-delay. Here, the convention is that $\hat{G}(s)$ is a model of of the true plant $G(s) = G_{df}(s)e^{-s\tau_d}$. This notation is used to expose the fact that model and physical plant will differ in practice. Using this definition, A diagram of a Smith predictor for a plant with time-delay is shown in Figure 2.11.

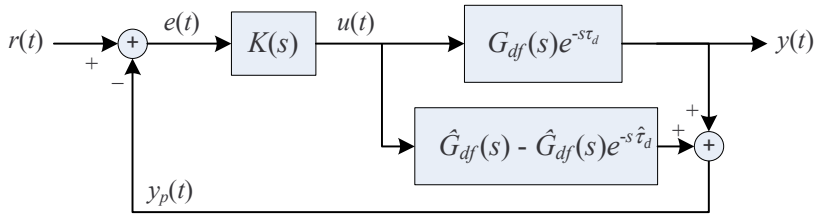

Figure 2.11: A Smith Predictor for a Stable Plant with Time-Delay

Now, a prediction of the output without delay ($y_p(t)$) is used in the feedback path. In the nominal case (i.e., when $\hat{G}(s) = G(s)$) with no external disturbances, the nominal closed-loop system is given by

$$M(s) = \frac{G(s)K(s)}{1 + K(s)(G(s) + G_{df}(s) - G(s))} = \frac{G(s)K(s)}{1 + G_{df}(s)K(s)}. \tag{2.64}$$

This shows that the feedback loop (i.e., $1 + G_{df}(s)K(s)$) does not contain the time delay $e^{-s\tau_d}$. This removal of the time delay from the feedback loop may greatly improve the tracking performance on plants with dead-time. However, the main limitations to this method are

1. The performance can degrade quickly if there is a mismatch in time delay (i.e., $\hat{\tau}_d \neq \tau_d$).

2. The disturbance rejection performance is dominated by the slowest time constant of the plant.

The last limitation may be overcome by using the modified Smith predictor discussed on in the next section. For more information on the limitations of Smith predictors, see [1].

2.9.2 The Modified Smith Predictor

The modified Smith predictor is a generalization of the original Smith predictor that uses a (potentially) different plant model. The modification allows for improved disturbance rejection and can be used to remove other plant dynamics (e.g., right-half plane zeros) from the feedback path. An example implementation of a modified Smith predictor is shown in Figure 2.12.

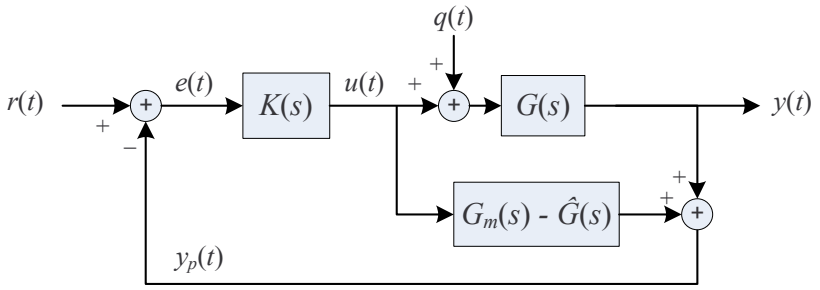

Figure 2.12: A Modified Smith Predictor for a Stable Plant

In this setting, $\hat{G}(s)$ is the full plant model (with potential time delays), $G_m(s)$ is the modified plant model (without time delays), and $K(s)$ is the feedback controller.

In the nominal case (i.e., $\hat{G}(s) = G(s)$) with no disturbances, the predicted signal being fed back is the output of $G_m(s)$, and the closed-loop transfer function from $r(t)$ to $y(t)$ is given by

$$M(s) = \frac{Y(s)}{R(s)} = \frac{G(s)K(s)}{1 + K(s)(G(s) + G_m(s) - G(s))} = \frac{G(s)K(s)}{1 + G_m(s)K(s)}. \tag{2.65}$$

For the controller architecture that we will present later, we use a specific $G_m(s)$ to achieve our desired results. This will be presented in Chapter 5.

Let $q(t)$ be a disturbance on the plant input shown in Figure 2.12, $G(s) = \frac{N_G(s)e^{-s\tau_d}}{D_G}$, $G_m(s) = \frac{N_m(s)}{D_m(s)}$, and $G_K(s) = \frac{N_K(s)}{D_K(s)}$. Then, the response to the disturbance $q(t)$ is given by

$$\begin{aligned}
\frac{Y(s)}{Q(s)} &= \frac{G(s)(1 + G_m(s)K(s) - G(s)K(s))}{1 + G_m(s)K(s)} \\
&= G(s)\left(1 - \frac{G(s)K(s)}{1 + G_m(s)K(s)}\right) \\
&= \frac{N_G(s)e^{-s\tau_d}}{D_G^2(s)}\left(\frac{D_G(s)D_m(s)D_K(s) - N_G(s)N_K(s)D_m(s)e^{-s\tau_d}}{D_m(s)D_K(s) + N_m(s)N_K(s)}\right)
\end{aligned} \tag{2.66}$$

This transfer function exposes one of the limitations of a Smith predictor, which is that the poles of the original system $G(s)$ may appear as poles in the disturbance response. In particular, if there are slow poles in the plant, they have the potential to dominate the disturbance rejection performance (i.e., the amount of time to correct for a disturbance is dominated by the slowest pole in the original plant). One method for dealing with this is to design $G_m(s)$ such that the slow poles of $G(s)$ are canceled in the formulation $G(s)(1 + G_m(s)K(s) - G(s)K(s))$, which results in a form of cancelation control. For stable pole/zero cancelations, this often works well [1]. It has been argued that the same methodology may be extended to handle unstable systems by putting constraints on $K(s)$ and $G_m(s)$ [1; 56]. These methods are discussed later in

the section on internal stability.

2.9.3 The Unified Smith Predictor

The unified Smith predictor is a general formulation of the previously discussed Smith predictors. By using this formulation, we will show that the previous two Smith predictors are special cases of the unified Smith predictor. An example implementation of a unified Smith predictor is shown in Figure 2.13.

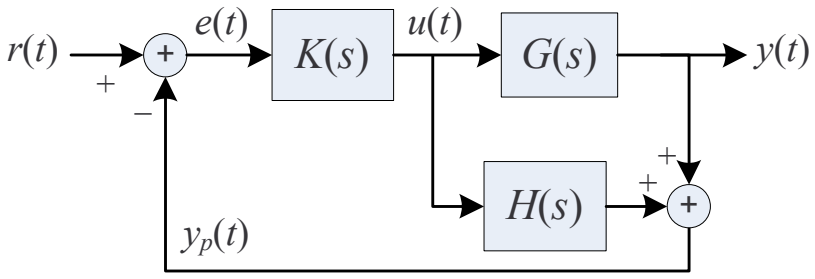

Figure 2.13: An Unified Smith Predictor

Here, $H(s)$ and $K(s)$ are design parameters that are picked to meet a certain design objective. For the next discussion, we will assume the nominal case (i.e., perfect plant models). Some common choices of $H(s)$ are:

$$
\begin{aligned}
H(s) = \hat{G}_{df}(s) - \hat{G}(s) &\quad : \quad \text{Original Smith Predictor} \\
H(s) = G_m(s) - \hat{G}(s) &\quad : \quad \text{Modified Smith Predictor} \\
H(s) = -\hat{G}(s) &\quad : \quad \text{Internal Model Control}
\end{aligned}
$$

Using the factorizations $\hat{G}(s) = \frac{\hat{N}_{nmp}(s)\hat{N}_{mp}(s)}{\hat{D}_u(s)\hat{D}_s(s)}e^{-s\hat{\tau}_d}$, $\check{G}_{df}(s) = \frac{\hat{N}_{nmp}(s)\hat{N}_{mp}(s)}{\hat{D}_u(s)\hat{D}_s(s)}$ and $G_m(s) = \frac{N_m(s)}{D_m(s)}$, these relations may also be expressed as

$$H(s) = \frac{\hat{N}_{mp}(s)\hat{N}_{nmp}(s) - \hat{N}_{mp}(s)\hat{N}_{nmp}(s)e^{-s\hat{\tau}_d}}{\hat{D}_s(s)\hat{D}_u(s)} \quad : \text{ Original Smith Predictor}$$

$$H(s) = \frac{N_m(s)\hat{D}_s(s)\hat{D}_u(s) - D_m(s)\hat{N}_{mp}(s)\hat{N}_{nmp}(s)e^{-s\hat{\tau}_d}}{D_m(s)\hat{D}_s(s)\hat{D}_u(s)} \quad : \text{ Modified Smith Predictor}$$

$$H(s) = \frac{-\hat{N}_{mp}(s)\hat{N}_{nmp}(s)e^{-s\hat{\tau}_d}}{\hat{D}_s(s)\hat{D}_u(s)} \quad : \text{ Internal Model Control}$$

These methods work well when $G(s)$ is stable. In these cases, the only restriction on $K(s)$ is that it must be stable. However, when $G(s)$ is unstable, $1 + K(s)H(s)$ is also required to have zeros at the location of the unstable poles of $G(s)$ in order to guarantee internal stability [56]. This restriction can severely limit the achievable performance [56]. In the case of the modified Smith predictor, some of these restrictions are overcome by designing $G_m(s)$ appropriately [1]. However, these methods can lead to implementation issues. Some of these issues are addressed in the next section.

2.9.4 Internal Stability

Internal stability of an LTI closed-loop system requires that all of the poles of the closed-loop system be in the left half plane (i.e., $\text{Re}(s) < 0$) *and* that there are no unstable pole/zero cancelations between the product of LTI systems in the feedback loop. By their very nature, Smith predictors are cancelation controllers that attempt to cancel all of the poles in a plant. This nature of Smith predictors makes them unsuitable for implementation on unstable systems. In this section, we outline the nominal stability conditions for controlling unstable systems using Smith predictors that are currently present in the literature and demonstrate their limitations. In the work presented in later chapters, we provide a model-based method that is more promising for controlling unstable systems. To begin the discussion here, consider the equivalent Smith predictor implementations shown in Figure 2.14.

Now, an equivalent Smith predictor may be expressed as the single feedback controller

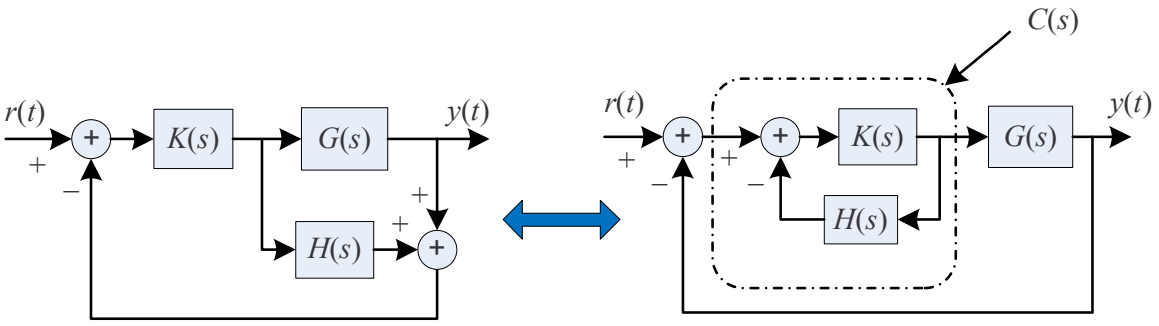

Figure 2.14: Equivalent Smith Predictor Implementations

$$C(s) = \frac{K(s)}{1 + K(s)H(s)}. \tag{2.67}$$

For internal stability, we require that there are no unstable pole/zero cancelations when forming $G(s)C(s)$. Using the previously established notation, we can define the equivalent feedback (Smith predictor) controller as

$$C(s) = \frac{\frac{N_K(s)}{D_K(s)}}{1 + \frac{N_K(s)}{D_K(s)}\frac{N_H(s)}{D_H(s)}} = \frac{N_K(s)D_H(s)}{D_K(s)D_H(s) + N_K(s)N_H(s)}. \tag{2.68}$$

It is clear from this representation that poles of $H(s)$ (i.e., the roots of $D_H(s)$) may appear as zeros of $C(s)$. If the unstable poles of $H(s)$ are also the unstable poles of $G(s)$, there is the potential for an unstable pole/zero cancelation when forming the loop gain $G(s)C(s)$, which would violate internal stability. In the previous section, we showed that the common choices of $H(s)$ will contain the same poles as $G(s)$, so this issue is present in the problem formulations considered here. It is possible to design $K(s)$ such that equivalent $C(s)$ does not have RHP zeros at the roots of $D_H(s)$. While this works in theory, there are implementation issues that that keep this from being a viable option in practice. To see this, let $H(s)$ (and also $G(s)$) have RHP poles at $s = p_k = 1 \ldots N$. Then, $D_H(p_k) = 0$. In order for $C(p_k) \neq 0$, we need the following condition

$$D_K(p_k)D_H(p_k) + N_K(p_k)N_H(p_k) = 0, \tag{2.69}$$

or equivalently,

$$1 + K(p_k)H(p_k) = 0. \tag{2.70}$$

This restriction limits the behavior of $K(s)$ near $s = p_k$. As a result, this can seriously limit the performance and robustness of the feedback system. A formal proof of this along with the controller limitations for the IMC case is provided in [56]. To help illustrate this point, let's consider the case of IMC for the unstable plant

$$G(s) = \frac{1}{s-1} \tag{2.71}$$

For this plant, a stabilizing proportional controller is given by

$$C(s) = K_p, \qquad K_p > 1. \tag{2.72}$$

Note that for IMC, $H(s) = -G(s) = \frac{-1}{s-1}$. Using eqn 2.67, the feedback controller $K(s)$ may be solved for as

$$K(s) = \frac{C(s)}{1 - C(s)H(s)} = \frac{K_p}{1 + \frac{K_p}{(s-1)}} = \frac{K_p(s-1)}{s + (K_p - 1)} \tag{2.73}$$

Notice that the condition for closed-loop stability also results in $K(s)$ being stable. For IMC (and for Smith predictors), it is well known that $K(s)$ must be stable in order for the closed-loop system to be stable. This is proven in [56].

In eqn 2.70, the condition for closed-loop stability is that $1 + K(s)H(s) = 0$ when $s = 1$. To verify this, note that

$$1 + K(s)H(s) = 1 + \frac{K_p(s-1)}{s + (K_p - 1)} \frac{-1}{s-1} = 1 - \frac{K_p}{s + (K_p - 1)}. \tag{2.74}$$

This means that

$$1 + K(1)H(1) = 1 - \frac{K_p}{1 + (K_p - 1)} = 0, \tag{2.75}$$

and the nominal stability requirement is met. A simulation diagram for this is shown in Figure 2.15.

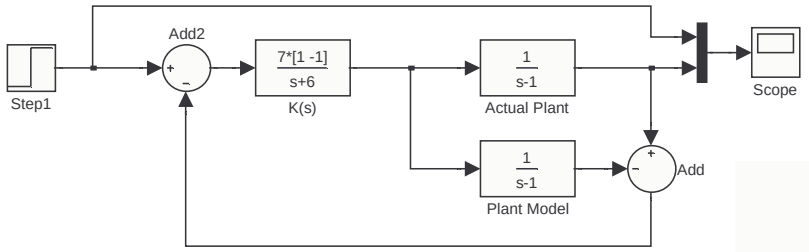

Figure 2.15: Interconnect for the Unstable IMC Example with $C(s) = K_p = 7$

The simulation of this to a step response with $K_p = 7$ is shown in Figure 2.16. It should be noted that this proportional controller does not provide very good steady state performance ($\approx 14.3\%$ steady-state tracking error). This example is not designed to provide good performance, but rather provide a numerical example of the nominal stability conditions outlined to this point.

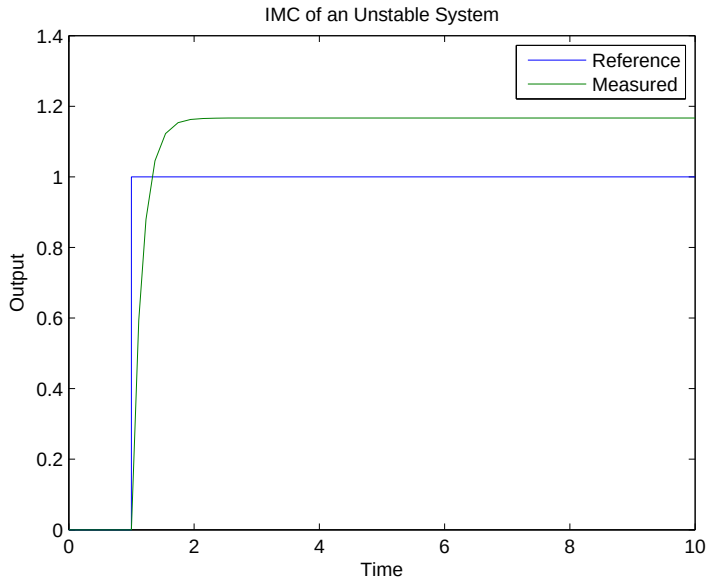

Figure 2.16: Example Simulation for the Unstable IMC Example with $C(s) = K_p = 7$

In eqn 2.74, there is an unstable pole/zero cancelation between $K(s)$ and $H(s)$. Even though these are both user defined systems (and therefore known exactly), there

is an important practical implementation issue, namely it violates internal stability! In particular, this structure assumes that the same exact control input ($u(t)$) will be the input to both the "Plant Model" and the "Actual Plant" in Figure 2.16. For the example here, nominal stability makes this assumption (i.e., that the two systems see the same exact input signal). However, this method of implementation does not satisfy internal stability. When there is any difference between the input signals to the "Plant Model" and the "Actual Plant", the unstable mode in the "Plant Model" (i.e., in the $H(s)$ for this example) will be excited and the closed-loop system will go unstable. With the current example, there is always noise in the form of numerical integration errors. Eventually, these errors will excite the unstable mode in the "Plant Model" and the closed-loop system will go unstable. This is demonstrated in Figure 2.17 where the same diagram from Figure 2.15 is simulated for a longer time. After about 35 seconds of simulation time, the numerical integration errors on the control signal ($u(t)$) excite the unstable mode in the "Plant Model" ($H(s)$). For this very reason, this method will not be pursued further here.

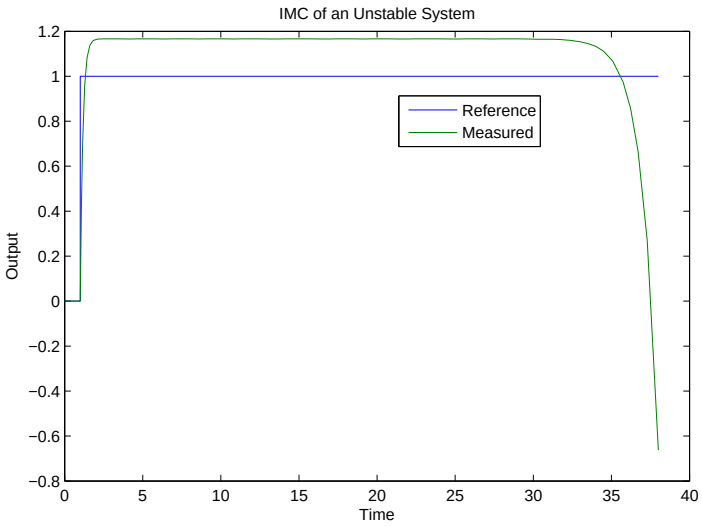

Figure 2.17: Example Simulation for the Unstable IMC Example with $C(s) = K_p = 7$

For the modified Smith predictor, methods have been proposed that remove the

unstable pole in $H(s)$ by choosing $G_m(s)$ appropriately (c.f.,[1] and the references therein). To see this, let p_k be an unstable pole in $G(s)$ (i.e., $D_G(p_k) = 0$ and $\mathrm{Re}(p_k) \geq 0$). Then, if $G_m(s)$ is defined such that

$$N_m(p_k) - N_G(p_k)e^{-p_k\tau_d} = 0, \tag{2.76}$$

there will not be an unstable pole at $H(p_k)$. In [1], they acknowledge the fact that this method may not be implemented using a transfer function representation of $G_m(s)$. This is true even in the nominal case. In [1], they provide an alternate implementation that suffers from its own limitations. Other approaches have tried designing $H(s)$ such that it does not contain any unstable poles (c.f., [57]). The design of Smith predictors for unstable plants remains a topic of ongoing research and will not be pursued further in the work presented here. Smith predictors are fundamentally used to improve reference tracking, and we have our own approach for doing this. Since Smith predictors are essentially cancelation controllers, they have a fundamental limitation that does not make them well suited for unstable plants and we will avoid using them in these situations. In light of this, we think that our approach in Chapter 4 that improves reference tracking is more promising for unstable plants.

2.10 Observers

Observers are used to estimate the internal state of a system. These state estimates have many uses such as providing state information for state feedback controllers and for model based feedforward control, the latter of which will be used in Chapter 11. States may be estimated by considering the state-space representation of a linear time invariant (LTI) model, namely

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du. \end{aligned} \tag{2.77}$$

An estimate of the state variable x is created by explicitly creating the model in equation (2.77). This estimated state variable is labeled $\hat{x}$, and the estimated output that it produces is $\hat{y}$. The state space equations for the modeled system state are

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu \\ \hat{y} &= C\hat{x} + Du \end{aligned} \tag{2.78}$$

Here, the actual input to the system, namely u, is used to produce the state and output estimates, namely $\hat{x}$ and $\hat{y}$, respectively. In practice, the initial conditions are unknown (i.e., $x(0)$ is unknown). After an initial startup period, the observer tracks the state vector $\hat{x}$, which overcomes this issue. Also, there are modeling errors in the A, B, C, D matrices along with external disturbances that will cause the estimated output $\hat{y}$ and measured output y to diverge away from each other over time. As a result, the state estimate $\hat{x}$ will diverge away from the actual state x. For the specified model in eqn (2.78), an observer may be used to track an estimate of x. A block diagram for an observer that asymptotically tracks x is shown in Figure 2.18. The part of the block diagram that implements equation (2.78) is surrounded by a dotted box.

In order to track the state x, the error signal between the estimated output $\hat{y}$ and the measured (actual) output y is used to correct the state estimate $\hat{x}$ through an observer gain (L). The closed-loop model of $\hat{x}$ is

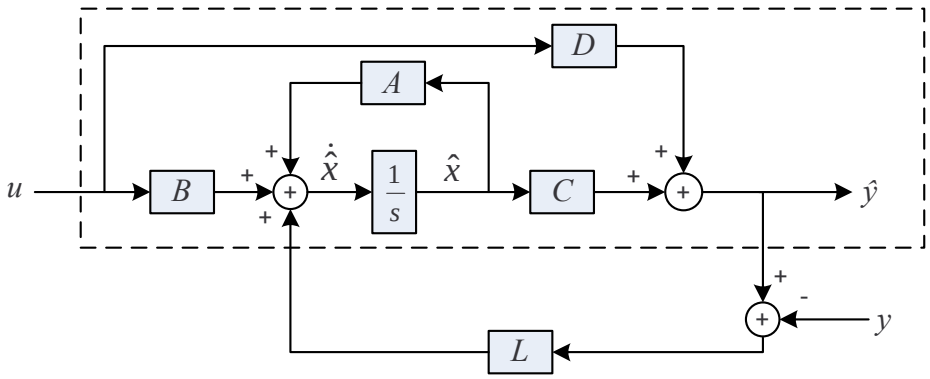

Figure 2.18: Basic Observer for an LTI System.

$$\begin{aligned}
\dot{\hat{x}} &= A\hat{x} + Bu + L(\hat{y} - y) \\
&= A\hat{x} + Bu + L[(C\hat{x} + Du) - (Cx + Du)] \\
&= A\hat{x} + Bu + LC(\hat{x} - x) \qquad (2.79)
\end{aligned}$$

The error signal between the estimated state and actual state is defined as

$$\begin{aligned}
e &= \hat{x} - x \\
\dot{e} &= \dot{\hat{x}} - \dot{x} \qquad (2.80)
\end{aligned}$$

With the help of equations (2.77) and (2.78), equation (2.80) becomes

$$\begin{aligned}
\dot{e} &= (A\dot{\hat{x}} + Bu + LC(\hat{x} - x)) - (Ax + Bu) \\
&= A(\dot{\hat{x}} - x) + LC(\hat{x} - x)) \\
&= (A + LC)e \qquad (2.81)
\end{aligned}$$

By placing the eigenvalues of $A + LC$ suitably in the open left half plane, global asymptotic stability is guaranteed for equation (2.81), which means that $\hat{x}$ will track x. As a general design rule, $\hat{x}$ should track x about 5 to 10 times faster than the

plant dynamics [45], which is the methodology used to design the observer based feedforward controller presented in Chapter 11.

Chapter 3

Problem Formulation

It is well known that the non-minimum phase components of a plant (i.e., right-half plane (RHP) zeros and time delays) can limit the level of achievable performance of a feedback control system. These limitations are quantified by the Bode sensitivity integrals given in [42]. In Section 2.8, we showed that feedback is unable to move zeros and that internal stability does not allow for unstable pole/zero cancelations between systems in the feedback loop. This means that RHP zeros cannot be eliminated using feedback. Also, theses zeros cannot be stably canceled using feedforward methods. Therefore, RHP zeros in the plant (or feedback controller) will always appear in the final closed-loop transfer function. In a similar fashion, time delays cannot be causally eliminated using feedforward or feedback controllers [1]. Therefore, non-minimum phase components of the plant will appear in the final closed-loop transfer function. In the work presented here, a controller architecture is developed that can quantify the achievable level of performance for a non-minimum phase system by defining the class of signals that a non-minimum phase system can perfectly track in the nominal case with no external disturbances.

After a brief overview of the nomenclature used, we will begin the problem for-

[1] In "preview control", the reference input is known ahead of time and the controller is able to act (non-causally) before the reference input occurs. This is not the case considered here.

mulation by considering a known feedforward plus feedback controller architecture that can provide perfect tracking control for minimum-phase systems. Even with this architecture, we can begin to formulate the restrictions that zeros at infinity impose on perfect tracking control. This initial observation provides insight into the two new architectures that we will provide for perfect tracking of non-minimum phase systems, which we will label the *dual feedforward predictive control* (DFFPC) architecture and the *dual feedforward Smith predictor* (DFFSP) architecture. One advantage to these architectures is that the feedforward controllers may be nonlinear and time-varying without affecting the closed-loop stability, provided the signals they inject into the feedback loop are bounded. A detailed discussion of these two methods will be provided in Chapters 4 and 5.

The methods presented here rely on model-based feedforward controllers. Therefore, the accuracy of these models will determine the performance of the overall system. Robustness tools are developed in Chapters 4 and 5 that can predict how well the controllers will perform for a given amount of model uncertainty. If more performance is desired, either a better model with less uncertainty should be developed a priori, or the plant model should be learned via adaptation. The methods for developing these models, evaluating robust performance, and improving the models using adaptation techniques will be discussed in Chapters 4, 5, 6, 7, and 8. Numerical examples that demonstrate the developed tools is provided in Chapter 9.

Each method presented here will contain two feedforward controllers. One controller provides a prediction of the signal in the feedback path and the other provides a feedforward control signal that will drive the plant output along the path provided by the first feedforward controller. Some methods for designing pieces of these controllers are: μ-synthesis, $\mathcal{H}_2$ optimal, and $\mathcal{H}_\infty$ optimal control. These methods will be presented in Chapter 6.

3.1 Nomenclature

The systems considered here are plants that are well modeled by proper linear time-invariant (LTI) systems with possible non-minimum phase components such as time delays and RHP zeros. The transfer function of the LTI plant $G(s)$ is defined to be

$$G(s) = \frac{K_{\rm DC}N(s)}{D(s)} = \frac{K_{\rm DC}N_{nmp}(s)N_{mp}(s)}{D_s(s)D_u(s)}e^{-s\tau_d}, \tag{3.1}$$

where $K_{\rm DC}$ is a gain and $N_{nmp}(s)$, $N_{mp}(s)$, $D_u(s)$, and $D_s(s)$ are the non-minimum phase, minimum-phase, unstable, and stable polynomials that were discussed in Section 2.1. When there are no integrators (i.e., poles at $s = 0$), $K_{\rm DC}$ is the DC gain of the system. Recall that if there are no roots in the half plane associated with the polynomial, then the polynomial is set to one (e.g., if there are no minimum-phase zeros in $G(s)$, then $N_{mp}(s) = 1$). The systems considered here are not allowed to have a zero at the origin (i.e., $G(0) \neq 0$, which means $N_{nmp}(0) \neq 0$). This is not much of restriction since a plant with single zero at the origin (i.e., $G(0) = 0$) would require a ramp input to asymptotically track step changes (i.e., an unbounded control signal is required to maintain a constant output). Without loss of generality, most of these polynomials may be defined to be equal to one when $s = 0$ (e.g., $N_{mp}(0) = 1$). The only exception is when there are poles at $s = 0$. If there is an l-th order pole at $s = 0$, then $D_u(s)$ may be expressed as $D_u(s) = s^l\check{D}_u(s)$, where $\check{D}_u(0) = 1$. This may be done by choosing the scaling $K_{\rm DC}$ appropriately. As an example, consider the transfer function

$$G(s) = \frac{75(-s+2)}{(s+5)(s+10)} = \frac{75(2)(\frac{-s}{2}+1)}{[5(\frac{s}{5}+1)][10(\frac{s}{10}+1)]} = \frac{3(\frac{-s}{2}+1)}{(\frac{s}{5}+1)(\frac{s}{10}+1)}. \tag{3.2}$$

Now,

$$
\begin{aligned}
K_{\mathrm{DC}} &= 3 \\
N_{nmp}(s) &= \frac{-s}{2} + 1 \\
N_{mp}(s) &= 1 \\
D_u(s) &= 1 \\
D_s(s) &= \left(\frac{s}{5} + 1\right)\left(\frac{s}{10} + 1\right).
\end{aligned}
$$

For the controller architectures in the next chapter, the plant model is split into two pieces, namely a piece that has a stable causal inverse (labeled $G_i(s)$) and a piece that does not have a stable causal inverse (labeled $G_{noi}(s)$) by defining $G(s)$ as

$$G(s) = G_{noi}(s)G_i(s). \tag{3.3}$$

In the next sections, we provide a decomposition that achieves this desired split.

3.1.1 Non-invertible/Invertible Decomposition (NID)

For the plant provided in eqn (3.1), the invertible/noninvertible decomposition (NID) is given by

$$
\begin{aligned}
G_{noi}(s) &= N_{nmp}(s)e^{-s\tau_d} && (3.4)\\
G_i(s) &= \frac{K_{\mathrm{DC}}N_{mp}(s)}{D_s(s)D_u(s)}. && (3.5)
\end{aligned}
$$

This split comes directly from the plant definition, and it is clear that that $G(s) = G_{noi}(s)G_i(s)$. Now, the stable causal (but not necessarily proper) inverse of $G_i(s)$ is given by

$$G_i^{-1}(s) = \frac{D_s(s)D_u(s)}{K_{\mathrm{DC}}N_{mp}(s)}. \tag{3.6}$$

For some of the controller synthesis methods used later in this chapter, it is necessary to split $G_i(s)$ into its stable and unstable parts. For this we will define

$$G_i(s) = G_{is}(s)G_{iu}(s) = \left(\frac{K_{\mathrm{DC}}N_{mp}(s)}{D_s(s)}\right)\left(\frac{1}{D_u(s)}\right), \tag{3.7}$$

where $G_{is}(s) = \frac{K_{\mathrm{DC}}N_{mp}(s)}{D_s(s)}$ and $G_{iu}(s) = \frac{1}{D_u(s)}$.

It should be noted that in this decomposition, $G_{noi}(s)$ and $G_i^{-1}(s)$ are not proper transfer functions by themselves, which means that they cannot be physically realized as individual systems. This will not be an issue in the controllers presented here since these expressions will be pieces of an overall controller that will be proper, and therefore, may be implemented using physical hardware.

3.1.2 Discrete-Time Representations

The NID has a discrete-time equivalent, where stable and minimum-phase components are now strictly inside the unit circle (i.e., $|z| < 1$) (instead of the open left-half plant (LHP)), and the unstable and non-minimum phase components are on or outside of the unit circle (i.e., $|z| \geq 1$) (instead of the closed RHP). For a discrete-time system, the decomposition is given by

$$G(z) = \frac{K_{\mathrm{DC}}N_{mp}(z)N_{nmp}(z)}{D_s(z)D_u(z)}z^{-N_d}. \tag{3.8}$$

Here, the N_d-sample delay translates to a delay of $\tau_d = N_dT_s$ seconds. In the applications considered here, $G(z)$ will be the zero-order hold (ZOH) equivalent of a continuous-time plant ($G(s)$). It should be noted that this formulation does not require the plant delay of the underlying continuous-time plant to be an integer multiple of sample period, namely T_s, of the discrete-time system. For continuous-time systems that have a non-integer delay, we can use a modified Z-transform to get a different $G(z)$ that contains the effect of the fractional part of the delay (i.e., the fractional part of τ_d/T_s) [41]. Given the appropriate $G(z)$, the factorization in eqn (3.8)

may be defined.

For illustrative purposes, we will continue with the previous example. For the continuous-time plant in eqn (3.2), the ZOH equivalent of the continuous-time plant with sampling period $T_s = 0.01$ is given by

$$G_{ZOH}(z) = \frac{-0.68874(z - 1.0202)}{(z - 0.9512)(z - 0.9048)}. \tag{3.9}$$

At DC (i.e., $\omega = 0$) the value of $z = e^{j\omega T_s}$ is $z = 1$. In order to guarantee the same polynomial constraint from before (i.e., $G(z) = 1$ when $\omega = 0$, or $G(1) = 1$), the following method is used to redefine the polynomials and appropriate DC gain.

$$G_{ZOH}(z) = \frac{-0.68874(1 - 1.0202)\frac{z-1.0202}{1-1.0202}}{(1 - 0.9512)\frac{z-0.9512}{1-0.9512}(1 - 0.9048)\frac{z-0.9048}{1-0.9048}} = \frac{3\frac{z-1.0202}{1-1.0202}}{\left(\frac{z-0.9512}{1-0.9512}\right)\left(\frac{z-0.9048}{1-0.9048}\right)}. \tag{3.10}$$

Here, the DC gain is the same as before[2] (i.e., $K_{\mathrm{DC}} = 3$) and the other polynomials are given by

$$\begin{aligned} N_{nmp}(z) &= \frac{z - 1.0202}{1 - 1.0202} \approx -49.4667z + 50.4667 \\ N_{mp}(z) &= 1 \\ D_u(z) &= 1 \\ D_s(z) &= \left(\frac{z - 0.9512}{1 - 0.9512}\right)\left(\frac{z - 0.9048}{1 - 0.9048}\right) \approx (20.504z - 19.504)(10.508z - 9.508). \end{aligned} \tag{3.11}$$

Here, the approximately equal signs are to account for rounding errors in the values displayed.

[2]Due to the finite precision of polynomial representations given in eqn (3.10), K_{DC} will appear to be $\frac{-0.68874(1-1.0202)}{(1-0.9512)(1-0.9048)} = 2.995$. However, with more precision, the value of K_{DC} is actually 3.

3.1.3 Perfect Tracking

Throughout the work presented here, the term "perfect tracking" is used in a very specific context. In particular, we use it to describe systems in the nominal case (i.e., the plant model is perfect) when there are no external disturbances. In this case, perfect tracking means that the feedback error is zero for all time (i.e., $e(t) = 0$), which must hold for all reference inputs ($r(t)$). This is equivalent to having no sensitivity to inputs (i.e., $S(s) = \frac{E(s)}{R(s)} = 0$), which may be achieved by defining a (feedforward) filtered reference input (r_{ff}) that the plant can actually follow in the nominal case with no external disturbances. In this case, the error signal is really $e(t) = y(t) - r_{ff}(t)$ and not the standard feedback error $e(t) = y(t) - r(t)$. This is necessary to achieve perfect tracking for all strictly proper minimum-phase and all proper non-minimum phase systems. The only class of systems that may be perfectly tracked under the condition $e(t) = y(t) - r(t) = 0$ are stable minimum-phase biproper systems (i.e., $G_i^{-1}(s)$ is a biproper feedforward controller and $G_i^{-1}(s)G(s) = I$ is a trivial form of perfect tracking control). This idea of perfect tracking for minimum-phase systems and its extensions to non-minimum phase systems is developed next.

3.2 A Known Two-Stage Feedforward Controller Architecture

For the control architecture discussed here, we will restrict the discussion to minimum-phase plants with no time delays (i.e., $G(s) = G_i(s) = \frac{K_{\text{DC}} N_{mp}(s)}{D_s(s) D_u(s)}$). Perfect tracking for minimum-phase plants may be achieved using a known architecture, which will be presented next. This architecture forms the base for the architectures presented later. Collectively, we will refer to these methodologies that provide perfect tracking as *two-stage feedforward control* (TSFFC).

3.2.1 Minimum-phase Biproper Plants

To begin, we consider minimum-phase biproper plants. In the biproper case, $G_i^{-1}(s) = \frac{D_s(s)D_u(s)}{K_{\text{DC}}N_{mp}(s)}$ is a stable biproper transfer function that is realizable in hardware. In this case, any reference signal may be tracked with zero tracking error in the nominal case with no external disturbances (i.e., perfect tracking of $r(t)$ is achieved). A TSFFC architecture that achieves this is shown in Figure 3.1.

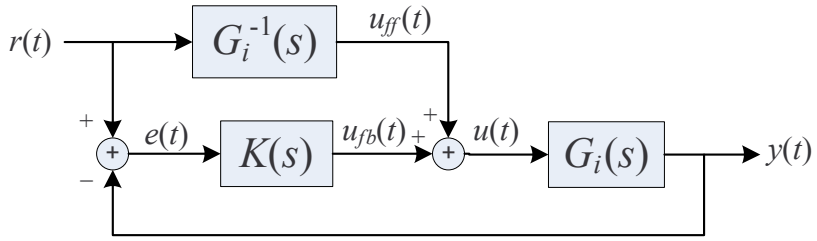

Figure 3.1: Two-stage Feedforward Control for a Minimum-phase Biproper Plant

The controller architecture for a biproper minimum-phase plant shown in Figure 3.1 has sensitivity function

$$S(s) = \frac{G_i^{-1}(s)G_i}{1 + G_i(s)K(s)} - \frac{1}{1 + G_i(s)K(s)} = 0, \tag{3.12}$$

and closed-loop transfer function

$$M(s) = \frac{G_i^{-1}(s)G_i}{1 + G_i(s)K(s)} + \frac{G_i(s)K(s)}{1 + G_i(s)K(s)} = 1. \tag{3.13}$$

Notice that the unstable poles of $G_i(s)$ become non-minimum phase zeros in $G_i^{-1}(s)$. It should be noted that this implementation does not violate internal stability since there are no direct unstable pole/zero cancelations between the systems $G_i^{-1}(s)$ (in the feedforward controller) and plant $G_i(s)$, but rather an interaction between a stable signal ($u_{ff}(t)$) and a stable feedback loop formed by $G_i(s)$ and $K(s)$. This is a subtle, but important, point. Internal stability of the system in Figure 3.1 requires the feedback loop (defined by $K(s)$ and $G_i(s)$) to be stable. If this feedback loop is

stable (i.e., the poles of $1+G_i(s)K(s)$ are in the open LHP and there are no unstable pole/zero cancelations in the formation of $G_i(s)K(s)$), then the overall system will be stable provided all of the external inputs are bounded. In this case, the external signals that must be bounded are $r(t)$ and $u_{ff}(t)$. Since $G_i^{-1}(s)$ will be stable, $u_{ff}(t)$ will be bounded if $r(t)$ is bounded. If $G_i(s)$ contains an unstable pole, then $u_{ff}(t)$ will contain an inverse response to step changes that is a result of the non-minimum phase zero in $G_i^{-1}(s)$. In a sense, this is an unstable pole/zero cancelation between the *signal* $u_{ff}(t)$ and the *system* $G_i(s)$, which is not the same as a cancelation between two *systems*. To explore this further, let $\epsilon(t)$ be a non-zero time function. If the feedforward control signal is perturbed to $u_{ff}(t)+\epsilon(t)$, the feedback loop (defined by $K(s)$ and $G_i(s)$) will still be stable, since the feedback loop is designed to be internally stable. In this case, the feedback controller will use $u_{fb}(t)$ to stably correct for the error caused by $\epsilon(t)$.

For the illustrative examples considered in the work presented here, performance will be demonstrated by tracking reference input step functions. The fact that $M(s) = 1$ for minimum-phase biproper plants means that any reference input ($r(t)$) may be perfectly tracked (i.e., $y(t) = r(t)$). In particular, this means that reference step inputs may be perfectly tracked. To see this, consider the biproper unstable minimum-phase plant

$$G_1(s) = \frac{s+3}{-s+1} = \frac{3(\frac{s}{3}+1)}{-s+1} \tag{3.14}$$

For this plant, the plant components from eqn (3.1) are given by $K_{\mathrm{DC}} = 3$, $N_{nmp}(s) = 1$, $N_{mp}(s) = (\frac{s}{3}+1)$, $D_s(s) = 1$, and $D_u(s) = -s+1$, and the resulting NID is $G_i(s) = G_1(s)$ and $G_{noi}(s) = 1$.

A feedback controller was designed for this plant using a weighted μ-synthesis algorithm with additive uncertainty. The result was

$$K(s) = \frac{-46.3348(s+200)(s+30.02)(s+10)(s+0.3865)}{s(s+189.3)(s+2.399)(s^2+136.2s+5750)} \tag{3.15}$$

A simulation of the feedback only system with the above controller is shown in Figure 3.2.

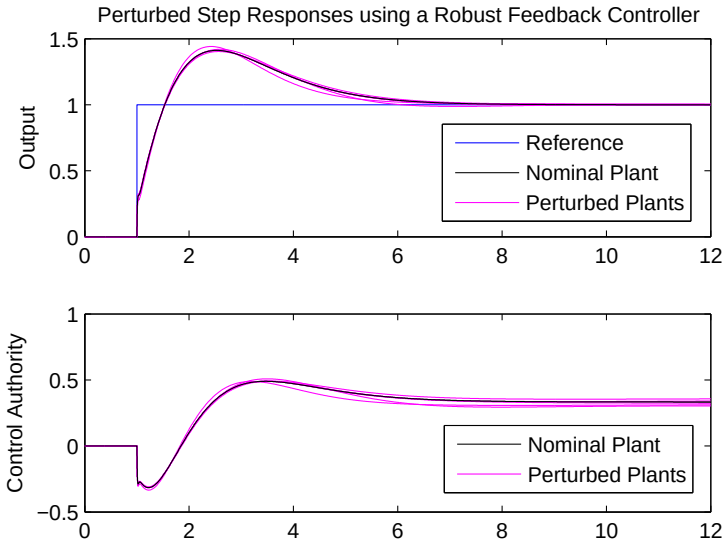

Figure 3.2: Nominal and Perturbed Step Responses with a Feedback Controller

In this example, both the nominal plant $G_1(s)$ and randomly perturbed plants $G_1(s) + \Delta(s)$ with $\|\Delta(s)\|_\infty = 0.3$ are plotted. The results show that similar performance is achieved in the nominal and perturbed cases.

Using the TSFFC block diagram in Figure 3.1, steps may be tracked perfectly. A simulation of the TSFFC is shown in Figure 3.3.

In the top graph, the input step reference (solid red line) and plant output (dotted black line) are identical, which demonstrates perfect tracking. This is echoed in the bottom graph where the feedback error is zero for all time (i.e., $e(t) = 0$). Also in the bottom graph, the control signal that provides perfect tracking is shown. Since the plant and controller are both biproper, there is an instantaneous jump in both the control signal and the plant output, which allows for the perfect tracking of a step reference input.

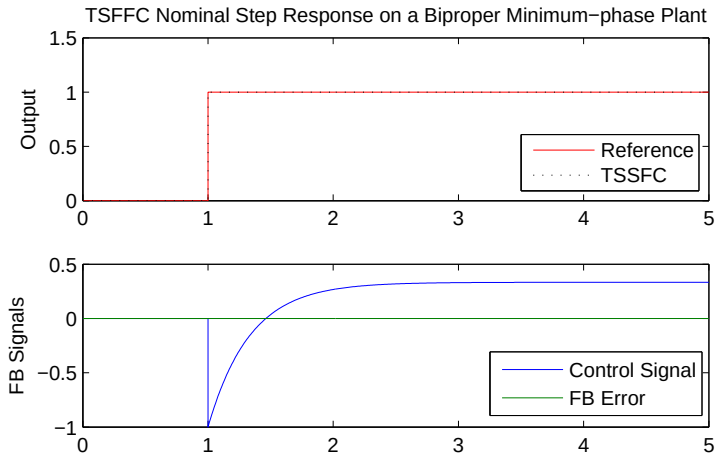

Figure 3.3: Perfect Tracking of a Biproper System using TSFFC

Previously, it was claimed that the TSFFC architecture in Figure 3.1 does not violate internal stability. This is demonstrated using two perturbation techniques. In the first method, we leave $G_i^{-1}(s) = \frac{-s+1}{s+3}$, but we define $G_i(s) = \frac{s+3}{-s+1.2}$. Now there is a mismatch between the RHP zero in $G_i^{-1}(s)$ and the unstable pole in $G_i(s)$. Provided the feedback controller is able to stabilize this new plant, the TSFFC will remain stable. A simulation of this is shown in Figure 3.4.

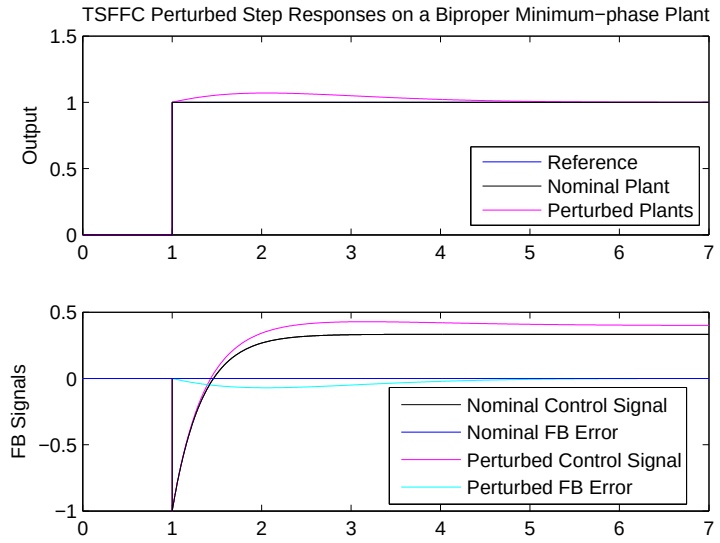

Figure 3.4: TSFFC Perturbed Unstable Pole Step Response

Here, the TSFFC architecture is not providing perfect tracking, but the response is still better than the feedback only structure. Also the overall system is stable, since closed-loop stability of the system is determined solely by $K(s)$ and is unaffected by the stable dynamics in the signal $u_{ff}(t)$.

As a second example, the same perturbations used for the feedback only simulations in Figure 3.2 are used to perturb the plant. As in the previous example, $G_i^{-1}(s) = \frac{-s+1}{s+3}$, but we define $G_i(s) = G_i(s) + \Delta(s)$ with $\|\Delta(s)\|_\infty = 0.3$ using the same $\Delta(s)$ perturbations from feedback simulation example. The results are shown in Figure 3.5.

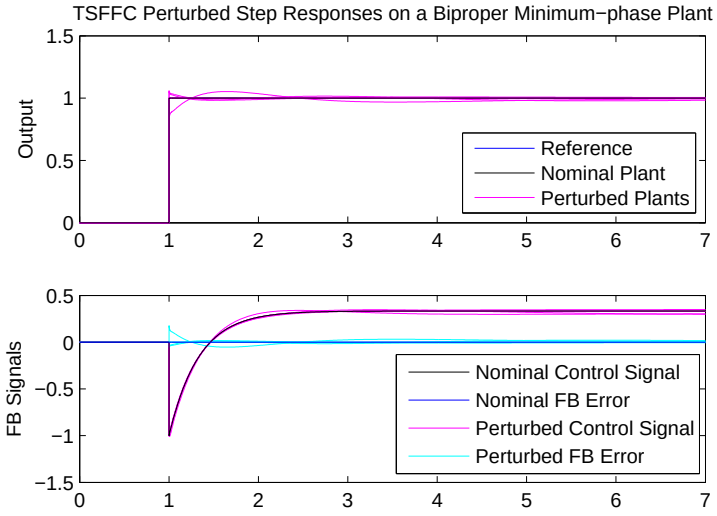

Figure 3.5: TSFFC Perturbed Additively Perturbed Step Responses

For this case, a similar level of robustness is achieved for both the perturbed TSFFC and perturbed feedback only structures. The addition of the feedforward controller acts to "center" the reference trajectory that the feedback controller must correct against. Unlike the robust feedback only controller, which is trying to both track "large" reference signals and reject "small" disturbance signals, the feedback controller in the TSFFC architecture is only trying to reject "small" disturbances, which is well suited to the robust and optimal controller synthesis algorithms used.

3.2.2 Minimum-phase Strictly Proper Plants

For strictly proper minimum-phase plants, $G_i(s) = G(s)$ and $G_i^{-1}(s)$ is stable. Unlike the biproper case, $G_i^{-1}(s)$ is not a proper transfer function, which means that it not physically realizable. To account for this, a filter $P_{\text{des}}(s)$ may be created such that the feedforward controller $P_{\text{des}}(s)G_i^{-1}(s)$ is strictly proper. In this case, the overall control system will be able to perfectly track reference signals that have been filtered by $P_{\text{des}}(s)$. A controller architecture that achieves perfect tracking for proper minimum-phase systems is shown in Figure 3.6.

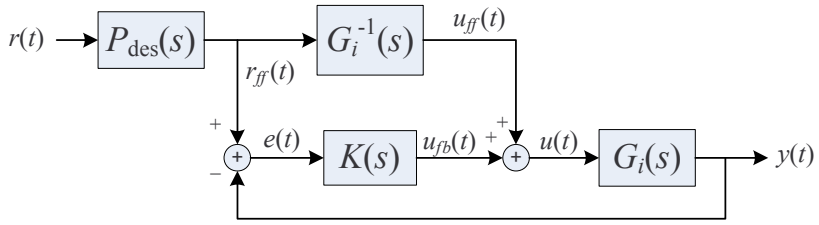

Figure 3.6: TSFFC for a Stable Minimum-phase Plant

In the nominal case with no external disturbances, the feedforward control signal ($u_{ff}(t)$) will drive the plant output trajectory ($y(t)$) along the predicted filtered reference ($r_{ff}(t)$). When the plant is biproper (as in the previous example), $P_{\text{des}}(s) = 1$ is a valid option and the architecture in Figure 3.6 reduces down to the architecture in Figure 3.1. When the plant $G_i(s)$ is strictly proper, $G_i^{-1}(s)$ is not proper; however, the cascade of systems formed by $P_{\text{des}}(s)G_i^{-1}(s)$ can be designed to be proper. To see this, let $G_i(s)$ have m zeros and n poles. Then, the relative degree of $G_i(s)$ is $n - m > 0$ and $G_i^{-1}(s)$ has the negative relative degree of $m - n < 0$, which is not proper. However, if the relative degree of $P_{\text{des}}(s)$ is at least $n - m$, then $P_{\text{des}}(s)G_i^{-1}(s)$ will have a positive relative degree, and therefore, be proper. For the TSFFC architecture in Figure 3.6, the sensitivity function is given by

$$S(s) = \frac{P_{\text{des}}(s)G_i^{-1}(s)G_i}{1 + G_i(s)K(s)} - \frac{P_{\text{des}}(s)}{1 + G_i(s)K(s)} = 0, \tag{3.16}$$

and closed-loop transfer function

$$M(s) = \frac{P_{\text{des}}(s)G_i^{-1}(s)G_i}{1 + G_i(s)K(s)} + \frac{P_{\text{des}}(s)G_i(s)K(s)}{1 + G_i(s)K(s)} = P_{\text{des}}(s). \tag{3.17}$$

Unlike the biproper case, strictly proper systems cannot track a step perfectly. Instead they can perfectly track a step that has been filtered by $P_{\text{des}}(s)$. This additional design parameter is required to make feedforward controller $P_{\text{des}}(s)G_i^{-1}(s)$ proper. To see the implications of this, consider the strictly proper plant

$$G_2(s) = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} = \frac{0.5}{s^2 + 0.1s + 0.25}, \tag{3.18}$$

where, $K = 2$, $\xi = 0.1$, and $\omega_n = 0.5$ rad/sec. For this plant, the plant components from eqn (3.1) are given by $K_{\text{DC}} = 2$, $N_{nmp}(s) = 1$, $N_{mp}(s) = 1$, $D_s(s) = (\frac{s^2}{\omega_n^2} + \frac{2\xi s}{\omega_n} + 1) = (4s^2 + 0.4s + 1)$, and $D_u(s) = 1$, and the resulting NID is $G_i(s) = G_2(s)$ and $G_{noi}(s) = 1$. For the example considered here, we will choose

$$P_{\text{des}}(s) = \frac{1}{(\tau s + 1)^2}, \tag{3.19}$$

where the design parameter τ determines the bandwidth $(1/\tau)$ of the filter $P_{\text{des}}(s)$. Specifically, a smaller τ will result in a larger bandwidth that will result in a faster step response. This is illustrated in Figure 3.7, where step responses with $\tau = 1, 0.5,$ and 0.1 are shown.

The TSFFC controller from Figure 3.6 is able to perfectly track the filtered references in Figure 3.7 (i.e. $y(t) = r_{ff}(t)$). While it is not possible to perfectly track an ideal step (i.e., the unfiltered step $r(t)$ in Figure 3.7) in the proper case, the filtered reference can be made arbitrarily close to an ideal step by making τ arbitrarily small. However, as the bandwidth increases, so does the magnitude of the control signal. This is illustrated in the bottom graph of Figure 3.7, where the peak magnitude of the control signal is rapidly increasing as τ becomes smaller. This will lead to robustness issues, which will put an additional design constraint that was not present in the

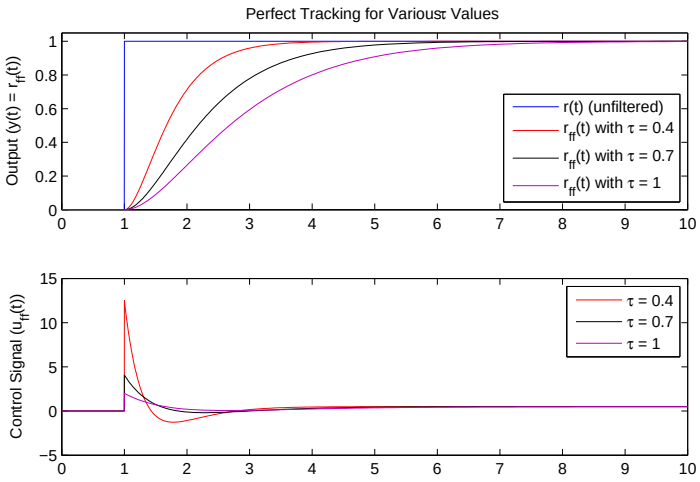

Figure 3.7: Nominal Step Responses for a Strictly Proper Minimum-phase Plant

biproper case. In the chapters that follow, robust analysis tools are developed that may be used to guide the $P_{\text{des}}(s)$ design process. This example will be studied more thoroughly in Chapter 9.

