THESIS

# CONTROL DESIGN FOR GENERATOR OF NONLINEAR HIGH FREQUENCY PLASMA SYSTEM

Submitted by

Prajay Vora

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2021

Master's Committee:

Advisor: Peter Young

Daniel Zimmerle
James Cale

# ABSTRACT

## CONTROL DESIGN FOR GENERATOR OF NONLINEAR HIGH FREQUENCY PLASMA SYSTEM

This document aims to develop control systems for a generator of a nonlinear high frequency plasma system. Initial modelling was done by Advanced Energy Industries, Inc. (AE) which was passed on to Colorado State University environment for further research into developing controllers for this special model. This thesis documents all the work done by Colorado State University till Summer of 2020. The first phase of the collaboration included finding metrics for the feedback system with the nonlinear load modelled by AE. The metrics serve for better understanding of the modelling and also to generate effective control criteria suited to AE requirements. AE required for the user defined wave-forms to be tracked in an average sense without significantly changing the real time tracking criteria. This tradeoff was also addressed while developing metrics. A preliminary approach for control design was a PID controller to study its effects in a nonlinear environment. A robust control approach called $H_\infty$ loop-shaping is the primary control design developed by CSU for this specific application. The nonlinear system was approximated with a transfer function and the controller developed for that approximation. The purpose of the approximation is to generate a controller that is highly robust considering the uncertainties in high frequency plasma loads. The metrics discussed above are used for confirming the efficiency of the controllers. Controller design was the second phase of the project. Finally, in phase three, Nelder-Mead optimization was used to generalize the $H_\infty$ controller for various generator and set-point specifications. A system identification processes was also developed consisting of curve fit models for the nonlinear load. This was done with a view to the future for classifying different loads and plasma to develop customised controllers.

ACKNOWLEDGEMENTS

I would like to thank Advanced Energy Industries, Inc. for initiating and supporting the project. I would like to thank Prof. Peter Young for his guidance and helping me generate concrete ideas. I would also like to thank Dr. Chad Samuels at Advanced Energy Industries, Inc. for his prompt responses and all the help he offered during the course of the research. Finally, I would like to thank Dr. David Christie, Dr. Gideon van Zyl, Electrical and Computer Engineering department and everyone who helped to make this project a success.

# DEDICATION

*To my parents, my grandmother and my brother who have been my support in the difficulties.*

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# Chapter 1

# Introduction

Advanced Energy (AE) has collaborated with Colorado State University (CSU) on a project developing advanced control strategies for radio frequency power supplies. AE makes products which deliver power to nonlinear plasma loads at high frequency. These power supplies typically use a network to match the plasma load impedance so that the generators can efficiently deliver power. The generator is modulated at a higher rate than the network used to match the impedance, so as to cope with the variation in impedance, and at least keep the average impedance matched. A Field-programmable gate array (FPGA) is used at very high frequency to keep up with the changes in the power delivered to the load. A much slower microprocessor controls the overall system. The plasma load can vary according to gas mixture, gas pressure and changes in the plasma apparatus in the plasma chamber. The plasma generator has fast dynamics, allowing for quick and high-level changes in the power setpoint. Note that the load impedance matched at the generator is a function of the power delivered to the load. At the same time, the output power is a function of the matched impedance. This inherent feedback loop in the system makes it an interesting and challenging problem for control design.

## 1.1 Real-Time vs Average Tracking

Real-Time tracking of a control system corresponds to accurately track the given setpoint considering the transient overshoots and undershoots. Average tracking, according to the AE system, corresponds to tracking of a power level on the whole as an average without considering the the transient and steady state responses. As an example, if a particular response at any power level has an overshoot and goes into a steady state that might be acceptable for real-time tracking but would not be appropriate for the average tracking since the average power would not be not be minimised.

## 1.2 Metrics

Phase 1 of the project aimed to develop metrics for the Advanced Energy (AE) system. The metrics are based on errors generated between the setpoint and the power delivered to the load. The metrics, having numerical values and graphical representation, are supposed to capture different aspects of performance of the AE system. Most important of those aspects are real time tracking and average tracking. The metrics used have been designed with absolute values and average values of errors to focus on real time tracking and average tracking. These metrics are then stated in the form of optimization problems to understand the goals that needed to be achieved while designing a controller. The metrics were calculated after every design was implemented.

## 1.3 Control Design

$H_\infty$ loopshaping methodology was selected because of the high frequency of the plasma load and the ease of implementation of the controller. This process was a part of the phase 2 of the project which included controller design and implementation. This method solves an optimization problem called 'Normalised Coprime Factor $H_\infty$ robust stabilization problem' and uses loopshaping methods, both of which together guarantee closed loop stability at all frequencies. A PID controller was also used as a preliminary control method to learn of its usefulness in a nonlinear and high frequency environment. A simple system identification consisting of curve fitting was done on the open loop plant containing the dynamics to generate a set a equations and visualize the plant at various levels of power and voltage.

## 1.4 Optimization

The controller was designed for a special combination of a '$50\Omega$ point' (source impedance for radio generator), Power Setpoint and $Z\_rotation$ (Cable length in terms of rotation angle). The plasma and the generator system, however, cannot be of the same specifications with every application. Hence, an optimization routine had to be devised for designing the controller for a range of the specifications to generalize the controller. This part of the project was included in the Phase 3 of the project timeline. The optimization creates an optimal set of specifications for the $H_\infty$ loopshaping controller and runs the optimal controller for the system keeping Power Setpoint and 50$\Omega$ point as constants and varying $Z\_rotation$. Parallel processing was used on CSU linux servers to reduce simulation time. The Nelder-Mead optimization routine was used for this purpose because of its easy and quick simplex calculations without the computation of gradients and its capability to handle nonlinear and nonsmooth objective functions.

## 1.5 Thesis Structure

The thesis consists of the following structure:

- Chapter 2 consists of present and past literature for the project, specified control design and optimization.
- Chapter 3 is an overview of the AE system.
- Chapter 4 discusses the shortcomings of the AE system that CSU needs to solve.
- Chapter 5 discusses the above theories CSU used for solving the problem.
- Chapter 6 describes the applications of these theories on the AE system and their simulation studies.
- Chapter 7 includes conclusion and future work.

# Chapter 2

# Literature Review

Two United States Patents registered by Gideon van Zyl serve as the basis of the project. The patent published [1] in 2009 gives the idea inspiring the invention. It consists a method for modifying the interactions between a power generator and a nonlinear load. The system described has a control signal that serves as an input to the controller. The controller is supposed to deliver a controlled signal to the nonlinear power load. The controlled signal is in the form of power, voltage and current. Sensors measure the impedance and uncontrolled signals from the nonlinear load and feed it back in the power generator for generating the controlled signal. The control signal drives through the nonlinear load dynamics reducing the transient response as much as possible. The publication gives a logical proof for the power that is generated by the generator changing the load impedance and hence the load impedance in turn changing the power generated. It is also evident in the publication that the sensitivity of the generator to small changes in load impedance reduces when a resistive source impedance is matched to the load impedance.

The patent published [2] in 2017 describes a few aspects of adjusting the source impedance of the generator as a better control approach. One method uses two signals that are generated and combined using a multiplexer. The output of the multiplexer delivers a controlled output to a plasma load. An isolated system is attached to the output of the combiner for adjusting the variable impedance that is fed back to the generator. Another method describes the two signals with an amplifier and then combined together. The multiplexer has output ports for first amplifier signal, second amplifier signal, power delivered to the load and one for adjusting the variable impedance.

In the publication [3] by K.J. McLaughlin, T.F. Edgar and I. Trachtenberg, a process control method is applied by the authors on a plasma etching system that analyses the system using Relative Gain Array and Signular Value Decomposition. A multivariable control design approach was used with a block relative gain method by Manousiouthakis [4]. An online system identification method was developed in [5] which uses a recursive parameter estimation method.

The robust control strategy used in this project by CSU is the $H_\infty$ loopshaping design procedure by D. McFarlane and K. Glover [6]. It solves a particular $H_\infty$ optimization problem to guarantee closed loop stability and robust stability at all frequencies. In the standard loopshaping approach, the specifications of closed loop design are expressed as open loop gains (singular values for MIMO) of the compensated system. Whereas in the $H_\infty$ loopshaping approach, the specifications of the closed loop system are expressed as closed loop gains (singular values) of the weighted transfer function and a specific optimization routine is used to obtain an optimal control according to the specifications. The method uses loopshaping to give desired singular values of the plant at low and high frequencies and the optimization problem shapes the plant.

The standard loopshaping approach requires for plant phase to be considered while designing. This creates some limitations for the loop shape design which makes the design more involved. Whereas in the $H_\infty$ loopshaping design approach, the phase is taken care of in the optimization problem, making the design process easier at all frequencies. 'The normalized coprime factor $H_\infty$ robust stabilization problem' has been rigorously solved in [7] and [8].

The AE/CSU system requires for unconstrained optimization for genralizing the results. The Nelder-Mead algorithm ( [9] and [10]) was used here for its direct numerical search techniques without the use of gradients. This method uses the concept of simplex for searching. This method is different than the popular Dantzig's simplex algorithm [11] which is a method of linear programming. Since there are not many past studies in this specific application, it becomes an interesting study not knowing where the challenges occur. In this thesis, we describe in detail, the problem that needs to be solved and the approaches that were used to solve the problem.

# Chapter 3

# AE System Overview

## 3.1 System Overview

The overall system schematic is illustrated below in figure 3.1. We describe below the basic function of each of the components below.

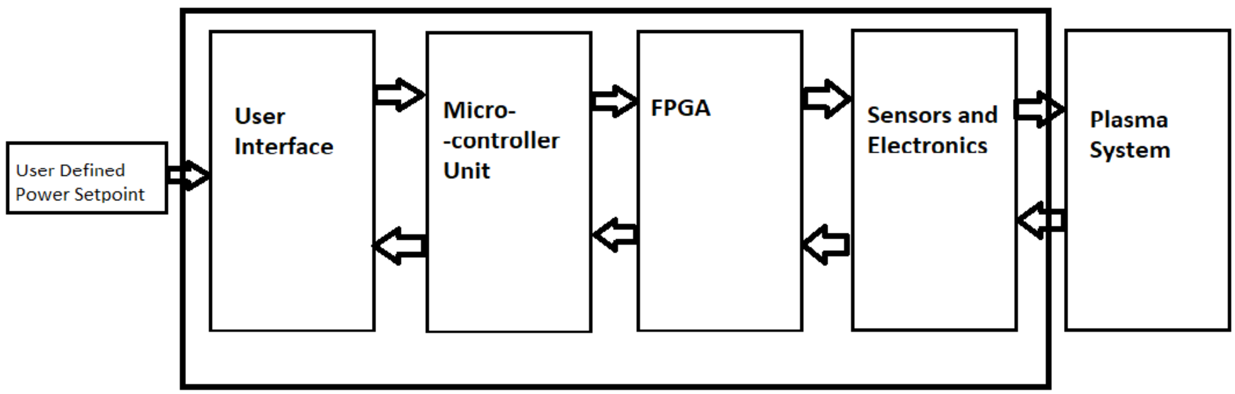


**Figure 3.1:** Plasma Power Generator Schematic

The input to the system is a user defined waveform for the plasma load and a Virtual Front Panel (VFP) provides for the user a GUI medium for setting in the desired setpoints, i.e., specify the desired power waveform to be tracked. The Microcontroller Unit (MCU) is used for relatively slow, high-level processing and complex algorithms. In contrast the Field Programmable Gate Array (FPGA) has a very fast clock pulse and is well suited to fairly simple algorithms that need to run quickly. This gives us the ability to have a two-loop control approach, with a fast inner loop implementing simple control algorithms (FPGA), while at the same time more complex processes run in the slower outer loop (MCU). The Electronics part of the 'Sensors and Electronics' block consists of RF circuitry which matches the input and output impedance for maximum power transfer between the generator and the nonlinear load. It also converts the digital signal from the

FPGA to an analog signal which is then supplied to the power amplifier. Sensors provide analog measurements of power, voltage, current and impedance from the plasma load. These signals then pass through the analog to digital converter, and on to the feedback control loop.

The current AE control system architecture is shown below in figure 3.2. We now provide a description of the basic control approach.

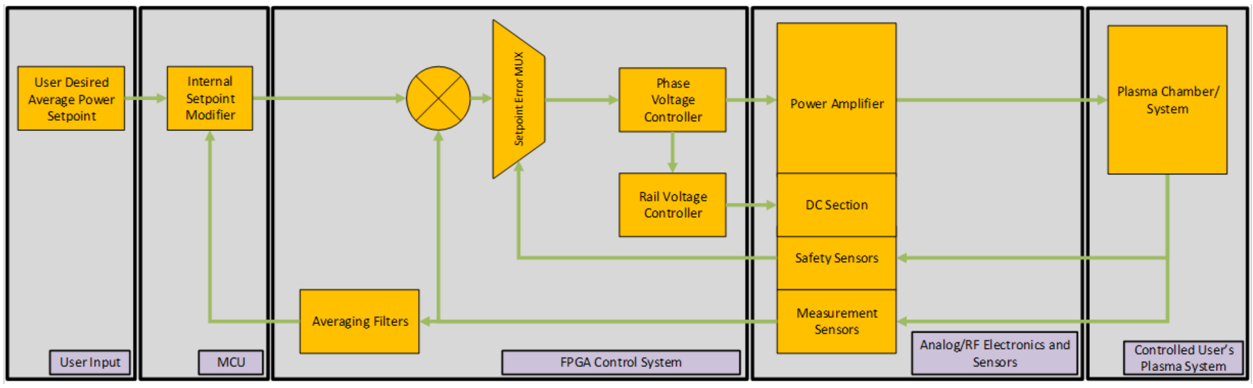


**Figure 3.2:** Plasma Power Generator Block

There are a few definitions that need to be addressed before the explanation of the AE system. A power state according to the AE system is a predefined magnitude of power in Watts that constitutes the setpoint defined by the user. For example 500W is a power state. A pulse cycle is a discrete waveform generated by power states merging together that occur in real time. For example, one pulse cycle can be assigned the power states 500W, 1000W, 800W, 500W, 700W and they form a discrete waveform. There needs to be timing/duty-cycle data in addition to the power states. This waveform is input by the user as setpoint. An averaging bandwidth is a set of samples in real time over which the average is calculated.

The FPGA utilizes the real-time measurements to perform real-time setpoint tracking, in a conventional control sense, constituting the fast inner loop of the feedback control system. At the same time the FPGA performs averaging of the sensed delivered power. An averaging bandwidth is used for calculating the average. The average is calculated in real time within the bandwidth as a running average and reset after the bandwidth, and the averaging processes reset at each power state. The average values are passed back to the MCU. Also note that the average values at each power state are calculated separately and stored for reuse.

The internal setpoint modifier constitutes the slow outer loop running on the MCU. It only receives averaged measurements, and uses these to modify the user defined power setpoint values, passing the modified setpoint (as a command) on to the FPGA control system. This setpoint modification is continuously updated, with the goal of matching the average power delivered to the original (user defined) power setpoint.

Note that each state is modelled separately in the setpoint modifier, and the modifier uses a PID controller for estimating the setpoint. Furthermore the algorithm attempts to make the average power track a second order critically damped system. This is to ensure there is no overshoot (or undershoot) in the step response of the average power, which is an important goal for the control design. Since the step inputs of power are huge in magnitude, the setpoint modifier attempts to approximate the step input as a gradually increasing steps. This is done to reduce the high magnitude step to small steps which avoids large overshoots in the controller and hence a better response. It should also be noted that when a power state change (change in power magnitude) occurs, the control signals of the previous power state are saved to avoid transient response when that state appears again. Also, in order to avoid the transient dynamics of the plasma load, the AE system has a provision to blank out all the samples that contribute to the transient dynamic response, hence, minimizing the overshoots (or undershoots).

Within the FPGA, an algorithm for rail and phase voltage is implemented. The goal is to minimize fluctuations in Vrail, while at the same time keeping the value as low as possible, subject to keeping Vphase close to its desired reference value. The rail and phase voltage control signals utilize integral control originally. Since Phase voltage output is used as an input for Rail voltage the system can be reduced to a single input single output system.

These digital control signals are then passed through a DAC for conversion to analog signals, and then a Power Amplifier. Note also that in addition to the basic control running there are additional safety sensors and systems which may overrule the control system under certain circumstances, to prevent the plasma system and the generator from damage.

Overall we see that the fast inner loop on the FPGA performs real-time tracking. However the setpoint tracked may not be the original one provided by the user, but rather a modified one provided by the slow outer loop on the MCU. The modification is done so that the average power accurately tracks the original (unmodified) setpoint. The control system designed by AE is non-conventional in the sense that it tracks average power with previous knowledge of the setpoint to ensure average tracking.

Since the modelling was done based on average tracking, the system failed to deliver appropriate real-time tracking. This trade-off is described in detail in the next chapter.

# Chapter 4

# Discussions of Current Approach/Shortcomings

## 4.1 Average Tracking vs Real-Time Tracking

The averaging is performed by the FPGA since it requires fast processing. It utilizes a counter for blankout time (as discussed in the previous chapter which blanks out the transient dynamics and a counter for calculating the (length of the) average). For that reason, the averaging algorithm blanks (i.e., ignores) the initial few samples to remove this transient behavior from the calculations. The averages are calculated separately for each (power setpoint) power state.

The average of each state is the total sum of power delivery measurements, divided by the counter (number of measurements), where the counter increments at the FPGA clock pulse rate. Note that this average is not calculated for the blankout period, and resets for every state.

The appropriate bandwidth (number of samples) to utilize for averaging is a complex issue. The entire system may become unstable when the averaging is done at an inappropriate bandwidth (too low or too high), due to aliasing-type (oversaturation/undersaturation due to counting the samples too many/little times) effects with respect to averaging. Hence there is a tradeoff between 'goodness of filter' and bandwidth. AE has developed tools to manage this tradeoff in such a way as to minimize aliasing-type effects, whilst also maintaining 'goodness of the filter' with respect to the number of samples used for averaging.

As discussed earlier the user setpoint data is modified before actual tracking. The modification has the goal of providing more accurate tracking for the average power. In order to illustrate this, consider the example tracking response in figure 4.1. Note in figures 4.1-4.4 the desired response is shown in red and the actual (real-time) response is shown in brown.

It is apparent from figure 4.1 that the real-time tracking error is very poor for the vast majority of the time. This control design would not be acceptable from a real-time tracking standpoint. Note, however, that the average tracking error over the whole pulse is zero (ideal). Hence it is clear

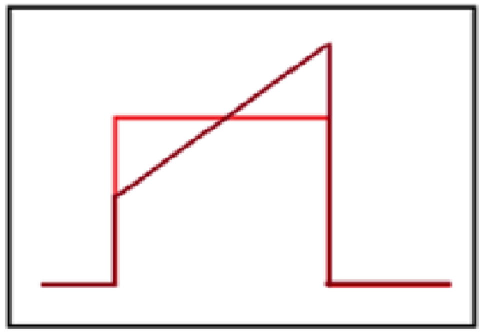

**Figure 4.1:** Example Real Time Tracking Response 1

that average tracking and real-time tracking are quite different goals, and this fact must be clearly taken into account in the control system criteria and subsequent design.

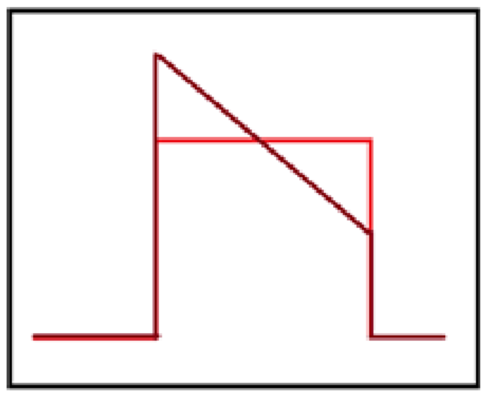

**Figure 4.2:** Example Real Time Tracking Response 2

The discussion for figure 4.1 applies equally well to the situation shown in figure 4.2. However, whereas the response in figure 4.1 is acceptable under the current control scheme at AE, the response in figure 4.2 is not. The reason is that if you calculate how the average proceeds over time (i.e., as the counter increases across the width of the pulse), the average in figure 4.1 approaches the setpoint from below, whereas the average in figure 4.2 approaches the setpoint from above. In AE parlance the 'average step response' in figure 4.1 has no overshoot, whereas that in figure 4.2 does overshoot significantly. The current control approach at AE requires that the average step response has no overshoot in addition to very little steady-state error. Figure 4.1 meets these goals whereas figure 2.4 does not (due to overshoot). Of course neither of figures 4.1, 4.2 would be acceptable if we considered real-time tracking as a goal.

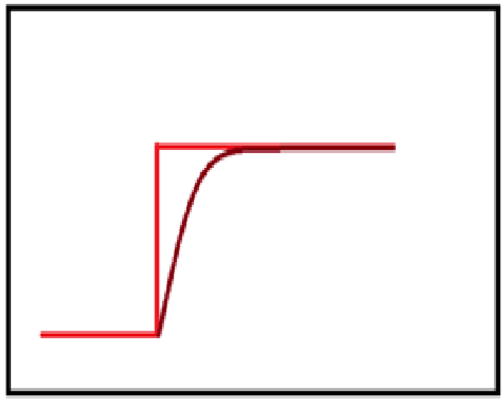

**Figure 4.3:** Example Real Time Tracking Response 3

Now consider the response in figure 4.3. From a real-time tracking viewpoint this looks very good. It approaches the setpoint with no overshoot, and quickly settles to zero steady-state (real-time) tracking error. However if we consider the average step response then although it has no overshoot, it has a large steady-state error (average error is not minimized). This design would be unacceptable from the (current AE) viewpoint of average tracking. Note also that in order to achieve average tracking error of zero, then any time the real-time response spends below the setpoint, must be balanced by some time above the setpoint. This is contrary to desired real-time behavior where the real-time response spends as much time as possible at (or close to) the setpoint. Hence real-time and average tracking are somewhat conflicting goals, and this tradeoff must be managed in the control system design.

Of course at the end of the day tracking well is a common feature to both goals, and so it is still possible to achieve high scores in both measures. For instance consider the situation illustrated in figure 4.4 below.

In this case tracking is almost perfect, and the result would be excellent from either a real-time or average tracking standpoint. In both cases there is almost no overshoot and almost zero steady state error. Of course achieving tracking close to this performance may not be realistic, and hence

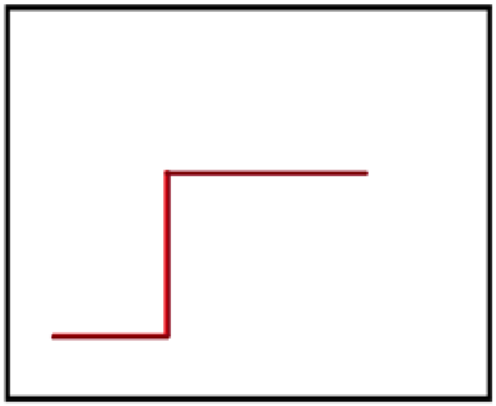

**Figure 4.4:** Example Real Time Tracking Response 4

in practice it becomes a challenging problem to achieve acceptable performance for both real time tracking and average tracking. Hence, practically, CSU aims to optimize both the requirements. As we will observe in the subsequent chapters, by analysing the metrics on the original AE system, the average tracking goal has been achieved to some extent but real time tracking goal is poor.

