

Automated Propulsion Control

A Thesis

Submitted to the Faculty

of

Drexel University

by

Woodrow Clifton II

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy

April 2014

© Copyright 2014
Woodrow Clifton II. All Rights Reserved.

Dedications

*To Nydia and Caden,
every rung goes higher and higher.*

Acknowledgments

While pursuing this degree, I have had a number of people who supported me during this process. Each of their efforts are very much appreciated. Firstly, I would like to thank Dr. B.C. Chang for sticking with me during this process. Along with Dr. Chang, I would like to thank Dr. Harry Kwatny, Dr. Prawat Nagvajara, Dr. Wei Sun, Dr. Ajmal Yousuff and Dr. Jack Zhou for serving on my committee.

I would also thank a couple of gentlemen who completed their degree before me. These are Dr. Rodney M. Holland and Dr. Thomas E. Herald, Jr. In shifts, these guys continuously inquired of my progress. When things were not progressing as I would have liked, they would always encourage me with little nuggets from their own experiences.

I would like to thank my parents, Woodrow and Margaret Clifton. Throughout my life, they have always stood behind me and supported my efforts and desires. From the beginning, my Dad instilled a sense of discipline within me. To the end, my Mother always encouraged me to pursue my dreams.

Lastly, and certainly most importantly, I would like to thank my wife, Colette J. Stone Clifton. She has endured this process with me. There were times when things seemed bleak and I did not feel like continuing, she encouraged me to press on to completion. It goes without saying, I owe her immensely and for that I am forever grateful.

Table of Contents

LIST OF TABLES	vii
LIST OF FIGURES	viii
ABSTRACT	x
1. INTRODUCTION	1
1.1 Motivation	1
1.2 Literature Survey	2
1.3 Problem Formation	6
1.4 Solution Strategy	8
1.5 Contributions of the Thesis	9
1.6 Organization of the Thesis	10
2. PROPULSION SYSTEM	11
2.1 Propulsion Components	11
2.1.1 Propulsors	12
2.1.2 Propulsion Engines	13
2.1.3 Shafting	14
2.2 Propulsion Dynamics	14
2.2.1 Gas Turbine Engines Dynamics	14
2.2.2 Propeller and Shaft Dynamics	17
2.2.3 Ship Thrust Dynamics	19
2.2.4 System Equations	22
3. HYBRID CONTROL	25
3.1 Hybrid Systems Background	25
3.1.1 Classes of Hybrid Systems	25
3.1.2 Hybrid Petri Net Methods	25

3.1.3	Hybrid System Control	27
3.2	Propulsion Controller	28
3.2.1	Controller/Plant Interfaces	29
3.2.2	APC Implementation Approach	31
3.2.3	Continuous System Control	32
3.2.4	Controller Discrete Logic	46
4.	PROPULSION MODEL SIMULATION	54
4.1	Propulsion Model	54
4.1.1	Propulsion Plant Model	54
4.1.2	Hybrid Control Model	58
4.1.3	Test Input Model	61
4.2	Simulated Results	62
4.2.1	Automatic Cold Start	62
4.2.2	Increase Ship Speed	64
4.2.3	Reduce Ship Speed	65
5.	CONCLUSION	69
5.1	Summary of Research	69
5.2	Future Research Opportunities	70
	LIST OF REFERENCES	72
	APPENDIX A: LIST OF SYMBOLS AND ABBREVIATIONS	75
A.1	List of Symbols	75
A.2	Abbreviations	77
	APPENDIX B: SYSTEM EQUATIONS LINEARIZATION	78
B.1	Trailshaft Mode	78
B.2	Split Plant Mode	79
B.3	Full Power Mode	80
	APPENDIX C: DISCRETE CONTROL RATIONALE	82

C.1	Reconfigure Propulsion Plant	82
C.2	Align to Full Power	83
C.3	Bring Engine Online	84
C.4	Align to Split Plant	84
C.5	Align to Trailshaft	90
C.6	Align to Off	91
C.7	Power Turbine Brake Commands	92
	APPENDIX D: ADDITIONAL TEST SCENARIOS	93
D.1	Emergency Stop Engine	93
D.2	Engines Unavailable	93
D.3	Shifting of Engines	94
D.4	Issue Manual Commands	96
	VITA	98

List of Tables

3.1	Polynomial Coefficients for Throttle Input Command to Fuel Flow	36
3.2	Goodness of Fit for Throttle Input Command to Fuel Flow	36
3.3	LQR Output of Linearized Model by Ship Mode	41
4.1	Gas Turbine Model Inputs	55
4.2	Gas Turbine Model Outputs	56
4.3	Propulsion Shaft Inputs and Outputs	57
4.4	Hybrid Controller Outputs	58
4.5	Hybrid Controller Inputs	59
4.6	Test Input Signals	62
B.1	Trailshaft Operating Points	78
B.2	Split Plant Operating Points	79
B.3	Full Power Operating Points	80

List of Figures

1.1	Hybrid System with Continuous States	8
2.1	Physical Layout of a Ship Propulsion System	11
2.2	Gas Generator Speed Control Algorithm	16
2.3	Power Turbine Constraint Algorithm	16
2.4	Propeller Pitch and Slip	19
3.1	System Block Diagram	29
3.2	Controller Functional Diagram	29
3.3	Evaluation of Continuous Portion of Hybrid Controller	31
3.4	Hybrid Controller Block Diagram	32
3.5	Correlation of Throttle Input Command to Fuel Flow	36
3.6	Continuous Controller Block Diagram	45
3.7	Discrete Logic of Controller	46
3.8	Discrete Controller Block Diagram	51
3.9	Discrete Logic Flow Diagram	51
4.1	Automatic Cold Start Engine Commands	63
4.2	Automatic Cold Start Ship Speed	64
4.3	Increasing Ship Speed Engine States and Throttles	65
4.4	Increasing Ship Speeds	66
4.5	Reducing Ship Speed Engine Start/Stop Commands and States	67
4.6	Reducing Ship Speeds	68
C.1	Reconfigure Propulsion Plant	82
C.2	Command Full Power	83
C.3	Bring Engine Online	84
C.4	Command Split Plant	86

C.5 Split Plant — Outboard Engines	86
C.6 Split Plant — Inboard Engines	87
C.7 Split Plant — Start 1 Engine	87
C.8 Split Plant — Start Outboard/Inboard Engines	88
C.9 Split Plant — Start 2 Engines	88
C.10 Split Plant — Power Down Engines	89
C.11 Command Trailshaft	90
C.12 Trailshaft — Start Engine	90
C.13 Reset Engine Alarms/Errors	91
C.14 Power Turbine Brake Logic	92
D.1 Emergency Stop Engine Commands and Ship Response	94
D.2 Engine Unavailable Engine Commands and Ship Response	95
D.3 Engine Shifts Engine Commands and Ship Response	96
D.4 Manual Engine Commands and Ship Response	97

Abstract
Automated Propulsion Control
Woodrow Clifton II
Dr. B.C. Chang

Propulsion systems that power the ships of today's navies require some level of operator input. These inputs include manually starting and stopping propulsion engines and adjusting the throttle, which commands the speeds on the engines that power the ship. Ultimately, the job of the operator is to ensure that the ship safely travels at a desired speed. This objective can be achieved without much human interaction, but with the implementation of an automated propulsion controller that is presented in this work.

The automated propulsion controller uses hybrid control theory to automatically achieve both the continuous and discrete control objectives of which the operator is usually responsible. The approach presented includes an integrated hybrid controller, which applies the outputs of the continuous control algorithms as inputs to the discrete logic and the outputs of the discrete logic are inputs of the continuous control algorithms. Various continuous control approaches are explored before a rather simplified control approach is implemented. The discrete control logic is designed using a Petri net approach with the objective to align the ship to its optimum configuration. The hybrid controller is modeled, along with a ship propulsion system, to verify its desired functionality and to ensure that the primary control objectives are achieved. In multiple tests, the automated propulsion controller successfully achieved its desired objectives.

Chapter 1: Introduction

Hybrid Control is the intersection between continuous control and discrete control. Continuous control applies to systems that have a constant input applied to maintain a desired set point. Discrete control applies to systems that have discrete events, which causes the system to transition to a different state. Therefore, hybrid control applies to systems that must maintain a desired system set point that corresponds to a given system state.

An example of a hybrid system, which can be governed using hybrid control, is a ship. A ship can be operated in various modes. Each of these modes equates to a system state. Each of these states have their associated operating limits. When a desired ship speed is commanded, hybrid control can be used to ensure that the ship is operating in its optimal system state.

1.1 Motivation

Automatic ship control is a desire of many of today's navies. Newer ships, such as the Zumwalt Class Destroyer, are being designed and built with much more automation than ships of the previous generation. With a few minor hardware changes and a software upgrade, new automatic ship control algorithms can be implement into older Navy ships, such as the Arleigh Burke Class Destroyers.

Today, there are no automated approaches for controlling a ship's propulsion system within the Arleigh Burke Class Destroyers. When the ship is commanded to achieve a desired speed, an operator is required to adjust the throttle in an attempt to achieve the desired speed. However, if that speed can not be achieved and additional resources are available, those available engines are commanded ON to supply additional power. The number of online engines define the operating mode of the ship. When speed is not of the essence, the full power mode, where four propulsion engines power the ship, is not used, but the ship is run in either Split Plant or Trail Shaft mode. In Split Plant mode, the starboard propeller is powered by only one of its two available engines. Similarly, the port shaft is powered by a single engine as opposed to two. However, in the cruise

speed range of approximately 20 knots or less, the engines are not fully loaded and fuel efficiency would have degraded 30% or more below full power design values. Because of this, Trail Shaft mode is used at lower speeds. In this mode of operation, only one engine is powered on a single shaft. The other non-powered, or trailing, shaft spins freely. The fuel efficiency benefits of operating on a single more highly loaded propulsion engine have been shown to more than compensate for the increased hydrodynamic losses of asymmetric powering. Pressures to reduce fuel usage are resulting in increasing use of trail shafting in the existing fleet [1].

With the addition of an automated supervisory controller that fully controls the propulsion system, which includes the continuous engine and shafting dynamics and the discrete states of the system, personnel resources could be minimized and a systematic approach to ship control can be implemented. By reducing the number of ship operators required to man the ship, few individuals would be in harms way when strategic mission position the ship in hostile environments. Additionally, when automated algorithms are implemented, a more systematic, known control method can be employed. With humans in the control loop, errors are prone to happen which could inevitable lead to failures that could have been avoided. These are just a few reasons that motivated this research.

1.2 Literature Survey

There are numerous textbooks that discuss various issues regarding propulsion systems and their associated control. Some of which include Basic Principles of Ship Propulsion [2], Maritime Engineering Reference Book - A Guide to Ship Design, Construction and Operation [3] and Marks' Standard Handbook for Mechanical Engineers (11th Edition) [4], where basic principles of ship propulsion and marine engineering are discussed. In Principles of Naval Architecture Volume II [5], ship resistance, propulsion and vibration are the main items discussed, In Volume III [6], the focus is on motions in waves and controllability. In Introduction to Naval Architecture [7], relevant topics include discussions on general topics associated to ship propulsion, ship dynamics and resistance forces that the ship may encounter. Computer based tools and naval surface ships are discussed in Ship Design and Construction, Volumes 1-2 [8], [9]. The general principles of powering a ship and associated applications are discussed in Basic Ship Theory [10]. The Handbook of Human Sys-

tems Integration [11] mainly discusses several non-propulsion related topics. However, a section is included that discusses Human-Centered Shipboard Systems and Operations.

Because of the controlling complexity of gas turbines that power the ship, these units normally have their own lower level controller which processes inputs and outputs at 5 to 100 millisecond intervals. Control specifications are often documented by the gas turbine manufacturer. For example, control requirements for General Electric's LM2500 gas turbine engine are outlined in M50TF3832-S2 [12] and M50TF3886-S15 [13].

Lockheed Martin manufactures marine gas turbine engine controllers. In [14], Petrey discusses how Ada was used in Lockheed Martin's Universal Engine Controller to control General Electric's LM2500 gas turbine engine. The unit's hardware configuration and associated interfaces are discussed along with some discussion on the controller's self test features, which are used to monitor the health of the unit. However, an in depth discussion on the control algorithm used to govern the gas turbine is lacking.

Lockheed Martin engine controller testing ranges from that of the digital fuel control option installed on the Universal Engine Controller discussed in [15], [16] and [17] to most recently the Backfit Engine Controller, which is discussed in [18]. In [16] and [15], Howell reviews the integration and test process and how hardware in the loop was used to test the engine controller. The hardware in the loop consisted of a test stand that included a simulated engine that was developed in MatrixX. These simulated test were conducted prior to testing the engine controller with a real engine. A similar process was used for the Backfit Engine Controller as discussed in [18]. General Electric's Provisional Certification for adhering to the control specifications was granted after test vector analysis and completing tests on the Engine Controller Automated Test Stand (ECATS). A MATLAB/Simulink model of the engine was embedded in the ECATS. Actual engine testing at the Land Based Engineering Site (LBES) in Philadelphia, PA for the Universal Engine Controller is documented in [17].

In 1983, McMahon outlined a technical discussions associated to marine propulsion systems [19]. The focus of the paper was on basic propulsion system operating principles and mechanical drive

systems that include reduction gears, line shafting, clutches, etc. Control requirements for these sub-systems were discussed along with associated subsystem interface requirements. Control operation modes (manual and programmed (pseudo-automatic)), were also discussed.

The U.S. Navy Arleigh Burke class Machinery Control System (MCS) is reviewed in [20]. The MCS is responsible for control and monitoring of the ship's electrical, propulsion, auxiliaries and damage control systems by using a distributed architecture. Hardware and software components are discussed, along with their associated interfaces. Integration and test efforts and performance in service are also reviewed.

In [21], Banning, Johnson and Grimble present a control design procedure that combines linear optimal H_∞ control theory with non-linear control techniques used to address nonlinearities of the propulsion system. The turbo-charged marine diesel engine propulsion system model was discussed along with the control strategy. Simulation results were reviewed and showed that the controller satisfied the control objective of saving fuel and optimizing efficiency while having adequate tracking capabilities.

Grimble, Carr, and Katebi describe an integrated control system for total ship control in [22]. Four coupled subsystems for Surge, Roll, Sway/Yaw and Heave/Pitch are presented. A H_∞ robust control design technique is employed to account for subsystem interaction. After the claim, examples are presented.

Kashima and Takata investigate an optimal control approach for a marine propulsion system in [23]. The propulsion plant is first mathematically modeled. The performance criterion for the optimal trajectory formations is defined with propeller angular acceleration to accomplish smooth transitions of the system at any movement condition. A numerical algorithm is then developed based on the Transition-Matrix-Algorithm. Computer simulations of the optimal trajectories are then discussed. It is concluded that the optimal control scheme accomplishes smoother responses of the state variable and improvements to ship speed response was observed. However, it is proposed that development of an online optimal control scheme for more flexible control systems be further investigated.

Juang and Chang apply robust control theory to a ship control problem with plant uncertainties caused by wind-generated waves and commanded ship speed changes in [24]. Nomoto's model was used as the ship model. A wave model, a sensitivity weighting function, control weighting function, pre-compensator and uncertainty associated to the ship speed changes were all characterized. Perturbation blocks were added for robust performance consideration. A robust controller was obtained using μ -synthesis, where an optimal H_∞ -controller was found. An example of a merchant cargo ship was then presented. The paper concluded that the μ -synthesized system has longer settling time for step tracking, but has much better robust stability and performance.

In [25], a hardware in the loop propulsion plant simulation is discussed. Allen Bradley programmable logic controllers (PLC) are used as the control hardware platform for the propulsion plant. Plant equipment is modeled in MATLAB/Simulink. The model is executed using The MathWorks Real-Time Workshop® tool and xPC TargetTM. Magma peripheral component interconnect (PCI) extension boxes are used to interface the models inputs and outputs to the terminal blocks of the the PLC. The study merely proved that a PLCs platform could be used for propulsion control. However, propulsion control algorithms were not discussed in this paper.

In [26], Cantrell discusses how automation controllers, such as those provided by Siemens, can be used to control combined propulsion plant equipment. The combined propulsion plant is defined to be a propulsion system that uses more than one engine to power a single propulsion line. The engines may or may not be the same type and may have different power ratings. Automation controllers can be used with lower level controllers, such as those for the propulsion engines, and other local operator panels. Automation controllers provide some advantages over larger controlling consoles. Similar to [25], this paper merely discusses a hardware platform and no discussion is made to algorithm development.

In [27], Leal and Cury discuss threshold-event-driven hybrid systems, where the changes in the discrete state can occur only when continuous state variables encounter specified thresholds. The continuous dynamics are determined by the discrete condition/state. A special MATLAB toolbox was developed and its used discussed. The steps used to develop the solution include:

1. Build the hybrid system model.
2. Generate a finite state condition/event automaton that represents an approximated logical behavior of the hybrid system.
3. Calculate “modular” supervisors to monitor specified portions of the automaton.

The solution required by Azhmyakov, Boltyanski and Poznyak in [28] introduces an auxiliary hybrid optimization control problem that is governed by a hybrid system with autonomous location (places) transitions without jumps in the continuous state. This replaces the original hybrid control system. A Lagrange approach is then applied. For a hybrid trajectory with defined switching times, a set of equations are solved in an iterative process to determine the reduced cost function.

Bemporad and Giorgetti discuss similar approaches in [29] and [30]. (The introduction of [29] includes a good history on hybrid systems). The solution for the optimal control of the hybrid systems involve reformulating the discrete portion of the system into a piecewise affine (PWA) system in [29] and a mixed integer linear programming (MILP) in [30]. The optimal solution is found by applying a satisfiability logic based bound and branch algorithm. In [29] a supply chain management example is presented, while a motorbike gearing systems is presented in [30].

From the papers reviewed, the solution includes either an algorithm that must be followed or requires the iterative calculations of a set of equations. When considering the approach offered by [27], the proposed research does not include defined thresholds. The methods employed in [28], [30] and [29] require “conversion” of the hybrid system to determine the optimal solution. Ideally, the hybrid system should be maintained or at a minimum analyzed as a discrete event system with continuous components.

1.3 Problem Formation

The dynamics of a ship align with Newton’s Second Law of Motion,

$$\begin{aligned}
 F &= ma & (1.1) \\
 F &= m \frac{dV}{dt}
 \end{aligned}$$

$$\begin{aligned}\frac{dV}{dt} &= \frac{F}{m} \\ \dot{V} &= \frac{F}{m}\end{aligned}$$

The speed of a ship, V , is then

$$V = \int \frac{F}{m} dt \quad (1.2)$$

The force, F , used to power a ship is generated from the thrust, T , produced by the ship's propellers.

As the propellers rotate, thrust is produced. The rotational speed of the propellers is a function of shaft speed

$$\begin{aligned}Q &= I\alpha \quad (1.3) \\ Q &= I \frac{d\omega}{dt} \\ \frac{d\omega}{dt} &= \frac{Q}{I} \\ \dot{\omega} &= \frac{Q}{I}\end{aligned}$$

The shaft speed, ω , is then

$$\omega = \int \frac{Q}{I} dt \quad (1.4)$$

The above equations align to continuous dynamics. However, the shafts are powered by multiple engines, which can be ON or OFF. These discrete states of the engine contribute to the motor torque used to power the ship.

$$Q_E = \sum Q_{e_i} \quad (1.5)$$

where Q_E is the total torque provided by all of the engines and Q_{e_i} is the torque provided by each individual engine. For each ON engine, the continuous dynamics of the input throttle command to outputs power turbine speed and associated torque used to rotate the shaft should also be considered.

Differing sets of equations define the ship dynamics based on the discrete state of the engine. Therefore, when considering a controller for the ship, a hybrid controller is required to govern both the discrete states and continuous dynamics. An automatic controller that governs each of the

engines, which are used to achieve a desired ship speed is discussed in this thesis.

1.4 Solution Strategy

A ship, which has multiple engines and more than one shaft, can be aligned in multiple configurations. The configuration of the systems defines the operating mode of the ship. Each operating mode has its own set of dynamic equations. The continuous dynamics are determined by a discrete condition that depends on the current discrete state (mode) of the system [27]. An example of this type of system is displayed in Figure 1.1, taken from [31]. In this figure, each of the discrete states contain

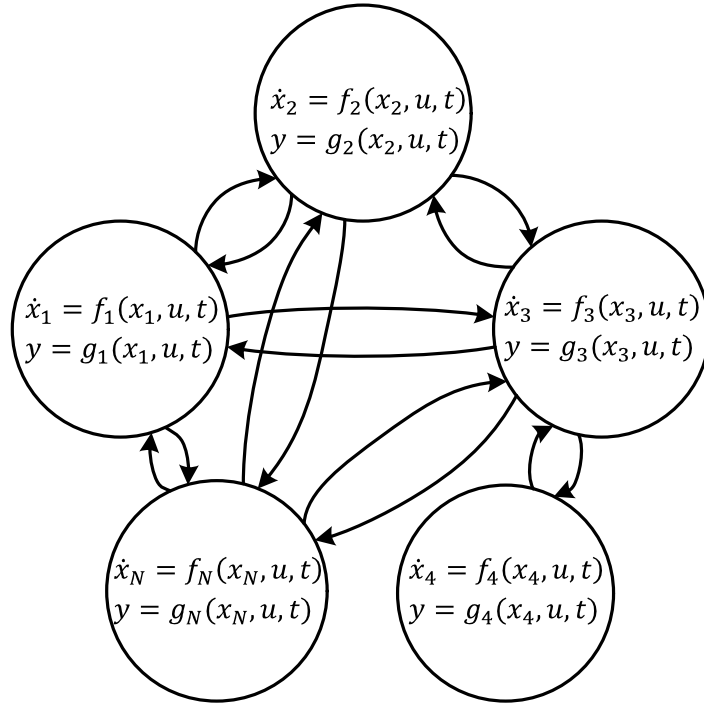


Figure 1.1: Hybrid System with Continuous States

its own set of time-dependent continuous equations,

$$\begin{aligned}\dot{x}_i &= f_i(\mathbf{x}, \mathbf{u}, t) \\ y &= g_i(\mathbf{x}, \mathbf{u}, t)\end{aligned}\tag{1.6}$$

where $i = 1, 2, 3, 4, \dots, N$ and is its own discrete state. The discrete state of the system transitions to other states as defined by the arcs displayed. These discrete transitions occur when commands

are issued to update the mode of ship.

The controller for the ship must ensure that the ship is able to maintain a desired ship speed. When an updated ship speed command is issued, the controller must align the propulsion system by commanding ON or OFF engines to achieve the updated speed commanded. A hybrid controller, which issues both discrete and continuous commands, is presented to satisfy the problem discussed in Section 1.3.

The controller uses a Proportional-Integral (PI) algorithm approach to govern the continuous dynamics. More complex discrete logic is used to govern the state (ON or OFF) of each of the engines. When the desired speed can be achieved and additional power is required, the discrete logic commands ON the “best” engine or set of engines. When the available power is more than what is required, the discrete logic identifies the appropriate engine, or set of engines, to power OFF.

1.5 Contributions of the Thesis

There are two primary contributions presented in this thesis:

1. An integrated approach to a hybrid controller is presented.
2. An automated controller is presented that could replace the functions of an operator, thus reducing the required manpower required to operate the ship.

While developing the controller presented in this work, the integration of continuous control algorithms with discrete logic was presented. The resulting hybrid controller was integrated in such a way that the output of continuous control algorithms were used as key inputs to the discrete logic. Similarly, the outputs of the discrete logic were inputs used in the control algorithms. Numerous approaches were investigated while developing the continuous control algorithms. The discrete logic aligned with a Petri net approach. Together, this integrated hybrid controller approach proved effective in satisfying its control objectives.

After the integrated hybrid controller was developed, it was used in the Automated Propulsion Controller. This controller can be used to automatically control the propulsion system of a ship, such that it can effectively achieve a desired speed with minimal human interaction. Today, the propulsion

plant operator is required to start and stop the engines on the ship and adjust the throttle so that the ship can achieve its commanded speed. With the implementation of the Automated Propulsion Controller, the role of this operator is not required, thus reducing the manpower required to operate the ship. After receiving a desired ship speed, the controller is responsible for appropriately aligning the propulsion plant of the ship and commanding the online engines to achieve the desired speed.

1.6 Organization of the Thesis

After this introductory chapter, the basics of a propulsion system are presented in Chapter 2. This includes a description of the components of the propulsion system and development of the system equations that are used to represent the propulsion system. Within Chapter 3, the development of the hybrid controller used to control the propulsion plant is documented. This chapter also includes both a general background to the approaches investigated and the detailed information of the continuous algorithms and discrete logic embedded in the hybrid controller. To verify the functionality of the hybrid controller, a Simulink model was developed and realistic test scenarios were executed. This model and corresponding test results are included in Chapter 4. Concluding remarks and recommendations for future improvements are documented in Chapter 5. Additionally, appendices are included to define terms and variables used this work, provide more details for the system equation development, rationale for the discrete logic and additional test results, which were not included in the main body of the thesis.

Chapter 2: Propulsion System

2.1 Propulsion Components

The propulsion system of a ship is primarily composed of two components; a propulsor and a propulsion plant. The propulsor is a mechanical device that is used to drive the ship through the water. The propulsion plant is used to power the propulsor. As displayed in Figure 2.1 from [32], a representative physical layout of the propulsion system can be observed. Included in this figure are

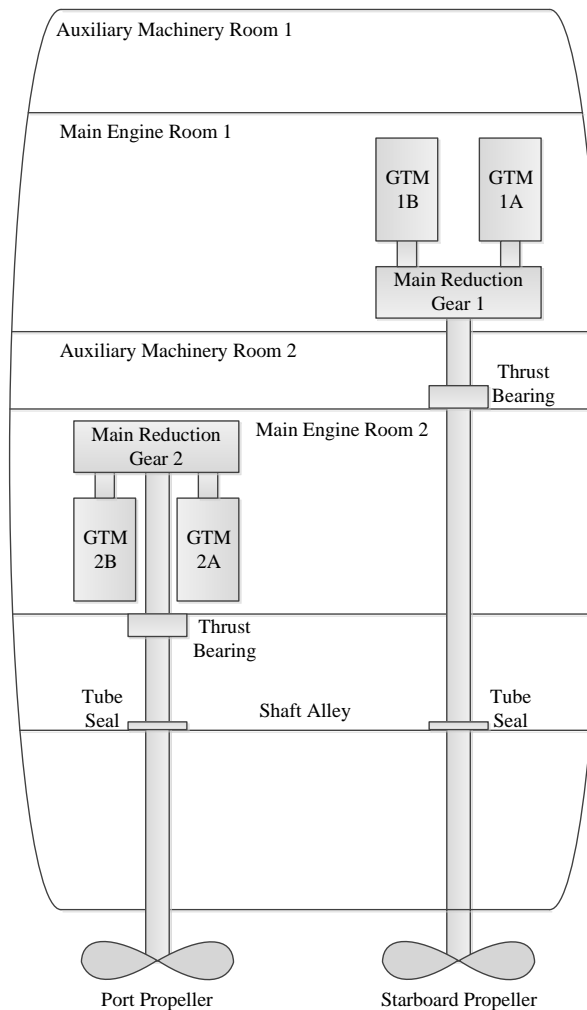


Figure 2.1: Physical Layout of a Ship Propulsion System

two propulsors, which are the propellers at the bottom of the figure. Also included is the propulsion

plant, which includes four propulsion engines, denoted as GTMs in the figure, and shafting, which includes the shaft and main reduction gear. More details on these components are discussed below.

2.1.1 Propulsors

There are primarily two devices that can be used for ship propulsion. They are a screw propeller and a waterjet. The most common of the two propulsors is a screw propeller. Ship propellers are classified in two categories; fixed and controllable pitch.