3.3 Limitations with Current Methods

In the biproper minimum-phase case, there are no RHP zeros in $G(s)$ and ideal steps may be perfectly tracked. When $G(s)$ is strictly proper, but still minimum-phase, it has a right-half plane zero at infinity (i.e., the region of convergence of $G(s)$ includes the RHP as it extends toward ∞ and $\lim_{s\to\infty} G(s) = 0$). The result is that ideal steps can no longer be perfectly tracked; however, filtered reference signals may be perfectly tracked (with an appropriate $P_{\text{des}}(s)$ filter). While not currently presented this way in the literature, the requirement of a reference input filter for perfect tracking demonstrates that a RHP zero at infinity will limit the achievable performance.

When there is a finite RHP zero or time delay in the plant, the methods, as they have been presented here, will not be able to provide perfect tracking. For perfect tracking of a non-minimum phase plant, an augmentation to the architecture

is required. Two architectures are presented in the next two chapters that address this limitation.

For the architectures in this chapter and the ones used in the next two chapters, robust performance will directly related to the accuracy of models used in the feedforward controllers and the particular choice of $P_{\text{des}}(s)$. Currently, tools do not exist for designing and analyzing the effectiveness of the controller architecture when there are inaccuracies in the model. For this, robustness tools are developed that may aid in the design process. Also, real-time adaptation techniques are developed that will be useful for improving the plant models in the feedforward controllers (and hence the overall performance of the control system).

In [39], a numerical example was used to shown how a specific version of the architecture presented in this chapter could provide perfect tracking to a system with a finite number of finite RHP zeros. However, the details for applying these techniques to general non-minimum phase plants were not given. Also, the robustness analysis tools and adaptation methods were never explored. The work presented here was developed independent of the work in [39] and is presented for a wider class of systems (i.e., for all LTI systems that may be expressed using eqn (3.1)) and provides tools for analyzing and improving robust performance.

Chapter 4

Dual Feedforward Predictive Control

The first method for implementing the general architecture is *dual feedforward predictive control* (DFFPC), which handles the prediction, design constraint, and ballistic response entirely in the two feedforward paths. The presentation of this architecture is for single-input single-output (SISO) systems. Potential extensions to multiple-input multiple-output (MIMO) systems are discussed as future work in Chapter 12. The perfect tracking property presented here is valid for a larger class of plants than the results currently provided in the literature (c.f., [39]). Specifically, this architecture may provide perfect tracking for:

- Stable and unstable systems.
- Biproper and strictly proper systems.
- Minimum and non-minimum phase systems (including time delays).

The methods discussed here are able to provide perfect tracking in both a continuous-time and a discrete-time framework. This is an advantage over existing perfect tracking methods (c.f., [34; 35; 36]) that use multi-rate digital controllers to achieve perfect tracking at the sampling points of the slowest sampling rate in the system. For the digital controllers presented here, the use of a faster rate controller is not required for the perfect tracking property.

The initial presentation of this method is done in the continuous-time domain. At the end of this chapter, we address the discrete-time implementation of the method.

4.1 Dual Feedforward Predictive Control

Recall that the perfect tracking property was defined as a control system that had zero feedback tracking error when the plant model was perfect and there were no disturbances. In order to obtain this perfect tracking property, there are two signals that need to be generated, namely a desired reference trajectory that the nominal plant will follow ($r_{ff}(t)$), and the control signal ($u_{ff}(t)$) that will drive the nominal plant along the desired reference trajectory.

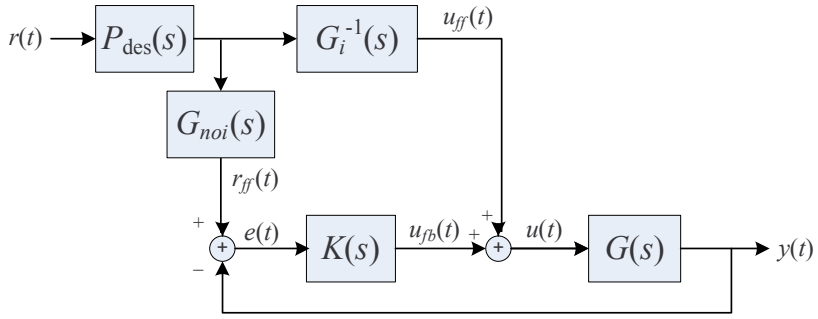

Figure 4.1: Nominal LTI DFFPC Architecture

The block diagram of the DFFPC architecture is shown in Figure 4.1. The blocks $G_{noi}(s)$ and $G_i^{-1}(s)$ are the non-invertible and inverse of the stable minimum-phase dynamics of the plant $G(s)$ as they were defined in eqns (3.4) and (3.6), and $P_{\text{des}}(s)$ is a design parameter that determines the desired reference trajectory ($r_{ff}(t)$) and associated feedforward control signal ($u_{ff}(t)$).

The two feedforward controllers that will be implemented are $FF1(s) = P_{\text{des}}(s)G_{noi}(s)$, which generates $r_{ff}(t)$ from $r(t)$, and $FF2(s) = P_{\text{des}}(s)G_i^{-1}(s)$, which generates $u_{ff}(t)$ from $r(t)$. There are three design constraints that must be satisfied by $P_{\text{des}}(s)$, namely

1. The steady state gain from $r(t)$ to $r_{ff}(t)$ must be unity (i.e., $P_{\text{des}}(0)G_{noi}(0) = 1$)

2. The feedforward controller $FF1(s) = P_{\text{des}}(s)G_{noi}(s)$ must result in a stable proper transfer function

3. The feedforward controller $FF2(s) = P_{\text{des}}(s)G_i^{-1}(s)$ must result in a stable proper transfer function

The first condition is required so that the steady-state reference equals the actual reference (i.e., $r_{ff}(t) = r(t)$ as $t \to \infty$). The second and third conditions are required to make the two feedforward controllers realizable in hardware. From the definition that $G_{noi}(0) = 1$ in eqn (3.4) and the fact that $G(s)$ is proper, these conditions will be met if:

- $P_{\text{des}}(0) = 1$ (satisfies first condition above).
- $P_{\text{des}}(s)$ is stable, and the relative degree of $P_{\text{des}}(s)$ is greater than or equal to the relative degree of $G_i(s)$ (satisfies second and third condition above).

The overall sensitivity transfer function (i.e., from $r(t)$ to $e(t)$) for DFFPC is

$$\begin{aligned} S_{\text{DFFPC}}(s) &= P_{\text{des}}(s)G_{noi}(s)S(s) - P_{\text{des}}(s)G_i^{-1}(s)G(s)S(s) \\ &= P_{\text{des}}(s)G_{noi}(s)S(s) - P_{\text{des}}(s)G_{noi}(s)S(s) \\ &= 0. \end{aligned} \tag{4.1}$$

This means that the *nominal performance* is perfect (i.e., $S_{\text{DFFPC}}(s) = 0$) and the condition $\|W_1(s)S_{\text{DFFPC}}(s)\|_\infty < 1$ is trivially satisfied for any finite $W_1(s)$. The feedback error signal is zero and the feedforward controller is providing perfect tracking of the filtered reference $r_{ff}(t)$, which is a reference signal that the plant can exactly follow. This is possible because $r_{ff}(t)$ contains the effects of the non-minimum phase part of the plant that neither the feedforward nor feedback components can stably and causally invert. An example of this is when there is a time delay in the plant

(e.g., $G_{noi}(s) = e^{-s\tau_d}$ for some time delay of τ_d seconds). In this case, the $FF2(s)$ control signal will be applied to the plant, but the plant output $y(t)$ will not start changing for τ_d seconds due to the time delay. In this case, it is appropriate to delay the filtered reference signal until the output resulting from $FF2(s)$ can be sensed at the output.

The intention of this method is not to remove the non-minimum phase components from the plant, but rather utilize them to define a class of signals that may be perfectly tracked. To see this, consider the closed-loop transfer function (from $r(t)$ to $y(t)$):

$$\begin{aligned} M_{\text{DFFPC}}(s) &= \frac{P_{\text{des}}(s)G_{noi}(s)G(s)K(s)}{1+G(s)K(s)} + \frac{P_{\text{des}}(s)G_i^{-1}(s)G(s)}{1+G(s)K(s)} \\ &= P_{\text{des}}(s)G_{noi}(s)\frac{1+G(s)K(s)}{1+G(s)K(s)} \\ &= P_{\text{des}}(s)G_{noi}(s). \end{aligned} \tag{4.2}$$

Here, the non-minimum phase components appear in the closed-loop transfer function as $G_{noi}(s)$ and $P_{\text{des}}(s)G_{noi}(s)$ defines the class of signals that may be perfectly tracked. The nominal design objective is to design $P_{\text{des}}(s)$ to get the desired closed-loop characteristics. Even though the closed-loop poles (i.e., the roots of $1 + G(s)K(s)$) do not appear in the nominal closed-loop transfer function $M_{\text{DFFPC}}(s)$, they will affect performance in terms of disturbance rejection and correcting for modeling errors. Also, internal stability is always required. Therefore, the controller $K(s)$ should be designed to guarantee internal stability and to provide the desired level of robust performance in terms of disturbance rejection. Since the feedforward signals are essentially bounded external signals to the feedback loop, they do not affect closed-loop stability. Therefore, it makes sense that nominal stability is unaffected by the addition of the feedforward components. By the same argument, robust stability is also unaffected by the addition of the feedforward components, though they do affect performance. This is addressed again later in the chapter.

In this architecture, the nominal tracking requirements are provided by the feedforward paths, and the feedback controller focuses on correcting for model inaccuracies and disturbance rejection. This is a different design perspective than the standard robust feedback controller that is designed to provide both good tracking performance and good disturbance rejection in the presence of model uncertainty. Now, the feedback controller is only used to provide disturbance rejection and to correct for modeling errors, which is the preferred setting for robust and optimal controller synthesis.

For the setting considered here, the feedforward and feedback designs are decoupled, since the feedforward controller is designed for nominal tracking performance, and the feedback controller is designed for disturbance rejection, correcting for modeling errors, and providing robust stability. Later in the chapter, we will develop a robust performance criterion based on the weighted perturbed closed-loop sensitivity function. These criteria expose the inherent trade-offs between the two designs. For example, if the feedback controller tries for too much performance, it will result in smaller stability margins. This will limit the amount of performance improvement that should be sought from from the addition of the feedforward controllers and still maintain the robust performance criteria. However, in this case, the feedback controller will provide better disturbance rejection. Conversely, if the feedback controller is designed with larger stability margins, the feedforward controllers may be more aggressive and still maintain the robust performance criteria. However, in this case, the feedback controller will take longer to correct for modeling errors and disturbances. In the first case, it is assumed that the accuracy of the plant model (and hence the quality of the two feedforward controllers) is lacking. Therefore, the feedback controller is more active correcting for model inaccuracies. In the latter case, the plant model is assumed to be more accurate, which means that there will be more performance gain from the addition of the feedforward controllers. In general,

the amount of performance improvement gained from the feedforward controllers is dependent on the quality of the plant model used. Specifically, better plant models translate to a larger robust performance gain from the addition of the feedforward controllers on a physical system. This means that for better performance gain, good plant models must either be known a priori or learned via adaption (which is the focus of Chapter 7). For this architecture, the adaptation of the feedforward controllers does not affect closed-loop stability, which is an advantage to using adaptation with this method.

4.2 Controller Design

While the design constraints given above are required for the feedforward controllers to be proper, they also provide limitations on achievable performance. In particular, $P_{\text{des}}(s)G_{noi}(s)$ determines the class of signals that may be perfectly tracked, and $P_{\text{des}}(s)G_i^{-1}(s)$ is the feedforward controller that will provide the associated control signal to achieve perfect tracking. The class of signals that may be perfectly tracked includes the effects of the non-minimum components that neither feedforward nor feedback controllers can eliminate. This phenomenon was discussed in Chapter 2.8.

There are two pieces that need to be designed, namely the feedforward and feedback controllers. The feedforward controller is designed for "large" signal reference tracking and the feedback controller is designed "small signal" disturbance rejection. Since the feedforward design is common to the known architecture presented in the previous chapter, the DFFPC architecture presented in this chapter, and the controller architecture presented in the next chapter, a full discussion of the feedforward design is postponed until Section 6. The design of the feedback controller is cast as a traditional feedback design (e.g., provide internal stability and desired performance). There are many standard methods for designing the feedback controller (c.f., [44; 54; 43; 45]). A particular example of a feedback design that is well suited for this problem

is presented next.

4.2.1 Feedback Controller Design

The role of the feedback controller is to provide internal stability, disturbance rejection performance, and to correct for modeling errors in the two feedforward paths. Since nominal reference tracking is handled by the two feedforward controllers, the feedback control design is cast as a disturbance rejection problem instead of a reference tracking problem. Also, since the feedforward parts do not contribute to the disturbance rejection, they are not included in the design. For the method presented here, the focus here will be on designing robust controllers using μ-synthesis as it is described in [58].

For the disturbance rejection problem considered here, we will use the design interconnect shown in Figure 4.2.

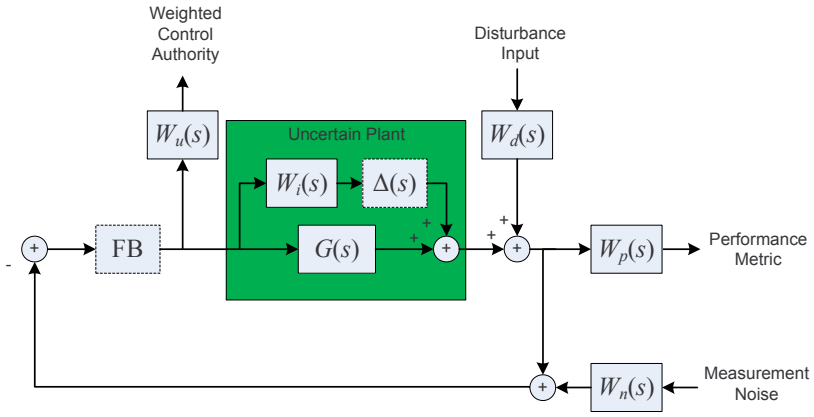

Figure 4.2: Disturbance Rejection Robust Controller Synthesis Interconnect

Here, the objective of μ-synthesis is to minimize the $\mathcal{H}_\infty$ norm from (weighted) exogenous inputs to (weighted) exogenous outputs in the presence of (weighted) uncertainty. The design weights may be used to shape the closed-loop frequency response. Example weights are provided in Figure 4.2 that could be used to reject low frequency disturbances.

Here, the two exogenous inputs are (weighted) external disturbances to the plant

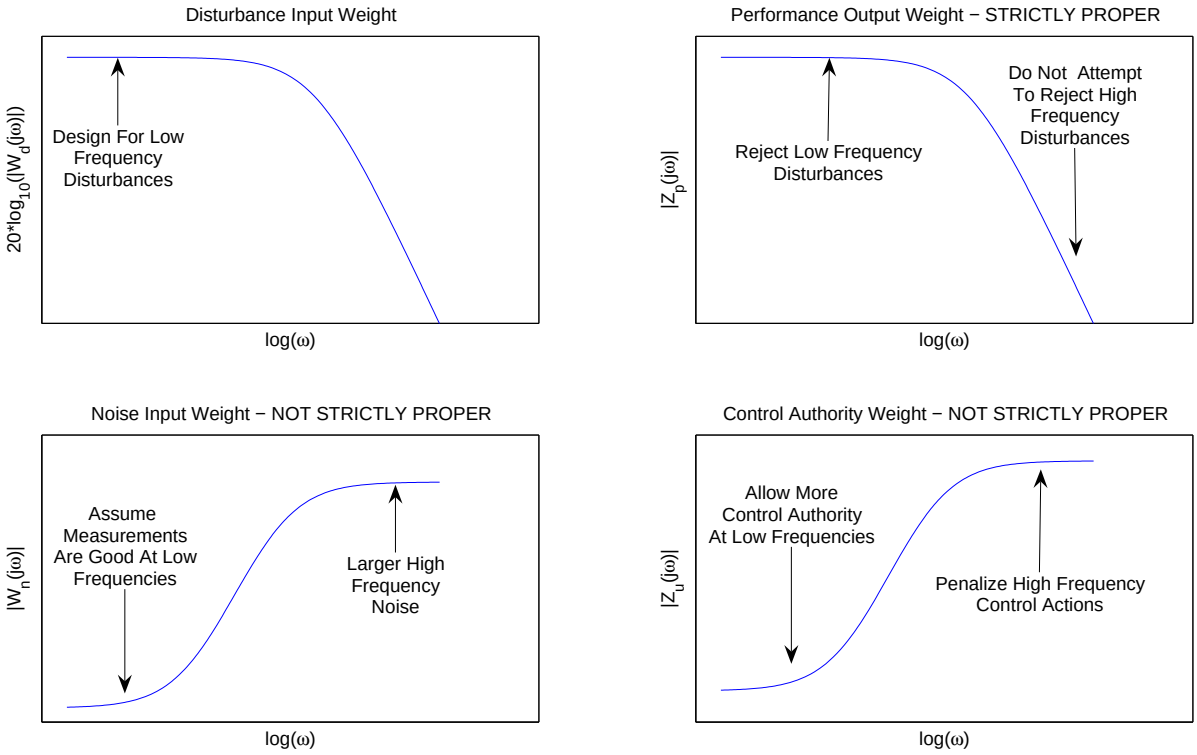

Figure 4.3: Example Robust Controller Synthesis Weights

output and (weighted) sensor noise on the output measurement. In practice, sensors tend to provide good low frequency measurements, but are unable to provide accurate high frequency measurements. Therefore, the weight is selected as a high pass filter that injects less noise at lower frequencies and more noise at higher frequencies. Due to the placement of this weighted input in Figure 4.2, the sensor noise only affects the measured signal that is used by the feedback controller. In contrast, the weighted disturbance input also affects the weighted performance output. In many applications, the sensor noise input may be considered optional. However, when the weighted sensor noise input (W_n) is not used, the disturbance weight (W_d) must be not strictly proper. For this example, we would like to reject low frequency disturbances, which means that the disturbance input weight is large at low frequencies, where we assume disturbances will reside, and small at larger frequencies, since the assumed disturbances will have little high frequency content.

For this feedback design, the two exogenous outputs are the (weighted) performance output and (weighted) control authority. In the frequency ranges where these weights are large, the $\mathcal{H}_\infty$ optimization penalizes these signals (i.e., tries to make them small). Similarly, in the frequency ranges where these weights are small, the $\mathcal{H}_\infty$ optimization allows signals to have larger amplitudes. An example set of weights and their associated trade-offs are shown in Figure 4.3. An example application where we used these style of weights is provided in [59].

4.3 Robustness Analysis

This section explores the robust performance metrics on plants with additive and multiplicative uncertainty.

4.3.1 Additive Uncertainty

For robustness analysis against model uncertainty, an uncertain plant model with additive uncertainty is considered. In this case, the nominal plant model G is replaced with the perturbed plant model $\tilde{G} = G + W_2\Delta$. The block diagram for this model is shown in Figure 4.4.

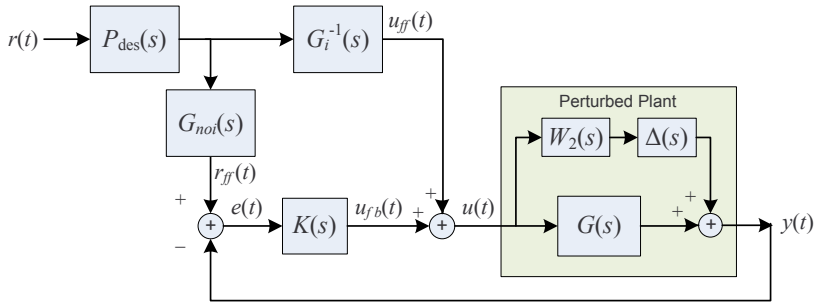

Figure 4.4: Uncertain LTI DFFPC Architecture with Additive Uncertainty

The perturbed sensitivity function becomes

$$\begin{aligned}
\tilde{S}_{\text{DFFPC}} &= \frac{P_{\text{des}}G_{noi}}{1+(G+\Delta W_2)K} - \frac{P_{\text{des}}G_i^{-1}(G+\Delta W_2)}{1+(G+\Delta W_2)K} \\
&= \frac{P_{\text{des}}G_{noi} - P_{\text{des}}G_{noi} - P_{\text{des}}G_i^{-1}\Delta W_2}{1+(G+\Delta W_2)K} \\
&= -\frac{P_{\text{des}}G_i^{-1}\Delta W_2/(1+GK)}{(1+GK+\Delta W_2 K)/(1+GK)} \\
&= -\frac{P_{\text{des}}G_i^{-1}\Delta W_2 S}{1+\Delta W_2 KS} \qquad (4.3)
\end{aligned}$$

The nominal and perturbed feedback characteristic equations (i.e., the denominator polynomials of the feedback loops) are $(1 + L)$ and $(1 + L)(1 + \Delta W_2 KS)$. These are identical to the characteristic equations for the feedback only architecture seen in Chapter 2.6.1. Using the same reasoning provided there (which assumes that $\|\Delta\|_\infty \leq 1$), robust stability requires $\|W_2 KS\|_\infty < 1$.

For robust performance, we require robust stability and a norm bound on the weighted perturbed sensitivity function. This is given by

$$\|W_2 KS\|_\infty < 1 \text{ (RS)} \quad \text{and} \quad \left\| W_1 \frac{P_{\text{des}}G_i^{-1}\Delta W_2 S}{1+\Delta W_2 KS} \right\|_\infty < 1, \quad \forall \|\Delta\|_\infty \leq 1. \qquad (4.4)$$

These criteria may be shown to be an exact test for robust performance. To show this, the above equation, with a slight abuse of notation, may equivalently be expressed as

$$s \mapsto |W_1(s)P_{\text{des}}(s)G_i^{-1}(s)W_2(s)S(s)| + |W_2(s)K(s)S(s)|, \qquad (4.5)$$

which is denoted $|W_1 P_{\text{des}}G_i^{-1}W_2 S| + |W_2 KS|$. Now, a necessary and sufficient condition for robust performance is given by

$$\| |W_1 P_{\text{des}}G_i^{-1}W_2 S| + |W_2 KS| \|_\infty < 1, \qquad (4.6)$$

or equivalently,

$$|W_1(j\omega)P_{\text{des}}(j\omega)G_i^{-1}(j\omega)W_2(j\omega)S(j\omega)| + |W_2(j\omega)K(j\omega)S(j\omega)| < 1 \quad \forall\omega. \tag{4.7}$$

This may be further expressed as

$$\begin{aligned} & |W_1(j\omega)P_{\text{des}}(j\omega)G_i^{-1}(j\omega)W_2(j\omega)S(j\omega)| < 1 - |W_2(j\omega)K(j\omega)S(j\omega)| \quad \forall\omega & (4.8)\\ \iff & \frac{|W_1(j\omega)P_{\text{des}}(j\omega)G_i^{-1}(j\omega)W_2(j\omega)S(j\omega)|}{1 - |W_2(j\omega)K(j\omega)S(j\omega)|} < 1 \quad \forall\omega & (4.9)\\ \iff & \left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S}{1 - |W_2 K S|} \right\|_\infty < 1. & (4.10) \end{aligned}$$

Note that $1 - |W_2(j\omega)K(j\omega)S(j\omega)| > 0 \quad \forall\omega$ is a result of the RS requirement. Therefore, the inequality in eqn (4.9) holds.

Theorem 6. *(DFFPC with Additive Uncertainty Model) A necessary and sufficient condition for robust performance is*

$$\| |W_1 P_{des} G_i^{-1} W_2 S| + |W_2 K S| \|_\infty < 1, \tag{4.11}$$

Proof. ($\Leftarrow$) Assume eqn (4.11), or equivalently,

$$\|W_2 K S\|_\infty < 1 \quad \text{and} \quad \left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S}{1 - |W_2 K S|} \right\|_\infty < 1 \tag{4.12}$$

Fix Δ and assume that each of the transfer functions are evaluated at an arbitrary point $j\omega$. Then,

$$1 = |1 + \Delta W_2 K S - \Delta W_2 K S| \leq |1 + \Delta W_2 K S| + |W_2 K S|$$

and therefore

$$1 - |W_2 K S| \leq |1 + \Delta W_2 K S|.$$

This implies that

$$\left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S}{1 - |W_2 K S|} \right\|_\infty \geq \left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S}{1 + \Delta W_2 K S} \right\|_\infty .$$

This along with eqn (4.12) yields

$$\left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S}{1 + \Delta W_2 K S} \right\|_\infty < 1 \tag{4.13}$$

($\Rightarrow$) Assume that

$$\|W_2 K S\|_\infty < 1 \quad \text{and} \quad \left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S}{1 + \Delta W_2 K S} \right\|_\infty < 1, \quad \forall \Delta. \tag{4.14}$$

Pick a frequency ω where

$$\frac{|W_1 P_{\text{des}} G_i^{-1} W_2 S|}{1 - |W_2 K S|} \tag{4.15}$$

is maximum. Now pick an all-pass Δ such that

$$1 - |W_2 K S| = |1 + \Delta W_2 K S|$$

.

The idea is that $\Delta(j\omega)$ should have a unity magnitude (i.e., $|\Delta(j\omega)| = 1 \quad \forall \omega$) and the phase should the negative of the phase of $W_2(j\omega)K(j\omega)S(j\omega)$. This may be done using an all-pass as was demonstrated in Section 2.3.1. Now,

$$\begin{aligned} \left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S}{1 - |W_2 K S|} \right\|_\infty &= \frac{|W_1 P_{\text{des}} G_i^{-1} W_2 S|}{1 - |W_2 K S|} \\ &= \frac{|W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S|}{1 + \Delta W_2 K S} \\ &\leq \left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S}{1 + \Delta W_2 K S} \right\|_\infty . \end{aligned} \tag{4.16}$$

The equality in eqn (4.16) follows from the fact that Δ is all-pass. From this and eqn (4.14), there follows eqn (4.12) and therefore eqn (4.11).

□

4.3.2 Multiplicative Uncertainty

For robustness analysis against model uncertainty, an uncertain plant model with multiplicative uncertainty is considered. In this case, the nominal plant model G is replaced with the perturbed plant model $\tilde{G} = (1 + W_2\Delta)G$. The block diagram for this model is shown in the Figure 4.5.

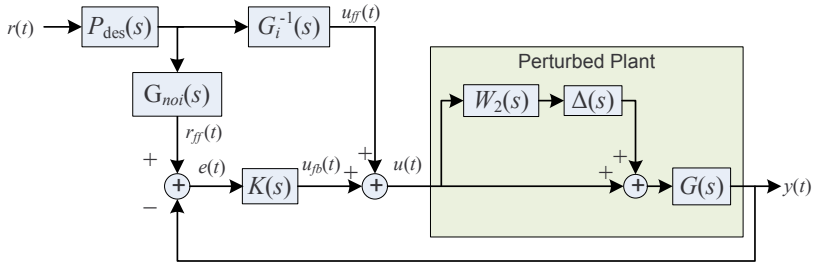

Figure 4.5: Uncertain LTI DFFPC Architecture with Multiplicative Uncertainty

The perturbed sensitivity function becomes

$$\begin{aligned}
\tilde{S}_{\text{DFFPC}} &= \frac{P_{\text{des}}G_{noi}}{1+(1+\Delta W_2)GK} - \frac{P_{\text{des}}G_i^{-1}(1+\Delta W_2)G}{1+(1+\Delta W_2)GK} \\
&= \frac{P_{\text{des}}G_{noi} - P_{\text{des}}G_{noi} - P_{\text{des}}G_{noi}\Delta W_2}{1+(1+\Delta W_2)GK} \\
&= -\frac{P_{\text{des}}G_{noi}\Delta W_2/(1+GK)}{(1+GK+\Delta W_2 GK)/(1+GK)} \\
&= -\frac{P_{\text{des}}G_{noi}\Delta W_2 S}{1+\Delta W_2 T}
\end{aligned} \tag{4.17}$$

It should be noted that the nominal and perturbed feedback characteristic equations are $(1 + L)$ and $(1 + L)(1 + \Delta W_2 T)$. These are identical to the characteristic equations for the feedback only architecture seen in Chapter 2.6.2. Using the same reasoning provided there, robust stability requires $\|W_2T\|_\infty < 1$ for all $\|\Delta\|_\infty \leq 1$.

For robust performance, we require robust stability and a norm bound on the weighted and perturbed sensitivity function. This is given by

$$\|W_2T\|_\infty < 1 \text{ (RS)} \quad \text{and} \quad \left\| W_1 \frac{P_{\text{des}}G_{noi}\Delta W_2 S}{1+\Delta W_2 T} \right\|_\infty < 1, \quad \forall \|\Delta\|_\infty \leq 1 \tag{4.18}$$

By a similar method used for the additive uncertainty case, a necessary and sufficient condition for robust performance is

Theorem 7. *(Multiplicative uncertainty model) A necessary and sufficient condition for robust performance is*

$$\| |W_1 P_{des} G_{noi} W_2 S| + |W_2 T| \|_\infty < 1, \tag{4.19}$$

Proof. The proof of the multiplicative case follows the same line of reasoning as the additive uncertain case. In a similar fashion, an all-pass filter is required in the necessity part of the proof. □

4.4 Discrete-Time Implementations

In most applications, the controller will be implemented in discrete-time using a processor (e.g., a microcontroller or a digital signal processor). In these cases, the feedforward controller must be designed for the equivalent discrete-time plant. When a zero-order hold digital to analog converter is used, it is appropriate to design the discrete-time feedforward controllers based on the zero order hold equivalent of the plant. The design process starts by determining the zero order hold equivalent plant

$$G(s) \longrightarrow G_{ZOH}(z) = \frac{K_{\mathrm{DC}} N_{nmp}(z) N_{mp}(z)}{D(z)} z^{-N_d}, \tag{4.20}$$

where the zero-order hold equivalent plant $G_{ZOH}(z)$ is calculated using standard methods (c.f., [41]). This process was demonstrated in Chapter 3.1.2. This new discrete-time equivalent plant is factored into $G_{ZOH}(z) = G_i(z) G_{noi}(z)$, where

$$G_i(z) = \frac{K_{\mathrm{DC}} N_{mp}(z)}{D(z)} \tag{4.21}$$

$$G_{noi}(z) = N_{nmp}(z) z^{-N_d}. \tag{4.22}$$

The two feedforward controllers are defined as

$$FF1(z) = P_{\text{des}}(z)G_{noi}(z) \tag{4.23}$$

$$FF2(z) = P_{\text{des}}(z)G_i^{-1}(z) \tag{4.24}$$

Now, the design objective is to design $P_{\text{des}}(z)$ with the same design constraints (i.e., relative degree and unity at DC constraints). Recall that DC is when $\omega = 0$. For discrete-time systems, $z = e^{j\omega Ts}$, so the unity at DC constraint becomes $P_{\text{des}}(1) = 1$,

In this scenario, the controller blocks are discrete-time blocks; however, the plant is still a continuous time system. This is illustrated in Figure 4.6.

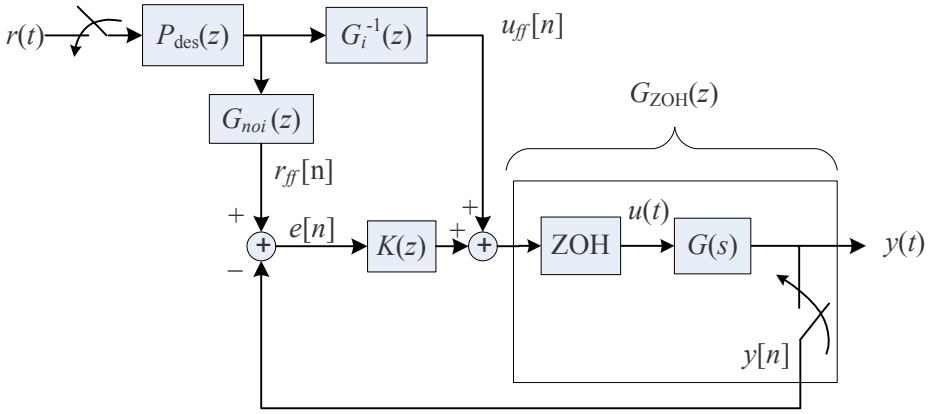

Figure 4.6: Discrete-time Implementation of the DFFPC Architecture.

The discrete-time implementation of DFFPC is shown in Figure 4.6. For this implementation, perfect tracking still holds and the nominal closed-loop map is given by $P_{\text{des}}(z)G_{noi}(z)$. To see this, observe that the discrete-time sensitivity function (from $r[n]$ to $e[n]$) is

$$\begin{aligned} S_{\text{DFFPC}}(z) &= P_{\text{des}}(z)G_{noi}(z)S(z) - P_{\text{des}}(z)G_i^{-1}(z)G(z)S(z) \\ &= P_{\text{des}}(z)G_{noi}(z)S(z) - P_{\text{des}}(z)G_{noi}(z)S(z) \\ &= 0, \end{aligned} \tag{4.25}$$

which verifies perfect tracking. Similarly, the discrete-time closed-loop transfer function (from $r[n]$ to $y[n]$) is given by

$$\begin{aligned} M_{\text{DFFPC}}(z) &= \frac{P_{\text{des}}(z)G_{noi}(z)G(z)K(z)}{1+G(z)K(z)} + \frac{P_{\text{des}}(z)G_i^{-1}(z)G(z)}{1+G(z)K(z)} \\ &= P_{\text{des}}(z)G_{noi}(z)\frac{1+G(z)K(z)}{1+G(z)K(z)} \\ &= P_{\text{des}}(z)G_{noi}(z), \end{aligned} \tag{4.26}$$

which is the discrete-time version of eqn (4.2). The discrete-time implementation will prove to be very useful in the adaptive cases where system identification is used to identify the discrete-time plant model $G_{ZOH}(z)$.

4.5 Conclusions

A method was presented that characterizes the general class of signals that may be perfectly tracked for all LTI systems of the form of eqn (3.1). These systems include:

- Stable and unstable systems.
- Minimum and non-minimum phase systems.
- Biproper and strictly proper systems.
- Systems with and without time delays.
- Continuous or discrete-time systems.

The fact that this architecture is able to guarantee perfect tracking for non-minimum phase systems (with possible time delays) is an extension to previously existing methods. The characterization of perfect tracking for any LTI system is due to the chosen plant factorization (i.e., $G(s) = G_{noi}(s)G_i(s)$), which allows for the addition of $G_{noi}(s)$ to one of the feedforward paths. This formulation will also prove

to be useful for the adaptation methods that will be presented in Chapter 7, which identify the plant $G(s)$ and use the plant factorization to update the two feedforward controllers.

Robustness tools were developed that may be used to measure the expected performance on the physical plant (for a given level of model uncertainty). These robustness tools may also be used in the $P_{\text{des}}(s)$ design process. This will be discussed in more detail in Chapter 6 and numerical examples will be provided in Chapter 9.

Chapter 5

Dual Feedforward Smith Predictor

This chapter provides another method for achieving perfect tracking (in the nominal case with no external disturbances) that is based on a feedforward augmentation to the standard Smith predictor (SP) discussed in Section 2.9. This method is referred to as a *dual feedforward Smith predictor* (DFFSP). As with the DFFPC architecture, this method is presented for single-input single-output (SISO) systems, with potential extensions to multiple-input multiple-output (MIMO) systems being discussed as future work in Chapter 12. The main difference between this architecture and the DFFPC architecture is that the non-invertible parts of the plant are handled in the feedback loop instead of the first feedforward path.

To begin, we examine the standard Smith predictor as we will use it for DFFSP. Robustness analysis of the standard Smith predictor is posed in the SISO case using the robustness framework utilized in Section 2.6. This particular Smith predictor structure and associated analysis tools will form the basis of the DFFSP architecture presented here. The primary application for the DFFSP architecture will be on stable plants. Due to the inherent limitation of a Smith predictor on an unstable plant, it is not recommended to use the DFFSP architecture on unstable plants. As with the DFFPC architecture, the discrete-time implementation of the DFFSP is presented.

5.1 The Smith Predictor as a Feedback Controller

The traditional Smith predictor for controlling plants with dead-time was discussed in Chapter 2.9. In this setting, the stable plant model was defined as $\hat{G}(s) = \hat{G}_{df}(s)e^{-s\hat{\tau}_d}$, where $\hat{G}_{df}(s)$ was the delay free stable plant model and $e^{-s\hat{\tau}_d}$ was the plant time-delay. For reference tracking, this Smith predictor structure is able to eliminate the effect of the time delay in the feedback loop in the nominal case with no external disturbances. Here, the time delay is a specific example of a non-minimum phase component. This idea may be trivially extended to eliminate any stable non-minimum phase components (e.g., RHP zeros) from the feedback loop.

In the setting considered here, we can use the invertible / non-invertible decomposition to define $G(s) = G_i(s)G_{noi}(s)$ and use the Smith predictor structure to eliminate the (stable) non-invertible dynamics $G_{noi}(s)$ from the feedback loop. In this case, a diagram of the Smith predictor that eliminates the (stable) non-invertible plant dynamics is shown in Figure 5.1.

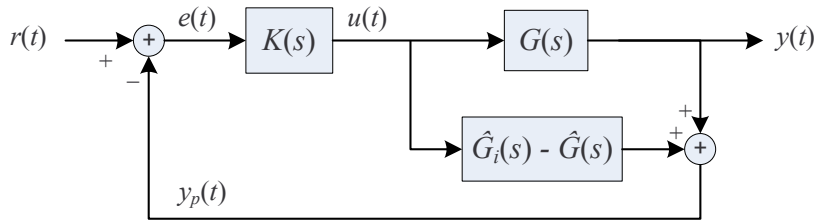

Figure 5.1: A Smith Predictor for a Non-minimum Phase Plant

For this chapter, we will assume that $\hat{G}(s) = G(s)$ (i.e., the nominal condition where the plant model is perfect). In order to address errors in the model, a bounded uncertainty description will be used. In the nominal case, the sensitivity function of a Smith predictor is given by

$$S_i(s) = \frac{E(s)}{R(s)} = \frac{1}{1 + G_i(s)K(s)}. \tag{5.1}$$

Note that the notation here is a subscript i to denote that only the invertible part

of the plant ($G_i(s)$) appears in the loop gain (e.g., the denominator polynomial of $S_i(s)$). In a similar fashion, the loop gain of a Smith predictor will be denoted $L_i(s) = G_i(s)K(s)$. Using the standard definition of complementary sensitivity function (i.e., $T(s) = 1 - S(s)$), the complementary sensitivity function of a Smith predictor is given by

$$T_i(s) = 1 - S_i(s) = \frac{G_i(s)K(s)}{1 + G_i(s)K(s)} = \frac{Y_p(s)}{R(s)}.. \tag{5.2}$$

Unlike a standard feedback loop, $T_i(s)$ is not the closed-loop transfer function (i.e., $T_i(s) \neq \frac{Y(s)}{R(s)}$), but instead is the transfer function from the reference input to the predicted output that is fed back (i.e., $T_i(s) = \frac{Y_p(s)}{R(s)}$). This is due to the fact that the non-invertible plant dynamics do not appear in the numerator of $T_i(s)$. The closed-loop transfer function of a Smith predictor is given by

$$M_i(s) = \frac{Y(s)}{R(s)} = \frac{G(s)K(s)}{1 + G_i(s)K(s)} = T_i(s)G_{noi}(s). \tag{5.3}$$

It is interesting to note that the closed-loop transfer function of a Smith predictor is similar to the closed-loop transfer function for DFFPC in the sense that it is the product of a stable minimum-phase transfer function with the non-invertible parts of the plant. In both situations, the stable minimum-phase part is used to shape the ideal closed-loop response. In the DFFPC case, this was done by picking $P_{\text{des}}(s)$, and in the Smith predictor case, this is done by designing $K(s)$ to shape $T_i(s)$. A disadvantage to the Smith predictor is that the feedback controller $K(s)$ is being designed for both (large signal) reference tracking and (small signal) disturbance rejection, which can be conflicting design objectives. This is addressed here by augmenting the Smith predictor with two feedforward controllers that will result in the perfect tracking of the signal $y_p(t)$.

Using methods similar to those presented in Chapter 2.9.4, the Smith predictor in Figure 5.1 may be equivalently written as a feedback only controller without affect-

ing the sensitivity or closed-loop transfer function. To see this, consider the Smith predictor block diagram shown in Figure 5.2.

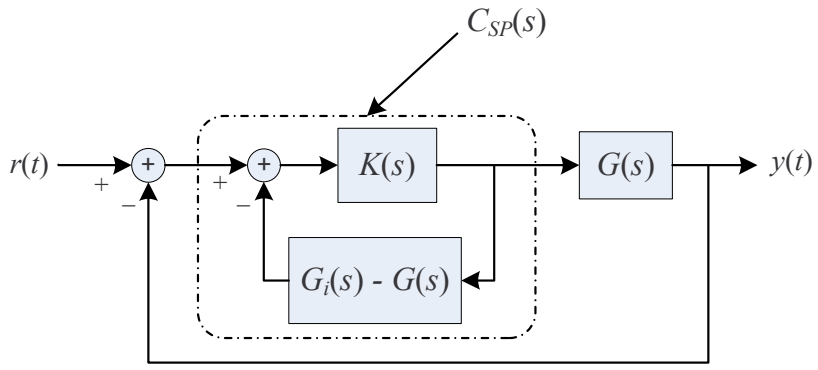

Figure 5.2: A Smith Predictor Drawn as a Single Feedback Controller

In this configuration, the equivalent Smith predictor feedback controller may be expressed as

$$C_{SP}(s) = \frac{K(s)}{1 + (G_i(s) - G(s))K(s)}. \tag{5.4}$$

While this does provide an equivalent block diagram from $r(t)$ to $y(t)$, it feeds back the plant output signal ($y(t)$) that includes the non-invertible dynamics (e.g., the process delay), and there is no longer a predicted signal (such as $y_p(t)$) in the feedback loop. This method for implementing a Smith predictor is generally not used. Instead, the internal model control structure given in Figure 5.1 is preferred [56]. Also, for achieving perfect tracking control using the DFFSP presented here, we will require the Smith predictor structure shown in Figure 5.1.

5.2 SISO Robustness Analysis of the Smith Predictor

In Chapter 2.9, it was shown that the Smith predictor is essentially a cancelation controller that cancels out the nominal plant dynamics. For this reason, it is not well

suited for unstable plants, which is the reason for restricting the initial discussion to stable plants.

The Smith predictor considered here is able to essentially eliminate the non-minimum phase dynamics from the feedback loop. However, real applications are never equal to the nominal case, which raises the question of robustness. One of the first papers to thoroughly address this issue was [60]. In this paper, the authors considered both uncertainty in the delay free plant model and uncertainty in the time delay, and provided some multiplicative weight selections to handle these uncertainties. These weights are similar in form to the multiplicative uncertainty weight provided for uncertain time delays provided in Chapter 2.6.3. In [61], the authors provide criteria for nominal stability, nominal performance, robust stability, and robust performance of a Smith predictor applied specifically to plants with time delays. This work was used in [62; 46; 63; 64] to provide methodologies for designing robust Smith predictors for systems with time delays. For completeness, the robust stability and robust performance results for the Smith predictor shown in Figure 5.1 are provided here. These results are analogous to those in [61], except that they are not restricted to Smith predictors that only eliminate time delays.

5.2.1 Additive Uncertainty

For the SISO robust performance criterion with additive uncertainty considered here, the Smith predictor is able to eliminate the non-invertible parts of the plant from the final robust performance criterion. This has been documented in the literature (c.f., [61]) using an analytical argument. Specifically it was shown that the satisfying μ robust performance on the delay free plant was equivalent to satisfying μ robust performance on the full plant. Here, we derive a similar criterion on the norm bounded weighted perturbed sensitivity function (i.e., the robust performance criteria used here). To begin, consider a Smith predictor with additive uncertainty, which is shown

in Figure 5.3.

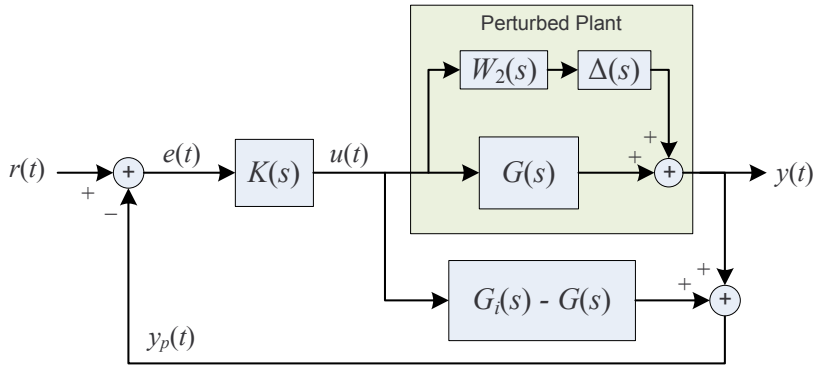

Figure 5.3: Smith Predictor with Additive Uncertainty

The perturbed sensitivity function becomes

$$\begin{aligned}
\tilde{S}_i &= \frac{1}{1+(G+\Delta W_2+G_i-G)K} \\
&= \frac{1}{1+(G_i+\Delta W_2)K} \\
&= \frac{1/(1+G_iK)}{(1+G_iK+\Delta W_2K)/(1+G_iK)} \\
&= \frac{S_i}{1+\Delta W_2KS_i}
\end{aligned} \tag{5.5}$$

For the Smith predictor with additive uncertainty, the nominal and perturbed feedback characteristic equations are $(1+G_iK)$ and $(1+G_iK)(1+\Delta W_2KS_i)$. By using the same reasoning provided in Chapter 2.6.1, robust stability requires for all $\|\Delta\|_\infty \leq 1$ that $\|W_2KS_i\|_\infty < 1$.

For robust performance, we require robust stability and a norm bound on the weighted perturbed sensitivity function. This is given by

$$\|W_2KS_i\|_\infty < 1 \text{ (RS)} \quad \text{and} \quad \left\| W_1 \frac{S_i}{1+\Delta W_2KS_i} \right\|_\infty < 1, \quad \forall \|\Delta\|_\infty \leq 1. \tag{5.6}$$

Using an analogous method from Chapter 2.6.1, the following theorem may be proved.

Theorem 8. *(Additive uncertainty model) A necessary and sufficient condition for robust performance with the Smith predictor in Figure 5.3 is*

$$\| |W_1 S_i| + |W_2 K S_i| \|_\infty < 1, \tag{5.7}$$

Proof. The proof is analogous to the Theorem 2 in Chapter 2.6.1 □

For the additive uncertainty case, the plant in the feedback loop is $G + \Delta W_2 + G_i - G = G_i + \Delta W_2$. Since our robustness criteria are defined by the sensitivity function, an equivalent block diagram of the perturbed sensitivity function is shown in Figure 5.4. Note that the actual perturbed output ($\tilde{y}(t)$) does not appear in the perturbed block diagram, but rather the perturbed prediction ($\tilde{y}_p(t)$).

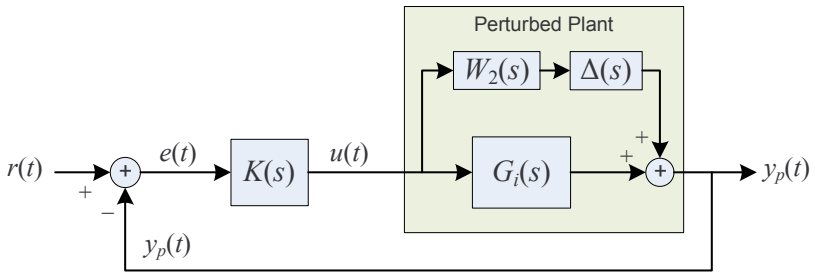

Figure 5.4: Smith Predictor with Additive Uncertainty

For the additive uncertainty case, this observation does not change the additive uncertainty robust performance criteria, since the criteria is based on $S_i(s)$, which only contains the invertible part of the plant $G_i(s)$ and not the full plant $G(s)$. A similar observation may not be made for the multiplicative uncertainty case.

5.2.2 Multiplicative Uncertainty

Unlike the additive uncertainty case given in the previous section, the SISO Smith predictor robust performance criterion with multiplicative uncertainty does not fully eliminate the non-invertible parts of the full plant. Instead, the full plant will appear

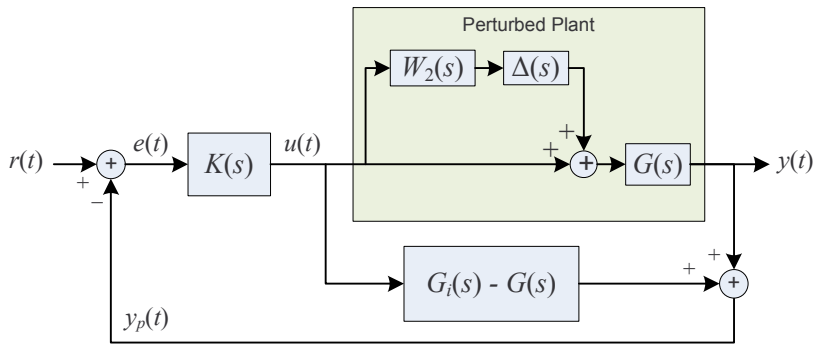

Figure 5.5: Smith Predictor with Multiplicative Uncertainty

in the final robust performance criterion. To see this, consider, consider a Smith predictor with multiplicative uncertainty shown in Figure 5.5.

The perturbed sensitivity function becomes

$$\begin{aligned}
\tilde{S}_i &= \frac{1}{1+(G+\Delta W_2 G+G_i-G)K} \\
&= \frac{1}{1+(G_i+\Delta W_2 G)K} \\
&= \frac{1/(1+G_iK)}{(1+G_iK+\Delta W_2 GK)/(1+G_iK)} \\
&= \frac{S_i}{1+\Delta W_2 M_i}
\end{aligned} \tag{5.8}$$

For the Smith predictor with multiplicative uncertainty, the nominal and perturbed feedback characteristic equations are $(1+G_iK)$ and $(1+G_iK)(1+\Delta W_2 M_i)$. By using the same reasoning provided in Section 2.6.1, robust stability requires for all $\|\Delta\|_\infty \leq 1$ that $\|W_2 M_i\|_\infty < 1$. Notice that robust stability is a norm condition on the weighted nominal closed-loop transfer function $W_2(s)M_i(s)$. In the traditional feedback case, the nominal closed-loop transfer function was also the complementary sensitivity function (i.e., $M(s) = T(s)$ in the traditional feedback case presented in Section 2.5).

For robust performance, we require robust stability and a norm bound on the weighted perturbed sensitivity function. This is given by

$$\|W_2 M_i\|_\infty < 1 \text{ (RS)} \quad \text{and} \quad \left\| W_1 \frac{S_i}{1 + \Delta W_2 M_i} \right\|_\infty < 1, \quad \forall \|\Delta\|_\infty \leq 1. \tag{5.9}$$

Using an analogous method from Section 2.6.2, the following theorem may be proved.

Theorem 9. *(Multiplicative uncertainty model) A necessary and sufficient condition for robust performance with the Smith predictor in Figure 5.5 is*

$$\| |W_1 S_i| + |W_2 M_i| \|_\infty < 1, \tag{5.10}$$

Proof. The proof is analogous to the Theorem 4 in Chapter 2.6.2 □

As with the additive uncertainty case, the result is analogous to the feedback only case with only the invertible part of the plant appearing in the feedback loop (i.e., the denominator of M_i, which is $1 + G_i K$, only contains G_i and not $G = G_{noi} G_i$). However, the numerator of M_i does contain the full plant $G = G_{noi} G_i$, which means that a block diagram rearrangement that only includes G_i is not possible for the multiplicative uncertainty case.

5.3 Dual Feedforward Smith Predictor

The DFFSP architecture is another method for achieving perfect tracking for minimum and non-minimum phase systems in the nominal case without disturbances. This method is similar to the DFFPC architecture presented in the previous chapter in the sense that two feedforward controllers are used to provide perfect tracking. The differences are that we restrict the plant model to be stable and the non-invertible plant dynamics ($G_{noi}(s)$) are addressed by the Smith predictor feedback loop. The restriction to stable plants makes this architecture less general than the DFFPC architecture in the previous chapter. The block diagram for DFFSP is given in Figure 5.6. The blocks G, G_i^{-1} and P_{des} are analogous to the blocks used in Chapters 3 and 4.

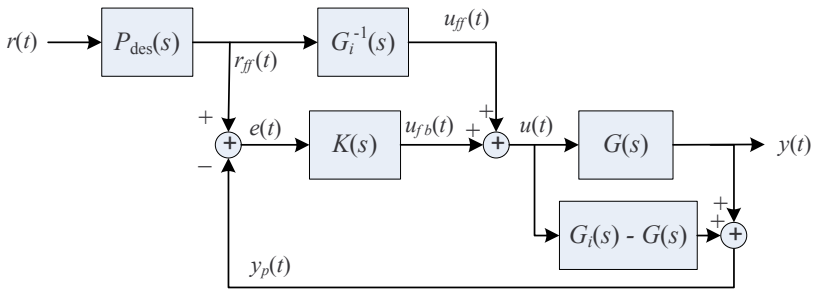

Figure 5.6: Nominal LTI DFFSP Architecture

There are three design constraints that must be satisfied by $P_{\text{des}}(s)$, namely

1. The steady state gain from $r(t)$ to $r_{ff}(t)$ must be unity (i.e., $P_{\text{des}}(0) = 1$)

2. The feedforward controller $FF1(s) = P_{\text{des}}(s)$ must result in a stable proper transfer function

3. The feedforward controller $FF2(s) = P_{\text{des}}(s)G_i^{-1}(s)$ must result in a stable proper transfer function

The first two conditions are required so that the filtered reference $r_{ff}(t)$ asymptotically tracks the ideal reference $r(t)$, and hence $y(t)$ asymptotically tracks $r(t)$. The third condition is required to make the second feedforward controller $FF2(s)$ realizable in hardware. Unlike the DFFPC architecture, G_{noi} does not appear in the feedforward path, which means that the feedforward constraints do not include G_{noi}. Instead, the non-minimum phase components (G_{noi}) are addressed in the feedback path of the Smith predictor. The above conditions will be met when the following hold:

- $P_{\text{des}}(s)$ is stable and $P_{\text{des}}(0) = 1$ (satisfies first condition above), and

- Relative degree of the $P_{\text{des}}(s)$ is greater than or equal to the relative degree of $G_i(s)$ (satisfies second and third conditions above)

These are the same conditions required for the DFFPC architecture in the previous chapter. The sensitivity transfer function for DFFSP is

$$\begin{aligned} S_{\text{DFFSP}}(s) &= P_{\text{des}}(s)S_i(s) - P_{\text{des}}(s)G_i^{-1}(s)G_i(s)S_i(s) \\ &= P_{\text{des}}(s)S_i(s) - P_{\text{des}}(s)S_i(s) \\ &= 0, \end{aligned} \tag{5.11}$$

Similarly to the DFFPC case, *nominal performance* is perfect for the DFFSP architecture and the condition $\|W_1 S_{\text{DFFSP}}\|_\infty < 1$ is trivially satisfied for any finite $W_1(s)$. However, the sensitivity function is now defined as the gain from $r(t)$ to $e_{\text{DFFSP}}(t) = y_p(t) - r_{ff}(t)$, whereas the DFFPC sensitivity was defined as the gain from $r(t)$ to $e_{\text{DFFPC}}(t) = y(t) - r_{ff}(t)$. This means that these sensitivity functions are defined on different signals.