## 4.2 Integral Control

The AE system has obtained a working feedback control for the nonlinear plasma load which consists of a setpoint reference input, rail and phase voltage controllers, plant (plasma) and measurements. Other calculations consist of setpoint modifier and average calculations. Integral control is used by the AE system as the control strategy for rail and phase voltage. The voltage control signals are then converted to a power signal that is delivered to the plasma load. The integrators in both the rail and phase voltage use lookup tables for setting in the initial values and reset values of the integrator at the start of each power state to generate the required control signal. With the start of new power state, the previous power state's settings are saved for reusing the control signal when the same power state appears again. The disadvantage of such an approach is that the lookup tables are prepared considering only a particular set of setpoint samples. For various plasma applications and plasma systems, this is not a practical way of approaching the problem. CSU aims to eliminate this special approach and use a general approach which would be applicable for most of the radio frequency plasma systems.

# Chapter 5

# Theoretical Approach - Metrics, Control and Optimization

## 5.1 Metrics

In order to capture and precisely quantify system tradeoffs for real time and average tracking we need to develop some performance metrics. After studying a number of scenarios typical of AE control situations, we have decided to use the following metrics in our analysis,

- Define the tracking error at each point as $e_i = (Pdel\_setpoint_i - Pdel_i)$. Note that in this case negative values correspond to overshoot.
- Mean Absolute Error: $MAE = \frac{\sum_{i=1}^{N} |e_i|}{N}$. This metric calculates numerical values, and it resets at each pulse cycle.
- Mean Error: $ME = \frac{\sum_{i=1}^{N} e_i}{N}$. This metric calculates numerical values, and it resets at each pulse cycle.
- Note that the above two variations can be used for all the metrics involving tracking errors to follow. In other words for any metric we can have the 'Error' version using $ei$ and the 'Absolute Error' version using $|ei|$. In the interest of brevity we will only detail the 'Error' version for each metric from here on.
- Mean Accumulated Error: $MAcE_j = \frac{\sum_{i=1}^{j} e_i}{N_j}$ where $N_j = j$. This equation starts at $MAcE_1 = e_1$, then adds the errors for the first two samples, takes an average, then adds it to the error at the next sample, and takes another average and so on. This metric creates a plot (time series), and it resets at each pulse cycle.
- Mean Moving Error: At each transition we start running a moving average calculation for each point $j$ as: $MME = \frac{\sum_{i=N_{start}}^{j} e_i}{N_{length}}$, where $j = N_{start} + N_{length}$. In order to initialize this

calculation correctly at the start we first fix $N_{start}$ at $N_{start} = 1$, and compute as above for $N_{length} = 1, 2, ..., N_{average-end}$. At this point we then fix the value of $N_{length}$ at $N_{length} = N_{average-end}$ , and now carry on incrementing $N_{start}$ as $N_{start} = 1, 2, ...$ through to the next transition, running the above calculation at each point $j$. $N_{average-end}$ is kept constant throughout. This metric creates a plot (time series), and it resets at each transition.

Note that these metrics correspond to different aspects of the performance. For example suppose we seek to minimize $MAE$. This will correspond to placing the focus on real-time tracking – every large value for $|e_i|$ makes the metric worse, and high performance will correspond basically to driving $|e_i|$ to a small number quickly. This metric does not care about the sign of $e_i$.

Conversely if we utilize ME then it will minimize average tracking. Negative values for $e_i$ can be balanced out by positive values for $e_i$. The sign of $e_i$ is crucial and in general the average is what matters. Note also that a negative value corresponds to average overshoot which is highly undesirable.

Thus it is seen that 'Absolute Error' metrics tend to place emphasis on real-time tracking performance, whereas 'Error' metrics tend to place emphasis on average tracking performance. In the latter case the sign also informs us about potential overshoot.

## 5.2 PID Control

PID controllers constitute a very common control approach, widely used in industry, and with a suite of compatible tools (e.g., anti-windup, bumpless transfer) and numerous design methodologies for tuning the controller gains $K_P$, $K_I$, $K_D$.

$$K(s) = K_p + \frac{K_I}{s} + K_D s$$

They are very familiar to practicing engineers and can be implemented with a high degree of confidence. Of course they do not afford the sophisticated robust stability and performance guarantees of more modern approaches such as $H_\infty$ Loopshaping. The PID controller was studied for the AE system as a preliminary approach before starting the Robust Control ($H_\infty$ loopshaping) method.

## 5.3 H$_\infty$ Loopshaping Control Theory

The control problem presented here requires both stability and stringent tracking performance for a plant with significant nonlinearities and a high degree of uncertain dynamics. Hence robust performance is essential - although we have a plant model, the control approach cannot rely on it too heavily because it is subject to so much variation for different systems and applications. At the same time this robustness cannot come at the price of sluggish/poor performance – we require extremely accurate steady-state tracking while at the same time looking for improvement in the transient performance.

In order to deliver robustness we consider perturbations to our nominal model, and hence we need to characterize not only the nominal model, but also these perturbations, which are typically referred to as uncertainties. However we do not wish to utilize a complex uncertainty structure, as this would require a rather precise characterization of said uncertainties, which is counter to our original goal/problem statement. Rather we would like a fairly unstructured uncertainty description, which will afford a broad degree of robustness.

### 5.3.1 Uncertainty Descriptions

Linear system models often utilize a transfer function description of the form:

$$G(s) = \frac{n(s)}{d(s)}$$

where $n(s)$, $p(s)$ are polynomials (in the Laplace variable $s$) which describe the system dynamics. This SISO system description can be extended to MIMO systems via a transfer matrix description of the form:

$$G(s) = [g_{ij}(s)] = \left[\frac{n_{ij}(s)}{d_{ij}(s)}\right]$$

where $n_{ij}(s)$, $d_{ij}(s)$ are again polynomials. However another option, which turns out to have numerous useful properties (even in the SISO case) is to use a (right) coprime factor description:

$$G(s) = N(s)M^{-1}(s)$$

where $N(s)$, $M(s)$ are themselves linear system descriptions (i.e., transfer functions/matrices). Furthermore $M(s)$ is square and invertible, and $N(s)$, $M(s)$ are coprime factors. This means that $N(s)$, $M(s)$ are both stable systems, and they have no stable common factors. Note of course that this does not restrict us to stable systems for $G(s) = N(s)M^{-1}(s)$ because $M^{-1}(s)$ may or may not be stable. Hence this affords us another general way of describing (SISO or MIMO) systems $G(s)$.

Using this description effectively provides separate numerator and denominator dynamics (described by $N(s)$, $M(s)$ respectively), each of which can be subject to a perturbation of the form $\Delta s$ which is a stable dynamic system with bounded norm. This means that the set of perturbed plants $G_p(s)$ is of the form (we drop the explicit dependence on $s$ where convenient to avoid notational clutter):

$$G_p(s) = N_p M_p^{-1} = (N + \Delta_N)(M + \Delta_M)^{-1}$$

where $\Delta_N$ and $\Delta_M$ are stable systems with bounded $H_\infty$ norm:

$$\left\| \begin{pmatrix} \Delta_N(s) \\ \Delta_M(s) \end{pmatrix} \right\|_\infty \leq \frac{1}{\gamma}$$

for some parameter $\gamma$ (which characterizes the robustness level achieved). This type of uncertainty description is referred to as Normalized Coprime Factor Uncertainty, and it is illustrated in standard form in figure 5.1 below (with nominal system interconnect given by $P(s)$). It is an excellent fit to our problem, for reasons described earlier (note that it does not require a precise a-priori characterization of $\Delta_N$, $\Delta_M$), and it will form the basis of our controller design approach described in the next section.

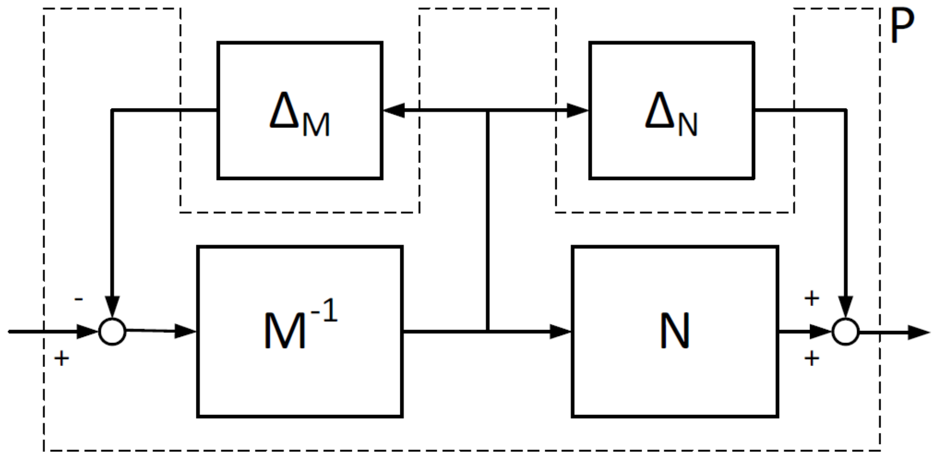


**Figure 5.1:** Normalised Coprime Factor Uncertainty Description

### 5.3.2 Robust Controller Design Algorithm

In order to tackle the robustness problem in figure 5.1, we employ the Normalized Coprime Factor Synthesis approach, also known as $H_\infty$ Loopshaping. This controller is obtained as the solution to a particular $H_\infty$ Optimal Control problem, which we will describe shortly. In so doing it affords robust stability (of a certain level $\gamma$) against Normalized Coprime Factor Uncertainty, as described above, and as illustrated in figure 5.1.

The $H_\infty$ Optimal Control performance specification is given as the $H_\infty$ norm of the gain from $\begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$ to $\begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$ in figure 5.2. This can be interpreted as tracking performance ($w_1$ to $z_1$) as well as disturbance rejection ($w_2$) and a penalty on control authority ($z_2$). Note that by convention this performance specification utilizes a positive feedback loop, but it is straightforward to modify it to use a negative feedback loop, which we do for application to the AE systems.

There is another way to interpret the performance specification of this control design approach. First note that the two loop gains illustrated in figure 5.3 are equivalent. Hence, as illustrated in figure 5.3, we define the controller K and shaped plant $G_S$ as:

$$K = W_1 K_S W_2 \qquad\qquad G_S = W_2 G W_1$$

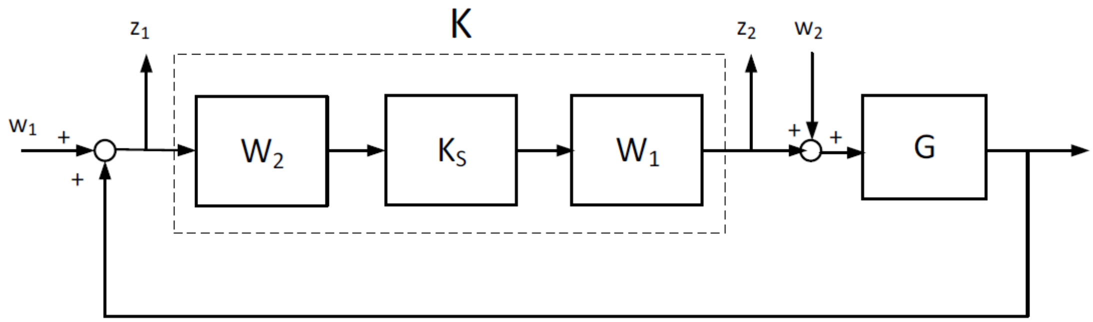


**Figure 5.2:** Normalised Coprime Factor Synthesis $H_\infty$ Design Interconnect

then it is apparent that implementing controller K on plant G is equivalent to implementing controller $K_S$ on plant $G_S$.

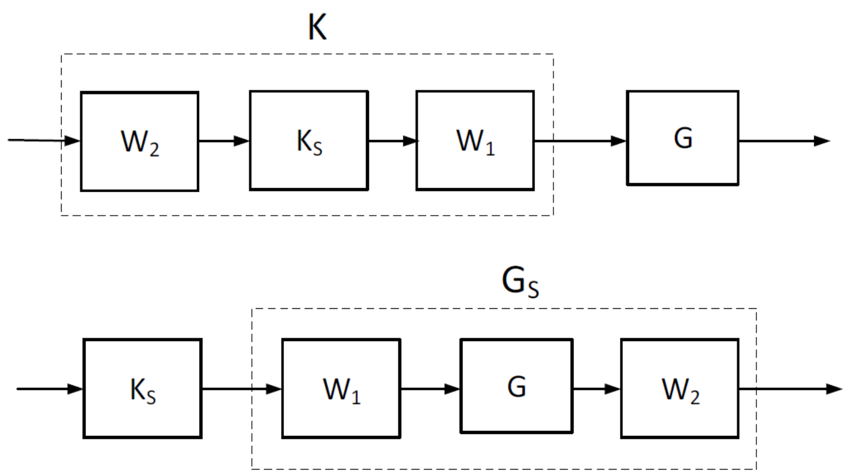


**Figure 5.3:** Equivalent Loop Gains for Weighted Controller and Plant

The $H_\infty$ Optimal Controller Synthesis solution provides $K_S$ solving for the global minimum of:

$$\inf_{K_S} \left\| \begin{pmatrix} I \\ K_S \end{pmatrix} (I - G_S K_S)^{-1} (I \quad G_S) \right\|_\infty = \gamma$$

and the resulting controller $K_S$ provides robust stability to coprime factor uncertainty for the shaped plant $G_S$, with robustness level $\gamma$. Furthermore, it can be shown that $K_S$ does not substantially affect the loop shape magnitude for frequencies where the gain of $G_S$ is either large or small. At the same time $K_S$ guarantees good stability margins (gain and phase) for $G_S$ in the crossover region.

The net effect of the above is that one can think of choosing the weights $W_1, W_2$ so that the shaped plant $G_S = W_2GW_1$ has the desired loop shape magnitude. The controller $K_S$ will provide good stability margins while keeping the loop shape magnitude as close as possible to the desired specification. That means that the controller $K = W_1K_SW_2$ will provide that same loop shape magnitude and stability margins for the original plant $G$. Hence the term Loopshaping – one simply chooses the desired loop shape magnitude and the $H_\infty$ Optimization solves for the controller that delivers as close to that as it can, whilst still having good robust stability margins.

There are several features of this approach that are highly desirable. Firstly, as mentioned, even though we are using a sophisticated $H_\infty$ optimization approach (which guarantees to find the global optimum controller), the design specification is simply based on familiar loop gain attributes. Secondly there is no detailed a-priori uncertainty characterization required, and yet the controller delivers guaranteed robustness against coprime factor uncertainty. Finally note that the controller is calculated as $K = W_1K_SW_2$, so that the weights $W_1$, $W_2$ are specifically included in the controller. This means that any specific desired features in the controller (e.g., notches, integral action), can be enforced simply by directly incorporating these features in the design weights $W_1$, $W_2$.

Note also that the $H_\infty$ Loopshaping optimization is highly numerically robust (e.g., allows for pure integrators in the loop) and it runs in seconds even on moderate computational resources. Finally, note that for SISO systems all transfer functions commute, and hence we can easily re-arrange to $G_S = W_2GW_1 = GW_2W_1 = GW$ (with $W = W_2W_1$) for which the corresponding controller is simply $K = WK_S$. Thus in the SISO case we need only design a single weight $W$ so that $GW$ has the desired loopshape magnitude.

## 5.4 Nelder-Mead Simplex Optimization

In order to characterize robust stability properties from the controller design, an approach for stability region calculation was developed in collaboration with AE. The idea was to generate a grid of Power Setpoint (and 50 ohm points) versus the cable length Zrotation angle. This grid shows the stability regions through colors. We chose the unstable regions per row optimize the controller weights till the controller is stable for that particular configuration and move on towards the right till the end of the row. The optimization used here is the Nelder Mead Simplex method.

### 5.4.1 Nelder-Mead Algorithm

This algorithm minimizes a nonlinear function without the use of derivatives which falls under direct search methods. This method minimizes a function $f(x) \in R^n$. We require four coefficients that are needed to be provided to define this approach, namely reflection($\rho$), expansion($\chi$), contraction($\gamma$), and shrinkage($\sigma$), where $\rho > 0$, $\chi > 1$, $0 < \gamma < 1$ and $0 < \sigma < 1$.
The default values are $\rho = 1$, $\chi = 2$, $\gamma = \frac{1}{2}$ and $\sigma = \frac{1}{2}$. At any iteration k, a simplex $\Delta_k$ is given with vertices labeled as $x_1^{(k)}$, $x_2^{(k)}$, ..., $x_{n+1}^k$. We order these vertices such that $f_1^{(k)} \leq f_2^{(k)} \leq ... \leq f_{n+1}^{(k)}$. Where $f_i^{(k)}$ is the function of $x_i$ we want to minimize. Since $f_1^{(k)}$ is the lowest, $x_1$ is considered to be the 'best point' and the point at $f_{n+1}^{(k)}$ as the 'worst point'. The following describes the flow of one iteration:

- Order: Order the vertices to follow the function order mentioned above.

- Reflection: Calculate the reflection point using the formula, $x_r = x_0 + \alpha(x_0 - x_{n+1})$, where $x_0$ is the centroid of all points except $x_{n+1}$. The iteration is complete if $x_r$ is the new 'best point'. If it is not the 'best point', we replace the 'worst point' by the reflected point and continue with the first step.

- Expansion: If the reflected point is the 'best point', calculate the expansion point by the formula, $x_e = x_0 + \gamma(x_r - x_0)$. If this point is better than the reflection point, we replace the 'worst point' by expansion point. If not we replace it with the reflected point.

- Contraction: If $f(x_r) \geq f(x_n)$, calculate the contraction point by the formula, $x_c = x_0 + \rho(x_{n+1} - x_0)$. If this point is better than the 'worst point', we replace the 'worst point' by contracted point.

- Shrink: Replacing all the points except the 'best point' by the formula, $x_i = x_1 + \sigma(x_i - x_1)$.

To terminate the iterations, we may select a tolerance level or use a default tolerance level. The implementation of this optimization method is via MATLAB 'fminsearch' command.

# Chapter 6

# Application to AE System and Results

## 6.1 Metrics

### 6.1.1 Implementation

As discussed earlier that development of metrics was essential to quantify the tradeoffs between average and realtime tracking. These metrics provide a numerical confirmation of the tradeoffs and the efficiency of the controllers. This section deals with generating metrics for the AE system and in the following sections, these metrics will be generated for every controller CSU developed within the AE system.

Matlab scripts have been developed to interface with the existing AE simulation tools and calculate these metrics for desired scenarios. The metrics define the relation between the power setpoint and power delivered. Note that the metrics are calculated after the simulation ends. In the scripts the following variations for numerical metrics have been implemented (together with corresponding notation):

i ‘Mean Absolute Error Unmodified’ corresponds to MAE when the setpoint modification algorithm is inactive.

ii ‘Mean Absolute Error Modified’ corresponds to MAE when the setpoint modification algorithm is active.

iii ‘Mean Real Error Unmodified’ corresponds to ME when the setpoint modification algorithm is inactive.

iv ‘Mean Real Error Modified’ corresponds to ME when the setpoint modification algorithm is active.

In each of these metrics, we’re calculating metrics with respect to ‘e’ whether or not the internal setpoint is modified, i.e., the error is always with respect to the user-defined setpoint.

Note that for the plots there are also some additional variations, and corresponding notation:

i 'Mean Accumulated Error' corresponds to MAcE.

ii 'Mean Accumulated Error with Blanks' corresponds to MAcE where blanking is implemented for the first few samples (makes the smaller errors more visible).

iii 'Mean Accumulated Abs Error' corresponds to absolute values of MAcE.

iv 'Mean Accumulated Error Pre Blank' corresponds to MAcE but the blanking is implemented before the error calculation. This makes the errors more visible and smooths out the sharp transitions seen in 'Mean Accumulated Error with Blanks'.

v 'Moving Mean Error' corresponds to MME.

### 6.1.2 Simulation Studies

AE has developed a detailed simulation environment for the plasma-power supply system (see figure 3.1), including their full control system implementation (see figure 3.2). This simulation tool runs in the Matlab/Simulink environment, and the overall top-level model is shown in figure 6.1. In this thesis, we use the simulation model provided by AE as the plant - and take data from that. However, in practice this could all be done with experimental data from a real physical system.

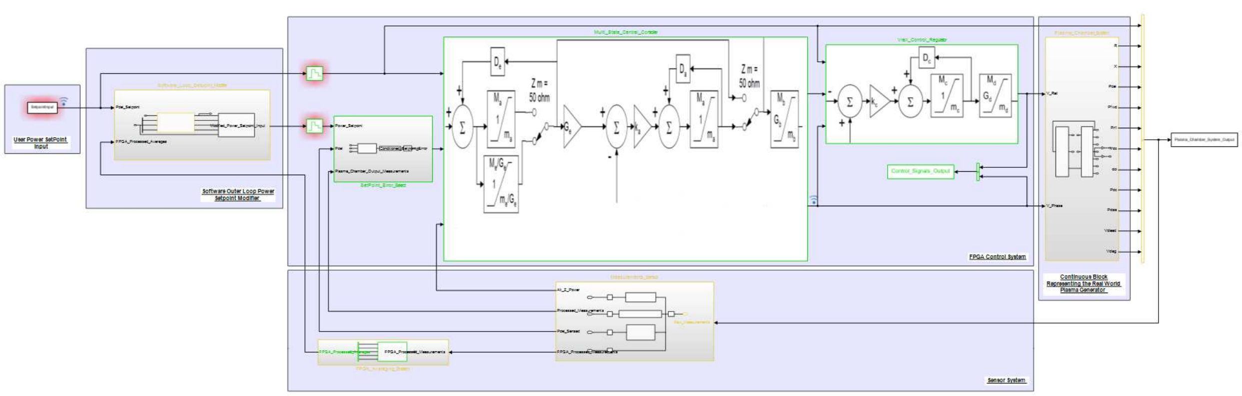

**Figure 6.1:** AE Plasma-Power Supply System Simulink Simulation Tool

We now consider several representative scenarios, all using this AE system, and analyze the performance of the system using, among other things, some of the metrics discussed earlier. Note that the focus of the current AE control approach is average tracking without average overshoot (and with acceptable control effort). Their controllers are designed with this in mind and are not attempting to deliver real-time tracking as a priority.