On a fixed propeller, the blades are mounted with a defined pitch angle, which cannot be changed by any means. The ship speed and direction of a fixed pitch propeller is solely based on the rotational speed and direction of the shaft. As the rotational speed of the shaft increases, the ship speed increases. Whereas, as the rotational speed of the shaft decreases, the ship speed also decreases. The direction of the shaft determines which way the ship will go. If the blades of the shaft are oriented such that the ship will traverse in the forward direction when the shaft rotates clockwise, when the shaft rotates counter-clockwise, the ship will traverse in the astern (backwards) direction.

On a controllable pitch propeller, the blades are mounted so that the blades can be adjusted to any desired pitch angle. The pitch angle is adjusted by hydraulic oil within a servomotor cylinder. The servomotor piston is connected to a piston rod, which is contained in and extends the length of the propeller drive shaft. The opposite end of the piston rod is connected to mechanical linkages housed in the hub assembly of the propeller. When the piston rod is extended or retracted, the linkages adjust the orientation of the propeller blades. When hydraulic oil is ported within the servomotor from the astern reservoir to the forward reservoir, the blades of the propeller are adjusted and oriented in the forward direction. Conversely, when oil is ported from the forward reservoir to the astern reservoir, the propeller blades are rotated to drive the ship in the astern direction [33]. The shaft speed and pitch angle are both used to determine the speed and direction of the ship. Given a constant shaft speed, the speed of the ship can be varied by adjusting the pitch angle. Neglecting the hydroelastic properties of water or slip, the theoretical ship speed can be determined by the pitch angle and rotational speed of the shaft.

2.1.2 Propulsion Engines

Propulsion plants that power the propulsor come in a variety of types. The three main types are steam plants, diesel engines and gas turbines. Also, it is not uncommon for a ship to utilize any combination of steam plants, diesel engines and gas turbines.

Steam plants are composed of the main boilers, steam turbines, a feedwater system and a condensate system. The options for main boilers are coal-fired, oil-fired or gas-fired boilers. Coal-fired boilers are not favored as they require additional work and storage to handle ashes. Oil-fired boilers are preferred on most ships, but gas-fired boilers remain as an additional option. The main boiler essentially produces the steam that is used to power the steam turbine. The steam turbine, then, provides the rotational power to turn the shaft propeller. The feedwater system is composed of the pumps, piping and controls used to provide feedwater to the boilers to generate the steam. The condensate system is used to recover and return the feedwater to the feedwater system.

Steam plants may also utilize nuclear powered reactors. The nuclear reactor essentially replaces the main boiler. An interesting fact about nuclear power is that the fission of 1 pound of uranium is equivalent to the combustion of about 86 tons, or 87,380 kg, of 18,500 Btu/lb fuel oil [4]. However, the main disadvantage of nuclear power is the safety risk that is associated to it. This risk requires an abundance of additional training of ship personnel.

Diesel engines are the most common type of engine used for ship propulsion. Not only are diesel engines used in naval vessels, they are used as fishing boats, tugs, ferries, dredges, river boats, cargo ships and tankers. Diesel engines are manufactured to be either low, medium or high speed engines. Low speed engines operate in the range of less than 300 RPM, while high speed engines operate in excess of 1200 RPM.

Gas turbines offer a small, light weight option for ship builders when compared to the medium and high speed diesel engines. Gas turbines can accelerate quickly. These turbines can go from off to online and power the ship in approximately a minute. Originally used in the airline industry, gas turbines have been modified and adapted for maritime use. Marine gas turbines are modular in design and are often referred to as a GTM or gas turbine module. The GTM is composed of

three primary parts; a gas generator, compressor and a power turbine. The gas generator sucks in air, which is then compressed in the compressor. The energy created when the compressed air is decompressed can turn the power turbine at speeds in excess of 3500 rpm. One of the most common marine gas turbines is the General Electric LM2500 Marine Gas Turbine. This gas turbine has been used in more than 400 ships in 30 world navies, fast ferries, coast guard cutters, supply ships and cruise ships. Collectively, the estimated 1200 engines have more than 11 million hours in naval service [18].

2.1.3 Shafting

With the exception of low speed diesel engines, which can be connected directly to a propeller's shaft, most ships require a reduction gear. In these cases, the propulsion engine is connected to the reduction gear, which is then connected to the shaft. The purpose of this reduction gear is to translate the high speed, low torque output of the propulsion engine to the low speed, high torque required to turn the propulsion shaft. In some cases a reduction gear can be connected to more than one propulsion engine. In these cases, a clutch is used to enable the output of an engine. When the engine is clutched into the reduction gear, or engaged, the engine is said to be online and provides rotational power to turn the shaft. When multiple engines are online, the shaft has the potential to rotate faster than when a single engine is online.

2.2 Propulsion Dynamics

Each of the propulsion system components mentioned in the previous section can be modeled for analysis. To properly model these propulsion systems, the dynamics of their governing system equations must be known. This section addresses the dynamics of the key components of the propulsion plant.

2.2.1 Gas Turbine Engines Dynamics

The propulsion engine chosen for this study is the gas turbine engine. The dynamics and control of a gas turbine engine are extremely complex. Because of the complexity of this intricate piece of equipment, gas turbines are controlled by their own local controller as opposed to a higher level

controller, such as one that may control the entire propulsion system. Even though the propulsion system controller does not directly control the gas turbine engine, it does interact with the engine controller. The propulsion system controller sends commands to the engine controller, which in turn issues the corresponding command to the engine. As the engine controller monitors the state of the engine through numerous feedback signals, a subset of these signals are sent from the each engine controller to the propulsion system controller to ensure that the entire propulsion system is being controlled in accordance to its overall control objectives.

The primary input to a gas turbine is the throttle input command. This command maps to a desired gas generator speed of the gas turbine. To achieve this speed, the fuel metering valve of the gas turbine is slightly adjusted from its current position to be either more opened or more closed. Figure 2.2 taken from [16] highlights a small portion of the complex algorithm used in the engine controller.

The engine controller uses some of the monitored signals to control the operation of the gas turbine. For example, as displayed in Figure 2.3, also from [16], control loops are in place to limit the gas turbine's power turbine torque, power turbine speed and power turbine acceleration. These three control loops are some of the first to be implemented in an engine controller. Over the years, additional control loops have been implemented for increased control and efficiency. These additional loops are discussed in [12].

In addition to monitoring engine output parameters and governing the engine per the above control loops, the engine controller also generates alarms when certain parameter values are exceeded or out of a specified range [13]. Depending on the parameter and its associated value, the engine controller may initiate a shutdown of the engine to ensure the safety of the equipment and those who may be in its vicinity.

When considering the possibility of a local controller issuing an engine shutdown, the engine control being varied between one of a dozen control loops and the engine control being based on numerous tables as shown in Figures 2.2 and 2.3, the system equations that govern the dynamics of a gas turbine are extremely nonlinear. These nonlinearities can easily be modeled as a plant system

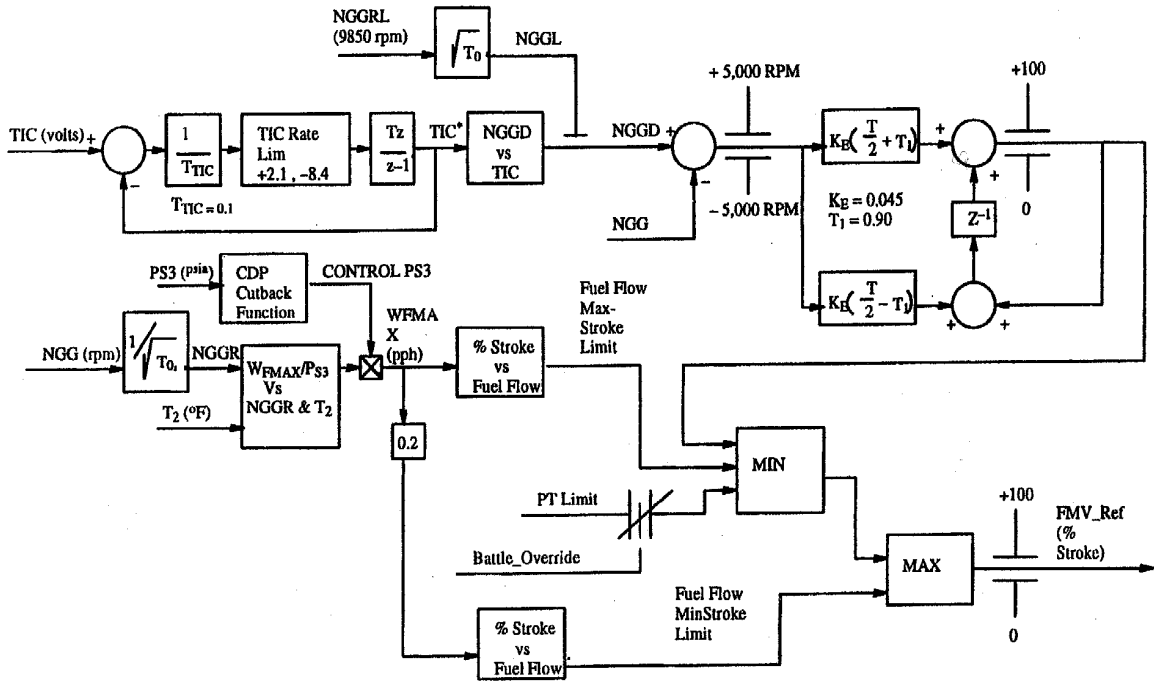


Figure 2.2: Gas Generator Speed Control Algorithm

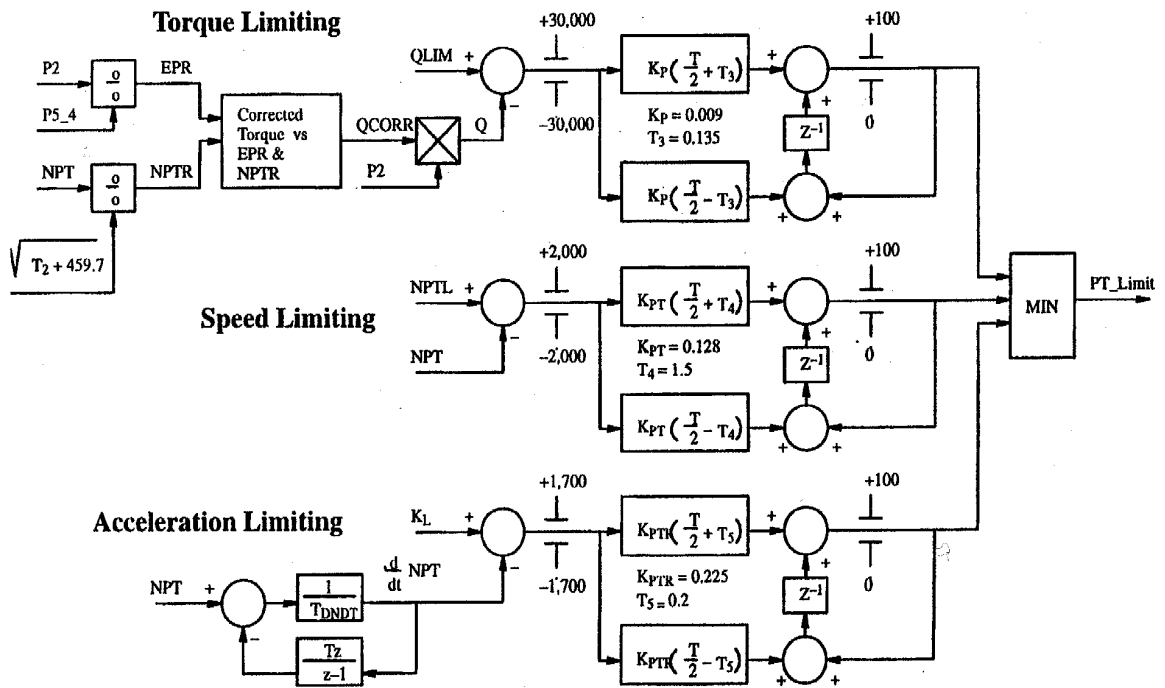


Figure 2.3: Power Turbine Constraint Algorithm

in MATLAB/Simulink. However, for control analysis, it is advisable to collect a subset of data and linearize this data.

2.2.2 Propeller and Shaft Dynamics

Understanding the dynamics of a propeller are accomplished by realizing that rotational forces (torque) cause the propeller turns in water. When the propeller turns, axial forces (thrust) are generated. These axial forces then propel the ship through water.

Beginning with the equation of motion to determine the rate of change of the angular momentum of a rigid body about its mass center, the unbalanced torque (Q) is calculated as shown in Equation 1.3 [34] where I is the moment of inertia of the rigid body about a centroidal axis perpendicular to the rigid body and α is the angular acceleration, which is the derivative of the angular velocity ω ($\alpha = \frac{d\omega}{dt}$). In this form, the units are in radians per second (RPS).

The rotational speed of the propeller is a function of the unbalanced torques. (If the sum of all of the rotational forces applied on the propeller were equal to zero, this would yield a static condition and the shaft would not turn). Therefore, the sum of the applied torques on the propeller are calculated to determine the unbalanced torque, Q_u .

$$\begin{aligned} Q_u &= \sum_i Q_i \\ &= Q_d - Q_p - Q_f \end{aligned} \quad (2.1)$$

where Q_d is the delivered torque, Q_p is the torque from the propeller and Q_f is the frictional losses from the shaft.

The delivered torque is the product of the reduction gear and coupled engine output

$$Q_d = K_g \sum_1^2 Q_{e_i} \quad (2.2)$$

When an engine is online, or clutched into the reduction gear, it provides the rotational forces (Q_e) to turn the reduction gear. The generated forces from each online engine are summed to determine

the total torque applied to the reduction gear. As previously stated, the reduction gear translates, the engine(s) output to the torque required to turn the shaft. The reduction gear constant (K_g) represents the ratio from engine output to delivered torque.

The frictional losses Q_f are primarily a function of the shaft speed ω and its associated frictional forces. These frictional forces are attributed to losses from the shaft's bearings and the reduction gear including its connections to the power turbines of the engines. The frictional losses are calculated as

$$Q_f = f(K_f, \omega) \quad (2.3)$$

where K_f is the shaft's coefficient of friction. This value tends to vary between ships and is often determined by using available ship data.

One approach for calculating the frictional losses is to use the shaft's moment of inertia to yield the equation

$$Q_f = K_f I \omega^2 \quad (2.4)$$

This approach ensures that the units of frictional torque are in the correct form.

The propeller torque Q_p is calculated by applying the following equation,

$$Q_p = K_q \rho \omega^2 d^5 \quad (2.5)$$

where d is the diameter of the propeller, ρ is the density of water and K_q is the propeller torque coefficient. This coefficient is varied based on the propeller pitch ratio and the advance coefficient of the propeller.

$$K_q = f(J, pr) \quad (2.6)$$

The pitch ratio is the ratio between the propeller pitch and the propeller diameter. Propeller pitch is measured to be the distance that the propeller turns itself forward, or screws, in one revolution if it were going through a solid material. Additionally, pitch is measured at $0.7r$, where r is the radius of the propeller and half of the propeller diameter ($r = \frac{d}{2}$). See Figure 2.4 for further details. The

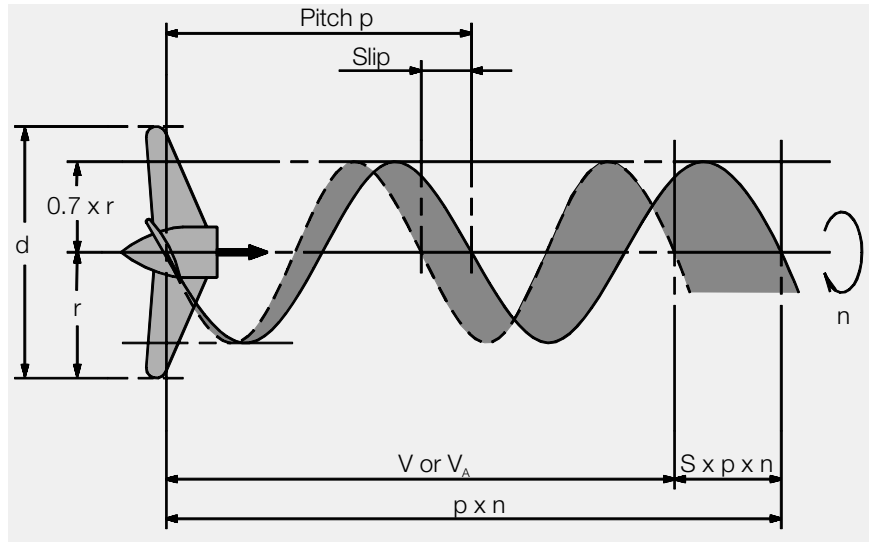


Figure 2.4: Propeller Pitch and Slip

advance coefficient, J , is found by

$$\begin{aligned} J &= \frac{V(1-w)}{\omega d} \\ &= \frac{V_a}{\omega d} \end{aligned} \quad (2.7)$$

where the added variables are V for ship speed and w , which represents the wake coefficient. The speed of advance (V_a) is the speed of water at the propeller. This speed is less than the speed of the ship as the wake forces slightly assist in propelling the ship through the water.

Now, knowing each of the components of the unbalance torque, Q_u , as defined in Equation 2.1 and understanding the relationship between torque and shaft speed from Equation 1.3, which can be modified to find the shaft speed by

$$\dot{\omega} = \frac{Q_u}{I} \quad (2.8)$$

2.2.3 Ship Thrust Dynamics

Knowing the speed of the shaft is one major component of calculating the speed of the ship. The shaft speed is a component of determining thrust, or the main force used to move a ship through water. Recalling Newton's second law of motion as shown in Equation 1.1, where, the force component is

composed of ship thrust less the towing resistance. Ship thrust, T , is calculated by knowing the propeller thrust less a deduction factor, or

$$T = T_p(1 - t_d) \quad (2.9)$$

where T_p is the propeller thrust and t_d is the thrust reduction factor. The propeller thrust is found by calculating

$$T_p = K_t \rho \omega^2 d^4 \quad (2.10)$$

Here, the new term that is being added is K_t or the propeller thrust coefficient. The propeller thrust coefficient is similar to the propeller torque coefficient (K_q) and is varied based on the values of the pitch ratio, pr , and the advance coefficient, J . Therefore

$$K_t = f(J, pr) \quad (2.11)$$

The thrust reduction factor varies based on the direction of the ship and is calculated as

$$\begin{aligned} t_d &= +0.1 \frac{T_p}{T_{p_{max}}} \quad \text{ahead} \\ &= -0.2 \frac{T_p}{T_{p_{max}}} \quad \text{astern} \end{aligned} \quad (2.12)$$

where $T_{p_{max}}$ is the maximum propeller thrust.

All variables required to calculate the thrust component of force are now known. The other variable required in determining the applied force is the towing resistance, which is calculated by knowing the resistance forces that are applied due to air and water.

$$R_t = R_f + R_r + R_a \quad (2.13)$$

These resistance forces include frictional resistance (R_f), residual resistance (R_r) and air resistance (R_a).

To determine these forces applied on a moving ship, Bernoulli's equation, which states

$$p_d = p_s + \frac{1}{2}\rho V^2 + \rho gh \quad (2.14)$$

is first considered. Here, the dynamic pressure applied on the moving ship (p_d) is found by knowing the static pressure (p_s), density of water (ρ), ship's speed (V), gravitational acceleration (g) and elevation (h). Considering there is no static pressure being applied and the ship is at sea level (no elevation), the first and third term of Equation 2.14 can be neglected. The pressure applied to the moving ship is now solely based on the second term, which includes the density of water and the ship's speed. The resulting force is found by multiplying this pressure by the affected area. If the applied forces due to water are first considered, the affected area of the ship is the wetted surface area, or the surface area of the ship that is underwater. This resulting force due to water is

$$K = \frac{1}{2}\rho V^2 A_s \quad (2.15)$$

where A_s is the ship's wetted surface area.

The resulting forces due to water resistance can be separated into two categories, one attributed to fouling and another due to waves and eddies. Fouling of the ship's underwater surface area is caused when algae, barnacles and other objects become fixed to the ship's underwater surface. A resistance coefficient due to fouling, C_f is multiplied by the reference force to determine the resistance force due to fouling.

$$R_f = C_f K \quad (2.16)$$

The residual resistance (R_r) is composed of the resistance due to waves, which are created and pushed against the ship as it propels through the water, and eddies, or the swirling currents in the rear of the ship that are created as it moves forward. A residual coefficient if multiplied by the reference force to calculate the resistance force due to residual waves and eddies.

$$R_r = C_r K \quad (2.17)$$

The last major component of towing resistance is the resistance due to air. Overall, this resistance force is not nearly as great as the resistance due to water and accounts for only approximately 2% of the total resistance. Bernoulli's equation can again be used. However, since the affected area is based on air, the portion of the ship above the water is considered. The reference coefficient often used for the resistance due to air is equal to 90%. Now, the resistance force due to air is calculated as

$$R_a = 0.90 \left(\frac{1}{2} \right) \rho_{air} V^2 A_{air} \quad (2.18)$$

where ρ_{air} is the density of air and A_{air} is the cross sectional area of the vessel above water.

Now, from Equation 1.1, the total force is known considering thrust and towing resistance. The mass of the ship can then be found. Added to the ship's mass is a virtual or entrained mass that is attributed to the water that moves with the ship and through the propeller(s) [35]. This entrained mass can add approximately 8% to the ship's mass. Therefore, the total mass of the ship is determined by

$$m = m_s + m_e \quad (2.19)$$

where m_s is the mass of the ship and m_e is the entrained water mass.

Now, Equation 1.1 can be updated to calculate the velocity of the ship.

$$\dot{V} = \frac{1}{m} (T - R_t) \quad (2.20)$$

2.2.4 System Equations

The system equations that represent the propulsion plant can be found by using Equations 2.8 and 2.20. First, to solve for the shaft speed, the unbalanced torque as shown in Equation 2.1 is found by substituting Q_d , Q_f and Q_p from Equations 2.2, 2.3 and 2.5, respectively to yield

$$Q_u = K_g \sum_{i=1}^2 Q_{e_i} - f(J, pr) \rho \omega^2 d^5 - K_f I \omega^2 \quad (2.21)$$

Now, Equation 2.8 may be updated as follows

$$\dot{\omega} = \frac{K_g \sum_{i=1}^2 Q_{e_i} - f(J, pr) \rho \omega^2 d^5 - K_f I \omega^2}{I} \quad (2.22)$$

To solve for the ship speed, the components of thrust and total resistance from Equations 2.9 through 2.19 are substituted into Equation 2.20 to yield

$$\dot{V} = \frac{f_t(J, pr) \rho \omega^2 d^4 (1 - 0.1 \frac{f_t(J, pr) \rho \omega^2 d^4}{T_{pmax}}) - (C_f + C_r) \frac{1}{2} \rho V^2 A_s - 0.90 \left(\frac{1}{2}\right) \rho_{air} V^2 A_{air}}{m_s + m_e} \quad (2.23)$$

Because of the complex, non-linear nature of Equations 2.22 and 2.23, assumptions were made to simplify these equations. These assumptions include the following:

1. Only forward motion of the ship is considered ($V \geq 0$).
2. The propeller pitch is always fixed ($pr = 100\%$).
3. The shaft resistance force Q_f is negligible ($Q_f \approx 0$).
4. The wake coefficient is 0.03 ($w = 0.03$).
5. The Propeller Torque Coefficient (listed as $f_q(J, pr)$ in Eqn. 2.22) can be estimated with a linear equation ($K_q = -0.1238 \frac{0.97V}{\omega d} + 0.1862$).
6. The Propeller Thrust Coefficient (listed as $f_t(J, pr)$ in Eqn. 2.23) can be estimated with a linear equation ($K_t = -0.6361 \frac{0.97V}{\omega d} + 0.9348$).
7. The Engine Thrust can be estimated with a linear equation ($Q_e = 9388tic - 198.405$).
8. The frictional losses due to air are negligible ($\rho_{air} \approx 0$).

Now, based on these assumptions, Equations 2.22 and 2.23 can be simplified to yield

$$\dot{\omega} = \frac{K_g \sum_{i=1}^2 (9388tic - 198.405)_i - (-0.1238 \frac{0.97V}{\omega d} + 0.1862) \rho \omega^2 d^5}{I} \quad (2.24)$$

$$\dot{V} = \frac{\sum_{j=1}^2 \left(\left(-0.6361 \frac{0.97V}{\omega d} + 0.9348 \right) \rho \omega^2 d^4 \left(1 - 0.1 \frac{\sum_{j=1}^2 \left(\left(-0.6361 \frac{0.97V}{\omega d} + 0.9348 \right) \rho \omega^2 d^4 \right)_j}{T_{pmax}} \right)_j \right) - (C_f + C_r) \frac{1}{2} \rho V^2 A_s}{m_s + m_e} \quad (2.25)$$

From the updated system equations of Equations 2.24 and 2.25, the primary continuous input is the throttle input command, tic , which is essentially equivalent to how much the gas pedal is pressed. Other discrete inputs are i and j , which represent the number of engines powering the shaft and the number of powered shafts, respectively. Additionally, the variables K_g , d , ρ , I , T_{pmax} , C_f , C_r , A_s , m_s and m_e are all constant values.

Chapter 3: Hybrid Control

3.1 Hybrid Systems Background

Before discussing what was developed in the current work, a brief background on hybrid systems is presented.

3.1.1 Classes of Hybrid Systems

Hybrid systems combine time-driven dynamics with event-driven dynamics and when studying these, two approaches are available. One approach is to extend the traditional event-driven models to include time-driven dynamics. An example of this approach is to extend the time state automata and Petri net models to allow state transition times to be determined by time-driven dynamics. Conversely, the other is to extend the traditional time-driven models to include event-driven dynamics. An example of this is to extend time-driven models with multiple time scales. A fast scale can be used for the time-driven dynamics and a slow scale for the event-driven dynamics. Singular perturbation theory can be used where discrete events are injected as jump processes in the time-driven model [36].

Another class of hybrid system is called threshold-event-driven hybrid system. In this class of hybrid system, the changes in the discrete state of the system can only occur when the continuous state variables reach specified thresholds. The continuous dynamics are determined by a discrete condition that depends on the current discrete state of the system [27]. An example of this type of system is displayed in Figure 1.1, where each of the discrete states contain its own set of time-dependent continuous equations. However, the threshold parameters of the arcs are not included in the figure.

3.1.2 Hybrid Petri Net Methods

The time constraints used in Time Petri nets are considered as enabling conditions. This approach can be extended to include the dynamic response of differential equations for continuous systems,

or difference equations for discrete-time systems. In a hybrid Petri net, continuous places are added to the DES. In [37], a hybrid Petri net is defined as a 7-tuple

$$HPN = (P, T, D^+, D^-, h, \tau, \mu(0)) \quad (3.1)$$

where P, T, D^+, D^-, τ and $\mu(0)$ are as follows:

\mathbf{P} is the finite set of n places.

\mathbf{T} is the finite set of m transitions.

D^+ is the integer matrix that represents the weight of arcs from transitions to places.

$$D^+ \in \mathbb{Z}^{n \times m}$$

D^- is the integer matrix that represents the weight of arcs from places to transitions.

$$D^- \in \mathbb{Z}^{n \times m}$$

\mathbf{h} indicates if a place is continuous (C) or discrete (D).

τ represents the time associated to the firing of a transition, where the time delay is d_j

for a discrete transition T_j and the maximal firing speed for a continuous transition

$$T_j \text{ is } V_j = 1/d_j.$$

$\mu(0)$ is the initial marking of the Petri net.

Another type of hybrid Petri net is a Global Petri net (GPN). Places in a GPN correspond to system parameters, variables or states. Transitions represent processes or the operation of a component. The marking can include any real number, positive or negative. There are two types of arcs. One type includes event-driven, or asynchronous arcs that have weights assigned by the weight matrices D^+ and D^- . The other type of arcs are the synchronous, time dependent, arcs that are represented by A and B matrices. (If these matrices contain all zeros, the system has no time dependent effects and the GPN is reduced to a simple Petri net). A GPN is defined as:

$$GPN = (P, T, D^+, D^-, h, \tau, \mu(0), A, B) \quad (3.2)$$

where each of the variables are the same as in equation 3.1 and the following has been added:

A is an $n \times m$ matrix of synchronous arc weights drawn from continuous places to transitions.

