UC Irvine
UC Irvine Electronic Theses and Dissertations

Title
Model Predictive Control of Advanced Hybrid Powertrain Systems

Permalink
https://escholarship.org/uc/item/07q886k0

Author
Donikian, Vatche

Publication Date
2016

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA,

IRVINE

Model Predictive Control of Advanced Hybrid Powertrain Systems

THESIS

submitted in partial satisfaction of the requirements

for the degree of

MASTER OF SCIENCE

in Mechanical and Aerospace Engineering

by

Vatche Donikian

Thesis Committee:

Professor Gregory Washington, Chair

Professor Faryar Jabbari

Professor Giorgio Rizzoni, OSU

2016

© 2016 Vatche Donikian

Table of Contents

List of Figures

List of Tables

Acknowledgements

First and foremost I would like to thank my advisor, Dr. Washington, for all the guidance he has given me. Working under someone so passionate about his field of expertise motivates me to take my research a step further. Working under him has made me a better student, researcher, and human being. I would also like to thank Professor Rizzoni, for taking the time out of his busy schedule to advise me during my graduate studies, and for providing me with resources necessary, such as vehicle models from The Ohio State University, to keep my research moving forward. I thank Professor Jabbari as well for being there to advise me whenever needed.

In addition, I would like to thank the members of the ISSL lab, specifically Kyle Van Volkinburg and Joseph Bell, who have been present during the duration of my time in the lab. They have guided me, listened to and pitched in on my ideas, and pushed me to work hard every day, and for this I appreciate them. Robin Jeffers also made a big impact on my educational path and for that I thank her.

Lastly, I would like to thank my family and loved ones. My brother Vicken, my sister Nayiri, and my parents Roupen and Zovig, have given me unconditional support throughout my life and educational career. And to Nubia, I need to say thank you, for always being by my side, pushing me to never give up, and motivating me to stay focused. I share this accomplishment with you.

Abstract

Model Predictive Control of Advanced Hybrid Powertrain Systems

By

Vatche Donikian

Master of Science in Mechanical Engineering

University of California, Irvine, 2016

Professor Gregory Washington, Chair

In today's automotive industry, much of the focus has shifted to advanced vehicle propulsion systems. This is due to many factors, the main ones being climate change and the diminishing supply of gasoline. This thesis addresses some of the control objectives involved when coupling two power supplies as hybrid vehicles do. The main focus of the research is the use of Model Predictive Control to achieve both thermally efficient and fuel efficient algorithms to control the internal combustion engine in a hybrid powertrain. This is done with the use of two operational modes: Fuel Use Mode which limits the fuel consumption of the engine, and Efficiency Mode which maximizes the thermal efficiency of the engine. A mathematical overview of MPC is described to bring the reader into context. Next, this type of control is applied to a series-parallel hybrid electric vehicle. The formulation of the vehicle model in Simulink is discussed in detail. Then, a practical form of MPC is implemented, by using a torque vector to estimate the outputs of engine model, and minimizing the resulting cost function. Next, MPC parameter selection is discussed, which covers the choices for receding horizon length, cost function weights, and uncertainty correction parameters. After all the parameters have been chosen, simulations are run for the US06 and FUDS drive cycles, and the results are analyzed. The results show that Fuel Use Mode yields a higher miles-per-gallon (MPG) rating, and Efficiency Mode helps maintain the charge level of the batteries.

1 Introduction

According to the U.S. Department of Energy, every year vehicles release more than 1.5 billion tons of greenhouse gases into the air [1]. These gases, such as carbon dioxide, pollute the atmosphere and contribute heavily to global warming. Advanced engine management technologies have improved the efficiencies of internal combustion engines, but the engines still produce CO_2 and other greenhouse gases and the problem still remains. The automotive industry is currently shifting to alternative energy sources, mainly batteries and fuel cells, in order to curb the use of fossil fuels. In most automotive applications today, these energy sources are used in combination with a gasoline engine, to form a hybrid vehicle powertrain. As with any new addition of hardware or technology, the complexity of a vehicle's architecture increases with the addition of energy sources and propulsion systems. Figure 1 shows the most common architectures for a hybrid electric vehicle (HEV):