**Simulation Results for Ideal Case**

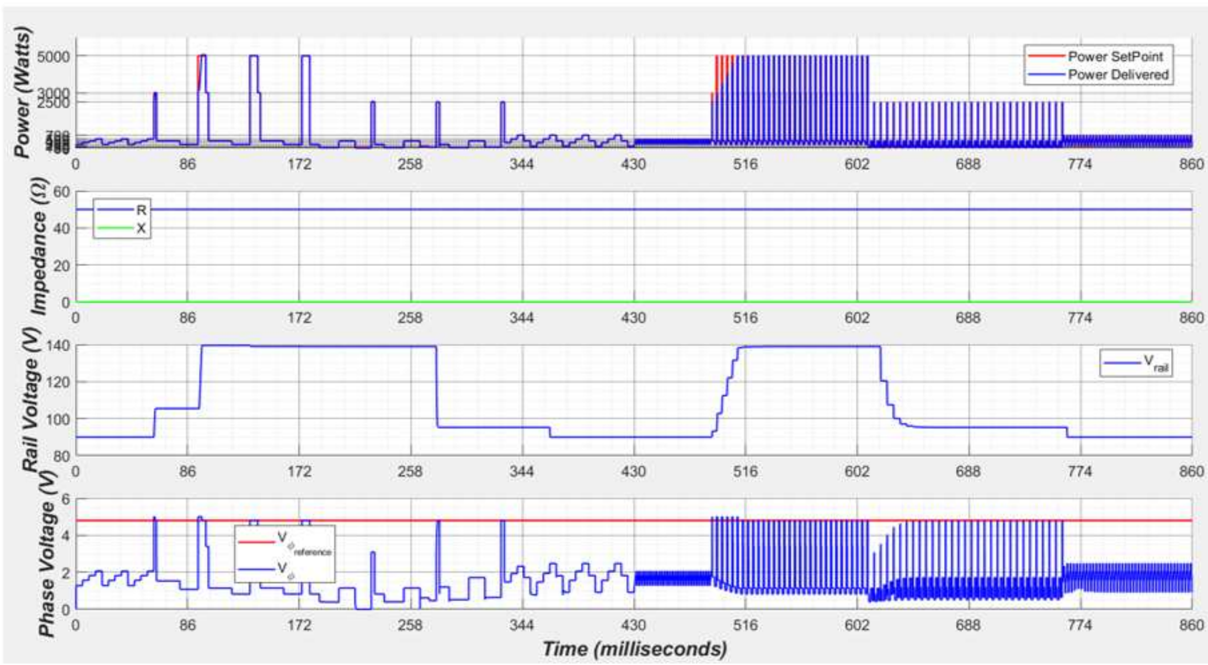


**Figure 6.2:** Real-Time Tracking Performance for Ideal Case

The ideal case is the AE system with the load having no dynamic impedance and matched impedance with the generator. This response should be ideal for this system.

The power setpoint profile is provided by AE and designed to test out the controller performance over a wide variety of conditions (large and small signals, low and high duty cycles, slow and fast variations). Figure 6.2 shows real time tracking response of the ideal case. The overall performance in this case appears to be very good. Figure 6.3 shows average tracking response of the ideal case. The average tracking locks in on the correct values with high accuracy and without


**Figure 6.3:** Average Tracking Performance for Ideal Case

overshoot. This usually happens fairly quickly but it can be seen (around 94ms and again around 490ms) that sometimes the average tracking is slower to converge, and you can see the corresponding errors in the real-time tracking during these periods. It appears that this often occurs when Vrail is slow to transition to the correct value for that state.

This system was simulated both with and without setpoint modification algorithms active, and the (numerical) metrics in table 6.1 were calculated for these scenarios.

**Table 6.1:** Numerical Tracking Metrics for Ideal Case

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 0.23285 | 1.35895 | 0.23268 | -0.07826 |
| 2 | 32.48597 | 34.45961 | 30.09951 | 28.60247 |
| 3 | 0.64017 | 7.27334 | 0.19296 | 1.00059 |
| 4 | 0.45103 | 2.25952 | -0.15003 | 0.35116 |
| 5 | 0.12803 | 0.49109 | 0.12784 | -0.12074 |
| 6 | 35.07161 | 35.51380 | 31.85883 | 32.28396 |
| 7 | 1.80869 | 3.55465 | 0.56000 | -0.81988 |
| 8 | 0.47146 | 1.91627 | -0.17043 | -0.55379 |

Upon examining these metrics, a comparison of the ‘error modified’ to ‘error unmodified’ columns shows that the setpoint modification algorithm in this case offers an improvement versus the unmodified setpoint for some pulse cycles (but not all). This means that it may offer an improvement in average tracking for some cases. However, when we make this comparison for the ‘absolute error’ columns, we see that the absolute errors are not minimized when the setpoint modification algorithm is added. This shows that the current controller does not emphasize the optimization of real time tracking.

Now consider the evolution of the error over time. This can be seen for a variety of metrics in figure 6.4. The top plot shows the actual vales of Pdeli and Pdelsetpointi, together with various metrics (including ‘Error’ and ‘Absolute Error’ metrics) on the same scale. Once again the performance is pretty good across the board, though the (inevitable) sharp transitions that occur without blanking are apparent. In figure 6.5, we zoom in at one of the transitions of a pulse cycle. Mean Accumulated Error with Blanks corresponds to calculation of the metrics and equating the metrics in the blanking range to zero, Mean Accumulated Error corresponds to calculation of metrics, Mean Accumulated Abs Error corresponds to calculation of metrics with absolute errors, Mean Accumulated Error Pre Blank corresponds to calculation of metrics where we blank first and then calculate the metrics.

Now the effect of blanking is clearly apparent in terms of avoiding very large transitions caused by the initial transients (note that the error is bound to be almost 100% for the first few samples after a sharp transition).

**Simulation for a system with aliasing**

Here we change the averaging bandwidth and enable old averaging systems which has the effect of causing an aliasing-type response in the system performance. Old averaging system is similar to the one described in the ideal case. Although in this case, the software interrupt time (8ms) goes slower than the FPGA sampling time (80ns) and so the average is not calculated in time. This leads to slower response time as long as the average is concerned.

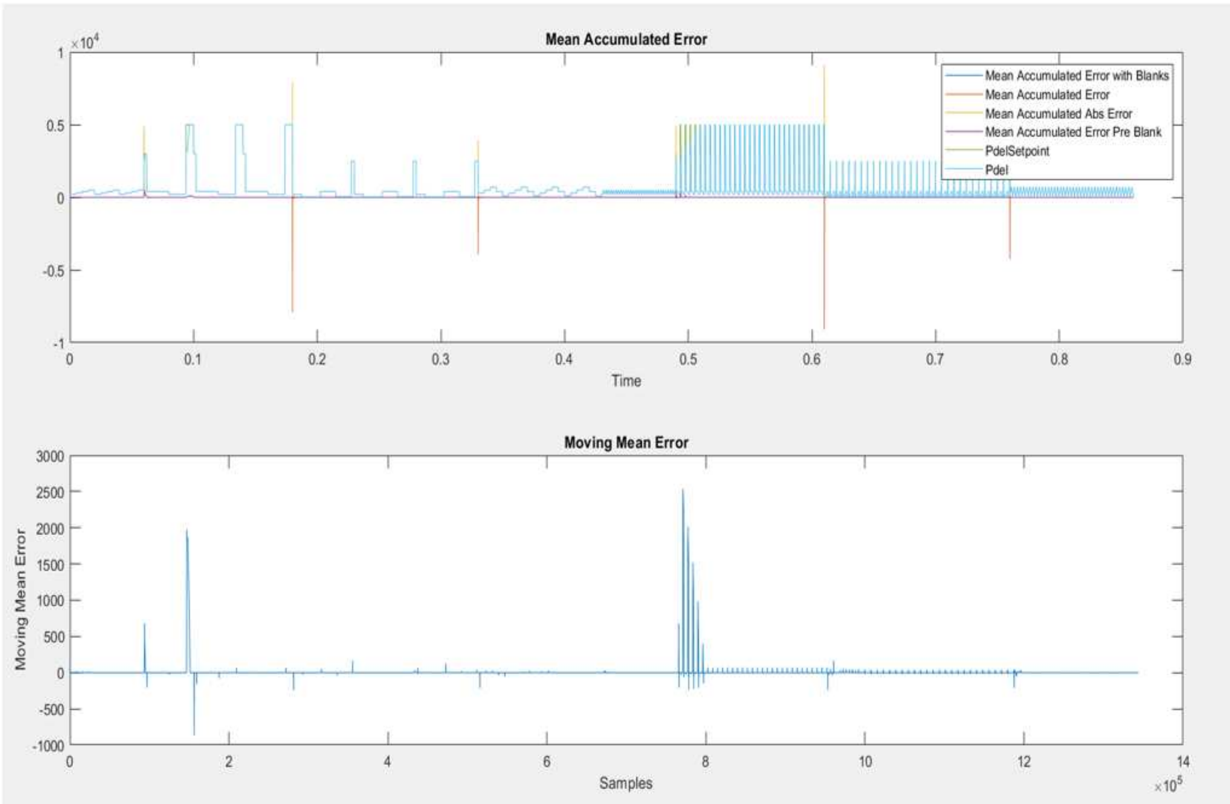


**Figure 6.4:** Moving Average Errors for Ideal Case

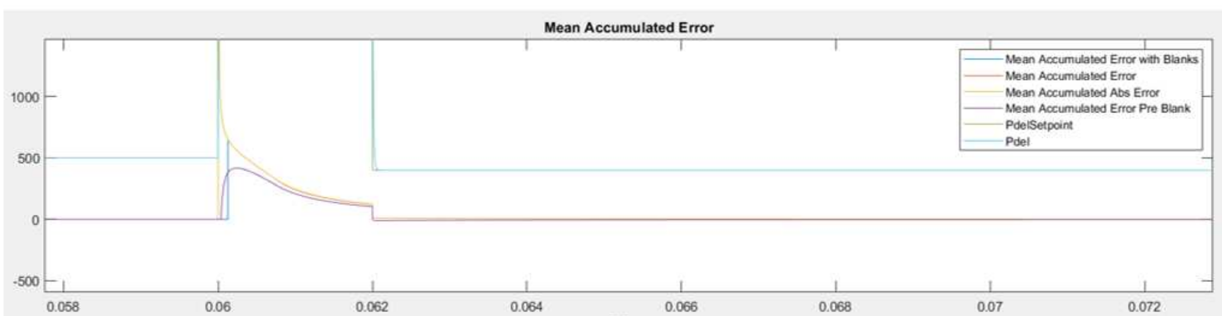


**Figure 6.5:** Zoomed In Moving Average Errors for Ideal Case

Examining the plots in figures 6.6, 6.7 the performance is seen to be markedly worse overall. Real-time tracking (in figure 6.6) can be seen to be somewhat worse at many points in the plot. However, average tracking performance (in figure 6.7) is dramatically worse and does not really deliver acceptable average tracking performance at all. As a result looking at the metrics in table 6.2 they can be seen to be worse across the board versus the ideal case in figure 6.1. Comparing

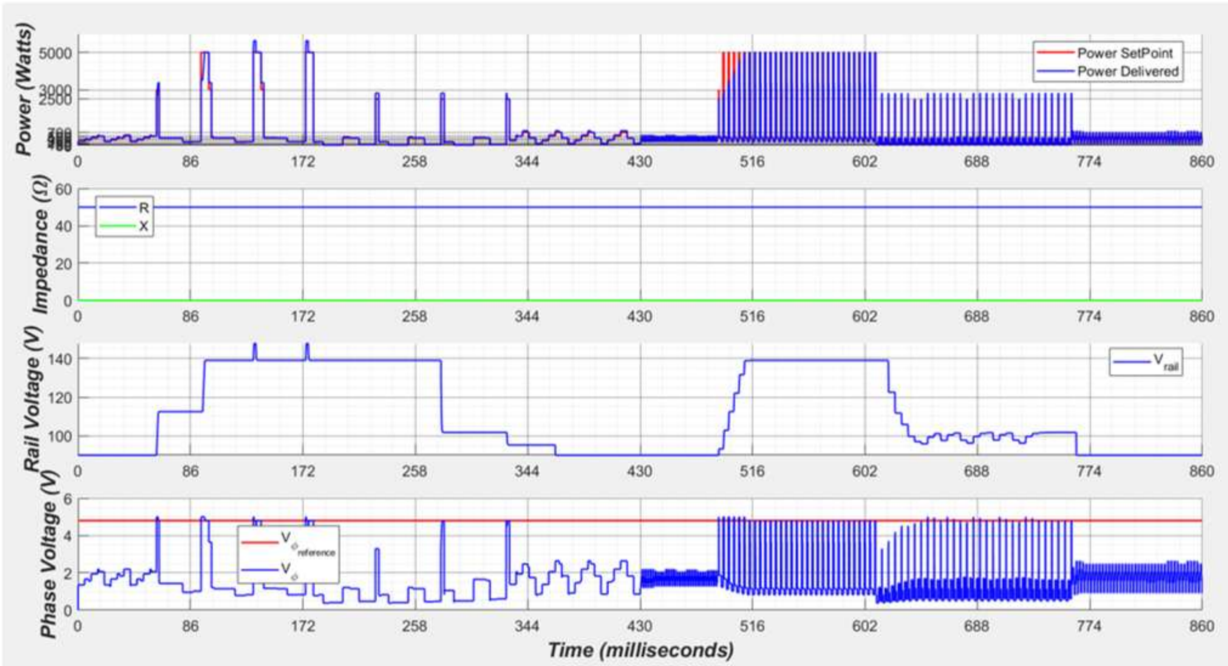


**Figure 6.6:** Real-Time Tracking Performance for System with Aliasing

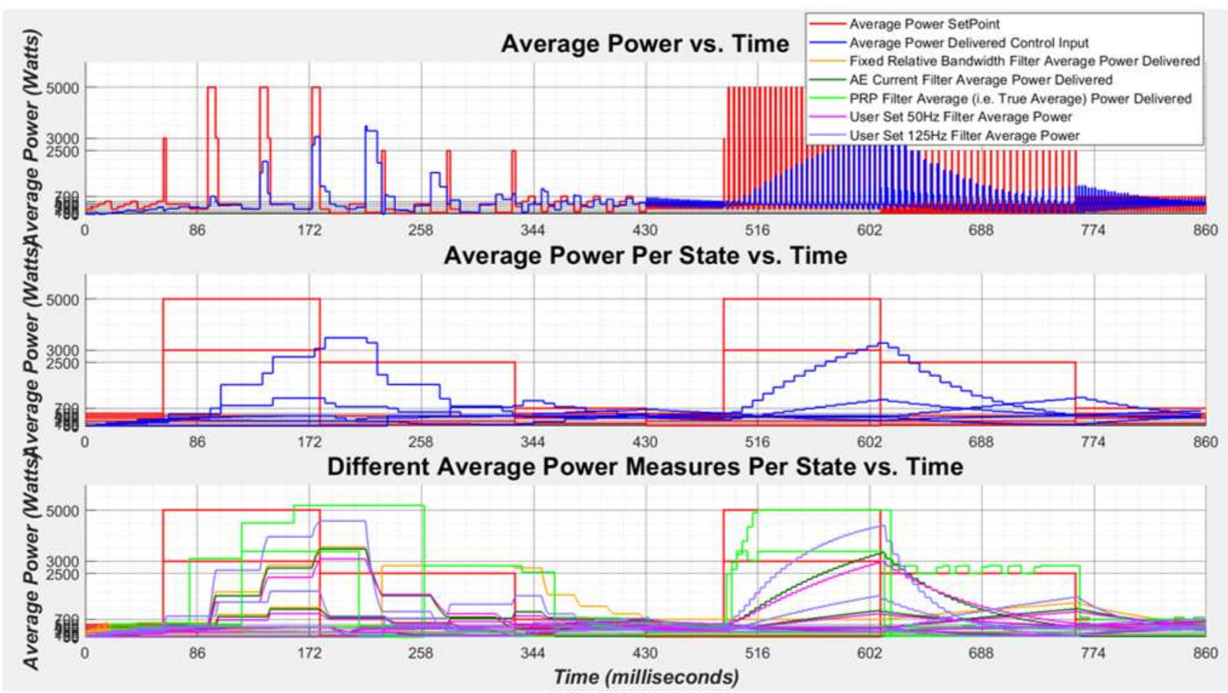


**Figure 6.7:** Average Tracking Performance for System with Aliasing

'mean_absolute_error_unmodified' to 'mean_absolute_error_modified', we find that the modification does not improve real time tracking but does well for the average tracking ('mean_real_error').

**Table 6.2:** Numerical Tracking Metrics for System with Aliasing

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 0.23285 | 29.94002 | 0.23268 | -29.65120 |
| 2 | 32.48597 | 68.70882 | 30.09951 | -6.49359 |
| 3 | 0.64017 | 19.55861 | 0.19296 | -13.08889 |
| 4 | 0.45103 | 31.04547 | -0.15003 | -15.78391 |
| 5 | 0.12803 | 12.48896 | 0.12784 | -3.03751 |
| 6 | 35.07161 | 54.69414 | 31.85883 | 12.12480 |
| 7 | 1.80869 | 15.95196 | 0.56000 | -13.56417 |
| 8 | 0.47146 | 7.42021 | -0.17043 | -3.72718 |

The inferior performance can also be seen in the time series plots of the metrics shown in figure 6.8.

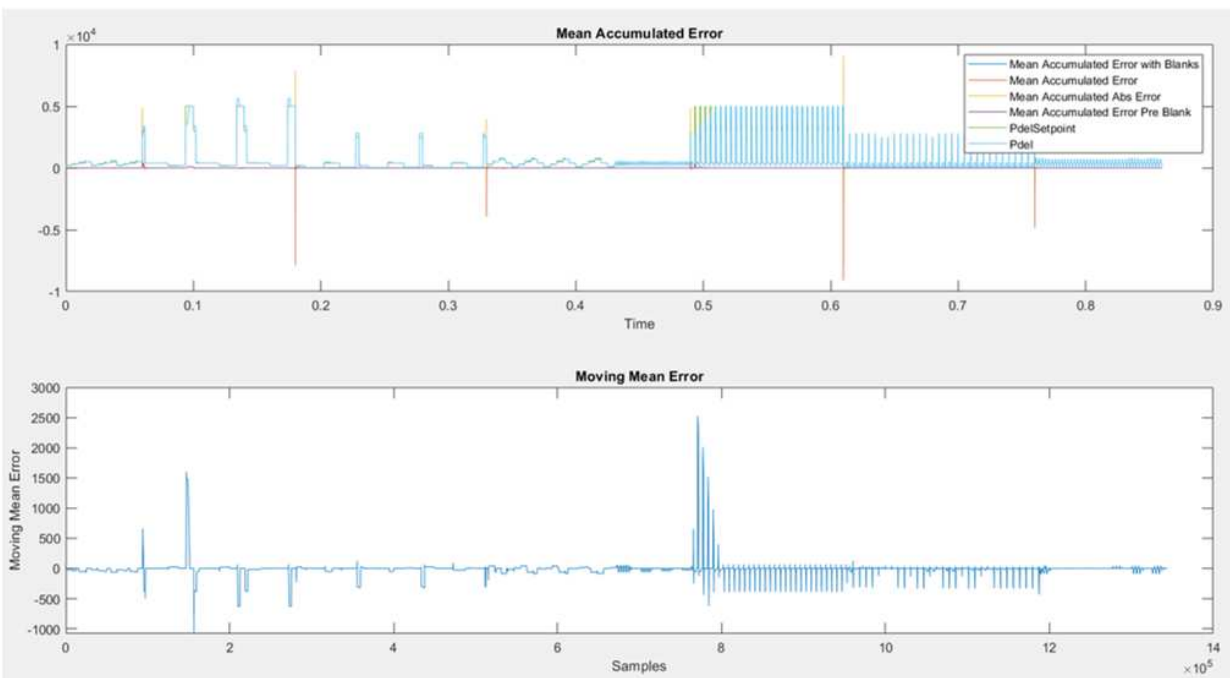


**Figure 6.8:** Moving Average Errors for System with Aliasing

### Simulation Results for an Unstable System

Here we take the previous system and now limit the maximum value of Vrail to 120V (140V was used for the previous cases). This pushes the system out of its stability region and an unstable system response results.

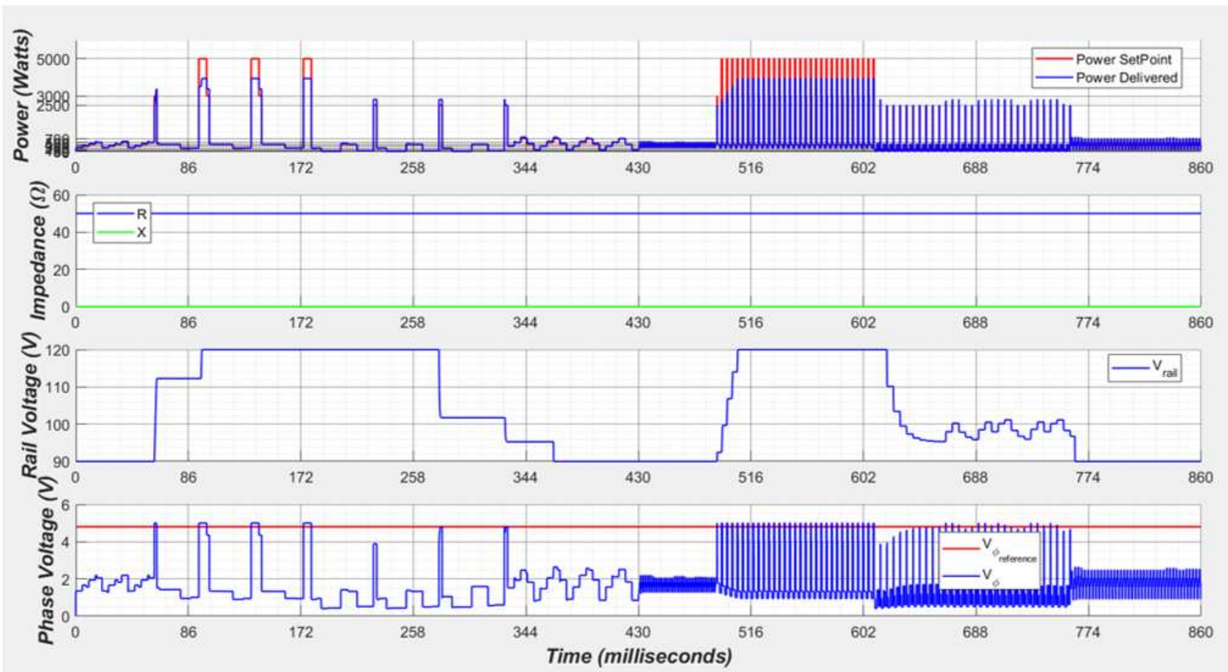


**Figure 6.9:** Real-Time Tracking Performance for an Unstable System

The real time tracking performance (see figure 6.9), and the average tracking performance (see figure 6.10) are both very poor. The corresponding numerical metrics in table 6.3, and the time-series plots in figure 6.11, also indicate very poor performance, so that again a decision based on these metrics would reject this control design.