B is an $m \times n$ matrix of synchronous arc weights drawn from transitions to continuous places.

Hybrid Petri nets, including GPNs, have proven to be effective in fault detection analysis [31], [37], [38], [39]. More detailed information on discrete, continuous and hybrid petri nets is discussed in [40] and [41].

3.1.3 Hybrid System Control

A hybrid control system controls both the continuous and discrete dynamics of a given system. Controls are applied to govern the continuous dynamics of a given discrete state. Similarly, controls are applied to set a given discrete state. As mentioned above in Section 3.1.1, when certain thresholds are exceeded a transition to a different discrete state may result. From a controlling aspect, a user-initiated or automated command can initiate a discrete state transition.

In [42], an automata is extended to a hybrid system with continuous controls in its discrete places. A controller for this hybrid dynamic system is defined as

$$H_c = [Q, \Sigma, \mathbf{A}, \mathbf{V}, \mathbf{G}, \mathbf{C}, \mathbf{F}] \quad (3.3)$$

where:

Q is the set of discrete states.

$\Sigma = \{\Sigma_q\}_{q \in Q}$ is the collection of controlled dynamic systems, where $\Sigma_q = [X_q, \Gamma_q, f_q, U_q]$ represent the controlled dynamic system and X_q are the continuous state spaces, f_q are the function of the continuous dynamics system equations and U_q is the set of continuous controls.

$\mathbf{A} = \{A_q\}_{q \in Q}$, $A_q \subset X_q$ for each $q \in Q$ is the collection of autonomous jump sets.

$\mathbf{V} = \{V_q\}_{q \in Q}$ is the collection of the transition control set V_q . These represent the discrete dynamics and controls.

$\mathbf{G} = \{G_q\}_{q \in Q}$, where $G_q : A_q \times V_q \rightarrow S$ is the autonomous jump transition map.

$\mathbf{C} = \{C_q\}_{q \in Q}$, $C_q \subset X_q$, is the collection of controlled jump sets.

$\mathbf{F} = \{F_q\}_{q \in Q}$, where $F_q : C_q \rightarrow 2^S$ is the collection of controlled jump destination maps.

Therefore, $S = \bigcup_{q \in Q} X_q \times \{q\}$ is the hybrid state space of H . If the sets U_q and \mathbf{V} , \mathbf{C} , and \mathbf{F} are empty, this results in the dynamics of an uncontrolled hybrid system.

When considering a controller for a hybrid system, one of the control objectives may be to minimize the cost of a given function over a specified amount of time or during a given operation where the system could transition over multiple states. A cost function (or performance index) may be established. The equations can account for the continuous dynamics and would require extending to account for the discrete dynamics of a system. Considering Equation 3.3, one approach is to determine Σ and \mathbf{V} in such a way that it minimizes a desired key parameter of H_c .

$$J = \min_{\Sigma(t), \mathbf{V}(t)} \int_0^T H_c dt \quad (3.4)$$

A simple approach for determining the optimal configuration that complies with Equation 3.4 is to determine the optimal continuous settings and then determine the optimal discrete settings. This approach is further discussed in the following sections.

3.2 Propulsion Controller

Leveraging what was presented in the prior section, a hybrid propulsion controller can be implemented to control the propulsion system as discussed in Chapter 2. As displayed in Figure 3.1, a hybrid controller, which has inputs of a desired ship speed and certain plant feedback signals, can be used to control a ship's propulsion system. From a continuous control perspective, the controller would primarily be responsible for achieving the desired ship speed. From a discrete perspective, the

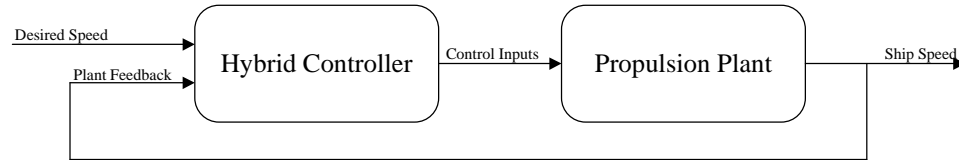


Figure 3.1: System Block Diagram

controller would be responsible for automatically aligning the propulsion system to ensure the desired ship speed can be achieved. Since this controller is used to automatically control the propulsion system, it is referenced as the Automated Propulsion Controller (APC) throughout this work.

3.2.1 Controller/Plant Interfaces

As mentioned above, the APC is responsible for aligning and controlling the plant to operate with its ideal settings. The component interfaces of the APC are displayed in Figure 3.2. In this figure,

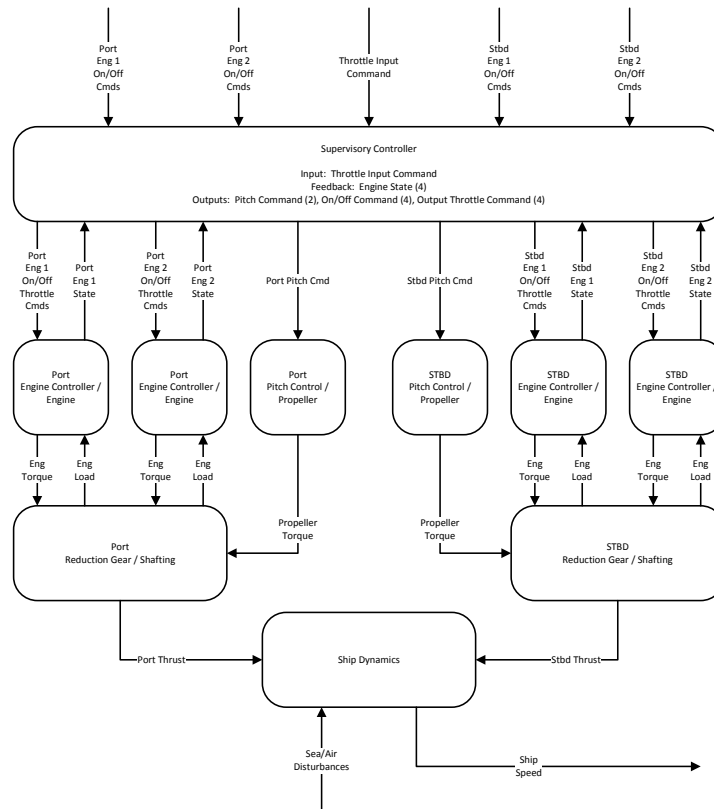


Figure 3.2: Controller Functional Diagram

the propulsion controller is displayed at the top. The next tier of objects includes each of the

controlled elements, or the main propulsion equipment. This equipment then interfaces with other ship components to determine the actual ship response. Each tier is further discussed below.

The main input to the propulsion controller is the throttle input command. This input command to the controller is a user specified command and can also be interpreted as the desired speed of the ship. Other inputs to the controller include feedback responses. The plant feedback that is monitored by the controller includes the ship speed and the state of each propulsion engines. Based on these inputs to the controller, the outputs are determined. The controller outputs are the pitch angle command of the shaft propeller, the on/off command to each of the engines and the output throttle command to each of the engines. (The throttle output command to the propulsion engines is equivalent to the “gas” of an automobile).

The propulsion equipment includes four propulsion engines (two on the port side of the ship and two on the starboard side of the ship) and two shaft propellers (one of the port side and another on the starboard side). The controllable pitch propeller receives the desired pitch command from the propulsion controller and adjusts the pitch angle of the propellers’ blades accordingly. As the pitch angle increases, the blades of the propeller take a deeper slice into the water. This, then, increases the propeller torque applied to the shafting and thrust applied to the ship.

The propulsion engines each have an engine controller, which receives the on/off and output throttle commands from the main propulsion controller. The engine controller also monitors other engine parameters to ensure that the engine operates in a safe manner. In the event that key parameters exceed their respective critical thresholds. The engine controller can shutdown the engine. To accurately control the propulsion plant, the state of the engine is integral to the propulsion controller’s function. The propulsion engines must respond to the load applied through the shafting. This load is a function of the propeller’s pitch angle. The engine must output a torque greater than the applied load to actually cause the shaft to rotate. As a greater torque is applied, the shaft spins faster, causing more thrust to be applied to the ship.

The ship responds to the applied torques of the propulsion engines, via the reduction gear, and the applied torque of the propeller connected to the shaft. The thrust from the port and starboard

shafts correlate to the speed of the ship. The ship speed can be adversely affected by the current sea state and other adverse disturbances or conditions. The actual ship speed is feed back to the propulsion controller so that adjustments can be made so that the desired ship speed can be achieved.

3.2.2 APC Implementation Approach

An approach for implementing the APC is displayed below in Figure 3.3, where the subscripts identify the mode of the ship; *ts* for trailshaft, *sp* for split plant and *fp* for full power. The desired

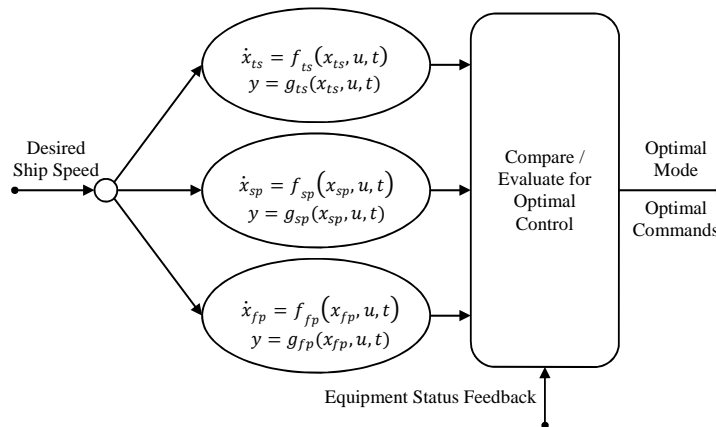


Figure 3.3: Evaluation of Continuous Portion of Hybrid Controller

speed is an input to each mode of the ship. As displayed, each mode has its own set of system equations. The output to these system equations are evaluated within discrete logic to determine if the ship is in its optimal configuration.

The hybrid controller is integrated in such a way that an input to the continuous logic is an output from the discrete logic. Similarly, an input to the discrete logic is an output from the continuous logic. A hybrid controller should include this type of integrated functionality to maximize its functionality. The control inputs to the propulsion plant in Figure 3.1 are broken down to include the control inputs and input commands in Figure 3.4. The control inputs are the outputs of the continuous control. These are essentially analog commands to the propulsion plant, which are used to speed up and slow down the four engines. The input commands are the output of the discrete control logic. These commands are used to start and stop the four engines. These commands are also used to start and stop other auxiliary functions of the propulsion plant.

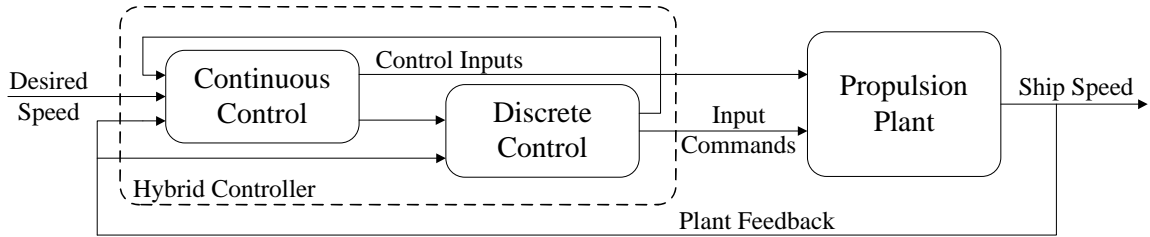


Figure 3.4: Hybrid Controller Block Diagram

The plant feedback is used to provide a status of plant equipment and confirm the optimal mode of the ship. If the optimal mode of the ship can not be realized because of failed plant equipment, the controller then selects the next best mode and its associated output commands to send to the propulsion plant. The interaction of both the discrete and continuous parts of the hybrid controller are critical to its operation in determining the ideal settings for the ship propulsion controller.

3.2.3 Continuous System Control

In an effort to implement the most effective continuous controller, various approaches were investigated using the linearized system equations from Section 2.2.4. These include implementing a Lagrange multiplier approach, implementing a linear quadratic regulator and implementing a tracking controller. A brief summary of these implementation approaches follows.

Lagrange Multiplier Approach

When using the Lagrange multiplier approach, the goal of the optimum control algorithm is to find

the control vector $\mathbf{u} = \begin{Bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{Bmatrix}$ which minimizes the cost functional or performance index,

$$J = \int_0^T f_o(\mathbf{x}, \mathbf{u}, t) dt \quad (3.5)$$

where, $\mathbf{x} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix}$ is the state vector, which represents key parameters that are used to describe the ship's propulsion system. The parameter t represents time and T is the terminal time or ending time. The variable f_o is used to describe an initial function that is characterized by the variables \mathbf{x} , \mathbf{u} , and t . The state variables x_i and the control variables u_i are related as

$$\frac{dx_i}{dt} = f_i(x_1, x_2, \dots, x_n; u_1, u_2, \dots, u_m, t), \quad i = 1, 2, \dots, n \quad (3.6)$$

or

$$\dot{x}_i = f_i(\mathbf{x}, \mathbf{u}, t), \quad i = 1, 2, \dots, n \quad (3.7)$$

but more specifically for the propulsion system,

$$\begin{Bmatrix} \dot{\omega}_i \\ \dot{V}_i \end{Bmatrix} = f_i(\mathbf{x}, \mathbf{u}, t), \quad i = 1, 2 \quad (3.8)$$

or for a linear system,

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \quad (3.9)$$

where \mathbf{A} is an n by n state matrix and \mathbf{B} is an n by m input matrix.

A Lagrange multiplier λ_i , also known as the adjoint variable for the i th constraint in Equation 3.7 is introduced. Additionally, an augmented functional J^* is introduced as

$$J^* = \int_0^T \left[f_o + \sum_{i=1}^n \lambda_i (f_i - \dot{x}_i) \right] dt \quad (3.10)$$

The Hamiltonian functional, H , is defined as

$$H = f_o + \sum_{i=1}^n \lambda_i f_i \quad (3.11)$$

such that

$$J^* = \int_0^T \left(H - \sum_{i=1}^n \lambda_i \dot{x}_i \right) dt \quad (3.12)$$

Since the integrand

$$F = H - \sum_{i=1}^n \lambda_i \dot{x}_i \quad (3.13)$$

depends on \mathbf{x} , \mathbf{u} , and t , there are $n + m$ dependent variables (\mathbf{x} with n and \mathbf{u} with m). The Euler - Lagrange equations now become

$$\frac{\partial F}{\partial x_i} - \frac{d}{dt} \left(\frac{\partial F}{\partial \dot{x}_i} \right) = 0, \quad i = 1, 2, \dots, n \quad (3.14)$$

$$\frac{\partial F}{\partial u_j} - \frac{d}{dt} \left(\frac{\partial F}{\partial \dot{u}_j} \right) = 0, \quad i = 1, 2, \dots, m \quad (3.15)$$

Using Equation 3.13, Equations 3.14 and 3.15 can be rewritten as

$$-\frac{\partial H}{\partial x_i} = \dot{\lambda}_i, \quad i = 1, 2, \dots, n \quad (3.16)$$

$$\frac{\partial H}{\partial u_j} = 0, \quad j = 1, 2, \dots, m \quad (3.17)$$

Equations 3.16 and 3.17 are known as adjoint equations, where Equation 3.16 represents the co-state variables and Equation 3.17 represents the optimality conditions.

The optimum solutions for \mathbf{x} , \mathbf{u} and $\boldsymbol{\lambda}$ can be obtained by solving the differential equations of Equation 3.7 and partial differential equations of Equations 3.16 and 3.17. These yield $2n + m$ equations with n state variables (x_i), n co-state variables (λ_i) and m control input variables (u_j) that represent the unknowns. If the initial conditions $x_i(0)$, where $i = 1, 2, \dots, n$ and the terminal conditions $x_j(T)$, where $j = 1, 2, \dots, l$, with $l < n$ are known, the terminal values of the remaining variables $x_j(T)$, $j = l + 1, l + 2, \dots, n$ are free. Hence the free end conditions must be used and

$$\lambda_j(T) = 0; \quad j = l + 1, l + 2, \dots, n \quad (3.18)$$

When expanded, the equations of Equation 3.18 represent the transversality conditions. [43]

Since an objective could be to minimize fuel consumption, a relationship between this and the system variables must be established. In the case of the gas turbine engine, throttle input command relates to fuel flow. Simulated data was collected to establish this relationship and is displayed in Figure 3.5. Also displayed in this figure is the curve fit of the data points. A 7th order polynomial

$$f(x) = p_7x^7 + p_6x^6 + p_5x^5 + p_4x^4 + p_3x^3 + p_2x^2 + p_1x + p_0 \quad (3.19)$$

was used to fit this data. The coefficients of the polynomial with 95% confidence bounds are listed in Table 3.1 and the goodness of fit is listed in Table 3.2.

The cost function can be represented by Equation 3.19 with the coefficients listed in Table 3.1. Applying this and the state equations of Equation 3.8, the Hamiltonian function of Equation 3.11 can be updated as

$$H = f_0 + \lambda_1\dot{\omega} + \lambda_2\dot{V} \quad (3.20)$$

The Hamiltonian partial equations of Equations 3.16 and 3.17 can be updated as

$$-\frac{\partial H}{\partial \omega} = \dot{\lambda}_1 \quad (3.21)$$

$$-\frac{\partial H}{\partial V} = \dot{\lambda}_2 \quad (3.22)$$

$$\frac{\partial H}{\partial pr} = 0 \quad (3.23)$$

$$\frac{\partial H}{\partial TIC} = 0 \quad (3.24)$$

The optimal solution can be obtained by solving Equations 3.8 through 3.24 for ω , V , pr , TIC , λ_1 and λ_2 . When these equations are expanded, they are rather complex. Therefore, a tool was required to determine the solution. MuPAD, a tool part of the Symbolic Math toolbox of MATLAB, was selected to generate analytical solutions. By determining analytical solutions as opposed to numerical solutions, the resulting solutions can be easily updated with applicable coefficients to reflect a different configuration or ship class. This increases the option for a common and flexible solution. However, MuPAD failed to converge and generate a common solution. Therefore, other

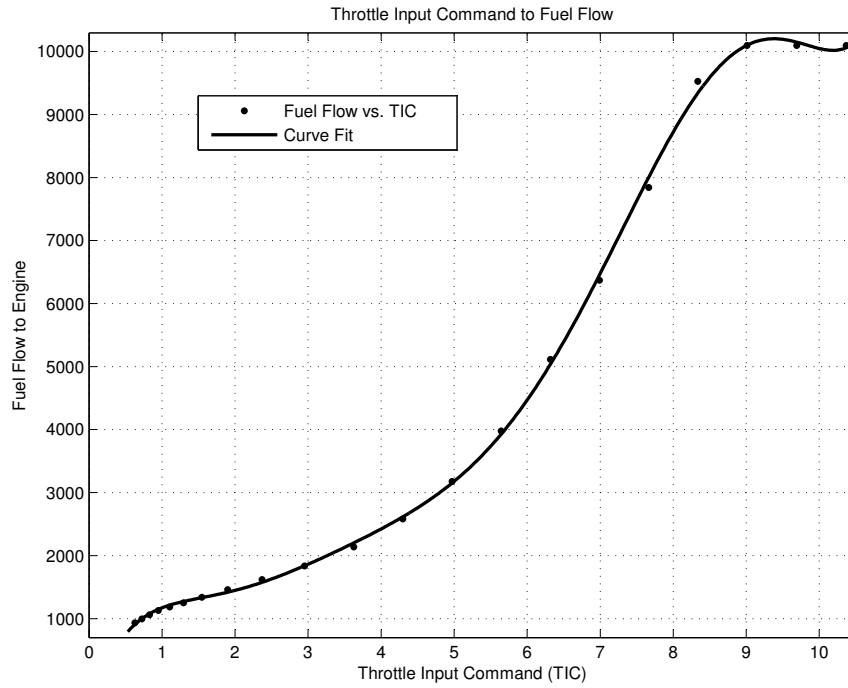


Figure 3.5: Correlation of Throttle Input Command to Fuel Flow

Table 3.1: Polynomial Coefficients for Throttle Input Command to Fuel Flow

Coefficient	Value	95% Bounds
p_7	0.1221	(0.08978, 0.1544)
p_6	-4.37	(-5.595, -3.145)
p_5	61.33	(42.66, 80)
p_4	-434.6	(-580.7, -288.4)
p_3	1682	(1059, 2305)
p_2	-3482	(-4892, -2071)
p_1	3825	(2301, 5348)
p_0	-475.6	(-1061, 110)

Table 3.2: Goodness of Fit for Throttle Input Command to Fuel Flow

Method	Value
SSE	1.847e+005
R-square	0.9996
RMSE	73.7

methods were required to be explored to determine an optimal solution.

Linear Quadratic Regulator Approach

Another optimal control theory method is the Linear Quadratic Regulator (LQR) approach. Given a system described by Equation 3.9 with given initial conditions ($x(t_0) = x_0$), a control objective is to establish a control function ($u(t)$) defined on $[t_0, T]$, which can be a function of the state $x(t)$, such that the state $x(t)$ is driven to a small neighborhood of origin at time T . This is defined as the regulator problem. [44]

For most physical systems, the controlling vector will have some physical constraints. For example,

$$\int_{t_0}^T \|u\| dt, \int_{t_0}^T \|u\|^2 dt, \sup_{t \in [t_0, T]} \|u\| \quad (3.25)$$

Additionally, there may be some constraints on the transient response of the state vector $x(t)$. These may include something similar to the controlling vector constraints.

$$\int_{t_0}^T \|x\| dt, \int_{t_0}^T \|x\|^2 dt, \sup_{t \in [t_0, T]} \|x\| \quad (3.26)$$

It may be more appropriate to apply weights to certain elements of either the control vector or the state vector constraints. These weighted constraints are applied by weighting functions W_u and W_x for the control and state vectors, respectively. Equations 3.25 and 3.26 are updated as

$$\int_{t_0}^T \|W_u u\| dt, \int_{t_0}^T \|W_u u\|^2 dt, \sup_{t \in [t_0, T]} \|W_u u\| \quad (3.27)$$

$$\int_{t_0}^T \|W_x x\| dt, \int_{t_0}^T \|W_x x\|^2 dt, \sup_{t \in [t_0, T]} \|W_x x\| \quad (3.28)$$

An optimal control problem is defined with the assumptions that terminal time T approaches infinity ($T \rightarrow \infty$) and $t_0 = 0$. With these assumptions $[t_0, T] \rightarrow [0, \infty)$. The cost function, or

performance index, to be minimized is defined as

$$\min_{u(t)} \int_0^{\infty} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix}^T \begin{bmatrix} \mathbf{Q} & \mathbf{S} \\ \mathbf{S}^T & \mathbf{R} \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} dt \quad (3.29)$$

If the assumption is made that $S = 0$, Equation 3.29 becomes

$$\min_{u(t)} \int_0^{\infty} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix}^T \begin{bmatrix} \mathbf{Q} & 0 \\ 0 & \mathbf{R} \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} dt \quad (3.30)$$

Equation 3.30 can be rearranged as

$$\min_{u(t)} \int_0^{\infty} (x^T(t)\mathbf{Q}x(t) + u^T(t)\mathbf{R}u(t)) dt \quad (3.31)$$

where $\mathbf{Q}^T = \mathbf{Q} \geq 0$, $\mathbf{R}^T = \mathbf{R} > 0$ and $x(0) = x_0$.

Assuming \mathbf{R} is equal to the identity matrix, \mathbf{I} , the integrand of Equation 3.31 is equivalent to

$$x^T\mathbf{Q}x + u^T u = \begin{bmatrix} x^T & u^T \end{bmatrix} \begin{bmatrix} \mathbf{Q} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \quad (3.32)$$

Additionally, since

$$\begin{bmatrix} \mathbf{Q} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \geq 0 \quad (3.33)$$

This can be factored as

$$\begin{bmatrix} \mathbf{Q} & 0 \\ 0 & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{C}^T \\ \mathbf{D}^T \end{bmatrix} \begin{bmatrix} \mathbf{C} & \mathbf{D} \end{bmatrix} \quad (3.34)$$

The performance index is updated and arranged into the \mathcal{L}_2 or quadratic format similar to the middle portion of Equations 3.27 and 3.28 [45]

$$\min_{u(t)} \int_0^{\infty} x^T\mathbf{Q}x + u^T u dt = \min_{u(t)} \int_0^{\infty} \|\mathbf{C}x + \mathbf{D}u\|_2^2 dt \quad (3.35)$$

The LQR problem is defined as the minimization of

$$\min_{u(t) \in \mathcal{L}_2[0, \infty)} \| \mathbf{C}x + \mathbf{D}u \|_2^2 \quad (3.36)$$

for

$$\dot{x} = \mathbf{A}x + \mathbf{B}u \quad x(0) = x_0 \quad (3.37)$$

which also implies from Equation 3.34 that $\mathbf{C}^T \mathbf{D} = 0$, $\mathbf{D}^T \mathbf{C} = 0$ and $\mathbf{D}^T \mathbf{D} = \mathbf{I}$.

When determining the state space equations for the LQR problem, it is key to make the assumption that (\mathbf{A}, \mathbf{B}) is stabilizable. This assumption ensures that there exist a state feedback control element that makes the system stable. Another assumption is that (\mathbf{C}, \mathbf{A}) is detectable and that the initial state can be determined given the history of the inputs and outputs over a specified time range. With these assumptions the Hamiltonian matrix is defined as

$$H = \begin{bmatrix} \mathbf{A} & -\mathbf{B}\mathbf{B}^T \\ -\mathbf{C}^T \mathbf{C} & -\mathbf{A}^T \end{bmatrix} \in \text{dom}(\text{Ric}) \quad (3.38)$$

for the Algebraic Riccati Equation (ARE)

$$\mathbf{A}^T \mathbf{X} + \mathbf{X} \mathbf{A} + \mathbf{C}^T \mathbf{C} - \mathbf{X} \mathbf{B} \mathbf{B}^T \mathbf{X} = 0 \quad (3.39)$$

where $\mathbf{X} = \text{Ric}(H) \geq 0$.

Considering the linear quadratic regulator problem of Equations 3.36 and 3.37 and the corresponding algebraic Riccati equation of Equation 3.39, which has a unique solution \mathbf{X} , the optimal input is defined as [46]

$$u(t) = -\mathbf{B}^T \mathbf{X} x(t) = \mathbf{F} x(t) \quad (3.40)$$

where the optimal gain state feedback matrix is $\mathbf{F} = -\mathbf{B}^T \mathbf{X}$ and is independent of the initial

conditions x_0 . The stable optimal state trajectory is then

$$\dot{x}(t) = \mathbf{A} - \mathbf{B}\mathbf{B}^T\mathbf{X}x(t) \quad (3.41)$$

For linear systems that are stabilizable and detectable, the optimal system inputs can be found using the LQR approach. When these optimum inputs are applied, the optimal state trajectory for the linear system can be realized.

Using the state space equations and Equations 3.39 and 3.40, the solution to the Algebraic Riccati Equation (ARE) can be found and multiplied by the opposite of the transpose of the input matrix ($-\mathbf{B}^T$) to yield the optimal gain state feedback matrix \mathbf{F} .

As opposed to calculating these by hand, tools are available to solve the ARE and perform matrix multiplication. This calculation was performed in MATLAB by using the `lqr` command. The inputs of this command are the matrices \mathbf{A} , \mathbf{B} , \mathbf{Q} and \mathbf{R} . Recalling from Equation 3.34,

$$\begin{aligned} \mathbf{Q} &= \mathbf{C}^T\mathbf{C} \\ &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \end{aligned} \quad (3.42)$$

and the assumption that \mathbf{R} is equal to the identity matrix \mathbf{I} ,

$$\mathbf{R} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (3.43)$$

With these inputs, the `lqr` command yields the optimal feedback matrix \mathbf{F} , solution to the ARE \mathbf{X} and closed loop Eigenvalues of $\mathbf{A} - \mathbf{B}\mathbf{F}$. These output values for the three modes of operation are listed in Table 3.3.