As with the DFFPC architecture, the intention of this method is not to remove the non-minimum phase components from the plant, but rather utilize them to define a class of signals that may be perfectly tracked. To see this, consider the closed-loop transfer function (from $r(t)$ to $y(t)$):

$$\begin{aligned} M_{\text{DFFSP}}(s) &= \frac{P_{\text{des}}(s)G(s)K(s)}{1+G_i(s)K(s)} + \frac{P_{\text{des}}(s)G_i^{-1}(s)G(s)}{1+G_i(s)K(s)} \\ &= \frac{P_{\text{des}}(s)G_{noi}(s)G_i(s)K(s) + P_{\text{des}}(s)G_{noi}(s)}{1+G_i(s)K(s)} \\ &= P_{\text{des}}(s)G_{noi}(s)\frac{1+G_i(s)K(s)}{1+G_i(s)K(s)} \\ &= P_{\text{des}}(s)G_{noi}(s), \end{aligned} \tag{5.12}$$

which results in the same nominal closed-loop transfer function as the DFFPC architecture.

As with the DFFPC architecture, the non-minimum phase components appear in the closed-loop transfer function of the DFFSP architecture as $G_{noi}(s)$ and

$P_{\text{des}}(s)G_{noi}(s)$ defines the class of signals that may be perfectly tracked. The nominal design objective is to design $P_{\text{des}}(s)$ to get the desired closed-loop characteristics. Even though the closed-loop poles of the Smith predictor (i.e., the roots of $1 + G_i(s)K(s)$) do not appear in the nominal closed-loop transfer function $M_{\text{DFFSP}}(s)$, they will affect performance in terms of disturbance rejection and to correct for modeling errors. Also, internal stability is always required. Therefore, the controller $K(s)$ should be designed to guarantee internal stability and to provide the desired level of robust performance to disturbance rejection. Since the feedforward signals are essentially bounded external signals to the feedback loop, they do not affect closed-loop stability. Therefore, it makes sense that nominal stability is unaffected by the addition of the feedforward components. By the same argument, robust stability is also unaffected by the addition of the feedforward components. This is addressed again later in the chapter.

5.4 Controller Design

While the design constraints given above are required for the feedforward controllers to be proper, they also provide limitations on achievable performance. In particular, $P_{\text{des}}(s)G_{noi}(s)$ determines the class of signals that may be perfectly tracked, and $P_{\text{des}}(s)G_i^{-1}(s)$ is the feedforward controller that will provide the associated control signal to achieve perfect tracking. The class of signals that may be perfectly tracked includes the effects of the non-minimum components that neither feedforward nor feedback controllers can eliminate. This phenomenon was discussed in Chapter 2.8.

There are two pieces that need to be designed, namely the feedforward and feedback controllers. The feedforward controller is designed for "large" signal reference tracking and the feedback controller is designed "small signal" disturbance rejection. A full discussion of the feedforward design is postponed until Chapter 6. The design of the feedback controller is cast as a traditional Smith predictor feedback design

(e.g., provide internal stability and desired performance). There are many standard methods for designing the feedback controller in Smith predictor (c.f., [65; 66; 67]). A particular example of a Smith predictor feedback design that is well suited for this problem is presented next.

5.4.1 Feedback Controller Design

The objective of the feedback controller in the DFFSP architecture is analogous to the DFFPC case in the sense that it is designed to provide internal stability, disturbance rejection performance, and to correct for modeling errors in the two feedforward paths. As with DFFPC, the DFFSP feedback control design is cast as a disturbance rejection problem instead of a reference tracking problem. The main difference between the two methods is that the known non-invertible dynamics are addressed in the feedback loop. When the performance criteria are properly assigned, this allows for a feedback controller design that only uses the stable minimum-phase dynamics in the controller synthesis. To see this, consider the Smith predictor synthesis interconnect shown in Figure 5.7.

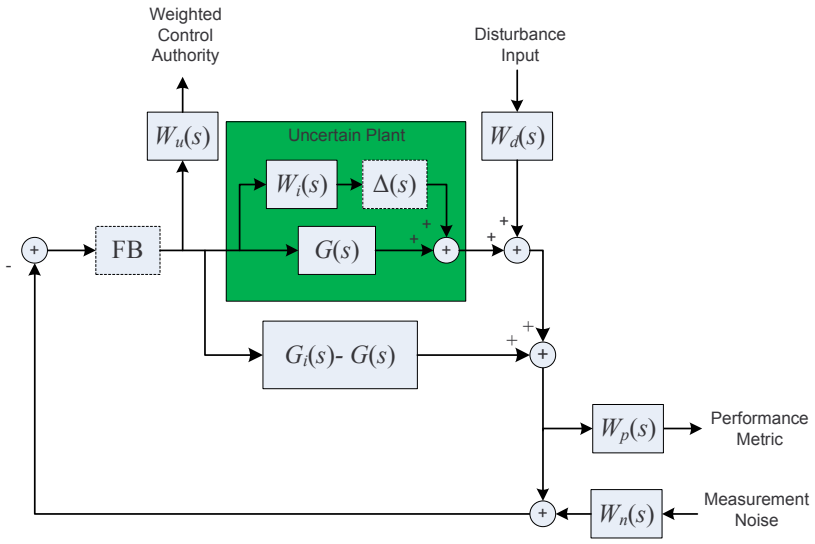

Figure 5.7: Robust Controller Synthesis Interconnect for Disturbance Rejection with Additive Uncertainty

In Figure 5.7, the objective of the optimization is to minimize the effects that the disturbance will have on the predicted output $y_p(t)$ and not the actual plant output $y(t)$. The reason for this is that the feedback controller will use error between the filtered reference and the predicted out $y_p(t)$ for its feedback control signal (and not the error between the filtered reference and actual output $y(t)$). With this problem formulation, the feedback controller synthesis may be recast as the synthesis diagram shown in Figure 5.8.

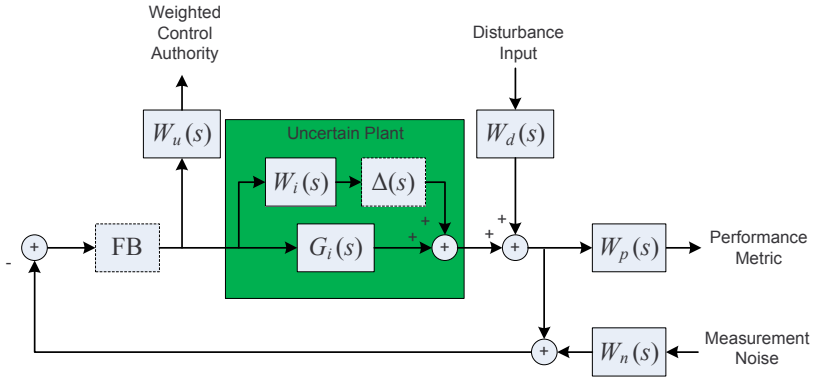

Figure 5.8: Reduced Robust Controller Synthesis Interconnect for Disturbance Rejection with Additive Uncertainty

After these observations have been made, the weight selection is similar to the methods used for the feedback design in Chapter 4.2.1. Since the design here is done only on the invertible dynamics of the plant, the actual values for the optimization weights may differ. However, the methodology will be the same.

5.5 Robustness Analysis

This section explores the robust performance metrics on plants with additive and multiplicative uncertainty.

5.5.1 Additive Uncertainty

For robustness analysis against model uncertainty, an uncertain plant model with additive uncertainty is considered. In this case, the nominal plant model G is replaced

with the perturbed plant model $\tilde{G} = G + W_2\Delta$. The block diagram for this model is shown in Figure 5.9.

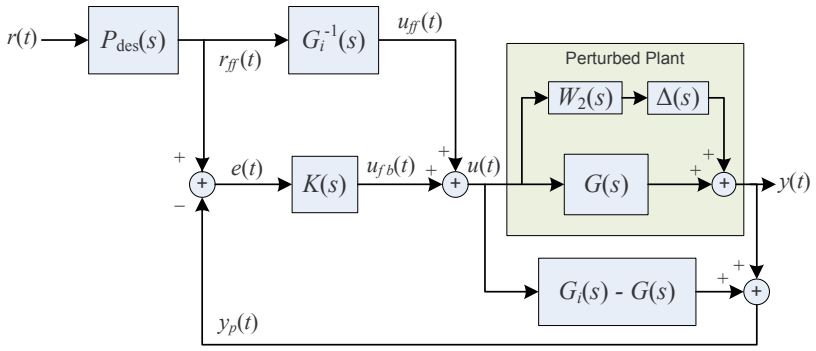

Figure 5.9: Uncertain LTI DFFSP Architecture with Additive Uncertainty

The perturbed sensitivity function becomes

$$\begin{aligned}
\tilde{S}_{\text{DFFSP}} &= \frac{P_{\text{des}}}{1+(G_i+\Delta W_2)K} - \frac{P_{\text{des}}G_i^{-1}(G_i+\Delta W_2)}{1+(G_i+\Delta W_2)K} \\
&= \frac{P_{\text{des}} - P_{\text{des}} - P_{\text{des}}G_i^{-1}\Delta W_2}{1+(G_i+\Delta W_2)K} \\
&= -\frac{P_{\text{des}}G_i^{-1}\Delta W_2/(1+G_iK)}{(1+G_iK+\Delta W_2K)/(1+G_iK)} \\
&= -\frac{P_{\text{des}}G_i^{-1}\Delta W_2 S_i}{1+\Delta W_2 K S_i}
\end{aligned} \tag{5.13}$$

The nominal and perturbed feedback characteristic equations (i.e., the denominator polynomials of the Smith predictor feedback loops) are $(1 + L_i)$ and $(1+L_i)(1+\Delta W_2 K S_i)$. These are identical to the characteristic equations for the Smith predictor only architecture presented earlier in the chapter. Using the same reasoning provided there, robust stability requires for all $\|\Delta\|_\infty \leq 1$ that $\|W_2 K S_i\|_\infty < 1$.

For robust performance, we require robust stability and a norm bound on the weighted perturbed sensitivity function. This is given by

$$\|W_2 K S_i\|_\infty < 1 \text{ (RS)} \quad \text{and} \quad \left\| W_1 \frac{P_{\text{des}}G_i^{-1}\Delta W_2 S_i}{1+\Delta W_2 K S_i} \right\|_\infty < 1, \quad \forall \quad \|\Delta\|_\infty \leq 1. \tag{5.14}$$

These criteria may be shown to be an exact test for robust performance. To show this, the above equation, with a slight abuse of notation, may equivalently be expressed as

$$s \mapsto |W_1(s)P_{\text{des}}(s)G_i^{-1}(s)W_2(s)S_i(s)| + |W_2(s)K(s)S_i(s)|, \tag{5.15}$$

which is denoted $|W_1 P_{\text{des}} G_i^{-1} W_2 S_i| + |W_2 K S_i|$. Now, a necessary and sufficient condition for robust performance is given by

$$\||W_1 P_{\text{des}} G_i^{-1} W_2 S_i| + |W_2 K S_i|\|_\infty < 1, \tag{5.16}$$

or equivalently,

$$|W_1(j\omega)P_{\text{des}}(j\omega)G_i^{-1}(j\omega)W_2(j\omega)S_i(j\omega)| + |W_2(j\omega)K(j\omega)S_i(j\omega)| < 1 \quad \forall\omega. \tag{5.17}$$

This may be further expressed as

$$\begin{aligned}
& |W_1(j\omega)P_{\text{des}}(j\omega)G_i^{-1}(j\omega)W_2(j\omega)S_i(j\omega)| < 1 - |W_2(j\omega)K(j\omega)S_i(j\omega)| && (5.18)\\
\iff & \frac{|W_1(j\omega)P_{\text{des}}(j\omega)G_i^{-1}(j\omega)W_2(j\omega)S_i(j\omega)|}{1 - |W_2(j\omega)K(j\omega)S_i(j\omega)|} < 1 \quad \forall\omega && (5.19)\\
\iff & \left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S_i}{1 - |W_2 K S_i|} \right\|_\infty < 1. && (5.20)
\end{aligned}$$

Note that $1 - |W_2(j\omega)K(j\omega)S_i(j\omega)| > 0 \quad \forall\omega$ is a result of the RS requirement. Therefore, the inequality in eqn (5.19) holds.

Theorem 10. *(DFFSP with Additive Uncertainty Model) A necessary and sufficient condition for robust performance is*

$$\||W_1 P_{des} G_i^{-1} W_2 S_i| + |W_2 K S_i|\|_\infty < 1, \tag{5.21}$$

Proof. ($\Leftarrow$) Assume eqn (5.21), or equivalently,

$$\|W_2KS_i\|_\infty < 1 \quad \text{and} \quad \left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S_i}{1 - |W_2 K S_i|} \right\|_\infty < 1 \tag{5.22}$$

Fix Δ and assume that each of the transfer functions are evaluated at an arbitrary point $j\omega$. Then,

$$1 = |1 + \Delta W_2 K S_i - \Delta W_2 K S_i| \leq |1 + \Delta W_2 K S_i| + |W_2 K S_i|$$

and therefore

$$1 - |W_2 K S_i| \leq |1 + \Delta W_2 K S_i|.$$

This implies that

$$\left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S_i}{1 - |W_2 K S_i|} \right\|_\infty \geq \left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S_i}{1 + \Delta W_2 K S_i} \right\|_\infty .$$

This along with eqn (5.22) yields

$$\left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S_i}{1 + \Delta W_2 K S_i} \right\|_\infty < 1 \tag{5.23}$$

($\Rightarrow$) Assume that

$$\|W_2KS_i\|_\infty < 1 \quad \text{and} \quad \left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S_i}{1 + \Delta W_2 K S_i} \right\|_\infty < 1, \quad \forall \Delta. \tag{5.24}$$

Pick a frequency ω where

$$\frac{|W_1 P_{\text{des}} G_i^{-1} W_2 S_i|}{1 - |W_2 K S_i|} \tag{5.25}$$

is maximum. Now pick an all-pass Δ such that

$$1 - |W_2 K S_i| = |1 + \Delta W_2 K S_i|$$

.

The idea is that $\Delta(j\omega)$ should have a unity magnitude (i.e., $|\Delta(j\omega)| = 1 \quad \forall\omega$) and the phase should the negative of the phase of $W_2(j\omega)K(j\omega)S_i(j\omega)$. This may be done using an all-pass as was demonstrated in Section 2.3.1. Now,

$$\begin{aligned} \left\| \frac{W_1 P_{\text{des}} G_i^{-1} W_2 S_i}{1 - |W_2 K S_i|} \right\|_\infty &= \frac{|W_1 P_{\text{des}} G_i^{-1} W_2 S_i|}{1 - |W_2 K S_i|} \\ &= \frac{|W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S_i|}{1 + \Delta W_2 K S_i} \\ &\leq \left\| \frac{W_1 P_{\text{des}} G_i^{-1} \Delta W_2 S_i}{1 + \Delta W_2 K S_i} \right\|_\infty . \end{aligned} \tag{5.26}$$

The equality in eqn (5.26) follows from the fact that Δ is all-pass. From this and eqn (5.24), there follows eqn (5.22) and therefore eqn (5.21).

□

5.5.2 Multiplicative Uncertainty

For robustness analysis against model uncertainty, an uncertain plant model with multiplicative uncertainty is considered here. In this case, the nominal plant model G is replaced with the perturbed plant model $\tilde{G} = (1 + W_2\Delta)G$. The block diagram for this model is shown in the Figure 5.10.

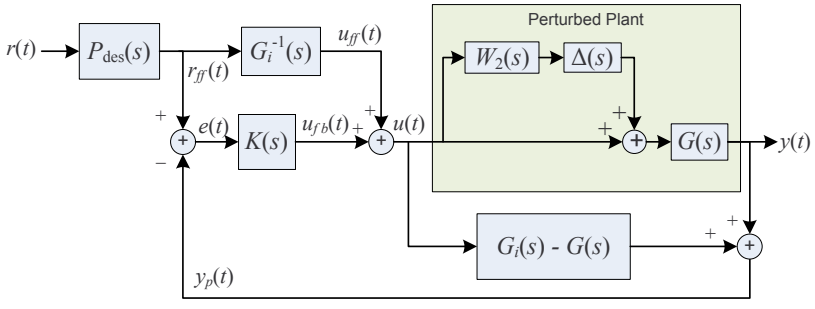

Figure 5.10: Uncertain LTI DFFSP Architecture with Multiplicative Uncertainty

The perturbed sensitivity function becomes

$$\begin{aligned}
\tilde{S}_{\text{DFFSP}} &= \frac{P_{\text{des}} - P_{\text{des}} G_i^{-1}((1+\Delta W_2)G + G_i - G)}{1 + ((1+\Delta W_2)G + G_i - G)K} \\
&= \frac{P_{\text{des}} - P_{\text{des}} - P_{\text{des}} G_{noi} \Delta W_2}{1 + (G_i + \Delta W_2 G)K} \\
&= -\frac{P_{\text{des}} G_{noi} \Delta W_2 / (1 + G_i K)}{(1 + G_i K + \Delta W_2 G K)/(1 + G_i K)} \\
&= -\frac{P_{\text{des}} G_{noi} \Delta W_2 S_i}{1 + \Delta W_2 M_i}, \qquad (5.27)
\end{aligned}$$

It should be noted that the nominal and perturbed feedback characteristic equations are $(1 + L_i)$ and $(1 + L_i)(1 + \Delta W_2 M_i)$. These are identical to the characteristic equations for the Smith predictor only architecture presented earlier in this chapter. Using the same reasoning provided there, robust stability requires for all $\|\Delta\|_\infty \leq 1$ that $\|W_2 M_i\|_\infty < 1$.

For robust performance, we require robust stability and a norm bound on the weighted and perturbed sensitivity function. This is given by

$$\|W_2 M_i\|_\infty < 1 \text{ (RS)} \quad \text{and} \quad \left\| W_1 \frac{P_{\text{des}} G_{noi} \Delta W_2 S_i}{1 + \Delta W_2 M_i} \right\|_\infty < 1, \quad \forall \|\Delta\|_\infty \leq 1 \qquad (5.28)$$

By a similar method to that used for the additive uncertainty case, a necessary and sufficient condition for robust performance is

Theorem 11. *(DFFSP with Multiplicative Uncertainty Model) A necessary and sufficient condition for robust performance is*

$$\| |W_1 P_{des} G_{noi} W_2 S_i| + |W_2 M_i| \|_\infty < 1, \qquad (5.29)$$

Proof. The proof of the multiplicative case follows the same line of reasoning as the additive uncertain case. In a similar fashion, an all-pass filter is required in the necessity part of the proof. □

5.6 Discrete-Time Implementations

In most applications, the controller will be implemented in discrete-time using a processor. Similar to the DFFPC architecture, the feedforward and Smith predictor controllers for the DFFSP architecture may be defined based on a zero order hold equivalent discrete-time plant. For reference, the zero order hold discrete-time equivalent plant may be obtained via

$$G(s) \longrightarrow G_{ZOH}(z) = \frac{K_{\mathrm{DC}} N_{nmp}(z) N_{mp}(z)}{D(z)} z^{-N_d}, \tag{5.30}$$

where the zero-order hold equivalent plant $G_{ZOH}(z)$ is calculated using standard methods (c.f., [41]). This process was demonstrated in Chapter 3.1.2. This new discrete-time equivalent plant may be factored into $G_{ZOH}(z) = G_i(z)G_{noi}(z)$, where

$$G_i(z) = \frac{K_{\mathrm{DC}} N_{mp}(z)}{D(z)} \tag{5.31}$$

$$G_{noi}(z) = N_{nmp}(z) z^{-N_d}. \tag{5.32}$$

For the DFFSP architecture, the two feedforward controllers are defined as

$$FF1(z) = P_{\mathrm{des}}(z) \tag{5.33}$$

$$FF2(z) = P_{\mathrm{des}}(z) G_i^{-1}(z), \tag{5.34}$$

and the unity at DC constraint remains $P_{\mathrm{des}}(1) = 1$,

In this scenario, the controller blocks are discrete-time blocks; however, the plant is still a continuous time system. This is illustrated in Figure 5.11.

The discrete-time implementation of DFFSP is shown in Figure 5.11. For this implementation, perfect tracking still holds and the nominal closed-loop map is given by $P_{\mathrm{des}}(z)G_{noi}(z)$. To see this, observe that the discrete-time sensitivity function (from $r[n]$ to $e[n]$) is

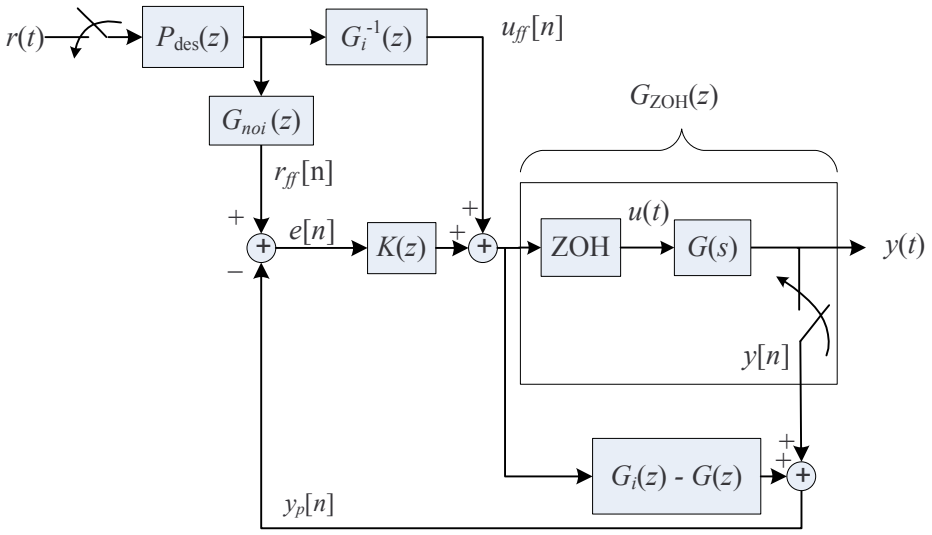

Figure 5.11: Discrete-time Implementation of the DFFSP Architecture.

$$\begin{aligned} S_{\text{DFFSP}}(z) &= P_{\text{des}}(z)S_i(z) - P_{\text{des}}(z)G_i^{-1}(z)G_i(z)S_i(z) \\ &= P_{\text{des}}(z)S_i(z) - P_{\text{des}}(z)S_i(z) \\ &= 0, \end{aligned} \tag{5.35}$$

which verifies perfect tracking. Similarly, the discrete-time closed-loop transfer function (from $r[n]$ to $y[n]$) is given by

$$\begin{aligned} M_{\text{DFFSP}}(z) &= \frac{P_{\text{des}}(z)G(z)K(z)}{1+G_i(z)K(z)} + \frac{P_{\text{des}}(z)G_i^{-1}(z)G(z)}{1+G_i(z)K(z)} \\ &= \frac{P_{\text{des}}(z)G_{noi}(z)G_i(z)K(z) + P_{\text{des}}(z)G_{noi}(z)}{1+G_i(z)K(z)} \\ &= P_{\text{des}}(z)G_{noi}(z)\frac{1+G_i(z)K(z)}{1+G_i(z)K(z)} \\ &= P_{\text{des}}(z)G_{noi}(z), \end{aligned} \tag{5.36}$$

which is the discrete-time version of eqn (5.12). The discrete-time implementation will prove to be very useful in the adaptive cases where system identification is used to refine the zero-order hold equivalent plant model $G_{ZOH}(z)$.

5.7 Conclusions

Another method was presented that characterizes the general class of signals that may be perfectly tracked for all *stable* LTI non-minimum phase system of the form of eqn (3.1). Note that this Smith predictor architecture is restricted to stable plants, unlike the tools from Chapter 4. However, for stable plants, it does provide another method for achieving perfect tracking. In Chapter 2, it was shown that the disturbance rejection properties could be improved by using a modified Smith predictor; however, that option is not available in this setting, because of the specified Smith predictor feedback loop required for perfect tracking. In particular, the modified model must be $G_m(s) = G_i(s)$ for perfect tracking to hold, and not a free parameter that may be used to improve the disturbance rejection properties. Therefore, the disturbance rejection performance will be determined by the feedback controller $K(s)$ (as it was in the DFFPC architecture). If the same order feedback controller $K(s)$ is used for both the DFFPC and DFFSP architectures, the DFFSP architecture will result in a higher order controller architecture, because of the extra states introduced by the term $G_i(s) - G(s)$ in the feedback loop. This higher order controller may be less desirable to implement. The performance trade-offs between this method and the DFFPC architecture will be studied using numerical examples in Chapter 9.

Since part of the model (i.e., $G_i(s) - G(s)$) appears in the feedback loop, it is more difficult to stably adapt. Instead of adapting the components inside the feedback loop, the feedforward part of the DFFSP structure may be modified for adaptation. The modification makes the adaptive DFFSP architecture look like the adaptive DFFPC architecture, which suggests that the DFFPC architecture is preferable to the DFFSP architecture when adaptation is being used.

Chapter 6

Robust and Optimal Feedforward Design

The role of the feedforward controllers is to provide nominal tracking performance. There are three pieces that need to be defined for the feedforward controllers, namely $P_{\text{des}}(s)$, $G_{noi}(s)$, and $G_i^{-1}(s)$. While $G_{noi}(s)$ and $G_i^{-1}(s)$ come straight from the plant model, $P_{\text{des}}(s)$ is designed for performance. In general, the desired response will be a multiobjective design that will have conflicting design objectives. As an example, it may be desirable to have a fast rise time to steps without using too much control authority in the presence of model uncertainty. Here more model uncertainty and less control authority will increase the step rise time. These types of trade-offs are common in many control applications. However, the design methodology is different for feedforward versus the usual feedback case. For example, it is possible to directly design the feedforward controllers to provide a specific rise time with no overshoot, which is more difficult to do with a (weighted) feedback controller design.

Design constraints have been provided to ensure that the feedforward controllers are proper and that the output $y(t)$ asymptotically tracks the input $r(t)$. These design constraints also provide limitations on achievable performance. In particular, $P_{\text{des}}(s)G_{noi}(s)$ determines the class of signals that may be perfectly tracked, and $P_{\text{des}}(s)G_i^{-1}(s)$ is the feedforward controller that will provide perfect tracking.

Two methodologies for designing these pieces will be discussed here, namely a

direct model-based design and robust/optimal feedforward controller design. The robust and optimal feedforward designs were inspired by the work presented in [68]. The work of [68] is investigated further here to show how the current robust and optimal designs may be used to design a $P_{\text{des}}(s)$. As will be demonstrated for the controller synthesis methods chosen here, the final robust and optimal feedforward controllers may be constructed to be of the form $P_{\text{des}}(s)G_i^{-1}(s)$. From observations of the final feedforward controller, model reduction techniques may be used to approximate a reduced order $P_{\text{des}}(s)$.

6.1 Direct Model-Based Feedforward Design

For the direct model based design, $P_{\text{des}}(s)$ must adhere to the following design constraints.

$P_{\text{des}}(s)$ Design Constraints:

- $P_{\text{des}}(s)$ stable, and relative degree of $P_{\text{des}}(s) \geq$ Relative degree of $G_i(s)$ (assuming $G(s)$ is proper).
- $P_{\text{des}}(0) = 1$ (assuming that $G_{noi}(0) = 1$ from the problem formulation).

Adhering to these constraints will guarantee that the the transfer functions $P_{\text{des}}(s)G_{noi}(s)$ and $P_{\text{des}}(s)G_i^{-1}(s)$ are proper, and therefore may be implemented in physical hardware. This constraint also provides a limitation on the achievable ideal closed-loop response $P_{\text{des}}(s)G_{noi}(s)$. For example, if the relative degree of $G_i(s)$ is three and there are no right-half plane zeros (i.e., $G_{noi}(s) = 1$), then the the class of signals that can be perfectly tracked must have at least third order roll off (i.e., 60 dB/decade or more) at high frequencies. However, the frequency at which this roll-off occurs can be pushed out to arbitrarily high frequencies. The trade-off is that a $P_{\text{des}}(s)$ with a larger bandwidth will generally lead to larger and faster control signals. Therefore, analysis should be performed to determine that the actuator will

have enough control authority to respond to the control signals that are required for a given $P_{\text{des}}(s)$. If the control signals fall outside of the operating range of an actuator, then the bandwidth of $P_{\text{des}}(s)$ should be decreased. Also, increasing the bandwidth of $P_{\text{des}}(s)$ will generally reduce the overall system robustness. Next, numerical examples are presented that show how the ideal step response may be shaped using a specified structure for $P_{\text{des}}(s)$. The robustness trade-offs will be demonstrated later in the illustrative examples of Chapter 9.

6.1.1 Numerical Example

A sample non-minimum phase system is selected to demonstrate the perfect tracking property and design trade-offs. The plant in question is given as

$$G(s) = \frac{3(\frac{-s}{2}+1)}{(\frac{s}{5}+1)(\frac{s}{10}+1)}, \tag{6.1}$$

which may be factored into

$$G_{noi}(s) = \left(\frac{-s}{2}+1\right) \qquad G_i(s) = \frac{3}{(\frac{s}{5}+1)(\frac{s}{10}+1)}. \tag{6.2}$$

For the first example, the design parameter is given as

$$P_{\text{des}}(s) = \frac{1}{(\frac{s}{5}+1)^2}, \tag{6.3}$$

which results in the two feedforward controllers

$$FF1(s) = P_{\text{des}}(s)G_{noi}(s) = \frac{(\frac{-s}{2}+1)}{(\frac{s}{5}+1)^2} \tag{6.4}$$

$$FF2(s) = P_{\text{des}}(s)G_i^{-1}(s) = \frac{(\frac{s}{5}+1)(\frac{s}{10}+1)}{3(\frac{s}{5}+1)^2} = \frac{(\frac{s}{10}+1)}{3(\frac{s}{5}+1)}. \tag{6.5}$$

A MATLAB Simulink simulation diagram for DFFPC is shown in Figure 6.1

For the simulations shown here, the feedback controller is set to an arbitrary (stabilizing) controller, and the rest of the continuous-time system blocks (denoted

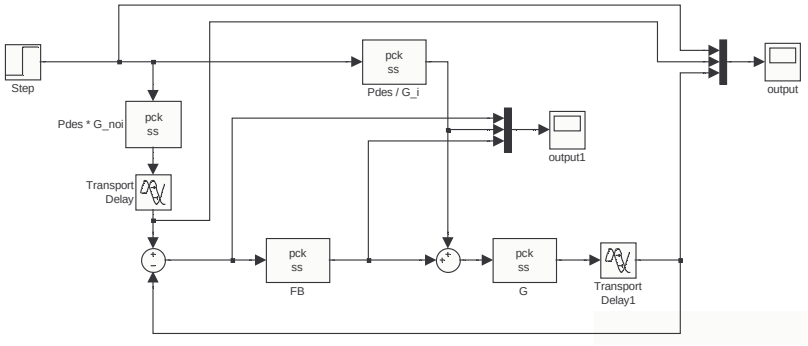

Figure 6.1: Continuous-time LTI DFFPC Simulink Diagram

`pck ss` in Figure 6.1) are defined according to the methods outlined earlier in this chapter. For the first simulation, the input reference ($r(t)$), the filtered reference ($r_{ff}(t)$), and actual output ($y(t)$) are plotted along with the feedback error ($e(t)$) and feedforward control authority ($u_{ff}(t)$). The results are shown in Figure 6.2.

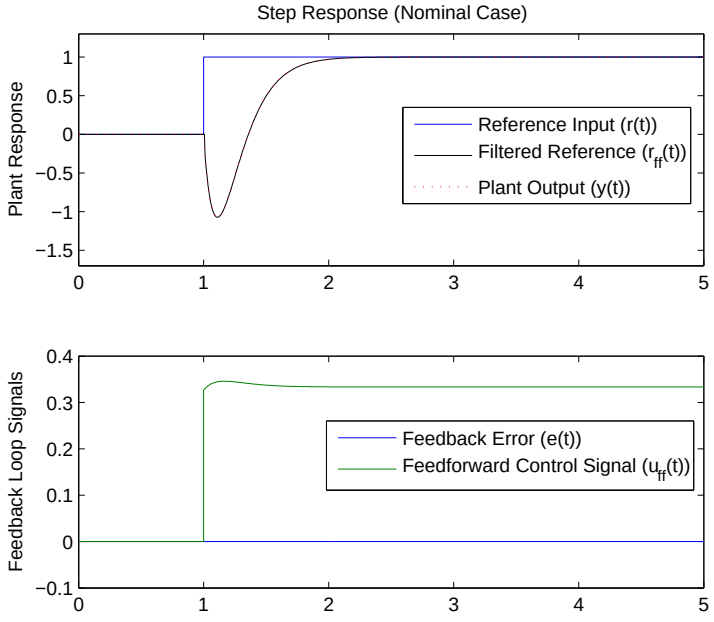

Figure 6.2: Perfect Tracking Control using DFFPC

Here, the filtered reference (solid black line) and actual output (dotted red line) are the same in the top plot of Figure 6.2. As a result, the feedback error is zero and the feedforward controllers are providing perfect reference tracking when there are no

modeling errors or external disturbances. For this choice of $P_{\text{des}}(s)$, the undershoot is about 100%. This provides an example of the fundamental limitation (i.e., the significant undershoot) imposed by the right-half plane zero that is close to the origin that neither feedforward nor feedback controllers can remove.

Since the transfer function $FF2(s) = P_{\text{des}}(s)G_i^{-1}(s)$ is biproper, the feedforward control signal instantaneously jumps from 0 to 0.3 when the step is applied. In many applications, the actuator will not be able to respond to this instantaneous jump, which means the perfect tracking property will not be achieved on the physical system. Therefore, a practical design rule is to choose the relative degree of $P_{\text{des}}(s)$ to be strictly greater than the relative degree of $G_i(s)$. An example of this is provide in the next example where a high frequency pole is added to $P_{\text{des}}(s)$ to make $FF2(s)$ strictly proper. For the next set of simulations, $P_{\text{des}}(s)$ is redefined to be

$$P_{\text{des}}(s) = \frac{1}{(\frac{s}{100}+1)(\tau s+1)^2}, \tag{6.6}$$

and simulations are run for various values of τ. Note that smaller values of τ will result in a larger bandwidth in the feedforward controllers (i.e., the bandwidth is determined by $1/\tau$). Since the perfect tracking property was demonstrated in Figure 6.2, only the actual output is plotted for various values of τ. The results are shown in Figure 6.3.

Figure 6.3 illustrates the fundamental trade-offs that are encountered when designing $P_{\text{des}}(s)$, namely that a faster settling time requires larger control signals and that more undershoot will result. The performance metrics for the various $P_{\text{des}}(s)$ designs are summarized in Table 6.1.1. Here, percent undershoot is given in terms of undershoot per step size and the settling times are measured from when the step occurred, to when the final value was within Δ percent of the step size. For the cases considered here $\Delta = 5\%$ and $\Delta = 2\%$ are given.

For a particular system, $P_{\text{des}}(s)$ should be chosen based on design requirements with the understanding that a faster settling time will incur more overshoot and

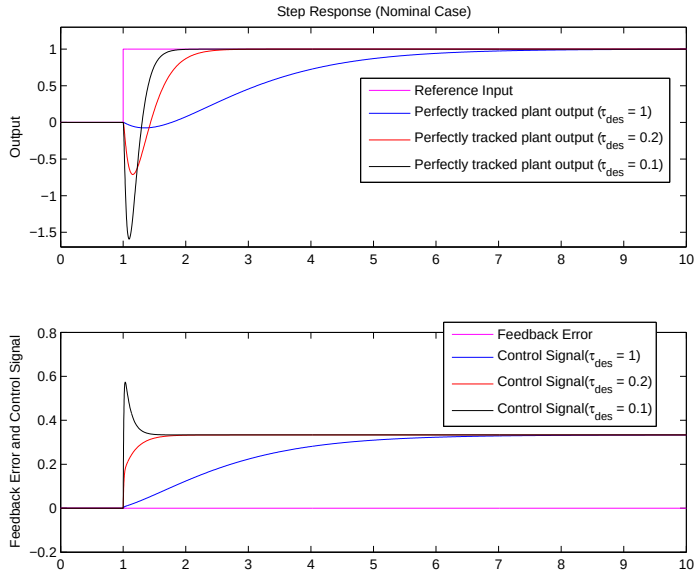

Figure 6.3: Perfect Tracking Control Bandwidth Effects

τ	% undershoot	5% Settle Time	2% Settle Time	max control authority
1	7%	5.18 sec	6.25 sec	0.33
0.2	71%	1.23 sec	1.45 sec	0.33
0.1	160%	0.68 sec	0.79 sec	0.57

Table 6.1: Perfect Tracking Control Trade-offs for a Non-minimum Phase Plant

require a larger control signal. Assuming that $P_{\text{des}}(s)$ satisfies its design constraints, the transfer function $P_{\text{des}}(s)G_{noi}(s)$ gives the class of signals that may be perfectly tracked using the architectures presented here. Therefore, a search may be performed over candidate $P_{\text{des}}(s)$ transfer functions to determine if the performance requirements can be met.

6.2 Robust and Optimal Feedforward Controller Design

In the known (Chapter 3), DFFPC, and DFFSP architectures, a feedforward controller is used to invert the minimum-phase dynamics of a system. In many situations, the term $G_i^{-1}(s)$ is not a proper transfer function. However, the product

$P_{\text{des}}(s)G_i^{-1}(s)$ may be designed such that it is a stable proper transfer function. In this section, robust and optimal controller synthesis is explored for the feedforward controller designs. A method is provided that will result in the robust and optimal design of $P_{\text{des}}(s)$. In many cases, the results show that the robust and optimal controller methods discussed here result in proper transfer functions that tend to essentially invert the stable minimum-phase dynamics in the system. Through the use of standard model reduction techniques, is possible to find a low order $P_{\text{des}}(s)$ without losing much performance.

For the initial presentation of this method, it will be assumed that the plant is stable. This is required since the resulting robust and optimal controllers are required to be internally stabilizing, which is not possible for unstable plants with the formulation considered here. However, the design methodology may be modified to address this, which is provided at the end of this section.

The use of μ-synthesis to design feedforward controllers was first discussed in [68]. In their work, a method was developed for designing robust feedforward controllers for a measurable disturbance rejection problem. An example application was given that showed the controllers improved ability to reject a measurable temperature disturbance on a two tank process. The applicability of using μ-synthesis for feedforward control was studied and conditions were provided to show when feedforward control is beneficial; however, the structure of the final controller was never explored to see if it was trying to essentially invert the plant. Here, we explore the structure of μ-optimal, $\mathcal{H}_2$-optimal, and $\mathcal{H}_\infty$-optimal feedforward controllers.

For the robust and optimal controller synthesis considered here, μ-synthesis is achieved using the D-K iteration method given in [58], and the optimal controllers will be synthesized using the software package in [69], which are based on the methods in [70]. Through demonstration, we are able to offer that the μ-optimal, $\mathcal{H}_2$-optimal, and $\mathcal{H}_\infty$-optimal feedforward controllers essentially invert the stable minimum-phase

dynamics of the plant. For the parts of the plant that are not perfectly inverted by the feedforward design, model reduction techniques may be used to extract a lower order $P_{\text{des}}(s)$.

The basic design process for the robust and optimal controllers is as follows:

1. Design a robust/optimal $FF2(s)$ (see details later).
2. Define $Z(s) = \frac{1}{K_\mu} FF2(s)G_i(s)$ (where K_μ is selected to make $Z(0) = 1$).
3. Use a minimum realization and model reduction techniques to reduce the order of $Z(s)$.
4. Define $P_{\text{des}}(s) = Z(s)F(s)$, where $F(s)$ is a stable and proper transfer function with $F(0) = 1$. NB: $F(s) = 1$ is a valid (and common) choice here.

The robust/optimal design will result in a $FF2(s)$ that is strictly proper and the formulation of $Z(s)$ in step 2 will always guarantee that same the design constraint on $P_{\text{des}}(s)$ is satisfied. In step 4, $Z(s)$ is used to define $P_{\text{des}}(s)$. In many cases, the design of choice will be $P_{\text{des}}(s) = Z(s)$ (i.e., $F(s) = 1$).

The formulation of $Z(s)$ presented here is valid independent of the structure of $FF2(s)$. However, $Z(s)$ in step 2 will, in practice, contain some stable pole/zero cancelations (and possibly some stable pole/zero near misses). These stable cancelations (and near misses) may be eliminated from the final $P_{\text{des}}(s)$ design. This observation is also important for the design of $FF2(s)$ on unstable plants, which is addressed at the end of this section.

The intention of $FF2(s)$ is to invert $G_i(s)$ and not the full plant $G(s)$. For the robust and optimal designs it is desirable to know if the $FF2(s)$ design may done on just $G_i(s)$ or if the full plant $G(s)$ should be used. The use of these models in the $FF2(s)$ design are considered in the next two sections and it is argued that it is preferred to design $FF2(s)$ using only $G_i(s)$. These next two sections also

demonstrate that $FF2(s)$ is, in many cases, almost of the form $P_{\text{des}}(s)G_i^{-1}(s)$, which motivates the use of a reduced order $P_{\text{des}}(s)$ for implementation.

6.2.1 Feedforward 2 Designed on the Full Plant $G(s)$

A $FF2(s)$ μ-synthesis interconnect is shown in 6.4 that contains the full non-minimum phase plant in the design.

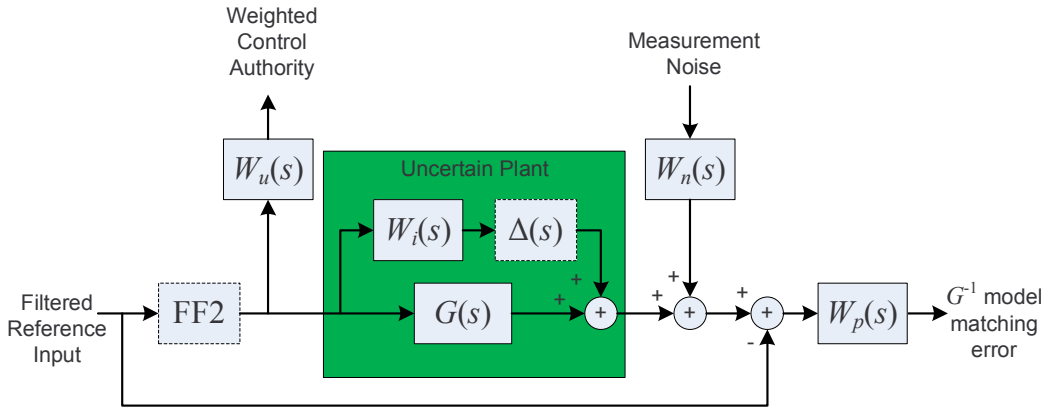

Figure 6.4: $FF2(s)$ Design Interconnect with the Full Plant Model

The μ-synthesis optimization attempts to minimize the gain from the exogenous inputs to exogenous outputs in the presence of model uncertainty. Here, the exogenous inputs are the filtered reference and measurement noise, and the exogenous outputs are the weighted control authority and model matching error.

Recall from Chapter 3 that the invertible part of the plant is defined to be

$$G_i(s) = \frac{K_{\text{DC}}N_{mp}(s)}{D(s)} \Rightarrow G_i^{-1}(s) = \frac{D(s)}{K_{\text{DC}}N_{mp}(s)}. \tag{6.7}$$

The μ-optimal $FF2(s)$ controller resulting from the design interconnect shown in Figure 6.4 may be expressed as

$$FF2_\mu(s) = \frac{K_\mu N_\mu(s)}{D_\mu(s)}. \tag{6.8}$$

In order to ensure that $FF2_\mu(s)$ is of the desired form, it may be expressed as

$$FF2_\mu(s) = \left(\frac{K_\mu K_{DC} N_\mu(s) N_{mp}(s)}{D_\mu(s) D(s)}\right)\left(\frac{D(s)}{K_{\rm DC} N_{mp}(s)}\right) \quad (6.9)$$

$$= V(s) G_i^{-1}(s). \quad (6.10)$$

Using extensive numerical testing, the numerator and denominator polynomials of $V(s)$ tend to have the following properties:

$$N_\mu(s) = N_1(s) D(s) \quad (6.11)$$

$$D_\mu(s) \approx D_1(s) N_{mp}(s) \quad (6.12)$$

In other words, the (stable) poles of $G_i(s)$ are minimum-phase zeros in $FF2_\mu(s)$ and the minimum-phase zeros of $G_i(s)$ are very near stable poles in $FF2_\mu(s)$. The notion of stable poles in $FF2_\mu(s)$ being very close to the minimum-phase zeros of $G_i(s)$ will be demonstrated via example later in this section. Based on these observations, these cancelations may be used to get a reduced order $V(s)$ as

$$V(s) = \frac{K_\mu K_{DC} N_\mu(s) N_{mp}(s)}{D_\mu(s) D(s)} = \frac{K_\mu K_{DC} N_1(s) N_{mp}(s)}{D_\mu(s)} \approx \frac{K_\mu K_{DC} N_1(s)}{D_1(s)} \quad (6.13)$$

From this, $Z(s)$ may be defined as

$$Z(s) = \frac{1}{V(0)} V(s) = \frac{1}{K_\mu K_{DC}} V(s) = \frac{N_1(s)}{D_1(s)}. \quad (6.14)$$

It should be noted that this final design of $Z(s)$ does not rely on the observation made in eqn (6.13); however, this observation (when it is valid) will result in a lower order $Z(s)$ (and hence (potentially) lower order $P_{\rm des}(s)$). Given $Z(s)$, the final design $P_{\rm des}(s)$ may be given by

$$P_{\rm des}(s) = Z(s) F(s) \quad : \quad F(s) \text{ stable and proper with } F(0) = 1, \quad (6.15)$$

where $F(s)$ provides another degree of freedom. For example, $F(s)$ could be used to limit the amount of control authority by creating more high frequency roll off in $P_{\text{des}}(s)$. However, in many situations, $F(s) = 1$ is the best choice and $P_{\text{des}}(s) = Z(s)$.

In order to demonstrate the structure of the controller, the following non-minimum phase plant is considered.

$$G(s) = \frac{12(-\frac{s}{2}+1)(s+1)}{(\frac{s}{5}+1)(\frac{s}{7}+1)(\frac{s}{10}+1)} \tag{6.16}$$

For the $FF2(s)$ μ-synthesis, an input uncertainty weight of $W_i(s) = 1/100$ is chosen, and the performance weight is given by

$$W_p(s) = \frac{K_p\omega_p}{s+\omega_p}, \tag{6.17}$$

which is a low pass filter. Here, K_p is the gain at low frequencies and ω_p is the bandwidth (in radians per second) that performance is sought after in the μ-optimal design. This means that the final controller will attempt to approximate G^{-1} such that $|G^{-1}(j\omega) - FF2(j\omega)| < 1/K_p$ for $\omega < \omega_p$. The design was carried out for both small and large K_p and ω_p. A minimum realization of the product $G_i(s)FF2(2)$ was taken for each in MATLAB. The results are shown in Table 6.2.1.

	$K_p = 1$	$K_p = 10000$
$\omega_p = 1$ rad/sec	$Z(s) = \frac{1}{(\frac{s}{3.21}+1)(\frac{s^2}{1973}+\frac{s}{30.78}+1)}$	$Z(s) = \frac{1}{(\frac{s}{67.13}+1)(\frac{s^2}{161018}+\frac{s}{283.4}+1)}$
$\omega_p = 100$ rad/sec	$Z(s) = \frac{1}{(\frac{s}{3.41}+1)(\frac{s^2}{587829}+\frac{s}{542.1}+1)}$	$Z(s) = \frac{1}{(\frac{s}{147.7}+1)(\frac{s^2}{26276282}+\frac{s}{3625}+1)}$

Table 6.2: Resulting $Z(s)$ Transfer Functions for Various $FF2(s)$ μ-synthesis Designs

Regardless of the design parameters chosen, the μ-optimal controller always placed minimum-phase zeros in $FF2(s)$ wherever there were stable poles in the plant (i.e., $D(s)$ always appeared in the numerator of $FF2(s)$). For the cases where the plant

inverse was approximated out to 1 rad/sec (i.e., $\omega_p = 1$ rad/sec), the μ-optimal controller put a pole at $s = -1.00004387147732$, which is very close to the minimum-phase zero at $s = -1$. This is a (stable) near pole/zero cancelation, which was removed from the final $Z(s)$ shown in Table 6.2.1. This is an example of the comment above where it was claimed that $FF2_\mu(s)$ contained stable poles very near the minimum-phase zeros of $G_i(s)$. It is common in standard model reduction techniques to eliminate the near pole/zero cancelation from $Z(j\omega)$ with little consequence on the frequency response or performance of the overall control system. For the cases where the plant inverse was approximated out to 100 rad/sec (i.e., $\omega_p = 100$ rad/sec), the μ-optimal controller put a pole so close to the minimum-phase zero at $s = -1$ that the minimum realization of $Z(s)$ in MATLAB had a pole/zero cancelation. From observations, this cancelation of the minimum-phase zeros in $G(s)$ by $FF2(s)$ does not always happen. Therefore, the factorization in eqn (6.10) should be performed. If the resulting $V(s)$ (and hence DC corrected $Z(s)$) has a near pole/zero cancelation, then a reduced order model may be used to represent the final $Z(s)$. Regardless of this possible pole/zero cancelation, the μ-optimal controller $FF2(s)$ appears to always cancel the (stable) poles of $G(s)$.

In the case where $F(s) = 1$, the final feedforward controller will be $FF2(s) = Z(s)G_i^{-1}(s)$. In this case, the transfer function $Z(s)$ may be expressed as

$$Z(s) = FF2(s)G_i(s), \tag{6.18}$$

and $Z(j\omega)$ may be used to infer how well $FF2(s)$ approximates $G_i^{-1}(s)$. In particular, the complex frequency response $Z(j\omega)$ will be close to one (i.e., magnitude one and phase of 0° modulo 360°). A Bode plot of the various $Z(j\omega)$ in Table 6.2.1 are shown in Figure 6.5.

Figure 6.5 illustrates the effects of the two parameters. Since the relative degree of the plant $G(s)$ is two, the controller was designed to be strictly proper, which

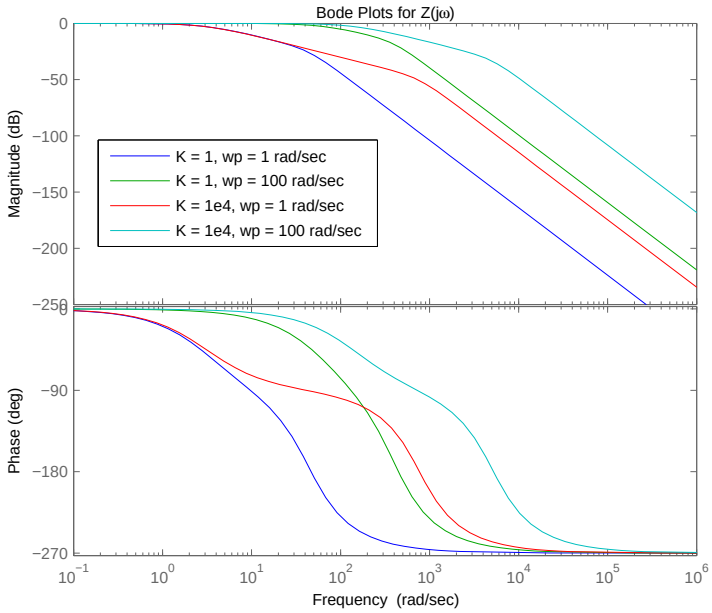

Figure 6.5: Bode Plots of $Z(j\omega)$ from Table 6.2.1.

means that the controller $FF2(s)$ had a relative degree of three. This is seen by the third order roll off (i.e., 60 dB/decade in the magnitude and -270 degrees in phase) for $Z(j\omega)$ at high frequencies. For the two cases that are approximating the inverse out to $\omega_p = 1$ rad/sec (shown in blue and red), the magnitude plots of $Z(j\omega)$ are approximately equal one out to 1 rad/sec and then roll off at higher frequencies. Similarly, the two cases that are approximating the inverse out to $\omega_p = 100$ rad/sec (shown in green and cyan), the magnitude plots of $Z(j\omega)$ are approximately equal one out to 100 rad/sec and then roll off at higher frequencies. For the two cases that have $K_p = 1$, all three poles roll off at about the same time. However, for the cases where $K_p = 10000$, only one pole rolls off around $\omega = \omega_p$, and then, the other two poles roll off at higher frequencies. The result is that these controllers are trying to hold the plant inverse approximation out to higher frequencies. Between these two parameters, ω_p has a larger effect on the effective bandwidth of the approximated plant inverse.

6.2.2 Feedforward 2 Designed on Invertible part of the plant $G_i(s)$

In the known, DFFPC, and DFFSP structures, the first feedforward controller provides the ideal response along with the effects of the non-minimum phase components (if there are any), and the second feedforward controller is designed for the invertible part of the plant, namely $G_i(s)$. In the previous section, it was shown that the μ-optimal controller approximates the inverse of the stable minimum-phase part of the plant out to a bandwidth that is specified by the design parameters. In this section, we explore the option of designing $FF2(s)$ using only the invertible part of the plant, namely $G_i(s)$. Now, the design of $FF2(s)$ is done using the design interconnect shown in Figure 6.6.

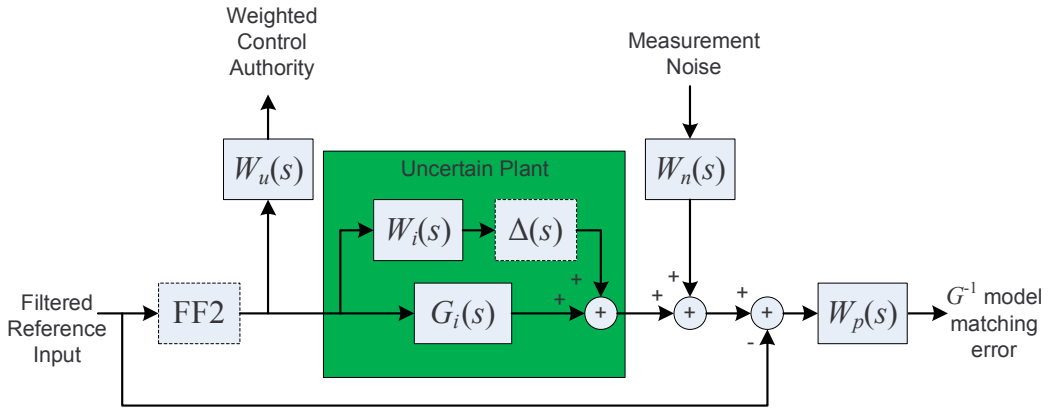

Figure 6.6: $FF2(s)$ Design Interconnect with the Invertible Piece of the Plant Model

The resulting controller will have the same common zeros and poles in $FF2(s)$ and a reduced order $Z(s)$ is obtained. However, the polynomials $N_1(s)$ and $D_1(s)$ that define $Z(s)$ in eqn (6.13) will be different, which means that the approximation of the plant inverse will also be different. In particular, the bandwidth of the approximation will be larger and the phase response will be different at higher frequencies. To see this, consider the same plant from eqn (6.16) with the same design parameters from Table 6.2.1. Now, the plant is defined as

$$G_i(s) = \frac{12(s+1)}{(\frac{s}{5}+1)(\frac{s}{7}+1)(\frac{s}{10}+1)} \tag{6.19}$$

The resulting $Z(s)$ transfer functions from using the design interconnect in Figure 6.6 are shown in Table 6.2.2.