**Figure 6.10:** Average Tracking Performance for an Unstable System

**Table 6.3:** Numerical Metrics for Unstable System

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 0.23285 | 29.94002 | 0.23268 | -29.65120 |
| 2 | 174.91792 | 190.43936 | 173.6272 | 155.70524 |
| 3 | 0.62724 | 19.57364 | 0.12977 | -13.15205 |
| 4 | 0.45104 | 36.49415 | -0.15001 | -21.31691 |
| 5 | 0.12803 | 12.67792 | 0.12784 | -3.22647 |
| 6 | 178.20187 | 98.85725 | 175.87878 | 155.13668 |
| 7 | 1.77832 | 15.94393 | 0.48292 | -13.68088 |
| 8 | 0.47146 | 11.46205 | -0.17043 | -7.76907 |

### 6.1.3 Discussion

The metrics are calculated as (Pdelsetpoint - Pdel). Thus, negative error corresponds to overshoot which is highly undesirable. Whichever metrics we choose to use, a good controller should have errors close to zero, and must have few negative errors. However there are many choices for implementing controller tradeoffs subject to these basic requirements. For example we could consider controllers designed based on any of the following optimizations:

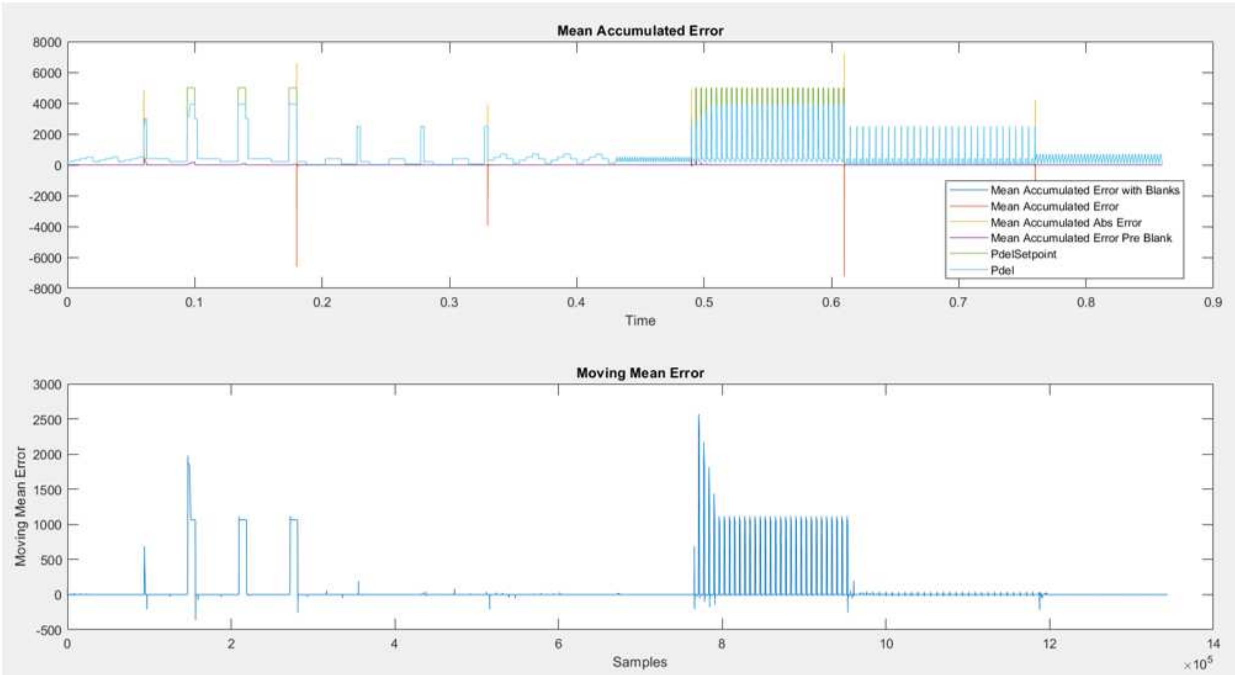


**Figure 6.11:** Moving Average Errors for Unstable System

i Minimize $|ME|$, subject to $ME \geq -\varepsilon_1$ and $MAcEj \geq -\varepsilon_2 \forall j$

ii Minimize $MAE$, subject to $max_i|OS_i| \leq \varepsilon_1$and $max_i|US_i| \leq \varepsilon_2$

iii Minimize $MAcE$, subject to $e \to 0$ and $e \geq -\varepsilon$

iv Minimize $MME$, subject to $e \to 0$ and $e \geq -\varepsilon$

where each $\epsilon i \geq 0$ and OS and US denote overshoot and undershoot respectively. Each of these approaches would yield a different controller design. For instance option i) is essentially the current AE approach, which focuses solely on the average tracking performance (as discussed earlier). On the other hand option ii) would focus on real-time tracking alone and not consider average tracking. One could then contemplate blended approaches which attempt to achieve some reasonable level of performance for both average and real-time tracking. One could also consider variations on these metrics as in options iii) and iv) (where the metrics for MAcE and MME also give a graphical representation of the error mismatch to be minimized).

## 6.2 PID Control Results

Our first observation is that since the existing AE control scheme is (reset) integrator based, then we can consider this as the integral portion of a PID controller. Simply wrapping a PD controller in parallel with this existing AE integrator affords a PID controller. This is illustrated in the Vphase controller subsystem in figure 6.12 below. Note that in practice you would also need to change the anti-windup scheme for output limiting the integrator, since the PD portion of the controller now affects the overall output. We are not suggesting PID control as our recommended approach in the long run, and so we do not bother to do this here. However, we note there are a number of available approaches to implement this augmented anti-windup for output limiting, which could readily be adapted to fit this application if desired.

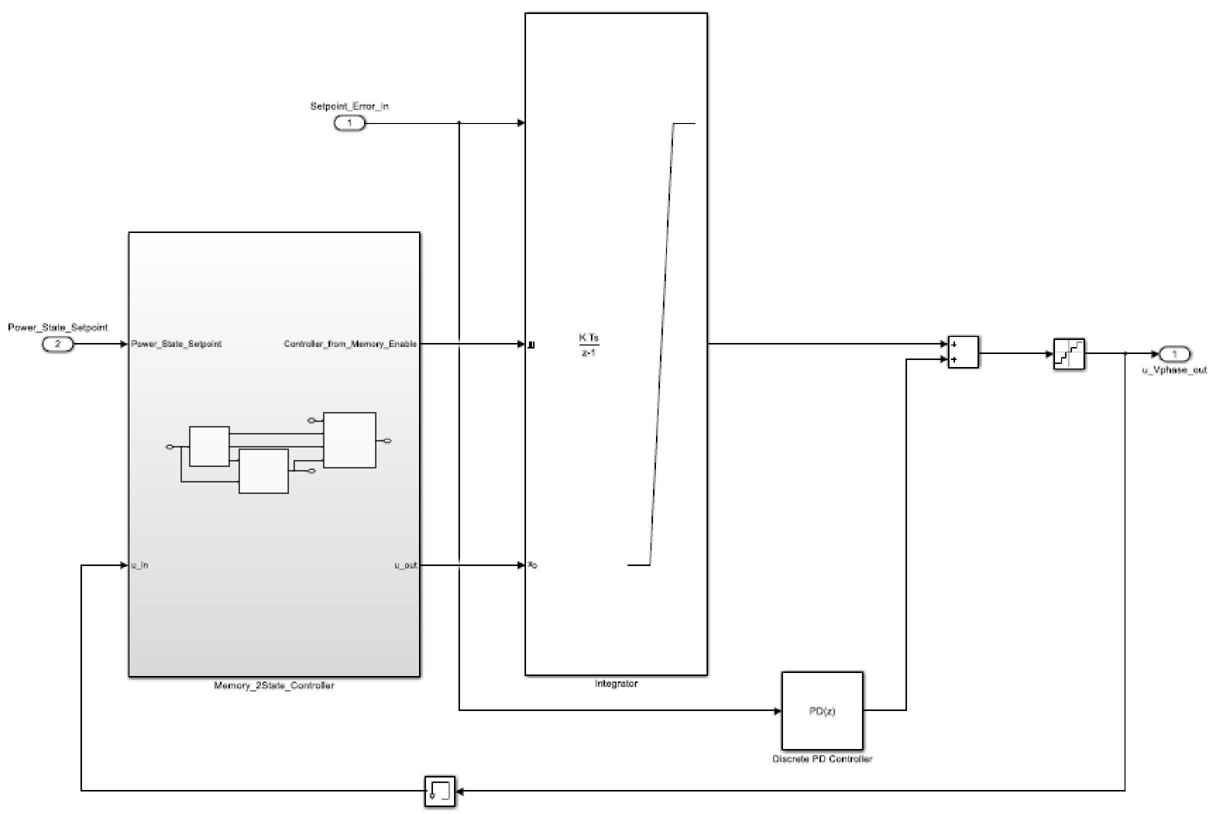


**Figure 6.12:** PID Controller in Vphase Subsystem

Next we observe that the Vrail control scheme also has a reset integrator at its heart. Hence this same reasoning can be applied there too. The implementaion of PID control in the Vrail subsystem (via PD control in parallel with the AE integrator) is illustrated in figure 6.13 (again we do not bother implementing the enhanced anti-windup here but it could be done).

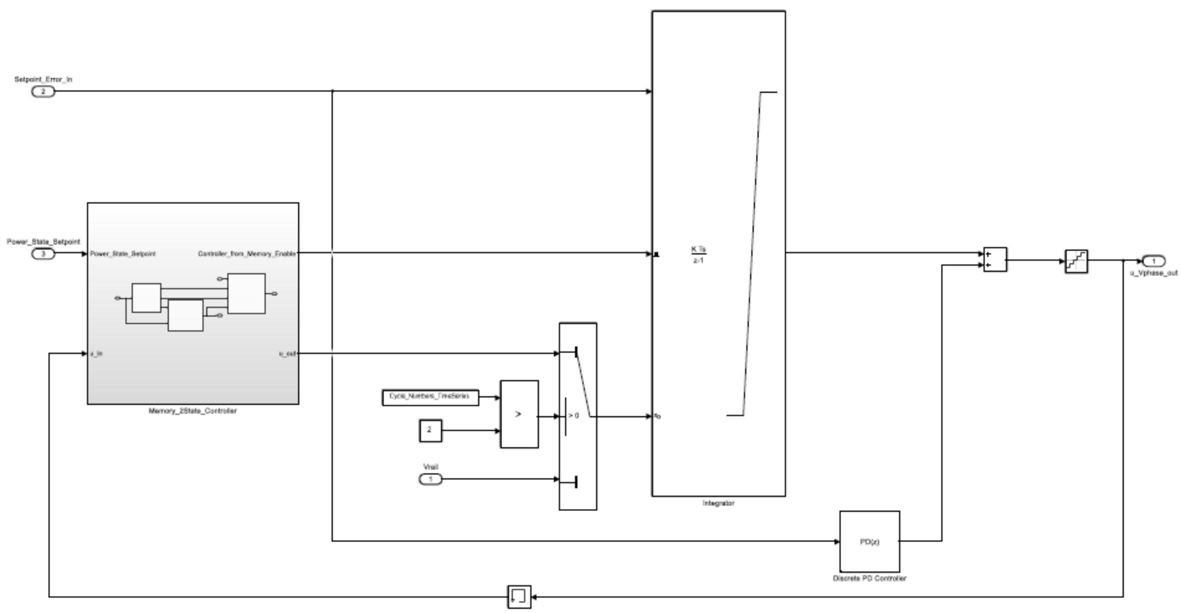


**Figure 6.13:** PID Controller in Vrail Subsystem

These PID controllers were implemented with gains tuned as:

Vphase PID gains: $K_P = 3.57 \times 10^{-8}, K_I = 4, K_D = 5 \times 10^{-8}$

Vrail PID gains: $K_P = 1.043 \times 10^{-6}, K_I = 15, K_D = 6.26 \times 10^{-10}$

Note that here the values of $K_P$ and $K_D$ are percentages of the overall gain from the setpoint error to the Vphase/Vrail outputs. Using these controllers results in the real-time tracking shown in figure 6.14, and the average tracking shown in figure 6.15. The corresponding metrics are shown in table 6.4.

It can be seen that the real-time tracking converges quickly due to the faster Vrail convergence, which may or may not be desirable from the AE hardware viewpoint. However there are some

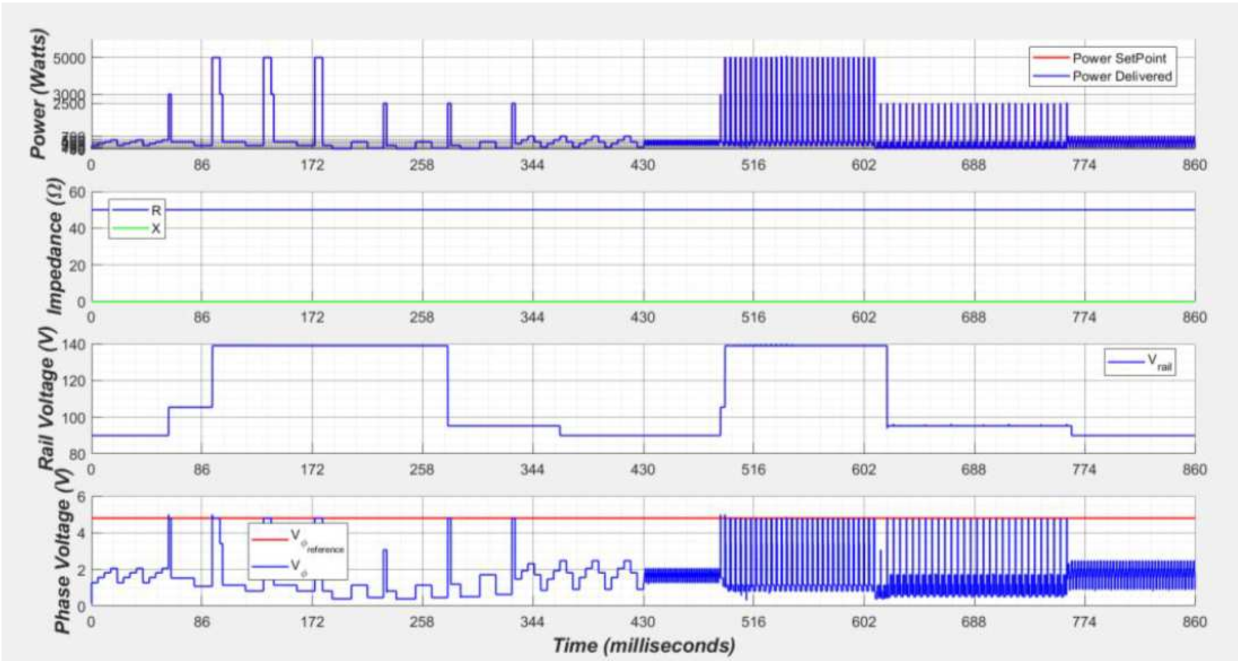


**Figure 6.14:** Matched Impedance Setpoint Tracking for PID Control Scheme


**Figure 6.15:** Matched Impedance Average Setpoint Tracking for PID Control Scheme

oscillations in the response and this is even clearer in the unmatched impedance case shown in figures 6.16 and 6.17, with the corresponding metrics shown in table 6.5.

**Table 6.4:** Metrics for PID Control in Matched Impedance Case

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 0.0573 | 1.1896 | 0.0528 | -0.2561 |
| 2 | 2.9939 | 5.6478 | 1.6931 | -0.4837 |
| 3 | 0.3052 | 6.9227 | 0.1460 | 0.9049 |
| 4 | 0.1434 | 1.9751 | 0.0244 | 0.5237 |
| 5 | 0.0719 | 0.4333 | 0.0091 | -0.2394 |
| 6 | 10.6200 | 11.6219 | 2.5923 | 3.5377 |
| 7 | 1.4954 | 3.3317 | 0.4785 | -0.9071 |
| 8 | 0.2496 | 1.6905 | -0.0139 | -0.3961 |

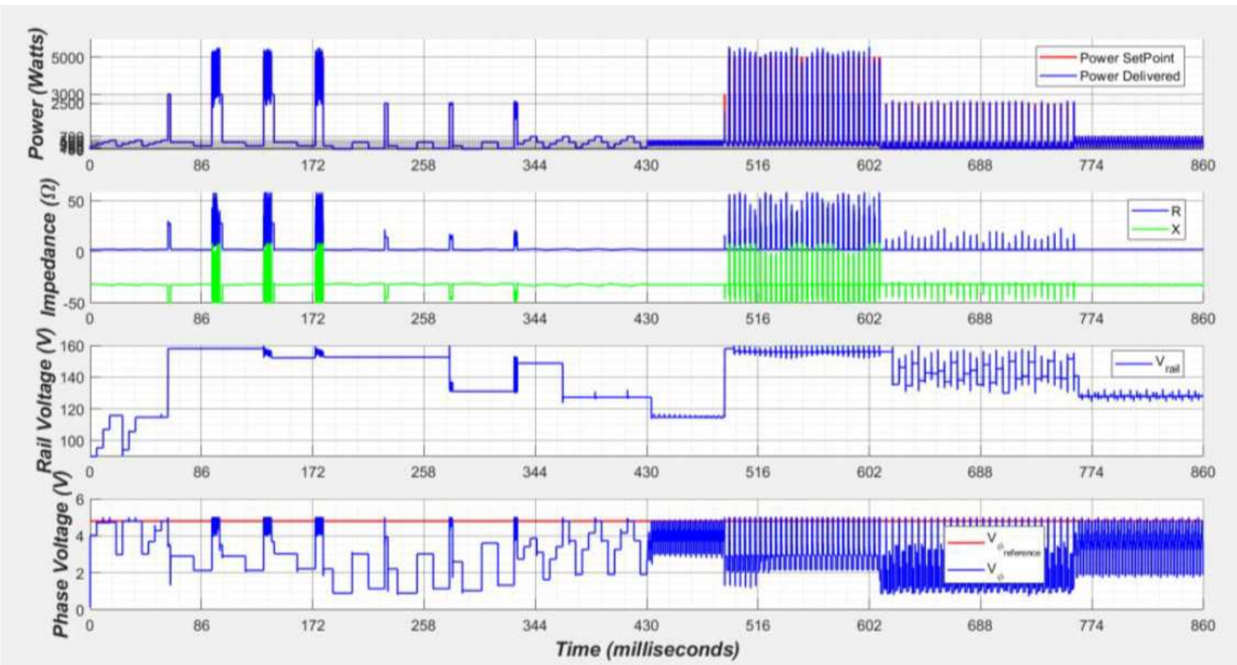


**Figure 6.16:** Unmatched Impedance Setpoint Tracking for PID Control Scheme

**Table 6.5:** Metrics for PID Control in Unmatched Impedance Case

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 1.0343 | 2.1372 | 0.0039 | -0.3026 |
| 2 | 174.6634 | 177.5027 | 155.7579 | 157.2271 |
| 3 | 16.8323 | 22.9285 | 15.3770 | 16.5507 |
| 4 | 0.5723 | 2.0761 | 0.1399 | 0.4336 |
| 5 | 1.7339 | 2.0076 | 0.3427 | 0.0937 |
| 6 | 217.2014 | 217.4469 | 188.9153 | 188.8889 |
| 7 | 36.4584 | 38.4644 | 31.1528 | 29.9012 |
| 8 | 3.6026 | 4.7425 | 0.5573 | 0.0622 |


**Figure 6.17:** Unmatched Impedance Average Tracking for PID Control Scheme

## 6.3 $H_\infty$ Loopshaping Control

### 6.3.1 Implementation

The current AE control strategy consists of a Vphase and a Vrail controller. The Vrail controller is a heuristic design, which takes into account a number of specific desirable features and constraints for Vrail. The Vrail controller takes the Vphase command and then adjusts Vrail until:

i The power setpoint is reached.

ii Vphase does not exceed the upper limit (4.8volts).

iii Vrail is as small as possible (subject to the above).

iv Vrail does not adjust too quickly or too often.

These properties are ensured by a set of heuristics for the Vrail controller, which have been developed by AE over the years. Note that this means that essentially Vrail is just a function of Vphase, so that the heart of the control approach is to design a good controller for Vphase. Note further that this means that we have a SISO control problem for Vphase, since the Vrail controller effectively just becomes part of the open-loop plant (simply a function of Vphase). Our focus then is to design a robust/optimal control approach for Vphase.

This reset scheme has advantages and disadvantages. It has been quite successful in delivering accurate average power tracking for periodic waveforms. However the transient response is not always very good and a more sophisticated approach might deliver better transient performance. Furthermore, the use of switched integrators can raise stability concerns, so it is not clear that it is a good strategy in the long run, particularly if one wants robustness to afford applicability to a wide variety of systems, applications, and operating conditions.

We have developed a control approach based on $H_\infty$ Loopshaping for AE systems. The first part of the approach is to develop a simplified model of the open-loop AE system (figure 6.18), which for reasons discussed earlier includes the (heuristic) Vrail controller. This simplified model will only be used for controller design purposes – we will still use the full accurate nonlinear simulation model for all controller testing.

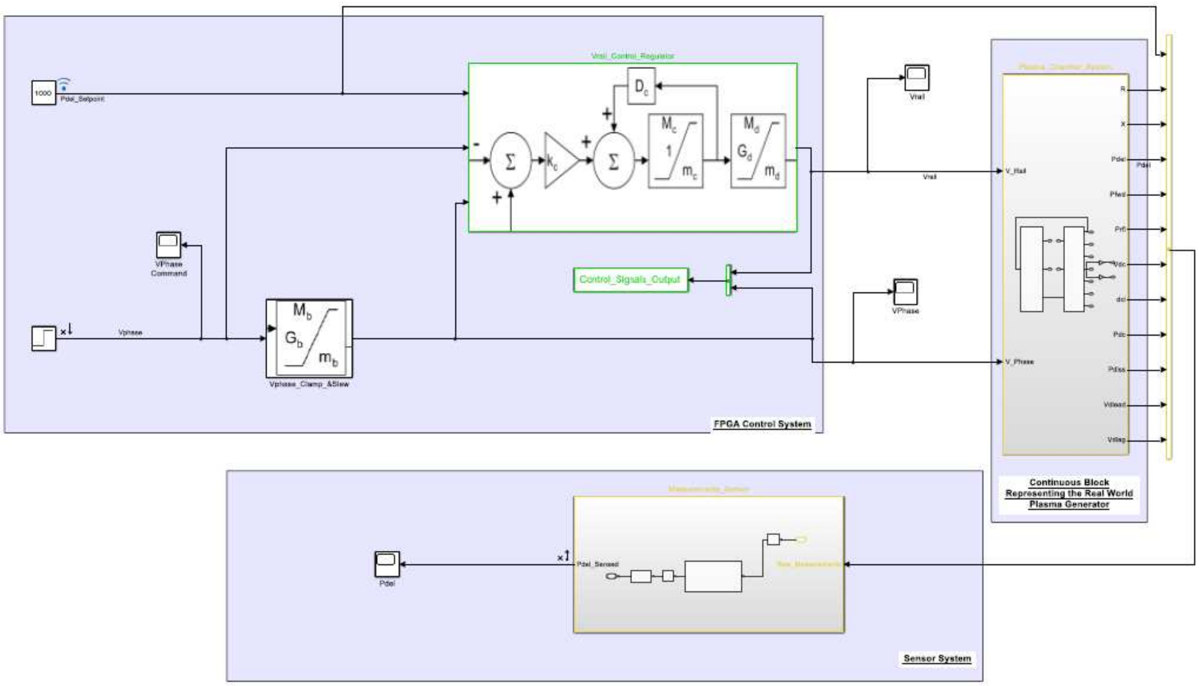


**Figure 6.18:** Open Loop Plant Model (includes Vrail Controller)

From this open loop model there are a number of different System Identification approaches that are viable. The Linear Analysis tool in Matlab/Simulink was used which allows for linear model extraction about different setpoints for nonlinear systems. These models were confirmed via step responses from Vphase input to Pdel (power delivered) output. It turns out that a very simple model of the form (for appropriate open-loop plant parameters $C_P$, $T_P$):

$$G(s) = \frac{C_P}{1 + sT_P}$$

is sufficient to describe the open-loop dynamics for the $H_\infty$ Loopshaping controller design approach. It remains to (determine the parameters $C_P$, $T_P$ and) design appropriate alternatives for the loopshaping weight $W(s)$. One set of designs we carried out considered weights of the form:

$$W(s) = C_k \left( \frac{1}{1 + sT_K} \right)^2 \left( K_P + \frac{K_I}{s} \right)$$

Note that this shapes the DC gain (via $C_K$), the bandwidth (via $T_K$), and also incorporates a Proportional-Integral (PI) controller for low frequency and steady state tracking. The proportional and integral gains ($K_P$, $K_I$) determine how aggressively the tracking is pursued.