From Equation 3.40, the optimal input is realized when the optimal feedback matrix is multiplied

Table 3.3: LQR Output of Linearized Model by Ship Mode

Mode	Optimal Feedback Matrix \mathbf{F}	ARE Solution \mathbf{X}	Eigenvalues
Trailshaft	$\begin{bmatrix} -1.6120 \times 10^6 & -1.6644 \times 10^4 \\ -5.8549 \times 10^4 & -592.1954 \end{bmatrix}$	$\begin{bmatrix} 1.3014 \times 10^{12} & 1.3155 \times 10^{10} \\ 1.3155 \times 10^{10} & 1.3307 \times 10^8 \end{bmatrix}$	$\begin{bmatrix} -0.9998 \\ -0.7161 \end{bmatrix}$
Split Plant	$\begin{bmatrix} -4.0718 \times 10^6 & -9.2670 \times 10^4 \\ -8.5273 \times 10^5 & -1.9359 \times 10^4 \end{bmatrix}$	$\begin{bmatrix} 8.6566 \times 10^{12} & 1.9652 \times 10^{11} \\ 1.9652 \times 10^{11} & 4.4614 \times 10^9 \end{bmatrix}$	$\begin{bmatrix} -0.9997 \\ -0.6677 \end{bmatrix}$
Full Power	$\begin{bmatrix} -4.2490 \times 10^6 & -9.2534 \times 10^4 \\ -2.2478 \times 10^6 & -4.8901 \times 10^4 \end{bmatrix}$	$\begin{bmatrix} 1.1559 \times 10^{13} & 2.5147 \times 10^{11} \\ 2.5147 \times 10^{11} & 5.4706 \times 10^9 \end{bmatrix}$	$\begin{bmatrix} -0.9995 \\ -0.5436 \end{bmatrix}$

by the state vector. When realistic values of that state vector are multiplied by \mathbf{F} as listed in Table 3.3, the resulting input values were well outside of the admissible range of the inputs. This is because the values of \mathbf{F} were extremely large. These values were large because the values within the input matrix \mathbf{B} were quite small. These small values are a result of the system having a limited level of controllability.

Tracking Control Approach

Another approach investigated while implementing the continuous control portion of the hybrid controller was using a tracking controller as discussed in [47] and [48]. The propulsion system can be represented using the state space equations of Equation 3.44,

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}_1\mathbf{w}_1(t) + \mathbf{B}_2\mathbf{u}(t) \quad (3.44)$$

$$\mathbf{y}(t) = \mathbf{C}_2\mathbf{x}(t) + v(t) \quad (3.45)$$

where w_1 and v represent noise with the following covariances

$$\begin{aligned} E(\mathbf{w}_1\mathbf{w}_1^T) &= \mathbf{I} \\ E(\mathbf{v}\mathbf{v}^T) &= \mathbf{I} \\ E(\mathbf{w}_1\mathbf{v}^T) &= \mathbf{0} \end{aligned} \quad (3.46)$$

Additionally, when the ship is commanded to traverse at a different rate of speed, the commanded velocity can be considered as a step change, either increasing or decreasing from one speed to another.

The following equation can be used to represent this change in ship speed

$$\dot{V}_r(t) = ZV_r(t) + w_r(t) \quad (3.47)$$

where $w_r(t)$ represents an impulse function and Z is chosen to be 0 if $V_r(t)$ is a step function.

A control variable $z(t)$ as defined in Equation 3.48 is added to Equation 3.44 to represent the generalized propulsion plant.

$$z(t) = \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} = \begin{bmatrix} C_{1u} \\ 0 \end{bmatrix} x(t) + \begin{bmatrix} D_{11u} \\ 0 \end{bmatrix} V_r(t) + \begin{bmatrix} 0 \\ D_{12d} \end{bmatrix} u(t) \quad (3.48)$$

Now, the objective is to find a controller K such that:

1. The closed-loop system is internally stable
2. Both tracking error and alignment error are zero at steady state (i.e. $\lim_{t \rightarrow \infty} z_1(t) = 0$).
3. The performance index $J = \lim_{T \rightarrow \infty} \frac{1}{T} E \left[\int_0^T z^T(t) z(t) dt \right]$, where $E[X]$ is the expected value of X , is minimized.

A multivariable controller was implemented, which combined regulator and H_2 theory. Regulatory theory is used to ensure that the second controller objective of no (zero) tracking and alignment error at steady state. H_2 theory is used to ensure controller objectives of 1 and 3 are satisfied.

Steady-state regulation of an internally stable closed loop system is achieved if W and U are chosen such that Equations 3.49 and 3.50 are satisfied.

$$AW + B_2U - WZ = 0 \quad (3.49)$$

$$C_{1u}W + D_{11u} = 0 \quad (3.50)$$

A condition of existence for the controller is that the system (A, B_2, C_2) is stabilizable and detectable.

Using H_2 theory, an observer is constructed assuming (A, C_2) is detectable

$$\hat{\dot{x}}(t) = (A - LC_2)\hat{x}(t) + B_2u(t) + Ly(t) \quad (3.51)$$

where the observer gain, L , is

$$L = YC_2^T \quad (3.52)$$

and Y is the positive semi-definite stabilizing solution of the algebraic Riccati equation,

$$AY + YA^T - YC_2^T C_2 Y + B_1 B_1^T = 0 \quad (3.53)$$

To determine the state feedback gain, F , we define the following

$$\begin{aligned} Q &= C_{1u}^T C_{1u} \\ R &= D_{12d}^T D_{12d} \end{aligned} \quad (3.54)$$

From this, the state feedback gain can be found as

$$F = -R^{-1} B_2^T X \quad (3.55)$$

where X is the positive semi-definite stabilizing solution of the algebraic Riccati equation,

$$A^T X + XA - X B_2 R^{-1} B_2^T X + Q = 0 \quad (3.56)$$

The desired (reference) speed of the ship V_r can be measured and used as feedback for the

controller. The output of the controller can be defined as

$$y_c = \begin{bmatrix} y \\ V_r \end{bmatrix} \quad (3.57)$$

Updating the generalized plant state space equations as defined by Equations 3.44 and 3.48 with the updated controller output yields

$$\begin{aligned} \dot{x}_k(t) &= A_k x_k(t) + B_k y_c(t) \\ u(t) &= C_k x_k(t) + D_k y_c(t) \end{aligned} \quad (3.58)$$

where

$$\begin{aligned} A_k &= A - LC_2 + B_2 F \\ B_k &= \begin{bmatrix} L & 0 \end{bmatrix} \\ C_k &= F \\ D_k &= \begin{bmatrix} 0 & U - FW \end{bmatrix} \end{aligned} \quad (3.59)$$

To implement this approach, the linearized system equations in the state space form were used in a MATLAB script to find a controller described by Equation 3.58, where $x = \begin{bmatrix} \omega \\ V \end{bmatrix}$ and $u = tic$. This aligns with the assumptions made in Section 2.2.4 when simplifying the system equations. However, with the assumption that the control input is one dimensional and only a function of tic , steady state regulation could not be achieved because it was not possible to chose W and U such that both Equations 3.49 and 3.50 were satisfied.

Simplified Approach

After these previous attempts in developing the continuous control portion of the hybrid controller failed to produce a viable option, a more general continuous control approach was taken. Pursuant

to this simplified approach, a Proportional Integral (PI) method is employed. This method was used because PI algorithm results produce a relatively small steady state error. A Proportional Integral Derivative (PID) was not selected because the derivative terms can sometimes lead to additional fluctuations at steady state. Additionally, other controllers used within the propulsion plant, like the gas turbine engine controllers, use a PI control algorithm as it has proven to be an effective controller for propulsion control.

From Figure 3.6, it can be observed that the continuous control logic is essentially a PI control loop with the primary input being the error between the desired speed and the actual speed. Additionally, this error is multiplied by a Boolean (0 or 1) if the throttle command is enabled for

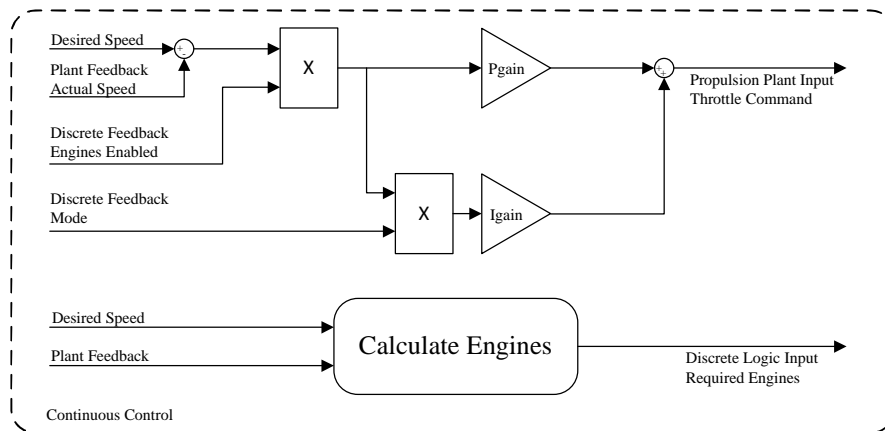


Figure 3.6: Continuous Controller Block Diagram

at least one of the engines. There is no need to control if there are no engines ON or ONLINE to respond to the control command. This resultant is multiplied by the mode and used by the integral gain. The value of mode will be 0 if no engines are ONLINE; 1 if the propulsion plant is in Trailshaft mode; 2 if the propulsion plant is in Split Plant mode; or 4 if the propulsion plant is in Full Power Mode. This is done to increase the speed of the response when more engines are available.

The outputs of the continuous logic were the throttle command, pitch command (not shown in the figure) and required engines. The throttle command was the result of the PI control algorithm. This output was clamped, or bounded, to be within the operational minimum and maximum settings. Based on existing data, the optimal pitch was often found to be in the neighborhood of 100%. Therefore, for the purposes of assigning a designated pitch command, this value was set to 100% for

driving shafts during this research effort (reference assumptions in Section 2.2.4).

Even though, the required engines were calculated in the continuous portion of the controller, discrete logic was also used in the determination of the number of engines required. If the difference between the desired speed and the actual speed exceeded a specified value and the actual speed of the ship was not increasing and these conditions were true for a specified amount of time, additional power (more engines) were required. If, for a specified amount of time, the actual speed was greater than the desired speed and the minimum throttle was commanded, less power (fewer engines) were required. If the ship's mode (actual number of engines online) did not align with the required number of engines after a user specified amount of time, it was assumed that the mode, which was calculated in the discrete logic portion of the controller, could not be achieved. The required number of engines was then updated to the same as the current mode. (Actually, this block is somewhat of a hybrid in itself, as it has both continuous and discrete functionality).

3.2.4 Controller Discrete Logic

The propulsion controller is responsible for controlling the operating state and alignment of the plant equipment, in particular, the alignment of the propulsion engines. Figure 3.7 shows a Petri net, which represents the propulsion plant and its allowable transitions as it reconfigures. The plant states include the following as displayed in the figure:

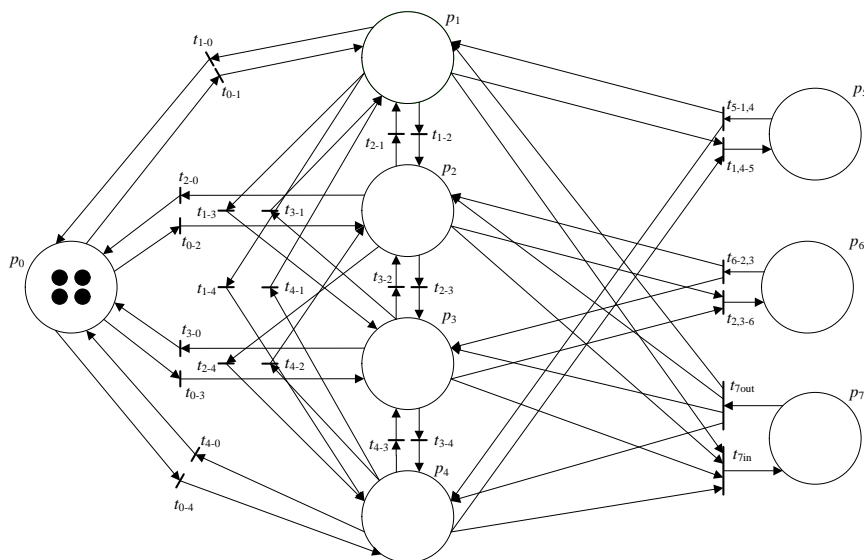


Figure 3.7: Discrete Logic of Controller

p_0 represents the off state of each of the propulsion engines. This is considered the initial condition of the controller. Four tokens, one for each of the engines, are displayed.

p_1 represents the state of the propulsion plant when Port Engine 1 is operating in trailshaft mode and all other engines are OFF.

p_2 represents the state of the propulsion plant when Port Engine 2 is operating in trailshaft mode and all other engines are OFF.

p_3 represents the state of the propulsion plant when Starboard Engine 1 is operating in trailshaft mode and all other engines are OFF.

p_4 represents the state of the propulsion plant when Starboard Engine 2 is operating in trailshaft mode and all other engines are OFF.

p_5 represents the state of the propulsion plant when the outboard engines (Port Engine 1 and Starboard Engine 2) are operating in split plant mode and the inboard engines (Port Engine 2 and Starboard Engine 1) are OFF.

p_6 represents the state of the propulsion plant when the inboard engines are operating in split plant mode and the outboard engines are OFF.

p_7 represents the state of the propulsion plant when it is in the full power configuration with each of the propulsion engines ONLINE.

The transitions included in Figure 3.7 represent different functions. The following transitions represent on/off transitions of the propulsion engines. Transitions t_{0-1} , t_{0-2} , t_{0-3} and t_{0-4} represent the OFF to ON transitions of the engines. Transitions t_{1-0} , t_{2-0} , t_{3-0} and t_{4-0} represent the ON to OFF transition of the engines.

The following transitions represent a shift in the trailshaft mode where one engine comes ONLINE to replace another. In the Petri net, one token will transition between p_1 , p_2 , p_3 or p_4 . Even though it is not represented in the Petri net, it is assumed, as stated in the place definitions above, that the off-going engine will shutdown even though t_{1-0} , t_{2-0} , t_{3-0} or t_{4-0} does not fire. The trailshaft shift transitions include t_{1-2} , t_{1-3} , t_{1-4} , t_{2-1} , t_{2-3} , t_{2-4} , t_{3-1} , t_{3-2} , t_{3-4} , t_{4-1} , t_{4-2} and t_{4-3} .

The following transitions represent split plant transitions. Transitions $t_{1,4-5}$ and $t_{2,3-6}$ enter the ship into a split plant configuration. For these transitions to fire, two trailshaft mode (single engine) states must each contain a token. This requirement, somewhat, contradicts the definition of these places. Therefore, it is assumed that one token will merely transition through the state that is not currently active and marked with a token. For example, if p_1 is active (Port Engine 1 is ONLINE) and the plant is required to transition to a split plant configuration, Starboard Engine 2 will start and transition $t_{1,4-5}$ will immediately fire causing the plant to enter the output engine split plant configuration. When transition $t_{5-1,4}$ or $t_{6-2,3}$ fires, the configuration of the plant will go from the split plant configuration to the trailshaft configuration. Similarly, one token will merely transition through the trailshaft mode state to the OFF state, leaving only one of the trailshaft mode states active.

The following transitions represent full power transitions. Transition t_{7in} aligns the plant in a full power configuration and t_{7out} transitions from a full power state. Similar to the split plant transitions, it is assumed that tokens will merely transition through the split plant states to enable the full power transition t_{7in} . Tokens will also transition through the split plant states when t_{7out} fires and the full power state is no longer active. For example, if the current active state is p_5 and the ship is in the split plant configuration, but the full power configuration is required, transitions $t_{5-1,4}$, t_{0-2} and t_{0-3} must fire to provide each of the engines a token to enable transition t_{7in} . If the plant is later required to transition from the full power configuration to trailshaft mode where Port Engine 2 is identified as the optimal engine, transition t_{7out} must fire, immediately followed by transitions t_{1-0} , t_{3-0} and t_{4-0} , leaving p_2 as the only active state marked with a single token.

In Figure 3.7, the active state is always marked with a single token. This includes places p_5 , p_6 and p_7 . The enabling transition for the split plant states require 2 tokens to fire, while the full power transition requires 4 tokens. When the disabling transitions fire, only 1 token is required for the transition to fire. However, the associated split plant tokens generate 2 tokens and the full power transition generates 4. The incidence matrix of this Petri net is displayed in Equation 3.60. (The

eight rows represent the 8 places and the twenty-six columns represent the 26 transitions).

$$A = \begin{pmatrix} -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 1 & -1 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \dots & & & & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 \\ 0 & -1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & 1 \\ 0 & 1 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & -1 & 1 \\ \dots & & & & & & & & & & & & & \\ -1 & 0 & 0 & 1 & -1 & 1 & -1 & -1 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \quad (3.60)$$

The discrete control logic for the Automated Propulsion Controller was implemented in accordance with the logic flow outlined in the Petri net displayed in Figure 3.7. However, instead of implementing this logic flow with Petri nets, a finite-state machine approach was utilized. The Wiley Electrical and Electronics Engineering Dictionary defines a finite-state machine or a finite state automaton as [49]:

A machine which can be completely described by a finite set of defined states. Such a machine must be in one of these states at any given moment, and there is a set of conditions which determine when it moves from one state to another.

Automaton and Petri nets are considered to be state machines because the discrete events or system conditions drive the system to a given state. Theoretically, these methods can be applied when studying Discrete Event Systems. However, in practice, one may desire to use a tool that is readily available to analyze Discrete Event Systems. One example of such a tool is Stateflow. Stateflow is an event-based modeling tool developed by The MathWorks, Inc. This tool is integrated into the MATLAB environment via Simulink models and used to evaluate the execution of discrete event logic.

The discrete control logic block diagram as implemented in the controller is displayed in Figure 3.8 and aligns with the process flow included in Figure 3.9. The primary function of the discrete control logic is to align the propulsion plant in its optimum configuration based on the required engines determined within the continuous control logic. To determine, the optimum configuration, the plant feedback is used to determine the actual state of the propulsion plant. The state of the propulsion plant aligns to the mode; Off, Trailshaft, Split Plant or Full Power.

From Figure 3.9, the discrete logic receives the required mode from the continuous portion of the controller. No action is required if the current mode is the same as the required mode. However, the discrete logic attempts to realign the propulsion plant to a different state if a change is required. When the propulsion plant is required to reconfigure, user specified engine parameters are evaluated for each of the engines. Based on the value for each of the associated engines, a priority is established. The priority is then used to determine which engine(s) should be started if more thrust is required or which engine(s) should be shutdown if less thrust is required. When transitioning from one state to another engines are either powered on and brought online to provide additional power for the propulsion plant to generate ship thrust or the engine is shutdown as the current power level exceeds the required power level. The discrete control logic is used to initiate and monitor these state transitions.

If the required mode is successfully achieved, confirmation is sent back to the continuous portion of the controller. However, if the mode is not successfully achieved, the required mode is updated. It is assumed that engines will always shutdown. Therefore, this update is primarily applicable

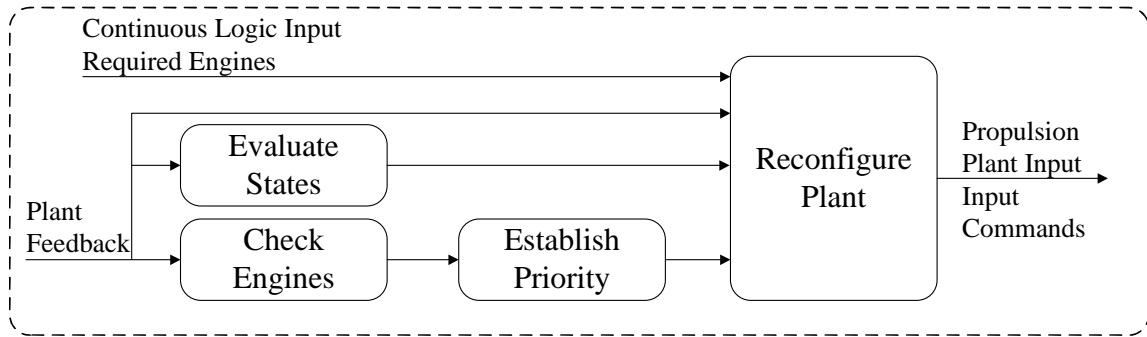


Figure 3.8: Discrete Controller Block Diagram

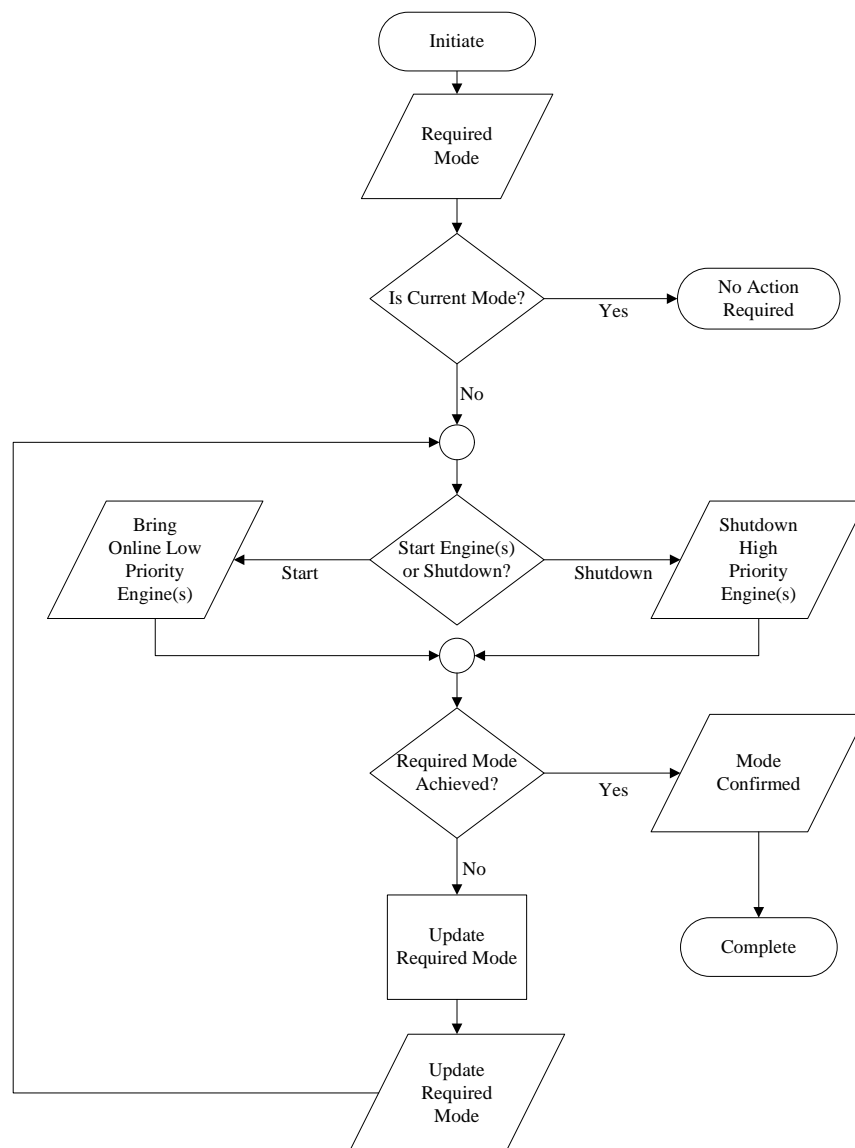


Figure 3.9: Discrete Logic Flow Diagram

to when engines are required to be brought online. For these updates, the number of engines is essentially reduced from either 4 (Full Power) to 2 (Split Plant), 2 to 1 (Trailshaft) or 1 to 0 (Off). The updated required mode logic is then reapplied to the discrete logic in an attempt to realign the propulsion plant.

When the propulsion plant state is required to transition into Full Power, the control logic attempts to bring each of the four engines online. If the engine is already online, no additional commands are sent to that engine. If the engine is ON, but not currently providing power to the propulsion plant, the engine is commanded to come online. If the engine is OFF, the command is issued to start the engine and bring it online. During this process, checks are in place to ensure that the engine comes online. If this does not occur, errors are generated and the propulsion plant attempts to align into a Split Plant configuration.

When the propulsion plant is required to transition into Split Plant, the control logic attempts to bring online either the inboard engines or outboard engines. The outboard engines are those mounted closest to the exterior of the ship on the port and starboard sides. The inboard engines are those mounted closest to the center of the ship on the port and starboard sides. From Figure 2.1 (Page 11), the inboard engines are GTM 1A and 2B, while the outboard engines are GTM 1B and 2A. Logic is in place to bring online the optimal set of either the inboard or outboard engines. The optimal set is determined based on either not starting the set that has errors, starting the partner engine if one is already online or starting the two engines with the lowest priority. If engines are online that are not required to be online, a shutdown command is issued to those engines. If the either of the two Split Plant configurations can not be achieved, the propulsion plant attempts to align into the Trailshaft configuration.

When the propulsion plant is required to transition to Trailshaft, the control logic attempts to bring only one engine online. The engine with the lowest priority is selected as the engine to go online or remain online. If multiple engines are already online, the engines with the lowest priority are set to shutdown. As mentioned in Section 2.2.4, the pitch command is assumed to be 100%. This is for a driving shaft. However, if a shaft is trailing (not powered by an engine), its pitch command

will be set to the user specified Trailshaft pitch command.

When in trailshaft, the engines are commanded to shift, or alternate, after a user specified amount of time. During this shifting process, the engine with the next lowest priority is started and brought online and the currently operating engine is issued a shutdown. This is done to maintain equivalency between each of the available engines.

When the propulsion plant is required to transition to OFF, each of the online engines are issued a shutdown command. The discrete control logic is then reset by clearing errors and re-initializing each of the discrete control parameters.

More detail of the discrete control logic and its associated rationale are included in Appendix C.

Chapter 4: Propulsion Model Simulation

4.1 Propulsion Model

The Automated Propulsion Control Model was developed using the simulation tool Simulink by The MathWorks, Inc. Within Simulink, the dynamics of a plant system can be modeled. Control algorithms can then be developed to govern the plant to ensure that the desired control objectives are satisfied.

The Automated Propulsion Control Model includes the following sub-models, or subsystems, which are further discussed in the sections below.

Propulsion Plant Model includes the models that mimic a ship's propulsion systems.

Hybrid Control Model includes the control algorithms and logic presented in this work.

Test Input Model includes user commands and tester fault signals to verify the proposed control algorithms and logic.

4.1.1 Propulsion Plant Model

The propulsion plant model is designed to mimic a ship's propulsion system. This model includes the dynamics of gas turbine engines, shafting, propellers and supporting auxiliaries.

The most important system of the Propulsion Plant Model is the gas turbine model. This research effort developed only one referenced gas turbine model. Within Simulink, multiple references to a single model can be embedded in a higher level model. This approach was used with the single gas turbine model. It was referenced for Starboard Engine A (1a), Starboard Engine B (1b), Port Engine A (2a) and Port Engine B (2b). These four referenced models were integrated into the Propulsion Plant Model.

The gas turbine model was built using the equations outlined in [50]. As mentioned in Section 2.2.1, the dynamic equations for a gas turbine are not linear. These equations are extremely

nonlinear because of the use of lookup tables and logical switches. Recall Figures 2.2 and 2.3 on Page 16. These lookup tables are composed of data points collected during testing. Given a specified test input value for a given parameter, associated output values of other parameters are measured and data points are collected to form these lookup tables. The data contained in these lookup tables provides the ability to determine what system output parameter values will be given a specified input. Similarly, logical switches are used to change the way a system parameter is calculated. Logical switches are updated, or changed, when the state of the system changes. When these changes occur, the method used to determine a given system parameter may also change.