	$K_p = 1$	$K_p = 10000$
$\omega_p = 1$ rad/sec	$Z(s) = \frac{1}{(\frac{s}{17.67}+1)(\frac{s^2}{227.6}+\frac{s}{12.79}+1)}$	$Z(s) = \frac{1}{(\frac{s}{1532}+1)(\frac{s^2}{11000}+\frac{s}{63.8421}+1)}$
$\omega_p = 100$ rad/sec	$Z(s) = \frac{1}{(\frac{s}{109.5}+1)(\frac{s^2}{11910}+\frac{s}{108.8}+1)}$	$Z(s) = \frac{1}{(\frac{s}{1451}+1)(\frac{s^2}{2101000}+\frac{s}{1448}+1)}$

Table 6.3: Resulting $Z(s)$ Transfer Functions for Various $FF2(s)$ μ-synthesis Designs on $G_i(s)$

As in the previous case where the full plant is used, the μ-optimal controller does not quite do a perfect cancelation of the minimum-phase zeros of $G_i(s)$ and the stable poles of $G_i(s)$ are canceled by minimum-phase zeros in $FF2(s)$. For the final controller implementation, the near miss pole/zero cancelations of can be removed without having a major effect on the final $FF2(s)$ controller.

This new design methodology changes the Bode plots of $Z(j\omega)$. To see this consider the plots shown in Figure 6.7

The two main differences for this method are that the approximations go to a larger bandwidth and that all three poles roll off at the same time for each method. This suggests that the $FF2(s)$ controllers designed on the full plant are attempting to alter the phase in order to compensate for the non-minimum phase components in the system. Since the $FF2(s)$ does not need to be designed for the non-minimum phase components, the design interconnect in Figure 6.6 is preferred.

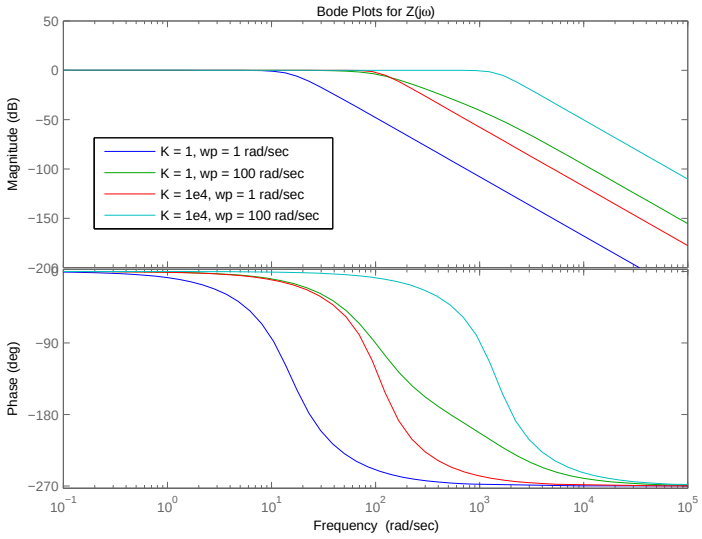

Figure 6.7: Bode Plots of $Z(j\omega)$ From Table 6.2.2.

6.2.3 Feedforward 2 Designed on Plants with Time Delays

For the μ-synthesis tools considered here, the plant must be a rational transfer function. Therefore, time-delays are approximated using Pade approximations [43]. A first order Pade approximation of time delay is given by

$$e^{-s\tau_d} = \frac{e^{\frac{-s\tau_d}{2}}}{e^{\frac{s\tau_d}{2}}} \approx \frac{1 - \frac{\tau_d}{2}s}{1 + \frac{\tau_d}{2}s}, \tag{6.20}$$

which has a non-minimum phase zero at $s = \frac{2}{\tau_d}$ and a stable pole at $s = -\frac{2}{\tau_d}$. Therefore, a μ-optimal design on the plant with the time delay will place a minimum-phase zero at $s = -\frac{2}{\tau_d}$ and the phase will try to compensate for the non-minimum phase zero at $s = \frac{2}{\tau_d}$. For the same reasons given in the previous section, namely the $FF2(s)$ only needs to be handle the stable minimum-phase components of the plant, time delays (and their Pade approximations) are not used to design the $FF2(s)$.

6.2.4 $\mathcal{H}_2$ and $\mathcal{H}_\infty$ Optimal Feedforward Design

Another method for synthesizing the $FF2(s)$ controller is to use standard optimal controller design methodologies, such as $\mathcal{H}_2$ and $\mathcal{H}_\infty$ optimal controller designs. For

these designs a design connect without model uncertainty is used, which is shown in Figure 6.8.

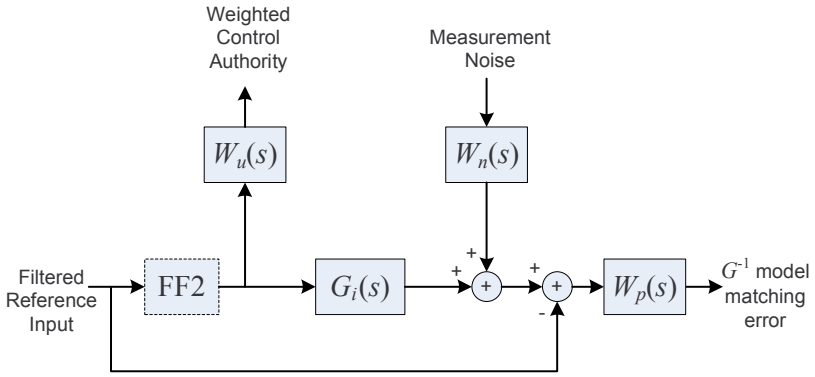

Figure 6.8: Robust $FF2(s)$ Design Interconnect with the Invertible Part of the Plant

As with the μ-optimal feedforward design, the final feedforward controller may be of the form $FF2(s) = Z(s)G_i^{-1}(s)$, where $Z(s)$ has been reduced by removing pole/zero cancelations.

6.2.5 Unstable Plants

In the methods presented in this section, the robust and optimal controllers require that the final controller be internally stabilizing, which does not allow for any unstable pole/zero cancelations. Also, the final controller does not contain feedback, which means that it is unable to move or cancel unstable poles in $G(s)$ (or $G_i(s)$). The result is that the overall feedback connect in Figures 6.4 or 6.6 cannot be stabilized when the plant is unstable.

When the plant is stable, a minimal realization may be made to eliminate the redundant (stable) plant poles and zeros in $Z(s)$. This observation may be used to remove the unstable poles from $G(s)$ (or $G_i(s)$) before the $FF2(s)$ controller is synthesized. In particular, the unstable poles of the plant may be reflected across the $j\omega$-axis. Then, after the controller has been synthesized, $Z(s)$ may be adjusted to remove the effects of this unstable pole reflection.

To begin, let the plant denominator polynomial be given by $D(s) = D_s(s)D_u(s)$, which has been decomposed into its stable and unstable pieces, respectively. Let $J(s)$ be an all-pass filter with

$$J(s) = \frac{D_u(s)}{D_u(-s)}, \tag{6.21}$$

which has non-minimum-phase zeros where $G(s)$ has unstable poles and stable poles that are the reflection of the unstable poles of $G(s)$ across the $j\omega$-axis (e.g., if $G(s)$ has an unstable pole at $s = 2$, $J(s)$ will have a stable pole at $s = -2$ and a non-minimum phase zero at $s = 2$). Now, define

$$\overline{G}_i(s) = G_i(s)J(s) = \frac{K_{\mathrm{DC}}N_{mp}(s)}{D_s(s)D_u(s)}\frac{D_u(s)}{D_u(-s)} = \frac{K_{\mathrm{DC}}N_{mp}(s)}{D_s(s)D_u(-s)}, \tag{6.22}$$

which is now a stable system. In fact, it is $G_i(s)$ with all of its unstable poles reflected across the $j\omega$-axis. Let $\overline{FF2}(s)$ be the robust or optimal feedforward controller that has been designed for $\overline{G}_i(s)$. Using the notation and reasoning used in equations (6.10)-(6.10), assume that the cancelations[1] and scaling (to make $Z(0) = 1$) have taken place. Then, the final $\overline{FF2}(s)$ controller will be of the form

$$\overline{FF2}(s) = \left(\frac{N_1(s)D_u(-s)}{D_1(s)}\right)\left(\frac{D_s(s)}{K_{\mathrm{DC}}N_{mp}(s)}\right) \tag{6.23}$$

Now, let $FF2(s)$ be defined as

[1]This assumption is not required for this design method to work. However, if the cancelations do not occur, the resulting $FF2(s)$ will be a higher order transfer function.

$$
\begin{aligned}
FF2(s) &= \overline{FF2}(s)J(s) \\
&= \left(\frac{N_1(s)D_u(-s)}{D_1(s)}\right)\left(\frac{D_s(s)}{K_{\mathrm{DC}}N_{mp}(s)}\right)\left(\frac{D_u(s)}{D_u(-s)}\right) \\
&= \left(\frac{N_1(s)}{D_1(s)}\right)\left(\frac{D_s(s)D_u(s)}{K_{\mathrm{DC}}N_{mp}(s)}\right) \\
&= Z(s)G_i^{-1}(s),
\end{aligned}
\tag{6.24}
$$

which is in the desired form.

6.3 Conclusions

In this chapter, two methods for designing $P_{\mathrm{des}}(s)$ were explored. In the direct design, $P_{\mathrm{des}}(s)$ is parameterized to provide desired closed-loop properties such as rise time with no overshoot. In the second design method, standard optimal and robust controller synthesis methods were presented that allowed for multiobjective optimization such as fast rise while using limited control authority. In both design scenarios, the robustness tools from the previous chapters may be used to guide the design of $P_{\mathrm{des}}(s)$. This will be illustrated on examples in Chapter 9.

Chapter 7

Adaptation Techniques

The achievable performance for the presented controller architectures depends upon the accuracy of the models used for the feedforward components. In particular, better models allow for better performance. These models either need to be known a priori or they need to be learned via experience with the physical plant. One advantage of the proposed controller architecture is that the feedforward controllers may be adapted without affecting closed-loop stability. This is a different approach from our earlier work that focused on adapting controllers inside the feedback loop [71; 72; 73; 74; 75]. While these earlier methods do guarantee stability for adapting controllers inside a feedback loop, they are computationally intensive. Since the methods presented here only adapt the feedforward controllers, they do not affect closed-loop stability provided the signals that come out of the feedforward controllers are bounded. For the situations considered here, stable adaptation schemes are assumed, which means that closed-loop stability will be unaffected by the adaptation. The advantage to the methods in [71; 72; 73; 74; 75] is that adapting the feedback controller will improve both tracking performance and disturbance rejection performance, whereas adapting the feedforward controllers in the architectures presented here only improves tracking performance. The advantages to the methods presented here are greatly reduced computational overhead required to guarantee stability, and there are no issues with conservative robustness analysis bounds that can severely limit the adaptation. In

this section, two methods for adaptation are considered.

For systems that are well represented by linear models, *model identification adaptive control* (MIAC) schemes will be considered. For these methods, the plant identification will be done using standard system identification techniques. The unique part of this adaptation is the way that the identified plant model is decomposed into G_{noi} and G_i to redefine the pieces of the feedforward controllers. The use of MIAC for DFFPC is very straightforward, since all the model decomposition appears in the feedforward loops. For DFFSP, a prediction of the minimum-phase plant output is provided inside the feedback loop. For ease of stability analysis, the components inside the feedback loop are not adapted in the cases considered here. Instead, a feedforward augmentation is used to correct for the incorrect plant output prediction. This augmentation makes the adaptable feedforward controller for the DFFSP architecture look like the feedforward controller in the DFFPC architecture.

For systems that are nonlinear and time varying (NLTV), or high dimensional, deriving the the inverse dynamics analytically may be impractical. In these cases, better performance may be attained by using feedforward controllers that are NLTV. The only restriction is that the output of the feedforward controllers have bounded energy. While there are many NLTV techniques that may be used (c.f., [76]), we will utilize a *reinforcement learning* (RL) controller here. This controller will use echo state networks to provide the feedforward control signal. In the next chapter, we will discuss the stability properties of echo state networks and provide conditions that may be used to guarantee stability a priori.

For each of these methods, the learning may be done either on-line or in a batch format. While on-line optimization routines are often sought after, they can be sensitive to measurement noise. For sake of the discussion here, both will be considered. However, care should be given to decide which format is best for the application.

7.1 Model Identification Adaptive Control

MIAC comprises a large class of adaptive controllers that use model identification to update a controller based on an identified plant model. A basic diagram of MIAC is shown in Figure 7.1.

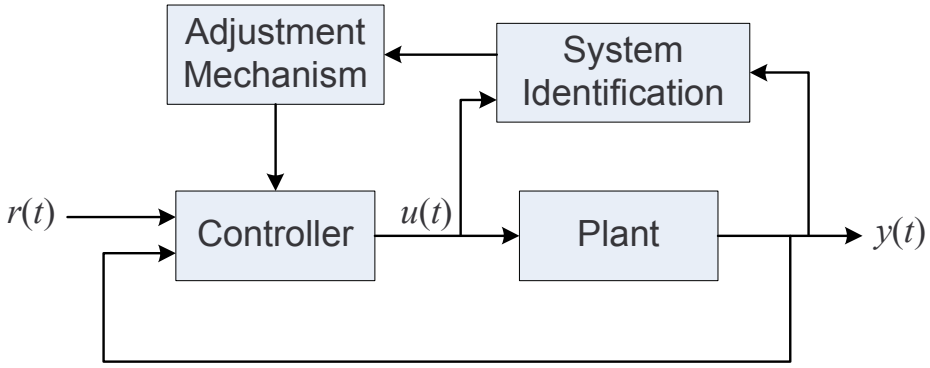

Figure 7.1: General MIAC Setup

In this setup, system identification is done based on the observed input-output behavior of the plant. System identification is a well studied problem and many methods exist for identifying both continuous-time and discrete-time plant models. While this piece of MIAC typically uses standard system identification algorithms, it is the adjustment mechanism that varies between the different MIAC schemes. If the adjustment mechanism updates a part of the controller that uses the current system output, namely $y(t)$, as an input (i.e., the controller is in the feedback path), then the feedback system is being adapted and special care must be taken to ensure closed-loop stability (c.f., [76; 71]. In order to simplify the closed-loop stability analysis here, only the feedforward controllers will be updated.

For the adaptation considered here, discrete-time controller implementations of the DFFPC and DFFSP architectures will be used. Therefore, the system identification part will focus on discrete-time system identification of the zero-order hold equivalent discrete-time plant ($G_{ZOH}(z)$). We will consider both recursive and batch methods. Recursive methods update the plant model at every time step based on a

correction (or innovation) that is determined by the current input into the systems and the error between the modeled and actual outputs. This type of adaptation can continually correct for differences in the model dynamics; however, it is restricted to a fixed model order and time delay. In contrast, batch system identification may be achieved by storing a history of the input-output time series and fitting a model to the data. In this case, a search over the various model orders and time delays may be used to determine the best model fit. In many practical cases, it may be desirable to perform online recursive system identification to track minor changes and then perform intermittent batch identification to correct for larger problems like time delay mismatches and to change the order of the identified model. Regardless of how the model is identified, it will be sent to the adjustment mechanism, which will determine the best method for updating the two feedforward controllers.

The specific structure of the DFFPC and DFFSP controllers allow for the adjustment mechanism to split the identified plant into its minimum-phase and non-minimum phase components and update the two feedforward controllers accordingly. For the DFFSP case, this non-minimum phase component appears in the feedback loop, which will not be adapted. Instead, an additional piece is added to the first feedforward controller that, under certain conditions, will restore the perfect tracking property as it was defined in the previous chapters. In a traditional Smith predictor, the feedback signal is a prediction of the delay free minimum-phase dynamics of the system. If there are mismatches in the dynamics or time delay between the plant and model, the predicted feedback signal will contain both minimum-phase and non-minimum phase dynamics. In this case, a structure that is similar to the DFFPC architecture may be used to exactly predict the feedback signal (i.e., it can restore the perfect tracking property). This will be discussed in more detail later in this chapter.

7.1.1 Model Identification Adaptive Dual Feedforward Predictive Control

One of the advantages to DFFPC is that both the minimum-phase and non-minimum phase parts of the plant appear in the feedforward paths. This means that the identified plant may be factored into its invertible and non-invertible parts and the transfer function coefficients in the two feedforward blocks may be updated directly. This is illustrated in Figure 7.2 where the blue lines show the blocks in the two feedforward paths that are updated.

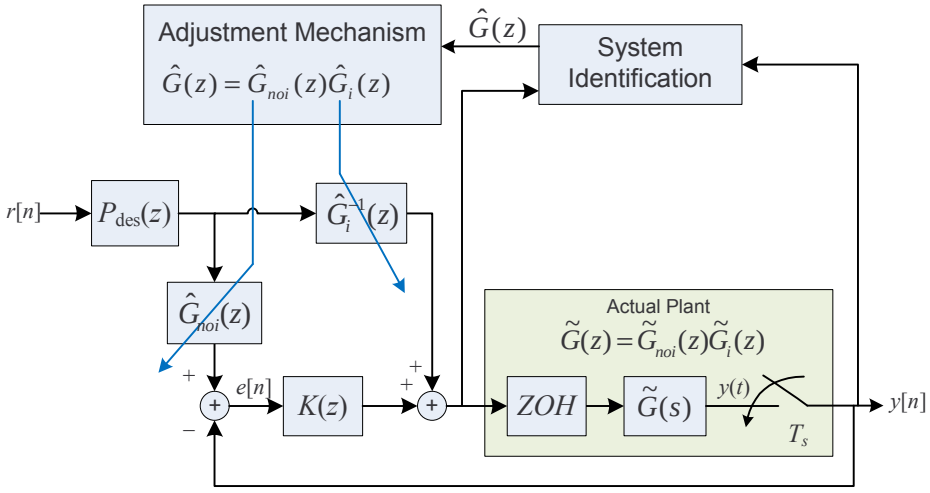

Figure 7.2: Model Identification Adaptive DFFPC

The adaptation scheme has two distinct phases, namely a plant identification and a controller update phase. This is illustrated by the following adaptation algorithm:

- System Identification

  1. Identify the plant
  2. If the order of the plant model has not changed since the last update,
     (a) Send plant model to the adjustment mechanism.
  3. If the order of the plant model has changed,
     (a) Send plant model to the adjustment mechanism.

(b) Send the most current inputs and outputs required to establish the initial conditions.

- Adjustment Mechanism

1. Perform Invertible/Non-invertible factorization on plant model
2. If the order of the plant model has not changed since the last update,
   (a) Update the model parameters in feedforward controllers.
3. If the order of the plant model has changed,
   (a) Replace the controllers with the new controller order.
   (b) Initialize the feedforward controllers using stored initial conditions

7.1.2 Model Identification Adaptive Dual Feedforward Smith Predictor

For the DFFSP, the non-invertible part of the plant appears in the feedback loop. In order to approach the feedforward adaptation for this method, the introduction of a new piece, labeled $X(z)$, is required in the first feedforward path. This is illustrated in Figure 7.3 where the blue lines show the blocks that are updated in the two feedforward paths.

In this Figure, there are three system models, namely $\hat{G}(z)$, $G(z)$, and $\overline{G}(z)$, where $\hat{G}(z)$ and $G(z)$ are chosen to match the notation used in Chapter 5. These systems are as follows:

$$\hat{G}(z) = \hat{G}_{noi}(z)\hat{G}_i(z) \qquad \text{Nominal Plant Model (Fixed)} \tag{7.1}$$

$$G(z) = G_{noi}(z)G_i(z) \qquad \text{Actual Plant} \tag{7.2}$$

$$\overline{G}(z) = \overline{G}_{noi}(z)\overline{G}_i(z) \qquad \text{Estimated Plant Model (Adapted)} \tag{7.3}$$

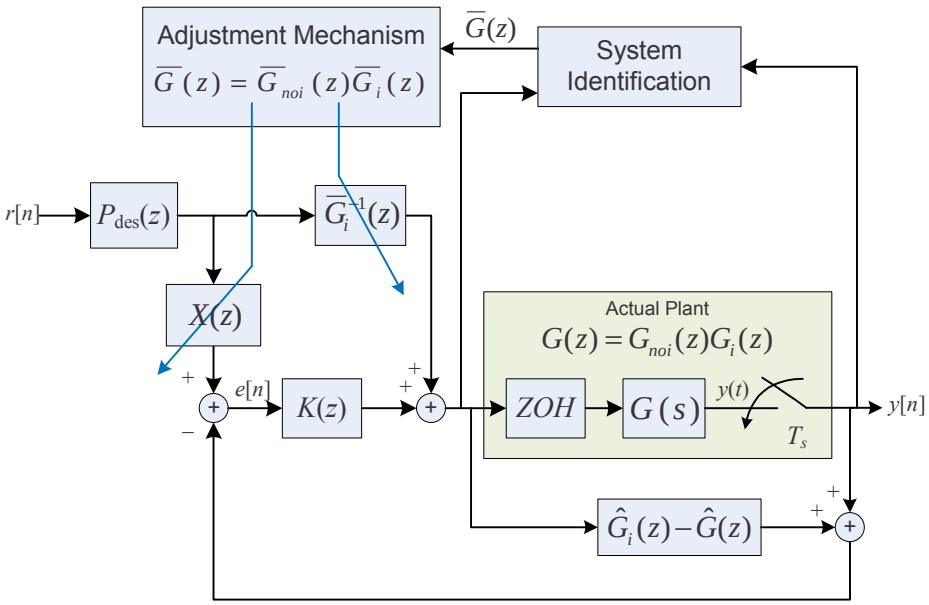

Figure 7.3: Model Identification Adaptive DFFSP

If $\hat{G}(z) \neq G(z)$, the perfect tracking property (i.e., $e[n] = 0$ for all n) will not hold. However, it is possible to pick $X(z)$ that will restore this property if the actual plant can be exactly modeled by $\overline{G}(z)$ (i.e., $\overline{G}(z) = G(z)$). To see this consider the sensitivity function $\overline{S}_{\text{DFFSP}}$ of Figure 7.3 with the estimated plant used in the two feedforward paths. To simplify notation, define the Smith predictor equivalent of the plant in the feedback to be

$$\begin{aligned} \hat{G}_{SP}(z) &= \hat{G}_i(z) + G(z) - \hat{G}(z) \\ &= \hat{G}_i(z) + \hat{E}(z), \end{aligned} \tag{7.4}$$

where $\hat{E}(z) = G(z) - \hat{G}(z)$ is the modeling error between the actual plant and the nominal plant model. When $\hat{E}(z) = 0$, $\hat{G}_{SP}(z) = \hat{G}_i(z)$ and the perfect tracking property for the DFFSP will hold. When $\hat{E}(z) \neq 0$, the sensitivity with the estimated plant is given as

$$\begin{aligned}\overline{S}_{\text{DFFSP}}(z) &= \frac{P_{\text{des}}(z)X(z)}{1+\hat{G}_{SP}K} - \frac{P_{\text{des}}\overline{G}_i^{-1}\hat{G}_{SP}}{1+\hat{G}_{SP}K} \\ &= \frac{P_{\text{des}}(z)(X(z)-\overline{G}_i^{-1}\hat{G}_{SP})}{1+\hat{G}_{SP}K} \end{aligned} \tag{7.5}$$

If we can choose $X(z) = \overline{G}_i^{-1}(z)\hat{G}_{SP}(z)$, we can recover the perfect tracking property. Upon further investigation, $X(z)$ is given as

$$\begin{aligned} X(z) &= \overline{G}_i^{-1}(z)\hat{G}_{SP}(z) \\ &= \overline{G}_i^{-1}(z)(\hat{G}_i(z) + G(z) - \hat{G}(z)), \end{aligned} \tag{7.6}$$

which requires knowledge of the true plant $G(z)$. However, if the true plant can be perfectly modeled (i.e., $\overline{G}(z) = G(z)$), then we can choose

$$\begin{aligned} X(z) &= \overline{G}_i^{-1}(z)(\hat{G}_i(z) + \overline{G}(z) - \hat{G}(z)) \\ &= \overline{G}_{noi}(z) + \overline{G}_i^{-1}(z)(\hat{G}_i(z) - \hat{G}(z)) \end{aligned} \tag{7.7}$$

which is the non-invertible part of the plant (as in the DFFPC architecture) plus the result of the second feedforward controller being applied to the feedback predictor path. Now, the $P_{\text{des}}(z)X(z)$ is providing the filtered reference ($r_{ff}[n]$) that the output ($y[n]$) can perfectly track (in the nominal case with no external disturbances). This makes the adaptive feedforward part of DFFSP structure become the adaptive part of the DFFPC structure, which motivates the preference of the DFFPC structure over the DFFSP structure. For the DFFSP architecture, $X(z)$ is modeling the non-minimum phase dynamics being fed back in the DFFSP architecture, whereas $G_{noi}(z)$ is modeling the non-minimum phase dynamics being fed back in the DFFPC architecture.

In the nominal case where $\hat{G}(z) = G(z)$, $X(z) = 1$ and we recover the original DFFSP structure. The implications to the closed-loop system are that $\overline{M}_{\text{DFFSP}}(z) =$

$P_{\text{des}}(z)\overline{G}_{noi}(z)$ when $\overline{G}(z) = G(z)$, which means that the achievable closed-loop transfer function is of the same form as before. This may be seen by examining the new closed-loop transfer function $\overline{M}_{\text{DFFSP}}(z)$.

$$\begin{aligned}
\overline{M}_{\text{DFFSP}} &= \frac{P_{\text{des}}X(z)G(z)K(z)}{1+\hat{G}_{SP}(z)K(z)} + \frac{P_{\text{des}}(z)\overline{G}_i^{-1}(z)G(z)}{1+\hat{G}_{SP}(z)K(z)} \\
&= \frac{P_{\text{des}}(z)\overline{G}_i^{-1}(z)\hat{G}_{SP}(z)G(z)K(z) + P_{\text{des}}(z)\overline{G}_i^{-1}(z)G(z)}{1+\hat{G}_{SP}(z)K(z)} \\
&= \frac{P_{\text{des}}(z)\overline{G}_i^{-1}(z)G(z)\hat{G}_{SP}(z)K(z) + P_{\text{des}}(z)\overline{G}_i^{-1}(z)G(z)}{1+\hat{G}_{SP}(z)K(z)} \\
&= P_{\text{des}}(z)\overline{G}_i^{-1}(z)G(z)\frac{1+\hat{G}_{SP}(z)K(z)}{1+\hat{G}_{SP}(z)K(z)} \\
&= P_{\text{des}}(z)\overline{G}_i^{-1}(z)G(z) \qquad (7.8) \\
&= P_{\text{des}}(z)\overline{G}_{noi}(z) \qquad \text{if } \overline{G}(z) = G(z)
\end{aligned}$$

In eqn (7.8), $\overline{G}_i^{-1}(z)G(z)$ represents the estimate of the non-minimum phase dynamics of the system. If the plant model is perfect (i.e., if $\overline{G}(z) = G(z)$), this reduces down to the expected result of $P_{\text{des}}(z)\overline{G}_{noi}(z)$ and the perfect tracking property holds. However, if there is a model mismatch, the term $\overline{G}_i^{-1}(z)G(z)$ may contain both minimum and non-minimum phase dynamics and the perfect tracking property will not hold.

For this case, the adaptation scheme for updating the two feedforward controllers is illustrated by the following adaptation algorithm:

- System Identification

    1. Identify the plant
    2. If the order of the plant model has not changed since the last update,
        (a) Send plant model to the adjustment mechanism.
    3. If the order of the plant model has changed,

(a) Send plant model to the adjustment mechanism.

(b) Send the most current inputs and outputs required to establish the initial conditions.

- Adjustment Mechanism

1. Perform Invertible/Non-invertible factorization on plant model
2. Calculate $X(z)$
3. If the order of the plant model has not changed since the last update,
   (a) Update the model parameters in feedforward controllers.
4. If the order of the plant model has changed,
   (a) Replace the controllers with the new controller order.
   (b) Initialize the feedforward controllers using stored initial conditions

These methods will be demonstrated in Chapter 9.

7.2 Reinforcement Learning Control

When the plant being controlled is highly nonlinear or high dimensional, the LTI techniques presented up to this point may not suffice. In these cases, nonlinear control methods such as reinforcement learning control may be appropriate. Reinforcement learning control seeks to find an optimal control policy by repeatedly interacting with a physical system. In the standard formulation of reinforcement learning, an *agent* takes actions in an *environment* in order to maximize some notion of a *reward* [77]. In terms of reinforcement learning as it will be applied as a feedforward controller here, the *agent* will be a controller that maps (filtered) reference inputs to outputs (or actions). For the stability required here, the only restriction is that the agent must have a bounded output. If this condition holds, the agent is not restricted to be

of any particular form. For example, anything from a static function approximator to a NLTV dynamic system may be used; however, given the nonlinear and dynamical requirements of the feedforward controller, a NLTV agent such as an echo state network is suggested. The *environment* that the agent will act upon is the physical plant and the notion of *reward* will be better (smaller) tracking error.

In the ideal (perfect tracking) case, the error $e[n] = r_{ff}[n] - y[n] = 0$ would hold for all discrete-time n. In terms of a reward function, the reinforcement learner develops a policy that finds a mapping from states to actions that minimizes all future absolute values of error, namely $|e[n]|$ for all future n. (NB: In terms of the original problem statement, $-|e[n]|$ could be the reward and the policy could be stated as maximize all future rewards $-|e[n]|$.) In the standard formulation of reinforcement learning, the interaction with the environment is stated as a map between states and actions, where the states define the state of the (dynamic) system. In (finite-dimensional) LTI systems, states are well understood. For example, a second order system has two states. This idea was demonstrated in the state-space representation given in Chapter 2.10, where the states of an LTI system were represented by the vector x. In this representation, there is an implicit mapping from input u to state x and then a mapping from state x to output y. It is this last mapping between state and output that a reinforcement learning controller uses to determine the best control policy. In many applications, the state may not be measured directly, but may be "observed" using the methods in Chapter 2.10. One method for determining the best policy is to use an actor-critic structure, which is discussed next.

7.2.1 Actor-Critic Reinforcement Learning Algorithm

An actor-critic reinforcement learning algorithm may be used to find the optimal control policy. In this setup, a critic is used to converge on the optimal policy from (internal) model states to control actions. The critic is not used directly as a con-

troller, but rather to provide updated information to an actor. The actor is the actual controller (agent) that maps (filtered) reference inputs to control actions. In most practical situations, the number of states is larger than the number of inputs. Therefore, the critic is usually has more inputs and is computationally more complex than the actor. In our previous work using an actor-critic algorithm for reinforcement learning [71; 72; 73; 74; 75], the critic was implemented as a multidimensional lookup table and the actor was implemented as neural network. The critic had many inputs (i.e., reference input, tracking error, feedback control signal, measured output, and measured (or observed) internal plant states), while the actor only had one input (i.e., tracking error). The main part of the reinforcement learning (i.e., determining the policy) is done by the critic. Based on the model learned by the critic, the actor is trained (e.g., via back-propagation) to mimic the behavior learned by the critic. In this setup, the critic learns the optimal behavior from state to output and the actor is trying to mimic the critic by mapping from (filtered) reference to control output. Inherently, this means that the actor needs to internally model the mapping from reference input to its own internal state and then from its own internal state to controller output. The rest of this section focuses on the development of the critic.

For the control objective considered here, the absolute value of the tracking error is used to reinforce (or penalize) the learner. Through experience, the learner is able to find the best action (a_n) based on the current state (o_n) of the system. This is stated more precisely by the *value* function

$$Q_\pi(o_n, a_n) = E_\pi \left\{ \sum_{k=0}^{N} \gamma^k R(o_{n+k}, a_{n+k}) \right\}, \tag{7.9}$$

where π refers to the policy learned, $E\{\cdot\}$ is the standard expected value, γ is a discount factor between 0 and 1 that weights the reinforcement received, and $R(o_n, a_n) = |e_n|$ is the reward for the actions (a_n) taken at discrete-time index n. When γ is closer to zero, the learner solves for immediate rewards, and when γ is

closer to one, the learner solves for delayed rewards. The state-action-reward state-action (SARSA) temporal-difference algorithm [77] provides an online way to calculate the Q function in eqn (7.9). SARSA updates the Q function according to

$$\begin{aligned} \Delta Q_\pi(o_n, a_n) &= \alpha_n \left[R(s_n, a_n) + \gamma Q_\pi(o_{n+1}, a_{n+1}) \right. \\ &\quad \left. - Q_\pi(o_n, a_n) \right], \end{aligned} \tag{7.10}$$

where α_n is the learning rate for the Q-function. Instead of trying to determine the state transition probabilities ahead of time (which is not practical), the SARSA temporal-difference algorithm continuously samples the state transitions by taking actions and keeping track of rewards. In the limit, this method is known to converge to the dynamic programming solution [77]. In this sense, the SARSA temporal-difference algorithm may be viewed as the Monte-Carlo approach to the value iteration algorithm in dynamic programming [77].

The Q function in eqn (7.9) implicitly defines a policy $\pi(o_n)$ that gives the best action for a given state. This is stated formally as

$$\pi(o_n) = \underset{a_n \in A}{\operatorname{argmin}} Q(o_n, a_n), \tag{7.11}$$

where A is the set of allowable actions. Based on the policy learned by the critic, an actor is trained to internally represent the optimal control trajectory for the current state of the system. Examples of this applied to a feedback reinforcement learning controller may be found in [71; 72; 73; 74; 75].

7.2.2 Reinforcement Learning Dual Feedforward Predictive Control

While reinforcement learning could be used to augment or replace any part of the feedforward controller in DFFPC, it is best suited to help improve the plant inverse model. An example diagram is shown in Figure 7.4. Here, $r_{des}[n]$ is the filtered

reference that the block $G_i^{-1}(s)$ is trying to invert. Since the reinforcement learner is trying to improve (and add nonlinear dynamics to) the block $G_i^{-1}(s)$, it gets the same input as $G_i^{-1}(s)$. The performance of this architecture will depend on how well the first feedforward controller is modeling the non-minimum phase components. For example, if there is a time delay in the plant, then modeling this could have a big impact on the achievable performance. This could lead to a hybrid style adaptation, where the model of the LTI non-minimum phase dynamics are adapted using model identification and the plant inversion dynamics are updated using a reinforcement learner.

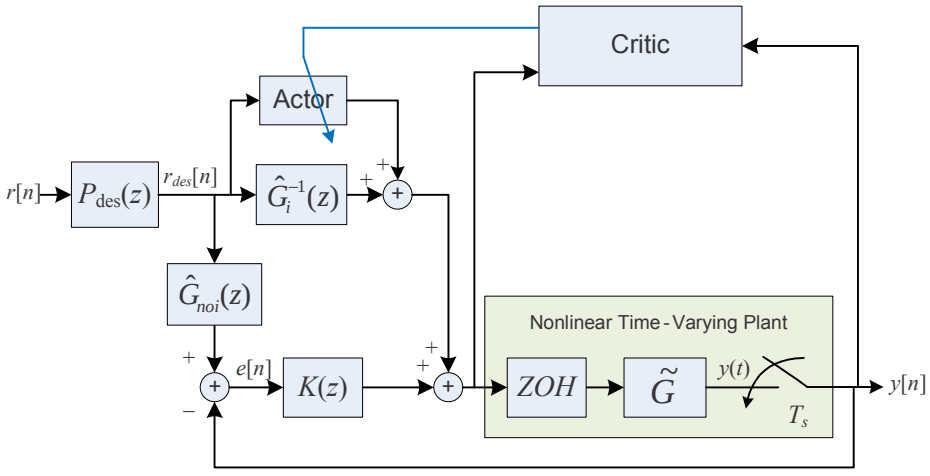

Figure 7.4: Reinforcement Learning DFFPC

7.2.3 Reinforcement Learning Dual Feedforward Smith Predictor

In a similar fashion, a reinforcement learner may be added to improve the block $G_i^{-1}(s)$ in the DFFSP. A block diagram of this is shown in Figure 7.5. As with the reinforcement learner that augments the DFFPC architecture, the performance of the controller will be dependent upon the accuracy of the modeled non-minimum phase components (e.g., the time delay part of the plant). In order to improve the performance, the same augmentation of $X(z)$ from the previous section may added

to the architecture. In this case, a hybrid style adaptation that is similar to the one described for the reinforcement learning DFFPC may be used.

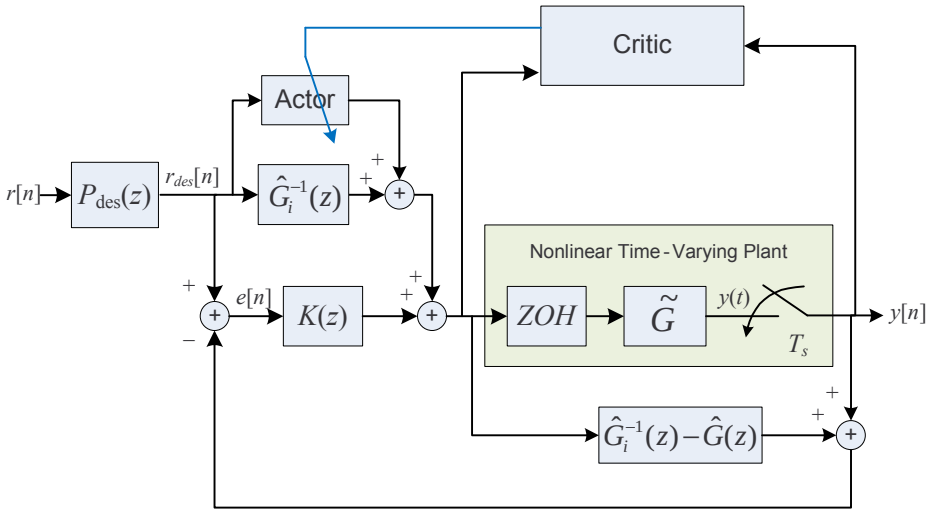

Figure 7.5: Reinforcement Learning DFFSP

7.2.4 Actor Selection

Artificial neural networks are a popular choice for the actor network in an actor-critic setup, because of their ability to model nonlinear dynamics. Two types of neural networks are commonly used for these applications, namely feedforward neural networks (FFNN) and recurrent neural networks (RNN). FFNN are attractive since they are easy to train in a stable manner (e.g. using back propagation), but are limited in the sense that they are only capable of providing a static map between inputs and outputs (i.e. they have no way of internally representing the state of a (nonlinear) dynamic system). In contrast, traditional RNNs may be very difficult (and take a long time) to train stably; however, the recurrent connections of a RNN form a dynamical system. It is this dynamical nature of RNN that allows them to capture the dynamics of a nonlinear system, which makes them more applicable to nonlinear system modeling required in the actor. For a review of traditional RNNs and the problems associated with training them, see either [78; 79; 80].

In more recent years, a RNN called an echo state network has been developed [81]. In this case, there is a fixed recurrent layer that is followed by an adaptive linear layer, which provides a good alternative to traditional RNNs, since the fixed recurrent layer can provide nonlinear dynamics, and the adaptive linear layer can provide fast training algorithms. Echo state networks also have a natural fit to the actor described previously, since they have a mapping from input to state (fixed nonlinear recurrent connections) and a mapping from state to output that may be adapted to a specific plant. Since the mapping from input to state is not adapted, a large "reservoir" of states is used. The idea is that the large number of states will each contain their own features and a combination of features from other states (i.e., the states are not completely independent). Then, the linear combination of these many features may be used to model a wide variety of input-output dynamics, which makes it appropriate for creating an actor. This approach would not work with our early methods presented in [71; 72; 73; 74; 75], because having the larger network inside the closed-loop would greatly increase the computational complexity required to guarantee closed-loop stability. However, in the approach presented here, echo state networks appear in the feedforward path, which means that stability analysis is only performed once when the echo state network is initialized. As it will be shown in the next chapter, guaranteeing the a priori stability of an echo state network (in a feedforward configuration) is readily achieved.

Chapter 8

Echo State Networks

The recent development of echo state networks [81] (ESNs) provides a class of RNNs that alleviate the difficulties of training a recurrently connected network. ESNs are characterized by their ability to uniquely map a temporal input history to an (internal) "echo state" that it can use to map to an output. In the context of reinforcement learning, echo state networks have shown the potential to create their own representation of the internal dynamic state (i.e., echo state) of a system from a history of the input time series. From this internal state representation, an ESN can learn a linear mapping (policy) from states to optimal actions. These concepts were first explored in [82].

From the view of stability, an ESN is said to be stable if it uniquely maps input histories to an echo state. An ESN that has this characteristic is said to have the *echo state property*. Previously, the echo state property was verified via two sufficient conditions, namely one for the existence of echo states for all inputs and one for the non-existence of echo states for certain inputs. In our work [40], these conditions are reformulated and further developed into separate necessary and sufficient conditions for the existence of echo states for all inputs. As mentioned in [81], the original sufficient condition for the existence of echo states appears, in practice, to be rather restrictive. We have addressed this problem by deriving a new sufficient condition that is less conservative. Specifically, a result that is well known in the *Robust Controls*

community (see Section 2.7) is used to reduce the conservatism and in some cases make the bounds tight (i.e., provide a single bound that is both necessary and sufficient). The results of that work are presented here.

8.1 ESN Overview

An ESN is a multilayered network that consists of a feed-forward input layer, a recurrently connected hidden layer, and a feed-forward output layer. This is shown in Figure 8.1.

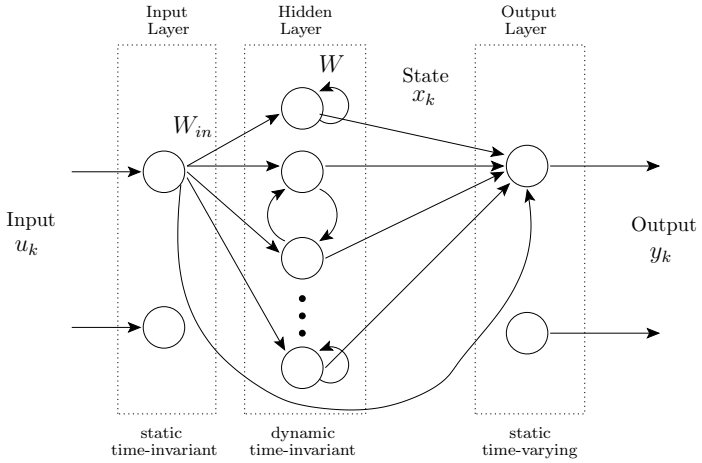

Figure 8.1: Echo State Network Architecture.

The input layer weights and the hidden layer weights are fixed (i.e. time-invariant) and only the output layer weights are trained (i.e. time-varying). This allows for a neural network that has the dynamic modeling capabilities from the (stable) recurrent connections in the hidden layer and the stable adaptive capabilities from the feed-forward network in the output layer. As will be shown, the echo state property holds for ESNs with asymptotically stable recurrent connections.

The input layer (with weights $W_{in} \in \mathbb{R}^{n \times m}$) is used to map the lower dimensional inputs $u_k \in \mathbb{R}^m$ to a larger dimension $W_{in}u_k \in \mathbb{R}^n$ (i.e. $n > m$). In practice, the weights W_{in} do not appear to have an impact on the performance [81]. The echo states x_k are generated from the input layer and the recurrently connected neurons

in the hidden layer. Specifically, for a given an input vector u_k, the transition from one (echo) state vector x_{k-1} to the next (echo) state vector x_k is defined to be

$$x_k = T(x_{k-1}, u_k) = f(W_{in}u_k + Wx_{k-1}), \tag{8.1}$$

where $W \in \mathbb{R}^{n\times n}$ is a square matrix (containing the hidden layer weights) that is applied to the state vector x_{k-1}, and $f(\cdot)$ is a nonlinear "squashing" function that is applied to every element of the vector $W_{in}u_k + Wx_k$. The state vectors x_k are fed into the final output layer. In the general statement of an echo state network, feedback connections from the output layer to the hidden layer are allowed; however, in most applications, these feedback connections are not used. The theory for dealing with these connections will not be explored here.

The hidden layer weight matrix (W) is a randomly generated sparse matrix (i.e. usually only 5% to 20% of the entries are non-zero) that is used to generate a (random) basis for the echo states x_k. The properties of this matrix may be used to determine if an ESN has the echo state property. Let $U^{-\infty}$ and $X^{-\infty}$ be compact sets that represent the set of left infinite sequences of the input and echo state time series, respectively. Then, the definition of an echo state network (i.e. an ESN that has the echo state property) is:

Definition (Jaeger [81]) Assume standard compactness conditions (i.e. the inputs u_k and states x_k come from the compact sets $U^{-\infty}$ and $X^{-\infty}$, respectively). Assume that the network has no output feedback connections. Then, the ESN has echo states (i.e., the echo state property) if every echo state vector x_k is uniquely determined for every left infinite input sequence $u^{-\infty} \in U^{-\infty}$.

This definition implies that nearby echo states must represent similar input histories. In turn, this means that the echo states should depend more heavily on the most recent inputs and states. Intuitively, this means that the state-space is not disjoint,

and the echo states represent the current dynamics, or state, of the system the ESN is modeling. In [81], this property is shown to be satisfied by requiring the echo states to have a certain convergence property. This is stated more formally in the following discussion.

In Jaeger's paper [81], the existence of echo states may be verified in terms of separate necessary and sufficient conditions on the (square) hidden layer weight matrix $W \in \mathbb{R}^{n \times n}$. The necessary condition is $\rho(W) < 1$ and the sufficient condition is $\bar{\sigma}(W) < 1$. Note that in fact, the necessary condition in [81] is actually stated as a sufficient condition for the nonexistence of echo states when $\rho(W) > 1$. The reason for this constraint is that if the underlying linear system is unstable, then the nonlinear system (resulting from the application of the squashing function) will also exhibit instability. From this point of view, the sufficient condition for the nonexistence of echo states is really $\rho(W) \geq 1$. The necessary condition for the existence of echo states ($\rho(W) < 1$) results from this. From a systems point of view, this requires that the nonlinear recurrent system be locally asymptotically stable at the origin. For global asymptotic stability, a more restrictive (sufficient) condition is required[1]. The original proof of the sufficient condition (taken from [81]) is outlined next.

Let x_k and $\tilde{x}_k$ be two distinct state vectors and $y_k = x_k - \tilde{x}_k$. From an equivalent definition, echo states exist if the states x_k and $\tilde{x}_k$ satisfy the convergence property $\|y_k\| \to 0$ as $k \to \infty$ for all right infinite input sequences $u^{+\infty} \in U^{+\infty}$. Note that in this definition, the vector norm is not specified; however, in Jaeger's proof, the standard Euclidean norm (i.e. the 2-norm) is used. Since all finite dimensional norms are equivalent, proving convergence in the 2-norm guarantees convergence in every other (finite dimensional) norm. Jaegers's original sufficient condition is now

[1]For an ESN, bounded-input bounded-out (BIBO) stability is satisfied trivially, since outputs from the squashing functions are bounded for all inputs. Here, the echo state property is defined in terms of a global asymptotic stability requirement on the echo state vectors.

(re)stated as Theorem 12.

Theorem 12 (Jaeger[81])**.** *Let an echo state network have a fixed internal weight matrix* $W \in \mathbb{R}$ *and let* $f(x) = \tanh(x)$*. If* $\overline{\sigma}(W) < 1$*, then the network has the echo state property, i.e.* $\lim_{k\to\infty} \|y_k\|_2 = 0$ *for all right infinite input sequences* $u^{+\infty} \in U^{+\infty}$*.*

Proof.

$$\begin{aligned}
\|y_{k+1}\|_2 &= \|x_{k+1} - \tilde{x}_{k+1}\|_2 \\
&= \|T(x_k, u_k) - T(\tilde{x_k}, u_k)\|_2 \\
&= \|f(W^{\text{in}}u_k + Wx_k) - f(W^{\text{in}}u_k + W\tilde{x}_k)\|_2 \\
&\leq \|(W^{\text{in}}u_k + Wx_k) - (W^{\text{in}}u_k + W\tilde{x}_k)\|_2 \qquad (8.2) \\
&= \|Wx_k - W\tilde{x}_k\|_2 \\
&= \|W(x_k - \tilde{x}_k)\|_2 \\
&\leq \|W\|_2\|y_k\|_2 \\
&= \bar{\sigma}(W)\|y_k\|_2, \qquad (8.3)
\end{aligned}$$

where W satisfies the contraction property $\bar{\sigma}(W) < 1$. Note that this clearly implies the required convergence property, namely $\lim_{k\to\infty} \|y_k\|_2 = 0$. □

Remark Note that eqn (8.2) assumes that the squashing function will shrink *every* element of the vectors $W^{\text{in}}u_k + Wx_k$ and $W^{\text{in}}u_k + W\widetilde{x}_k$ towards zero. Therefore, the difference of the "un-squashed" version will have a larger norm than the difference of the "squashed" version. In the proof given in [81], the squashing function is assumed to be $f(x) = \tanh(x)$; however, it is mentioned that any function satisfying the (element-wise) Lipshitz condition $|f(v) - f(z)| \leq |v - z| \ \forall\ v, z \in \mathbb{R}$ will do. As a side note, if $f(\cdot)$ is differentiable, then the Lipshitz condition is equivalent to $|f'(v)| \leq 1$ $\forall\ v \in \mathbb{R}$.

This proof yields a rather conservative sufficient condition [81]. A less restrictive sufficient condition may be derived by considering a different norm.

8.2 The Weighted Operator Norm

In linear algebra, it is a well known fact that there exists an operator norm (sometimes referred to as an induced norm) for a matrix that is arbitrarily close to the spectral radius of the matrix. This is summarized in Lemma 13.

Lemma 13. *For every matrix $W \in \mathbb{F}^{n\times n}$ and for every $\epsilon > 0$, there exists an operator norm $\|\cdot\|_D$ such that*

$$\rho(W) \leq \|W\|_D \leq \rho(W) + \epsilon \tag{8.4}$$

Proof. See [83]. □

Remark These bounds may be achieved by choosing an appropriate weighted operator norm, namely $\|W\|_D = \|DWD^{-1}\|$ with $D \in \mathbb{F}$ nonsingular, that is specific to the matrix W. This weighted operator norm does not depend on the underlying norm used (e.g. any of the p-norms such as $p = 1, 2$, or ∞), but rather on the weighting matrix $D \in \mathbb{F}$ that is selected based on the matrix W. Note that all finite-dimensional norms are equivalent. Therefore, choosing a different norm may require a different D; however, the property will hold for any chosen norm. For computational reasons, the 2-norm will be used.

8.2.1 The Vector D-Norm

The D-norm of a vector $x \in \mathbb{F}^n$ is defined to be $\|x\|_D = \|Dx\|$, where $D \in \mathbb{F}^{n\times n}$ is non-singular and $\|\cdot\|$ is a vector norm (e.g. one of the p-norms). It is easy to show that $\|\cdot\|_D$ is in fact a vector norm provided D is nonsingular [84]. In this paper, the D-norm will be defined in terms of the weighted 2-norm as $\|x\|_D = \|Dx\|_2$, where D is an arbitrary nonsingular matrix to be chosen later.

8.2.2 The Matrix Operator D-Norm

Let $x = D^{-1}y$, where $y \in \mathbb{F}^n$. Then, the induced D-norm of a matrix $W \in \mathbb{F}^{n\times n}$ is given as:

$$\begin{aligned} \|W\|_D &= \sup_{x\neq 0} \frac{\|Wx\|_D}{\|x\|_D} \\ &= \sup_{x\neq 0} \frac{\|DWx\|_2}{\|Dx\|_2} \\ &= \sup_{y\neq 0} \frac{\|DWD^{-1}y\|_2}{\|y\|_2} \\ &= \bar{\sigma}(DWD^{-1}), \end{aligned} \tag{8.5}$$

where $\bar{\sigma}(DWD^{-1})$ is the largest singular value of the matrix DWD^{-1}. Since D is nonsingular, $\mathcal{N}(D) = \mathcal{N}(D^{-1}) = \{0\}$, so $y = 0$ if and only if $x = 0$. Therefore the constraint $y \neq 0$ is equivalent to $x \neq 0$. The last equality in eqn (8.5) follows from the definition of the induced 2-norm of a matrix.

8.2.3 Minimizing the Matrix Operator D-Norm

Since D is arbitrary, it may be chosen such that $\|W\|_D = \bar{\sigma}(DWD^{-1})$ satisfies Lemma 13 for a given ϵ. If D is allowed to have full structure, then

$$\inf_{D\in\mathcal{D}} \bar{\sigma}(DWD^{-1}) = \rho(W), \tag{8.6}$$

where infimum is used instead of minimum since D (or D^{-1}), in many cases, may be approaching a singular matrix.

If $\mathcal{D}$ is a set of matrices that has some structure imposed upon it, say the set $\overline{\mathcal{D}} = \{\text{diag}(\delta_1, \ldots, \delta_n), \delta_i \in \mathbb{C}$, then $\|W\|_{D_\delta} = \bar{\sigma}(D_\delta W D_\delta^{-1})$ with $D_\delta \in \overline{\mathcal{D}}$ will not necessarily approach the spectral radius of W. Instead, the following relationship holds.

$$\rho(W) \leq \inf_{D_\delta \in \overline{\mathcal{D}}} \bar{\sigma}(D_\delta W D_\delta^{-1}) \leq \bar{\sigma}(W) \tag{8.7}$$

In eqn(8.7), the upper bound is obvious since $D_\delta = I$ is always an option. For a typical W, taking the infimum over all possible $D_\delta \in \overline{\mathcal{D}}$ will result in a measure that is less than $\overline{\sigma}(W)$ and greater than $\rho(W)$. However, there are classes of matrices for which the lower bound of eqn (8.7) is exact. This leads to the following theorem.

Theorem 14. *Let $W \in \mathbb{F}^{n\times n}$ be in one of the following two classes:*

1. *normal matrices, and*

2. *triangular matrices (upper or lower).*

Then, there exists a $D_\delta \in \overline{\mathcal{D}}$ such that $\|W\|_{D_\delta} = \rho(W) + \epsilon$ for all $\epsilon > 0$.

Proof. Class 1: The first class is the easiest to prove since the singular values of a normal matrix are equal to the absolute values of its eigenvalues. Therefore, the maximum singular value and spectral radius are equal.