Recall that this weight will be included in the controller K(s), and hence we guarantee that our controller includes integral action. Furthermore note that we can rewrite this weight as:

$$\begin{aligned} W(s) = C_k \left( \frac{1}{1 + sT_K} \right)^2 \left( \frac{K_P s + K_I}{s} \right) &= \left( \frac{C_K(K_P s + K_I)}{(1 + sT_K)^2} \right) \left( \frac{1}{s} \right) \\ &= W_{noi}(s) \left( \frac{1}{s} \right) \end{aligned}$$

where $W_{noi}(s)$ is a proper transfer function. Hence the final controller design (which includes this weight) can be implemented in the following form:

$$K(s) = K_{noi}(s) \left( \frac{1}{s} \right)$$

where $K_{noi}(s)$ is a proper controller, as illustrated in figure 6.19.

This affords us implementation options. To be specific we can either:

1. Implement the $H_\infty$ Loopshaping controller $K(s)$ as shown in figure 6.19 in place of the existing AE control scheme. Note that we can utilize output limits with anti-windup on the integrator to limit $Vphase$ as desired (0-5volts).

2. Utilize the current AE reset-based control scheme for the integrator portion of the above controller. In this case the $H_\infty$ Loopshaping controller $K_{noi}(s)$ acts as a prefilter to the existing control scheme.

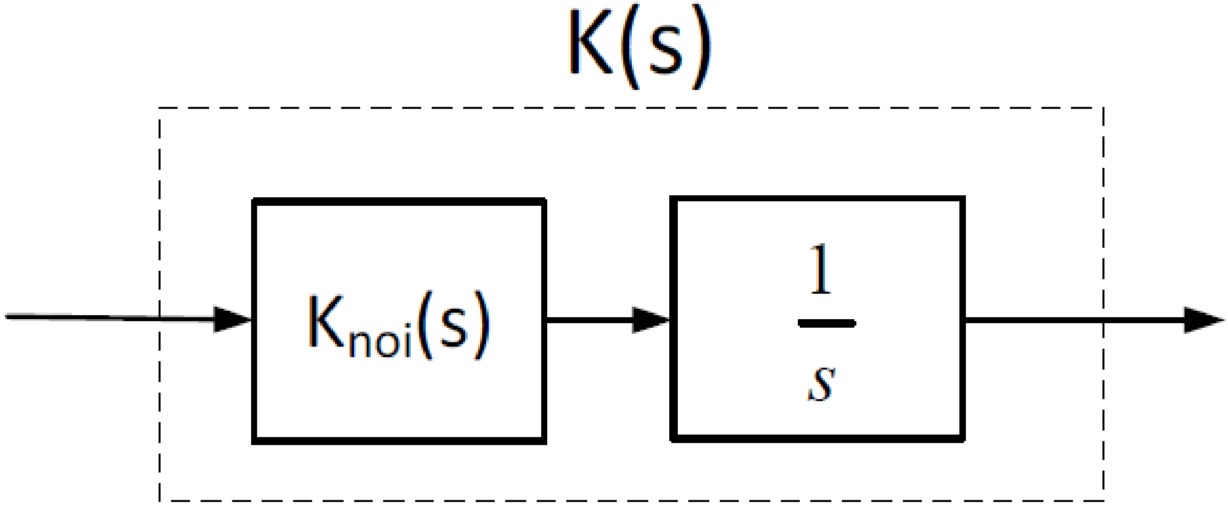


**Figure 6.19:** $H_\infty$ Loopshaping Controller as Cascade of Proper Filter and Integrator

As mentioned earlier there are pros and cons to utilizing the integrator reset-based AE control strategy and so we will investigate both of the above options. Note in either case we will transform the controller from continuous-time $K(s)$ to digital $K(z)$ via a Bilinear Transformation, operating at the usual sample period for the AE system (Tsample = 640ns).

### 6.3.2 Simulation Studies

In this section we develop some specific controller designs using the processes we have developed. As discussed earlier the designs are split into two broad groups, depending whether or not we incorporate the existing AE integrator reset-based control scheme.

#### $H_\infty$ Loopshaping Controller with AE Reset Integrator

Here the $H_\infty$ Loopshaping controller acts as a prefilter to the AE integrator scheme. The implementation in the Simulink model is illustrated in figure 6.20. The input to the $H_\infty$ Loopshaping controller is the '$Setpoint\_Error$' as expected by the control design.

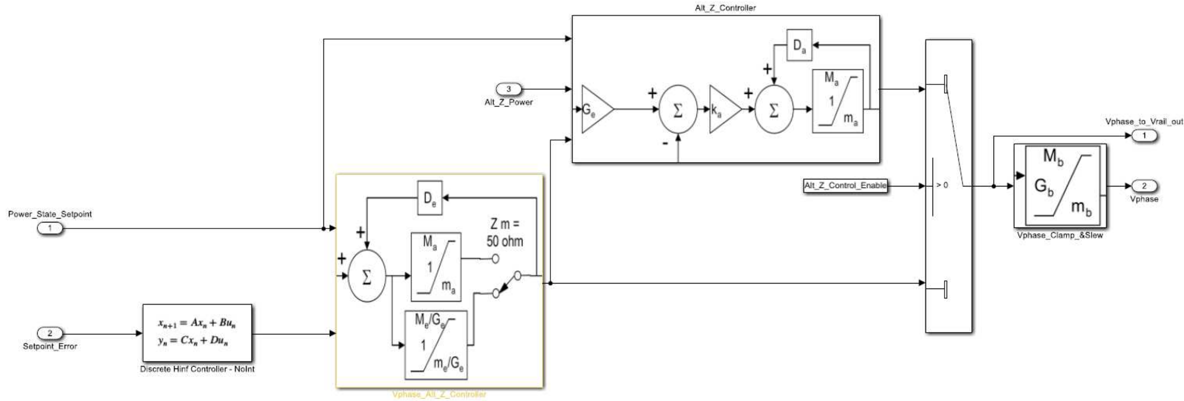


**Figure 6.20:** $H_\infty$ Loopshaping Controller Placement for System with AE Reset Integrator

For these designs, we utilize a linear approximation of the plasma system. The controller was designed using this approximate plant, chosen as

$$G(s) = \frac{86400}{6 \times 10^{-6}s + 1}$$

The high DC gain of the plant comes from the FPGA gain block in the '$Setpoint\_Error\_Select$' block which increases the overall gain of the loop. The open loop weighting transfer function was chosen as:

$$W(s) = \frac{1.1574 \times 10^{7}(s + 10^{5})}{s(s + 10^{6})^{2}}$$

Note that this includes rolloff and a PI stage as discussed earlier. Note also that the high loop gain pointed out above was compensated by choosing an appropriate weight. As discussed earlier we keep the existing AE heuristic Vrail algorithm. Since Vphase is used to generate Vrail, this makes the system SISO and allows us to also look at traditional loop gain measures of performance, stability and robustness (in addition to the rigorous Normalized Coprime Factor analysis presented earlier). The results from the controller design algorithm are discussed below. First the design plots are shown in figures 6.21 – 6.24. It can be seen that the controller delivers close to the desired loop gain (figure 6.22) while at the same time having good gain and phase (stability) margins (figure 6.24). Note that in Figure 6.23, the step response is generated directly from the

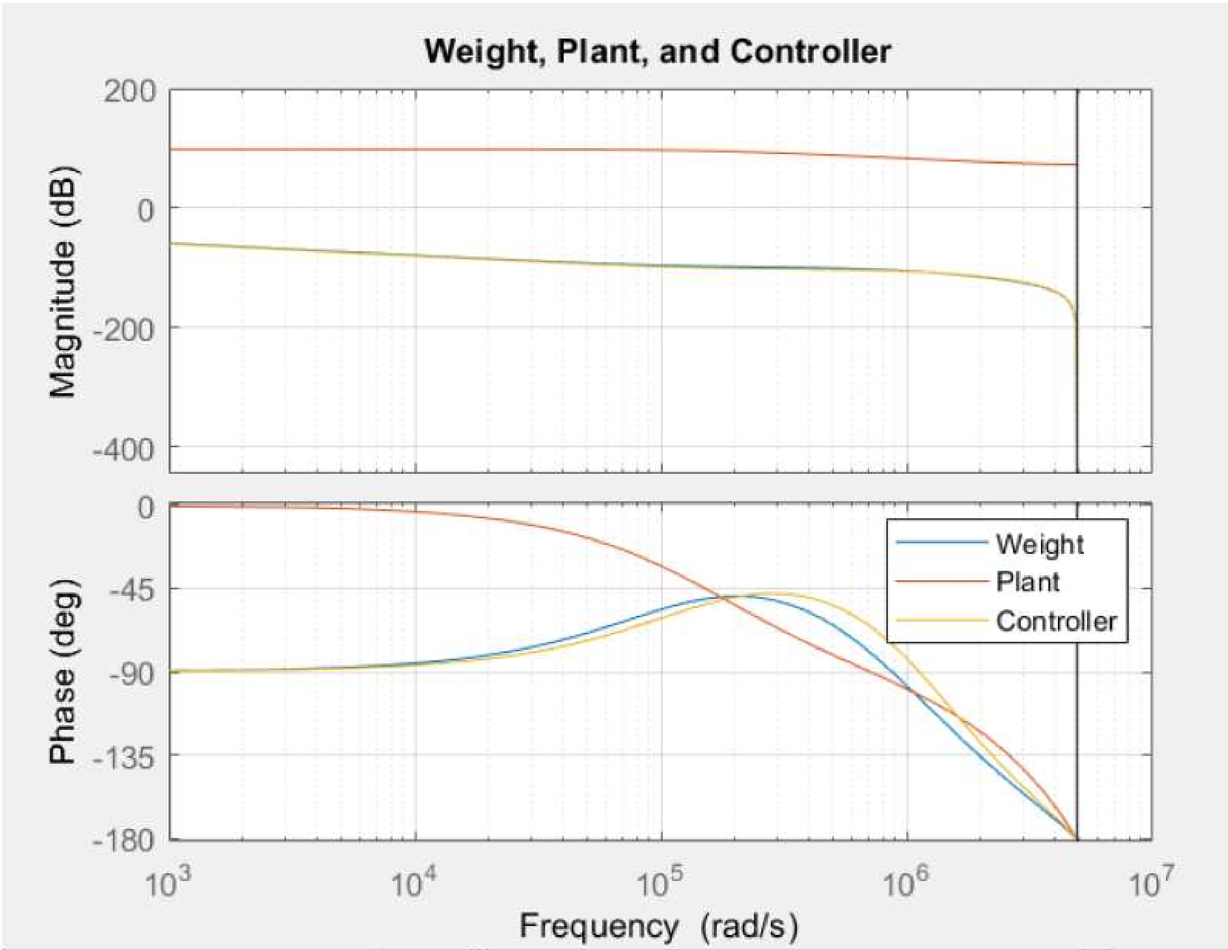


**Figure 6.21:** $H_\infty$ Loopshaping Bode Plots for Discrete Weight, Plant and Controller

optimization algorithm and includes only the approximate linear plant G(s) described above. The following simulation studies utilize the full nonlinear Simulink model of the AE system, together with the test pulse sequence provided by AE. Figures 6.25, 6.26 show the real-time and average setpoint tracking for the AE system using this controller and the AE test pulse setpoint sequence. Note here that controller was simulated for both matched impedance (between load and generator) and unmatched impedance.

We now calculate the metrics defined above. In each of these metrics, we're calculating metrics with respect to 'e' whether or not the internal setpoint is modified, i.e., the error is always with respect to the user-defined setpoint. The results for the control design simulations in figures 6.25, 6.26 are shown below in table 6.6.

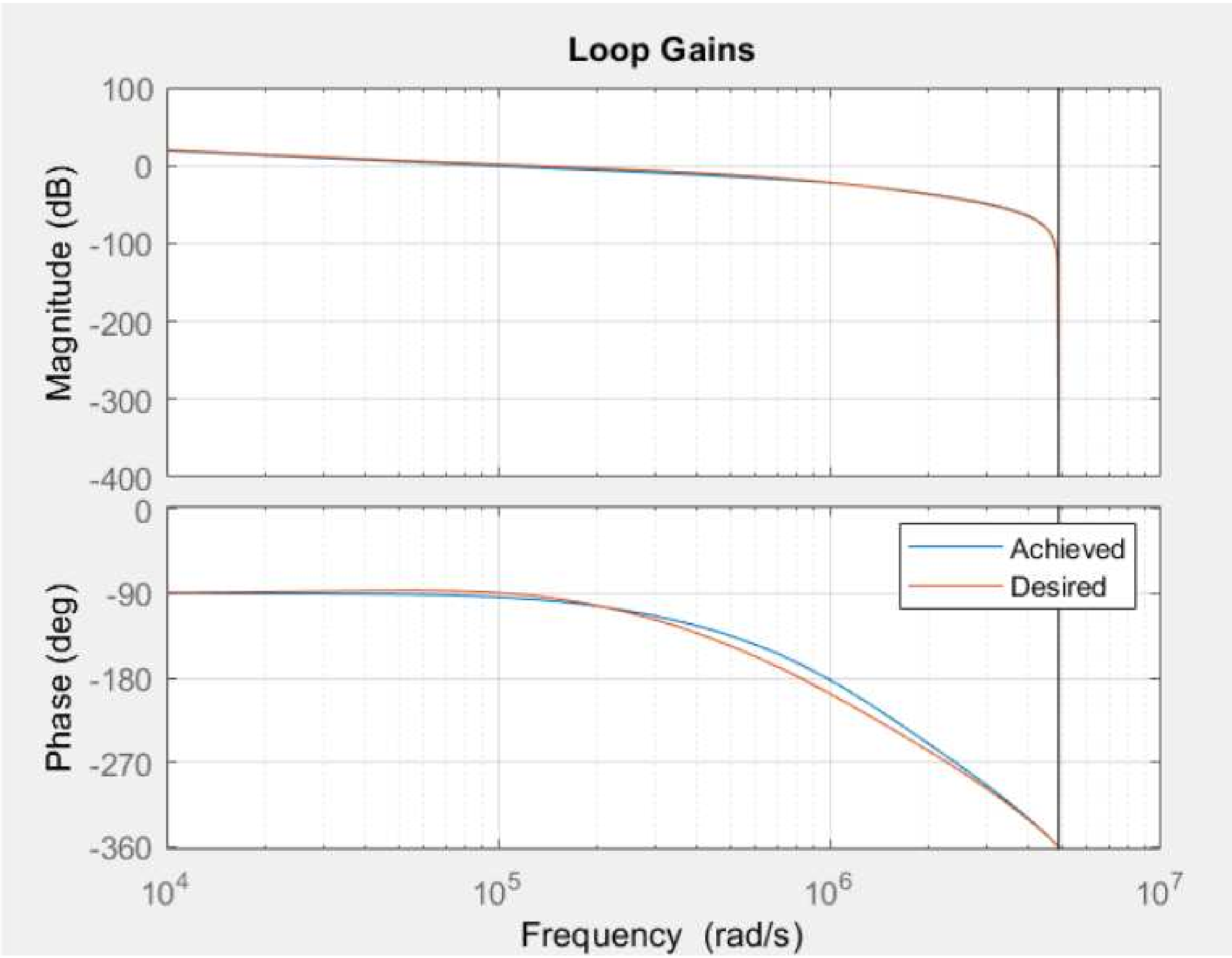


**Figure 6.22:** $H_\infty$ Loopshaping Achieved and Desired Loop Gains

**Table 6.6:** Metrics for Matched Impedance Case for $H_\infty$ Controller with Reset Integrator

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 0.2258 | 1.3478 | 0.2256 | -0.0862 |
| 2 | 32.5394 | 34.5219 | 29.9085 | 28.4117 |
| 3 | 0.6408 | 7.2503 | 0.2063 | 1.0156 |
| 4 | 0.4474 | 2.2474 | -0.1339 | 0.3702 |
| 5 | 0.1241 | 0.4872 | 0.1239 | -0.1247 |
| 6 | 35.0602 | 35.4454 | 31.6689 | 32.0989 |
| 7 | 1.8142 | 3.5490 | 0.5779 | -0.8093 |
| 8 | 0.4679 | 1.9124 | -0.1543 | -0.5379 |

This controller delivers a high level of performance here. A more challenging environment is provided by the unmatched impedance case. In this case the AE system is tending towards instability, and it is very hard to get any kind of acceptable performance. The results for this

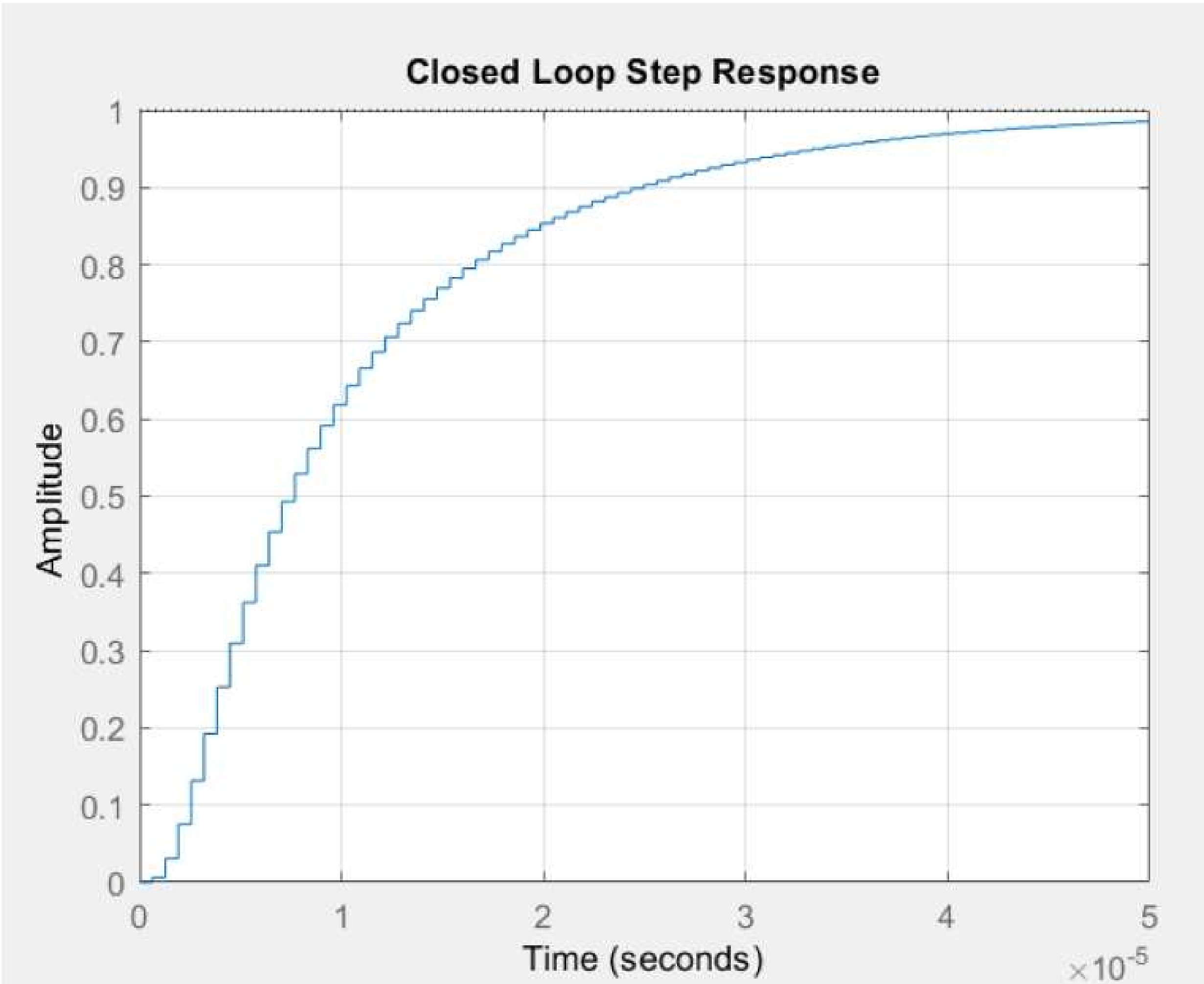


**Figure 6.23:** $H_\infty$ Loopshaping Step Response of the Closed Loop System with Approximate Plant

controller design approach ($H_\infty$ Loopshaping Controller with Reset Integrator) are shown in tables 6.27 – 6.7. The onset of instability is clear in the simulations, although for the most part the controller does contain this instability and still deliver some level of tracking performance from the AE system, even in the unmatched impedance case.

For comparison purposes, the values of these metrics are shown for the original AE reset-based integral controller. The exact same simulation package was used (though we don't bother to show the simulations themselves). The matched impedance case metrics are shown in table 6.8 and the unmatched impedance case in table 6.9 (both for the original AE reset-based integral controller).