When developing dynamic models, it is best to organize the system equations and discrete logic into functional groups. This approach was used in this research effort. The complete gas turbine model is broken up and composed of several subsystem models. Most of these subsystems are used to calculate various engine parameters such as gas generator and power turbine speeds, compressor discharge, lube oil and fuel oil manifold pressures, turbine inlet temperature and power turbine torque. Other subsystems include a portion of the engine controller logic used to regulate the engine speed or shutdown the engine if speed, temperature or pressure setpoints are exceeded.

The input signals of the gas turbine model are listed in Table 4.1. The purpose, or function, of these signals are also included. These input commands are sent from either the Automated Propulsion Controller (APC) or the Test Input model. Other feedback signals, such as shafting parameters and states, are issued from other parts of the Propulsion Plant model.

Table 4.1: Gas Turbine Model Inputs

Description	Source	Function
Engine Reset	APC	Reset Local Engine Controller Logic
Engine Start	APC	Initiate an Engine Start
Normal Stop	APC	Initiate an Engine Shutdown
Throttle Input Command	APC	Throttle Command to Engine
Emergency Stop	Test Input	Initiate an Emergency Shutdown
Start Permissive	Test Input	Disable an Engine Start (Out of Service)
Power Turbine Speed	Plant Feedback	Affects Local Engine Controller Logic
Clutch Status	Plant Feedback	Affects Local Engine Controller Logic
Plant Alignment	Plant Feedback	Affects Local Engine Controller Logic
Power On	Plant Feedback	Initializes the Engine Control Logic
Ambient Temperature	Test Input	Affects Engine Dynamic Response
Ambient Pressure	Test Input	Affects Engine Dynamic Response

The output signals of the gas turbine model are listed in Table 4.2. The majority of the output signals are key engine operating parameters, such as speeds, pressures and temperatures. Other outputs represent the operational state of the engine. These states include Ready to Start, Engine Online, Normal Stop Initiated and Normal Stop Complete. The gas turbine model outputs are used by either the APC or other propulsion plant systems.

Table 4.2: Gas Turbine Model Outputs

Description	Destination	Function
Normal Stop Initiated	APC	Feedback Response to Controller
Normal Stop Complete	APC	Feedback Response to Controller
Fuel Metering Valve Position	APC	Feedback Response to Controller
Variable Stator Vane Position	APC	Feedback Response to Controller
Inlet Air Temperature	APC	Feedback Response to Controller
Inlet Air Pressure	APC	Feedback Response to Controller
Compressor Discharge Pressure	APC	Feedback Response to Controller
Power Turbine Inlet Pressure	APC	Feedback Response to Controller
Engine Torque Output	APC	Feedback Response to Controller
	Propulsion Plant	Affect Propulsion Dynamics
Engine Horse Power	APC	Feedback Response to Controller
Lube Oil Pressure	APC	Feedback Response to Controller
Fuel Manifold Pressure	APC	Feedback Response to Controller
Gas Generator Speed	APC	Feedback Response to Controller
	Propulsion Plant	Affect Propulsion Dynamics
Power Turbine Inlet Temperature	APC	Feedback Response to Controller
Power Turbine Speed	APC	Feedback Response to Controller
Cooling Air Temperature	APC	Feedback Response to Controller
Engine Ready To Start	APC	Feedback Response to Controller
Engine Online	Propulsion Plant	Affect Propulsion Dynamics
Throttle Input Command	Propulsion Plant	Affects Propulsion Dynamics
Engine Fuel Consumption	Propulsion Plant	To Be Summed with Other Engines

The four referenced instances of the gas turbine model, which represent the Starboard A and B and Port A and B engines, interact with the other propulsion plant systems. These other systems include two subsystems that represent the ship's starboard and port shafts and propellers. The two starboard gas turbine model instances interact with the starboard shaft and propeller. The port shaft and propeller interacts with the two port gas turbine model instances.

The starboard and port shaft and propeller systems are very similar. These systems are used to represent the dynamics of the shaft, main reduction gear and propeller. The inputs and outputs of these systems are listed in Table 4.3.

Each of the input commands are issued from either the APC or as a test input. As mentioned

Table 4.3: Propulsion Shaft Inputs and Outputs

Input Output	Description	Function
Input	Pitch Command	Commanded Pitch of Propeller Blades
	Engage Brake (A)	Lock Engine (A) Power Turbine
	Disengage Brake (A)	Unlock Engine (A) Power Turbine
	Engage Brake (B)	Lock Engine (B) Power Turbine
	Disengage Brake (B)	Unlock Engine (B) Power Turbine
	Hydraulic Pump Off	Test Input that Affects Slew Rate of the Propeller Blades
Output	Clutch (A) Status	Feedback Response to Controller & Propulsion Plant
	Clutch (B) Status	Feedback Response to Controller & Propulsion Plant
	Actual Pitch	Feedback Response to Controller
	Shaft Speed	Feedback Response to Controller
	Shaft Torque	Feedback Response to Controller
	Propeller Thrust	Feedback Response to Propulsion Plant

before, the pitch command is used to adjust the angle of pitch of the propeller blades. The propeller dynamics, similar to those listed in Section 2.2.2, are included in this system model. The brake commands are used to either engage or disengage the power turbine brake of the gas turbine. When the power turbine brake is disengaged, the power turbine torque cause the shaft to spin. Recall Equation 2.1. This portion of the model mimics the Main Reduction Gear by converting the high speed, low torque of the power turbine to the low speed, high torque of the shaft. The hydraulic pump off command turns off the electric pump. When this pump is not on and the shaft speed is less than 85 rpm, the slew rate of the propeller blades is adversely affected when the propeller is commanded to a different pitch.

The shaft and propeller model outputs, which include clutch status, actual pitch, shaft speed and shaft torque are used by the APC in its control logic. The clutch status is also used by the gas turbine model to define the operational state of the engine. (The engine is ONLINE when the brake is disengaged and the clutch is engaged). The propeller thrust is used to calculate the ship speed. Recall Equations 2.9 through 2.20.

The other subsystems modeled in the Propulsion Plant model include those to mimic supporting auxiliaries or others used to calculate other propulsion plant values. The Hydraulic Oil models determine the status of the starboard and port hydraulic pumps. These statuses affect the slew rate of the propeller blades and how fast or slow the pitch angle changes. Test input signals are included

to fault the hydraulic pumps. Power Up Reset initializes the propulsion plant in its default condition when the simulation starts. The ambient temperature and pressure defined in the Ambient Bias subsystem affect the dynamics of each of the referenced gas turbine models. The Ship Dynamics subsystem calculates the ship speed based on the propeller thrust and other defined ship parameters. The ship speed is sent to the APC to verify that the commanded ship speed is realized.

As aforementioned, the gas turbine model uses table data captured during various engine testing efforts to best describe gas turbine engine characteristics. Table data is also used to calculate ship dynamics, shafting and propeller parameters. These tables contribute to the nonlinearities modeled in the propulsion plant.

4.1.2 Hybrid Control Model

The Automated Propulsion Control Model contains the Hybrid Control Model as a subsystem. This portion of the model contains the control logic developed in this work. This subsystem interfaces directly with the Propulsion Plant Model and Test Input model. Within this model are the continuous control algorithms and discrete control logic designed to automatically govern the operation of a ship.

Output commands are issued from the APC to start and stop engines, adjust the pitch angle of the propeller blades and adjust the throttle input command to the engines. These commands are listed in Table 4.4. The (4) included in this table indicates that the listed command is available for

Table 4.4: Hybrid Controller Outputs

Description	Destination	Function
Engine Reset Command (4)	Propulsion Plant	Resets of Local Engine Controller
Engine Start Command (4)	Propulsion Plant	Initiates an Engine Start
Normal Stop Command (4)	Propulsion Plant	Initiates an Engine Shutdown
Brake Engage Command (4)	Propulsion Plant	Locks Engine Power Turbine
Brake Disengage Command (4)	Propulsion Plant	Releases Engine Power Turbine
Throttle Input Command (4)	Propulsion Plant	Sets Requested Engine Output
Pitch Command (2)	Propulsion Plant	Adjusts Propeller Blades

each of the four engines. The (2) indicates that the pitch command is applicable to the starboard and port propellers.

Also, included in the Table 4.4 list of outputs are the Engine Reset Command, Brake Engage

and Brake Disengage Commands. An engine reset is required prior to an engine start to ensure that the engine’s local controller logic is in the required state to control the engine in a safe and effective manner. The Brake Disengage command is sent to the propulsion plant to ensure that the engine’s power turbine brake is released so that the power turbine can provide torque to turn the shaft. When the engine is shutdown, the Brake engage command is sent and locks the power turbine in place.

To know what commands are issued and when they are sent, the Automated Propulsion Controller has to understand the state of the propulsion plant and what is the desired state. To do this, the controller has to monitor a number of feedback commands from the propulsion plant. The controller must also receive commands from the operator, or in the simulation’s case from the Test Input portion of the model. The controller’s inputs are listed in Table 4.5.

Table 4.5: Hybrid Controller Inputs

Description	Source	Function
Desired Speed	Test Input	Sets Desired Speed of the Ship
Ship Speed	Propulsion Plant	Plant Feedback to Controller
Engine Ready (4)	Propulsion Plant	Confirms Engine is Ready to Start
Gas Generator Speed (4)	Propulsion Plant	Determines Engine State Engine Priority Criterion
Clutch Status (4)	Propulsion Plant	Determines Engine State
Power Turbine Speed (4)	Propulsion Plant	Engine Priority Criterion
Power Turbine Inlet Temperature (4)	Propulsion Plant	Engine Priority Criterion
Power Turbine Torque (4)	Propulsion Plant	Engine Priority Criterion
Compressor Discharge Pressure (4)	Propulsion Plant	Engine Priority Criterion
Reset On Time (4)	Test Input	Resets Accumulated Engine On Time
Reset Selection Timers (4)	Test Input	Resets Criterion Timers
Priority Check Test	Test Input	Method Used to Set Engine Priority
Manual Engine Reset (4)	Test Input	Issues System Reset to Engine
Manual Engine Start (4)	Test Input	Issues Start Command to Engine
Manual Engine Stop (4)	Test Input	Issues Engine Stop
Manual Output Throttle Command (4)	Test Input	Issues Throttle Command to Engine
Manual Pitch Command (2)	Test Input	Issues Propeller Pitch Command

If the outputs of the engine model as listed in Table 4.2 (Page 56) are compared against Table 4.5, a number of signals are not included. These signals include: Normal Stop Initiated, Normal Stop Complete, Fuel Metering Valve Position, Variable Stator Vane Position, Inlet Air Temperature, Inlet Air Pressure, Power Turbine Inlet Pressure, Engine Horse Power, Lube Oil Pressure, Fuel Manifold Pressure and Cooling Air Temperature. Similarly, from the shaft output signals in Table 4.3

(Page 57), shaft torque is not included in Table 4.5. These signals can be included in a future work effort. There are a variety of options available as to how these can be integrated into the controller. These signals can be included in the continuous control portion as feedback responses. The values of these feedback parameters can be used as controlling parameters. Alternatively, they can be evaluated and compared against nominal or alarm (out of range) setpoints and integrated into either the continuous or discrete portion of the controller. The unused signals can also be used in the discrete logic as state confirmation or used to determine the engine priority. However, for the current work, these signals were deemed to be not as important as those input signals listed in Table 4.5.

Some of the controller's inputs are analog feedback parameters from the engines. These engine parameters are used to dictate a portion of the discrete control logic. This portion of the logic is used to establish the engine selection priority. Via the model's Test Input block, the user defines the priority check test method. The engine selection priorities that are implemented in the current work are as follows:

Engine On compares the time that each engine's gas generator speed has exceeded the defined engine ON speed.

Gas Generator Overspeed compares the time that each engine's gas generator speed has exceeded the defined gas generator overspeed limit.

Power Turbine Overspeed compares the time that each engine's power turbine speed has exceeded the defined power turbine overspeed limit.

Turbine Inlet Temperature compares the time that each engine's turbine inlet temperature has exceeded the defined high temperature limit.

Over Torque compares the time that each engine's power turbine torque has exceeded the defined high torque limit.

Compressor Discharge Pressure compares the time that each engine's compressor discharge pressure has exceeded the defined high pressure limit.

Composite Each of the above parameters, with the exception of Engine On, are multiplied by associated weights and summed together. The resulting value, or composite number, of each engine is then compared.

The engine priority is established by ranking the engines by the selection method's accumulated value. As displayed in Figure 3.9, engines with the least accumulated hours, or those with the lowest priority, are selected first for engine starts. Conversely, as also displayed, engines with the highest accumulated hours, or highest priority are selected first for engine shutdowns.

As listed in Table 4.5, the gas generator speed is used to determine the engine priority and engine state. There are three primary states of the engine. As mentioned above, the engine is ON when the gas generator speed is greater than the defined engine ON speed. The engine is ONLINE when the engine is ON and the associated clutch status is engaged (true). The third primary state is OFF, when neither of the above are true. These engine states are used in the discrete logic when bringing engines online and when shutting engines down.

Another input to the discrete logic is the shift command. To avoid running a single engine for an extended period of time when the propulsion plant is in trailshaft mode, the engines can be shifted after the engine has been ONLINE for a defined amount of time. When this shift occurs, the discrete logic brings online the available low priority engine and initiates a shutdown of the engine that has exceed its shift time.

4.1.3 Test Input Model

Test inputs are available to affect various parameters in the Automated Propulsion Controller (APC) and the Propulsion Plant. Additionally, inputs are available to inject user/operator commands to the propulsion plant through the APC. Test inputs are listed in Table 4.6.

Test inputs that affect the propulsion plant include parameter settings for ambient temperature and pressure. These affect gas turbine engine dynamics. Other propulsion plant inputs inject casualty/fault commands that affect the engines and electric hydraulic pumps.

Test Inputs also affect the APC logic. Manual commands are available to override automated commands. These commands are available to reset the local engine controller logic, start and stop

Table 4.6: Test Input Signals

Description	Destination	Function
Desired Speed	APC	Sets Desired Speed of the Ship
Reset On Time	APC	Resets Accumulated Engine On Time
Reset Selection Timers	APC	Resets Accumulated Hours for Selection Criteria
Priority Check Test	APC	Defines Engine Priority Selection Method
Manual Output Throttle Command (4)	APC	Operator Commanded Engine Throttle
Manual Engine Reset (4)	APC	Operator Initiated Engine Reset
Manual Engine Start (4)	APC	Operator Initiated Engine Start
Manual Engine Stop (4)	APC	Operator Initiated Engine Stop
Manual Pitch Command (2)	APC	Operator Specified Propeller Pitch
Emergency Stop (4)	Propulsion Plant	Emergency Shutdown of Engine
Engine Ready To Start (4)	Propulsion Plant	Start Permissive of Engine
Ambient Temperature (4)	Propulsion Plant	Sets Ambient Temperature for Equipment Dynamics
Ambient Pressure	Propulsion Plant	Sets Ambient Pressure for Equipment Dynamics
Hydraulic Pump Off	Propulsion Plant	Turns Off Electric Hydraulic Pump

the engines, adjust the engine's throttle command and adjust the propeller's desired pitch angle. The last group of input commands include those that are used by the APC to set and reset the engine selection logic, which establishes the engine priority as discussed in Section 3.2.4.

4.2 Simulated Results

To validate the functionality of the APC, the propulsion model was executed to evaluate the algorithms and logic of the controller during a number of realistic scenarios. MATLAB test scripts were written for each test scenario. The following subsections detail the results of the simulated scenarios for three basic tests, which demonstrate the hybrid control of the APC. Additional tests, which demonstrate the expanded functionality of the APC, are documented in Appendix D.

4.2.1 Automatic Cold Start

When the ship propulsion system is cold, with each of the propulsion engines OFF, and a command is received for the ship to achieve a desired speed, the appropriate propulsion engine(s) are commanded to come online to provide the power required to achieve the desired speed. To simulate this type of scenario, the Propulsion Model was run with the expectation for the ship to achieve 25 knots. The

command for the ship speed to achieve 25 knots was issued at 50 seconds into the simulation.

If no propulsion engines are running, discrete logic is built into the APC, which command the appropriate number of engines online depending on the desired speed. In this case, since a command of 25 knots is desired, 2 propulsion engines should come online to power the engine. Depending on the user specified engine selection criteria, which is included in the MATLAB script, the most available engines should be commanded to start. For this test, the selection criteria was based on the least amount of operating hours and it selected the outboard engines, Starboard Engine A (1a) and Port Engine B (2b), to start. As displayed in Figure 4.1a, a start command is issued for these engines at the next processing cycle after 50 seconds. After these start commands are issued, the

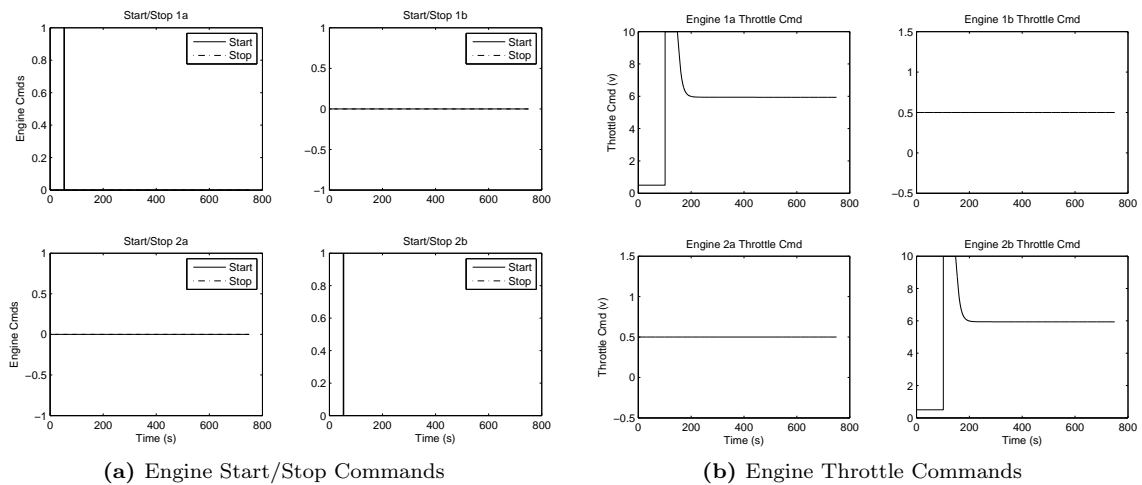


Figure 4.1: Automatic Cold Start Engine Commands

associated throttle commands, displayed in Figure 4.1b, are increased and momentarily set at the maximum command until the actual ship speed approaches the desired speed. Eventually, these throttle commands settle when the ship reaches its desired speed. As displayed in Figure 4.2, the desired ship speed is achieved and successfully maintained. Also displayed in this figure is that the actual ship speed does not begin to increase until after 100 seconds. This is because the engines take approximately 50 seconds to come up to the required speed for the power turbine to engage the clutch to provide the necessary torque to rotate the shaft and propeller.

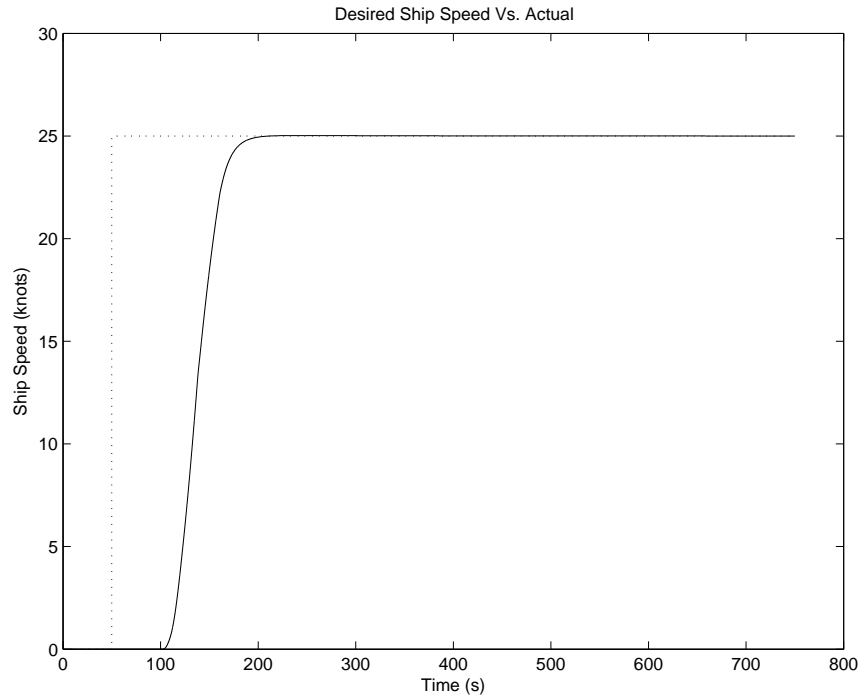


Figure 4.2: Automatic Cold Start Ship Speed

4.2.2 Increase Ship Speed

After the ship has been started and later additional speed is required, the APC must align the propulsion system by commanding the required engines to come online to achieve the new desired speed. To simulate this type of scenario, the Propulsion Model was run with the expectation for the ship to achieve 15 knots, then 25 knots and finally 35 knots. These speed commands for the ship speed were issued at 50 seconds, 500 seconds and 1000 seconds.

Since the initial desired ship speed was lower than that of the Automatic Cold Start Test (15 knots as opposed to 25 knots), only one engine was required to start and power the ship. Based on a different engine selection criteria, which was the least amount of time with a power turbine overspeed condition, Port Engine A (2a) was commanded to power the ship in a trailshaft configuration. As displayed in Figure 4.3a, Engine 2a achieved the desired state with the throttle commands displayed in Figure 4.3b.

Once the speed of 15 knots was achieved, as displayed in Figure 4.4, the ship was later commanded

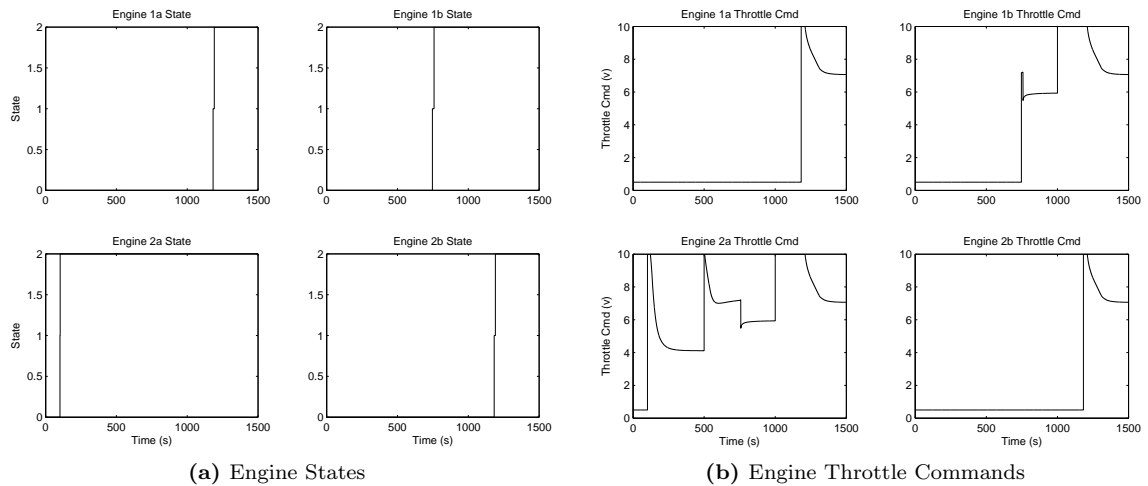


Figure 4.3: Increasing Ship Speed Engine States and Throttles

to 25 knots. However, the ship could not achieve the desired speed of 25 knots in the trailshaft mode. Therefore, the propulsion system was realigned and commanded to a split plant configuration. Therefore, the other inboard engine, Starboard Engine B (1b) was commanded to start at 698 seconds. Once this engine was online, the desired ship speed was achieved.

The ship speed was later set to 35 knots. This speed could not be achieved in the split plant mode. Therefore, full power was required and the outboard engines were commanded to start when the ship speed was no longer increasing when the inboard engine throttle commands were at the maximum. This occurred at 1134 seconds. With each of the engines online, the ship was able to achieve 35 knots.

4.2.3 Reduce Ship Speed

When speed is no longer of the essence and an updated lower speed command is issued, the propulsion system may command engines to shutdown because more power is available than is currently required. To simulate this type of scenario, the Propulsion Model was run with the expectation for the ship to initially achieve 35 knots. The desired speed was later reduced to 12 knots. These speed commands for the ship speed were issued at 50 seconds and 500.

Because the ship was initially commanded to achieve 35 knots, all of the propulsion engines were

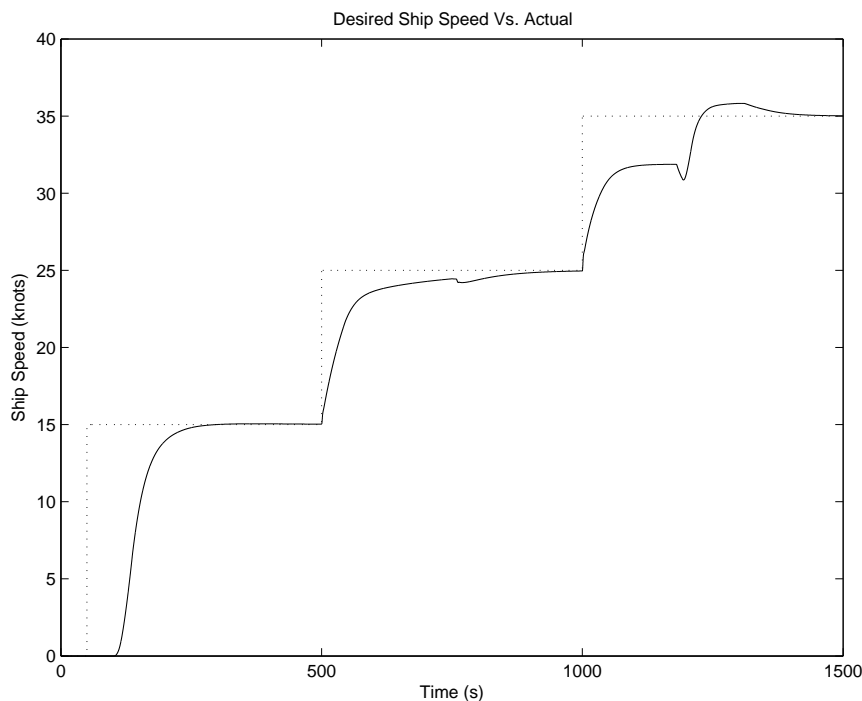


Figure 4.4: Increasing Ship Speeds

required to come online to power the ship. Since all of the engines were required, the engine selection criteria was not a factor in determining which engines should be started. However, when the speed command was reduced, the engine selection criteria was used to determine which engines should be shutdown. When the command was issued for 12 knots, it was realized that there was an excess of power. Based on the engine selection criteria, which was the engine with the most time with the turbine inlet temperature in an overtemperature condition, the outboard engines, Starboard Engine A (1a) and Port Engine B (2b), were commanded to shutdown at 567 seconds. This is displayed in Figure 4.5a. As displayed in Figure 4.5b, these engines began a 300 second cooldown and transitioned from ONLINE (Engine State 2) to ON (Engine State 1) before transitioning to OFF (Engine State 0).