Class 2: In contrast, the gap between the spectral radius and maximum singular value of a triangular matrix may be arbitrarily large, but the operator D_δ-norm can always be made arbitrarily close to the spectral radius. To see this, let $\delta > 0$, $W \in \mathbb{F}^{n\times n}$ be upper triangular, $D_\delta = \text{diag}(1, \delta, \delta^2, \ldots, \delta^{n-1})$, and $D_\delta^{-1} = \text{diag}(1, \frac{1}{\delta}, \frac{1}{\delta^2}, \ldots, \frac{1}{\delta^{n-1}})$. Then,

$$D_\delta W D_\delta^{-1} = \begin{pmatrix} w_{1,1} & \frac{1}{\delta} w_{1,2} & \cdots & \frac{1}{\delta^{n-1}} w_{1,n} \\ 0 & w_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \frac{1}{\delta} w_{n-1,n} \\ 0 & \cdots & 0 & w_{n,n} \end{pmatrix} \tag{8.8}$$

Using a limiting argument yields

$$\begin{aligned}\lim_{\delta\to\infty}\overline{\sigma}(D_\delta W D_\delta^{-1}) &= \overline{\sigma}\begin{pmatrix} w_{1,1} & 0 & \cdots & 0 \\ 0 & w_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & w_{n,n} \end{pmatrix} \\ &= \max_{1\le k\le n}|w_{k,k}| \\ &= \rho(W), \end{aligned} \tag{8.9}$$

which from Lemma 13 is the smallest that any operator norm may approach. □

Remark For a lower triangular matrix, the same argument would be used with the limit $\delta \to 0$.

Remark In this derivation, D^{-1} is approaching a (rank one) singular matrix, but a finite D may be chosen that is arbitrarily close to the to the infimum as stated in Lemma 13.

Remark Theorem 14 also holds for matrices that may be permuted to triangular matrices by swapping the matching rows and columns (e.g. swapping rows 3 and 5 followed by swapping columns 3 and 5). In the context of an ESN, this amounts to a re-labeling of the recurrently connected neurons.

The results from this section will be used to obtain a tighter (and in some cases exact) bound for the echo state property.

8.3 A New Sufficient Condition for the Echo State Property

In this section, a new sufficient condition is derived using the results from the previous section.

Theorem 15. *Let an echo state network have a fixed internal weight matrix $W \in \mathbb{R}$ and assume the squashing function $f(\cdot)$ satisfies the (element-wise) Lipshitz condition*

$|f(v) - f(z)| \leq |v - z| \ \forall \ v, z \in \mathbb{R}$. *If* $\inf_{D_\delta \in \overline{\mathcal{D}}} \overline{\sigma}(D_\delta W D_\delta^{-1}) < 1$, *then the network has the echo state property, i.e.* $\lim_{k\to\infty} \|y_k\|_{D_\delta} = 0$ *for all right infinite input sequences* $u^{+\infty} \in U^{+\infty}$.

Proof.

$$\begin{aligned}
\|y_{k+1}\|_{D_\delta} &= \|x_{k+1} - \tilde{x}_{k+1}\|_{D_\delta} \\
&= \|T(x_k, u_k) - T(\tilde{x}_k, u_k)\|_{D_\delta} \\
&= \|f(W^{\text{in}} u_k + W x_k) - f(W^{\text{in}} u_k + W \tilde{x}_k)\|_{D_\delta} \\
&\leq \|(W^{\text{in}} u_k + W x_k) - (W^{\text{in}} u_k + W \tilde{x}_k)\|_{D_\delta} \qquad (8.10) \\
&= \|W x_k - W \tilde{x}_k\|_{D_\delta} \\
&= \|W(x_k - \tilde{x}_k)\|_{D_\delta} \\
&\leq \|W\|_{D_\delta} \|y_k\|_{D_\delta} \\
&= \bar{\sigma}(D_\delta W D_\delta^{-1}) \|y_k\|_{D_\delta}. \qquad (8.11)
\end{aligned}$$

where $\inf_{D_\delta \in \overline{\mathcal{D}}} \overline{\sigma}(D_\delta W D_\delta^{-1}) < 1$ satisfies the contracting property. Note that this clearly implies the required convergence property, namely $\lim_{k\to\infty} \|y_k\|_{D_\delta} = 0$. □

Remark Equation (8.10) is the reason for using the set of diagonal scaling matrices $\overline{\mathcal{D}}$, which follows from removing the squashing function. If D_δ was allowed to have full structure (i.e. if $D \in \mathcal{D}$ was used), this inequality will not hold for all state vectors x_k and $\tilde{x}_k$. However, the diagonally structured D_δ ensures that every element of the "squashed" version will be less than the "un-squashed" version and hence the D_δ-norm will be less. Therefore, a sufficient condition for the existence of echo states is $\inf_{D_\delta \in \overline{\mathcal{D}}} \bar{\sigma}(D_\delta W D_\delta^{-1}) < 1$.

Remark Since all finite dimensional norms are equivalent, $\lim_{k\to\infty} \|y_k\|_{D_\delta} = 0$ implies that $\lim_{k\to\infty} \|y_k\|_2 = 0$. Therefore, the original echo state property is satisfied by the new constraint.

Corollary 16. *If* W *is a normal matrix or a (permuted) triangular matrix, then* $\rho(W) < 1$ *is both a necessary and sufficient condition for the existence of echo states for all inputs* u_k.

Proof. The proof of this follows from Theorem 14. □

Remark If $W \in \mathbb{F}$ is triangular, then $\rho(W) < 1$ is identical to $\max|\text{diag}(W)| < 1$.

Remark The stability requirement $\inf_{D_\delta \in \overline{\mathcal{D}}} \bar{\sigma}(D_\delta W D_\delta^{-1}) < 1$ is equivalent to a strong form of Lyapunov stability [49].

Since D must be diagonal, this new bound is not tight in the sense that it is not necessarily equivalent to $\rho(W) < 1$ for any arbitrary W, but it is considerably less conservative than the bound $\bar{\sigma}(W) < 1$.

The reason for using the operator 2-norm (as the underlying norm) is that there exists commercial software for minimizing $\|DWD^{-1}\|_2$ when D must be a structured matrix. This result was presented in Section 2, where it was denoted as $\mu(W)$. Using (for example) MATLAB's *μ-Robust Controls Toolbox*[58], the infimum of $\|W\|_{D_\delta}$ may be calculated using the command

```
[muUB,muLB] = mu(W);
```

Here, `muUB` $= \inf_{D_\delta \in \overline{\mathcal{D}}} \bar{\sigma}(D_\delta W D_\delta^{-1})$. Therefore, if `muUB` < 1, the ESN has the echo state property. Note that if `muUB` ≥ 1, a new ESN may be defined with internal matrix $\tilde{W} = \frac{.99}{\texttt{muUB}} W$, which will satisfy the new sufficient condition, and hence, have the echo state property. In terms of using an ESN for feedforward reinforcement learning control (as in Chapter 7), guaranteeing the echo state property will guarantee stability of the ESN.

Chapter 9

Illustrative Examples

Much of the work to this point has focused on a mathematical framework for providing perfect tracking and analyzing robustness. However, the engineering motivation is to apply these methodologies to real world applications. In particular, functional requirements (e.g., rise time, overshoot, settling time, and maximum control authority) may be directly designed for by choosing the ideal closed-loop transfer function $M(s) = P_{\text{des}}(s)G_{noi}(s)$ appropriately. As a consequence of this method, the maximum achievable performance may be determined for a given structure on $P_{\text{des}}(s)$, which will be demonstrated via the illustrative examples in this chapter.

In the nominal and unperturbed case, the feedforward controllers will provide perfect tracking; however, these feedforward controllers are extracted from plant models and do not respond to external disturbances. Hence, they do not cope with model uncertainty (errors) or signal uncertainty (disturbances). To deal with this, the feedback controller is designed to be robust to model uncertainties and optimal at rejecting (weighted) disturbance signals. Based on the robust analysis tools developed in the previous chapters, robust performance is guaranteed for a given level of model uncertainty. These analysis tools may be used to aid in the design of the various feedforward and feedback controllers. See Chapters 4, 5, and 6 for the design methodologies.

For the case of model uncertainty, adaptation may be used to improve the models in the feedforward controllers, and hence, improve the overall tracking performance

of the closed-loop system. In the case where the model uncertainty is a bounded LTI perturbation, adaptation is able to restore the perfect tracking property. This will be demonstrated via some illustrative examples presented in this chapter, namely:

1. Minimum-phase plant (non-adaptive).
2. Stable non-minimum phase plant with a right half plane zero and time delay.
    - Compare the DFFPC and DFFSP methods
    - Demonstrate adaptation techniques
3. Unstable non-minimum phase plant with a right half plane zero and time delay.
    - Demonstrate perfect tracking on a "difficult" plant
    - DFFPC only (not well suited for DFFSP)

9.1 Strictly Proper Minimum-Phase Plant

In the first illustrative example, we will show how the step response may be arbitrarily shaped by increasing the bandwidth of $P_{\text{des}}(s)$. However, this arbitrarily fast step response requires more control authority, which may not be available on the physical system. Also, increasing the bandwidth of $P_{\text{des}}(s)$ may reduce the achievable robust performance. These issues are addressed by designing feedforward controllers based on actuator constraints and utilizing the developed robustness tools. For this particular case study, the DFFPC and DFFSP architectures reduce down to the same structure.

For a minimum-phase plant, the nominal closed-loop system will perfectly track the filtered reference for any arbitrarily shaped closed-loop response that is defined by P_{des}. For example, a rise time of $\tau_r < \epsilon$ may be achieved (in the nominal case) with no overshoot for every $\epsilon > 0$. However, the resulting control authority (given by $P_{\text{des}} G_i^{-1}$) may exceed the actuator limits, and the closed-loop response may become very fragile

to modeling errors in the plant G. In the latter case, the robustness measures from Chapters 4 and 5 may be used to determine how fragile a system is to perturbations for a particular P_{des}. For the actuator constraints, various design tradeoffs need to be made when designing P_{des}. A specific example of how to make these tradeoffs for a given set of actuator constraints will be considered here. However, it should be noted that designing P_{des} is usually specific to an application and care should be given to make sure the final design meets the design objectives.

The first plant that will be considered is a lightly damped second order system with an oscillatory step response. The general form for a second order system is

$$H(s) = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}, \tag{9.1}$$

where K is the DC gain of the system, ξ is the damping ratio, and ω_n is the natural frequency of the system (c.f., [44] for more information on second order systems). For the system considered here, $K = 2$, $\xi = 0.1$, and $\omega_n = 0.5$ rad/sec, which results in a lightly damped system with an oscillatory open-loop step response. In some contexts, this type of system is referred to as an underdamped system. The resulting transfer function is

$$G(s) = \frac{(2)(0.5)^2}{s^2 + 2(0.1)(0.5)s + (0.5)^2} = \frac{0.5}{s^2 + 0.1s + 0.25}. \tag{9.2}$$

The open-loop step response of this system is shown in Figure 9.1, which shows that the system is very lightly damped, and is difficult to control using classical feedback methods. This is demonstrated next.

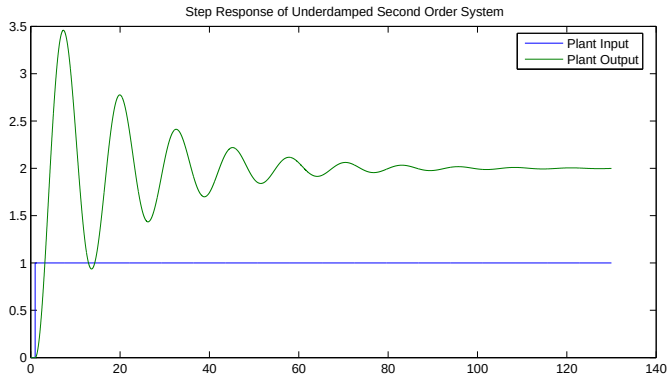

Figure 9.1: Open-loop Step Response

9.1.1 Feedback Designs

In this section both a standard PID (proportional-integral-derivative) controller and a robust controller will be designed for the plant to illustrate the difficulties with controlling the underdamped plant given in eqn (9.2).

9.1.1.1 PID Design

To begin, a standard PID controller is simulated. Ideally, a PID controller may be expressed as:

$$K_{\text{PID(ideal)}}(s) = K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s}, \tag{9.3}$$

which is not a proper system. In order to make this physically realizable in hardware, a high frequency pole is added to the transfer function to make the controller proper. This proper controller may be expressed as

$$K_{\text{PID(proper)}}(s) = \frac{K_D s^2 + K_P s + K_I}{s(\tau s + 1)}, \tag{9.4}$$

which places an additional pole at $s = -\frac{1}{\tau}$. The PID design methodology used here is based on the algorithms provided in [44]. In this methodology, a PI controller is designed to improve the steady state characteristics of the closed-loop transfer function,

and a PD controller (which is also a lead controller with the design methodology used) is designed to improve the transient response of the closed-loop system. The pole at $s = -\frac{1}{\tau}$ is determined by the PD design process. The final PID controller transfer function is formed by multiplying the PI and PD controller transfer functions. The resulting PID controller is

$$K(s) = \frac{13.74s^2 + 3.213s + 0.1602}{s(0.04727s + 1)} = \frac{290.5887(s + 0.07201)(s + 0.1619)}{s(s + 21.16)}, \qquad (9.5)$$

and the resulting closed-loop step response is shown in Figure 9.2.

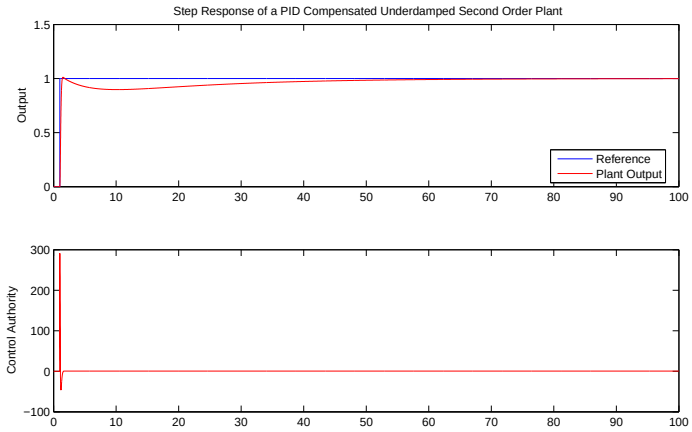

Figure 9.2: PID Compensated Step Response

While this PID controller is able to asymptotically track a step with zero steady state error with relatively low overshoot (1% overshoot for this case), it takes roughly 40 seconds for the output to settle within 2% of the final value. If little overshoot is required, then this is about the best a PID controller can do on this plant.

9.1.1.2 Robust Controller Design

A general robust controller synthesis methodology for disturbance rejection was provided in Chapter 4 (see Figure 4.2), which will be used here. For the robust controller design considered here, it was assumed that the additive model uncertainty was 0.1

at frequencies below 2 rad/sec and 1 at frequencies above 20 rad/sec. Therefore, an additive uncertainty of

$$W_i(s) = \frac{s+2}{s+20} \tag{9.6}$$

was chosen. Based on this uncertainty description, the performance weights were adjusted by hand until the peak μ value of the final controller was approximately one and the amount of overshoot was minimal. The resulting performance weights were used:

$$\begin{aligned} W_d(s) &= \frac{1}{s+15} && (9.7)\\ W_n(s) &= 0.0001 && (9.8)\\ W_p(s) &= \frac{10}{s+0.01} && (9.9)\\ W_u(s) &= \frac{0.0001(s+2)}{s+10} && (9.10)\\ & && (9.11) \end{aligned}$$

The resulting μ-optimal feedback controller was

$$K_\mu(s) = \frac{43.7637(s+20)(s+15.02)(s^2+0.1s+0.25)}{(s+44.42)(s+23.85)(s+0.00995)(s^2+4.397s+7.182)} \tag{9.12}$$

This controller has pole at $s = -0.00995$, which means that it has approximate integral action, but not zero steady state tracking (i.e., it does not have true integral action resulting from a pole at $s = 0$). This a fundamental limitation of the "μ-synthesis via $D-K$ iteration" method used to synthesize the controller. However, this can addressed after the design by approximating the μ-optimal controller with one that has integral action. For the design here, the pole at $s = -0.00995$ results from the choice of the $W_p(s)$ weight. Since this pole is almost at $s = 0$, we can shift

this pole to get integral action. This idea is explained in more depth in [43]. The adjusted controller is given by

$$\hat{K}_\mu(s) = \frac{43.7637(s+20)(s+15.02)(s^2+0.1s+0.25)}{s(s+44.42)(s+23.85)(s^2+4.397s+7.182)} \tag{9.13}$$

This substitution cannot be made without further investigating the effects that this will have on the nominal stability and robust performance. Specifically, we must verify that the closed-loop poles are stable and that the μ plot of the feedback system with the new controller does not degrade too much. Now, the closed-loop transfer function with this controller is given by

$$M(s) = \frac{G(s)\hat{K}_\mu(s)}{1+G(s)\hat{K}_\mu(s)} = \frac{21.8819(s+15.02)(s+20)}{(s+44.43)(s+23.84)(s+2.3)(s^2+2.09s+2.697)}. \tag{9.14}$$

which is stable.

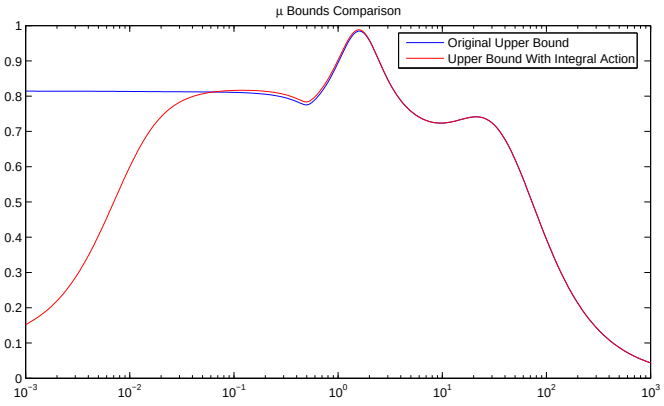

Figure 9.3: μ Plot Comparison

The peak μ value increased from $\mu = 0.9845$ to $\mu = 0.9883$. Also, the μ upper bound is decreased at lower frequencies. This is due to the integral action that will improve low frequency sensitivity to modeling errors (e.g., the controller will provide zero steady state tracking to step inputs). The step response of the two robust controllers is shown in Figure 9.4. As it may be seen, the step responses are almost

identical, except that the robust controller with a pole at $s = 0$ has zero steady state tracking error. The results of these robust controllers show better performance than a PID controller in the sense that the response settles to the final value faster. Also, the peak control authority for the robust controller is about 2, whereas the peak value of the PID controller is about 300, which may not be desirable. However, the achievable tracking for the robust controller (for the desired level of robustness) is still limited and there is some overshoot. These issues will be addressed next through the use of the feedforward controllers.

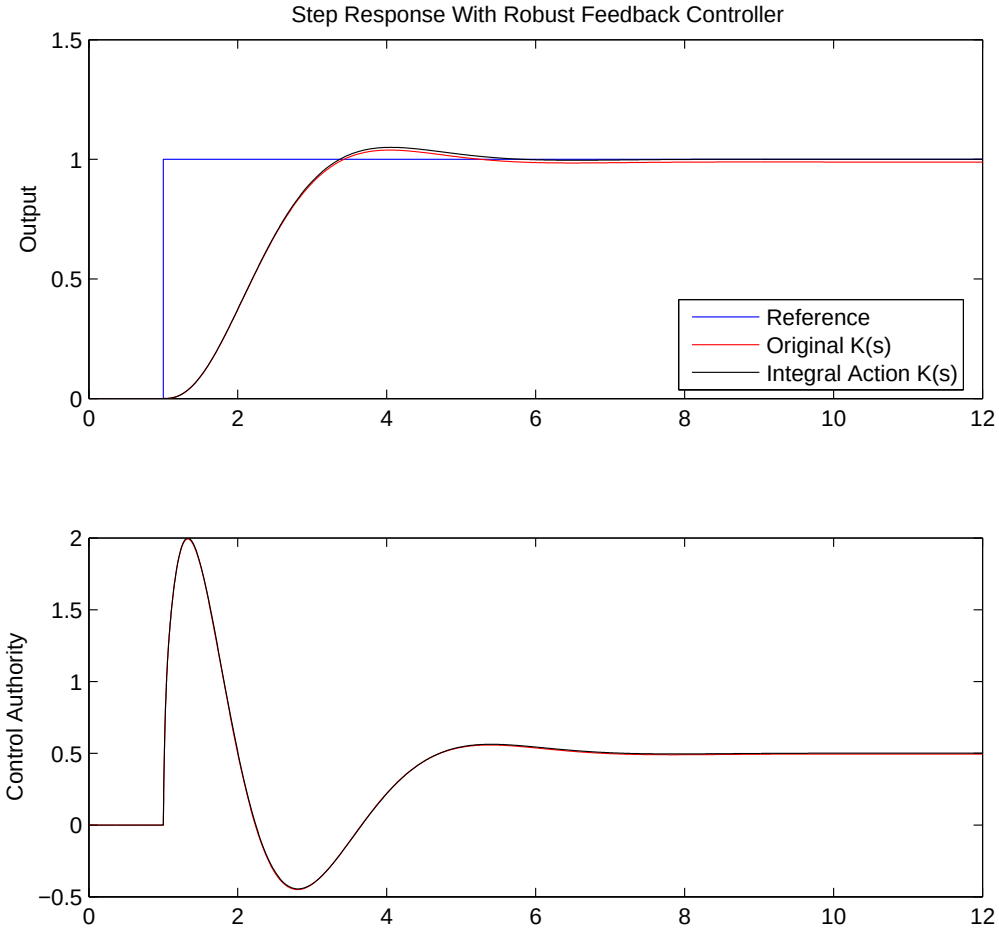

Figure 9.4: Step Response Comparison

9.1.2 Arbitrarily Shaped Nominal Closed-Loop Design

In robust (and optimal) feedback controller synthesis, design trade-offs are made by choosing various (frequency based) weights. While these methods are very effective at shaping the closed-loop frequency response of the system, it is often not clear how to design the closed-loop system to satisfy operation requirements (e.g., specify the rise time and maximum control authority with no overshoot). For the robust feedback controller design given in the previous section, many iterations were performed until the resulting closed-loop step response provided a good rise time with little overshoot.

In this section, we show how the nominal step response of the system can be arbitrarily shaped for a stable minimum-phase system by picking P_{des} appropriately. First, note that for a stable minimum-phase system $G(s) = G_i(s)$, which means that both the DFFPC and the DFFSP collapse down to the same architecture. This is due to the fact that for a stable minimum-phase plant, $G_{noi}(s) = 1$ in the first feedforward path in DFFPC which makes its feedforward portion identical to the DFFSP structure. Also, $G(s) = G_i(s)$ makes $G_i(s) - G(s) = 0$ in the Smith predictor feedback path, which reduces it down to a traditional feedback structure. The result of these observations reduces down to the known architecture shown in Figure 9.5. Also, the robust performance conditions from Chapters 4 and 5 reduce down to the same condition for each type of uncertainty. This is addressed later in this section.

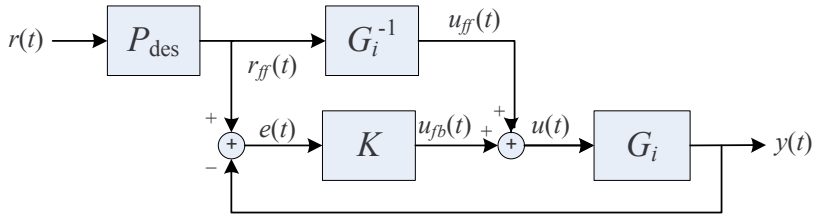

Figure 9.5: Two-stage Feedforward Control for a Stable Minimum-Phase Plant

The plant in eqn (9.2) is a stable minimum-phase plant with a relative degree of two. Therefore, $P_{\text{des}}(s)$ must have a relative degree of two or more so that the

controller $FF2(s) = P_{\text{des}}(s)G_i^{-1}(s)$ is proper. For the design considered here, we will choose

$$P_{\text{des}}(s) = \frac{1}{(\tau s + 1)^2}, \tag{9.15}$$

which defines the nominal closed-loop transfer function of the system (i.e., $M(s) = P_{\text{des}}(s)$), and $FF2(s)$ becomes

$$FF2(s) = P_{\text{des}}(s)G_i^{-1}(s) = \frac{s^2 + 0.1s + 0.25}{0.5(\tau s + 1)^2}. \tag{9.16}$$

For the nominal closed-loop system $P_{\text{des}}(s)$, the unit step response is given as

$$s_1(t) = 1 - \frac{t}{\tau}e^{-\frac{t}{\tau}} - e^{-\frac{t}{\tau}} \tag{9.17}$$

From this, the desired rise time may be determined as a function of τ. For example, the 5% to 95% rise time (i.e., the time from when $s_1(t) = 0.05$ to $s_1(t) = 0.95$) may be set by choosing

$$\tau = \frac{t_{r(\text{des})}}{4.3885}, \tag{9.18}$$

where $t_{r(\text{des})}$ is the desired rise time and the constant 4.3885 comes from solving for the rise time in eqn (9.17). For this example, Maple was used to find a numerical solution for this constant. Using eqn (9.18), τ was determined for a few different rise times. The resulting nominal closed-loop step responses are shown in Figure 9.6.

Figure 9.6 demonstrates the basic tradeoff with this design, which is that a faster rise time requires a larger control signal. In extreme cases, the rise time can be made arbitrarily fast. To demonstrate a this, a rise time of 0.0001 seconds (or 100 microsecond) was chosen. Using eqn (9.18), this resulted in $\tau = 0.000022787$. The plot of this choice of rise time is shown in Figure 9.7.

In this case, a rise time of 0.0001 seconds is achieved, but the required control authority is on the order of 10^9, which is likely not available with the actuator used.

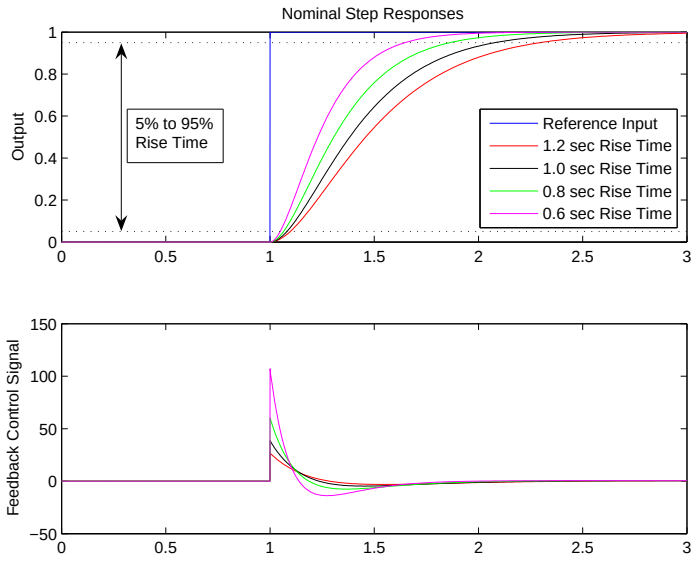

Figure 9.6: Nominal Closed-Loop Step Responses for Specified Rise Times

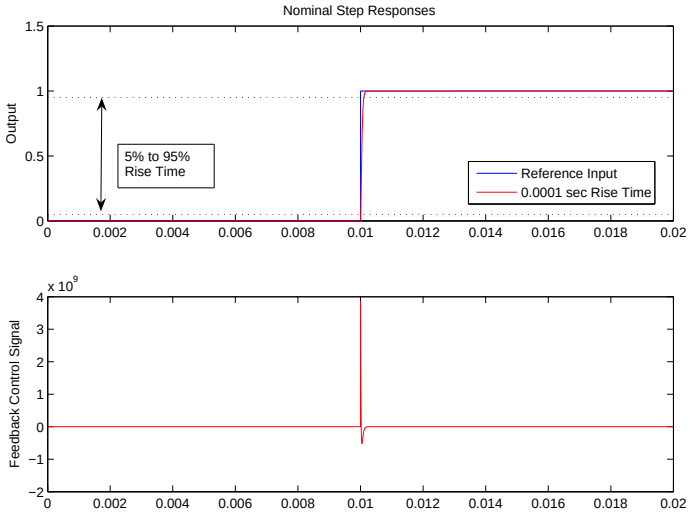

Figure 9.7: Nominal Closed-Loop Step Responses for a Fast Rise Time

If the design objective is to get the fastest rise time with no overshoot, while not exceeding a specified amount of control authority, then a search may be made on $t_{r(\text{des})}$ (or τ) to ensure that all of the design objectives are met.

9.1.3 Design for Robustness

One of the effects of making $FF2(s)$ biproper is that there is an instantaneous spike in the control signal, which may not be practical for most available actuators. By making $FF2(s)$ strictly proper, this instantaneous change in the control signal can be avoided. Also, a strictly proper $FF2(s)$ will generally result in a control signal with a smaller maximum amplitude, which may help keep the control signal inside the operating range of the actuator. The tradeoff is that more bandwidth is usually required in $P_{\text{des}}(s)$ to achieve the same rise time. For the example presented here, $P_{\text{des}}(s)$ is redefined to be the following.

$$P_{\text{des}}(s) = \frac{1}{(\frac{s}{\alpha_{\text{des}}} + 1)^3}, \tag{9.19}$$

where α_{des} is (roughly equal to) the bandwidth of $P_{\text{des}}(s)$. For the case considered here, the criteria for robust performance (with additive uncertainty) is

$$\| |W_1(s)P_{\text{des}}(s)G_i^{-1}(s)W_2(s)S(s)| + |W_2(s)K(s)S(s)| \|_\infty < 1$$

where $W_1(s)$ is the performance weight and $W_2(s)$ is the uncertainty weight[1]. In the example presented here, two different designs (i.e., values of α_{des}) will be examined. In the first case, the bandwidth will be $\alpha_{\text{des}} = 5$ rad/sec, and in the second case, the bandwidth will be $\alpha_{\text{des}} = 10$ rad/sec. For the robust performance check considered here, the following performance and uncertainty weights will be used.

$$W_1(s) = \frac{0.8333s + 1}{2.143s} \tag{9.20}$$

$$W_2(s) = 0.1. \tag{9.21}$$

[1]Since $G(s) = G_i(s)$, the condition for robust performance is equivalent for both the DFFPC and the DFFSP architectures. This equivalence between the architectures will hold for all of the robust performance criteria in the work presented here.

Here, the performance weight $W_1(s)$ is based on the weights used to design the feedback controller. Since the final feedback controller was designed to have integral action, this weight was selected to have a pole at $s = 0$, which is the equivalent of requiring integral action in the performance specification. The (additive) uncertainty weight $W_2(s) = 0.1$ (at all frequencies) was also chosen based on the design uncertainty used in the feedback controller design. For the fictitious numerical example presented here, the selection of these weights is arbitrary; however, in a real application, these weights are influenced by the design requirements.

For the specified weights, $P_{\text{des}}(s)$ with a bandwidth of $\alpha_{\text{des}} = 5$ will satisfy the robust performance criteria, while $P_{\text{des}}(s)$ with a bandwidth of $\alpha_{\text{des}} = 10$ will not satisfy the robust performance criteria. This is illustrated in Figures 9.8 and 9.9.

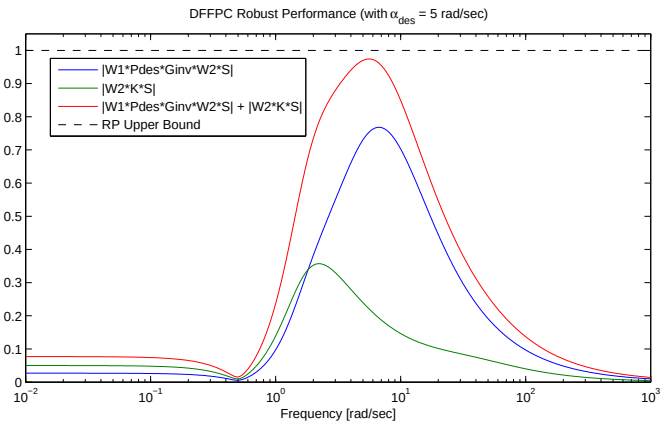

Figure 9.8: DFFPC Robust Performance Check with $\alpha_{\text{des}} = 5$

In order to demonstrate the robust performance of the two $P_{\text{des}}(s)$ designs, a perturbation was generated (using MATLAB). The specific perturbation used here is

$$\Delta(s) = \frac{0.053146(s + 0.9723)}{(s + 0.5173)} \tag{9.22}$$

It should be noted that $\|\Delta\|_\infty = 0.1$, which is at the bound defined by $W_2(s)$. Using this perturbation, the perturbed plant becomes

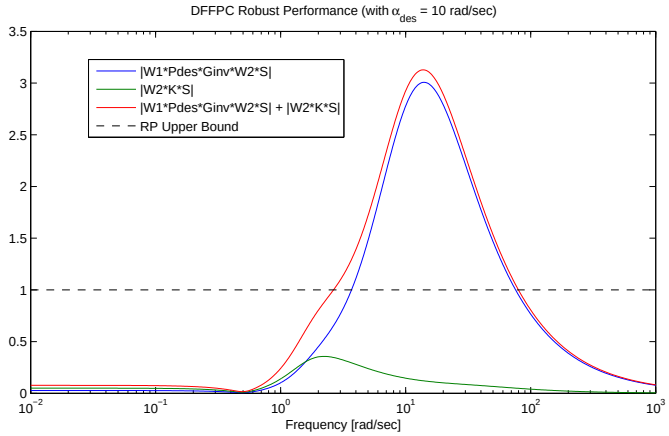

Figure 9.9: DFFPC Robust Performance Check with $\alpha_{\text{des}} = 10$

$$\tilde{G}(s) = G(s) + \Delta(s) = \frac{0.053146(s + 0.5397)(s^2 + 0.5326s + 9.468)}{(s + 0.5173)(s^2 + 0.1s + 0.25)} \tag{9.23}$$

For the simulations shown next, the same perturbed plant (given in eqn (9.23)) is used. The perturbed step response for $\alpha_{\text{des}} = 5$ is shown in Figure 9.10.

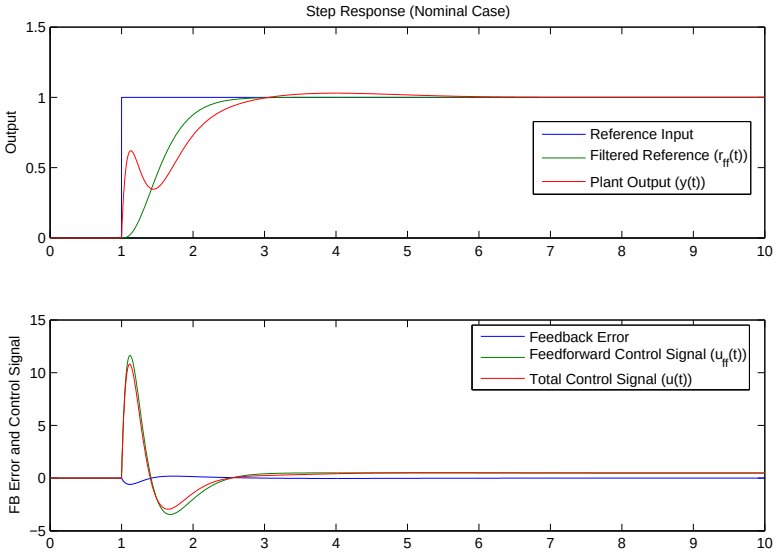

Figure 9.10: DFFPC Perturbed Step Response with $\alpha_{\text{des}} = 5$ rad/sec

In Figure 9.10, the actual output (in the top graph) is no longer following the filtered reference signal perfectly, since this is not the nominal case. The amount of

time it takes the output to reach 0.98 (i.e., 98% of the step size) went from about 1.3 seconds (in the nominal case) to about 1.65 seconds (in the perturbed case), which is an increase of 0.35 seconds in the perturbed case. In the nominal case, there is no overshoot, and in the perturbed case, the overshoot is about 3%. Since this (size of) perturbation and choice of $P_{\text{des}}(s)$ satisfy the robust performance condition used here, this perturbed step response still meets the performance objective defined by $W_1(s)$ in eqn (9.20).

In the second example, $\alpha_{\text{des}} = 10$ is used. It was shown in Figure 9.9 that this design does not meet the robust performance criteria. In fact, the maximum value of the robust criteria is about three times the required limit. Also, the term that contains $P_{\text{des}}(s)$ dominates the robust performance criteria (i.e., the term $|W_1(s)P_{\text{des}}(s)G_i^{-1}(s)W_2(s)S(s)|$ is the main contributing factor to the robust performance criteria not being satisfied in Figure 9.9). The step response for this design is shown in Figure 9.11.

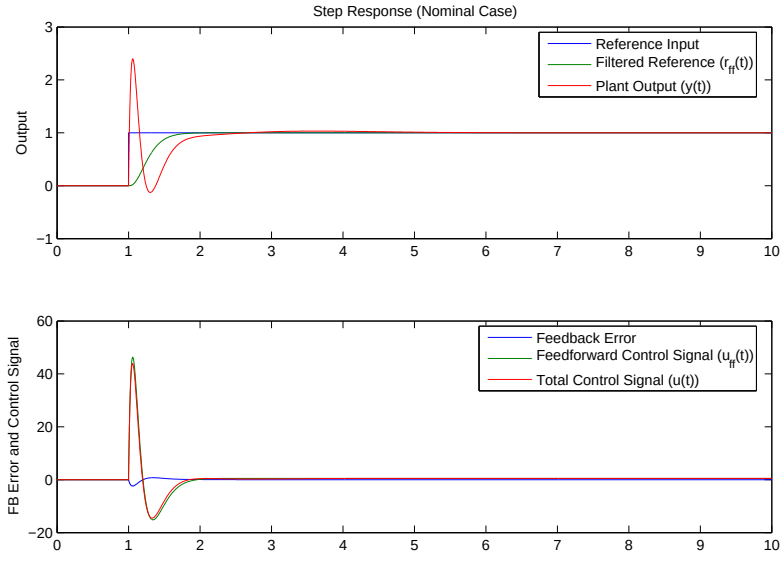

Figure 9.11: DFFPC Perturbed Step Response with $\alpha_{\text{des}} = 10$ rad/sec

In Figure 9.11, performance degradation is more pronounced than the perturbed

step response shown in Figure 9.10. For example, after returning from an initial overshoot of about 140%, it takes the controller about 1.4 seconds to reach 0.98. In the nominal case, this would take about 0.75 seconds, which means that it takes almost twice as long for the perturbed output to settle (when compared to the nominal case). By examining the bottom graph of Figure 9.11, it may be seen that the majority of the control signal is coming from the feedforward controller and very little is coming from the feedback controller. These examples illustrate the trade-offs associated with designing $P_{\text{des}}(s)$. Here, a faster rise time with no overshoot will make the overall control system more sensitive to modeling errors. This phenomenon is accurately predicted by the robust performance criteria derived in the work presented here.

9.2 Stable Non-minimum Phase Plant with a RHP Zero and Time Delay

For the second illustrative example, a non-minimum phase system is considered using both the DFFPC and DFFSP architectures. The non-minimum phase components of this system can severely limit the tracking performance of a feedback only controller. By addressing the non-minimum phase components in the feedforward controller, better tracking tracking performance may be achieved for a given level of robustness.

As with the previous example, the design process consists of synthesizing a feedback controller first and then designing the feedforward controller. The feedback controller may be synthesized using any method desired provided that it internally stabilizes the closed-loop system, satisfies the disturbance rejection requirements, and can adequately correct for modeling errors. Then, the feedforward controller is designed to satisfy tracking requirements. In the nominal case, tracking requirements are satisfied by designing $P_{\text{des}}(s)G_{noi}(s)$ appropriately. When the controller is designed for robust performance, the bounds given in Chapters 4 and 5 may be used. While these designs are done sequentially (i.e., feedback first, then feedforward), they

are not entirely independent when designing for robust performance. Specifically, the robust performance criteria are based on the $S(s)$, $M(s)$, $K(s)$, and $P_{\text{des}}(s)$. The first three terms are determined based on the feedback design $K(s)$, and the last term is determined by the feedforward design. Therefore, the overall performance that may be achieved is determined by both the feedback and feedforward designs. To illustrate this, two designs will be presented. In the first, an oscillatory feedback controller is used along with a direct feedforward design (i.e., $P_{\text{des}}(s)$ is parameterized). In this case, the overall closed-loop system is sensitive to errors in the modeled time delay and the DFFSP architecture performs better than the DFFPC architecture. In a second example, an almost critically damped feedback controller design is used along with robust and optimal feedforward design. The result is that the DFFPC and DFFSP architectures have an almost identical performance and that both are fairly robust.

One of the limitations of all model based designs is that they require accurate models in order to be effective. In order to overcome this limitation, adaptation will be used to improve the models used in the feedforward controllers. In the example that will be presented in this section, a discrete-time implementation of the DFFPC will be used.

9.2.1 Plant Definition

Perfect tracking of a nominal non-minimum phase plant with both a right-half plane zero and time delay is demonstrated in this section. To begin, we define the nominal non-minimum phase plant with a 0.7 second time-delay as

$$\begin{aligned} G(s) &= 25\frac{-s+6}{(s+5)(s+10)}e^{-0.7s} \\ &= 3\frac{\frac{-s}{6}+1}{(\frac{s}{5}+1)(\frac{s}{10}+1)}e^{-0.7s}. \end{aligned} \tag{9.24}$$

For this plant, $K_{\text{DC}} = 3$, $N_{nmp}(s) = (\frac{-s}{6}+1)$, $N_{mp}(s) = 1$, $D(s) = (\frac{s}{5}+1)(\frac{s}{10}+1)$,

and the time delay is given as $e^{-0.7s}$. The invertible/non-invertible decomposition is then

$$G_{noi}(s) = (\frac{-s}{6} + 1)e^{-0.7s}, \quad G_i^{-1}(s) = \frac{1}{3}(\frac{s}{5} + 1)(\frac{s}{10} + 1). \tag{9.25}$$

These components will be used to define the feedforward controllers designed later in this section.

9.2.2 Design #1

For both this design and the next, a robust feedback controller ($K(s)$) design is cast as a weighted disturbance rejection problem with multiplicative uncertainty that is synthesized using a μ-synthesis algorithm [58]. The specific weights (and hence controllers) in the two case are different, but the design methodology is the same. For the simulations that follow, the same $K(s)$ is used for both the DFFPC and DFFSP architectures. This is done to make a fair comparison between the two different structures.

For the feedback design, the robust controller interconnect for disturbance rejection provided in Figure 4.2 will be used here. For the robust controller design considered here, it is assumed that uncertainty is 20% at low frequencies and 200% at high frequencies. Therefore, the following multiplicative uncertainty weight is chosen.

$$W_i(s) = \frac{2(s+5)}{(s+50)} \tag{9.26}$$

For this first example, time delay uncertainty was not factored into the model uncertainty weight selection, but will be addressed in Design #2 in the next section. For Design #1, a set of weights is chosen that results in an oscillatory response with overshoot. This design will be contrasted with the feedback in Design #2 that will produce a faster response with almost no overshoot. For Design #1, the performance weights are:

$$W_d(s) = \frac{0.5}{(s+1)} \tag{9.27}$$

$$W_n(s) = 0.001 \tag{9.28}$$

$$W_p(s) = \frac{0.5}{s+0.001} \tag{9.29}$$

$$W_u(s) = 10 \tag{9.30}$$

$$\tag{9.31}$$

Using the same method from the previous section, the final controller was designed to have integral action. The resulting feedback controller is given by the following.

$$K(s) = \frac{41.0834(s+50)(s+10)(s+5)(s+1.011)}{s(s+1.17)(s+1488)(s^2+23.74s+170.5)} \tag{9.32}$$

A plot of the nominal step response with this controller is shown in Figure 9.12. This figure shows the oscillatory response of the feedback system. This feedback controller design will impact the robust performance that may be achieved by the addition of the feedforward controllers, which is demonstrated next.

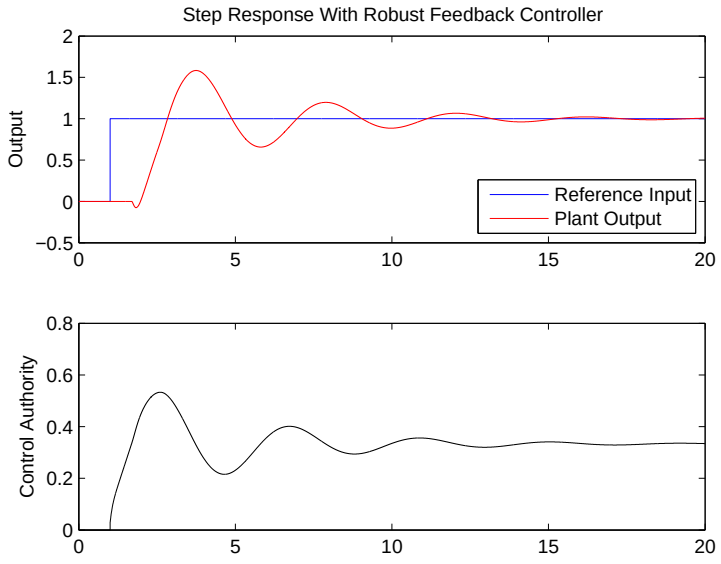

Figure 9.12: Design #1 Robust Feedback Controller Step Response

Based on the feedback controller design, the performance and multiplicative uncertainty weights for the DFFPC and DFFSP are defined to be

$$W_1(s) = \frac{2(s+1.4)^3}{s(s+0.3)(s+10)} \text{ and } W_2(s) = \frac{2(s+5)}{(s+50)}, \tag{9.33}$$

which will be used to guide the feedforward design process. For the example provided here, the feedforward design consists of parameterizing $P_{\text{des}}(s)$ and searching over parameters. For the nominal design, both the DFFPC and DFFSP will provide perfect tracking. Therefore, the resulting nominal designs may be used with either architecture. In order to guarantee that the two feedforward controllers are proper, $P_{\text{des}}(s)$ must have a relative degree of two or more. For the example here, we chose

$$P_{\text{des}}(s) = \frac{1}{(\frac{s}{\alpha_{\text{des}}}+1)^2}, \tag{9.34}$$

which satisfies the relative degree constraint and the design constraint that $P_{\text{des}}(0) = 1$. With this choice of $P_{\text{des}}(s)$, $FF2(s)$ is given by

$$FF2(s) = P_{\text{des}}(s)G_i^{-1}(s) = \frac{1}{3}\frac{(\frac{s}{5}+1)(\frac{s}{10}+1)}{(\frac{s}{\alpha_{\text{des}}}+1)^2}, \tag{9.35}$$

and the nominal closed-loop transfer function is given by

$$M_{\text{DFFPC}}(s) = M_{\text{DFFSP}}(s) = P_{\text{des}}(s)G_{noi}(s) = \frac{\frac{-s}{6}+1}{(\frac{s}{\alpha_{\text{des}}}+1)^2}e^{-0.7s}. \tag{9.36}$$

The right-half plane zero in this transfer function means that the step response will have an undershoot that is related to the bandwidth of $P_{\text{des}}(s)$, which is α_{des} rad/sec. In particular, a faster response will require a larger bandwidth, which will result in more undershoot. This phenomenon is described in [54].

Simulation results for various α_{des} are shown in Figure 9.13. In the top graph, the plant output ($y(t)$) is plotted for various α_{des}. For these cases, the plant model is perfect (nominal case) and $y(t) = r_{ff}(t)$ in the DFFPC architecture (or $y_p(t) = r_{ff}(t)$

in the DFFSP architecture). This is echoed again in the bottom graph where the feedback error is zero for each case.

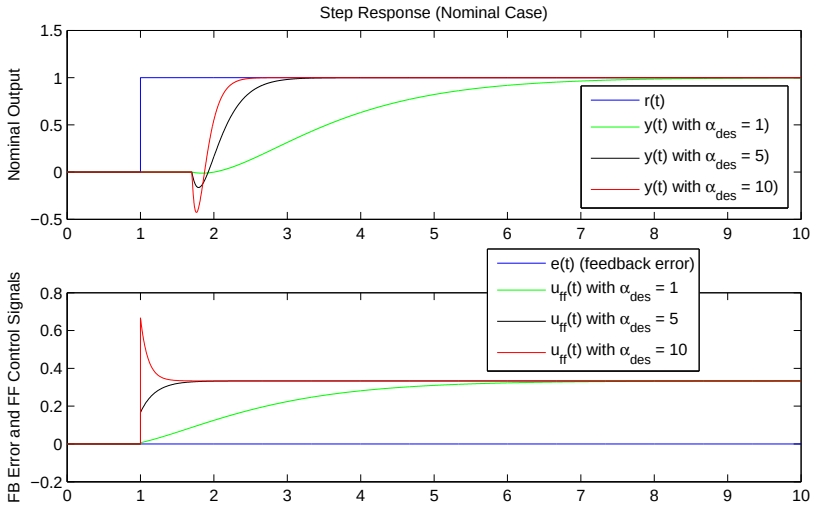

Figure 9.13: Nominal Plant Simulations with Various Choices of α_{des}

These simulations illustrate the design trade-offs for a system with a RHP zero. For a faster response (i.e., more bandwidth or a larger α_{des}) there will be more undershoot and it will take more control authority. For the case with $\alpha_{\text{des}} = 10$ rad/sec, the control signal spikes up quickly and then settles in to the final value. Depending on the actuator being used, this may be undesirable. In the opposite extreme, if the bandwidth is too small (e.g., $\alpha_{\text{des}} = 1$ rad/sec in Figure 9.13), the control signal will not increase so quickly, but the response may be sluggish. These design trade-offs are shown in Figure 9.13. For a good trade-off between control authority and fast response time, the design parameter $\alpha_{\text{des}} = 5$ rad/sec may be used. These types of trade-offs may be used to design the feedforward controllers for a specific plant.

For the rest of the example presented here, the bandwidth of P_{des} is set to 8 rad/sec (i.e., $\alpha_{\text{des}} = 8$), which is chosen based on the additive uncertainty robust performance criteria developed in Chapters 4 and 5 (see eqns (5.29) and (4.19)). The resulting additive uncertainty robust performance plots (versus frequency) are shown in Figure 9.14.

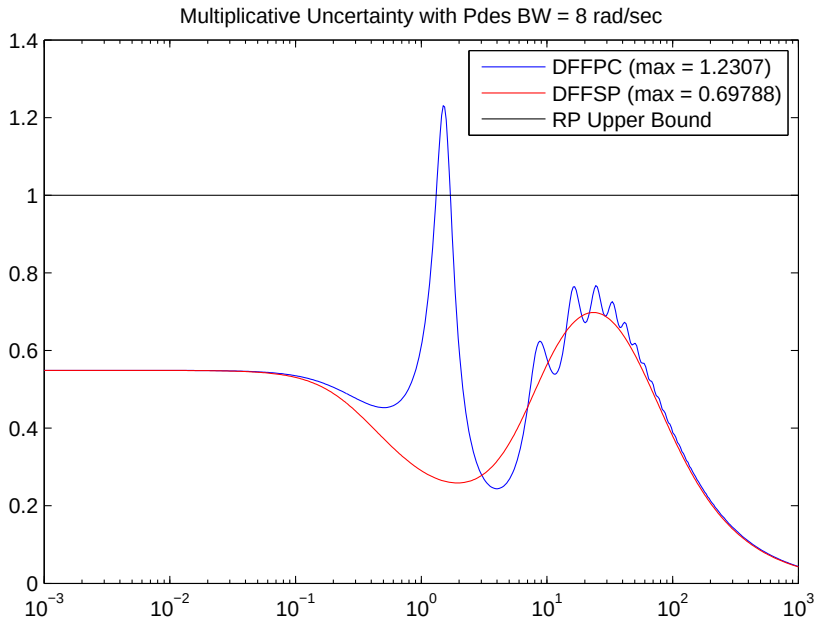

Figure 9.14: Multiplicative Uncertainty Robust Performance Criteria vs. Frequency

The robust performance conditions may be interpreted as: "If the plots stay below the upper bound of one, then the performance criteria will be met for the level of uncertainty specified". For the plots in Figure 9.14, the DFFSP architecture does satisfy the robust performance condition. In fact, it exceeds the performance level. However, the DFFPC architecture misses the robust performance condition. As a result, it is expected that the DFFSP structure will be more robust to model mismatches.

Recall that the nominal plant had a time delay of 0.7 seconds. For the perturbed simulations, the actual plant will have a 0.8 second delay, but the model used for controller design will have a delay of 0.7 seconds. The results of this time delay mismatch are shown in Figure 9.15.

From the simulation in Figure 9.15, it may be seen that there is a small mismatch between the ideal closed-loop response (shown in green) and the actual plant output (shown in red). The top graph shows the DFFPC output, which is oscillatory and has not settled by the end of the simulation. The bottom graph shows the DFFSP

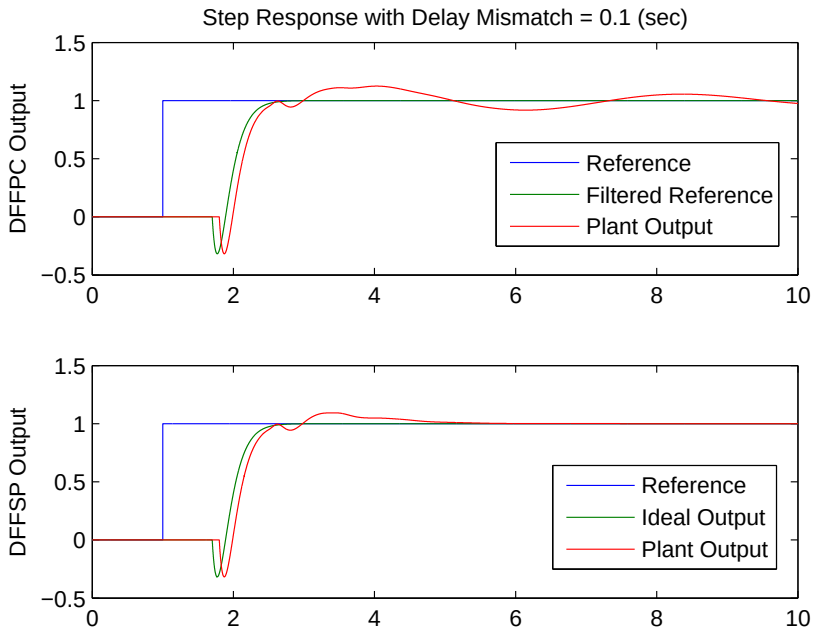

Figure 9.15: Design #1 Perturbed Step with an Actual Process Delay of 0.8 Seconds

output, which settles within four seconds from when the step occurred. As expected for this example, the DFFSP architecture is able to provide better performance when there is a modeling error, which was predicted by the robust performance conditions given in eqns 5.29 and 4.19.

9.2.3 Design #2

For this design, a different robust feedback controller ($K(s)$) is synthesized that produces a faster rise time with less overshoot than Design #1. As with the previous example, the same $K(s)$ is used for both the DFFPC and DFFSP architectures. This is done to make a fair comparison between the two different structures. For the design presented here, $P_{\text{des}}(s)$ will be designed using the robust and optimal techniques discussed in Chapter 6.

For the feedback design, the robust controller interconnect for disturbance rejection provided in Figure 4.2 will be used here. For the robust controller design considered here, it is assumed that uncertainty is 20% at low frequencies and that

the time delay uncertainty is 0.1 seconds. Therefore, the following multiplicative uncertainty weight is chosen.

$$W_i(s) = \frac{2.2(s+0.8)}{(s+8)} \tag{9.37}$$

Unlike the model uncertainty weight in Design #1, this model uncertainty weight "covers" the uncertainty of a time delay (up to 0.1 seconds). Given this new uncertainty description, a set of performance weights is chosen to provide a nominal step response that is almost critically damped, which resulted in the following weights:

$$\begin{aligned} W_d(s) &= \frac{0.5}{(s+1)} && (9.38)\\ W_n(s) &= 0.0025 && (9.39)\\ W_p(s) &= \frac{0.5}{s+0.001} && (9.40)\\ W_u(s) &= 2 && (9.41)\\ & && (9.42) \end{aligned}$$

The changes between this design and the previous feedback design are that the control authority penalty weight has been reduced and the noise weight has been increased. For this specific example, the decrease in control authority penalty allows for more control authority (i.e., larger signals with more bandwidth), and the increase in noise weight penalty decreased the aggressiveness of the feedback controller. The latter parameter was increased until the nominal output had (almost) no overshoot. It should be noted that these weight modifications worked well for this particular problem. In general, tuning the weights is specific to the plant model.