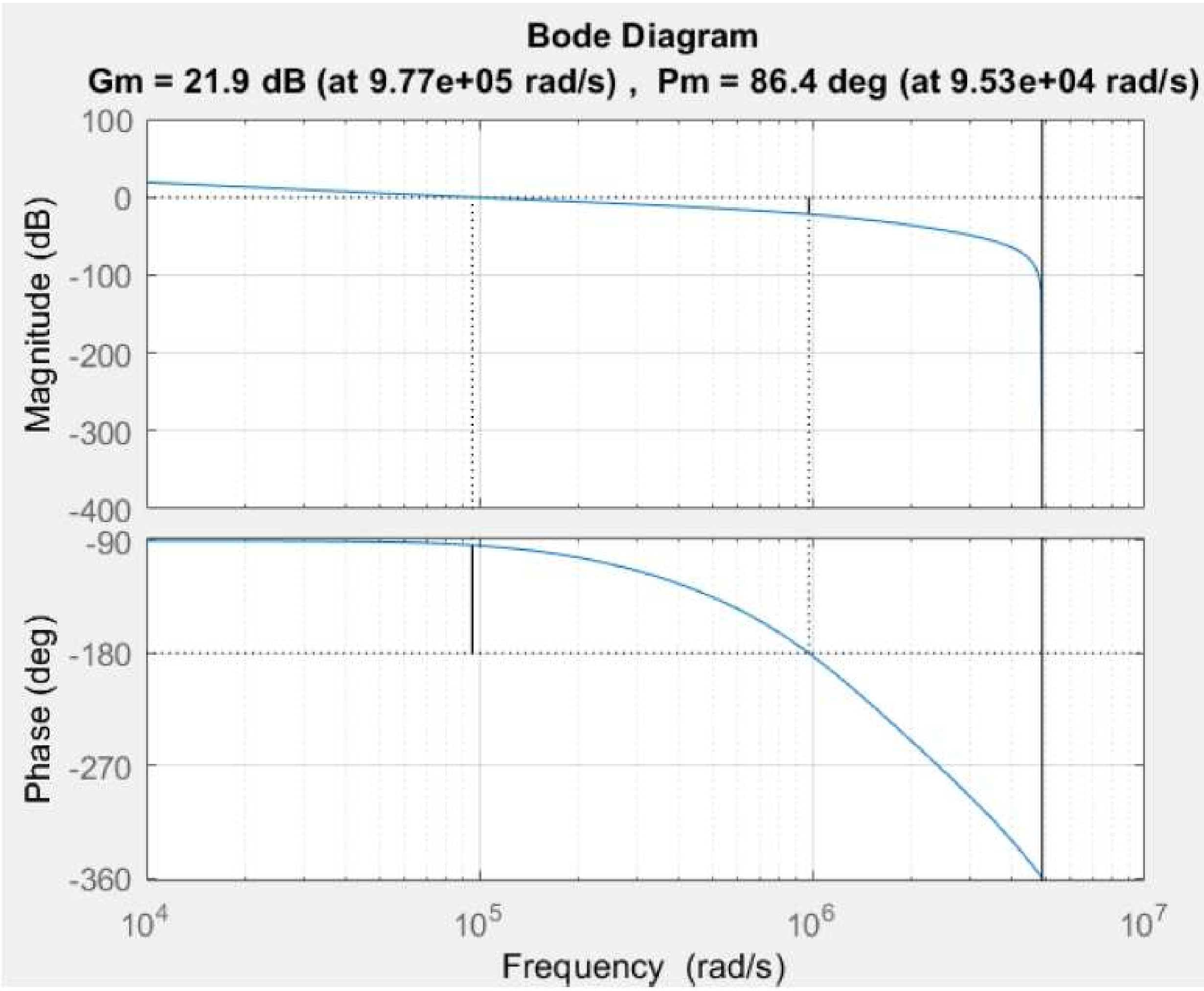


**Figure 6.24:** $H_\infty$ Loopshaping Gain and Phase Margins

**Table 6.7:** Metrics for Unmatched Impedance Case, $H_\infty$ Controller with Reset Integrator

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 1.9810 | 2.9003 | 0.5807 | 0.1477 |
| 2 | 166.8230 | 168.0620 | 159.0081 | 159.3952 |
| 3 | 11.3014 | 16.9728 | 7.0122 | 8.2255 |
| 4 | 0.7035 | 2.2718 | -0.0003 | 0.3852 |
| 5 | 1.8502 | 2.1423 | 0.7514 | 0.5016 |
| 6 | 271.7687 | 272.5597 | 232.5557 | 235.2705 |
| 7 | 40.1076 | 42.1279 | 32.1629 | 30.7269 |
| 8 | 3.1558 | 4.4529 | 0.6071 | 0.2165 |

Comparing tables 6.6 and 6.8 (matched impedance cases), we see that the $H_\infty$ Loopshaping Controller (with reset integrator) does improve the performance verses the original AE reset based

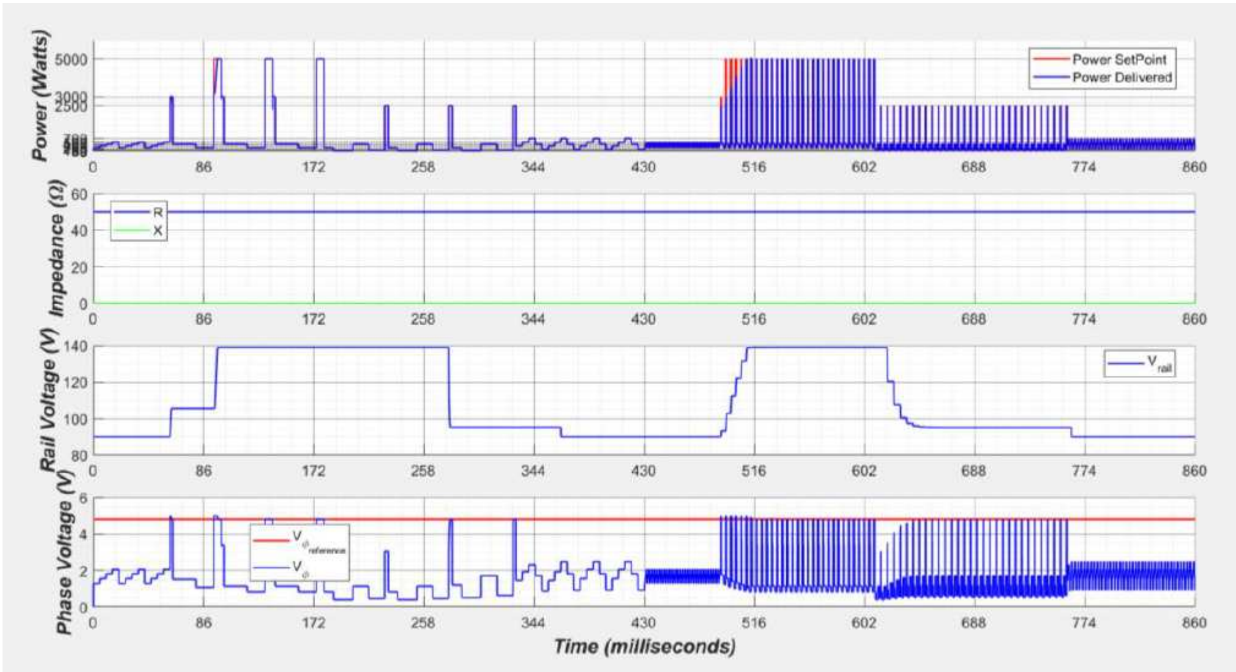


**Figure 6.25:** Matched Impedance Setpoint Tracking with the $H_\infty$ Controller with Reset Integrator


**Figure 6.26:** Matched Impedance Average Tracking with the $H_\infty$ Controller with Reset Integrator

pure integral controller. However, the new control approach only offers fairly small improvements, as the performance was already high in this case. On the other hand, comparing tables 6.7 and 6.9, we find that the new controller brings down the metrics significantly in this (unmatched

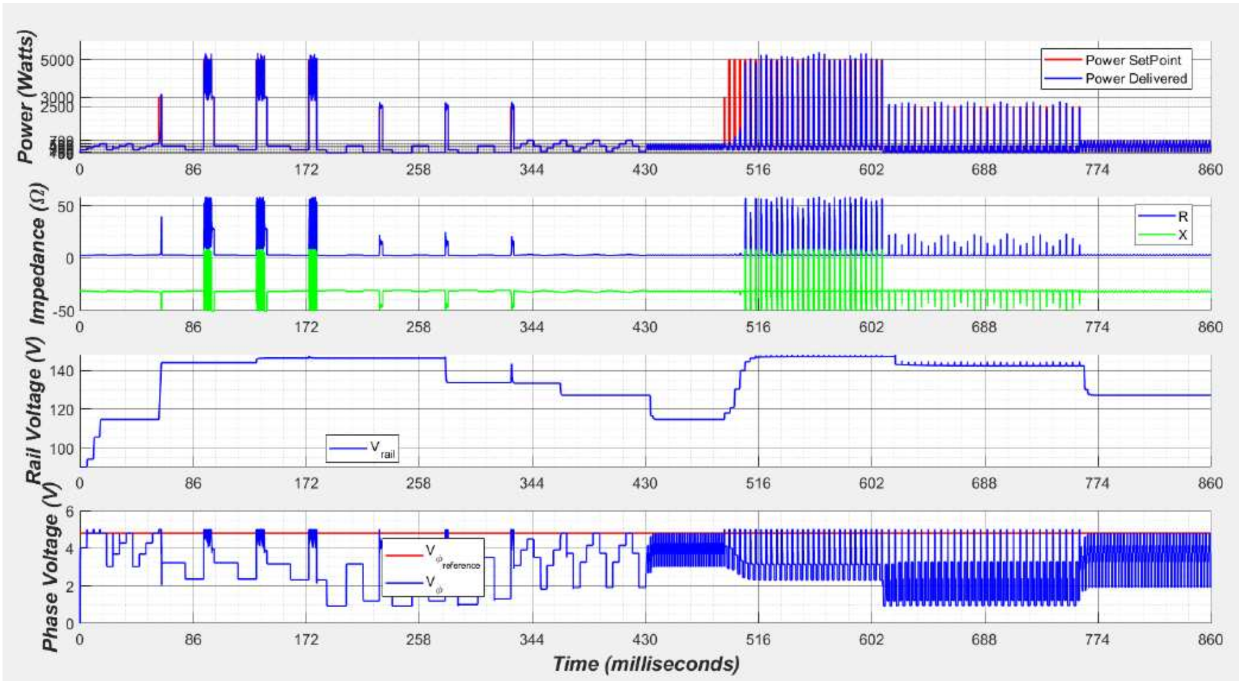


**Figure 6.27:** Unmatched Impedance Real Time Tracking, $H_\infty$ Controller with Reset Integrator

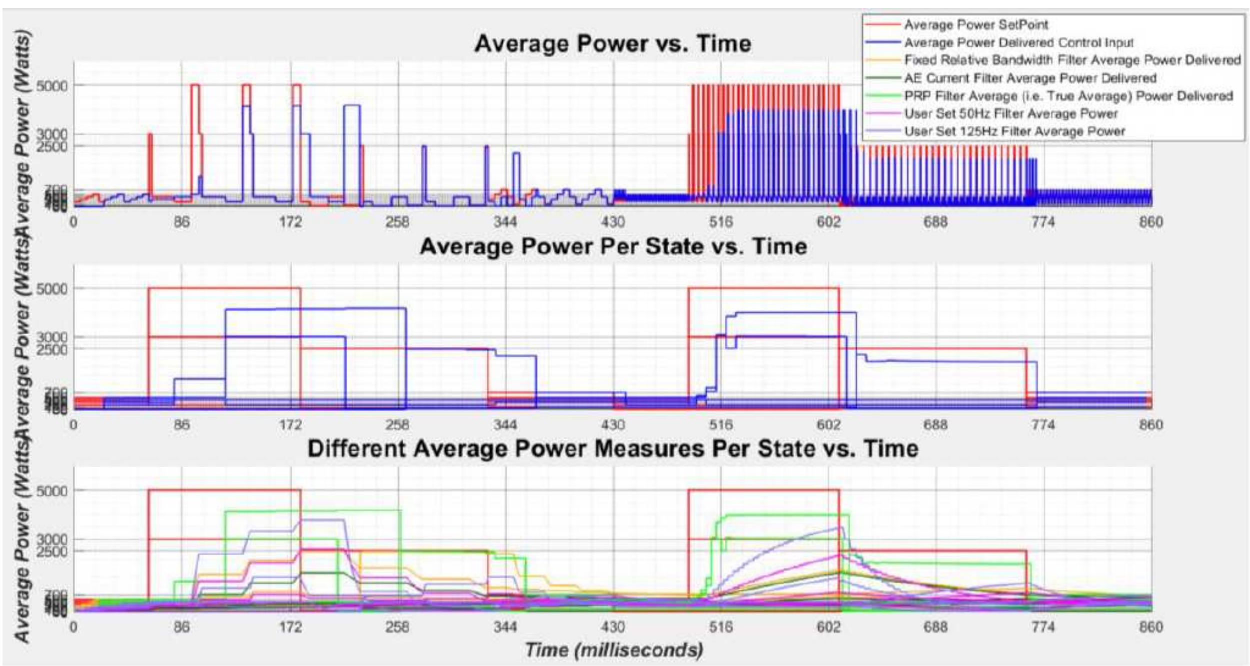


**Figure 6.28:** Unmatched impedance Average Tracking, $H_\infty$ Controller with Reset Integrator

impedance) case, due to the reduced oscillations afforded by the higher stability margins (though the performance is still not good for this case).

**Table 6.8:** Matched Impedance AE Reset Integral Controller

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---:|---:|---:|---:|---:|
| 1 | 0.23285 | 1.35895 | 0.23268 | -0.07826 |
| 2 | 32.48597 | 34.45961 | 30.09951 | 28.60247 |
| 3 | 0.64017 | 7.27334 | 0.19296 | 1.00059 |
| 4 | 0.45103 | 2.25952 | -0.15003 | 0.35116 |
| 5 | 0.12803 | 0.49109 | 0.12784 | -0.12074 |
| 6 | 35.07161 | 35.51380 | 31.85883 | 32.2836 |
| 7 | 1.80869 | 3.55465 | 0.56000 | -0.81988 |
| 8 | 0.47146 | 1.91627 | -0.17043 | -0.55379 |

**Table 6.9:** Unmatched Impedance AE Reset Integral Controller

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---:|---:|---:|---:|---:|
| 1 | 1.9895 | 2.9052 | 0.6564 | 0.2182 |
| 2 | 288.6988 | 264.1721 | 232.1695 | 203.5750 |
| 3 | 26.6183 | 32.2055 | 16.0240 | 16.9788 |
| 4 | 0.9645 | 2.3354 | 0.1267 | 0.4183 |
| 5 | 2.0212 | 2.2987 | 0.7561 | 0.5062 |
| 6 | 340.2521 | 342.4215 | 264.5478 | 268.9389 |
| 7 | 43.1863 | 44.9203 | 33.6613 | 32.0185 |
| 8 | 3.4661 | 4.7285 | 0.5572 | 0.1716 |

### $H_\infty$ Loopshaping Controller without AE reset Integrator

Here, we remove the existing AE reset-based integrator completely from the Vphase controller, and replace it with the $H_\infty$ Loopshaping Controller standalone as shown in figure 6.29. Looking

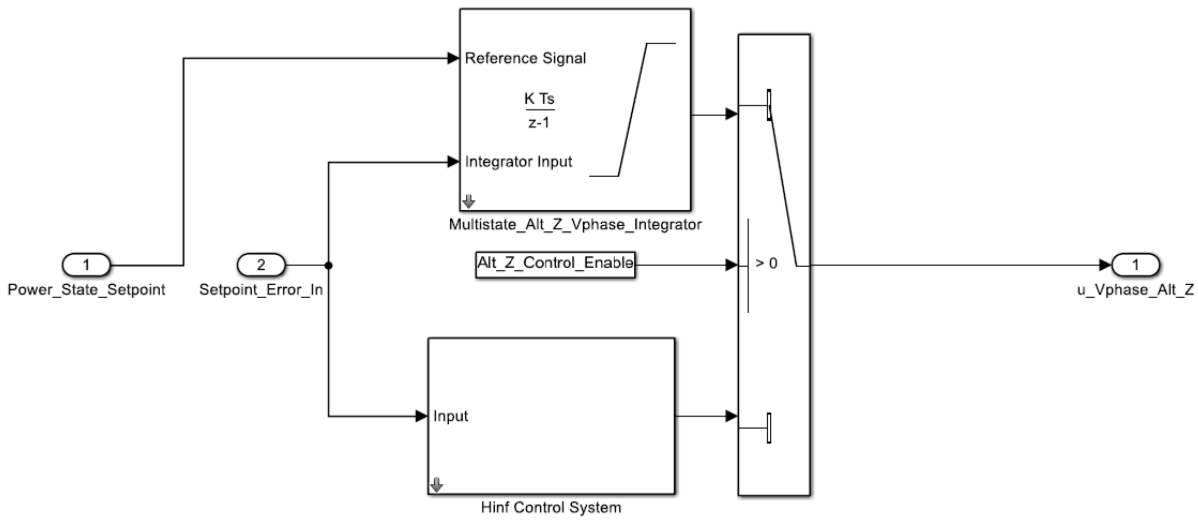


**Figure 6.29:** $H_\infty$ Controller Replaces Reset Integrator Block

under the mask of the 'Hinf Control System' block we see in figure 6.30 that the full controller is implemented standalone, but still as a cascade of a proper controller and an integrator. The integrator block implements output limits (0-5volts) with anti-windup (clamping). We run the

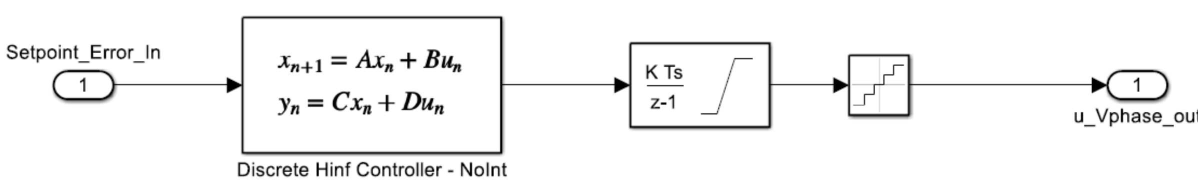


**Figure 6.30:** $H_\infty$ Controller Standalone Implemented under 'Hinf Control System' Mask

same simulation tests and metrics on this controller ($H_\infty$ Loopshaping Controller without Reset Integrator). First the simulation runs for the matched impedance case are shown in figures 6.31

(real-time tracking) and 6.32 (average tracking). The corresponding metrics are shown in table 6.10.

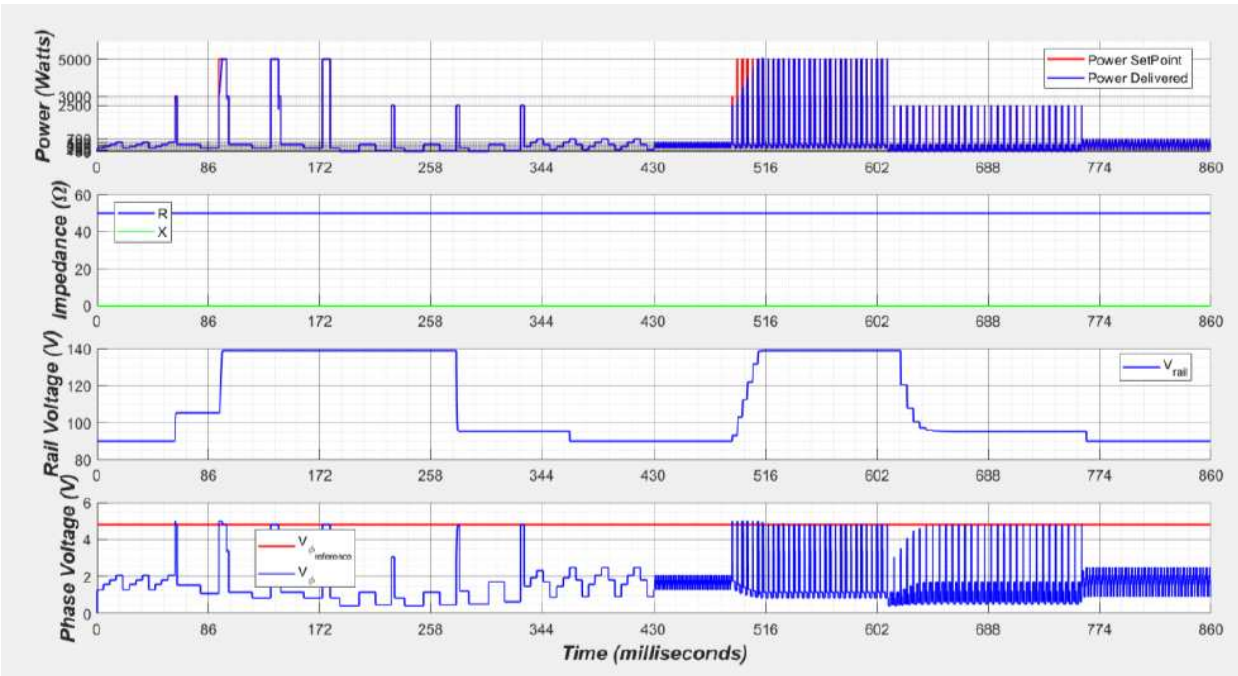


**Figure 6.31:** Matched Impedance Setpoint Tracking for the $H_\infty$ Controller without Reset Integrator

**Table 6.10:** Metrics for $H_\infty$ Controller without Reset Integrator for Matched Impedance

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 0.5630 | 1.6825 | 0.2256 | -0.0787 |
| 2 | 32.5935 | 34.5321 | 30.9734 | 29.4985 |
| 3 | 1.1463 | 7.7550 | 0.0214 | 0.7896 |
| 4 | 0.9560 | 2.7579 | -0.2323 | 0.2519 |
| 5 | 5.0137 | 5.2671 | 0.1239 | -0.1238 |
| 6 | 48.9144 | 49.3828 | 32.7771 | 33.1875 |
| 7 | 13.2742 | 15.0508 | 0.0106 | -1.3482 |
| 8 | 8.2675 | 9.5368 | -0.2468 | -0.6281 |

This controller is able to deliver close to the same performance, without the reset integrator. This would not be possible utilizing simple integral control – the pure integrator control approach fundamentally relies on the reset scheme to deliver any acceptable level of performance.


**Figure 6.32:** Matched Impedance Average Tracking for the $H_\infty$ Controller without Reset Integrator

Finally we look at the unmatched impedance case for the $H_\infty$ Loopshaping Controller without Reset Integrator. The real-time tracking performance is shown in figure 6.33, and the average tracking performance is shown in figure 6.34. The corresponding performance metrics are shown in table 6.11.

**Table 6.11:** Metrics for Unmatched Impedance for $H_\infty$ Controller without Reset Integrator

| Pulse Cycle | Mean Absolute Error Unmodified | Mean Absolute Error Modified | Mean Real Error Unmodified | Mean Real Error Modified |
|---|---|---|---|---|
| 1 | 2.1084 | 3.0535 | 1.1795 | 0.7492 |
| 2 | 166.0600 | 194.1123 | 159.6441 | 185.8936 |
| 3 | 11.4935 | 17.3940 | 6.5712 | 7.9302 |
| 4 | 1.5573 | 3.1354 | -0.1528 | 0.2128 |
| 5 | 11.7252 | 11.8455 | 0.3204 | 0.0736 |
| 6 | 267.6528 | 267.2916 | 245.1357 | 244.1282 |
| 7 | 47.4188 | 49.0603 | 29.0263 | 27.5043 |
| 8 | 15.2950 | 16.3851 | -0.1805 | -0.5560 |

Once again we observe that the new controller approach manages to minimize the metrics and deliver a decent level of performance without the resetting integrator algorithm.