With the inboard engines ONLINE, it was realized that there was still an excess of power, therefore, the propulsion system aligned from the split plant configuration to trailshaft mode, leaving only Starboard Engine B (1b) online to power the ship. As displayed in Figure 4.5a, Port Engine A (2a) was commanded to stop at 843 seconds. As displayed in Figure 4.6, the single engine achieved

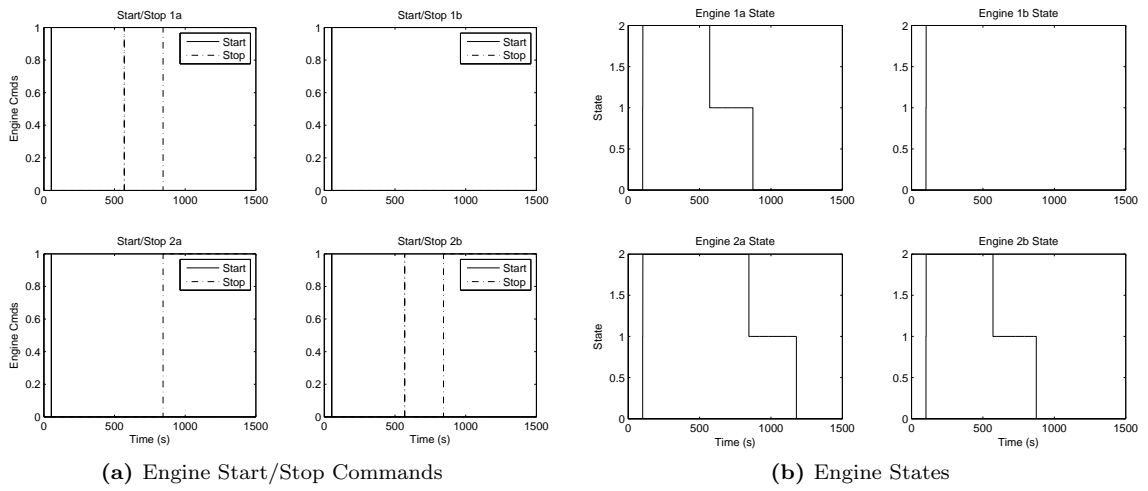


Figure 4.5: Reducing Ship Speed Engine Start/Stop Commands and States

a steady state of the desired speed of 12 knots.

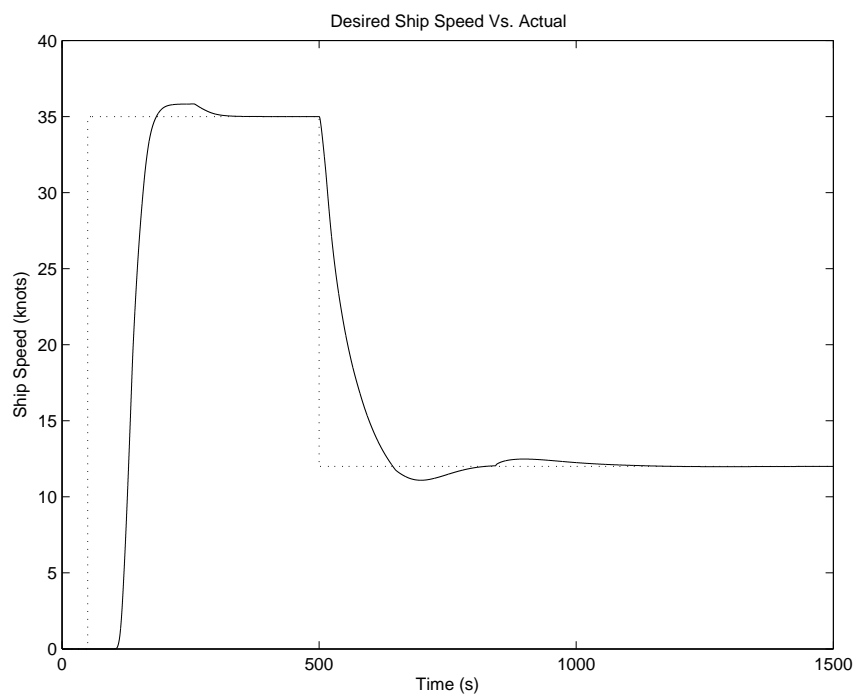


Figure 4.6: Reducing Ship Speeds

Chapter 5: Conclusion

The work included in this thesis presents an approach for using a hybrid controller to automatically control a propulsion system of a ship. However, this work could be further extended to improve upon some of the techniques and results that were realized.

5.1 Summary of Research

Today, to control the propulsion system of a ship, an operator has the responsibility to monitor the state of the propulsion system and issue discrete (On/Off) commands to start and stop propulsion engines when required. Additionally, the operator must issue continuous (throttle) commands to increase or decrease the output of the engines, which in turn increase or decrease the speed of the ship. To perform these functions, the operator must have a full understanding of the propulsion system and the potential impacts of his actions.

This work presents an approach for implementing a hybrid controller, which could be used to automatically control the propulsion system of a ship with little to no human interaction. The hybrid controller is designed with a continuous portion, which interacts with a discrete portion. Collectively, these two portions effectively enable the hybrid controller to perform its required functions.

The ship propulsion system used as a basis for this research is analogous to the propulsion system of the Arleigh Burke Class Guided Missile Destroyers used by the United States Navy and other navies throughout the world. The primary components of the propulsion system are the four propulsion engines and two shafts, each with an attached controllable pitch propeller. The propulsion system can be aligned in one of three nominal configurations: trailshaft mode, where one propulsion engine is used to power one of the two shafts, leaving the unpowered shafted to freely spin, or trail; split plant mode, where each shaft is powered by a propulsion engine (two engines are powering the ship); full power mode, where both shafts are powered by two propulsion engines (four engines are powering the ship). A single throttle command is issued to each of the powering propulsion engines.

Various approaches were taken to implement the hybrid controller. The majority of the approaches revolved around which continuous control method should be used for the controller. While performing this work, control methods investigated for the continuous portion of the controller included implementing a Lagrange multiplier approach, linear quadratic regulator approach and tracking control approach before a proportional integral approach was implemented. The PI approach proved to be functional and acceptable as the continuous controller.

The discrete logic of the hybrid controller can be modeled as a Petri net. The primary function of the discrete logic was to command the discrete states of the propulsion engines. Additionally, the discrete logic was used to realign the propulsion system when the desired speeds could not be achieved or when the power available was much greater than the power required to achieve the commanded speed.

A model was developed in Simulink to verify the performance of the hybrid controller. The Simulink model contained three primary systems. One system was a propulsion system model, which was used to mimic the propulsion system of the ship. The second system represented the Automated Propulsion Controller, which contained the continuous control algorithms and discrete logic of the hybrid controller. The third primary system was a user and test input system, which was used to inject operator commands to the controller and other plant casualty conditions to which the controller was to observe and respond accordingly.

The model was executed for a number of realistic scenarios. These test scenarios demonstrated the functionality of the hybrid controller. When the ship was commanded to achieve a desired speed, the propulsion system of the ship was appropriately aligned and the desired speed was achieved. The tests proved that the hybrid controller could automatically perform the functions that are manually performed by an operator today, thus taking a man out of the loop.

5.2 Future Research Opportunities

Even though the work developed for this thesis proved to present an effective hybrid controller for a ship propulsion system, additional work is possible to produce a better controller. The PI controller approach used for this work is an effective, but rather simply approach used to implement

the continuous control of the hybrid controller. Even though other approaches were investigated, which none proved implementable, more advanced algorithm approaches should be evaluated. If these evaluations do not yield anything else feasible, at a minimum a PI controller with a more dynamic, self switch approach should be investigated. The current work did include some switching of the PI controller, which modified the integral gain based on the mode. However, more work could be done to ensure that the gains are more appropriately weighted for both the proportional and integral gains.

After a modified continuous controller is implemented, the discrete logic should be reevaluated. Since the continuous algorithms and the discrete logic of the hybrid controller are tightly coupled, the timing embedded in the discrete logic may also require modification to ensure the controller works to its optimum performance.

From the test results, a few minor tweaks are also recommended. It was observed that when a propulsion engine was commanded to stop and later other propulsion engines were commanded to stop, the first engine also received that second stop command while it was in its cooldown period and not powering the ship. Since this was a stop command, there really was no adverse affects of this command being sent a second time. Another tweak that could be made to the controller would be to ensure a more constant rate of speed when engines are shifted while in trailshaft mode. It was observed when the propulsion engines shifted, a slight dip in the ship speed also occurred.

Also from the test results, it was observed that during some cases, there was an overshoot in the desired speed. This can be seen in the test cases where the ship was commanded to start at a high rate of speed. It is understood that the ship is large and once it is moving, it can not instantly adjust its speed. However, better controls should be in place to ensure that the ship does not overshoot its desired speed even though it eventually achieves and settles into the commanded speed.

List of References

- [1] Timothy Doyle, Timothy Nixon, Henry Robey, David Clayton, and Thomas Martin. Integrated propulsion cross-connect for the DDG 51. Technical report, Alion Science and Technology and Naval Sea Systems Command, May 2007.
- [2] Basic principles of ship propulsion. (visited 02Feb2009)
<http://www.scribd.com/doc/11562077/Basic-Principles-of-Ship-Propulsion>.
- [3] Anthony F. Molland. *Maritime Engineering Reference Book - A Guide to Ship Design, Construction and Operation*. Elsevier, 2008.
- [4] Michael C. Tracy. *Marks' Standard Handbook for Mechanical Engineers (11th Edition)*, chapter Marine Engineering, pages 11–40–11–58. McGraw-Hill, 2007.
- [5] Edward V. Lewis. *Principles of Naval Architecture (Second Revision), Volume II - Resistance, Propulsion and Vibration*. Society of Naval Architects and Marine Engineers, 1988.
- [6] Edward V. Lewis. *Principles of Naval Architecture (Second Revision), Volume III - Motions in Waves and Controllability*. Society of Naval Architects and Marine Engineers, 1989.
- [7] Eric. C. Tupper. *Introduction to Naval Architecture (4th Edition)*. Elsevier, 2004.
- [8] Thomas Lamb. *Ship Design and Construction*, volume 1. Society of Naval Architects and Marine Engineers, 2003.
- [9] Thomas Lamb. *Ship Design and Construction*, volume 2. Society of Naval Architects and Marine Engineers, 2004.
- [10] K.J. Rawson and E.C. Tupper. *Basic Ship Theory (5th Edition)*. Elsevier, 2001.
- [11] Harold R. Booher. *Handbook of Human Systems Integration*. John Wiley & Sons, 2003.
- [12] GE Aircraft Engines. Control system, LM2500CMS commercial marine application singular annular combustor adaptation, March 10, 2003.
- [13] GE Aviation. Electrical interface control document propulsion gas turbine system to applicable ship systems, October, 30 2009.
- [14] Jerry Petrey. Universal Engine Controller. *Embedded Systems Programming*, 8(9):80–96, September 1995.
- [15] Max Howell and Bill Hartshorn. Rapid design, integration and test of improved digital fuel control for the LM2500 gas turbine engine. In *Proceedings of the International Gas Turbine and Aeroengine Congress and Exposition*, Houston, Texas, 1995.
- [16] Max Howell. Integration and test of the digital fuel control option for the Universal Engine Controller. In *Proceedings of the 11th International Ship Control Systems Symposium*, Ottawa, Canada, 1997.
- [17] Marc A. Calabrese, Lee F. Skarbek, Parag H. Shah, and Mazen K. Yassine. Using a digital fuel control system and digital fuel engine controller for US Navy LM2500 engines. In *Proceedings of the American Society of Mechanical Engineers, International Gas Turbine Institute, Turbo Expo*, pages 1035–1039, Amsterdam, Netherlands, June 2002.

- [18] Woodrow Clifton and Dave Reisteter. Rapid development of the maritime advanced digital controller. In *Proceedings of the ASNE Automation and Controls Symposium*, Milwaukee, Wisconsin, 2010.
- [19] J.C. McMahon. Technical discussions: Marine propulsion systems, mechanical drives, electrical drives and combined operating cycles, September 1983.
- [20] J.D. Ingram, R.M. Adair, D.M. Gildart, R. Cunningham, and J.R. Mailhot. DDG 51 machinery control system. In *Proceedings of the 10th International Ship Control Systems Symposium*, Ottawa, Canada, 1993.
- [21] R. Banning, M.A. Johnson, and M.J. Grimble. Advanced control design for marine diesel engine propulsion systems. *Journal of Dynamic Systems, Measurement and Control*, 119:167–174, June 1997.
- [22] M.J. Grimble, S.A. Carr, and M.R. Katebi. Integrated ship control using H_∞ robust design techniques. In *Proceedings of the Third IEEE Conference on Control Applications*, volume 2, pages 1087–1091, Glasgow, Scotland, United Kingdom, August 1994.
- [23] Tadashi Kashima and Jun Takata. An optimal control of marine propulsion system considering ship dynamics. In *Proceedings of the 2002 IEEE International Conference on Control Applications*, Glasgow, Scotland, United Kingdom, September 2002.
- [24] Jeng Yih Juang and B. C. Chang. Robust control theory applied to ship maneuvering. In *Proceedings of the 38th IEEE Conference on Decision and Control*, volume 3, pages 2186–2191, Phoenix, Arizona, December 1999.
- [25] Joey A. Duvall, Lamar Davidson, and Woodrow Clifton. A real-time hardware-in-the-loop simulation environment for a ship propulsion system. In *Proceedings of the American Society of Naval Engineers Automation and Controls Symposium*, Biloxi, Mississippi, 2007.
- [26] Wayne Cantrell. Controlling combined propulsion systems using automation equipment. In *Proceedings of the 14th International Ship Control Systems Symposium*, Ottawa, Canada, 2009.
- [27] André B. Leal and José E. R. Cury. On the existence of optimal solutions for the modular supervisory control of hybrid systems. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, pages 941–946, Paradise Island, Bahamas, December 2004.
- [28] V. Azhmyakov, V.G. Boltyanski, and A. Poznyak. First order optimization techniques for impulsive hybrid dynamical systems. In *Proceedings of the International Workshop on Variable Structure Systems*, pages 173–178, Antalya, Turkey, June 2008.
- [29] Alberto Bemporad and Nicolò Gioretti. Logic-based solution methods for optimal control of hybrid systems. *IEEE Transactions on Automatic Control*, 51(6):963–976, June 2006.
- [30] Alberto Bemporad and Nicolò Gioretti. A logic-based hybrid solver for optimal control of hybrid systems. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, pages 640–645, Maui, Hawaii, December 2003.
- [31] Abolfazl Jalilvand and Sohrab Khanmohammadi. Integrating of event detection and mode recognition in hybrid systems by fuzzy petri net. In *Proceedings of the 2004 IEEE Conference on Robotics, Automation and Mechatronics*, 2004.
- [32] United States Naval Academy. Marine Electric Drive Overview. (visited 03Feb2009) http://www.usna.edu/EE/ee331/Handouts/Electric_Drive.pdf.
- [33] Harry A. Doell. What is the controllable-pitch propeller? *Marine Engineering*, 58(8):71–76, August 1953.

- [34] Ferdinand P. Beer and E. Russell Johnston Jr. *Vector Mechanics for Engineers: Statics and Dynamics, Fifth Edition*. McGraw-Hill Book Company, 1988.
- [35] Donald M. MacPherson, Vincent R. Puleo, and Matthew B. Packard. Estimation of entrained water added mass properties for vibration analysis. *SNAME New England Section*, June 2007.
- [36] Christos G. Cassandras. Optimal control of a class of hybrid systems. In *Proceedings of the 36th IEEE Conference on Decision and Control*, 1997.
- [37] Hogge Hu, Dagui Huang, Hong Hu, and Guangying Sun. Combining modeling and fault detection in automated manufacturing systems based on hybrid petri net. In *Proceedings of the 2004 International Conference on Intelligent Mechatronics and Automation*, pages 728–732, Chengdu, China, August 2004.
- [38] M. Rezai, P. D. Lawrence, and M. R. Ito. Analysis of faults in hybrid systems by global petri nets. In *Proceedings of the 1995 IEEE International Conference on Systems, Man and Cybernetics*, volume 3, 1995.
- [39] Fabio Balduzzi and Angela Di Febbraro. Combining fault detection and process optimization in manufacturing systems using first-order hybrid petri nets. In *Proceedings of the 2001 International Conference on Robotics & Automation*, 2001.
- [40] René David. Modeling of hybrid systems using continuous and hybrid petri nets. In *Proceedings of the 7th IEEE International Workshop on Petri Nets and Performance Models*, 1997.
- [41] René David. *Discrete, Continuous and Hybrid Petri Nets*. Springer, 2005.
- [42] Michael S. Branicky, Vivek S. Borkar, and Sanjoy K. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, January 1998.
- [43] Singiresu S. Rao. *Engineering Optimization Theory and Practice*. John Wiley & Sons, Inc., 2009.
- [44] Kemin Zhou and John C. Doyle. *Essentials of Robust Control*. Prentice Hall, Inc., 1998.
- [45] B. C. Chang. Robust control II. Lecture Notes, February 2008.
- [46] Martin J. Corless and Arthur E. Frazho. *Linear Systems and Control An Operator Perspective*. CRC Press, 2003.
- [47] B. C. Chang and Chunlong Hu. Multivariable control and failure accommodation in eye-head-torso target tracking. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, pages 978–983, Maui, Hawaii, December 2003.
- [48] Chunlong Hu. *Design and Implementation of Multivariable Cooperative Control and Failure Accommodation*. PhD thesis, Drexel University, Philadelphia, Pennsylvania, February 2005.
- [49] Steven M. Kaplan. *Wiley Electrical and Electronics Engineering Dictionary*. John Wiley & Sons, Inc., Hoboken, New Jersey, 2004.
- [50] Richard J. Skertic. Lm2500 nonlinear simplified engine model for iec. Technical report, General Electric, Advanced Technology Operation, October 1986.

Appendix A: List of Symbols and Abbreviations

A.1 List of Symbols

The following symbols were used within this thesis:

α	Angular Acceleration
η_s	Shaft Efficiency
λ	Lagrange Multiplier
μ	Marking of a Petri Net
ρ	Density of Water
ρ_{air}	Density of Air
τ	Firing Time of a Transition
ω	Angular Velocity
a	Ship Acceleration
A	Synchronous Arc Weights from Continuous Places to Transitions
A_{air}	Cross Sectional Area Above Water
A_s	Wetted Surface Area
B	Synchronous Arc Weights from Transitions to Continuous Places
C_f	Friction Resistance Coefficient
C_r	Residual Resistance Coefficient
d	Propeller Diameter
D⁺	Weight of Arcs from Transitions to Places
D⁻	Weight of Arcs from Places to Transitions
F	Force
F	State Feedback Matrix
h	Indication of Continuous or Discrete Place
H	Hamiltonian Function

I	Moment of Inertia
J	Speed of Advance
K	Reference Resistance Force
K_g	Reduction Gear Constant
K_q	Torque Coefficient
K_t	Thrust Coefficient
\mathbf{L}	Observer Gain Matrix
m	Total Mass
m_e	Entrained Water Mass
m_s	Ship Mass
n	Propeller Speed
\mathbf{P}	Set of Discrete Places
r	Propeller Radius
Q	Torque
Q_d	Delivered Torque
Q_e	Engine Torque
Q_f	Frictional Torque Losses
Q_p	Propeller Torque
Q_u	Unbalanced Torque
R_a	Air Resistance
R_f	Frictional Resistance
R_r	Residual Resistance
R_t	Towing Resistance
S_r	Real Slip Ratio
T	Ship Thrust
T	Terminal Time
\mathbf{T}	Set of Discrete Transitions

t_d	Thrust Reduction Factor
T_p	Propeller Thrust
t_{ic}	Throttle Input Command
\mathbf{u}	Control Vector
V	Ship Speed
V_a	Ship Speed of Advance
w	Wake Coefficient
W	Weighing Function
\mathbf{x}	State Vector

A.2 Abbreviations

The following abbreviations were used within this thesis:

APC	Automated Propulsion Controller
DDG	Guided Missile Destroyer
GPN	Global Petri Net
GTM	Gas Turbine Module
HPN	Hybrid Petri Net
TIC	Throttle Input Command

Appendix B: System Equations Linearization

The software tool Maple was used to linearize the system equations from Equations 2.22 and 2.23. The system was linearized about four operating points; one when the shaft began to rotate, one when the ship speed started to increase, another when both the shaft speed and ship speed were increasing and the last when both the shaft speed and ship speed were at a steady state. These operating points were found for each mode of the ship using simulated data.

B.1 Trailshaft Mode

The state equations for the propulsion system in Trailshaft are

$$\dot{\omega}_{ts} = \frac{K_g (9388tic - 198.405) - \frac{1}{73} (0.1862 - 0.1238 (\frac{0.97V}{\omega d})) \rho \omega^2 d^5}{I} \quad (\text{B.1})$$

$$\dot{V}_{ts} = \frac{\frac{K_t \rho \omega^2 d^4}{7.5} \left(1 - 0.1 \frac{K_t \rho \omega^2 d^4}{7.5 T p_{max}}\right) - \frac{(C_f + C_r) \rho V^2 A_s}{2}}{m} \quad (\text{B.2})$$

Table B.1 includes the operating points for the trailshaft mode.

Table B.1: Trailshaft Operating Points

	Increasing Shaft Speed	Increasing Ship Speed	Both Increasing	Steady State
<i>tic</i>	0.3	1.1	5	6.65
ω (rad/s)	1.085	4.694	10.73	12.43
V (m/s)	0.027	2.92	10.34	12.07

When linearizing about each operating points the following **A** matrices were found,

$$A_{ts(\text{shaft increasing})} = \begin{bmatrix} -0.1643357278 & 0.01023498546 \\ 0.02175373056 & -0.001869743361 \end{bmatrix} \quad (\text{B.3})$$

$$A_{ts(\text{ship increasing})} = \begin{bmatrix} -0.7368047481 & 0.04707149993 \\ 0.09612617629 & -0.08054435806 \end{bmatrix} \quad (\text{B.4})$$

$$A_{ts(\text{both increasing})} = \begin{bmatrix} -1.527546396 & 0.09933124135 \\ 0.1895319911 & -0.2727889541 \end{bmatrix} \quad (\text{B.5})$$

$$A_{ts(\text{steady state})} = \begin{bmatrix} -1.768721857 & 0.1150846291 \\ 0.2144792407 & -0.3198362267 \end{bmatrix} \quad (\text{B.6})$$

The **B** matrix was the same for each operating point.

$$B_{ts} = \begin{bmatrix} 1.553679773 \\ 0 \end{bmatrix} \quad (\text{B.7})$$

B.2 Split Plant Mode

The state equations for the propulsion system in split plant are

$$\dot{\omega}_{sp} = \frac{K_g (9388tic - 198.405) - \frac{1}{73} (0.1862 - 0.1238 (\frac{0.97V}{\omega d})) \rho \omega^2 d^5}{I} \quad (\text{B.8})$$

$$\dot{V}_{sp} = 2 \frac{\frac{K_t \rho \omega^2 d^4}{7.5} \left(1 - 0.1 \frac{K_t \rho \omega^2 d^4}{7.5 T p_{max}} \right) - \frac{(C_f + C_r) \rho V^2 A_s}{2}}{m} \quad (\text{B.9})$$

Table B.2 includes the operating points for the split plant mode.

Table B.2: Split Plant Operating Points

	Increasing Shaft Speed	Increasing Ship Speed	Both Increasing	Steady State
<i>tic</i>	0.3	1.25	5	6.65
ω (rad/s)	1.085	4.99	10.53	12.2
<i>V</i> (m/s)	0.0135	2.075	7.27	8.538

When linearizing about each operating points the following **A** matrices were found,

$$A_{sp(\text{shaft increasing})} = \begin{bmatrix} -0.1642083800 & 0.01023498546 \\ 0.04347300505 & -0.003739489918 \end{bmatrix} \quad (\text{B.10})$$

$$A_{sp(\text{ship increasing})} = \begin{bmatrix} -0.6839663765 & 0.04427928271 \\ 0.1787758651 & -0.1163505881 \end{bmatrix} \quad (\text{B.11})$$

$$A_{sp(\text{both increasing})} = \begin{bmatrix} -1.528902349 & 0.1012178746 \\ 0.3792644216 & -0.3957598115 \end{bmatrix} \quad (\text{B.12})$$

$$A_{sp(\text{steady state})} = \begin{bmatrix} -1.770267054 & 0.1172542574 \\ 0.4291953181 & -0.4610718249 \end{bmatrix} \quad (\text{B.13})$$

The \mathbf{B} matrix was the same for each operating point.

$$B_{sp} = \begin{bmatrix} 1.553679773 \\ 0 \end{bmatrix} \quad (\text{B.14})$$

B.3 Full Power Mode

The state equations for the propulsion system in full power are

$$\dot{\omega}_{fp} = \frac{2K_g (9388tic - 198.405) - \frac{1}{73} (0.1862 - 0.1238 (\frac{0.97V}{\omega d})) \rho \omega^2 d^5}{I} \quad (\text{B.15})$$

$$\dot{V}_{fp} = 2 \frac{\frac{K_t \rho \omega^2 d^4}{7.5} \left(1 - 0.1 \frac{K_t \rho \omega^2 d^4}{7.5 T_{pmax}}\right) - \frac{(C_f + C_r) \rho V^2 A_s}{2}}{m} \quad (\text{B.16})$$

Table B.3 includes the operating points for the full power mode.

Table B.3: Full Power Operating Points

	Increasing Shaft Speed	Increasing Ship Speed	Both Increasing	Steady State
<i>tic</i>	0.3	0.9	5	6.65
ω (rad/s)	2	6.01	15.18	17.56
V (m/s)	0.98	3.85	14.46	16.69

When linearizing about each operating points the following \mathbf{A} matrices were found,

$$A_{fp(\text{shaft increasing})} = \begin{bmatrix} -0.3022125394 & 0.01886633263 \\ 0.07986731272 & -0.00868891619 \end{bmatrix} \quad (\text{B.17})$$

$$A_{fp(ship\ increasing)} = \begin{bmatrix} -0.8746713274 & 0.05669332958 \\ 0.2267753967 & -0.1528331022 \end{bmatrix} \quad (\text{B.18})$$

$$A_{fp(both\ increasing)} = \begin{bmatrix} -2.164563682 & 0.1431954647 \\ 0.5011087551 & -0.5514393076 \end{bmatrix} \quad (\text{B.19})$$

$$A_{fp(steady\ state)} = \begin{bmatrix} -2.504285435 & 0.1656464006 \\ 0.5517389051 & -0.6346690767 \end{bmatrix} \quad (\text{B.20})$$

The \mathbf{B} matrix was the same for each operating point.

$$B_{fp} = \begin{bmatrix} 3.107359546 \\ 0 \end{bmatrix} \quad (\text{B.21})$$

Appendix C: Discrete Control Rationale

The rationale of the discrete control logic as discussed in Section 3.2.4 is further developed in this appendix. Also included are several figures from Stateflow. These figures display how the discrete logic was implemented.

C.1 Reconfigure Propulsion Plant

The main purpose of the discrete control logic implemented in the Automated Propulsion Controller is to realign the propulsion plant based on the desired speed of the ship. As displayed in Figure C.1, the logic initializes by setting variables used in the discrete logic to 0. From the Initialize state, it is possible to transition to the TrailShaft, SplitPlant or FullPower state. The state in which the logic transitions depends on the variable *req_engines*, which is the number of required engines.