Using the same method from the previous section, the final controller was designed to have integral action. The resulting feedback controller is given by the following.

$$K(s) = \frac{0.13089(s+8)(s+10)(s+5)(s+2.857)(s+1.061)}{s(s+10.61)(s+6.804)(s+1.058)(s^2+4.579s+14.65)} \tag{9.43}$$

A plot of the nominal step response with this controller is shown in Figure 9.16. This figure shows that the nominal step response has both a faster settle time and no overshoot when compared to Design #1. This design is also more robust than Design #1, which will allow for a more aggressive feedforward design (i.e., $P_{\text{des}}(s)$ may be designed to have a larger bandwidth and the overall controller will still satisfy the robust performance criteria).

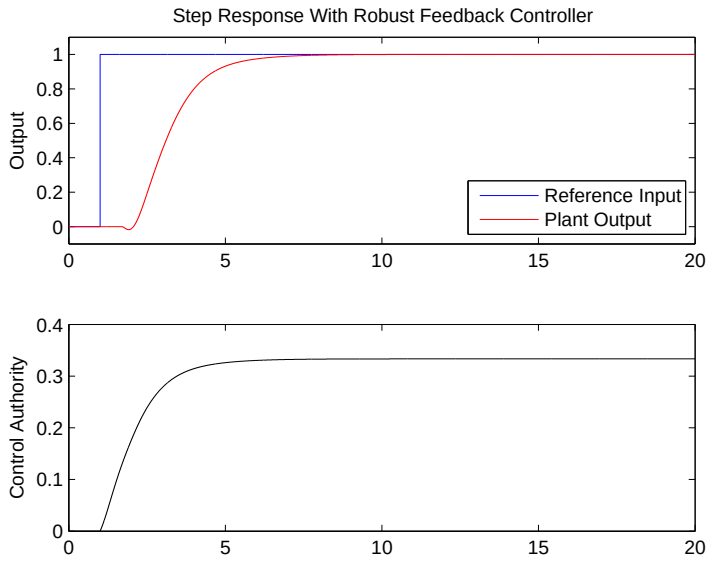

Figure 9.16: Design #2 Step Robust Feedback Controller Response

Based on the feedback controller design, the performance and multiplicative uncertainty weights for the DFFPC and DFFSP are defined to be

$$W_1(s) = \frac{(s+2)^3}{s(s+5)^2} \text{ and } W_2(s) = \frac{2.2(s+0.8)}{(s+8)}, \tag{9.44}$$

which will be used to guide the feedforward design process. For the example provided here, the feedforward design consists of designing μ-optimal, $\mathcal{H}_2$, and $\mathcal{H}_\infty$ robust and optimal $FF2(s)$ controllers. Based on the weight selection given above and the DFFPC and DFFSP robust performance criteria, one of the feedforward designs will be used to determine $P_{\text{des}}(s)$. The design will use the methods outlined in Chap-

ter 6.2.2. For this design, the design weights were chosen to be the following.

$$W_i(s) = 0.5 \tag{9.45}$$

$$W_u(s) = 10 \tag{9.46}$$

$$W_n(s) = 1 \tag{9.47}$$

$$W_p(s) = \frac{10}{s+10} \tag{9.48}$$

$$\tag{9.49}$$

For this design, the bandwidth of $W_p(s)$ was set to 10 rad/sec. The bandwidth of this affects the speed of the final $P_{\text{des}}(s)$. The control authority penalty weight was increased to the point where the nominal step responses had minimal overshoot. For the robust controller, the uncertainty weight was increased to further dampen the step response. The nominal step responses and associated robust performance criteria are shown in Figure 9.17.

For the design in Figure 9.17, the μ-optimal controller provides the fastest response, while still satisfying the robust performance condition. Therefore, it will be the design used going forward. The resulting feedforward controller is

$$FF2(s) = \frac{4.6074(s+5)(s+10)}{(s+11.78)(s^2+14.4s+58.68)} \tag{9.50}$$

Recall that $G_i(s) = \frac{3}{(\frac{s}{5}+1)(\frac{s}{10}+1)}$. Therefore,

$$P_{\text{des}}(s) = FF2(s)G_i(s) = \frac{691.1135}{(s+11.78)(s^2+14.4s+58.68)}. \tag{9.51}$$

It should be noted that this satisfies the design constraints, namely a relative degree of at least two and $P_{\text{des}}(0) = 1$. For comparison between design #1 and design #2, the same plant perturbation is used (i.e., a time delay mismatch of 0.1 seconds). The results are shown in Figure 9.18. It may be seen that there is a small mismatch between the ideal closed-loop response (shown in green) and the actual plant output

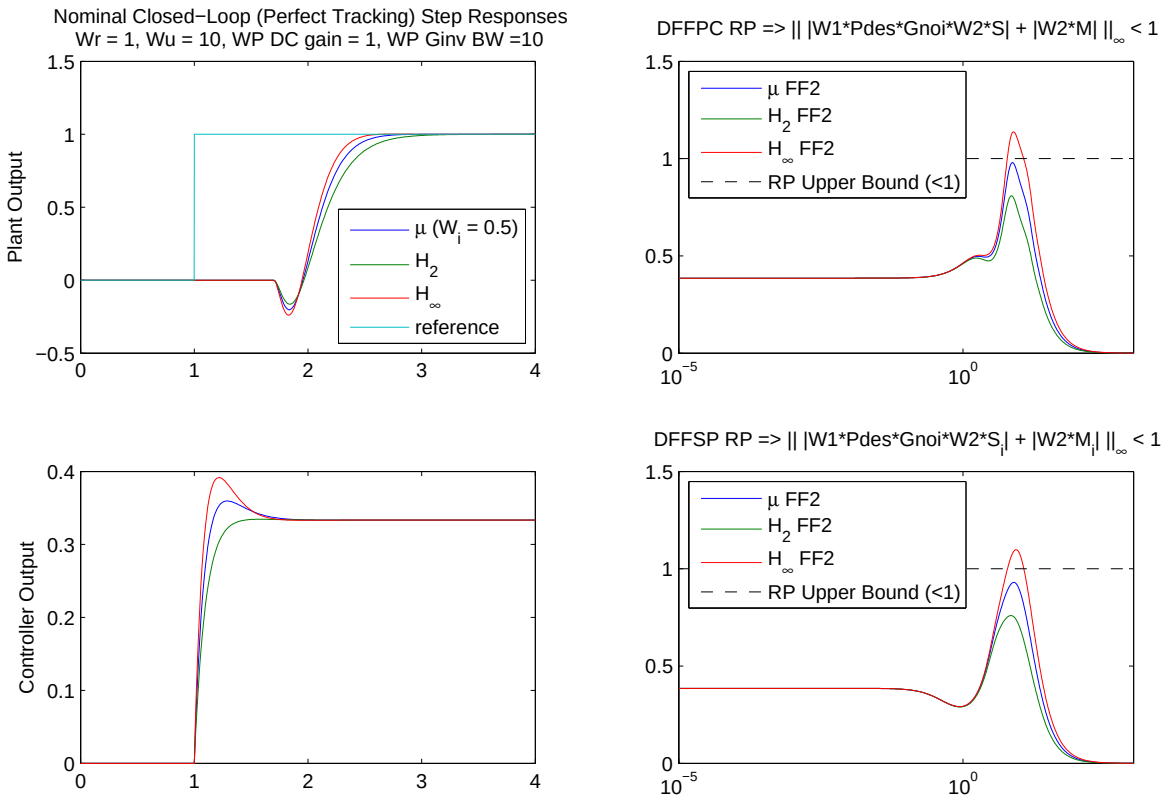

Figure 9.17: Design #2 $FF2(s)$ Designs

(shown in red). In this case, the two architectures have very similar performance on the system with time delay mismatch, which results from a different feedback controller design. As with design #1, this phenomenon was predicted by the robust performance conditions (see the robust performance plots on the right side of Figure 9.17 where the feedforward design meets the robust performance criteria for both architectures). This example also shows the inherent tradeoff between faster rise time and more undershoot for systems with RHP zeros.

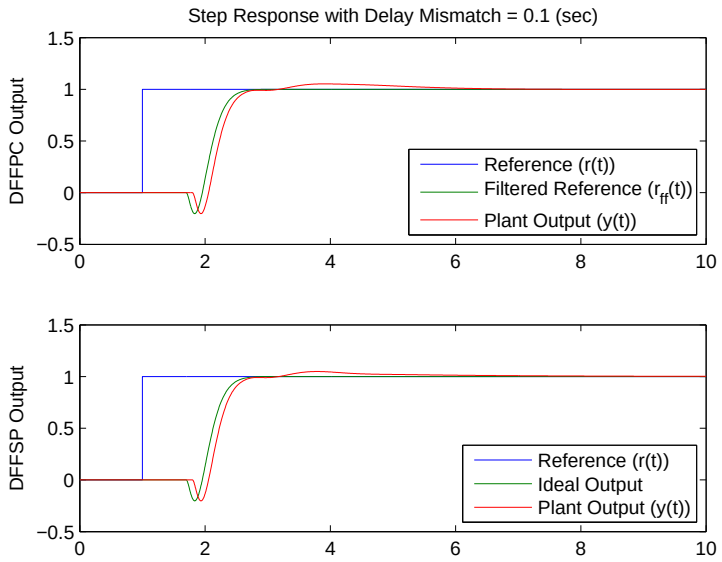

Figure 9.18: Design #2 Perturbed Step with an Actual Process Delay of 0.8 Seconds

9.2.4 Discrete-Time Implementation with Adaptation

In this section, the final controllers from design #2 above are implemented in discrete-time. The concept of perfect tracking is then verified for the discrete-time implementation. Then, the plant model and time delay parameter are perturbed to a different LTI system. Through model identification adaptation, the perfect tracking property is restored. For this section, only the DFFPC will be used.

To begin, the sampling rate is chosen to be $T_s = 0.01$ seconds. With this sampling rate, the continuous time plant is converted to the zero-order hold equivalent plant

$$G_{ZOH}(z) = \frac{-0.22482(z-1.062)}{(z-0.9512)(z-0.9048)} z^{-70}. \tag{9.52}$$

From this, the plant components are $K_{\text{DC}} = 3$, $N_{nmp}(z) = \left(\frac{z}{1-1.062} - \frac{1}{1-1.062}\right)$, $N_{mp}(z) = 1$, $D(s) = \left(\frac{z}{1-0.9512} - \frac{1}{1-0.9512}\right)\left(\frac{z}{1-0.9048} - \frac{1}{1-0.9048}\right)$, and the time delay is 70 samples (i.e., $70 \times T_s = 0.7$ seconds). The invertible/non-invertible decomposition is then

$$G_{noi}(z) = \left(\frac{z}{1-1.062} - \frac{1}{1-1.062}\right) z^{-70} \quad (9.53)$$

$$G_i^{-1}(z) = \frac{1}{3}\left(\frac{z}{1-0.9512} - \frac{1}{1-0.9512}\right)\left(\frac{z}{1-0.9048} - \frac{1}{1-0.9048}\right). \quad (9.54)$$

These components will be used to define the two feedforward controllers $FF1(z)$ and $FF2(z)$. The other two pieces that need to be discretized are $P_{\text{des}}(s)$ and $K(s)$. Both of these have discretizations have their own considerations. For $P_{\text{des}}(s)$, a bilinear Z-transformation may be used; however, there are some extra terms that result from the bilinear Z-transform that need to be removed for $P_{\text{des}}(z)$ to satisfy the relative degree design constraint. To illustrate this, consider the bilinear Z-transform of $P_{\text{des}}(s)$ from the previous section.

$$P_{\text{des}}(s) = \frac{691.1135}{(s+11.78)(s^2+14.4s+58.68)} \xrightarrow{\text{BLZ}} \frac{7.6e-005(z+1)^3}{(z-0.8888)(z^2-1.86z+0.8658)} \quad (9.55)$$

This Z-transform has three zeros at $z = 1$, which are a result of the zeros at infinity in $P_{\text{des}}(s)$ (i.e., $P_{\text{des}}(s)$ has relative degree of three). For this discrete-time $P_{\text{des}}(z)$, these zeros are not required and may be removed; however, the DC constraint that $P_{\text{des}}(1) = 0$ must hold for $P_{\text{des}}(z)$. This may be accounted for by scaling the Z-transform such that $P_{\text{des}}(1) = 0$. In this case, $(z+1)^3$ evaluated at $z = 1$ resulted in multiplication by 8, which produced

$$P_{\text{des}}(z)\frac{0.000608}{(z-0.8888)(z^2-1.86z+0.8658)}. \quad (9.56)$$

For the discrete-time controller $K(z)$, the design should include the effect of the sample and hold implementation. For more details on this, see [41]. For the example here, a bilinear transform is used to convert the $K(s)$ from design #2 (i.e., $K(s)$ in eqn (9.43)) to $K(z)$, which results in

$$K(z) = \frac{0.00066685(z-0.9231)(z-0.9048)(z-0.9512)(z-0.9718)(z-0.9894)(z+1)}{(z-0.8992)(z-0.9342)(z-0.9895)(z-1)(z^2-1.954z+0.9553)} \tag{9.57}$$

Now, the two feedforward controllers are given by

$$FF1(s) = P_{\text{des}}(z)G_{noi}(z) = \frac{-0.0098175(z-1.062)}{(z-0.8888)(z^2-1.86z+0.8658)}z^{-70} \tag{9.58}$$

$$FF2(s) = P_{\text{des}}(z)G_i^{-1}(z) = \frac{0.043667(z-0.9512)(z-0.9048)}{(z-0.8888)(z^2-1.86z+0.8658)} \tag{9.59}$$

Using these controllers, the discrete-time DFFPC architecture is simulated to demonstrate perfect tracking. This is shown in Figure 9.19. For display purposes, the outputs are plotted as "continuous" lines (i.e., straight lines connect each of the samples); however, the data really consists of samples spaced $T_s = 0.01$ seconds apart in time.

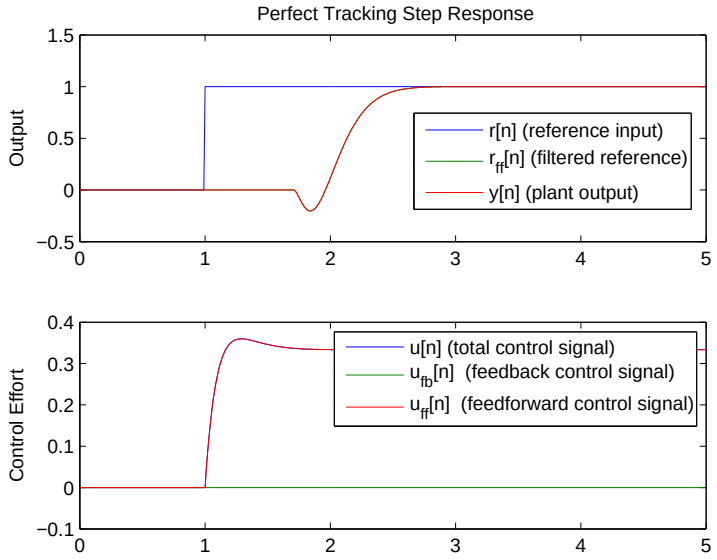

Figure 9.19: Perfect Tracking with a Discrete-time DFFPC Controller

This example demonstrates that perfect tracking may be achieved in both continuous-time and discrete-time.

9.2.4.1 Adaptation

For the final part of this example, the zero-order hold equivalent plant model is modified. In this case, the perfect tracking property will not hold. To see this, we redefine the plant to be

$$G_{ZOH}^{new}(z) = \frac{-0.21(z-1.07)}{(z-0.95)(z-0.91)} z^{-75}. \tag{9.60}$$

The plot of the original DFFPC controller on this plant is shown in Figure 9.20.

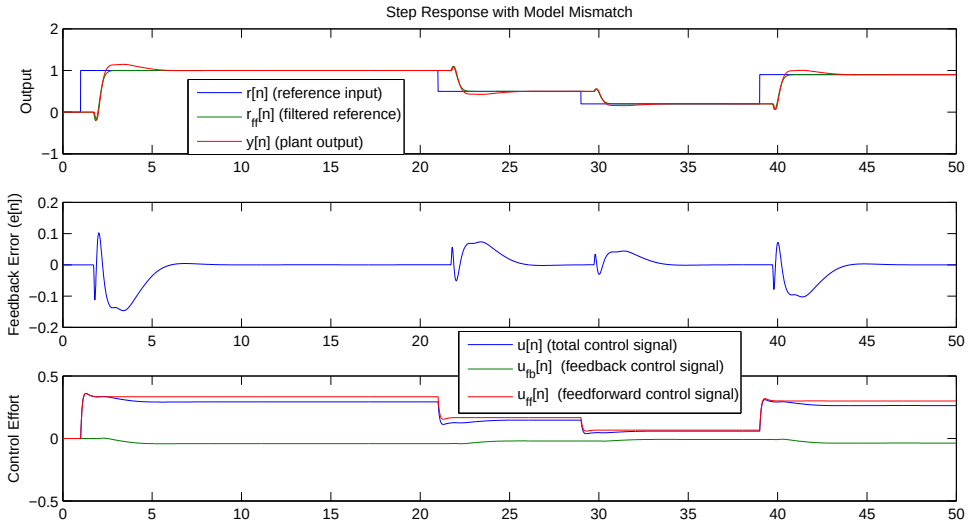

Figure 9.20: Perturbed Step Response using a Discrete-time DFFPC Controller

This new plant model has both a time delay mismatch of 0.05 seconds and a parametricaly perturbed plant. The result is that the perfect tracking property does not hold. For this particular example, the plant is LTI and may be identified using the system identification techniques provided in Chapter 7. For this example, we use a history of the time series data from $u[n]$ and $y[n]$ to identify the discrete-time plant. Specifically, an autoregressive moving-average model was fit to the data using the System Identification toolbox command `arx` in MATLAB [85]. Notice that the plant input (i.e., $u[n] = u_{ff}[n] + u_{fb}[n]$) is used for the plant identification and not just the

feedforward control signal $u_{ff}[n]$. For this particular example, the entire time series from Figure 9.20 (i.e., 50 seconds worth of data sampled every 0.01 seconds) was used for plant identification. In practice, this much data may not be needed to identify the model. Also, more frequent plant identification may be desirable. For the case presented here, the plant model was identified accurately and the perfect tracking property is restored. The results are shown in Figure 9.21.

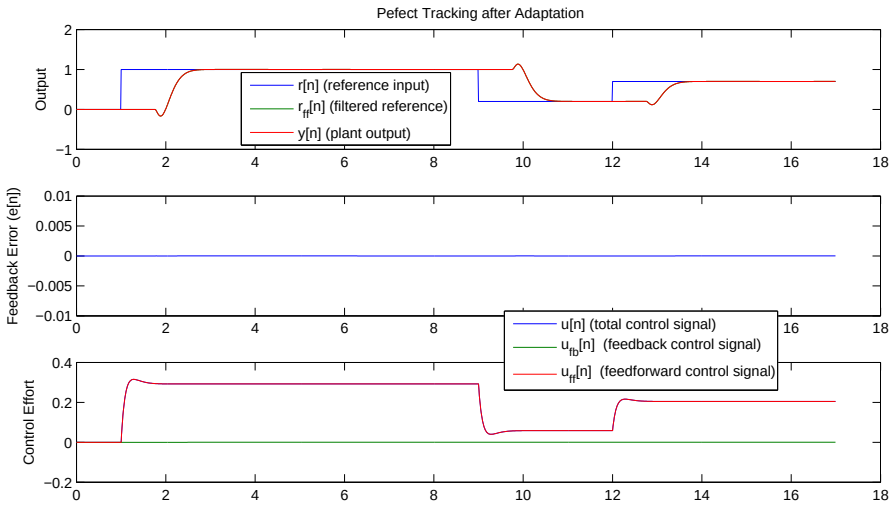

Figure 9.21: Perfect Tracking Restored after Plant Identification

This example shows one method of applying adaptation. The use of adaptation to improve controller performance is a topic of ongoing research. This is discussed in Chapter 12.

9.3 Unstable Non-minimum Phase Plant with a RHP Zero and Time Delay

The purpose of the last illustrative example is to show the general capabilities of the DFFPC architecture. For this example, only the nominal case with no external disturbances (i.e., perfect tracking) is presented. For this, an unstable non-minimum phase plant with a time delay is considered. This is an example of a plant that is

difficult to control using standard feedback controller techniques. The plant model is given by

$$G(s) = \frac{10(s-2)}{(s+10)(s-5)}e^{-2s} = \frac{0.4(\frac{-s}{2}+1)}{(\frac{s}{10}+1)(\frac{-s}{5}+1)}e^{-2s} \tag{9.61}$$

$K_{\rm DC} = 0.4$, $N_{nmp}(s) = (\frac{-s}{2}+1)$, $N_{mp}(s) = \frac{s}{10}+1$, and $D(s) = \frac{-s}{5}+1$. The invertible/non-invertible decomposition is then given by

$$G_{noi}(s) = (\frac{-s}{2}+1)e^{-2s}, \quad G_i^{-1}(s) = \frac{5}{2}(\frac{s}{10}+1)(\frac{-s}{5}+1). \tag{9.62}$$

In order to guarantee that the two feedforward controllers are proper, $P_{\rm des}(s)$ must have a relative degree of two or more. For the example here,

$$P_{\rm des}(s) = \frac{1}{(\frac{s}{\alpha_{\rm des}}+1)^2}, \tag{9.63}$$

which satisfies the relative degree constraint and the design constraint that $P_{\rm des}(0) = 1$. With this choice of $P_{\rm des}(s)$, $FF2(s)$ is given by

$$FF2(s) = P_{\rm des}(s)G_i^{-1}(s) = \frac{5}{2}\frac{(\frac{s}{10}+1)(\frac{-s}{5}+1)}{(\frac{s}{\alpha_{\rm des}}+1)^2}. \tag{9.64}$$

The nominal closed-loop transfer function is given by

$$M_{\rm DFFPC}(s) = P_{\rm des}(s)G_{noi}(s) = \frac{\frac{-s}{2}+1}{(\frac{s}{\alpha_{\rm des}}+1)^2}e^{-2s}. \tag{9.65}$$

The right-half plane zero in this transfer function means that the step response will have an undershoot that is related to bandwidth of $P_{\rm des}(s)$, which is $\alpha_{\rm des}$ rad/sec. In particular, for a faster response, a larger the bandwidth is required that will result in more undershoot. This phenomenon is described in [54].

The trade-offs for various $P_{\rm des}(s)$ bandwidths are shown in Figure 9.22. The limiting dynamics on the closed-loop response are the right half plane zero at $s = 2$ and the time delay of two seconds, which appear in the nominal closed-loop response

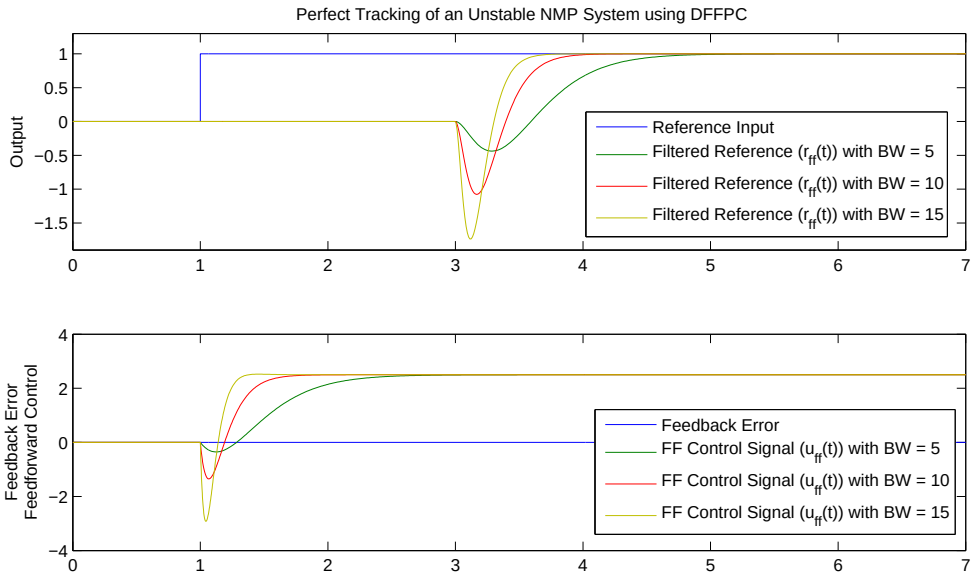

Figure 9.22: Perfect Tracking with Various α_{des} Bandwidths

$P_{\text{des}}(s)G_{noi}(s)$. The fact that the plant is unstable only means that the control signal that provides perfect tracking to steps has an undershoot, which is a result of the non-minimum phase zero in $G_i^{-1}(s)$. The fact that the plant is unstable makes it harder to control using standard feedback techniques, but does not affect the achievable closed-loop transfer function.

9.4 Summary

These examples show the overall characteristics and limitations of the controllers used. For example, when the plant is stable, $G_i^{-1}(s)$ is minimum-phase, which means that a minimum-phase control signal may be used to provide perfect tracking (provided there are no non-minimum phase components in $P_{\text{des}}(s)$. Conversely, when the plant is unstable, $G_i^{-1}(s)$ is non-minimum phase, which means that a non-minimum phase control signal is required to provide perfect tracking. These scenarios are illustrated in Figure 9.22.

When a plant is minimum-phase, closed-loop response may be shaped solely by

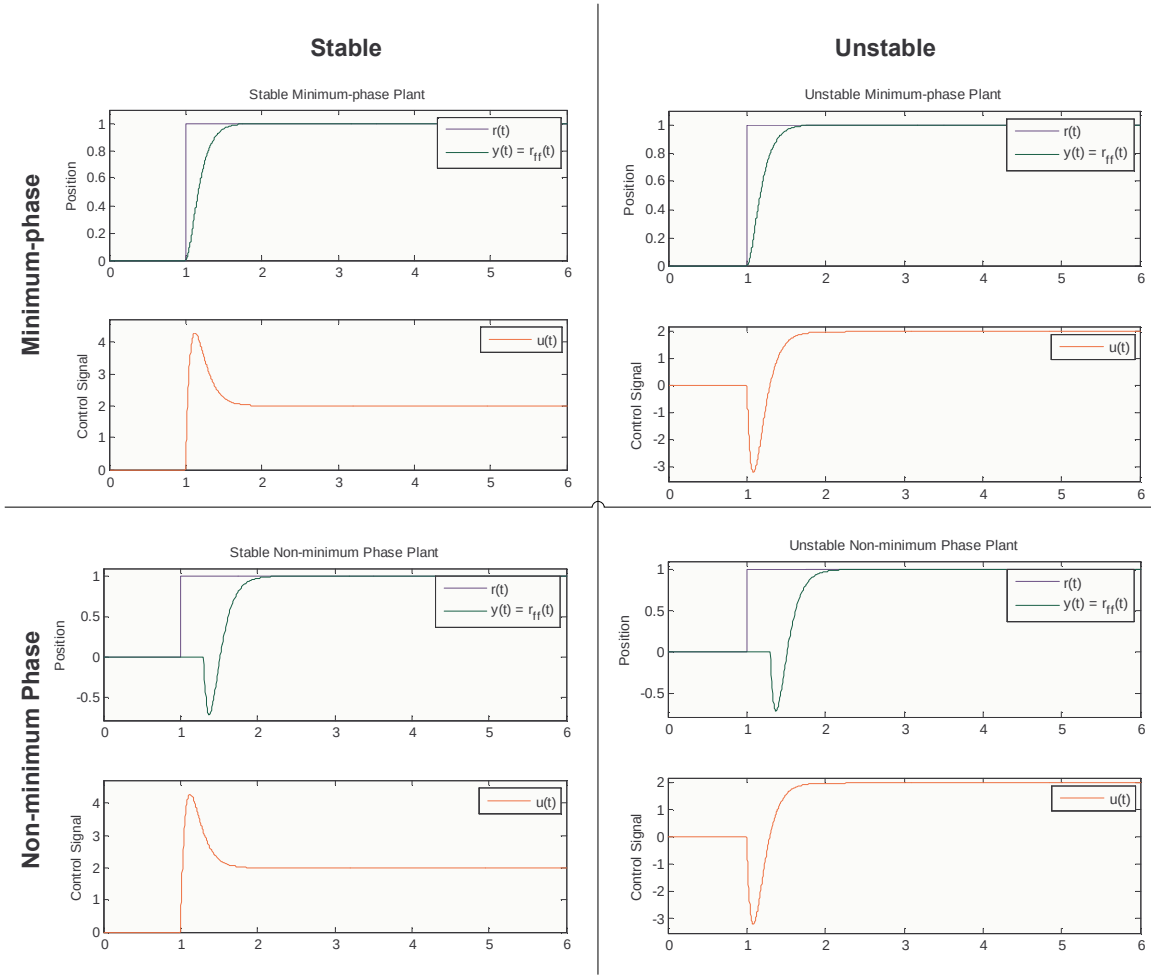

Figure 9.23: Plant Implications

the choice of $P_{\text{des}}(s)$. The only limitation in these cases result from the (right-half plane) zeros at infinity, which put a relative degree constraint on $P_{\text{des}}(s)$. When there are non-minimum phase components in the plant, the $FF2(s)$ controller is not able to stably invert these dynamics in a causal manner, which means that these dynamics will appear in the final closed-loop response. In these cases, the class of signals that may be perfectly tracked (using causal and stable techniques) is defined by $P_{\text{des}}(s)G_{noi}(s)$. Together, the examples from this chapter demonstrate the limitations that right-half plane zeros and time delays put on the achievable closed-loop performance.

Chapter 10

Microalgae Modeling

In this chapter, a photobioreactor (PBR) model is developed that characterizes microalgae growth in a resource limited environment. While many other PBR models exist, they mainly use an empirical model to model resource limited growth. Monod models describe how bacteria grow in a media where the growth changes from exponential in the beginning (when there are plenty of nutrients in the media) to a decaying growth phase, where nutrients deplete out of media (c.f., [86; 10]). In the final stages of Monod kinetics, the growth goes through a stationary phase, and finally a dying phase where biomass is lost. For the photobioreactors (PBRs) considered here, microalgae spend all of their time in the growth phases (up to a possible stationary phase), but are not left in the PBR long enough for the dying stage to occur. While Monod kinetic models do correlate with observed growth, they do not exploit the physics of microalgae growth inside a closed PBR, which we address here. Our early results on this method were published in [22].

Based on the measured model parameters, the *photosynthetic efficiency* (PE) and carbon dioxide (CO_2) uptake efficiency may be measured. These two performance metrics are related since microalgae is approximately 50% carbon which they get from dissolved CO_2 in the surrounding media. Since this CO_2 is provided to the media via a sparged gas, all of the biomass accumulation is a result of carbon sequestered from the input CO_2 gas stream. In this sense, CO_2 uptake and growth are

synonymous. From this, CO_2 uptake efficiency is measured as the amount of CO_2 consumed for growth versus the amount of CO_2 delivered. Similarly, PE measures the amount of algae produced for the amount of incident light versus a theoretical maximum yield. In practice, these two performance objectives are competing, since increased PE is achieved by continuously sparging through a gas mixture with the appropriate CO_2 concentration, and better CO_2 uptake efficiency is achieved through intermittently sparging CO_2. For the model developed here, both continuous and intermittent sparging may be simulated for pH regulation.

The developed model is applied to the flat panel PBR shown in Figure 10.1 that is growing the microalgae strain *Nannochloropsis oculata*. In this setting, certain operating conditions were held constant and only a selection of the modeled parameters was measured. This led to a reduced order model, which is used in the validation and verification (V&V) section. The performance of this system is given in terms of measured model parameters.

10.1 Dynamic PBR Model

The dynamic PBR model is constructed by breaking down the overall system into subsystems. These subsystems are modeled using a combination of the three basic types of microalgae models, namely physically-based, empirical fits from data, and biologically-based models. The different types of models with examples are shown in Figure 10.2.

A combination of all three techniques is used in most microalgae models. Empirical models use data fits to determine the performance of a specific reactor, but the fitted parameters are not in physical units and therefore cannot be applied to different scale reactors. More recently, microalgae models have focused on developing physically based models, since they are in real units and therefore may be applied to both small and large scale reactors. In physically based models, the model parameters often

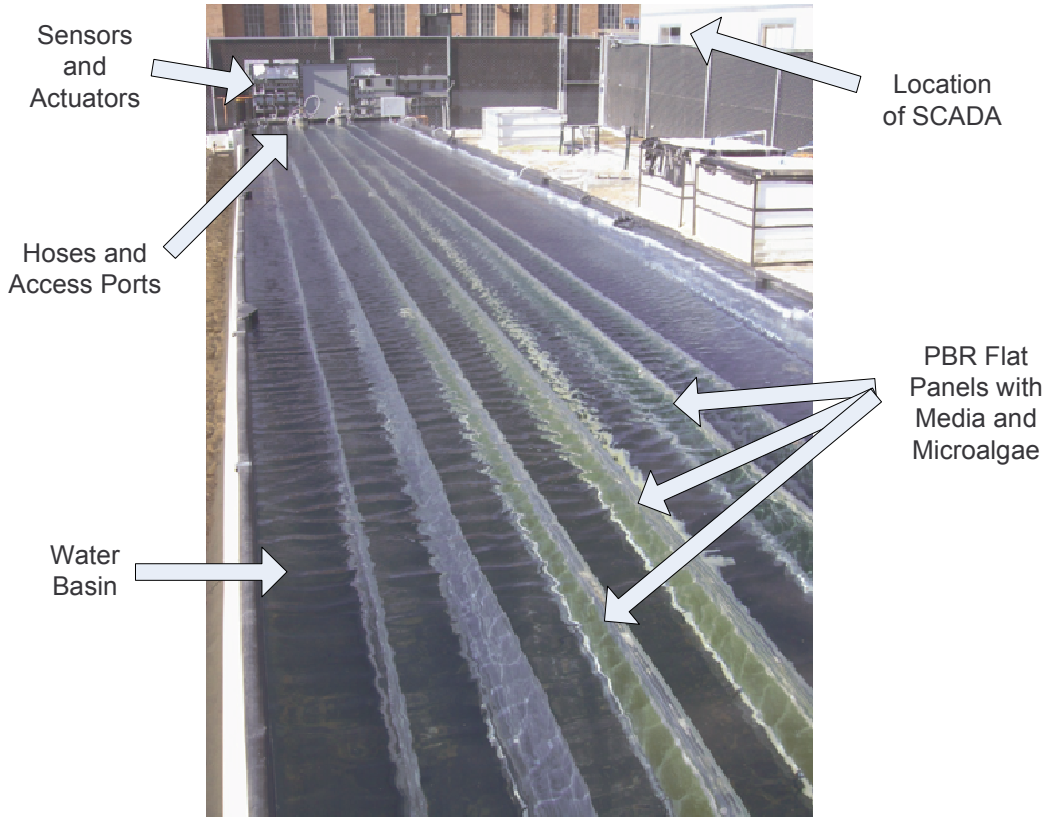

Figure 10.1: Early Photobioreactor used for Modeling and Controller Development

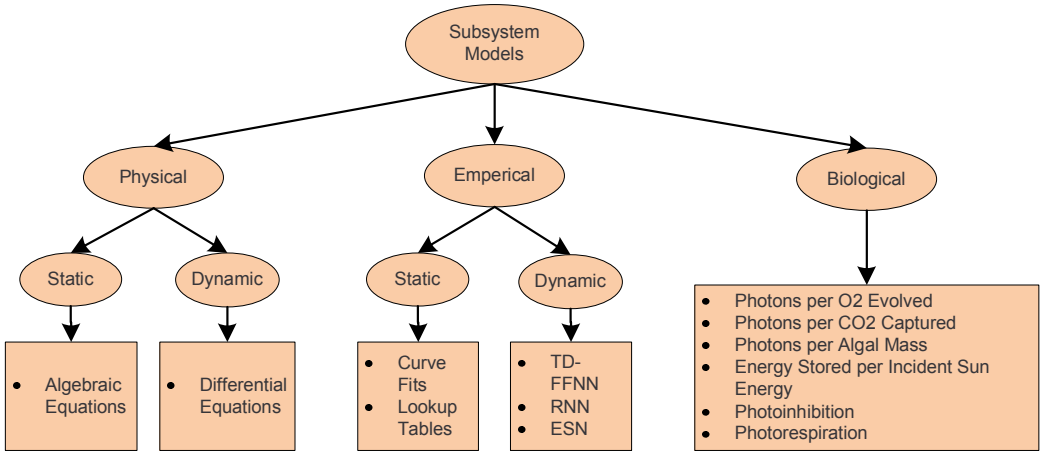

Figure 10.2: PBR Modeling Techniques

need to be measured. Biological models provide a way to convert between different measures of growth. For example, biomass accumulation and CO_2 uptake are both measures of growth that may be related based on accepted biological facts. Biological

models are also useful for creating PE models, since they provide finite constraints on the achievable growth for a given amount of photosynthetically active radiation (PAR) from the sun that is incident on the PBR. In the work presented here, the overall model is unique in that it takes a different approach to addressing the physics of microalgae growth inside a PBR.

The PBR shown in Figure 10.1 was an earlier prototype that was used for modeling and controller development. This prototype contained all of the components of a full PBR; however, much work has been done to improve the operation and overall yield from this early prototype. This experimental PBR testbed[1] consisted of

- a water basin that provides support for the closed flat panel PBR,
- closed flat panels that contain media and microalgae,
- infrastructure (e.g., hoses, pipes, and access ports) to deliver CO_2 and nutrients to the microalgae during growth and remove the microalgae at harvest time,
- sensors to measure key variables (e.g., basin temperature, pH, dissolved O_2, and culture density) and
- a *supervisory control and data acquisition* (SCADA) system that provides control inputs (e.g., commanded CO_2 flow rate and sparge air flow rate), store data from sensors, and send alarms when the system malfunctions.

In order to make a model that is scalable, physical units were selected as densities and concentrations. Therefore, the same performance metrics may be used to evaluate both small and large scale reactors. From these, the aggregate values may be obtained by multiplying the densities and concentrations by the reactor dimensions and mass (gas) flow rates, respectively. A density based model is developed by considering the

[1]In general, a production system would not contain as many sensors as this testbed.

growth dynamics in a single flat panel. Figure 10.3 illustrates the dynamic interactions of a single flat panel.

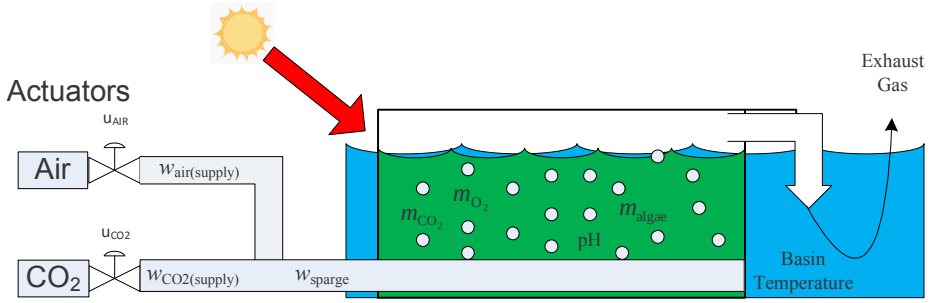

Figure 10.3: Individual Flat Panel

PAR from the sun is incident to the PBR water basin. Some of this light enters the bath and is used for photosynthesis by the microalgae, while the rest is reflected. The amount of reflection is based on the angle of incidence of the incoming PAR. This interaction makes up the light subsystem. The flat panel PBRs are submerged in a water basin that provides some structure for the flat panel PBRs, regulates the temperature of the microalgae and media mixture inside the closed flat panels, and distributes the PAR that enters the bath.

Inside the closed flat panel reactors, microalgae cells are suspended in an industry standard *media* that essentially mimics nutrient rich seawater. A CO_2 enriched gas stream is sparged [2] through the mixture of microalgae and media. The gas concentration in the sparging bubbles will seek equilibrium with the gas concentration in the media. In the presence of light, microalgae photosynthesize to produce more biomass. During this process, the microalgae interact with the media to remove nutrients and dissolved carbon while releasing dissolved oxygen (DO) back into the media. For sustained growth, the microalgae need the media to be supplied with additional dissolved carbon and purged of built up DO from photosynthesis. The interaction between the

[2]Sparging is the act of bubbling a gas through the media to provide gas-liquid mass transfer between the bubbles and media.

sparging bubbles and the media make up the water chemistry subsystem, and the interaction between the microalgae cells and and the nutrient rich media make up the photosynthesis subsystem. These phenomena are captured in the PBR model in Fig. 10.4 that contains the three major subsystems, namely the *light subsystem* (red), the *photosynthesis subsystem* (green), and the *water chemistry subsystem* (blue). The details of each subsystem are described in the remainder of this section.

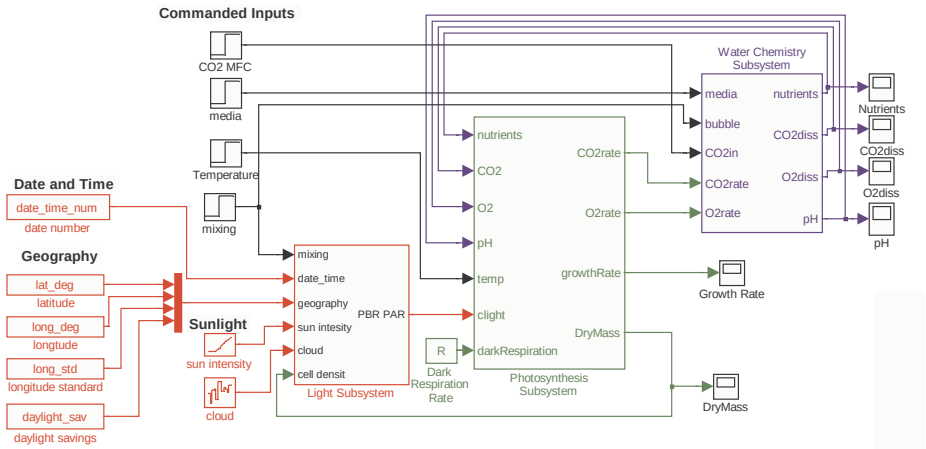

Figure 10.4: Simulink Model of a PBR.

10.1.1 Incident Light Subsystem

The incident light subsystem determines the amount of light that will reach the microalgae, which is a function of incident light to the PBR bath, sun position, amount of mixing, culture density, and PBR geometry. This section describes a model based on incident light. While mixing, culture density, and PBR geometry affect the amount of light received by the microalgae, they are fixed for the operating conditions studied here. Therefore, these parameters will be grouped into a sun utilization constant and a critical density in the growth model, which will be discussed in the next section. By grouping these terms together, a simplified model may be developed that still captures all of the necessary information about the microalgae interaction with the sunlight.

About 43% of the full spectrum of sunlight is photosynthetically active radiation (PAR) which is the amount of light available for photosynthesis on earth. PAR is the light intensity in the 400nm to 700nm range. When the sun is out, the primary component of incident PAR is direct light, which will hit the bath water at a certain angle depending on the position of the sun. A portion of this light will reflect back off the water and some will enters the PBR bath. Not all of the light that enters the bath will be absorbed, but modeling the amount of direct light that enter the bath captures enough information about the light available for photosynthesis to provide a realistic growth model. The following derivation of reactor light is based on the information in [87].

The amount of direct sunlight that enters the bath water is a function of the angle of incidence normal to the bath water. In turn, this angle is a function of the sun position, which depends upon the day of the year, time of day, and location (longitude and latitude). As the earth travels around the sun, the relative position of the sun in the sky changes with the seasons. This is captured by the sun declination, which is

$$\delta = 23.54 \sin\left(2\pi \frac{360}{365}(284 + n)\right) \tag{10.1}$$

where $1 \leq n \leq 365$ is the day of the year. The sun intensity is a function of solar time, where solar time is the local time adjusted so that the sun is the highest in the sky at solar noon. The conversion from local time to solar time is as follows:

$$B = \frac{360}{365}(n-1) \tag{10.2}$$

$$E = 0.000287 + 0.0072\cos(2\pi B) - 0.1225\sin(2\pi B) - 0.0558\cos(4\pi B) - 0.1562\sin(4\pi B)) \tag{10.3}$$

$$D = \begin{cases} 1 & : \quad n \text{ during daylight savings} \\ 0 & : \quad n \text{ during standard time} \end{cases} \tag{10.4}$$

$$\Delta t = \frac{(L_{\text{st}} - L_{\text{loc}})}{15} + E - D \tag{10.5}$$

$$t_{\text{solar}} = t_{\text{clock}} + \Delta t \tag{10.6}$$

In these equations, E is a correction in hours based on the day of the year (n). The variables L_{st} and L_{loc} are the standard[3] and actual longitude values in degrees for the PBR location, and the flag variable D in equation (10.4) is equal to one when it is during daylight savings and zero otherwise.

The next parameter to calculate is the "hour angle," which measures the number of degrees that the earth has traveled since solar noon. Because there are 360° of rotation in a 24 hour day, the earth travels 15 degrees every hour (hence the division by fifteen in equation (10.5)). The hour angle (in radians) is given by

$$\omega = 2\pi \times [15(t_{\text{solar}} - 12)]. \tag{10.7}$$

The angle of incidence (θ_{inc}) on a horizontal surface, such as the PBR bath, at a given latitude ϕ_{lat} is

$$\cos(\theta_{\text{inc}}) = \cos(\phi_{\text{lat}})\cos(\delta)cos(\omega) + \sin(\phi_{\text{lat}})\sin(\delta) \tag{10.8}$$

From Snell's Law, the angle of transmission into the water, namely θ_{water}, is given by

[3]The standard longitudes for the United States are 75° for the Eastern time zone, 90° for the Central time zone, 105° for the Mountain time zone, and 120° for the Pacific time zone.

$$\frac{n_{\text{air}}}{n_{\text{water}}} = \frac{\sin(\theta_{\text{air}})}{\sin(\theta_{\text{water}})} \tag{10.9}$$

where $n_{\text{air}} = 1$ and $n_{\text{water}} = 1.333$ are the indices of refraction for air and water, respectively, and $\theta_{\text{air}} = \theta_{\text{inc}}$ from equation (10.8). This is enough information to calculate θ_{water}. To get the fraction of direct beam radiation that is transmitted through the water, two more variables are used, namely the perpendicular and parallel components of unpolarized radiation, which are given by

$$r_{\perp} = \frac{\sin(\theta_{\text{water}} - \theta_{\text{air}})^2}{\sin(\theta_{\text{water}} + \theta_{\text{air}})^2} \tag{10.10}$$

$$r_{\parallel} = \frac{\tan(\theta_{\text{water}} - \theta_{\text{air}})^2}{\tan(\theta_{\text{water}} + \theta_{\text{air}})^2}. \tag{10.11}$$

From this, the reflectance is $\frac{r_{\perp}+r_{\parallel}}{2}$ and the transmittance (or fraction of the light that enters the bath) is given by

$$\eta_{\text{bath}} = 1 - \frac{r_{\perp} + r_{\parallel}}{2}. \tag{10.12}$$

If PAR_{sun} is the amount of PAR from the sun, then the amount that will enter the PBR bath is

$$\text{PAR}_{\text{bath}} = \eta_{\text{bath}}\text{PAR}_{\text{sun}} \tag{10.13}$$

The actual amount of PAR that the microalgae will use for photosynthesis is also a function of mixing and vertical flat panel geometry (i.e., panel thickness and orientation). Therefore, the amount of incident light available for algal photosynthesis will be

$$I_{\text{PAR}} = f_1(\text{PAR}_{\text{bath}}, \text{mixing}, \text{geometry}), \tag{10.14}$$

where $f_1(\cdot)$ is some nonlinear function. A simplified model of equation (10.14) is

$$I_{\mathrm{PAR}} = \eta_{\mathrm{PBR}} \mathrm{PAR}_{\mathrm{bath}}, \tag{10.15}$$

where η_{PBR} is the efficiency of the PBR for a given mixing and geometry. Currently, the term η_{PBR} is absorbed into the light utilization constant K_{PAR} in the next section. Therefore, $I_{\mathrm{PAR}} = \mathrm{PAR}_{\mathrm{bath}}$ is used for the growth model. However, as more information becomes available about the effects of mixing and geometry, it will be incorporated into calculating a more accurate I_{PAR}.

PAR may be measured in units of $\frac{\mu\mathrm{mol\ light}}{m^2 s}$; however, it is more convenient to convert PAR to units of $\frac{\mathrm{mol\ light}}{m^2 h}$. The convenience comes from the fact that 8 moles of light should produce 1 mole of O_2 and that the growth rate is measured biomass produced per hour. This will become apparent in the next section. The conversion between the two PAR units is given by

$$\frac{\mathrm{mol}}{m^2 h} = 0.0036 \, \frac{\mu\mathrm{mol}}{m^2 s}. \tag{10.16}$$

10.1.2 Growth Subsystem

The growth subsystem models the dynamics of the microalgae as they utilize photons from the sun, CO_2, and nutrients to produce oxygen (O_2) and more microalgae. The rate at which microalgae grow depends on their ability to utilize the incident light and on the availability of nutrients. Assuming there are ample nutrients available, microalgae growth is primarily a function of input light. Growth only happens when there is available light; however, they will respire all of the time (i.e., in both the presence and absence of light). During respiration, the microalgae will utilize O_2 and stored carbon as an energy source and release CO_2 back into the media. Respiration results in a loss of biomass. In the presence of light, growth (and hence carbon

assimilation) will dominate the metabolic process[4]. This means that growth will be a function of the amount of microalgae that are exposed to light. In closed PBRs, microalgae are grown to thick densities that will result in light limited growth. This phenomenon is illustrated in Figure 10.5.

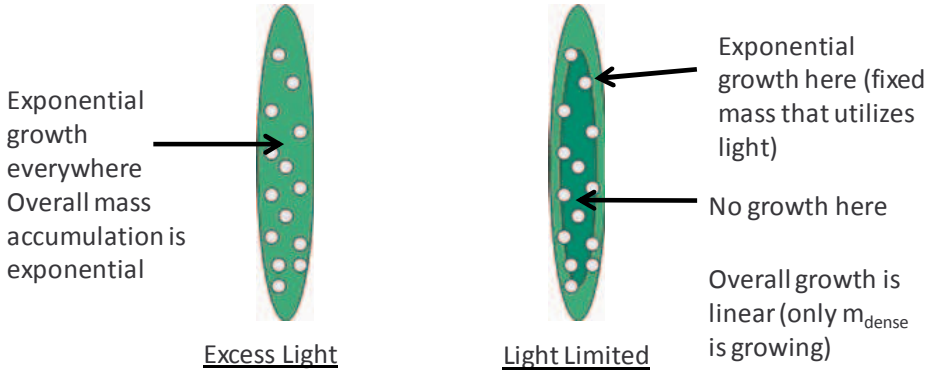

Figure 10.5: Light Limited Growth Inside a Closed PBR

When the culture is sparse, there are an excess number of PAR photons that are not being utilized. Under this condition, microalgae will grow exponentially, since the produced algal mass will not be limited by available photons. This is illustrated on the left side of Figure 10.5. At some point, the algal density will become great enough that all of the incident light will be utilized. At densities greater than this, the microalgae growth rate will be linear. This is illustrated on the right side of Figure 10.5. As the density continues to increase, a smaller fraction of the microalgae will be able to receive the amount of light required for photosynthesis and respiration will be the dominant metabolic activity. As this happens, the total microalgae growth in the PBR will cease and eventually begin to decay. In the model, this feature is captured by saturating the density in the growth term. When the density gets above a critical density, labeled m_{dense}, the amount of growth resulting from photosynthesis becomes linear while the density lost due to respiration remains exponential. These

[4]Microalgae contain the enzyme Rubisco that will utilize both CO_2 and O_2 as substrates.

effects are described by the following nonlinear differential equation.

$$\dot{m}_{\text{algae}} = K_{\text{PAR}} I_{\text{PAR}} \bar{m}_{\text{algae}} - R m_{\text{algae}} \tag{10.17}$$

where

$$\begin{aligned} \bar{m}_{\text{algae}} &= \min(m_{\text{algae}}, m_{\text{dense}}) \\ m_{\text{dense}} &= f(m_{\text{algae}}, \text{mixing, geometry}) \end{aligned} \tag{10.18}$$

The state variable m_{algae} is the amount of microalgae inside the PBR and its derivative, namely $\dot{m}_{\text{algae}}$, is the growth rate of microalgae inside the PBR. The term K_{PAR} is the *sun utilization* parameter that converts incident light, namely I_{PAR}, into microalgae growth rate, and the constant R is the rate of biomass loss due to respiration (in the dark). The microalgae also respire while exposed to sunlight, so the respiration term is still appropriate during growing times. However, the rate of respiration is known to be greater during sunlight exposure. This discrepancy is hard to measure directly. Instead, it is wrapped into the K_{PAR} parameter (i.e., photorespiration results in a smaller sun utilization parameter). Finally, the parameter m_{dense} is the critical density above which the growth becomes linear.