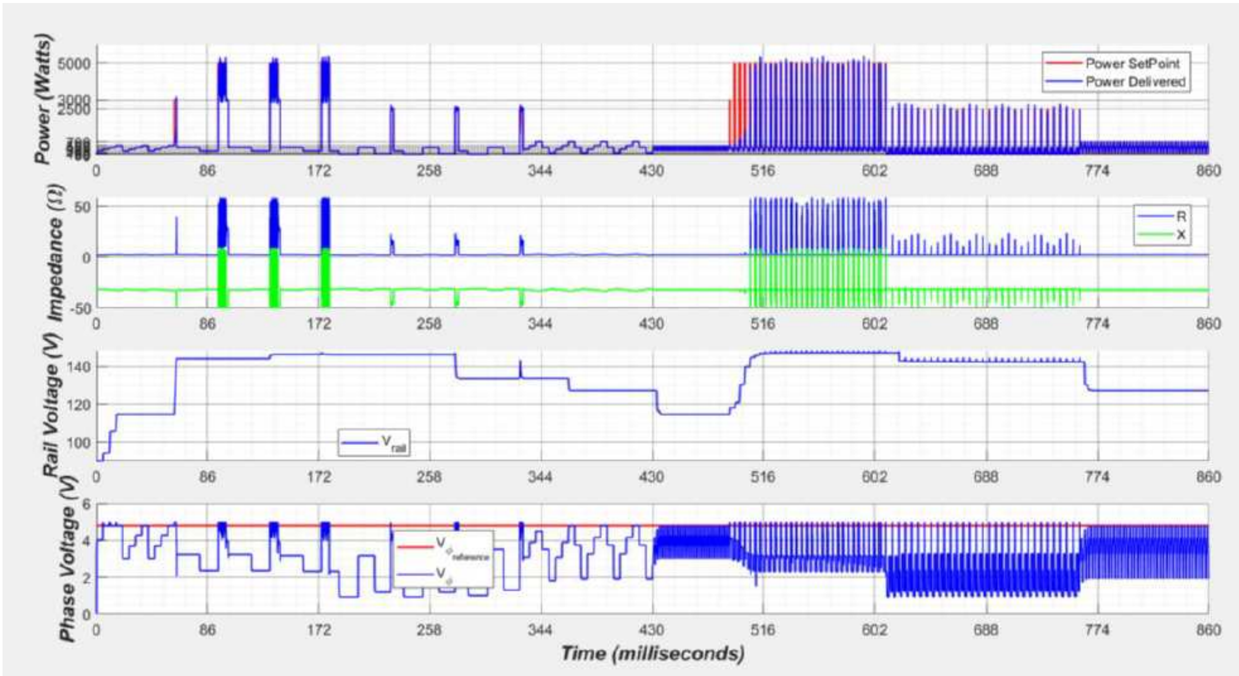


**Figure 6.33:** Unmatched Impedance Setpoint Tracking for $H_\infty$ Controller without Reset Integrator

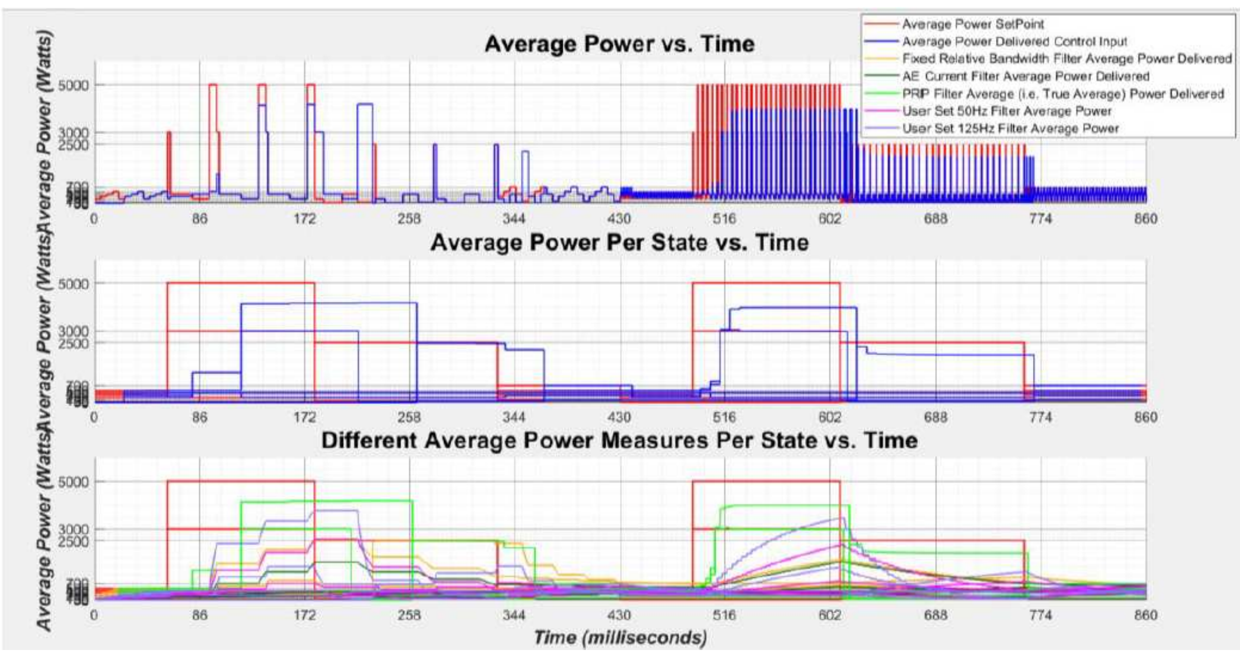


**Figure 6.34:** Unmatched Impedance Average Tracking for $H_\infty$ Controller without Reset Integrator

The $H_\infty$ controller was also tested out with the PID scheme developled earlier. For instance one could utilize the PID based controller for Vrail (resulting in a faster Vrail response) and combine that with the $H_\infty$ Loopshaping Controller for Vphase (with or without the AE reset integrator).

We implemented such a scheme and the response for the matched impedance case is shown in figures 6.35 and 6.36. It is seen that we obtain very high performance for real-time tracking using this combined approach. However, as mentioned earlier, the faster Vrail response may not be acceptable on the AE system for hardware reasons.

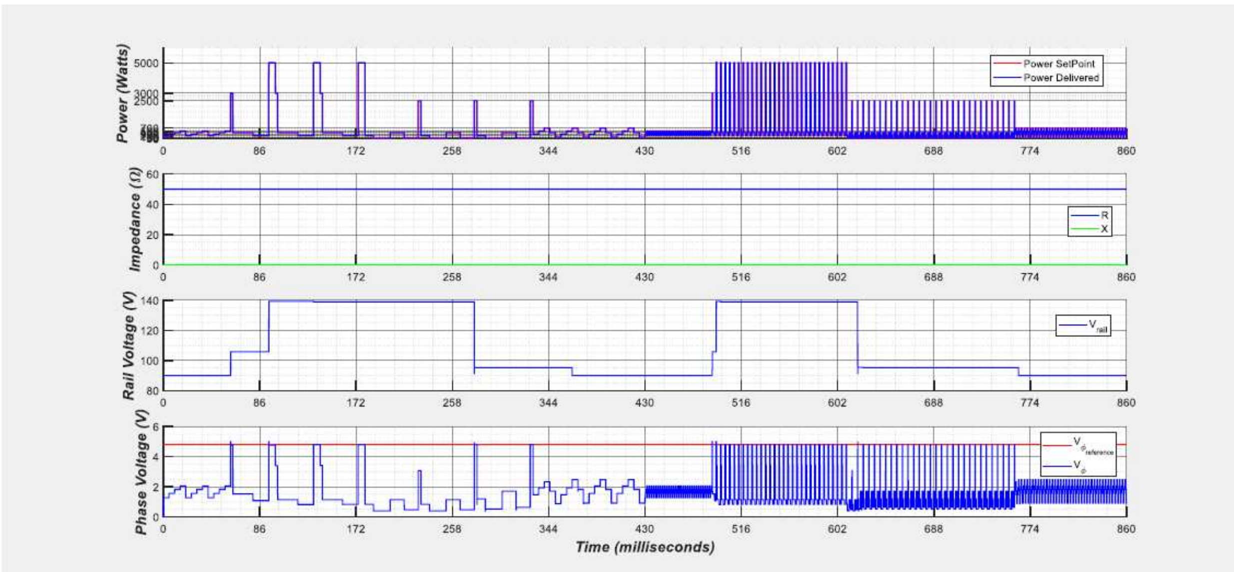


**Figure 6.35:** Matched Impedance Setpoint Tracking for the $H_\infty$ Controller with Vrail PID


**Figure 6.36:** Matched Impedance Average Tracking for the $H_\infty$ Controller with Vrail PID

## 6.4 Optimization

### 6.4.1 Stability Region Calculation

In order to characterize the robust stability properties of any particular controller design, an approach for stability region calculation was developed in collaboration with AE which is briefly described here. The idea is to first generate a grid of Power Setpoint (and $50\Omega$ point) versus Cable Length Zrotation Angle. For example let's say the Power Setpoint varies from 0 - 1,350 (Watts), and the Zrotation Angle from 0 - 360 (degrees). The source impedance ($50\Omega$ point) is kept equal to the Power Setpoint throughout the simulations. We generate a grid of these values and at each grid point we run a simulation as described below.

The Zrotation Angle is a free parameter which we simply assign in the simulation model. The Power Setpoint determines the input to that simulation. For example let's suppose the Power Setpoint for the current grid point is 800W. Then we generate a waveform as shown in figure 6.37.


**Figure 6.37:** Power Level Steps for 800W Setpoint Test

Consider the plot, and note that this input consists of a series of steps, both up and down, to 800W. We are essentially testing the step response to 800W, and calculating the average tracking error. If this average response settles within a specified error (usually 10%), within a specified time interval, then the response is deemed stable. The use of a series of steps allows time for the controller to learn (due to averaging, setpoint modification and reset integral), and we includes steps from both above and below to test for both rising-edge and falling-edge stability. In this way we thoroughly examine the controller response (even if it includes reset-type control).

In this way we determine stability (as defined by AE) for each of the grid points. This determines the stability region and an example plot, for the existing AE reset-integrator control scheme, is shown in figure 6.38

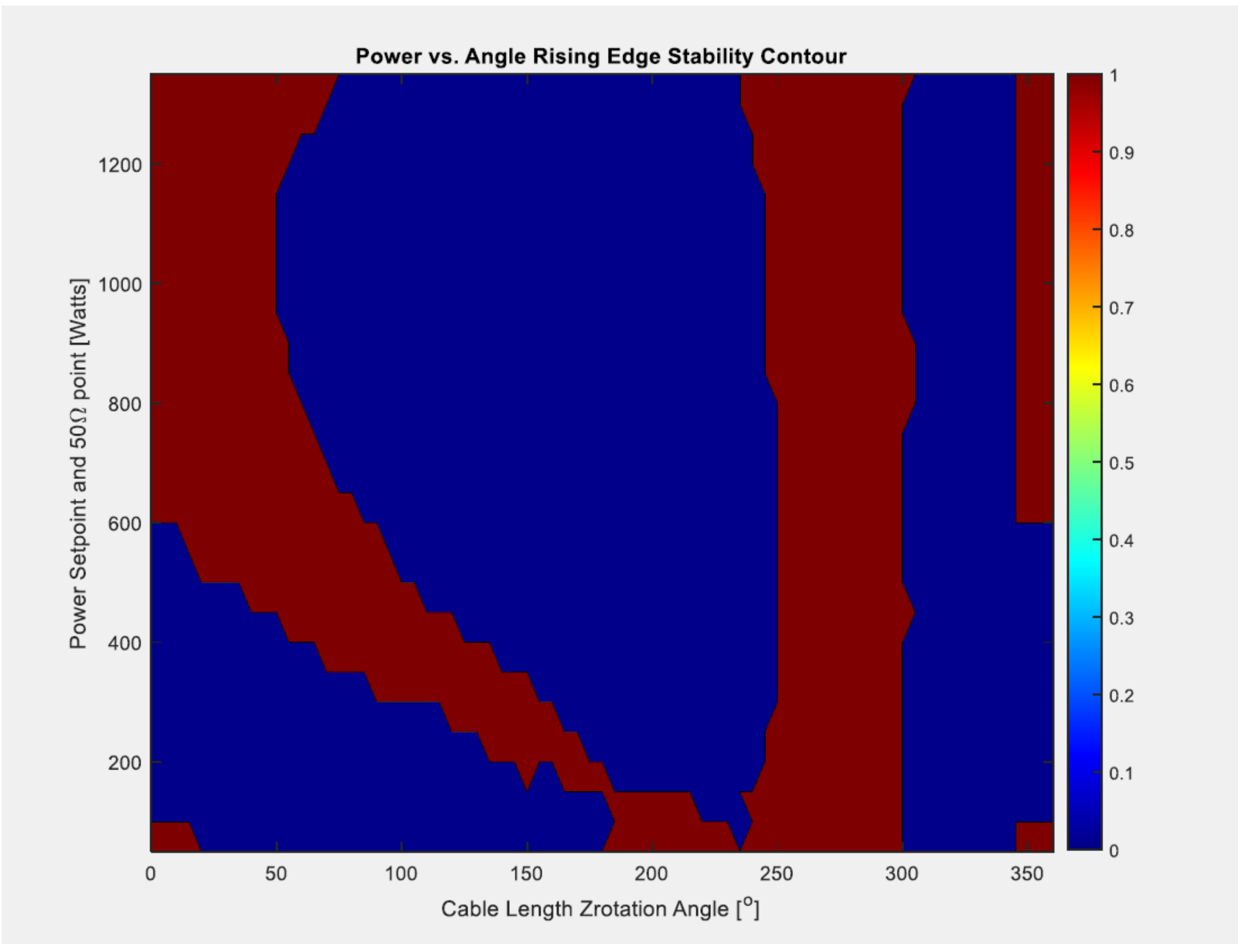


**Figure 6.38:** Rising Edge Stability Region for Current AE Controller

We only show the rising-edge stability region here, but the falling-edge stability region looks very similar. The Power Setpoint (Watts) is shown on the Y-Axis, and the Cable Length Zrotation Angle (degrees) is shown on the X-Axis. For each grid point we simply label it as stable (red) or unstable (blue), where the definition of stability used was given above. Note that this definition is simulation-based, and may be quite different from many textbook definitions of stability, but it is tailored to AE's needs. The resulting contour map shown in figure 6.38 clearly shows the stability region(s) for this controller.

### 6.4.2 Parallel Simulations

The plot in figure 6.38 was generated using 27 different power states (one per 50W), and 73 different Zrotation angles (one per 5 degrees). This results in a grid of 27x73 = 1,971 simulations. In order to facilitate faster processing, the Matlab parsim command was used, which allows for parallel processing of multiple simulations at the same time. An example of a parsim run (in progress) is shown in figure 6.39.

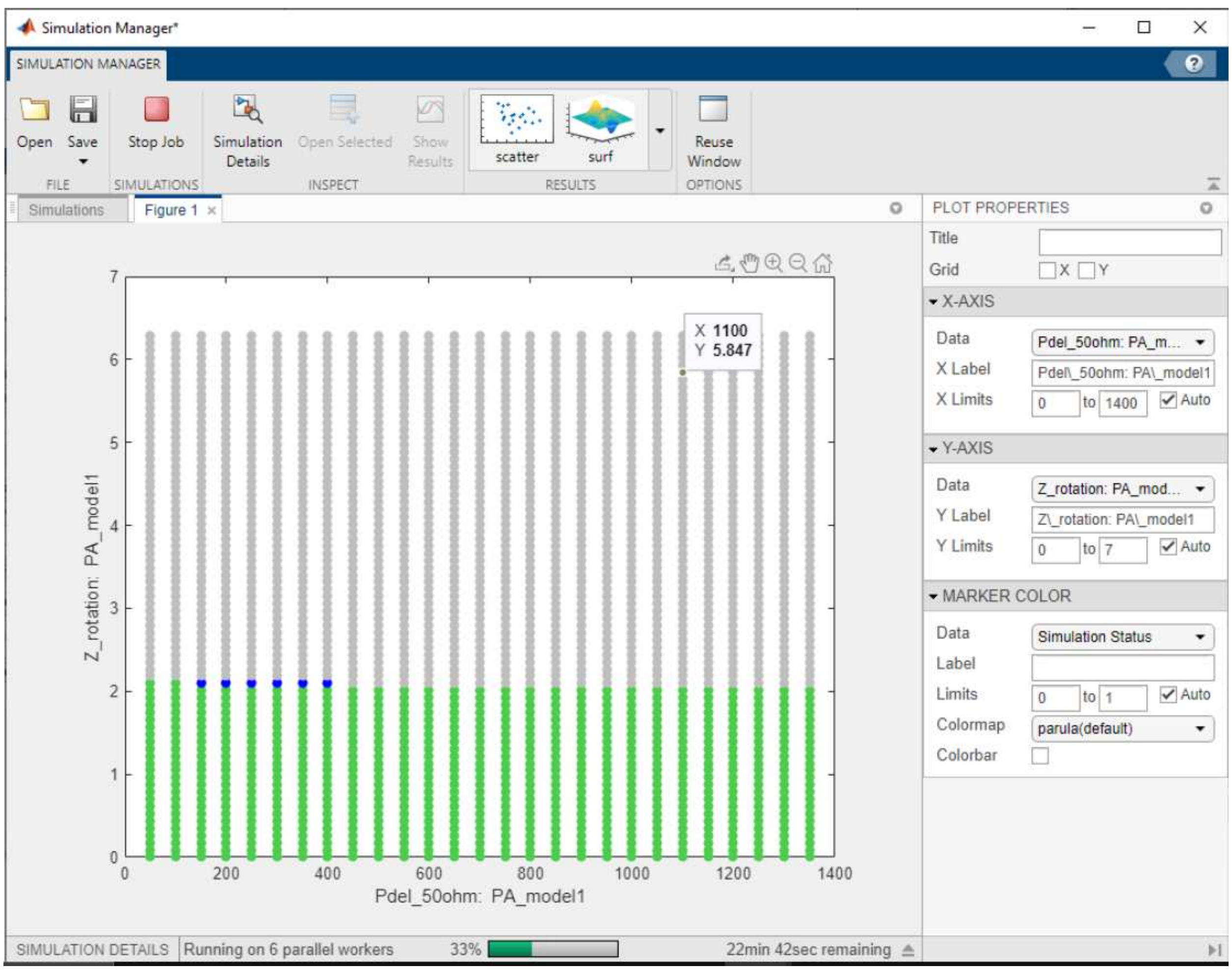


**Figure 6.39:** Parallel Simulation Processing via Matlab parsim Command

This figure shows the process of the 1,971 simulations (note that here Power Setpoint is on the X-Axis, Zrotation Angle is on the Y-Axis and is in radians). Completed simulations are in green (any completing with errors would be in red), those in progress are in blue, and those remaining to start are in grey. It can be seen we are about 33% of the way through (and no simulations have errors). Note also that the six blue dots indicate we are running 6 simulations (workers) in parallel. The use of this parallel approach speeds up the stability region calculation significantly.

### 6.4.3 Stability Region Size

The Stability Region Size (SRS), which will form the main objective function for our optimization approach, can again be defined in a variety of ways. Once again our definition is tailored to the needs of AE, and is specified as follows.

First we decide on an appropriate Power Setpoint of interest. In our case 800W was chosen as a test point. Then we consider the region, from 0 upwards, of the Zrotation Angle for which the system is stable. Once the first unstable point is encountered, the stability region is deemed to be ended.

This is easy to visualize on the plot from figure 6.39. Start at the point (0,800) on the Y-Axis. Draw a horizontal line in the red (stable) region until you first encounter a blue (unstable) point. The Stability Region Size is the length of that line (in this case 60 degrees). The SRS calculation is illustrated below in figure 6.40 (same contour map as in figure 6.38).

Note that there are other ways to define the Stability Region Size. For example, one could combine the rising-edge and falling-edge values. One could also consider the total amount of stable values across the entire horizontal line (our definition ignores the second stability region around 245-305 degrees). Any of these would work with our approach. We chose this particular definition, in consultation with AE, as just one example of the process (noting that many other choices are valid and will work with this same process).

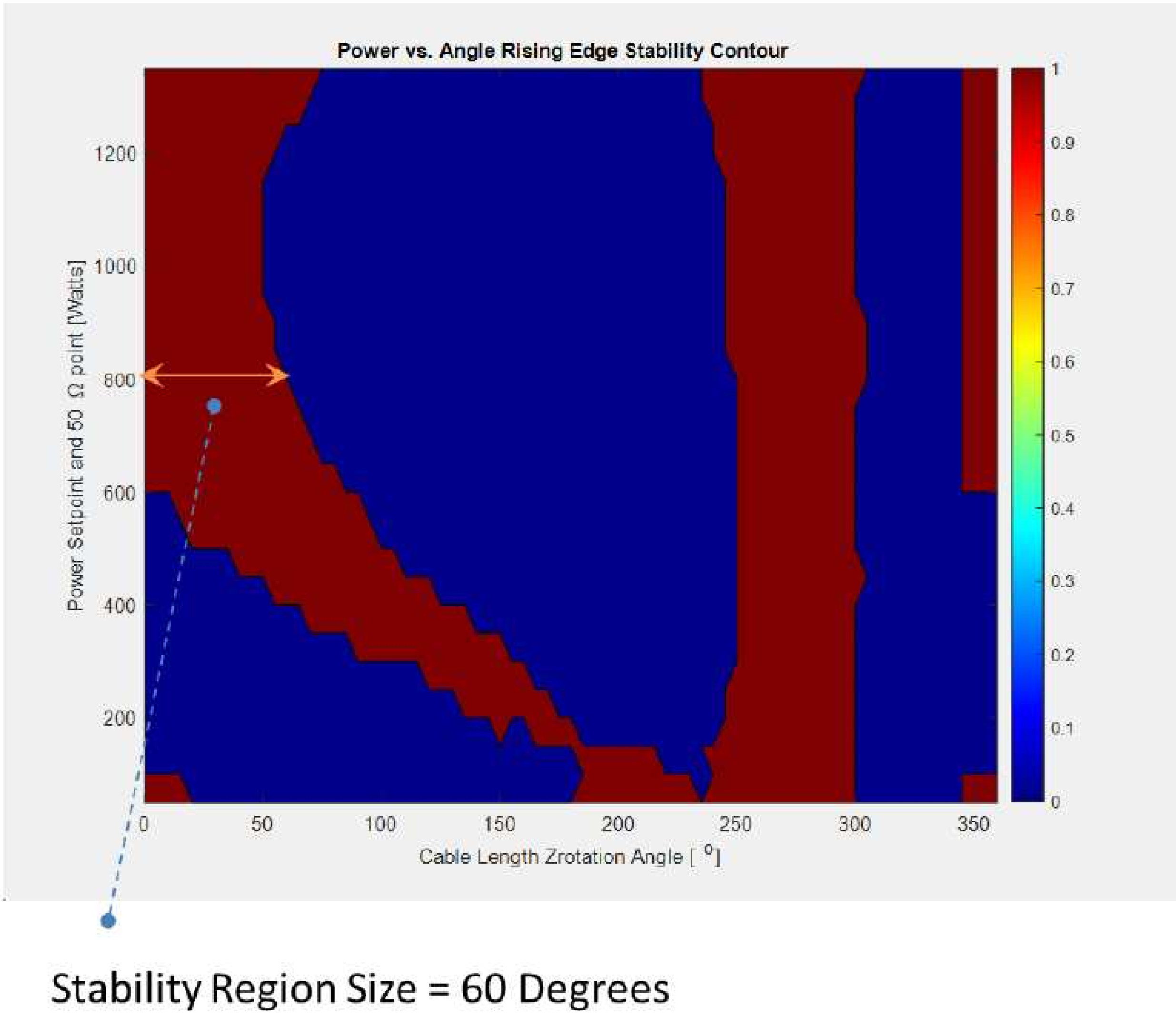


**Figure 6.40:** Stability Region Size Definition for 800W Power Setpoint (Current AE Controller)

### 6.4.4 Controller Optimization

We now turn our attention to implementing and tailoring the control approach for AE systems. Our $H_\infty$-Loopshaping control design approach is already designed specifically for the AE system under consideration, and it delivers both high performance and robust stability. The question now is whether we can tailor the controller to a particular AE system, with a specific set of design considerations/metrics that we wish to optimize?