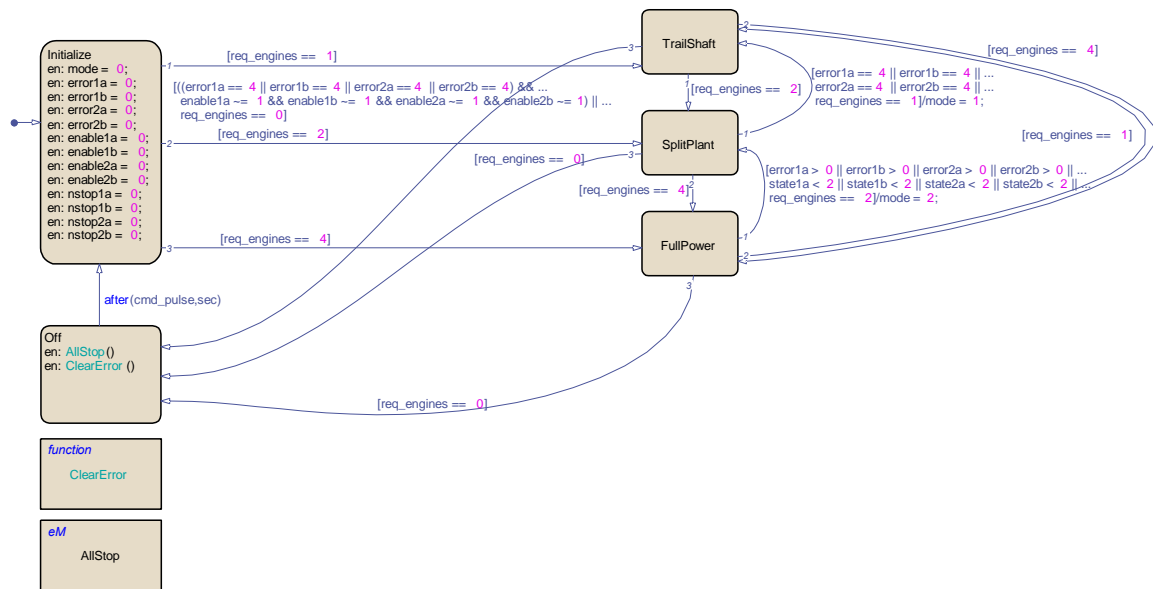


Figure C.1: Reconfigure Propulsion Plant

From the **FullPower** state, it is possible to transition to the **SplitPlant** state if the number of required engines drops from 4 to 2. This transition will also occur if either of the four engines experience any problems (errors) or does not achieve the **ONLINE** state. A transition to the **TrailShaft**

state will occur if the number of required engines drops from 4 to 1. If no engines are required, a transition to the OFF state will occur.

From the SplitPlant state, a transition to the FullPower state will occur if the number of required engines increases from 2 to 4. However, if the number of required engines decreases from 2 to 1, a transition to the TrailShaft state will occur. This transition will also occur if errors result and Split Plant can not be achieved. If no engines are required, a transition to the OFF state will occur.

From the TrailShaft state, a transition to the FullPower state will occur if the number of required engines increases from 1 to 4. If the number of required engines increase from 1 to 2, a transition to the SplitPlant state will occur. If no engines are required, a transition to the OFF state will occur. This transition will also occur if errors result and Trailshaft can not be achieved.

C.2 Align to Full Power

When the discrete logic enters the FullPower State, it initializes by attempting to bring all four engines ONLINE as displayed in Figure C.2. Once all four engines are ONLINE, the plant configuration variable *mode* is set to 3. This variable is used to confirm to the continuous control logic that the desired state has been achieved. In this case, 3 represents that the propulsion plant is aligned in the Full Power plant configuration.

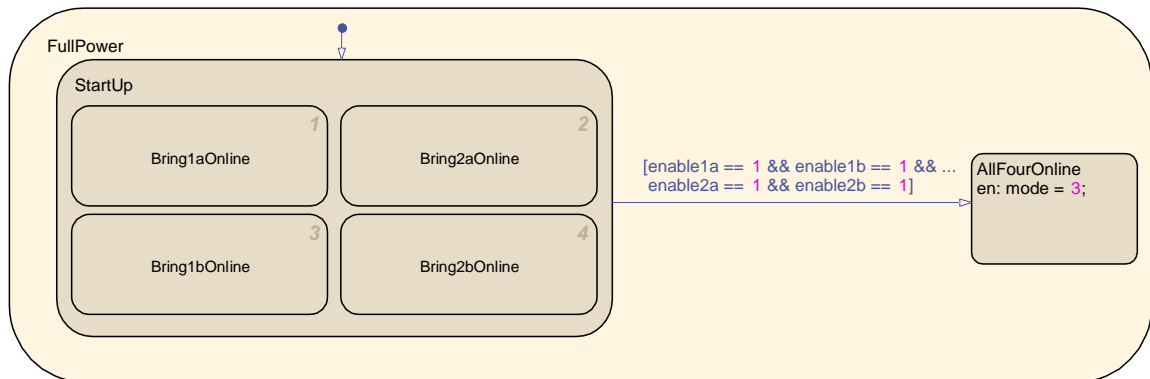


Figure C.2: Command Full Power

C.3 Bring Engine Online

When engines are brought ONLINE as displayed in Figure C.3, the discrete control logic issues commands to the engines. However, if the engine is already ONLINE ($state = 2$), no commands are issued. If the engine is ON ($state = 1$), the output throttle command is enabled so the throttle input command TIC coming from the continuous logic can be applied to the engine. If the engine is OFF ($state = 0$), a system reset is first sent to the local engine controller to reset its logic. If the engine is ready to start, a start command is issued to the engine. If it is not ready to start, an error will result. After the start command has been issued, the engine speed increases until it reaches the ON state. If the engine does not achieve the ON state within a prescribed amount of time, an error will result. However, once the engine is ON, the output throttle command is enabled to transition the engine to the ONLINE state.

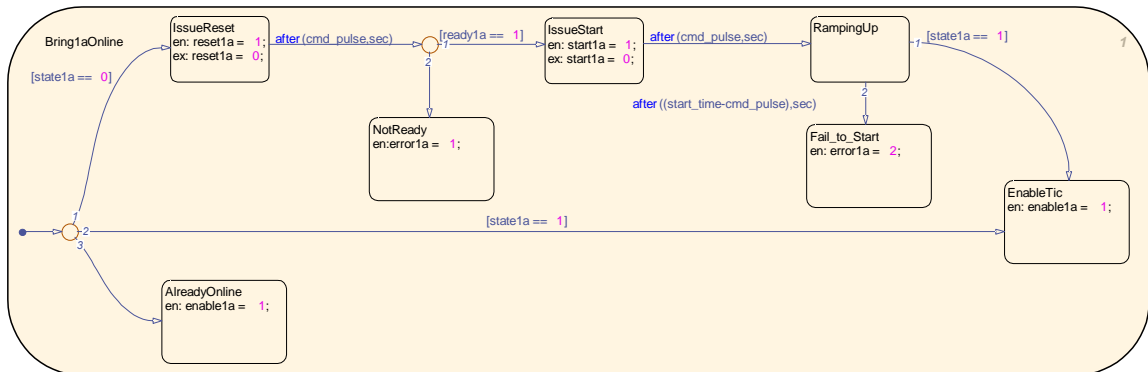


Figure C.3: Bring Engine Online

C.4 Align to Split Plant

As mentioned above, when errors are generated while attempting to set the propulsion plant to Full Power, the discrete logic attempts to set the propulsion plant into a Split Plant configuration. When Split Plant is the desired mode, the discrete logic is performed as displayed in Figure C.4.

The logic initializes by clearing any existing errors. A check is performed to see if any of the outboard engines are already ONLINE. If one of these is already ONLINE and the other outboard engine is ready to start, an attempt is made to set the Split Plant configuration with the outboard engines ONLINE. The discrete control logic as displayed in Figure C.5 is executed to achieve this

configuration.

After this first check is performed and no outboard engines are already ONLINE, a check is performed to see if any of the inboard engines are ONLINE with the other inboard engine ready to start. If this is true, the logic attempts to put the propulsion plant into a Split Plant configuration with the inboard engines. The discrete control logic for this is displayed in Figure C.6.

If one of the inboard or outboard engines are already ONLINE and an attempt is made to start its associated Split Plant engine, errors could result during the start. If errors result while attempting to bring the outboard engines ONLINE, this option is aborted and an attempt is made to bring the inboard engines ONLINE. Similarly, if errors result while attempting to bring the inboard engines ONLINE, this attempt is aborted and an attempt to bring the outboard engines ONLINE is then made.

Returning to Figure C.4, if no engines are already ONLINE, a check is performed to see if any of the engines are ON. Based on which engine is ON, its associated Split Plant engine is commanded to come ONLINE. This logic is displayed in Figure C.7. If an outboard engine is ON and its associated engine is ready to start when the inboard engines are both OFF, an attempt is made to align the propulsion plant in the Split Plant configuration using the outboard engines. Similarly, if an inboard engine is ON and its associated engine is ready to start when the outboard engines are both OFF, an attempt is made to use the inboard engines for the Split Plant configuration. If an inboard engine and an outboard engine are both ON, the engine priority as discussed in Section 3.2.4 is used to determine if the inboard or outboard engines will be used in the Split Plant configuration. Similar to the error logic as mentioned above, if errors result while attempting to use the outboard engines, the inboard engines will be used. The converse also applies.

The logic contained in the PowerUpOutboard and PowerUpInboard states of Figure C.7 is displayed in Figure C.8. In this figure, the inboard or outboard engines are simultaneously brought ONLINE in accordance with the logic displayed in Figure C.3.

From Figure C.4, if no engines are ON or ONLINE, the Start2 state is reached and two engines are attempted to start. This logic is displayed in Figure C.9. Either the inboard or outboard engines

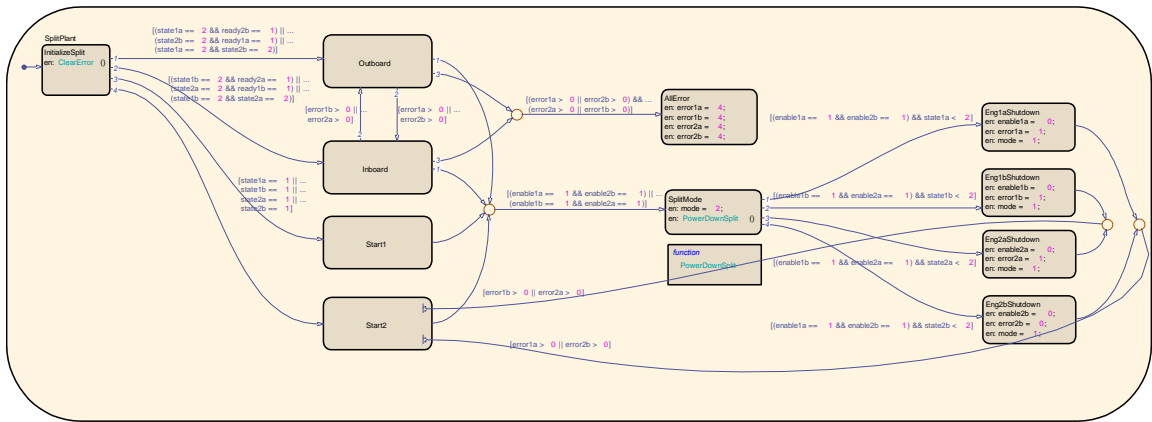


Figure C.4: Command Split Plant

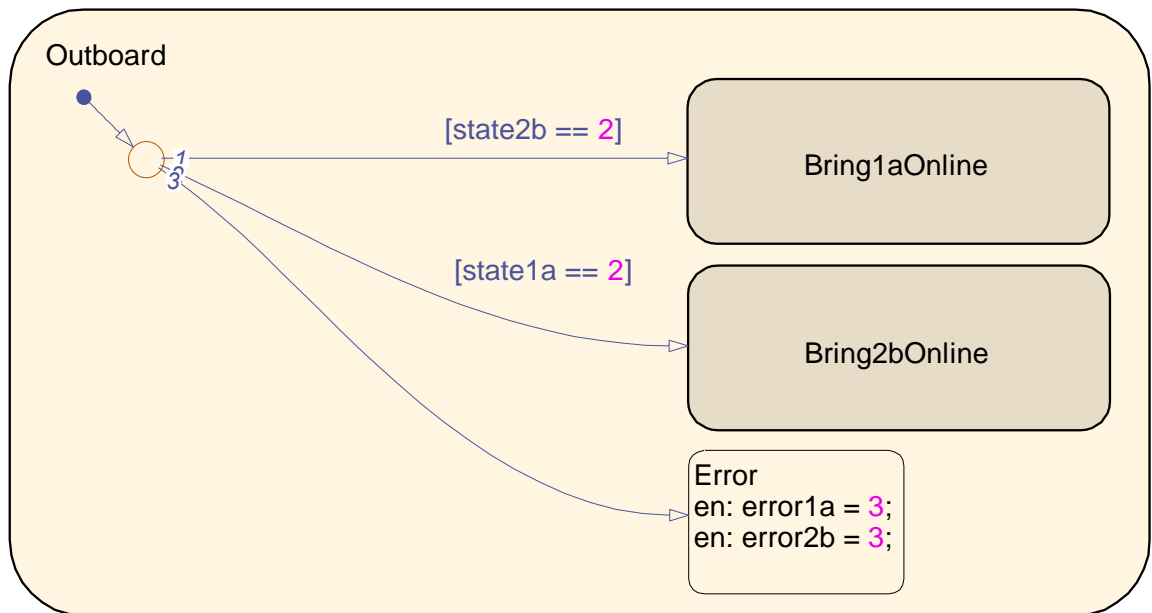


Figure C.5: Split Plant — Outboard Engines

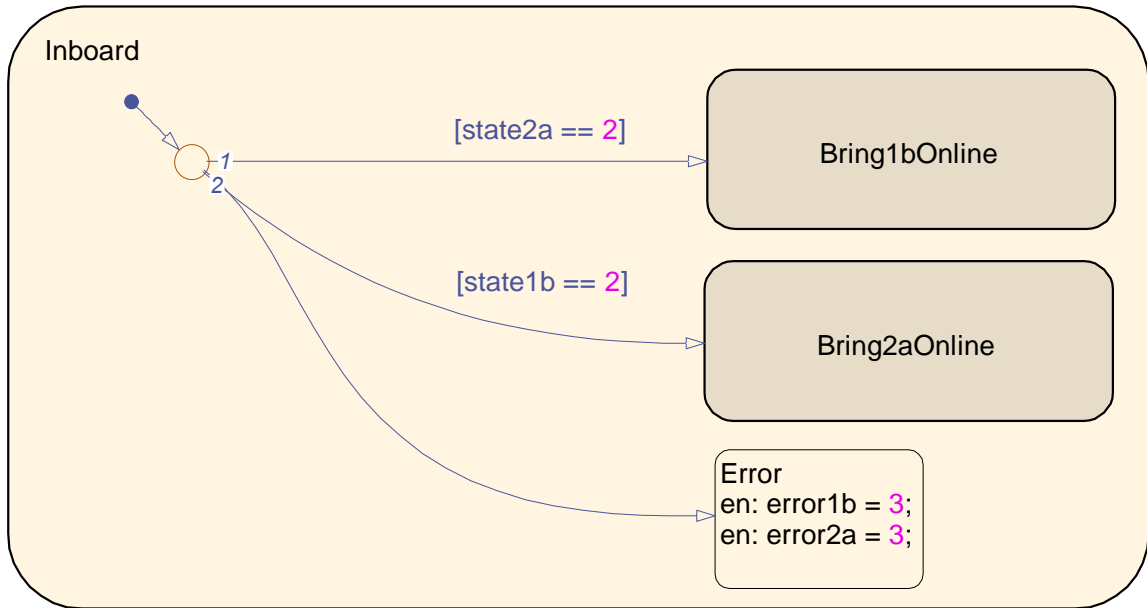


Figure C.6: Split Plant — Inboard Engines

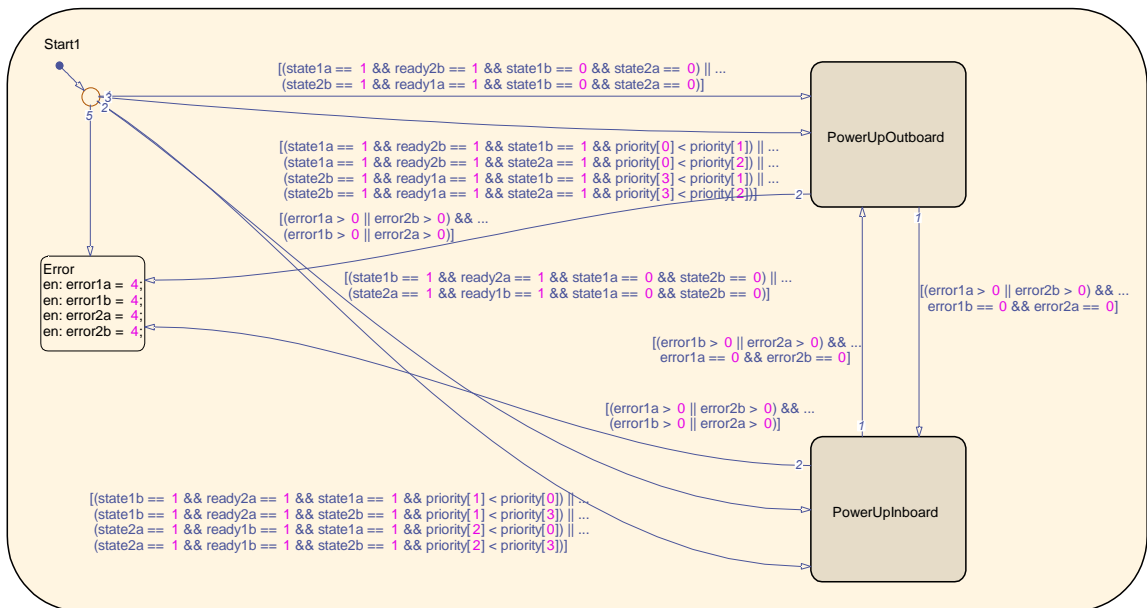


Figure C.7: Split Plant — Start 1 Engine

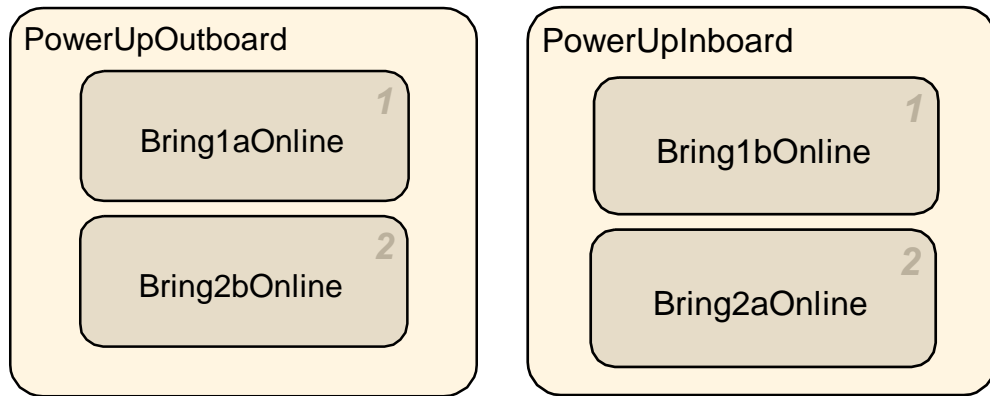


Figure C.8: Split Plant — Start Outboard/Inboard Engines

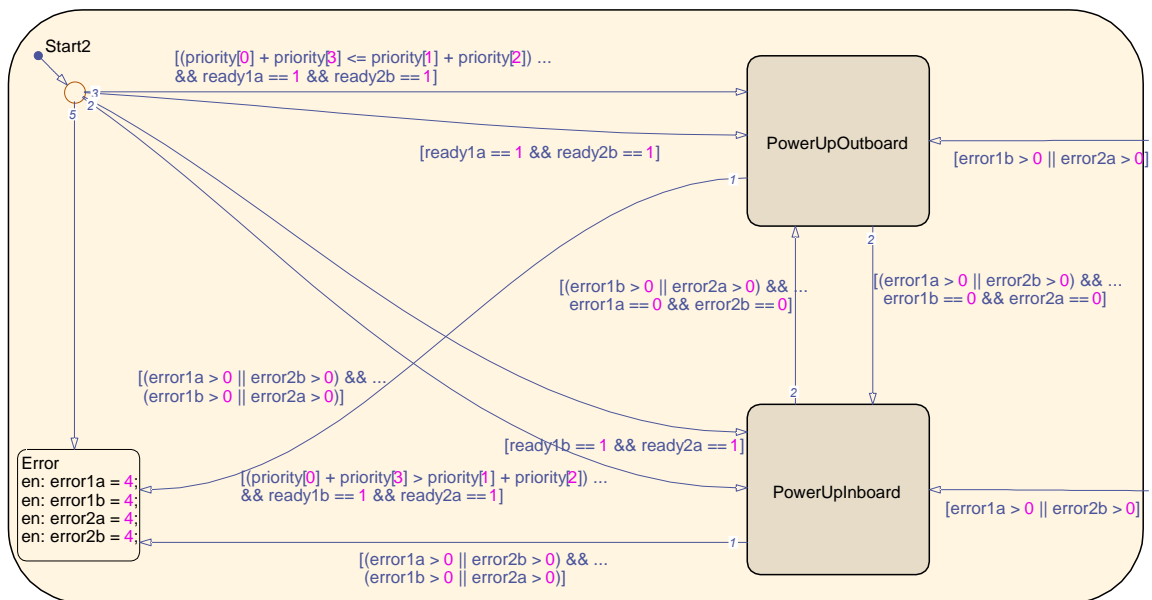


Figure C.9: Split Plant — Start 2 Engines

are commanded to start. The selection is made based on the engine priorities. If the sum of the outboard engine priorities is less than the sum of the inboard engine priorities, the outboard engines are started. Whereas if the converse is true and the sum of the inboard engine priorities is less than the output engine priorities, the inboard engines are started. The Split Plant error logic as previously discussed also applies when starting two engines.

When the Split Plant configuration is achieved by using either the inboard engines or outboard engines, the discrete control logic displayed in Figure C.4 transitions to the SplitMode state. Within this state, the mode is set to 2, confirming to the continuous control logic that the Split Plant Configuration has been achieved. The PowerDownSplit function as displayed in Figure C.10 is executed. The purpose of this function is to issue a shutdown command to the engines that are not

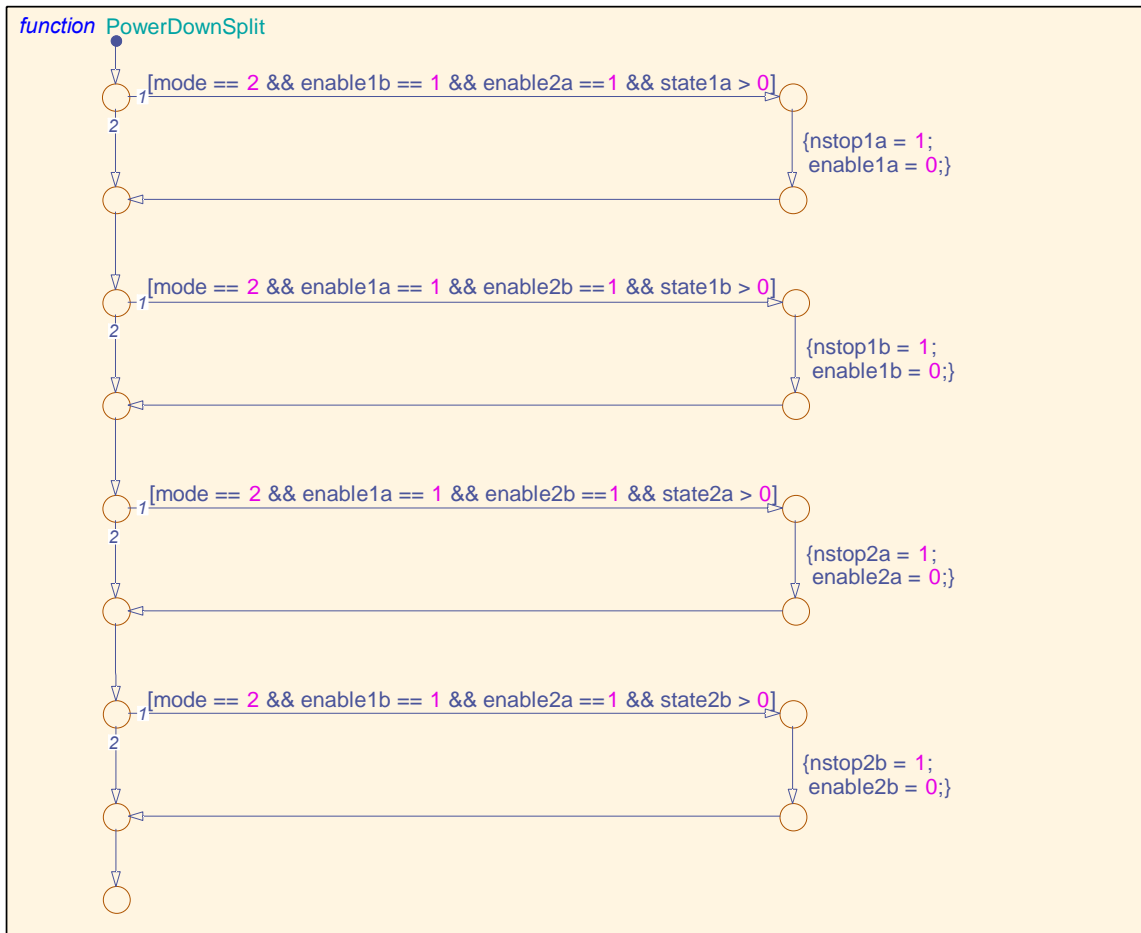


Figure C.10: Split Plant — Power Down Engines

required to be ONLINE based on the selected Split Plant configuration.

If the Split Plant configuration can not be achieved, the AllError state of Figure C.4 sets the error value of each of the engines to 4. The Error state of Figures C.7 and C.9 also sets the error value of each engine to 4. When the error value of an engine is set to 4, the attempt to align the propulsion plant to Split Plant is aborted and an attempt to align to Trailshaft commences.

C.5 Align to Trailshaft

When Trailshaft is commanded, the logic of Figure C.11 is executed. The logic is initialized by

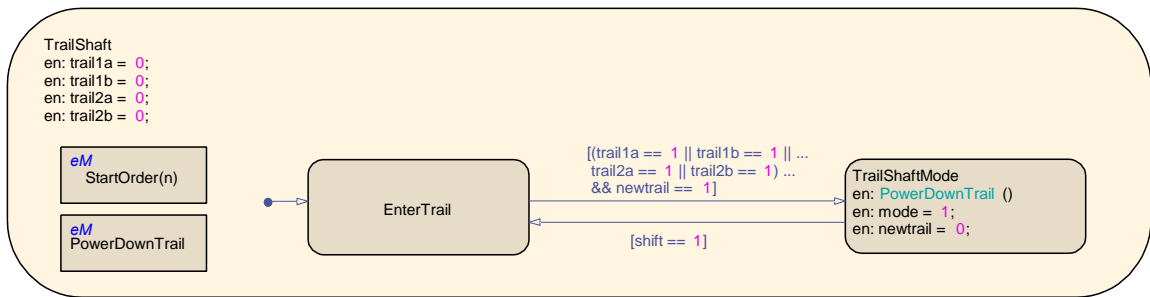


Figure C.11: Command Trailshaft

setting the trail value of each engine to 0. The EnterTrail state is then executed in accordance with the logic displayed in Figure C.12.