As microalgae grow, they consume carbon, which they get from dissolved CO_2 and other nutrients from their surroundings while releasing O_2. In general, microalgae biomass is 50% carbon by dry weight [2]. A mole of CO_2 has a mass of 44 grams and 12 of these grams come from carbon. Based on these premises, the expression that 1 gram of microalgae can fix 1.83 grams of CO_2 may be derived as follows

$$\frac{44 g\text{CO}_2/\text{mol}}{12 g\text{C}/\text{mol}} \frac{0.5 g\text{C}}{g_{\text{algae}}} = \frac{11}{6} \frac{g_{\text{CO}_2}}{g_{\text{algae}}} = 1.83 \frac{g_{\text{CO}_2}}{g_{\text{algae}}}. \tag{10.19}$$

A simplified equation for photosynthesis[5] is given by

$$12H_2O + 6CO_2 + \text{light} \longrightarrow C_6H_{12}O_6 + 6O_2 + 6H_2O \tag{10.20}$$

This equation states that for every gram of CO_2 consumed, there is a gram of O_2 produced. While this is true, it does not account for all of the O_2 that is produced. This results from the fact that O_2 molecules come from splitting water, which provide energy for all of the metabolic processes inside the microalgae. Therefore, there is not a one-to-one correspondence of O_2 molecules produced to CO_2 molecules fixed. The excess energy that is not used to fix CO_2 is used for other metabolic processes such as fixing nutrients from the surrounding media and cell repair. This is often echoed in the literature by the fact that it takes 8 photons of light to produce one O_2 molecule, but that it takes 8-12 photons of light to assimilate a CO_2 molecule (c.f., [88]). Assuming that 10 photons of light are required to fix one CO_2 molecule, the amount of O_2 produced will be

$$\frac{32 g_{O_2}/\text{mol}_{O_2}}{44 g_{CO_2}/\text{mol}_{CO_2}} \frac{8 \text{ mol}_{O_2}}{10 \text{mol}_{CO_2}} \frac{11}{6} \frac{g_{CO_2}}{g_{\text{algae}}} = 1.07 \frac{g_{O_2}}{g_{\text{algae}}}. \tag{10.21}$$

Since these values are based on densities, they are independent of reactor size, which means that the resulting models will be independent of scale. Based on the assumptions in eqns (10.19) and (10.21), the CO_2 consumption rate (density) and O_2 produced rate (density) may be expressed in terms of the growth rate (density). In particular, the mass production and consumption rates (per gram microalgae) of CO_2 and O_2, respectively, are

---

[5]Note that the reverse of photosynthesis is respiration, which is given by
$C_6H_{12}O_6 + 6O_2 + 6H_2O \longrightarrow 12H_2O + 6CO_2 + \text{energy}$

$$\dot{m}_{\mathrm{CO_2}(g/L/h)} = 1.83\dot{m}_{\mathrm{algae}(g/L/h)} \tag{10.22}$$

$$\dot{m}_{\mathrm{O_2}(g/L/h)} = 1.07\dot{m}_{\mathrm{algae}(g/L/h)}, \tag{10.23}$$

In general, the relationships may be expressed as

$$\dot{m}_{\mathrm{CO_2}(\mathrm{mass/time})} = K_{\mathrm{CO_2}}\dot{m}_{\mathrm{algae}(g/L/h)} \tag{10.24}$$

$$\dot{m}_{\mathrm{O_2}(\mathrm{mass/time})} = K_{\mathrm{O_2}}\dot{m}_{\mathrm{algae}(g/L/h)}, \tag{10.25}$$

where, $K_{\mathrm{CO_2}}$ and $K_{\mathrm{O_2}}$ are the amount of gas consumed/produced per mass of microalgae growth and may be in units other than grams gas per gram microalgae. An example of this is when the amount of CO_2 is measured in standard liters per minute (SLPM). Let $V_{\mathrm{PBR(L)}}$ be the volume of the PBR in liters and using the fact that there are 1.808 grams of CO_2 (gCO_2) per standard liter (SL), then $K_{\mathrm{CO_2}}$ may be expressed as

$$\begin{aligned}\dot{m}_{\mathrm{CO_2}(\mathrm{SLPM})} &= \frac{1h}{60\mathrm{min}}\frac{\mathrm{SL}}{1.808g_{\mathrm{CO_2}}}V_{\mathrm{PBR(L)}}1.83\frac{g_{\mathrm{CO_2}}}{g_{\mathrm{algae}}}\dot{m}_{\mathrm{algae}} \\ &= \frac{1.83V_{\mathrm{PBR(L)}}}{1.808\cdot 60}\dot{m}_{\mathrm{algae}(g/L/h)}\end{aligned} \tag{10.26}$$

Here, $\frac{1.83V_{\mathrm{PBR(L)}}}{1.808\cdot 60}$ is just $K_{\mathrm{CO_2}}$ in different units.

10.1.3 Water Chemistry Subsystem

The water chemistry subsystem models both the dissolved gases and nutrients available to the microalgae in the media. The dissolved gases are a function of both the gases being delivered by the MFCs and the internal gases being consumed and generated by the microalgae. The main purpose of sparging is to regulate the concentrations of dissolved O_2 and dissolved CO_2 through mass transfer. In general, the

gas transfer rates may be modeled locally as a first order dynamic system. Due to the distributed nature of the system, the model would require many cascaded first order systems, which is common with process models. This phenomenon may be essentially captured by using a first order plus dead time model [89], which is the method used here.

When the media in the PBR is at equilibrium with air, there is about 8.3 mg/L of dissolved O_2 in the media, which is maintained through sparging when there is no growth or respiration. During high growth periods, dissolved O_2 will build up in the system and is eventually purged at night. This is described by the following dynamic model.

$$\begin{aligned} \dot{m}_{\mathrm{DO}}(t) &= \alpha_{\mathrm{DO}} w_{\mathrm{sparge}}(m_{\mathrm{DO,gas}}(t-\tau_{\mathrm{d,gas}}) - m_{\mathrm{DO}}(t)) + \dot{m}_{\mathrm{O_2}}(t) && (10.27) \\ &= \frac{1}{\tau_{\mathrm{DO}}}(m_{\mathrm{DO,gas}}(t-\tau_{\mathrm{d,gas}}) - m_{\mathrm{DO}}(t)) + \dot{m}_{\mathrm{O_2}}(t) && (10.28) \end{aligned}$$

Here, w_{sparge} is the flow rate of gas into the PBR, α_{DO} is a lumped parameter that determines the lag time for mass transfer of DO between the media and sparging bubbles, $m_{\mathrm{DO,gas}}$ is the DO level that the media will equilibrate to, and $\dot{m}_{\mathrm{O_2}}$ is the rate of oxygen produced through photosynthesis. For a specific reactor setup, the lag time for reaching equilibrium will be a function of the size and frequency of the bubbles, the location and pattern of the sparging holes, the dimension of the flat panel reactor, and the volume of bubbles in the reactor during sparging. Since these parameters will vary from reactor to reactor due to manufacturing variability and operational variability (e.g., the reactors are not rigid, which means that their width and height of media in the bag is dependent on how much back pressure is in the system), they were grouped into two parameters, namely the sparge rate w_{sparge} that is proportional to the volume of bubbles in the reactor during sparging, and a lumped parameter α_{DO} that is the rate of mass transfer for a given bubble configuration. For the system here, the product $\alpha_{\mathrm{DO}} w_{\mathrm{sparge}}$ determines the overall rate of oxygen diffusion

between the sparge gas and the media.

As the rate of sparging (w_{sparge}) decreases, the lag time (given by $\tau_{\text{DO}} = \frac{1}{\alpha_{\text{DO}} w_{\text{sparge}}}$) increases, which means that the mass transfer between bubbles and media takes longer. In the limit as $w_{\text{sparge}} \to 0$ (i.e., sparging is turned off), $\tau_{\text{DO}} \to \infty$, which means that there is no mass transfer between the bubbles and media. In this case, DO will build up in the media at rate $\dot{m}_{\text{O}_2}(t)$. This is illustrated in eqn (10.28).

The input gas stream is an air-plus-CO_2 gas stream where the amount of added CO_2 varies. This variation may change the equilibrium value $m_{\text{DO,gas}}$. There is a delay from when the CO_2 concentration changes and when the new gas mixture arrives at the media, which is captured by the delay $\tau_{\text{d,gas}}$.

An analogous method may be used to model the dissolved inorganic carbon (DIC), which is modeled in eqn (10.29).

$$\dot{m}_{\text{DIC}}(t) = \frac{w_{\text{sparge}}}{\tau_{\text{DIC}}}(m_{\text{DIC,gas}}(t - \tau_{\text{d,gas}}) - m_{\text{DIC}}(t)) - \dot{m}_{\text{CO}_2}(t) \tag{10.29}$$

Here, $m_{\text{DIC,gas}}$ is the CO_2 gas concentration required for a specific pH. As CO_2 is removed from the media through photosynthesis (i.e., $\dot{m}_{\text{CO}_2}$), the value of $m_{\text{DIC,gas}}$ will be increased to help replace the consumed CO_2. Therefore, this value is always changing during active growth to maintain a constant pH. Due to the distributed nature of the system, there is a delay of $\tau_{\text{d,gas}}$ between when the commanded CO_2 concentration changes, and when the CO_2 reaches the media. This phenomenon is demonstrated in the next chapter, where CO_2 is used to regulate pH.

As CO_2 dissolves in the media, it breaks down into different species, namely aqueous CO_2 ($CO_{2(aq)}$), carbonic acid (H_2CO_3), bicarbonate (HCO_3^-), and carbonate (CO_3^{2-}). The combination of all of these species makes up the total DIC. The amount of aqueous CO_2, namely $CO_{2(aq)}$, is a function of temperature and pressure, which is governed by Henry's Law. Based on the DIC and Henry's law, the following chemical equation will reach equilibrium.

$$CO_{2(aq)} + H_2O \rightleftharpoons H_2CO_3 \rightleftharpoons H^+ + HCO_3^- \rightleftharpoons 2H^+ + CO_3^{2-} \tag{10.30}$$

The amount of each of carbonic acid, bicarbonate, and carbonate species determines the pH. As pH increases, the equilibrium shifts to the left. Similarly, as the pH decreases, the equilibrium shifts to the right. As microalgae grow, $CO_{2(aq)}$ is removed from the surrounding media, which causes the equilibrium to shift to the left and the pH to rise. Some strains of microalgae will also utilize bicarbonate as a carbon source. The microalgae still require CO_2, which they get by splitting bicarbonate into $HCO_3^- \rightleftharpoons CO_2 + OH^-$. The release of the OH^- also causes the pH to increase. It is unclear which method of carbon assimilation dominates the pH increase.

The addition of dissolved CO_2 decreases the pH of the media. Since it can take 2-3 seconds for carbon to completely dissolve and only a fraction of the input CO_2 dissolves before leaving the vent, there are some dynamics associated with the pH in the media, which are captured by the first order dynamics (transfer function) in eqn (10.31). The pH model is linearized about an operating pH (e.g., a pH of 7.3).

$$\dot{\text{pH}}(t) = \frac{1}{\tau_{\text{pH}}}(K_{\text{pH}} m_{\text{DIC}}(t) - \text{pH}(t)) \tag{10.31}$$

Here, τ_{pH} is the lag time associated with the DIC settling into the appropriate species and K_{pH} is the conversion factor from DIC to pH units.

10.2 Photosynthetic Efficiency

In order to quantify a PBR's performance, there are two questions that need to be answered, namely

- What is the maximum biomass yield achievable?
- What is the actual *photosynthetic efficiency* (PE) achieved for a specific PBR?

The first question is surrounded by some controversy that depends on the method used. In particular, the maximum yield is somewhere between $2.41 \frac{g \text{ algae}}{\text{mol photons}}$ and $3 \frac{g \text{ algae}}{\text{mol photons}}$. The methods that achieve these are presented in this section. Some of these methods are well documented in the literature (c.f., [90; 91; 8; 2]), while another method is provided based on assumptions gathered from the literature. The results may be used to provide an estimate of the maximum productivity for a given amount of incident PAR. This is addressed in the next section. For the second question, a combination of the model parameters and the maximum theoretical yield may be used to determine the PE. As a consequence, this translates the sun utilization model parameter K_{PAR} into a performance metric and puts fundamental limits on how large this parameter can be.

10.2.1 Theoretical Yields

There are a few different ways that PE can be defined. The two most common ways to define PE are based on biomass accumulated and energy stored. Biomass accumulation may be measured in terms of biomass produced, CO_2 consumed, O_2 produced, and photons of light captured. In the literature, there are a few assumptions that may be used to relate these various growth measures. These were stated in Section 10.1.2 and may be summarized as:

- 1.83 grams of CO_2 are consumed for every gram of microalgae produced [2].
- 8 photons of light are absorbed by the microalgae for every molecule of oxygen produced [91]
- Approximately 10 photons of light are absorbed for every molecule of CO_2 consumed [92; 88].

From these assumptions, the maximum yield based on biomass accumulation may be derived using the following relationship.

$$\left(\frac{1g \text{ algae}}{1.83g \text{ } CO_2}\right)\left(\frac{44.1g \text{ } CO_2}{1 \text{ mol } CO_2}\right)\left(\frac{1 \text{ mol } CO_2}{10 \text{ mol photons}}\right) = 2.41\frac{g \text{ algae}}{\text{mol photons}}. \quad (10.32)$$

This provides a maximum theoretical yield based on assumed accumulation per incident light. For a theoretical yield based on energy stored, a *mathematical abstraction of the photosynthesis equation* given in eqn (10.20) may be expressed as

$$2H_2O + CO_2 + 8 \text{ photons} \longrightarrow CH_2O + O_2 + H_2O. \quad (10.33)$$

It should be noted that this equation does not represent an actual step in photosynthesis. In photosynthesis, it takes six molecules of CO_2 to make one molecule of $C_6H_{12}O_6$.[6] However, it is common practice in the literature to write the true equation of photosynthesis as the mathematical abstraction in eqn (10.33). The reason for this is that it allows for an equation of photosynthesis that is based per molecule of CO_2. However, this is not a true chemical equation and there is not an actual CH_2O that occurs in nature during photosynthesis.

Equation (10.33) suggests that a CO_2 molecule is assimilated for every 8 photons absorbed. However, it is well known that mechanisms that release O_2 and assimilate carbon take place in two separate stages. Microalgae use light (or more specifically, PAR) to split water (H_2O) into hydrogen ions (H^+ ions) and oxygen molecules (O_2). This takes place in two subsystems that are traditionally labeled Photosubsystem I and Photosubsystem II. In these subsystems, it takes a total of 8 photons of light to release one O_2 molecule (c.f., [91]). All of the released energy (in the form of H^+ ions) are used by the microalgae to perform cell functions. The primary function is to store carbon, but some of this energy is used for other cellular activities such as cell repair. Therefore, not all of the energy captured from light is used for carbon

[6]See eqn (10.20). In that equation, "light" would equal 48 photons.

assimilation, which means that it takes more than 8 photons to assimilate one CO_2 molecule.

In the limiting case where all of the incident photons are used for carbon assimilation, eqn (10.33) may be used to derive a yield based on energy. For the discussion here, energy content is defined in terms of the amount of energy released when a substance is burned. For this, the amount of energy in a mole of (the fictitious element) CH_2O and the amount of energy in a gram of microalgae must be determined. There are some discrepancies as to how much energy is in each of these. These are discussed in detail in [90], and the following nominal values were obtained:

- One mole of CH_2O contains about 482.5 kJ of energy.
- One gram of microalgae contains about 21.6 kJ of energy.

Based on these assumptions, the maximum yield based on energy stored may be derived using the following relationships.

$$\left(\frac{1\text{mol } CH_2O}{8 \text{ mol photons}}\right)\left(\frac{482.5 \text{ kJ}}{1\text{mol } CH_2O}\right)\left(\frac{1g \text{ algae}}{21.9 \text{ kJ}}\right) = 2.75\frac{g \text{ algae}}{\text{mol light}}. \tag{10.34}$$

Based on this higher yield, eqn (10.32) may be used to determine that 8.75 photons of light are used to assimilate on CO_2 molecule[7], which further motivates the experimental range of 8-10 photons required to assimilate one CO_2 molecule. Similarly, the limiting case of eqn (10.32) with 8 photons per CO_2 molecule assimilated gives a maximum theoretical yield of 3 g algae per mole photons. While this does not give an exact value, it does suggest that the maximum achievable yield is about 2.4 to 3 g algae per mole photons. For the work presented here, a nominal value of 2.7 g algae per mole photons will be used.

[7]This relationship may be determined by considering $\left(\frac{1g \text{ algae}}{1.83g \text{ } CO_2}\right)\left(\frac{44.1g \text{ } CO_2}{1 \text{ mol } CO_2}\right)\left(\frac{1 \text{ mol } CO_2}{X \text{ mol photons}}\right) = 2.75\frac{g \text{ algae}}{\text{mol photons}}$ and solving for $X = 8.75$ mol photons of light.

10.2.2 Modeled Photosynthetic Efficiency

For the previously developed growth model, biomass accumulation is described in terms of biomass per volume (i.e., a density) and incident light is described in terms of incident light intensity per area. In order to convert these values into a photosynthetic yield, a reactor specific conversion is needed that relates PBR flat panel volume to PBR flat panel area. Let l, h, w determine the length, height, and width of a single PBR flat panel and assume that there are two sides to each panel. Using the fact that there are $1000L$ in a $1m^3$, the area per volume conversion is given by

$$C_{m^2/L} = \frac{A_{\text{PBR}(m^2)}}{V_{\text{PBR}(L)}} = \frac{2h_{(m)}l_{(m)}}{2h_{(m)}l_{(m)}w_{(m)}1000\frac{L}{m^3}} = \frac{0.001}{w_{(m)}}\left[\frac{m^2}{L}\right] \tag{10.35}$$

In the previous section, it was determined that the maximum yield is given by

$$Y^{\max}_{\text{algae}(g)/I\ (\text{mol})} = 2.7\frac{g\ \text{algae}}{\text{mol light}} \tag{10.36}$$

In terms of the growth model developed earlier, the actual (instantaneous) yield is given by

$$\tilde{Y}^{\text{act}}_{\text{algae}(g)/I\ (\text{mol})} = \frac{\dot{m}_{\text{algae}\ (g/L/h)}}{C_{(m^2/L)}I_{(\text{mol}/m^2/h)}} \tag{10.37}$$

From this, the instantaneous photosynthetic efficiency is given by

$$\tilde{\eta}_{\text{PE}} = \frac{\tilde{Y}^{\text{act}}_{\text{algae}(g)/I\ (\text{mol})}}{Y^{\max}_{\text{algae}(g)/I\ (\text{mol})}} \tag{10.38}$$

In terms of the growth model parameters provided earlier, the instantaneous yield is given by

$$
\begin{aligned}
\tilde{Y}^{\text{act}}_{\text{algae}(g)/I\ (\text{mol})} &= \frac{\dot{m}_{\text{algae}(g/L/h)}}{C_{(m^2/L)} I_{\text{PAR}(\text{mol}/m^2/h)}} \\
&= \frac{K_{\text{PAR}(m^2/\text{mol})} I_{\text{PAR}(\text{mol}/m^2/h)} \bar{m}_{\text{algae}(g/L)} - R_{(1/h)} m_{\text{algae}(g/L)}}{C_{(m^2/L)} I_{\text{PAR}(\text{mol}/m^2/h)}} \\
&= \left(\frac{K_{\text{PAR}(m^2/\text{mol})} \bar{m}_{\text{algae}(g/L)}}{C_{(m^2/L)}} - \frac{R_{(1/h)} m_{\text{algae}(g/L)}}{C_{(m^2/L)} I_{\text{PAR}(\text{mol}/m^2/h)}} \right) \left[\frac{g \text{ algae}}{\text{mol}} \right] \qquad (10.39) \\
&< \frac{K_{\text{PAR}} \bar{m}_{\text{algae}}}{C_{(m^2/L)}} \left[\frac{g \text{ algae}}{\text{mol}} \right] \\
&< \frac{K_{\text{PAR}} m_{\text{dense}}}{C_{(m^2/L)}} \left[\frac{g \text{ algae}}{\text{mol}} \right] \qquad (10.40)
\end{aligned}
$$

It should be noted that eqn (10.39) is only valid for $I_{\text{PAR}(\text{mol}/m^2/h)} > 0$. Also, the yield is only positive when $\dot{m}_{\text{algae}} > 0$ (i.e., when growth is occurring), which only happens when $I_{\text{PAR}} > \frac{R m_{\text{dense}}}{K_{\text{PAR}} \bar{m}_{\text{algae}}}$.

For a given reactor setup, both the rate of respiration in the dark (R) and the conversion factor $C_{(m^2/L)}$ will be fixed parameters, since they are specific to the strain of microalgae and physical reactor, respectively. Therefore, the only two parameters that may be affected by operating conditions are the sun utilization parameter K_{PAR} and critical density m_{dense}. While the critical density will be affected by PBR operating methods, it is primarily a function of mixing and reactor geometry, whereas the sun utilization parameter is capturing all of the effects from carbon and nutrient availability, inhibiting effects of DO build up, and sun utilization from proper mixing (i.e., microalgae need to spend some of their time in the light to capture photons, and some of their time in the dark to perform carbon assimilation). Therefore, the primary measures of a reactor's performance are based on the sun utilization parameters K_{PAR}. Using the developed relationships, an upper bound for K_{PAR} is given by

$$
K^{\max}_{\text{PAR}(m^2/\text{mol})} = \frac{2.7_{(g/\text{mol})} C_{(m^2/L)}}{m_{\text{dense}(g/L)}} \qquad (10.41)
$$

The instantaneous yield in eqn (10.37) gives the yield for a specific light intensity.

In some scenarios, it is beneficial to measure an overall yield and associated efficiency. For this, the total biomass yield versus total PAR must be calculated. This may be expressed as

$$\begin{aligned} Y^{\text{act}}_{\text{algae}(g)/I\ (\text{mol})} &= \frac{\int_0^t \dot{m}(\tau)_{\text{algae}(g/L/h)} d\tau}{C_{(m^2/L)} \int_0^t I(\tau)_{\text{mol}/m^2/h} d\tau} \left[\frac{g\ \text{algae}}{\text{mol}}\right] \\ &= \frac{m(t)_{\text{algae}(g/L/h)} - m(0)_{\text{algae}(g/L/h)}}{C_{(m^2/L)} \int_0^t I(\tau)_{\text{mol}/m^2/h} d\tau} \left[\frac{g\ \text{algae}}{\text{mol}}\right] \end{aligned} \tag{10.42}$$

where $m(0)_{\text{algae}(g/L/h)}$ and $m(t)_{\text{algae}(g/L/h)}$ are the beginning and ending biomass concentrations, respectively.

In this case, the aggregate PE is given by

$$\eta_{\text{PE}} = \frac{Y^{\text{act}}_{\text{algae}(g)/I\ (\text{mol})}}{Y^{\text{max}}_{\text{algae}(g)/I\ (\text{mol})}} \tag{10.43}$$

10.2.3 Summary

The PE yield in eqn (10.39) provides many insights into PBR design and efficiency trade-offs. At lower light intensities, the respiration is a dominant factor, which is intuitive since at very low light there is more respiration than growth in a PBR. As light intensity increases, the respiration term has less of a relative effect. As the mass increases up to m_{dense}, both the growth term (i.e., the positive term/first term on the right) increases and the amount of respiration (i.e., the second term/subtracted term on the right) increases as well. When the mass increases past the critical density, namely m_{dense}, the respiration term continues to increase while the growth term stays fixed. The net effect of this is that the efficiency decreases as the mass increases above m_{dense}. As the area to volume ratio increases, the effect of respiration is decreased, since there are more microalgae available on the surface of the PBR. On the growth side, the ratio of $K_{\text{PAR}(m^2/\text{mol})}$ to $C_{(m^2/L)}$ determines the amount of liters per mole of light. Equation (10.41) suggests that a larger surface area to volume ($C_{(m^2/L)}$

term) will result in a larger sun utilization ($K_{\mathrm{PAR}(m^2/\mathrm{mol})}$ term); however, increasing the surface to volume ratio may increase the cost to the point where it is no longer economically feasible to operate or build the reactor. Also, PE, as it is measured here, uses the ratio of $K_{\mathrm{PAR}(m^2/\mathrm{mol})}$ to $C_{(m^2/L)}$ as part of its calculation. Therefore, increasing $C_{(m^2/L)}$ by itself will not necessarily result in a more efficient use of PAR. Instead, the ratio of $K_{\mathrm{PAR}(m^2/\mathrm{mol})}$ to $C_{(m^2/L)}$ may be used to evaluate various PBR geometries.

10.3 Model Validation and Verification

The specific reactor used for model validation is shown in Figure 10.1, and a block diagram of the reduced order model is shown in Figure 10.6. This system was operated under the following conditions:

- pH regulated to 7.3 through CO_2 addition,
- Temperature unregulated (varied 20^oC to 25^oC),
- Batch operation (nutrients provided with the initial culture medium only),
- Air sparge rate was held constant and only the CO_2 flow rate was changed,
- The total gas sparge rate was about 40 standard liters per minute (SLPM) at any given time,
- The (fixed flow rate) air stream and (variable flow rate) CO_2 gas stream were mixed before being sparged through the microalge/media mixture,
- Light was the only factor driving growth
- Sensors are read and the model is updated every 5 seconds

There were also a limited number of measurements available. For the validation performed here, the following measurements were available (NB: Figure 10.6 uses the acronyms stated here):

- PAR sensor (used to measure the amount of light available for photosynthesis),
- pH sensor,
- Dissolved Oxygen (DO) sensor,
- Optical Density (OD) sensor (this measurement correlated well with biomass density),
- CO_2 mass flow rate into the reactor, and
- By-hand biomass measurement (taken twice a day - once in the morning and once at night)

Based on the amount of information available, a reduced order model was developed. This is covered in the next section. Based on this reduced order model, model parameters were fit and then verified on reactor data from a Summer growing season (i.e., from May until September).

10.3.1 Reduced Order Model

A diagram of the validation and verification (V&V) setup is shown in Figure 10.6, which was used to create a reduced order model. In this setup, an on-sight PAR sensor was available that was able to measure the rate of PAR photons incident to the reactor. This sensor measured both the direct and diffuse light and provided a good estimate for the amount of PAR that the microalgae were using. Given the experimental systems used, the light model was not verified. Instead, the PAR sensor was used as the light subsystem.

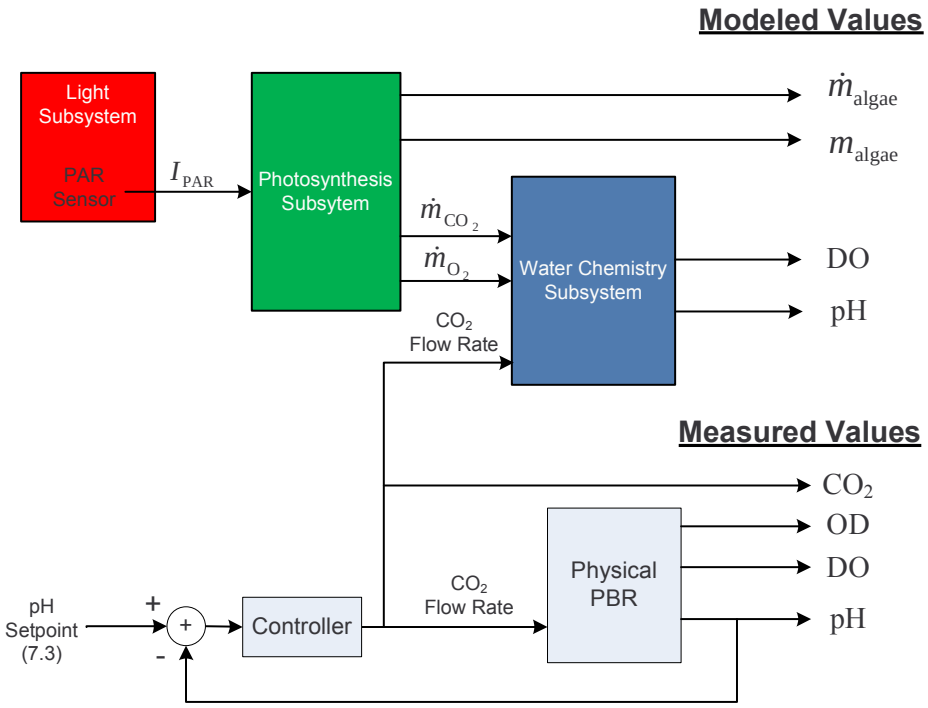

Figure 10.6: Model Validation and Verification Setup

The OD sensor measurement changed proportionally with the density of the biomass. For the results here, this sensor was correlated to dry mass in units g/L. Most of the time during daylight hours, this measurement was well correlated well with biomass density. At times, this sensor was unreliable. To help determine the reliability, by-hand measurements were taken twice daily. If the sensor values at the measurement times did not match the by-hand measurements, the data for that day was determined unreliable and the data was not used for validation.

Since the amount of dissolved CO_2 was not measured, a reduced order model was derived that had CO_2 flow rate as an input and pH as an output. This eliminates the model of total DIC from Section 10.1.3. Instead, it was assumed that regulating pH also kept the dissolved CO_2 in a range appropriate for accelerated growth. Since the operating conditions included continuous sparging and nutrients only being added in the beginning, there is no feedback to the growth model. This assumes that the available nutrients are consistent for each density among batches and that there are

no inhibiting effects from DO build up.

The resulting model equations for each subsystem are provided in the next section along with the methods used to fit parameters for each subsystem model.

10.3.2 Fitting Model Parameters

The PBR model for the test PBR (shown in Figure 10.1) required model parameters for both the growth subsystem and the water chemistry subsystem. The two known inputs were the input air flow rate and the input CO_2 flow rate. For the PBR model developed here, the input air flow rate was about 40 SLPM. This was not well regulated and there were times when this flow rate was not maintained. However, the data set used for validating the water chemistry model had a relatively constant input air flow rate. This flow rate will affect the amount of CO_2 required for pH regulation, since it is the the concentration of CO_2 that will affect the mass transfer between the sparging bubbles and the media. Therefore, the input CO_2 flow rate and input air flow rate were combined to get a total flow rate and CO_2 concentration (between 0 and 1). The models and fitted parameters are outlined in the next sections.

10.3.2.1 Growth Model

The growth model in Section 10.1.2 provided the nonlinear differential equations that described growth, which are stated again here with the appropriate units.

$$\dot{m}_{\text{algae}(g/L/h)} = K_{\text{PAR}(m^2/\text{mol})} I_{\text{PAR}(\text{mol}/m^2/h)} \bar{m}_{\text{algae}(g/L)} - R_{(1/h)} m_{\text{algae}(g/L)} \quad (10.44)$$

$$\dot{m}_{\text{CO}_2(\text{SLPM})} = \frac{K_{\text{CO}_2}}{\eta_{\text{CO}_2}} \dot{m}_{\text{algae}(g/L/h)} \quad (10.45)$$

$$\dot{m}_{\text{O}_2(mg/L/h)} = K_{\text{O}_2} \dot{m}_{\text{algae}(g/L/h)} \quad (10.46)$$

The rate of respiration in the dark (R) is a fundamental parameter of the microalgae used and may be measured during the night time when the microalgae are only

respirating. In particular, this value was measured by examining the rate of respiration over multiple nights throughout a Summer growing season. This was calculated by taking two points during the night and assuming exponential decay. For example, if $m_{\text{algae}}(0)$ is the dry mass at 11pm at night (assuming that sunset is well before 11pm) and $m_{\text{algae}}(5)$ is the dry mass at 4am the next day (assuming that sunrise is well after 4am), then the following models exponential decay.

$$m_{\text{algae}}(5) = m_{\text{algae}}(0)e^{-5R} \tag{10.47}$$

This method was repeated for various sampling times and over many days. For the growth model validation done here, this value is $R = 0.0045$, which is in normalized units.

The light driven growth portion of this model uses the PAR measurement, which is given by $I_{\text{PAR}(\text{mol}/m^2/h)}$, the sun utilization parameter $K_{\text{PAR}(m^2/\text{mol})}$, and the critical density parameter $m_{\text{dense}(g/L)}$ to determine the light driven growth rate. For the presentation of the model here, these values are normalized. Based on the data collected, nominal values of $K_{\text{PAR}} = 0.01$ (area/incident light) and $m_{\text{dense}} = 1$ were used. The sun utilization parameter (K_{PAR}) is assumed to be specific to a strain of microalgae. This parameter will be affected by external factors such as nutrient availability and general health of the microalgae. Since the operating conditions are kept the same for each batch grown (e.g., no nutrient depletion occurs in the growth phase), this parameter is held constant. An example plot of the modeled and measured growth is shown in Figure 10.7.

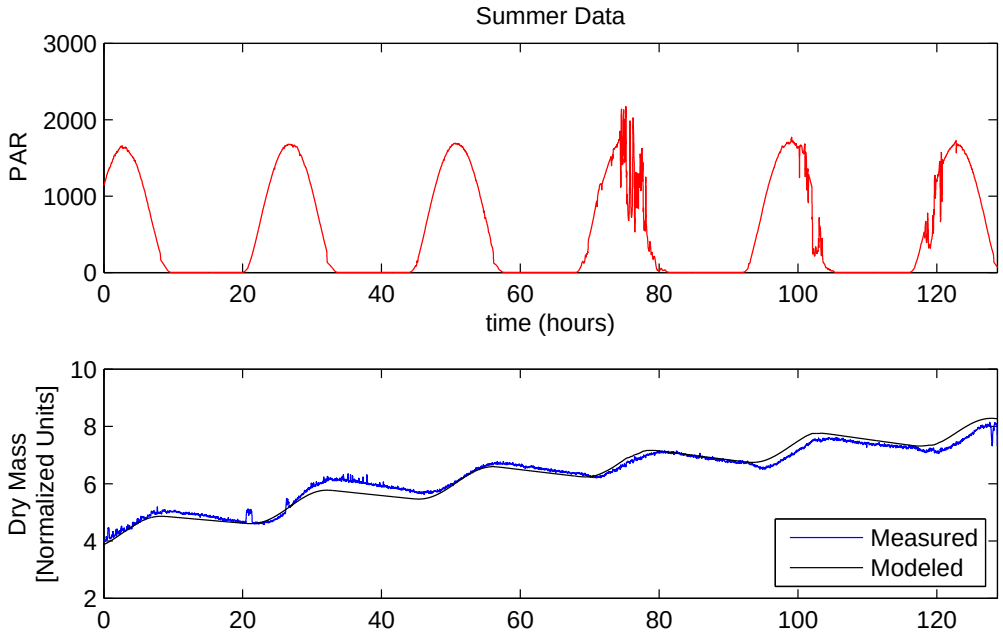

Figure 10.7: Growth Model Verification

10.3.2.2 Water Chemistry Subsystem

The water chemistry subsystem considered here models the DO and pH dynamics of the PBR system shown in Figure 10.1. The DO model follows from eqns (10.27) and (10.28). Here, the rates are in (mass/volume/time) and O_2 production rate from photosynthesis, namely $\dot{m}_{O_2(\text{mass/volume/time})}$, comes from the growth model described in the previous section. A line search algorithm was used to determine the lumped parameter of $\alpha_{\text{DO}} = 0.357$. A simulation with the fitted parameters is given in Figure 10.8.

In Figure 10.8, the top graph shows both the DO sensor and the modeled output. For reference, the input to the DO model (i.e., the DO production rate from the growth model) is provided. This bottom graph illustrates one of the challenges to growing microalgae, which is the potential problem of DO build up. During peak growing hours, DO is being produced at rates up to 7 (mass/volume/time), which has the potential to quickly raise the level of DO to a point where growth will cease. Even

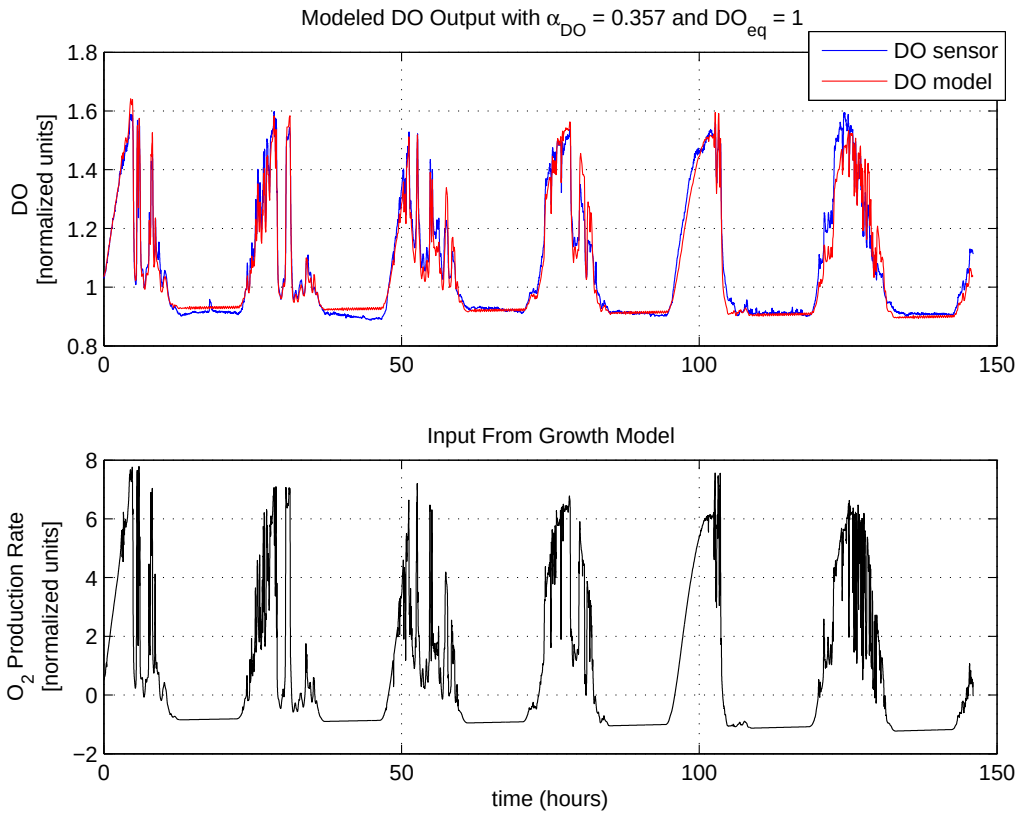

Figure 10.8: DO Model Verification

with continuous sparging, the DO levels build up to 1.5 times the normal equilibrium with air; however, the microalgae are still able to grow in this environment without many inhibitory effects.

In order to see the first order dynamics of the DO model, a zoomed in version of Figure 10.8 is shown in Figure 10.9. By examining the DO model and sensor in the top plot, the DO data is a slightly smoothed (and scaled) version of the modeled O_2 production from growth. For a particular example, examine the two graphs between hours 52 and 53 in Figure 10.9. Here, there are two distinct peaks in production rate (bottom graph in Figure 10.9) that appear as one smoothed peak to the sensor and model (top graph in Figure 10.9). This is a result of the mass transfer dynamics between the bubbles and the media.

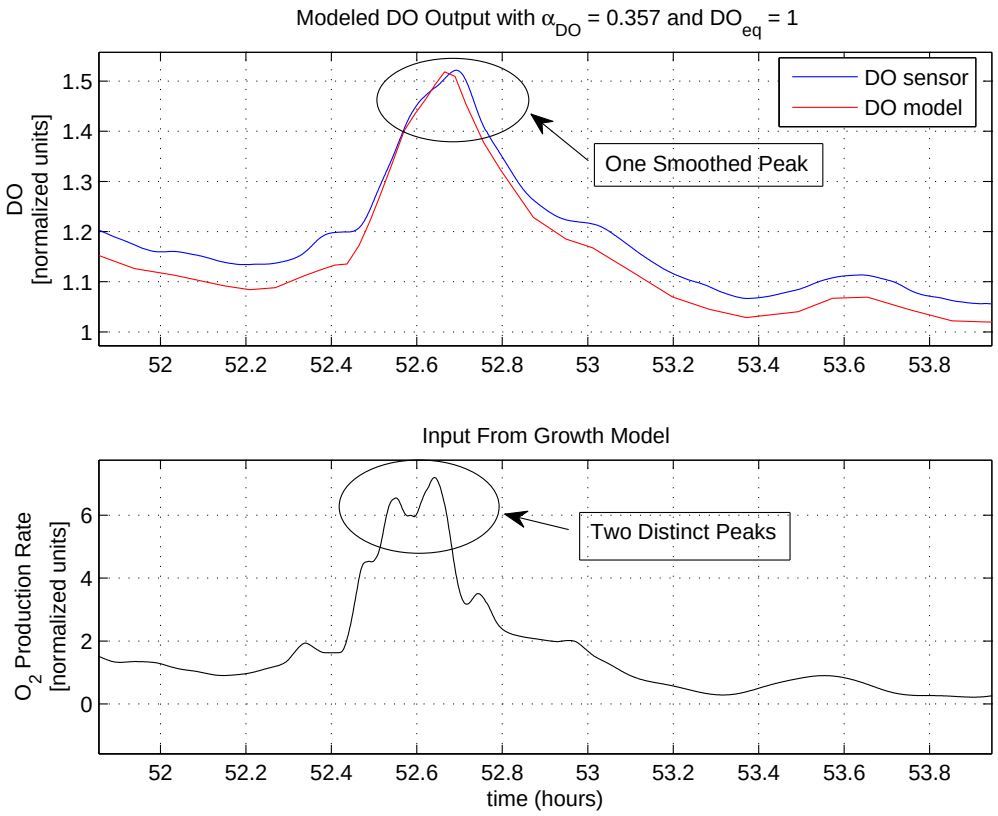

Figure 10.9: DO Model Verification (Zoomed In)

The pH model was created by combining the DIC and pH models from Section 10.1.3 into the first order model

$$\dot{y}_{\mathrm{pH}}(t) = \alpha_{\mathrm{pH}} w_{\mathrm{sparge}} (K_{\mathrm{pH}} x_{\mathrm{CO_2}}(t - \tau_{\mathrm{d,gas}}) - y_{\mathrm{pH}}(t)) - \dot{m}_{\mathrm{CO_2}}(t) \tag{10.48}$$

$$= \frac{w_{\mathrm{sparge}}}{\tau_{\mathrm{pH}}} (K_{\mathrm{pH}} x_{\mathrm{CO_2}}(t - \tau_{\mathrm{d,gas}}) - y_{\mathrm{pH}}(t)) - \dot{m}_{\mathrm{CO_2}}(t) \tag{10.49}$$

Due to the nature of PBR operation over the time the data was collected, the parameters measured for the pH model were inconsistent. Unlike the DO model, the model parameters were not well isolated. In the DO model, all of the DO produced came directly from photosynthesis and its only method for removal was sparging, which was held constant. Also, the oxygen content in the sparging gas was roughly constant during the operation, which meant the rate of oxygen mass transfer did not

change during operation. For the pH model, pH was affected by both photosynthesis and the concentration of CO_2 in the sparging gas stream. This extra degree of freedom caused much variation in the possible model parameters. As a part of ongoing research, more data and experiments are needed to develop a more accurate pH model. This issue is discussed in Chapter 12. In order to develop a simple model, it was assumed that the rate of gas exchange for CO_2 was similar to the rate of gas exchange O_2.

Chapter 11

Advanced Microalgae Control

For the PBR models and controllers presented here, the only commanded input is CO_2 flowrate, which affects the CO_2 concentration in the input sparging gas stream. It is well understood that the dissolved CO_2 concentration in media may be sensed via pH, which was describe in detail in the the previous chapter. Therefore, the control objective is to maintain the appropriate dissolved CO_2 concentration by regulating the pH of the media. Proper pH regulation via input CO_2 gas has been shown to have a large impact on microalgae production [23; 24; 25], and in some cases, can improve the biomass yield by up to 500%. In a larger PBR configuration, there may exist long transport delays, which will affect the achievable pH regulation. Through the use of the our developed controller architectures, we can characterize the achievable pH regulation that may be attained for a specified PBR configuration.

In the previous chapter, a growth model was developed that can provide an estimate of the CO_2 being consumed (or produced) by the microalgae during photosynthesis (or respiration). In our work [22], we presented a method for using this CO_2 growth model as a feedforward CO_2 control input to help improve pH regulation. In this scenario, the model was used in two configurations, namely

1. An open-loop model that estimates CO_2 consumption (and production) based on incident PAR.

2. An observer corrected model that used measured dry mass and incident PAR

to estimate CO_2 consumption (and production).

In the first part of this chapter, an observer-based growth model is presented in both continuous-time and discrete-time, with the latter being more suitable for implementation on a microcontroller. On a physical PBR, this observer could be used to improve the feedforward controller performance. In the second part to this chapter, pH regulation for various PBR configurations is presented. For this, a pH model is developed that may be used for controller synthesis and validation. Then, a robust feedback controller is synthesized, which characterizes a base level of performance that may be achieved using only feedback control. One of the biggest factors that limits performance is the (potentially) long CO_2 transport delay that results from a commercial size PBR setup. On commercial size PBR, the CO_2 supply may be located away from the actual PBR (e.g., the CO_2 source could be a power plant and the PBR with microalgae could be located a few hundred meters away from the power plant). This physical distance will create a transport delay that is a non-minimum phase component in this system. Even though this system contains nonlinear components, a version of the previously developed DFFPC architecture may be used to characterize the achievable pH regulation performance for a given transport delay. These two controllers (i.e., feedback only and DFFPC) are presented in simulation for various transport delays and their performance compared. Using the DFFPC architecture, the achievable pH regulation performance is characterized. This example also demonstrates how the DFFPC architecture could be extended to provide perfect tracking on an example nonlinear system with a time delay.

11.1 Observer Based Growth Model

To begin, let's consider the open-loop PBR growth model (from the previous chapter):

$$\dot{\hat{m}}_{\text{algae}} = K_{\text{PAR}} I_{\text{PAR}} \bar{\hat{m}}_{\text{algae}} - R\hat{m}_{\text{algae}}, \tag{11.1}$$

where $\hat{m}_{\text{algae}}$ is the modeled dry mass, $\bar{\hat{m}}_{\text{algae}}$ is the saturated mass used to model linear growth above a critical density, and $\dot{\hat{m}}_{\text{algae}}$ is the rate of microalgae growth as a function of incident light I_{PAR}. This model captures both growth from photosynthesis (when I_{PAR} is large enough) and respiration (which is always occurring). In this representation, the "hat" notation is used to distinguish that $\hat{m}_{\text{algae}}$ is a modeled (and not measured) value. Signals that are measured will not contain a "hat" (e.g., m_{algae} is a measurement of the actual dry mass). When there is a measurement of m_{algae} available, it may be used to improve the estimate of $\dot{\hat{m}}_{\text{algae}}$ by creating an observer (for more information, see Section 2.10). For the growth model considered here, the equations may be augmented to create an observer as follows:

$$\dot{\hat{m}}_{\text{algae}} = K_{\text{PAR}} I_{\text{PAR}} \bar{\hat{m}}_{\text{algae}} - R\hat{m}_{\text{algae}} + L(\hat{m}_{\text{algae}} - m_{\text{algae}}). \tag{11.2}$$

Here, m_{algae} represents the measured dry mass and $\hat{m}_{\text{algae}}$ is the modeled dry mass. Using these values, the growth rate $\dot{\hat{m}}_{\text{algae}}$ is corrected via an observer gain L. A Simulink diagram that implements eqn (11.2) is shown in Figure 11.1. This diagram also contains the components necessary to determine the CO_2 consumption and O_2 production rates. For the application considered here, the CO_2 consumption rate will be used to determine the feedforward control input used for pH regulation.

For digital implementations, the continuous time observer based growth model is converted to discrete-time. For this first order model, a (first-order) Backwards-Euler numerical integration method is used to convert the continuous-time system to a discrete-time system, which works well for this case (assuming the sampling time is sufficiently fast). For the simulations considered here, the sampling period is 5 seconds, which is significantly faster than the dynamics of the system (that are on the order of minutes). The discrete-time growth model is:

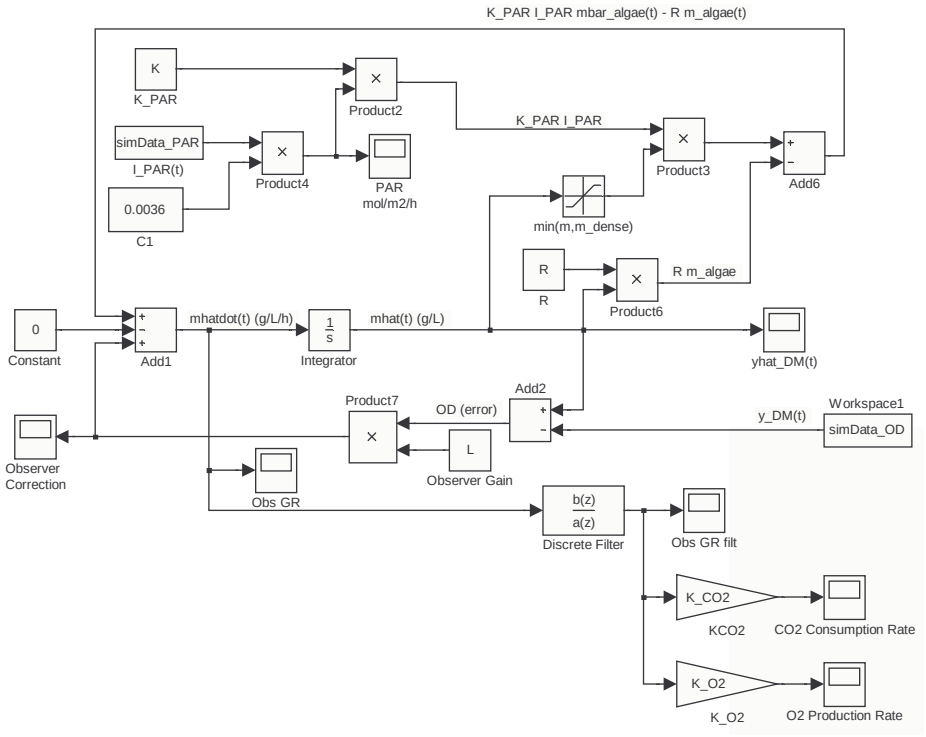

Figure 11.1: Continuous-time Nonlinear Observer Growth Model in Simulink.

$$\frac{\hat{m}_{\text{algae}}[k+1] - \hat{m}_{\text{algae}}[k]}{T} = K_{\text{PAR}} I_{\text{PAR}}[k] \bar{\hat{m}}_{\text{algae}}[k] - R\hat{m}_{\text{algae}}[k] + L(\hat{y}_{\text{DM}}[k] - y_{\text{DM}}[k]) \tag{11.3}$$

Simplifying equation (11.3) results in the following difference equation

$$\hat{m}_{\text{algae}}[k+1] = (TK_{\text{PAR}} I_{\text{PAR}}[k]) \bar{\hat{m}}_{\text{algae}}[k] + (1 - TR)\hat{m}_{\text{algae}}[k] + TL(\hat{y}_{\text{DM}}[k] - y_{\text{DM}}[k]) \tag{11.4}$$

A Simulink diagram of this is given in Figure 11.2.

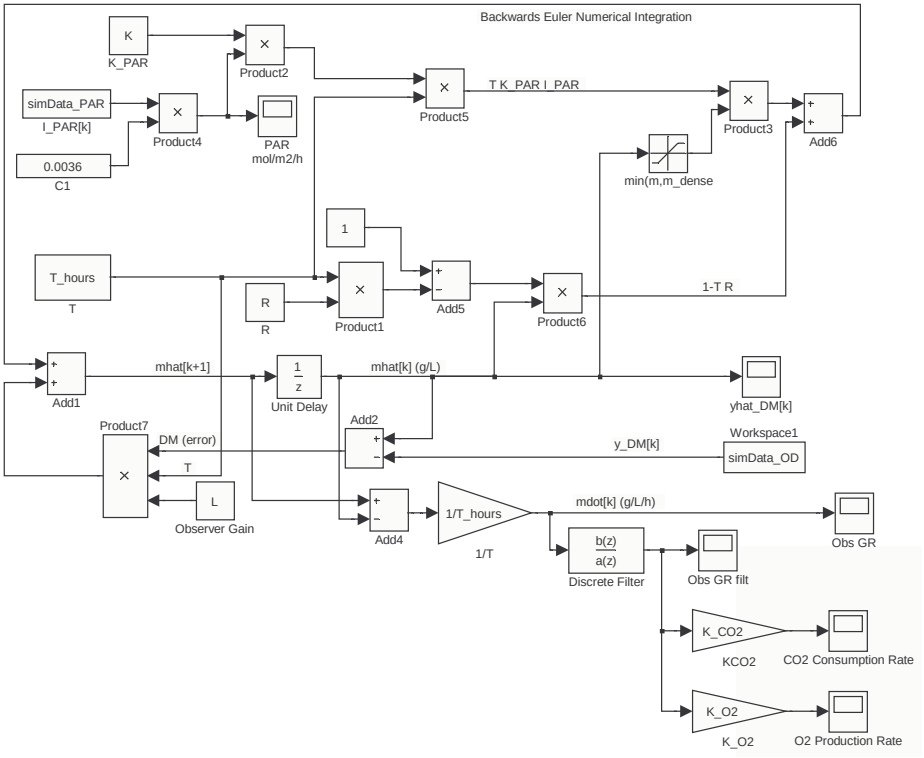

Figure 11.2: Discrete-time Numerical Integration of the Nonlinear Observer Growth Model in Simulink.

For both observer implementations (shown in Figures 11.1 and 11.2), the open-loop growth model may be recovered by setting the observer gain $L = 0$.

11.2 pH Model

In this section, a pH model is developed for the flat panel PBR (from the previous chapter) that may be used for controller development. The pH of the media is primarily a function of the amount of dissolved CO_2 in the media. The two main factors that affect the dissolved CO_2 concentration in the media are the CO_2 concentration in the sparging bubbles (which use mass transfer to affect the amount of dissolved CO_2 in the media), and dissolved CO_2 consumption and production by the microalgae (during photosynthesis and respiration). Therefore, the following two interactions will be modeled:

1. Dynamics between commanded CO_2 flow rate in sparge gas and the sensed pH in the media.

2. pH change due to microalgae activity (i.e., photosynthesis and respiration).

These interactions are developed in the next two sections.

11.2.1 Water Chemistry pH Model

For the water chemistry part of the pH model, an input CO_2 flow rate is commanded (e.g., by a mass flow controller) that affects the measured pH. This commanded flow rate changes the CO_2 gas concentration upstream from the PBR, which means that there is a transportation delay between when a CO_2 flow rate is commanded and when it reaches the PBR and is sensed via pH. When the commanded CO_2 reaches the PBR, there are diffusion dynamics between the media and sparging bubbles. These dynamics may be modeled as a first order lag, whose time constant is on the order of minutes[1]. There are also dynamics as the dissolved CO_2 reaches equilibrium in the media and as pH is detected by a sensor. Both of these dynamics may be modeled as first order lag, whose time constants are on the order of few seconds. These dynamics collectively modeled by a first order lag that dominated by the diffusion dynamics, along with a time delay associated with the transportation delay.

These modeled LTI dynamics are only valid around an operating point. This is shown in Figure 11.3, where the linearization offsets define the operating point. For the hypothetical example that will be used in this chapter, let a constant flow rate of 4 SLPM (standard liters per minute) of CO_2 correspond to a media pH of 7.3. As the commanded CO_2 flow rate varies around 4 SLPM, the model predicts that the pH will vary with first order plus time delay dynamics around a pH of 7.3. For

[1] A similar phenomenon was seen in the previous chapter where the time constant for diffusion of O_2 into the sparging bubbles was $\tau_{DO} = 0.07$ hours (or 4 minutes and 12 seconds).

the PBR model considered here, the CO_2 flow rate will be able to vary from 0 to 10 SLPM and the pH will vary from 8.5 (for a CO_2 flow rate of 0 SLPM) to a pH of 5.5 (for a CO_2 flow rate of 10 SLPM). In practice, the pH scale is defined to be between 0 and 14, which puts fundamental limits on the output. Also, it is not possible to command a negative CO_2 flow rate, which means that it is not possible to raise the pH above 8.5. Also, from experience on the physical PBR, it is difficult to lower the pH of the media below 5 using a CO_2 rich input gas stream. In general, the actual pH model becomes nonlinear as the CO_2 flow rate deviates much from its operating point of 4 SLPM; however, the dynamics are approximately linear of the operating range considered here. This is an example of modeling a nonlinear process as a linear process around an operating point. For more information on using linearization to model a nonlinear system, see [76].