In order to develop such a process, note that our controller design process utilizes a number of free parameters, which the designer selects when tuning the controller. For example, referring to our earlier discussion of $H_\infty$-Loopshaping design, we see that the design weight W(s) is a function of four design parameters: $C_K$, $T_K$, $K_P$, $K_I$ – which are all free for the design engineer to choose. These parameter choices result in different $H_\infty$-Loopshaping controller designs.

The idea now is to allow an optimization process to vary these parameters, and search for the 'best' design. Of course deciding which is the 'best' design amounts to choosing system performance metrics. We developed a number of such metrics in earlier work, and also, in consultation with AE, developed the notion of Stability Region Size in the previous sections. The process would work with almost any choice of metric but, in consultation with AE, the SRS was chosen. The overall optimization approach is shown in figure 6.41 below.

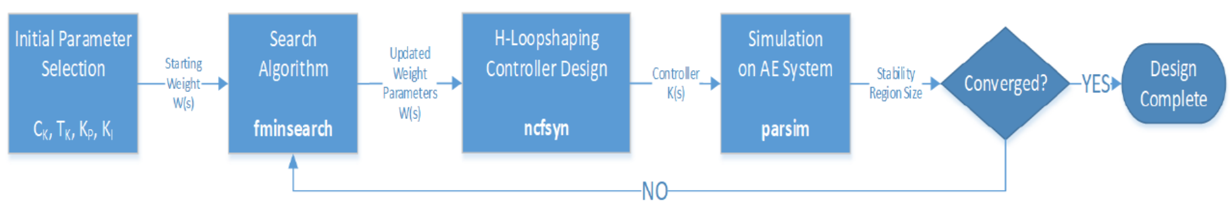


**Figure 6.41:** Controller Optimization Approach

The scheme runs as follows. The optimization utilizes the Nelder-Mead search algorithm, as implemented by the Matlab fminsearch command. The Nelder-Mead approach is chosen for a number of reasons, including the fact that it does not rely on the computation of gradients. As such it can handle nonlinear and nonsmooth objective functions. Section 5.4 describes this algorithm in detail.

The fminsearch command provides updated parameter values for $C_K$, $T_K$, $K_P$, $K_I$ (after the initial guess or previous iteration). These in turn determine the weight W(s) for $H_\infty$-Loopshaping controller design (via the Matlab ncfsyn command). The resulting controller K(s) is then used as the basis for controller simulations (via the Matlab parsim command) to determine the Stability Region Size. This proceeds until it is determined to have converged (or failed) resulting in a controller whose SRS has been optimized.

Note that here we are describing a general process to implement an optimized controller on an AE system. This particular solution is just one example of applying that process. For example, we chose the SRS as our objective function but there are many other choices for metrics, including the ones we developed in earlier work. In addition, there are other possibilities for the free parameters and the parameterized weights used in the $H_\infty$-Loopshaping controller design approach.

Furthermore, one can also consider controller design choices other than $H_\infty$-Loopshaping, and even consider search algorithms other than Nelder-Mead. Hence, it is important to consider the work described here as the development of a general process, and the results in the next section are just one particular example of applying that process.

### 6.4.5 Simulation Studies

The tools described in the preceding section were implemented with the goal of optimizing the Stability Region Size for a controller design on a particular AE system.. We chose a Power Setpoint level of 800W for this example (though of course you could choose any desired value).

As a baseline consider the current AE control approach. The stability contour plot for this controller is shown in figure 6.38. The SRS is calculated to be 60 degrees for the current AE control approach (see also figure 6.40 which illustrates the SRS calculation for this stability contour plot for a Power Setpoint level of 800W).

In order to compare with the new approach we design an initial controller. In our case this amounts to the design engineer making an initial choice for the weight parameters $C_K, T_K, K_P, K_I$ and then designing the corresponding $H_\infty$-Loopshaping controller. The resulting stability contour plot is shown in figure 6.42. The SRS (at 800W) for this initial controller design is calculated to be 70 degrees.

We can see the $H_\infty$-Loopshaping design approach already has some improved stability properties versus the current AE approach (compare figures 6.38 and 6.42). This results in an increased SRS (70° versus 60°). However we now consider this as just an initial guess and implement the controller optimization scheme outlined figure 6.41.

Our first attempt at running this optimization, using the above initial guess, did not yield any improvement in the SRS, and in fact did not alter the controller design at all. This is entirely possible. In the first place, the optimization algorithm can only ever guarantee to find local minima for nonconvex problems. In the second place, it is quite possible that we started out the optimization at one such local minimum (by tuning our controller design). Hence the search algorithm effectively did not move from the initial starting point.

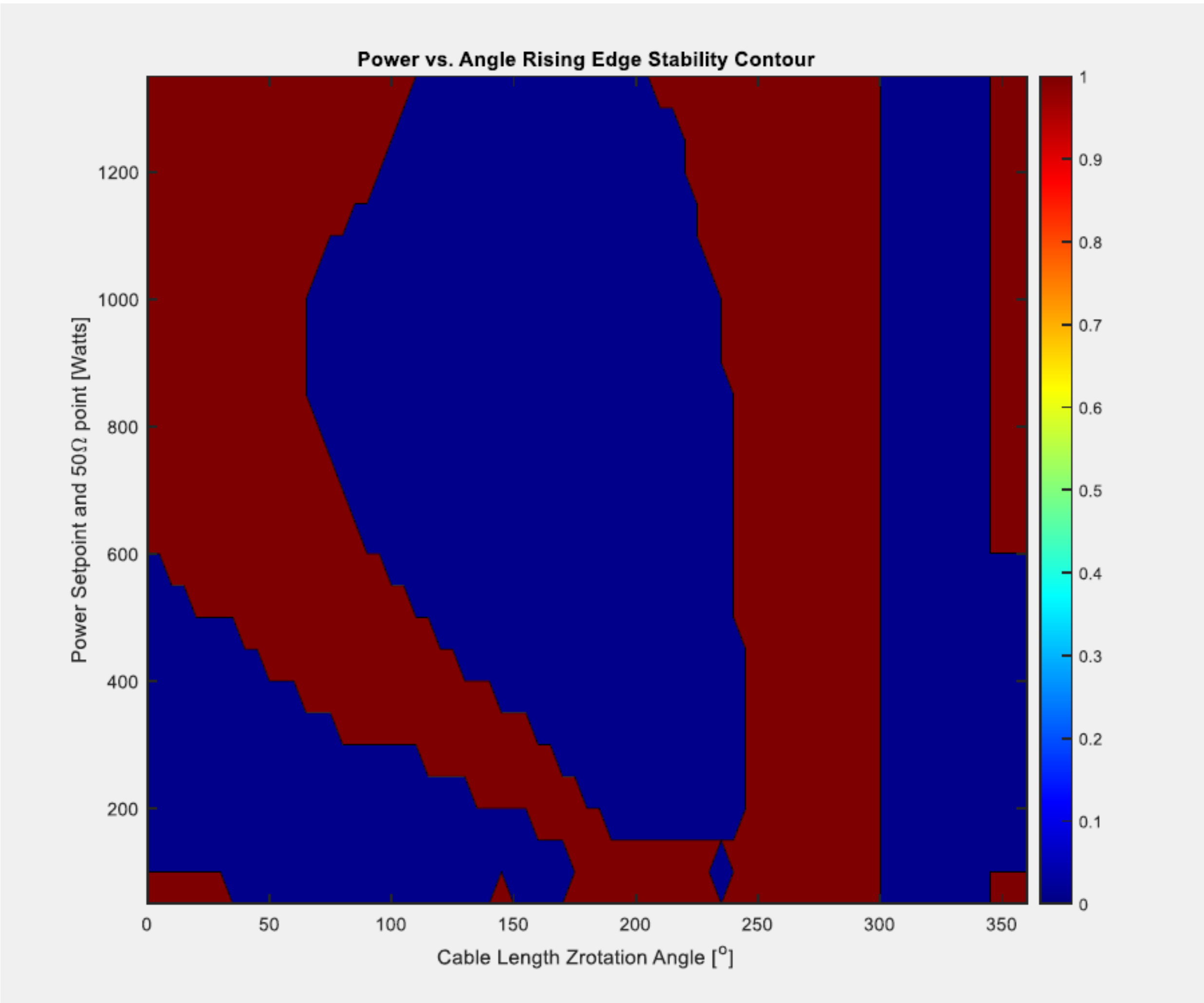


**Figure 6.42:** Stability Region for Initial $H_\infty$-Loopshaping Controller

In order to alleviate the problem described above, and further examine the solution space, we randomly varied the initial guess for the parameter starting point, and again implemented the controller optimization scheme, with the objective of maximizing the SRS at 800W. The resulting contour plot is shown in figure 6.43.

It can be seen the SRS at 800W is dramatically improved, to a value of 305 degrees (versus 70° for the initial starting point). The optimization scheme has done as asked, and maximized the SRS at 800W. Note that the SRS (at 800W) has improved from:

- Original AE Controller SRS: 60°

- $H_\infty$-Loopshaping Initial Controller SRS: 70°

- $H_\infty$-Loopshaping Optimized Controller SRS: 305°

This is a clear illustration of the power of the proposed optimization approach, which has delivered the desired maximized SRS at 800W.

There are, however, further considerations. Optimization schemes only optimize what they are told to do by the objective function. Indeed, comparing figures 6.43 and 6.42, it is apparent that the optimized controller stability contour plot is worse in some regions than the initial starting point (for example there is no stable region for power levels below 250W in figure 6.43, whereas figure 6.42 has some stable regions all the way down to 50W). Again this is entirely possible. The optimized controller may sacrifice performance/stability in some measures/regions in order to improve those values that matter to the objective function. In fact it has clearly done that here.

Recall that this is just one particular example of the proposed process, and it certainly illustrates that it works. However, in practice one will need to make a very careful choice of objective function. One needs to ensure that it accurately captures all of the desired features of the control design, so that the resulting optimized controller has desirable properties in all aspects of stability and performance that matter to AE.

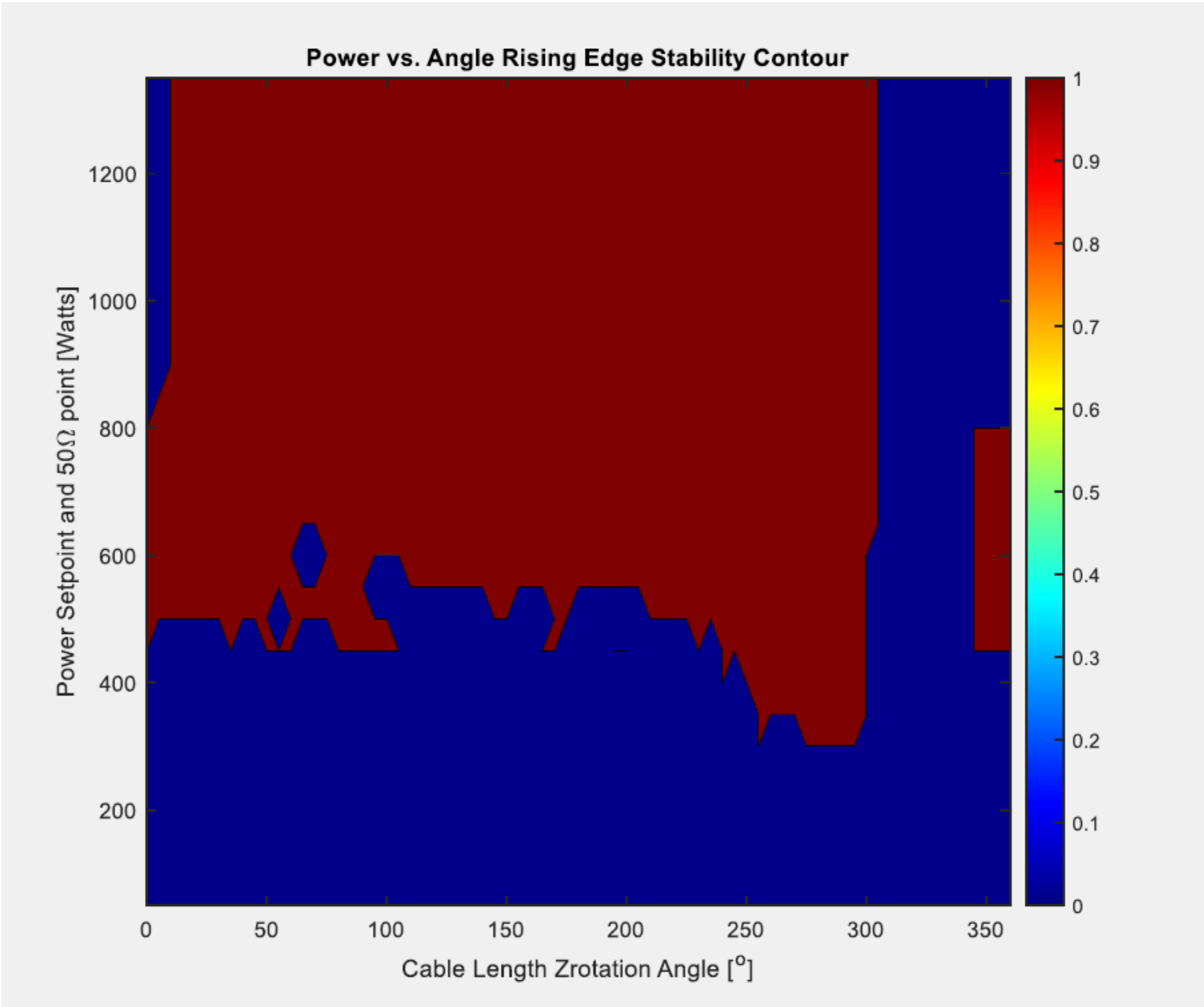


**Figure 6.43:** Stability Region for Optimized (at 800W) $H_\infty$-Loopshaping Controller

# Chapter 7

# Conclusion and Future work

In this thesis, a nonlinear high frequency plasma load was studied, controllers were developed for the generator and the results were optimised for a range of parameters. Metrics were first developed as a way of quantizing the AE simulation for generating specific goals that AE was interested in. Average and real time tracking were differentiated according to the AE system keeping in mind the tradeoffs between the two. These metrics were used for checking the performance of the controllers developed later in the project. These metrics can be used further in the project for developing specific optimal controllers according to the user specifications.

$H_\infty$ loopshaping approach was successfully implemented as a feedback control strategy for the nonlinear load. As described in section 4.2, the integral control of the AE system uses lookup tables for initial value and reset operations. The use of lookup tables for a specific plasma model implies special treatment and such a system would not work for other plasma models with different parameters. The new controller successfully eliminated the use of lookup tables in the original reset integral control. Alternatively, a PID approach was also implemented successfully on the AE system. There are a few other methods that can be used in the future like adaptive control methods and learning control methods. A wide range of these controllers could be designed depending upon some classification criteria for the load and can be used in commissioning for getting the best control option.

The plasma model in the AE Simulink model has an interpolation algorithm for Output Power, Current and Voltages. The interpolation is based on appropriate data that is generated experimentally from an actual plasma system. This data is used to generate appropriate equations for Power Output and can be thought of as calibration data that could be generated for each system/application as required during commissioning.

In order to identify this we remove the controllers from the Simulink model to leave the open loop system. We then manually generate Vrail and Vphase as a series of step inputs. Initially, Vrail is kept constant, with a series of Vphase step inputs divided equally throughout the range. The open loop simulation data is saved. We then increase Vrail, with the same set of Vphase inputs, again saving the open loop data. This process continues until Vrail reaches the maximum of its range. We continue the same cycle with Vphase as a constant input, and Vrail as a set of varying step inputs. We record all the steady state power outputs. This open loop recorded data is then used in a curve fit algorithm (MATLAB cftool, Polynomial curve fitting) to generate a three-dimensional surface and an approximate second order polynomial relating Power to Vrail and Vphase.

$$Pdel = 843.4 - 21.82Vrail + 0.099Vrail^2 - 586.9Vphase + 64.87Vphase^2 + 9.479VrailVphase$$

The equation describes the relationship of Pdel to Vrail and Vphase for the plasma model described in the Simulink model (and based on experimental data). It is illustrated in figure 7.1 below, showing both the data and the fitted surface.

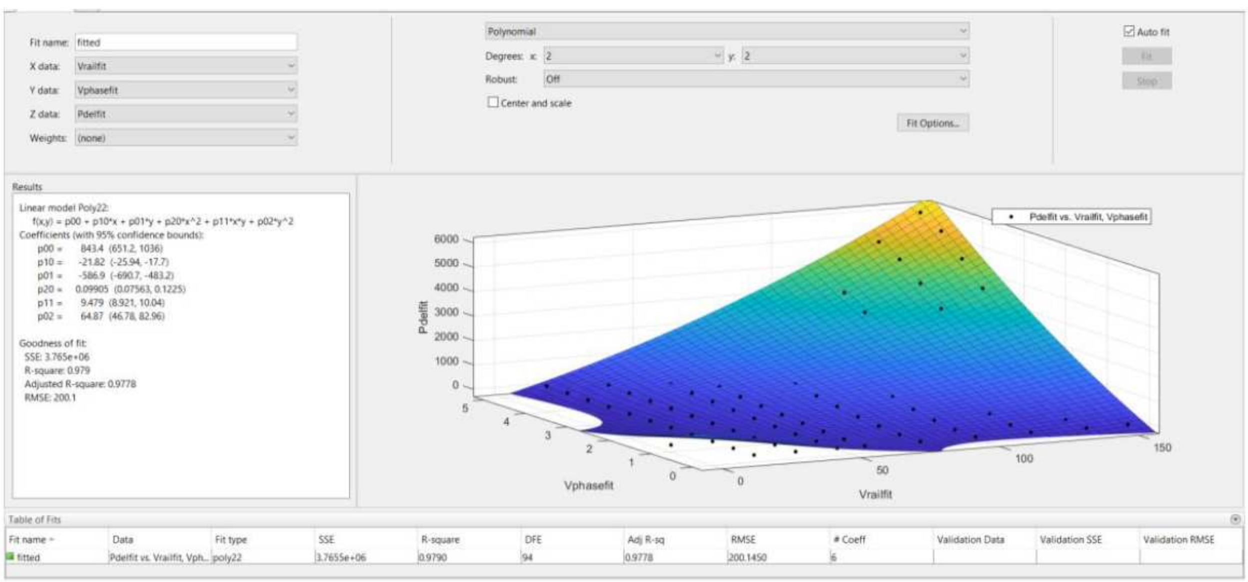


**Figure 7.1:** Polynomial Data Fit for Pdel, Vrail and Vphase

There are a couple of potential uses for this data-fit model. One option we tested was to replace the interpolation algorithm in the Simulink model by the above data-fit equation to test out control algorithms for faster processing. This affords a significant speedup and could be used in a rapid prototyping approach for controller design (although the final controller would still be tested on the full Simulink model using interpolation data for most accurate results).

Another potential use would be to utilize an automated version of the above procedure for other plasma models (the second order fit serving as a prototype). The idea is to then use an approximate (static nonlinear) inverse to this equation in the controller design and implementation, so as to get the best controller, custom designed for a specific plasma model. The commissioning/calibration scheme would be done on-site for each system/application. Another idea would be the use of black box system identification methods and estimation methods. They could be used for online/offline system identification and would give a better idea about the load for choosing the best controller.

Nelder-Mead optimization approach was executed for a range of hardware and software parameters (source impedance, power setpoint and cable length) that require new controller designs for every combination of these parameters. This algorithm allows for a more general approach, since we are generating optimal controllers for every combination. These controllers can be used in a classification criteria and the best controller can be selected during commissioning of the generator.

# Bibliography

[1] Gideon van Zyl. Method and apparatus for modifying interactions between an electrical generator and a nonlinear load, August 4 2009. US Patent 7,570,028 B2.

[2] Gideon van Zyl. System and method for control of high efficiency generator source impedance, August 22 2017. US Patent 9,741,544 B2.

[3] K. J. McLaughlin, T. F. Edgar, and I. Trachtenberg. Real-time monitoring and control in plasma etching. *IEEE Control Systems Magazine*, 11(3):3–10, 1991.

[4] Vasilios Manousiouthakis, Robert Savage, and Yaman Arkun. Synthesis of decentralized process control structures using the concept of block relative gain. *AIChE Journal*, 32(6):991–1003.

[5] Junmo Koo, Damdae Park, Sangwon Ryu, Gon-Ho Kim, and Youn-Woo Lee. Online system identification for the real time control of the plasma parameters. In Mario R. Eden, Marianthi G. Ierapetritou, and Gavin P. Towler, editors, *13th International Symposium on Process Systems Engineering (PSE 2018)*, volume 44 of *Computer Aided Chemical Engineering*, pages 2041–2046. Elsevier, 2018.

[6] D. McFarlane and K. Glover. A loop-shaping design procedure using h/sub infinity / synthesis. *IEEE Transactions on Automatic Control*, 37(6):759–769, 1992.

[7] K. Glover and D. McFarlane. Robust stabilization of normalized coprime factor plant descriptions with h/sub infinity /-bounded uncertainty. *IEEE Transactions on Automatic Control*, 34(8):821–830, 1989.

[8] K. Glover and D. McFarlane. Robust stabilization of normalized coprime factors: an explicit h solution. In *1988 American Control Conference*, pages 842–847, 1988.

[9] Jeffrey Lagarias, James Reeds, Margaret Wright, and Paul Wright. Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on Optimization*, 9:112–147, 12 1998.

[10] J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313, 01 1965.

[11] George B. Dantzig, Alex Orden, and Philip Wolfe. The generalized simplex method for minimizing a linear form under linear inequality restraints. *Pacific Journal of Mathematics*, 5(2):183 – 195, 1955.