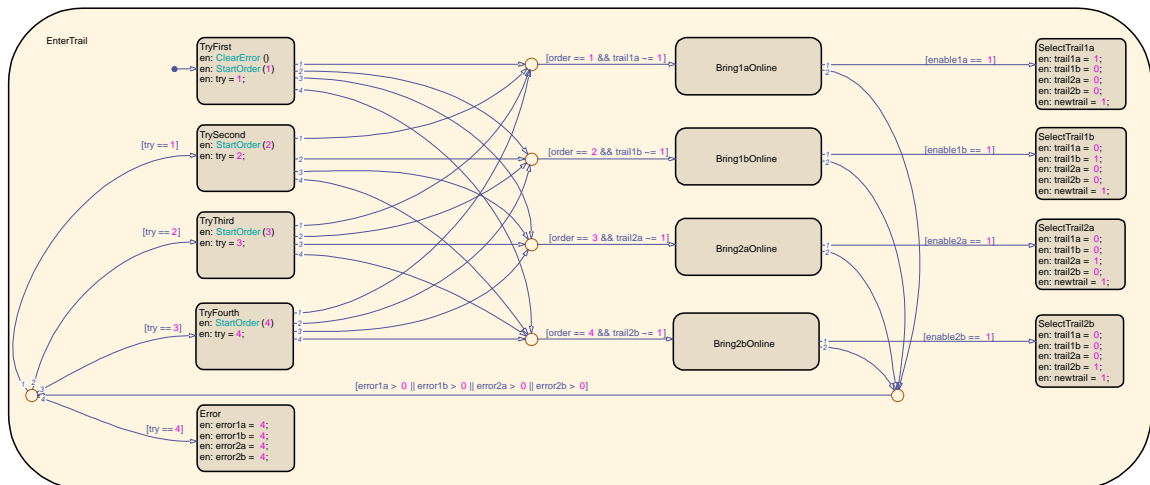
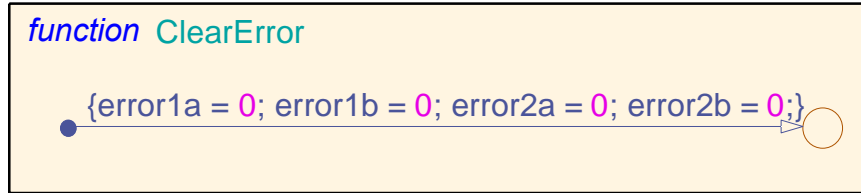


Figure C.12: Trailshaft — Start Engine

The EnterTrail logic is used to select and bring the selected engine ONLINE. For the initial attempt, all engine errors are cleared in accordance to the ClearError function displayed in Figure C.13. The StartOrder Embedded MATLAB function is also executed with an input of 1 (for 1st



```
function ClearError
    {error1a = 0; error1b = 0; error2a = 0; error2b = 0;}

```

The image shows a MATLAB function definition for `ClearError`. The function name is in blue. The code block contains a single line: `{error1a = 0; error1b = 0; error2a = 0; error2b = 0;}`. A blue dot is positioned at the start of the line, and a blue arrow points from the dot to the closing curly brace of the code block.

Figure C.13: Reset Engine Alarms/Errors

attempt). Within the StartOrder Embedded MATLAB function, engine priority is used to determine the first engine that should be started. Based on the engine priority and the engine Trailshaft status, the low priority engine is commanded to be brought ONLINE. If the selected (low priority) engine is brought ONLINE, its Trailshaft status is set to 1. However, if it does not come ONLINE, an attempted is made to start the engine with the 2nd lowest priority. If required, this process repeats a third and fourth time.

When the Trailshaft status is set to 1, the TrailShaftMode state of Figure C.11 is realized. When this state is reached, the PowerDownTrail Embedded MATLAB function is executed to shutdown any engine that is ON or ONLINE that is not required to be ONLINE. The mode is also set to 1, confirming to the continuous control logic that the Trailshaft mode has been achieved. If a shift command as discussed in Section 3.2.4 is issued, a transition from the TrailShaftMode state to the EnterTrail state results. A new engine will then be selected to enter Trailshaft mode and the engine currently ONLINE will be shutdown once the selected engine is ONLINE.

If no engines are successfully brought ONLINE, the Error state is reached. The error values of each engine is set to 4 and the attempt to align the propulsion plant to Trailshaft is aborted. The discrete control logic transitions to the OFF state displayed in Figure C.1.

C.6 Align to Off

When the discrete logic enters the OFF state, the AllStop Embedded MATLAB function is executed to shutdown all engines. The ClearError function displayed in Figure C.13 is also executed. This state is only active momentarily as a transition to the Initialize state is executed after only a few seconds.

C.7 Power Turbine Brake Commands

Separate from the ReConfigure_Plant Stateflow diagram is the discrete control logic used to engage and disengage the power turbine brake. This logic is displayed in Figure C.14. The brake initial-

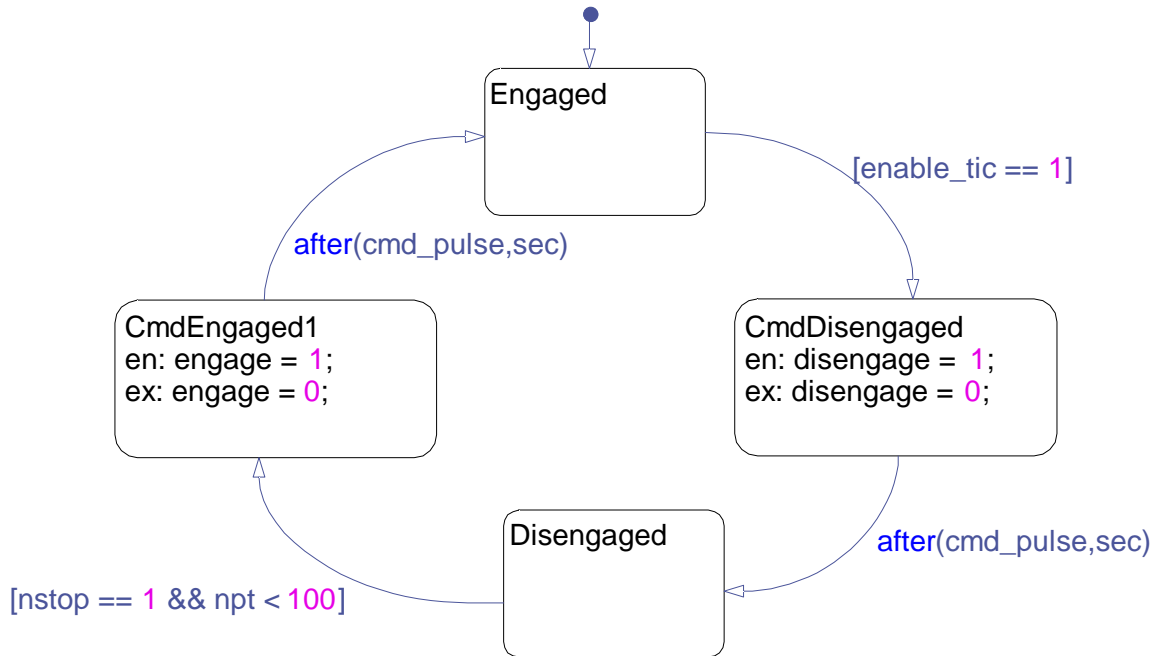


Figure C.14: Power Turbine Brake Logic

izes to “engaged” to lock the engine’s power turbine. When the engine’s throttle input command (*enable_tic*) is set to true, the power turbine brake is commanded to disengage. The power turbine brake remains disengaged until a normal stop (*nstop*) command is issued for the engine and the power turbine speed (*npt*) drops below 100 rpm. The brake is then commanded to engage.

Appendix D: Additional Test Scenarios

In addition to the simulated tests and results discussed in Section 4.2, more test scenarios were executed to verify some of the additional functionality included in the APC.

D.1 Emergency Stop Engine

To demonstrate the response of the APC to a failure condition, a test is run where one of the ONLINE engines is emergency stopped. Even though the emergency stop command was issued during this test, this would be equivalent to a propulsion engine failure and shutting down on its own.

During this test, the ship was initially commanded to a desired speed of 27 knots. When that command is issued, the inboard engines, Starboard Engine B (1b) and Port Engine A (2a), are started. Once the desired speed is achieved, Starboard Engine B (1b) is issued an emergency stop command at 350 seconds. When this occurs, its state immediately transitions from ONLINE to OFF as displayed in Figure D.1c.

In response to the shutdown, Port Engine A (2a), increases its throttle as displayed in Figure D.1b in an effort to maintain the desired ship speed of 27 knots. However, a single engine can not achieve this speed. Therefore, the propulsion system realigns from the now trailshaft mode to a split plant configuration. As displayed in Figure D.1a, the outboard engines are issued a start command while the remaining available inboard engine is commanded to stop. Once the outboard engines are ONLINE, the ship then achieves a steady state speed of 27 knots, as displayed in Figure D.1d.

D.2 Engines Unavailable

Because of faults within the propulsion systems, there are times when the APC can not achieve its control objectives. To demonstrate this scenario, two propulsion engines are set to unavailable to start. This unavailability could be due to ongoing maintenance of an engine or other existing faults that prevent the engine from starting. In this scenario, the ship is commanded to a desired

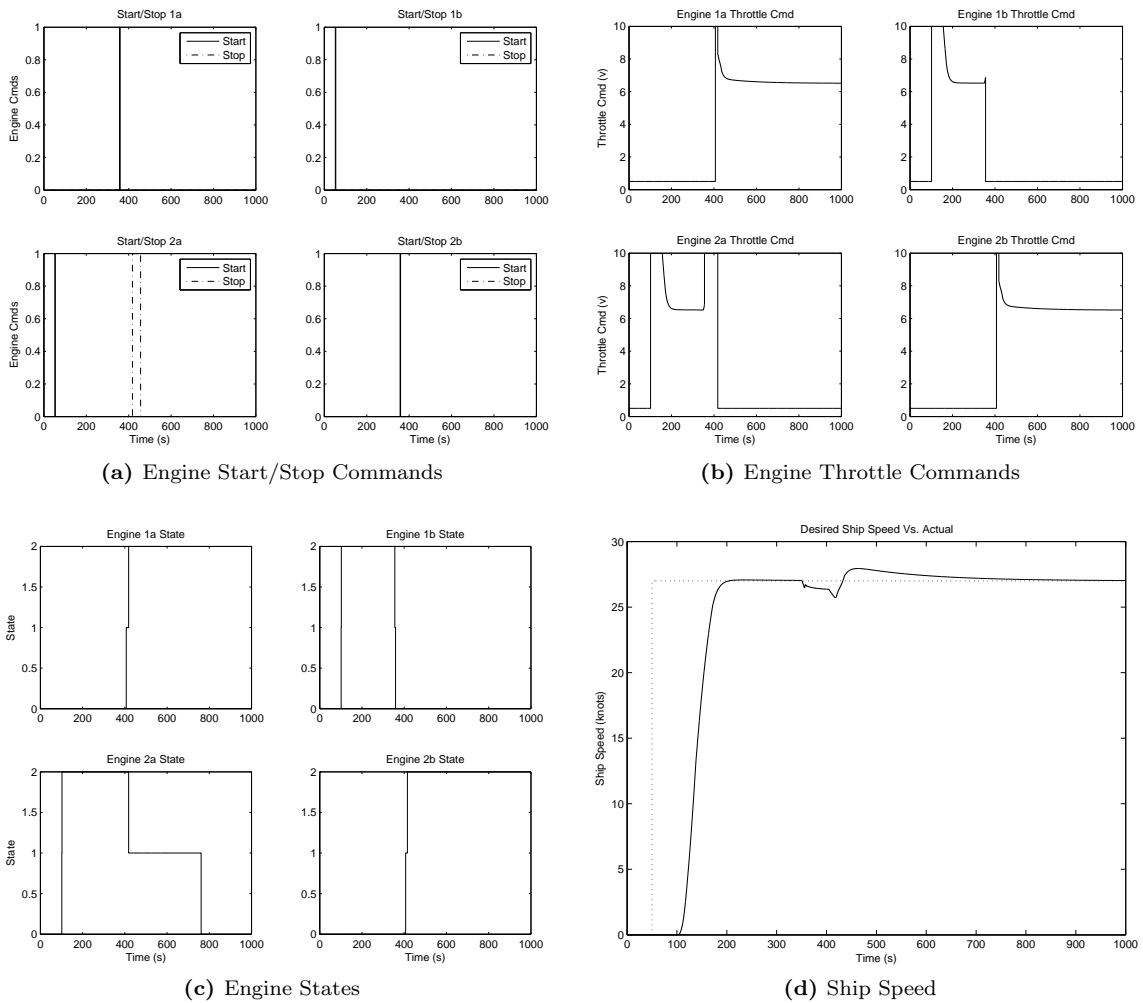


Figure D.1: Emergency Stop Engine Commands and Ship Response

speed of 34 knots. However, because the outboard engines are unavailable, only the inboard engines are commanded to start as displayed in Figure D.2a. As displayed in Figure D.2b, the throttle commands for these engines are set to the maximum. However, the ship is still not able to achieve the desired 34 knots. See Figure D.2d.

D.3 Shifting of Engines

An ideal objective in controlling a propulsion system is to maintain an equivalency of the operating hours for the propulsion engines. To prevent operating the same engine all of the time while in trailshaft mode, the APC is responsible for starting the appropriate engine based on the engine

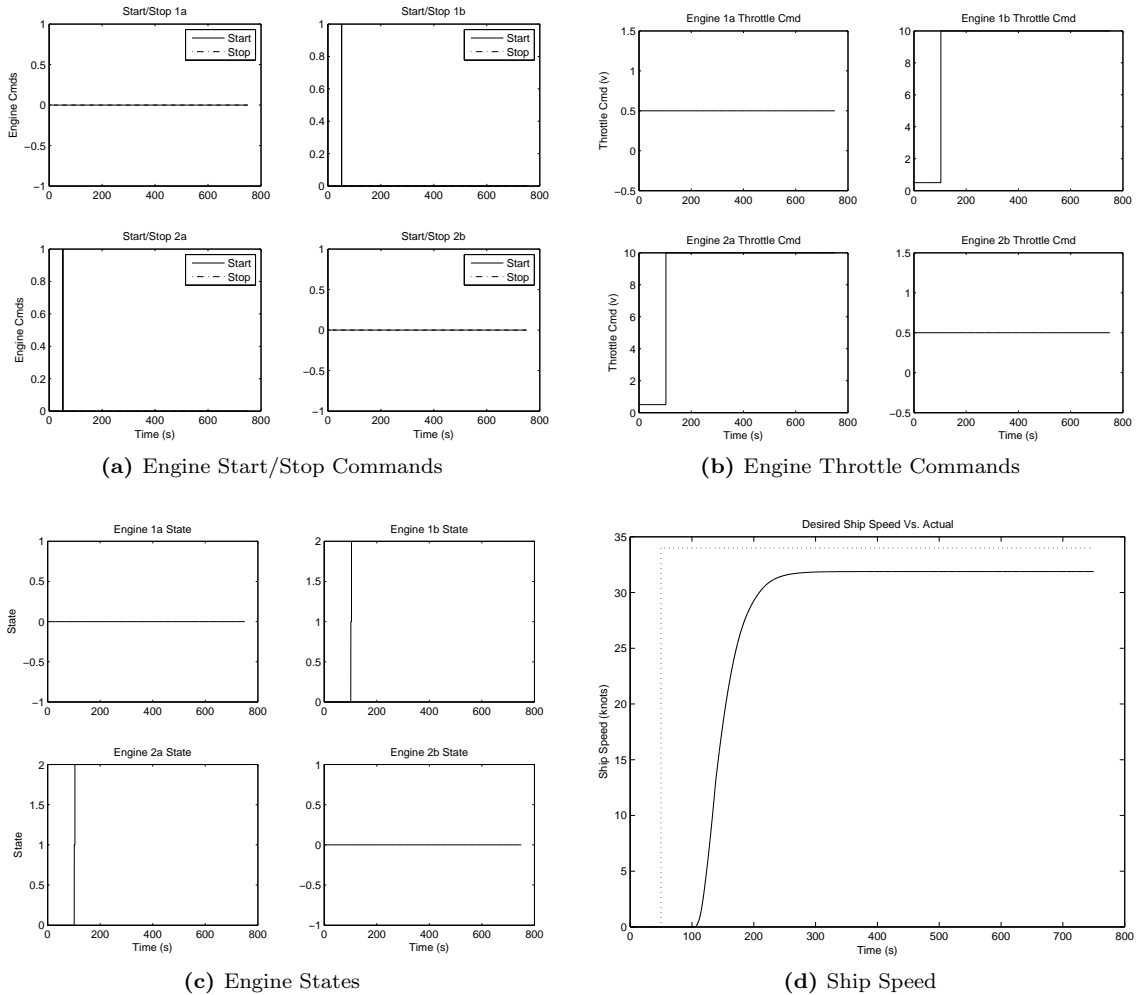


Figure D.2: Engine Unavailable Engine Commands and Ship Response

selection criteria. Additionally, the APC is responsible for shutting down the ONLINE engine after a user specified time and bringing the next available engine ONLINE to power the ship.

To demonstrate the shifting of the driving engines, a test scenario was executed where the desired ship speed was set to 15 knots. The user specified shift time was set to 350 seconds, where the powering engine was only to be ONLINE for 350 seconds. (This is not realistic, but was modified for testing purposes). From Figure D.3a, it can be observed that initially Starboard Engine A (1a) is commanded to power the ship in trailshaft mode. As displayed in Figure D.3d, the desired speed is achieved. After being ONLINE for 350 seconds, Starboard Engine B (1b) is brought ONLINE to power the ship. Each of the port engines are started in turn before Starboard Engine A is

brought ONLINE again. Throughout this process, the ship maintains its desired speed of 15 knots. Figure D.3b shows the associated engine throttle commands, while Figure D.3c displays the engine states.

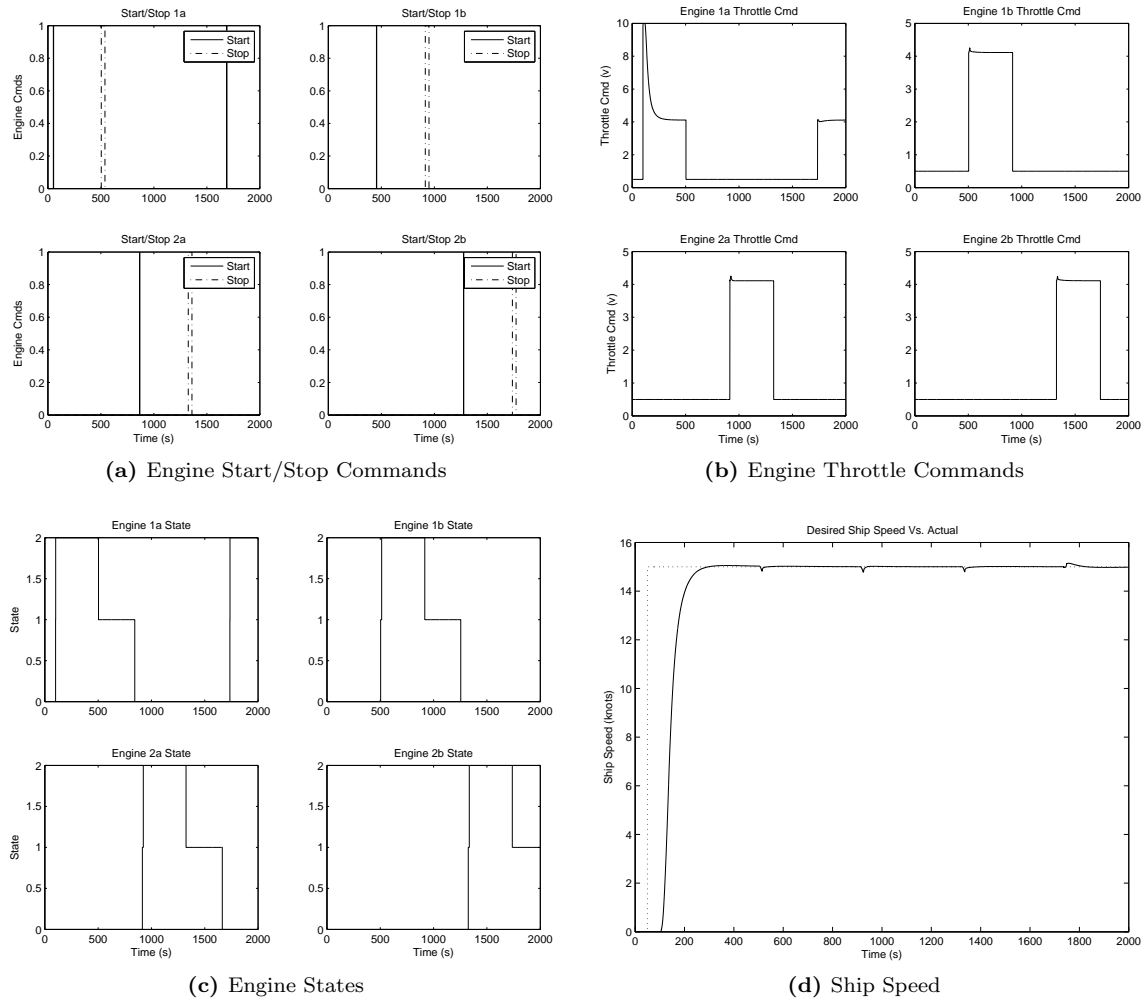


Figure D.3: Engine Shifts Engine Commands and Ship Response

D.4 Issue Manual Commands

The APC was design such that manual commands issued by the user would take precedence of those automatically issued. Even though the manual commands are of a higher priority, the APC is still responsible for monitoring the conditions of the ship. To demonstrate this functionality, the ship was initially set to achieve a desired speed of 15 knots. As displayed in Figure D.4a, the APC commanded Starboard Engine A (1a) to come ONLINE to power the ship. Once this speed was

achieved, manual commands were issued by the user.

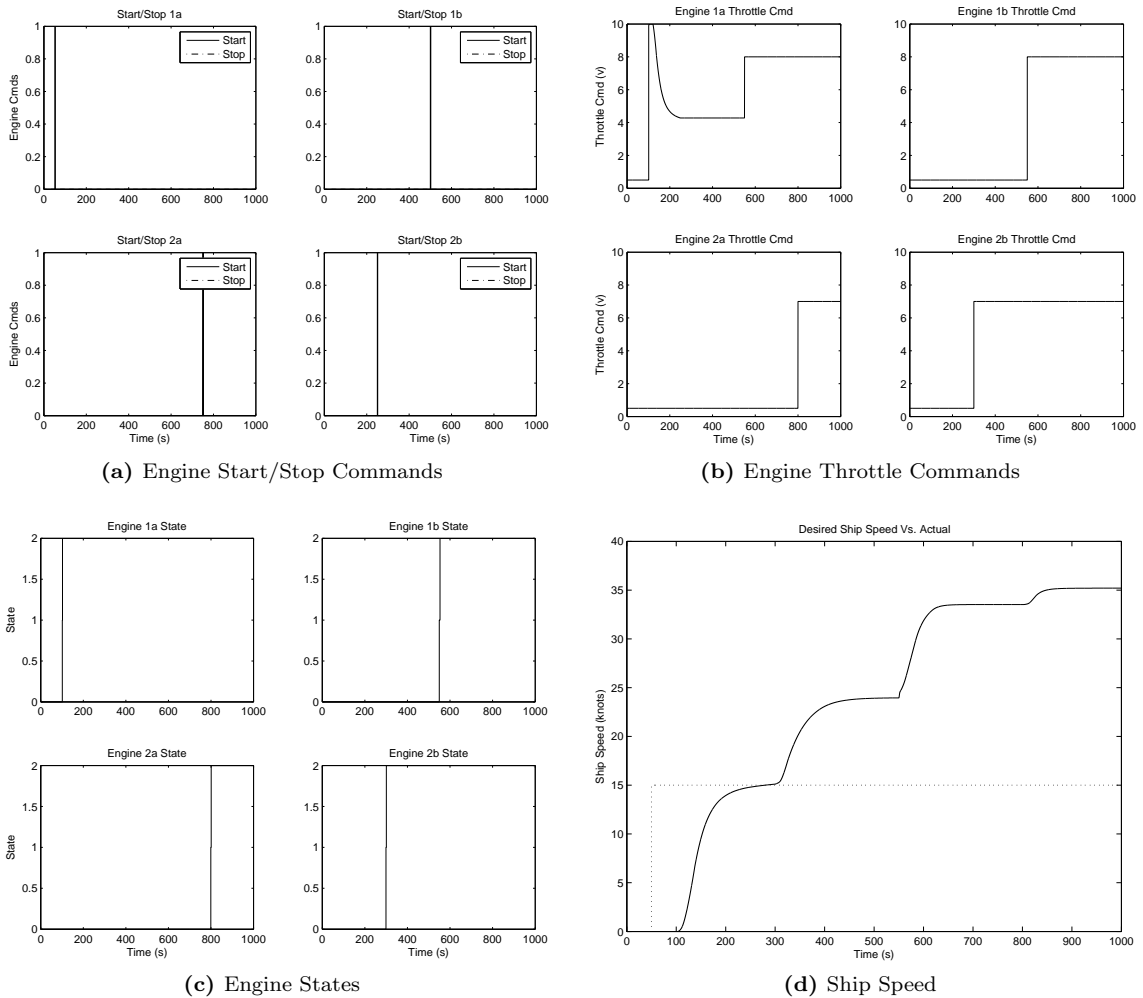


Figure D.4: Manual Engine Commands and Ship Response

The first manual command issued was to start Port Engine B (2b) at 250 seconds. At 300 seconds, its associated throttle command was set to 7. This is displayed in Figure D.4b. The ship speed begins to increase above that of the desired ship speed that was initially set. Later, Starboard Engine B (1b) was started and came ONLINE as displayed in Figure D.4c. When its associated throttle command was set, the throttle command for Starboard Engine A (1a) was also overridden from the APC set throttle command. Eventually, Port Engine A (2a) is started and its associated throttle command set. As the additional engines come ONLINE, the ship speed continues to increase as displayed in Figure D.4d.

Vita

Woodrow Clifton II

Education

Doctor of Philosophy, Mechanical Engineering, Drexel University, June 2014

Master of Science, Mechanical Engineering, North Carolina State University, June 1996

Bachelor of Science, Mechanical Engineering, North Carolina State University, December 1993

Professional Experience

Lockheed Martin Corporation, October 2005 - Present. Systems Engineer Staff.

Joint Strike Fighter Program, lead Production Asset Inspection Requirements team for Autonomous Logistics Informational System. Maritime Programs, lead modeling team for DDG-51 Arleigh Burke-class ships for Embedded Training, updated control algorithms for the LM2500 Gas Turbine Engines, lead modeling team for DDG-1000 Zumwalt Class Component and Cross Components efforts. Supported new business initiatives.

Merck & Company, Inc., April 2000 - September 2005. Senior Controls Engineer.

Sterile Operations Mechanical Services, Managed Control Systems staff, Manage programs for instrumentation and control systems required for manufacturing control systems, Managed DeltaV and Building Automation System, Coordinated maintenance of Lyophilizer cabinets and supporting systems.

BASF Corporation, December 1996 - April 2000. Engineer.

Tracked and monitored commitments and expenditures for large capital projects, Responsible for design implementation of SAP R/3 software into BASF's US sites, Implemented project for

the installation of an automated data collection system for paper coating testing laboratory,
 Maintained and evaluated energy conservation projects.

Graduate Work / Internships, NCSU College of Textiles, Raleigh, NC. Timken Research/The
 Timken Company, Canton, OH. NCSU Meteorology Department, Raleigh, NC. Hoechst Celanese
 Corporation, Charlotte, NC.

Major Publications

- Woodrow Clifton and Dave Reisteter. “Rapid development of the maritime advanced digital controller”. In Proceedings of the American Society of Naval Engineers Automation and Controls Symposium, Milwaukee, Wisconsin, 2010.
- Dan Petcovic, Woodrow Clifton and Sameh Mikhail. “Automated Watchstander Functions as a Means to Optimize Crew Size”. In Proceedings of the American Society of Naval Engineers Automation and Controls Symposium, Milwaukee, Wisconsin, 2010.
- Woodrow Clifton and Patricia Husband. “Embedded Training Plant Simulation for a Modernized Machinery Control System”. In Proceedings of the 14th International Ship Control Systems Symposium, Ottawa, Canada, 2009.
- Joey A. Duvall, Lamar Davidson, and Woodrow Clifton. “A real-time hardware-in-the-loop simulation environment for a ship propulsion system”. In Proceedings of the American Society of Naval Engineers Automation and Controls Symposium, Biloxi, Mississippi, 2007.
- Jeffrey W. Eischen and Woodrow Clifton. “Optimum Manipulation Strategies for Limp Fabric Materials”. In Proceedings of the Fifth Pan American Congress of Applied Mechanics Conference, San Juan, Puerto Rico, 1997.