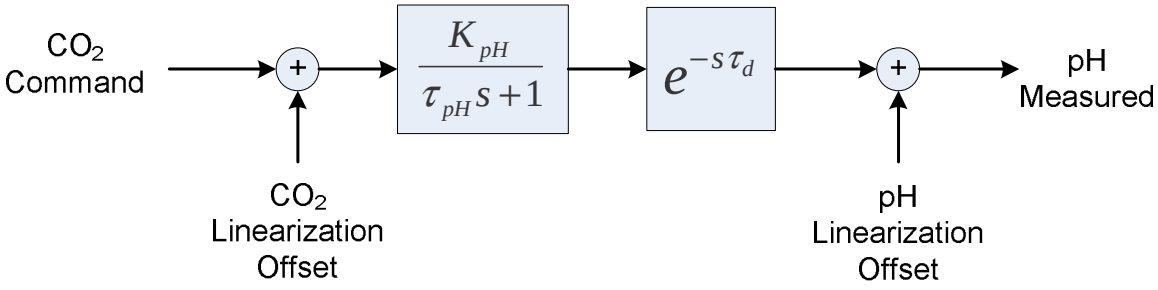

Figure 11.3: Water Chemistry pH Model.

This model takes into account the interaction between the sparge gas and media. When there are microalgae in the PBR, there is another interaction between the microalgae and media, which is discussed next.

11.2.2 PBR pH Model

As the microalgae consume and produce CO_2, they will interact with the media to change the pH. Microalgae will both photosynthesize (consume CO_2) and respirate (produce CO_2), which will affect the CO_2 concentration in the surrounding media. Photosynthesis will dominate during the daylight hours and respiration will dominate

in the dark. These phenomenon are captured by the previously developed growth model. For the pH model, it is assumed that the change in pH will be proportional to the change in CO_2 concentration. Therefore, a scaled version of the CO_2 consumption term from the growth model is added to the final pH value. This is illustrated in Figure 11.4. In this figure, $C_{\mathrm{pH,CO_2}}$ is the conversion between CO_2 consumed and increase in pH. Since this change in pH happens within a few seconds of photosynthesis or respiration, it does not require the transportation delay or diffusion dynamics.

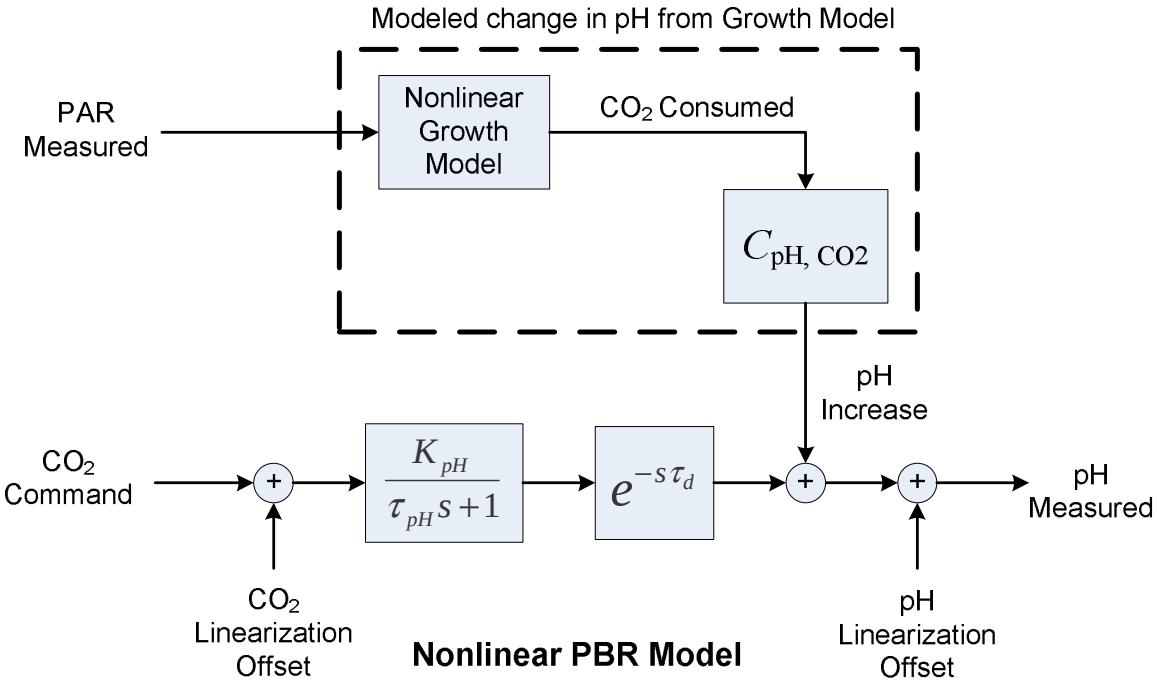

Figure 11.4: PBR pH Model with Growth and Water Chemistry Dynamics.

For the pH model considered here, the model parameters are set to:

$$K_{\mathrm{pH}} = -0.3 \quad [\mathrm{CO_2 \text{ flow units to pH}}] \tag{11.5}$$

$$\tau_{\mathrm{pH}} = 0.03 \quad [\mathrm{hours}] \tag{11.6}$$

$$\tau_d = 0.05 \quad [\mathrm{hours}] \tag{11.7}$$

$$C_{\mathrm{pH,CO_2}} = 5 \quad [\mathrm{CO_2 \text{ consumption units to pH}}] \tag{11.8}$$

The parameter K_{pH} was chosen such that the pH could vary from 5.5 to 8.5 as the CO_2 flow rate varies from 0 to 10 SLPM (assuming that the linearization point is 4 SLPM regulates the media to a pH of 7.3). The time constant and process delay were chosen based on estimates of the model parameters from experimental data; however, these parameters have not been validated. The conversion C_{pH,CO_2} from CO_2 consumption to change in pH was assigned for simulation purposes and has not been validated.

The growth model parameters were set to arbitrary values for illustrative purposes. These values are:

$$K_{\text{PAR}} = 0.01 \quad [m^2/\text{mol}] \tag{11.9}$$

$$R = 0.0045 \quad [1/\text{hours}] \tag{11.10}$$

$$m_{\text{dense}} = 1 \quad [g/L] \tag{11.11}$$

$$K_{CO_2} = 1 \quad [-] \tag{11.12}$$

Here, K_{CO_2} is absorbed into C_{pH,CO_2} (i.e., C_{pH,CO_2} is really being used to convert from microalgae growth rate to change in pH, since a true K_{CO_2} is not known. However, if a K_{CO_2} were known, then C_{pH,CO_2} would be the conversion from CO_2 consumed to change in pH). These parameters are used in the remaining sections of this chapter.

11.3 pH Regulation using Feedback Controllers

For the feedback controller design, a robust controller is designed based on the linear part of the diagram in Figure 11.3 (i.e., the linearization offsets are not used in the controller design). Therefore, the LTI plant used for controller synthesis is

$$G_{\text{pH}}(s) = \frac{K_{\text{pH}}}{\tau_{\text{pH}} s + 1} e^{-s\tau_d}, \tag{11.13}$$

where K_{pH}, τ_{pH}, and τ_d were defined in eqns (11.5)-(11.7). For the purpose of feedback controller design, the pH change due to microalgae growth (i.e., photosynthesis and respiration) is a disturbance on the plant output that the feedback controller must reject. Therefore, the robust feedback controller interconnect for disturbance rejection provided in Chapter 4 (see Figure 4.2) will be used. For this robust controller design, the following weights were chosen:

$$W_i(s) = \frac{0.1(s+20)}{(s+200)} \quad \text{(Additive Uncertainty Weight)} \tag{11.14}$$

$$W_d(s) = \frac{10}{(s+20)} \tag{11.15}$$

$$W_n(s) = 0.0001 \tag{11.16}$$

$$W_p(s) = \frac{10}{(s+0.0001)} \tag{11.17}$$

$$W_u(s) = 0.01 \tag{11.18}$$

$$\tag{11.19}$$

Using similar methods to those in Chapter 9, the final controller was designed to have integral action. The resulting feedback controller is given by the following.

$$K(s) = \frac{-50.2104(s+200)(s+40)(s+20.83)(s+14.29)}{s(s+81.03)(s+19.99)(s^2+59.43s+1849)} \tag{11.20}$$

For simulation, the Simulink block diagram shown in Figure 11.5 is used.

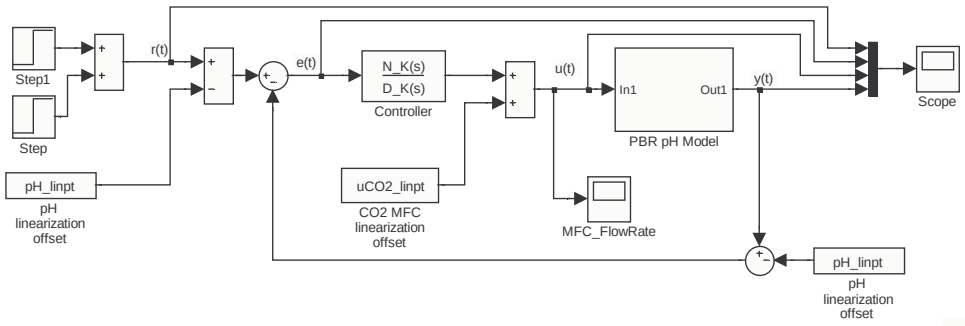

Figure 11.5: Feedback Only Microalgae pH Regulation Simulink Diagram

In Figure 11.5, the block labeled "PBR pH Model" is a Simulink block diagram of the full pH model shown in Figure 11.4. For a demonstration of the designed feedback controller, an example is created that regulates pH to 7.3 during the night (i.e., between 8pm and 4am) and a pH of 7 during the day (i.e., between 4am and 8pm). An actual data set of PAR is used to drive the growth model. The simulation (over a 24 hour period) is shown in Figure 11.6.

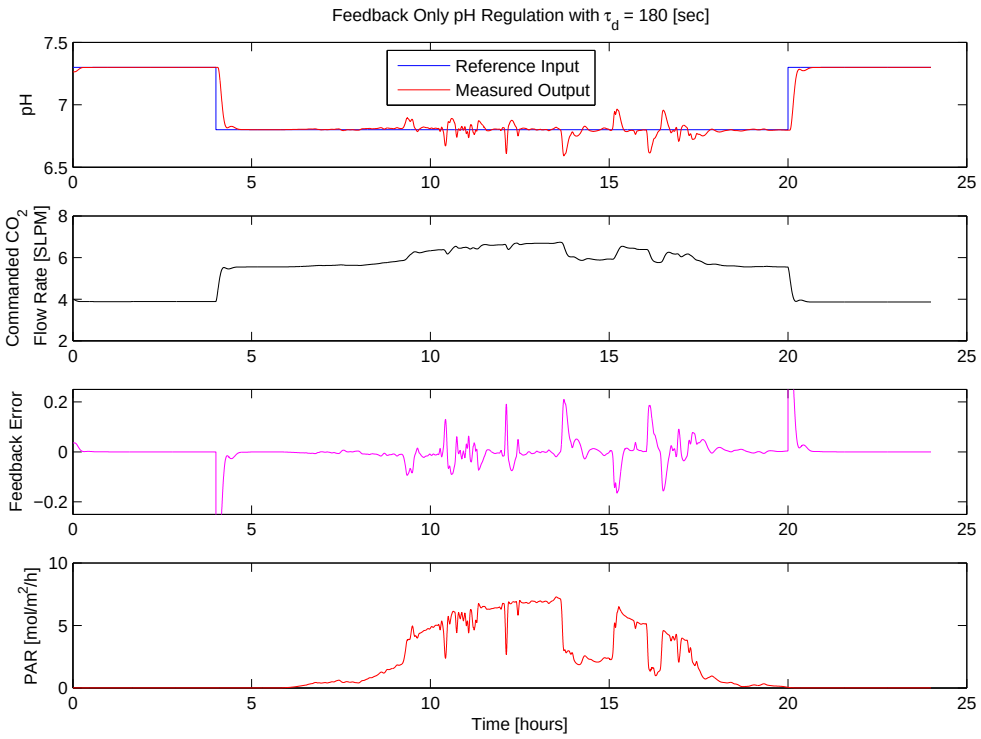

Figure 11.6: Simulated Feedback Only Microalgae pH Regulation

In Figure 11.6, it may be seen that the feedback controller is able to regulate pH to +0.2089 to -0.1652 around the pH setpoint of 7 during the high growth period (i.e., when the sun is out and measured PAR is more intense). For this example, the high growth period is considered to be between hours 9 and 17 in Figure 11.6. This provides a baseline for comparing the DFFPC architecture presented next.

11.4 pH Regulation using a DFFPC Architecture with Growth Compensation

If the pH model did not include the nonlinear growth model, the extension of pH regulation to a DFFPC architecture would be straight forward. The only modification would be to include the linearization offsets inside the feedback loop. However, the nonlinear growth model dynamics provide another non-minimum phase component. To see this, note that as the microalgae grow, the pH changes instantaneously; however, the controller must compensate for this change in pH through the transport delay τ_d. In the DFFPC architecture, this instantaneous change in pH due to growth may be captured in $r_{ff}(t)$ by adding it to the output of $P_{\text{des}}(s)G_{noi}(s)$. This will restore the perfect tracking of pH in the sense that the feedback error will be zero; however, only changing $r_{ff}(t)$ means that the measured pH will vary with photosynthesis and no compensation will be made by the second feedforward controller ($FF2(s)$) to correct for the amount of CO_2 (sensed via pH) that is being removed from the media during growth. In other words,

$$y(t) = r_{ff}(t) = r(t) + \text{“pH change due to growth”}. \tag{11.21}$$

This violates one of the design requirements which states that $r_{ff}(t)$ must asymptotically track $r(t)$ (and not $r(t)$ + “some non-zero offset”). This is addressed by subtracting the “pH change due to growth” from $r(t)$ before it goes into the two feedforward controllers. This will (asymptotically) cancel out the addition of the “pH change due to growth” that has been added to $r_{ff}(t)$. A simulation diagram of a modified DFFPC architecture that may provide perfect pH tracking is shown in Figure 11.7.

In Figure 11.7, the compensation of growth in the reference signals is labeled “Growth Compensation”. For this example, the plant factorization is

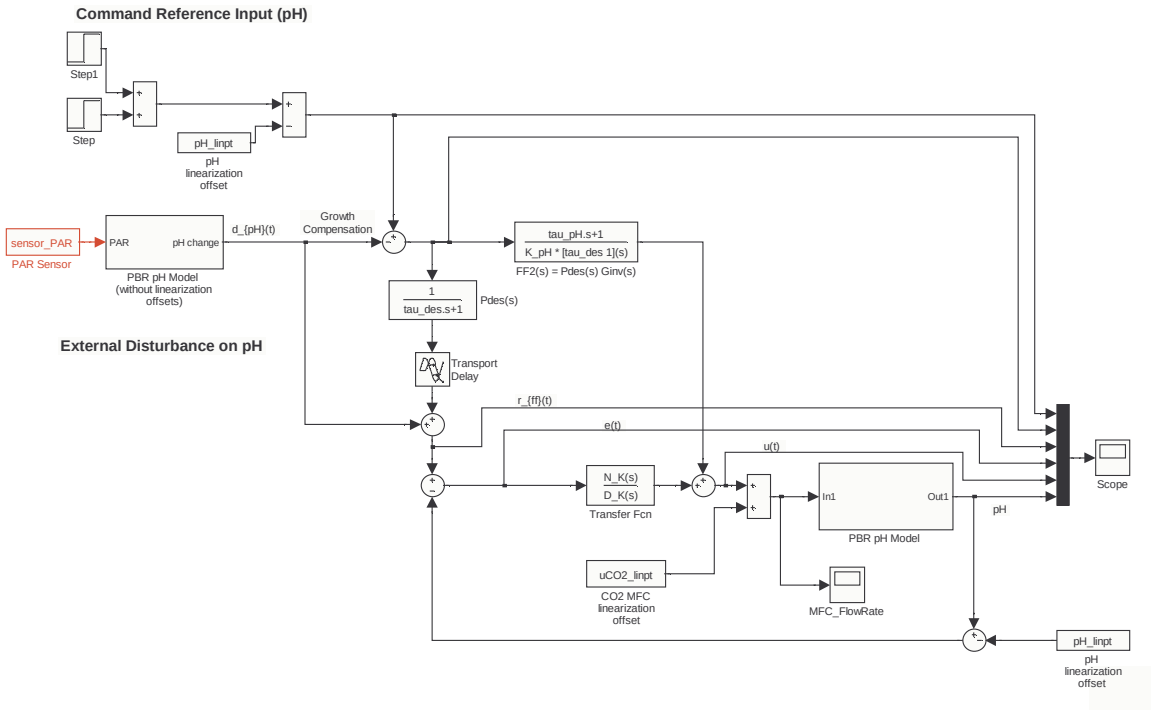

Figure 11.7: Microalgae pH Regulation using a Modified DFFPC Architecture Simulink Diagram

$$G_i(s) = \frac{K_{\text{pH}}}{\tau_{\text{pH}} s + 1} \tag{11.22}$$

$$G_{noi}(s) = e^{-s\tau_d}. \tag{11.23}$$

Let $d_{\text{pH}}(t)$ be the disturbance on pH that results from growth. Now, there are two inputs and one output, which means that there are two closed-loop transfer functions, namely one from reference input $r(t)$ to output $y(t)$ and one from disturbance $d_{\text{pH}}(t)$ to output $y(t)$. When there is no growth (i.e., $d_{\text{pH}}(t) = 0$), then the nominal closed-loop response from $r(t)$ to $y(t)$ is the expected one, namely

$$M_{r\to y}(s) = \frac{Y(s)}{R(s)} = P_{\text{des}}(s) G_{noi}(s). \tag{11.24}$$

However, the nominal closed-loop response from $d_{\text{pH}}(t)$ to $y(t)$ is

$$M_{d_{\mathrm{pH}}\rightarrow y}(s) = \frac{D_{\mathrm{pH}}(s)}{R(s)} = P_{\mathrm{des}}(s)(1 - e^{-s\tau_d}) = P_{\mathrm{des}}(s)(1 - G_{noi}(s)). \tag{11.25}$$

This piece has no counterpart in the DFFPC presented in Chapter 4, but this modification is required to get the desired tracking result. The overall closed-loop expression (that includes the effects of both the reference input and disturbance input) is

$$\begin{aligned} Y(s) &= M_{r\rightarrow y}(s)R(s) + M_{d_{\mathrm{pH}}\rightarrow y}(s)D_{\mathrm{pH}}(s) \\ &= P_{\mathrm{des}}(s)G_{noi}(s)R(s) + P_{\mathrm{des}}(s)(1 - G_{noi}(s))D_{\mathrm{pH}}(s) \\ &= P_{\mathrm{des}}(s)G_{noi}(s)R(s) + P_{\mathrm{des}}(s)(1 - e^{-s\tau_d})D_{\mathrm{pH}}(s). \end{aligned} \tag{11.26}$$

For demonstration purposes, the same set of PAR data from the previous section is used to demonstrate the performance improvement in the nominal case. For the simulation here,

$$P_{\mathrm{des}}(s) = \frac{1}{(\frac{s}{\alpha_{\mathrm{des}}} + 1)}, \tag{11.27}$$

where α_{des} is the bandwidth in rad/sec. An example simulation with $\alpha_{\mathrm{des}} = 20$ is shown in Figure 11.8.

In Figure 11.8, the ability to track the reference changes has been improved by $P_{\mathrm{des}}(s)$ and the amount of variation in the pH is +0.1981 to -0.1506 around the pH setpoint of 7 during the high growth period, which is about a 5% improvement over the feedback controller presented in the previous section. The advantage to this method over the feedback only case is improved performance (i.e., less total variation). Also, the amount of recovery time is less (i.e., faster recovery) with the DFFPC architecture. This may be seen by looking at the feedback error plots in Figures 11.6 and 11.8. With the DFFPC architecture, the actual pH that may be perfectly tracked (for this choice of $P_{\mathrm{des}}(s)$) will vary by at least +0.1981 to -0.1506

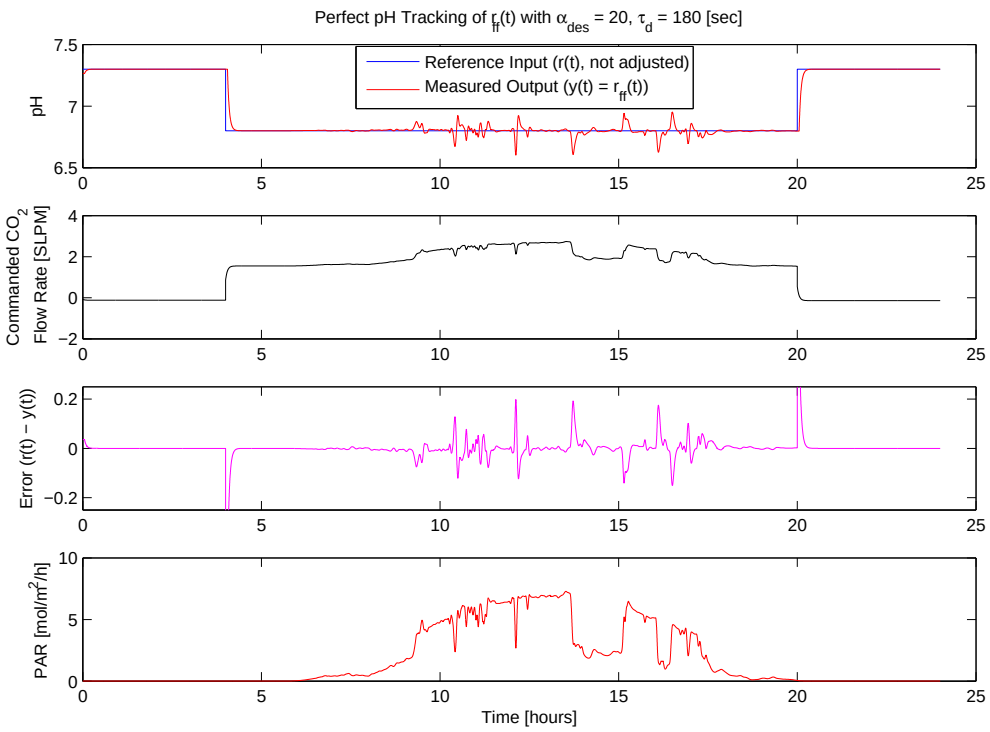

Figure 11.8: Simulated Microalgae pH Regulation using a Modified DFFPC Architecture

(around the pH setpoint). This is a fundamental limitation on how much the pH will vary. If this is unacceptable, then structural changes will need to be made. For example, the time delay could be reduced by putting more CO_2 actuators that are closer to the pH sensors. As an example, let the PBR be configured such that the time delay has been reduced by a factor of ten (i.e., the time delay becomes 0.005 hours or 18 seconds). A simulation of this is shown in Figure 11.9.

The simulation in Figure 11.9 shows that the pH variation due to growth is now +0.1418 to -0.0957 around the pH setpoint of 7 during the high growth period. This is an example of control-structure interaction (CSI) that may be used to evaluate the different PBR design trade-offs. The advantage that the DFFPC architecture provides over the feedback design is that it characterizes the achievable performance analytically. This alleviates the potential need to redesign a feedback controller for

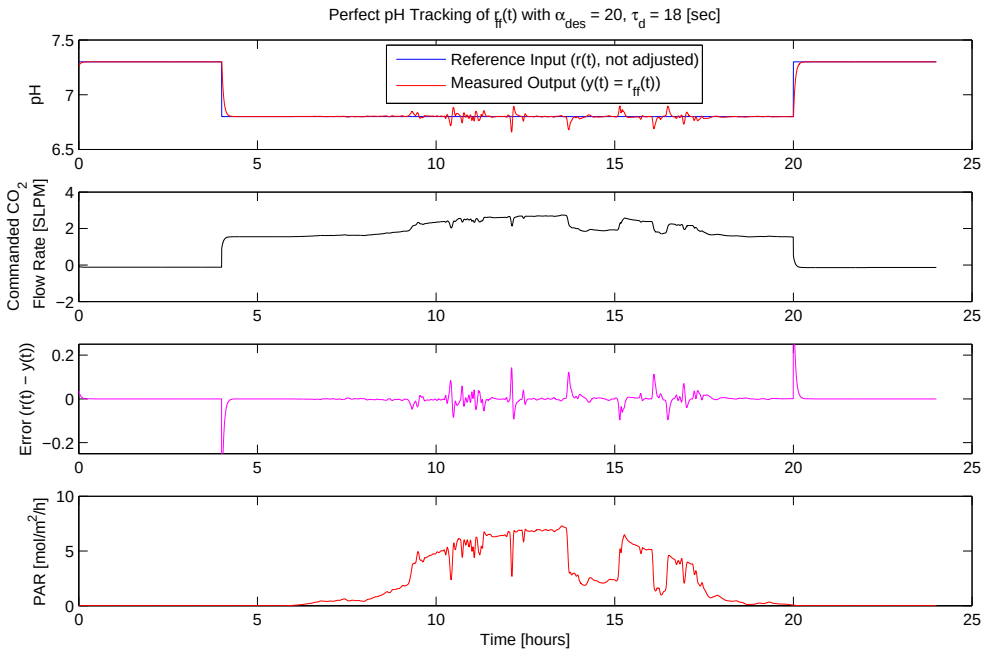

Figure 11.9: Simulated Microalgae pH Regulation using a Modified DFFPC Architecture with Smaller Transport Delay

each configuration, which would introduce another variation to what the expected performance will be for a given PBR setup. The idea of using the DFFPC and DFFSP architectures for CSI is a potential future direction and will be mentioned again in the next chapter.

11.5 Summary

In this chapter, a nonlinear pH model was developed that was suitable for controller development. Using this nonlinear model, both feedback and DFFPC controllers were developed for pH regulation, and their performance measured in simulation. The DFFPC architecture is able to characterize the expected pH regulation that may be achieved for a given process delay (and choice of $P_{\text{des}}(s)$). Using this characterization, the trade-offs between process delay and pH variation may be studied.

For this particular application, it may be observed that the CO_2 regulation nat-

urally follows the feedforward growth model. This was true even in the feedback only case, where the feedback controller was reacting to the changes in pH. Even in this case, the commanded CO_2 flowrate mimics the PAR driven growth rate during high growth periods. This motivates the use of feedforward control on this system, where the feedforward control input is well modeled by the nonlinear growth model. In the case of DFFPC, the second feedforward controller ($FF2(s)$) is providing the additional CO_2 flowrate based on current consumption (from the nonlinear growth model), and the feedback controller is correcting for small errors between the actual pH and the pH that the first feedforward controller ($FF1(s)$) is predicting.

Chapter 12

Conclusions and Future Directions

The goal of the research was two-fold. In the first part, perfect tracking was explored for linear time-invariant (LTI) systems (of the form provided by eqn (3.1) in Section 3.1), which includes stable, unstable, minimum-phase, and non-minimum phase systems, potentially with time delays. A class of signals that may be perfectly tracked was defined for these systems, along with two architectures that may achieve perfect tracking. Previously, this class of signals for non-minimum phase systems had not been fully explored. In the DFFPC architecture, perfect tracking is achieved for both stable and unstable systems; however, the DFFSP architecture is only suitable for stable systems. This fundamental limitation is due to the cancelation properties of the Smith predictor. This limitation was exposed in Section 2.9.

The performance of these two architectures is dependent upon the quality of the models used in the feedforward controllers. To address this, robustness tools and adaptation techniques were developed that may be used to guarantee robust performance and improve performance, respectively. It was shown in Chapter 9 that the robustness tools accurately predict the expected level of performance that may be achieved for a given design (i.e., for a specific $P_{\text{des}}(s)$). In the cases where there were modeling errors, adaptation was used to improve the models used in the feedforward controllers. In the cases where the plant was LTI, but was not correctly modeled, perfect tracking was achieved following adaption via system identification. Since this

adaptation occurs only in the feedforward controllers, guaranteeing stability is numerically much less computationally intensive than adapting the feedback components. In fact, the method for adaptation via system identification (for the DFFPC architecture) can never result in unstable feedforward controllers. This is due to the plant factorization that splits the identified plant into a the piece that does have a stable causal inverse and a piece that does not. Then, only the piece that does have a stable causal inverse is inverted and the other (non-minimum phase) piece is used to define the class of signals that may be perfectly tracked.

In the second part of the presented research, the focus was on modeling and control of photobioreactors (PBRs) growing microalgae for biofuel production. A nonlinear dynamic physics-based model was developed that accurately modeled microalgae growth in a resource limited (e.g., light-limited) environment. This model is a physics-based model that is independent of reactor size, which makes it a scalable model. The model was verified on experimental data from an actual PBR.

The biggest known factor affecting microalgae growth is proper pH regulation. For this, a pH model was developed that may be used for controller synthesis. The model includes both the media dynamics (that are approximately linear around an operating point), and the nonlinear growth dynamics (i.e., the dynamics of consuming and producing CO_2 as function of input PAR). Using this model, both a feedback robust controller and a DFFPC were synthesized and simulated. For the DFFPC, some modifications were required to define and achieve perfect tracking for this specific nonlinear system. The results provide an example of how the DFFPC architecture, which was presented for LTI systems, may be extended to a specific nonlinear system. While not presented here, a similar modification could be used to apply the DFFSP architecture for pH regulation on a nonlinear PBR. This example shows the potential versatility of the presented architectures to nonlinear systems. However, the modifications required will be specific to the nonlinear plant being controlled.

12.1 Feedforward Design

For the feedforward designs presented here, both design for specifications (e.g., design for specific rise time with no overshoot) and multiobjective optimization were presented for designing $P_{\text{des}}(s)$. In these methods, there is a design trade-off between control authority and nominal performance. These design methods were used in conjunction with the robustness tools to guide the design process to provide robust performance on the actual system (for a given level of uncertainty). In [39], a similar controller architecture was used to design $P_{\text{des}}(s)$ in terms of the signal $r(t) - y(t)$, where $r(t)$ is the unfiltered reference signal. In general, $r(t) - y(t) \neq 0$ (recall that perfect tracking was with respect to the filtered reference, namely $r_{ff}(t) - y(t) = 0$, so one can think of the filtered reference $r_{ff}(t)$ as a class of signals that the output can track perfectly). Design techniques such as those in [39] may also be considered for the architectures presented here.

In terms of perfect tracking as it has been presented here, the idea of perfect tracking was presented as: "Given $r(t)$, what is the $r_{ff}(t)$ that the output will perfectly track", and $P_{\text{des}}(s)$ is designed to shape $r_{ff}(t)$ to meet design specifications. Another way to approach the problem is ask: "What class of signals $r(t)$ can be perfectly tracked?". With the current presentation, we define $R_{ff}(s) = P_{\text{des}}(s)G_{noi}(s)R(s)$. For the question asked here, we are interested in determining $R(s) = P_{\text{des}}^{-1}(s)G_{noi}^{-1}(s)R_{ff}(s)$. In this formulation, $P_{\text{des}}^{-1}(s)G_{noi}^{-1}(s)$ will not necessarily be proper. Therefore, relative degree constraints will need to be put on $R_{ff}(s)$. This type of design would be desirable when $r(t)$ is a design parameter and the control objective is to track $r(t)$ perfectly.

12.2 Adaptation

The LTI adaptation techniques were based around known adaptation methods, such as standard system identification techniques (c.f., [85]) that perform identification

on a (stored) time-series of input-output data. While these may work well in some situations, it may be desirable to develop recursive methods that continuously update online at every time step (or even every couple of time steps). It may even be possible to identify the invertible and non-invertible dynamics separately, which could lead to a new class of system identification algorithms.

The use of reinforcement learning control is an active area of research. For the architecture presented here, any (stable) adaptation scheme may be used. This could include both static and dynamic function approximators (e.g., neural networks and echo state networks). More investigation needs to be done on the different methods available to see which methods are most suited for a specific (class of) systems. In addition to the architectural exploration, there is also the question of: "How to model reinforcement learning domains?". Some initial work for modeling reinforcement learning domains was presented in [82]. The modeling of reinforcement learning domains is in contrast to modeling the internal state using an observer. Observers use a model of the process and then "observe" the internal state, based on the model and input-output time series. In reinforcement learning domains, the (hidden) internal state is modeled using input/output data. One of the biggest challenges is modeling a (hidden) state from an input/output time series when there is no underlying knowledge of the model. This challenge, some initial solutions, and potential future directions were provided in [82].

12.3 Extensions to MIMO

In another form of the problem formulation, the plant may be factored into

$$G(s) = G_{mp}(s)G_{ap}(s)e^{-s\tau_d}, \tag{12.1}$$

where $G_{mp}(s)$ is a (potentially unstable) minimum-phase system that does have a causal stable inverse and $G_{ap}(s)e^{-s\tau_d}$ contains a stable all-pass $G_{ap}(s)$ with time

delay $e^{-s\tau_d}$. The latter piece will not have stable causal inverse, but will itself be stable. This factorization is known as an all-pass/minimum-phase decomposition. Using this factorization, the two feedforward controllers could be defined as $FF1(s) = P_{\text{des}}(s)G_{ap}(s)e^{-s\tau_d}$ (which defines the signals that may be perfectly tracked) and $FF2(s) = P_{\text{des}}(s)G_{mp}^{-1}(s)$ (which defines the feedforward signal that provides perfect tracking). There is no advantage to using this decomposition over the one presented in Chapter 3, since the same relative degree constraint will result from the formulation of $P_{\text{des}}(s)G_{mp}^{-1}(s)$. In fact, this method has the potential to be more limiting, since it forces the class of signals that may be perfectly tracked (defined by $P_{\text{des}}(s)G_{ap}(s)e^{-s\tau_d}$) to have stable poles that mirror the RHP zeros in $G(s)$. However, versions of this decomposition exist for general multiple-input multiple-output (MIMO) LTI systems, which could make this method a potential starting point for extending the DFFPC and DFFSP architectures to MIMO plants.

In the MIMO case, an all-pass/minimum-phase plant decomposition may be achieved through an inner-outer factorization [43; 93]. For the description here, a MIMO transfer function $H(s)$ is *inner* if it satisfies the property $H^*(s)H(s) = I$ and is analytic in the closed RHP (i.e., $s \geq 0$). In this definition, the real rational transfer function matrix $H(s)$ does not need to be square; however, when it is square, it is the MIMO equivalent of a stable all-pass transfer function. Similarly, a real rational transfer function matrix is *outer* if it is has no transmission zeros in the RHP, which is the MIMO equivalent of a minimum-phase transfer function. As a consequence, an outer transfer function matrix has a stable causal inverse. This factorization is given by

$$G(s) = G_{out}(s)G_{in}(s), \tag{12.2}$$

where $G_{in}(s)$ is inner and $G_{out}(s)$ is outer. From this definition, $G_{in}(s) = G_{out}^{\#}(s)G(s)$, where $G_{out}^{\#}(s)$ is the left inverse of $G_{out}(s)$. Using this method, there is the poten-

tial of creating the two feedforward controllers based on this MIMO decomposition. In [93], the inner-outer factorization may be applied to a wide range of LTI (e.g., polynomial/proper/improper) systems, whose transfer function matrices may be rank deficient and could have poles/zeros on the imaginary axis or at infinity [93]. While this method does show potential for factoring the plant, there are still the questions of:

- How to design $P_{\text{des}}(s)$?
- How to perform adaptation?

Once a $P_{\text{des}}(s)$ has been chosen, methods similar to those presented in Section 2.7 may be used to analyze the overall robustness of the closed-loop system. While the inner-outer factorization does look promising, it would be more desirable to factor a MIMO system into a piece that does have a causal and stable inverse, and a piece that does not have a causal and stable inverse. For this, a methodology that can handle improper systems in the MIMO case is needed. Some potential methods could include polynomial and rational matrices or descriptor systems; however, no known MIMO factorization currently exist.

12.4 Microalgae Modeling and Control

The PBR model developed was physics-based and validated on the experimental data. The measured model parameters were specific to the microalgae strain *Nannochloropsis oculata* in a specific PBR. However, the developed model has the potential to be used for system level optimization. For this, more data and experimentation would be needed to determine the sun utilization constant (K_{PAR}) for various algae strains and PBR configurations. Based on the system level model created from this data, the best algae strain and PBR configuration could be selected.

One of the main motivating applications for microalgae is to make biofuels from them. One way to promote lipid accumulation in microalgae (which will increase the overall biofuel content that may be extracted) is by depleting the microalgae of nutrients. For this, a lipid model could be created. This model would be fundamentally different than the growth model in the sense that no growth happens in the lipid accumulation stage. In fact, the growth model assumes that there are plenty of nutrients and that the only limiting resource is light. In a lipid model, the nutrients have been depleted, which means that growth will cease (and decaying will eventually begin to occur). In this case, the lipid model would contain the effects of bath temperature and incident PAR on lipid accumulation. The development of this "stress" model is a topic of ongoing research.

Advanced controllers were developed and demonstrated in simulation; however, they have not been validated on a test bed. For the DFFPC architecture, achievable tracking was defined for a specific $P_{\text{des}}(s)$ structure. Other methods for designing $P_{\text{des}}(s)$ and adapting of the growth model parameters would be useful. For adaptation, a real test bed would be needed to see how much benefit may be gained from adapting the model parameters. For the PBR setup considered here, plant model parameters could be adapted based on measured pH. For example, if the pH is lower than the expected value (e.g., the pH value that could be perfectly tracked by the DFFPC architecture), then the sun utilization constant K_{PAR} may be too large. Adaption could be use to adjust the parameter accordingly. This is one example of how adaptation may be used.

12.5 Control-Structure Interaction

In all physical applications, the location of sensors and actuators will contribute (in some form) to the overall dynamics of the system to be controlled. In the work presented here, a class of signals that may be perfectly tracked for non-minimum phase

systems is provided. If a desired level of performance is not attainable for the class of signals that is defined by the non-invertible dynamics of the plant, structural design considerations may be used to rearrange sensors and actuators in order to change the non-invertible dynamics of the system. An example was presented in Chapter 11 that used the time delay associated with CO_2 delivery in a PBR growing microalgae as a structural parameter. When the delay between when a CO_2 actuator opens, and when the CO_2 reaches the microalgae, was longer, there was more variation in the attainable pH regulation. However, reducing the amount of pH regulation would come at a potentially higher capital and operating cost, which means that a business trade-off would be required to determine the appropriate PBR configuration. While this is one specific example, it illustrates that there are fundamental limitations imposed by the construction of a physical system. By classifying the signals that may be perfectly tracked, it is possible to determine the achievable level of performance at the early stages of controller design. This knowledge may be useful for the structural design of the system by providing trade-offs between various actuator and sensor designs. These trade-offs may include bandwidth and operating range of the actuators and quality and placement of sensors.

REFERENCES

[1] J. E. Normey-Rico and E. F. Camacho, *Control of Dead-Time Processes*. Springer-Verlag, London, 2007.

[2] Y. Chisti, "Biodiesel from microalgae," *Biotechnology Advances*, vol. 25, no. 3, pp. 294–306, 2007.

[3] J. Sheehan, T. Dunahay, J. Benemann, and P. Roessler, "A look back at the U.S. department of energy's aquatic species program—biodiesel from algae," Tech. Rep. NREL/TP-580-24190, National Renewable Energy Laboratory, Golden, Colorado, July 1998.

[4] A. Richmond, "Microalgal biotechnology at the turn of the millennium: A personal view," *Journal of Applied Phycology*, vol. 12, pp. 441–451, 2000.

[5] J. Merchuk and X. Wu, "Modeling of photobioreactors: Application to bubble column simulation," *Journal of Applied Phycology*, vol. 15, pp. 163–169, 2003.

[6] M. Janssen, L. de Bresser, T. Baijens, J. Tramper, L. R. Mur, J. F. Snel, and R. H. Wijffels, "Scale-up aspects of photobioreactors: effects of mixing-induced light/dark cycles," *Journal of Applied Phycology*, vol. 12, pp. 225–237, 2000.

[7] M. Janssen, J. Tramper, L. Mur, and R. Wijffels, "Enclosed outdoor photobioreactors: light regime, photosynthetic efficiency, scale-up, and future prospects," *Biotechnology and Bioengineering*, vol. 81, no. 2, pp. 193–210, 2003.

[8] A. Richmond and N. Zou, "Efficient utilisation of high photon irradiance for mass production of photoautotrophic micro-organisms," *Journal of Applied Phycology*, vol. 11, pp. 123–127, 1999.

[9] A. P. Carvalho, L. A. Meireles, and F. X. Malcata, "Microalgal reactors: A review of enclosed system designs and performances," *Biotechnology Progress*, vol. 22, pp. 1490–1506, 2006.

[10] J. Deschenes, A. Desbiens, M. Perrier, and A. Kamen, "On simultaneous control of biomass and metabolite concentrations in perfusion bioreactors," in *DCDIS 4th Conference on Engineering Applications and Computational Algorithms*, (Guelph, Canada), pp. 663–667, 2005.

[11] F. Camacho Rubio, F. Garcia Camacho, J. M. Fernández Sevilla, Y. Chisti, and E. Molina Grima, "A mechanistic model of photosynthesis in microalgae," *Biotechnology and Bioengineering*, vol. 81, no. 4, pp. 459–473, 2003.

[12] F. Camacho Rubio, A. Sánchez Mirón, M. Cerón Garcia, F. Garcia Camacho, E. Molina Grima, and Y. Chisti, "Mixing in bubble columns: a new approach for characterizing dispersion coefficients," *Chemical Engineering Science*, vol. 59, no. 20, pp. 4369–4376, 2004.

[13] Y. Chisti, *Airlift Bioreactors*. Elsevier Applied Science, 1989.

[14] Y. Chisti, B. Halard, and M. Moo-Young, "Liquid circulation in airlift reactors," *Chemical Engineering Science*, vol. 43, pp. 451–457, 1988.

[15] Y. Chisti, M. Kasper, and M. Moo-Young, "Mass transfer in external-loop airlift bioreactors using static mixers," *The Canadian Journal of Chemical Engineering*, vol. 68, no. 1, pp. 45–50, 1990.

[16] N. Kurano and S. Miyachi, "Selection of microalgal growth model for describing specific growth rate-light response using extended information criterion," *Journal of Bioscience and Bioengineering*, vol. 100, no. 4, pp. 403–408, 2005.

[17] S. Celikovsky, S. Papacek, A. C. Herrera, and J. R. Leon, "Singular perturbation based solution to optimal microalgal growth problem and its infinite time horizon analysis," in *47th IEEE Conference on Decision and Control*, 2008.

[18] M. Berenguel, F. Rodriguez, F. Acien, and J. Garcia, "Model predictive control of ph in tubular photobioreactors," *Journal of Process Control*, vol. 14, pp. 377–387, 2004.

[19] F. Camacho Rubio, F. Acién Fernández, F. Garcia Camacho, J. A. Sánchez Pérez, and J. M. Fernández Sevilla, "Prediction of dissolved oxygen and carbon dioxide concentration profiles in tubular photobioreactors for microalgal culture," *Biotechnology and Bioengineering*, vol. 62, no. 1, pp. 71–86, 1999.

[20] L. Mailleret, O. Bernard, and J. P. Steyer, "Nonlinear adaptive control for bioreactors with unknown kinetics," *Automatica*, vol. 40, no. 8, pp. 1379 – 1385, 2005.

[21] L. Mailleret, J.-L. Gouze, and O. Bernard, "Nonlinear control for algae growth models in the chemostat," *Bioprocess and biosystems engineering*, vol. 27, no. 5, pp. 319–327, 2005.

[22] M. R. Buehner, P. M. Young, B. Willson, D. Rausen, R. Schoonover, G. Babbitt, and S. Bunch, "Microalgae growth modeling and control for a vertical flat panel photobioreactor," in *Proceedings of the 2009 American Control Conference*, 2009.

[23] "Center for the study of carbon dioxide and global change." Internet Website. http://www.co2science.org/.

[24] K. Logothetis, S. Dakanali, N. Ioannidis, and K. Kotzabasis, "The impact of high co2 concentrations on the structure and function of the photosynthetic

apparatus and the role of polyamines," *Journal of Plant Physiology*, vol. 161, pp. 715–724, 2004.

[25] L. Yue and W. Chen, "Isolation and determination of cultural characteristics of a new highly co2 tolerant fresh water microalgae," *Energy Conversion and Management*, vol. 46, pp. 1868 – 1876, 2005.

[26] D. Cooper, "Loop-pro software user's guide." Control Station.

[27] D. T. McRuer, R. E. Magdalena, and G. P. Moore, "A neuromuscular actuation system model," *IEEE Transactions on Man-Machine Systems*, vol. 9, no. 3, pp. 61 – 71, 1968.

[28] K. Furuta, M. Iwase, and S. Hatakeyama, "Internal model and saturating actuation in human operation from view of human-adaptive mechatronics," *IEEE Transactions on Industrial Electronics*, vol. 52, no. 5, pp. 1236 – 1245, 2005.

[29] S. Lee and D. Terzopoulos, "Biomechanical modeling and neuromuscular control of the neck," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 1188 – 1198, 2006.

[30] Y. Lin and S.-M. Sow, "Kinematic control and coordination of walking machine motion using neural networks," in *IEEE International Joint Conference on Neural Networks*, pp. 248 – 253, November 1991.

[31] I. D. Loram and M. Lakie, "Human balancing of an inverted pendulum: position control by small, ballistic-like, throw and catch movements," *Journal of Physiology*, vol. 540.3, pp. 1111 – 1124, 2002.

[32] S. Mangan, A. Zaslaver, and U. Alon, "The coherent feedforward loop serves as a sign-sensitive delay element in transcription networks," *Journal of Molecular Biology (2003) 334, 197204*, vol. 334, pp. 197 – 204, 2003.

[33] T. E. Milner, B. Ng, and D. W. Franklin, "Learning feedforward commands to muscles using time-shifted sensory feedback," *International Congress Series*, vol. 1291, pp. 113 – 116, 2006.

[34] H. Fujimoto, Y. Hori, T. Yamaguchi, and S. Nakagawa, "Proposal of seeking control of hard disk drives based on perfect tracking control using multirate feedforward control," in *International Workshop on Advanced Motion Control*, pp. 74 – 79, 2000.

[35] H. Fujimoto, Y. Hor, and S. Kondo, "Perfect tracking control based on multirate feedforward control and applications to motion control and power electronics," in *Power Conversion Conference*, 2002.

[36] F. Li, J. Lu, X. Zhao, and T. Yahagi, "Perfect tracking control of nonmiminum phase systems in magnetic levitation system," *IEICE Transactions on Electronics*, vol. E89-A, pp. 1437 – 1445, 2006.

[37] C. M. Schar, C. H. Onder, and H. P. Geering, "Control of an scr catalytic converter system for a mobile heavy-duty application," *IEEE Transactions on Control Systems Technology*, vol. 14, pp. 641 – 653, 2006.

[38] F. Lange, J. Langwald, and G. Hirzinger, "Predictive feedforward control for high speed tracking tasks," in *In Proceedings of the 1999 European Control Conference*, 1999.

[39] H. Okajima and T. Asai, "Performance limitation of tracking control problem for a class of references," in *Proceedings of the 47th IEEE Conference on Decision and Control*, (Cancun, Mexico), Dec. 9-11, 2008 2008.

[40] M. R. Buehner and P. M. Young, "A tighter bound for the echo state property," *IEEE Transactions on Neural Networks*, vol. 17, pp. 820–824, 2006.

[41] C. L. Phillips and H. T. Nagle, *Digital Control System Analysis and Design*. Prentice-Hall, third ed., 1995.

[42] J. C. Doyle, B. A. Francis, and A. R. Tannenbaum, *Feedback Control Theory*. Macmillan Publishing Company, 1992.

[43] K. Zhou, J. Doyle, and K. Glover, *Robust and Optimal Control*. New Jersey: Prentice Hall, 1996.

[44] C. L. Phillips and R. D. Harbor, *Feedback Control Systems*. Prentice Hall, 2000.

[45] K. Zhou and J. C. Doyle, *Essentials of Robust Control*. Prentice Hall, 1998.

[46] Z.-Q. Wang and S. Skogestad, *Analysis and Optimization of Systems: State and Frequency Domain Approaches for Infinite-Dimensional Systems*, ch. Robust controller design for uncertain time delay systems, pp. 610–623. Springer Berlin / Heidelberg, 1993.

[47] J. Doyle, "Analysis of feedback systems with structured uncertainty," *IEE Proceedings, Part D*, vol. 129, pp. 242–250, Nov. 1982.

[48] M. Safonov, "Stability margins for diagonally perturbed multivariable feedback systems," *IEE Proceedings, Part D*, vol. 129, pp. 251–256, 1982.

[49] A. K. Packard and J. C. Doyle, "The complex structured singular value," *Automatica*, vol. 29, pp. 71–109, 1993.

[50] P. M. Young, "Controller design with real parametric uncertainty," *International Journal of Control*, vol. 65, pp. 469–509, 1996.

[51] P. Young, "Robust control class notes," 1997.

[52] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control*. New York: John Wiley & Sons, 1996.

[53] R. D. Braatz, P. M. Young, J. C. Doyle, and M. Morari, "Computational complexity of μ calculation," *IEEE Transactions on Automatic Control*, vol. 39, pp. 1000–1002, 1994.

[54] L. Qiu and K. Kemin Zhou, *Introduction to Feedback Control*. Prentice Hall, 2010.

[55] O. Smith, "A controller to overcome dead time," *ISA Journal*, vol. 6, pp. 28–33, 1959.

[56] M. Morari and E. Zafiriou, *Robust Process Control*. New Jersey: Prentice Hall, 1989.

[57] K. Yamada, "Modified internal model control for unstable systems," in *7th Mediterranean Conference on Control and Automation*, 1999.

[58] G. Balas, J. Doyle, K. Glover, A. Packard, and R. Smith, "The μ analysis and synthesis toolbox." MathWorks and MUSYN, 1996.

[59] M. Anderson, M. Buehner, P. Young, D. Hittle, C. Anderson, J. Tu, and D. Hodgson, "MIMO robust control for heating, ventilating, and air conditioning (hvac) systems," *IEEE Transactions on Control Systems Technology*, vol. 16, pp. 475–483, May 2008.

[60] D. L. Laughlin, D. E. Rivera, and M. Morari, "Smith predictor design for robust performance," *International Journal of Control*, vol. 46, pp. 477 – 504, 1987.

[61] Z.-Q. Wang and S. Skogestad, "μ analysis and synthesis of time delay systems using smith predictor," in *Proceedings of the 30th Conference on Decision and Control*, 1991.

[62] T. H. Lee, Q. G. Wang, and K. K. Tan, "Robust smith-predictor controller for uncertain delay systems," *AIChE Journal*, vol. 42, pp. 1033 – 1040, 1996.

[63] Z.-Q. Wang and S. Skogestad, "Robust control of time-delay systems using the smith predictor," *International Journal of Control*, vol. 57, pp. 1405 – 1420, 1993.

[64] Y. J. Wang and J. B. Rawlings, "A new robust model predictive control method. ii: examples," *Journal of Process Control*, vol. 14, pp. 249–262, 2004.

[65] K.-S. Hong, D.-H. Kang, and J.-G. Kim, "Robust smith predictor design via uncertainty quantification: Application to a reclaimer," in *IFAC System Identification*, 2000.

[66] K. Yamada and H. Takenaga, "The parametrization of all stabilizing smith predictors for certain class of non-minimum phase time-delay plants," in *Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC'06)*, 2006.

[67] K. Yamada, H. Takenaga, and H. Yamamoto, "A design method for smith predictor for non-minimum-phase time-delay plants with multiple time-delays," in *The 3rd Intetnational Conference on Innovative Computing Information and Control (ICICIC'08)*, 2008.

[68] A. Faanes, *Controllability Analysis for Process and Control System Design*. PhD thesis, Norwegian University of Science and Technology, 2003.

[69] "Control system toolbox." Toolbox available from Mathworks. See `http://www.mathworks.com/products/control`.

[70] J. Doyle, K. Glover, P. Khargonekar, and B. Francis, "State space solutions to $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems," *IEEE Transactions on Automatic Control*, vol. 34, pp. 831–847, August 1989.

[71] R. M. Kretchmar, *A Synthesis of Reinforcement Learning and Robust Control Theory*. PhD thesis, Colorado State University, Department of Computer Science, 2000.

[72] R. Kretchmar, P. Young, C. Anderson, D. Hittle, M. Anderson, C. Delnero, and J. Tu, "Robust reinforcement learning control with static and dynamic stability," *International Journal of Robust and Nonlinear Control*, vol. 11, pp. 1469–1500, 2001.

[73] R. M. Kretchmar, P. M. Young, C. W. Anderson, D. Hittle, M. Anderson, J. Tu, and C. C. Delnero, "Robust reinforcement learning control," in *Proceedings of the American Control Conference*, pp. 902–907, 2001.

[74] C. W. Anderson, P. M. Young, M. R. Buehner, K. A. Bush, and D. C. Hittle, "Robust reinforcement learning control using integral quadratic constraints for recurrent neural networks," *IEEE Transactions on Neural Networks. Special Issue on Neural Networks in Controls Applications*, vol. 18, pp. 993–1002, 2007.

[75] M. R. Buehner, C. W. Anderson, P. M. Young, K. A. Bush, and D. C. Hittle, "Improving performance using robust recurrent reinforcement learning control," in *Proceedings of the European Control Conference*, pp. 1676–1681, 2007.

[76] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, 2001.

[77] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[78] J. J. Steil, *Input-Output Stability of Recurrent Neural Networks*. Göttingen: Cuvillier Verlag, 1999. (Also: Phd.-Dissertation, Faculty of Technology, Bielefeld University, 1999).

[79] A. F. Atiya and A. G. Parlos, "New results on recurrent network training: Unifying the algorithms and accelerating convergence," *IEEE-NN*, vol. 11, p. 697, May 2000.

[80] P.-G. Plöger, A. Arghir, T. Günther, and R. Hosseiny, "Echo state networks for mobile robot modeling and control.," in *RoboCup*, pp. 157–168, 2003.

[81] H. Jaeger, "The echo state approach to analyzing and training recurrent neural networks," Tech. Rep. 148, German National Research Center for Information Technology, 2001.

[82] K. A. Bush, *An Echo State Model of Non-Markovian Reinforcement Learning*. PhD thesis, Colorado State University, 2008.

[83] R. A. Horn and C. R. Johnson, *Matrix Analysis*. New York: Cambridge University Press, 1985.

[84] L. N. Trefethen and D. Bau III, *Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[85] "System identification toolbox." Toolbox available from Mathworks. See `http://www.mathworks.com/products/robust`.

[86] H. Dette, V. B. Melas, A. Pepelyshev, and N. Strigul, "Robust and efficient design of experiments for the Monod model," *Journal of Theoretical Biology*, vol. 234, pp. 537–550, 2005.

[87] J. A. Duffie and W. A. Beckman, *Solar Engineering of Thermal Processes*. Wiley-Interscience, 2nd ed., 1991.

[88] H. W. Johnston, "The biological and economic importance of algae. part 4: the industrial culturing of algae," *Tuatara : Journal of the Biological Society*, vol. 22, pp. 1–105, 1976.

[89] E. F. Camacho and C. Bordons, *Model Predictive Control*. Springer-Verlag, London, 2004.

[90] K. Weyer, D. Bush, A. Darzins, and B. Willson, "Theoretical maximum algal oil production," in *BioEnergy Research (to appear)*, 2009.

[91] J. V. Moroney, "Algal photosynthesis," in *Encyclopedia of Life Sciences*, Joh Wiley and Sons, Ltd., 2001.

[92] G. Edwards and D. Walker, *C3, C4: Mechanisms and Cellular and Environmental Regulation of Photosynthesis*. Blackwell Scientific Publications, 1983.

[93] C. Oara and A. Varga, "Computation of general innerouter and spectral factorizations," *IEEE Transactions on Automatic Control*, vol. 45, pp. 2307–2324, December 2000.