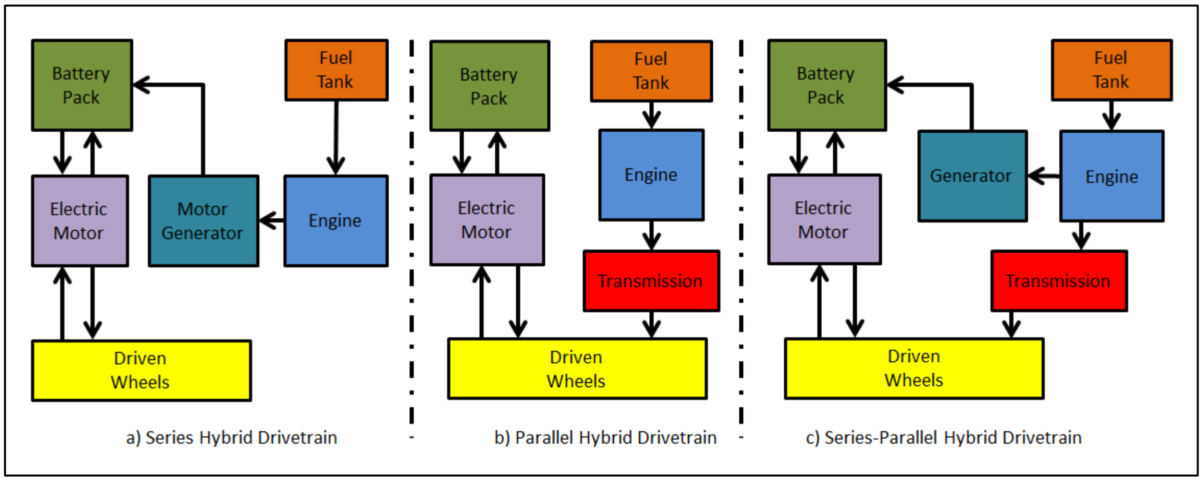


Figure 1: Common Hybrid Architectures

In a series configuration (part a), the engine's sole use is to create energy for the electric motor. The electric machine is the only component that propels the vehicle. Power created by the engine is converted to electricity with a generator, typically an electric motor with current running the opposite direction. In a parallel configuration (part b), both the engine and electric motor have the capability to propel the vehicle. This provides the ability to use either propulsion system or a combination of the two. However, in order to also have the capability to charge the batteries, a series-parallel configuration must be used (part c). This allows the engine to act as either a generator or a propulsion device, depending on the algorithm decided ([2], [3], [4]).

Due to the different architectures possible for a hybrid electric vehicle, there are different operating modes to consider. One of these modes is the charge-depleting mode. This is when the engine is shut off and the vehicle operates in an all-electric mode. The power is supplied purely from the batteries. Charge-depleting mode is used in situations when the gasoline engine operates inefficiently, such as starting from stop or in city traffic. The electric motor has instantaneous torque, so it is more than capable of propelling the vehicle on its own, without wasting gasoline. The other mode commonly used is charge-sustaining mode. This mode is when the engine is used to keep the battery state of charge (SOC) at a reasonable operating level. The engine is kept at its optimal operating point, charging the battery in the process. Any excess power from the engine can be used in conjunction with the driving motor to power the vehicle. A lot of research has been done to determine mode selection strategies, and there are many combinations depending on what your main goal is, such as lowest fuel consumption or highest thermal efficiency.

The powertrain configuration used in this thesis is a series-parallel hybrid configuration where the engine is attached to one axle, and the electric motor to the other. This type of configuration is called a Through-the-Road hybrid (TTR), since the torque coupling and battery charging happen by power being translated through the road, as described in [5], [6], [7], [8], [9] and [10]. A configuration of this architecture is seen in Figure 2:

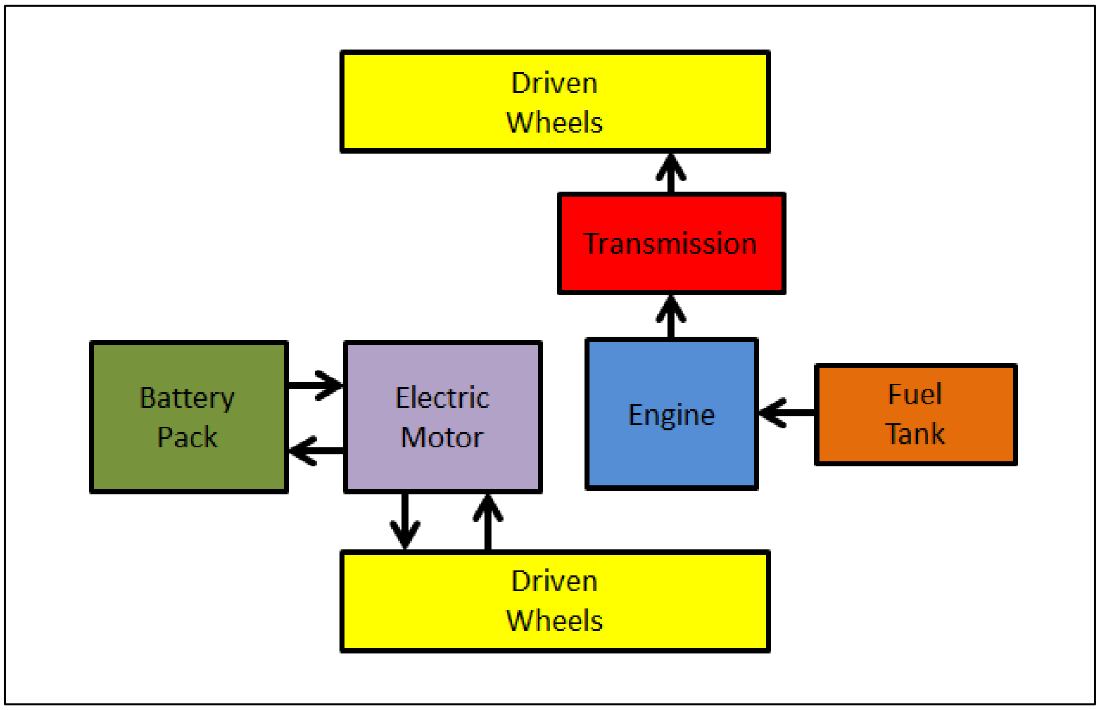


Figure 2: Through-The-Road Series-Parallel Hybrid Configuration

The benefits to a TTR hybrid, also known as a separated-axle hybrid, is the simpler design and cost friendliness [8]. There is no need to add additional components to couple the electric motor and internal combustion engine or concern oneself with matching the gear ratios of the propulsion devices; each power plant can have its own gear reductions, as in this thesis.

The work in this thesis focuses on the control of hybrid powertrains. This consists of supplying the supervisory controller with the correct torque and speed commands to operate the vehicle while meeting different efficiency or emissions requirements. The control methodology used in this thesis is Model Predictive Control (MPC), which uses optimization techniques with a prediction algorithm to determine the optimal control input. This control is discussed in Chapter

3. MPC is used due to its prediction capabilities and robustness. To proceed with the analysis when incorporating MPC in an automotive powertrain, performance requirements must be set. These requirements are discussed in Section 4.1, and as with many hybrid control algorithms, deal with engine efficiency and fuel consumption. Next, a model of a hybrid vehicle system must be constructed, which is explained in Section 4.2. Finally, control can be applied to make the system follow desired responses, which is covered in Chapters 5. Chapter 6 covers a possible implementation of the designed controller on-board a hybrid vehicle.

2 Literature Review

Many different control strategies using optimization techniques have been used to solve the torque-split problem involving hybrid vehicles [11]. These can consist of optimal control algorithms like single point optimization, where a cost function is minimized for only the current decision step [12]. Model Predictive Control builds on this, by optimizing over the current and upcoming steps. References [13] and [14] were consulted for the mathematics of Model Predictive Control. These were helpful in understanding the MPC process, including the concept of the receding horizon and cost function optimization. References [15]–[24] deal strictly with MPC for automotive applications. These papers were thoroughly read to explore the field, to see what has been done and how MPC was implemented, specifically looking at the prediction models used and cost functions minimized. References [16] and [17] were general overviews of the use of MPC in automotive controls. Hrovat et al. discussed the use of MPC for the automotive Idle Speed Control problem [16]. Chen et al. wrote an informative paper to show examples of MPC in the automotive field, discussing using MPC to model driver behavior [17]. Reference [22] uses a stochastic MPC approach to optimize a series hybrid configuration. The use of MPC was also shown for other hybrid configurations as well, such as fuel cell hybrids. Reference [19] used MPC to regulate air, fuel, and exhaust flows for a fuel cell hybrid vehicle. Linearization of a nonlinear powertrain model was used to predict system behavior. Beck et al. used MPC to improve the transition from electric to hybrid propulsion [20]. The model used for prediction was a lumped-inertial model. Another form of MPC use is with the sharing of power demand. Reference [18] is a paper that used MPC for a hybrid powertrain with emphasis on energy management, to reduce the changes in battery state of charge by regulating energy consumption from two separate energy storages. The model used in the paper was a state-space model of the battery-supercapacitor system. The emphasis in the current thesis is on the engine's fuel consumption and thermal efficiency which have been proven to improve fuel economy and battery use, as shown in the following sources. Mayr et al. studied the effects of MPC on operating the engine optimally while restricting the change in battery state of charge [24]. This process was done for cyclic operations such as bus routes. The cost was minimized over the complete cycle, which differs from the implementation used in the current research. Reference [15] used a mean-value model of throttle to torque dynamics to study the use of MPC to limit fuel consumption of an internal combustion engine without sacrificing significant performance. This paper was not studying a hybrid powertrain, but solely an internal combustion engine. Reference [21] used two separate model predictive controllers, one for each power plant, to control the torque split of the system. The simulations run for [21] were for a step throttle and a step load torque, thus the importance was placed on short term response. Yan et al. took a similar approach, by incorporating transient dynamics into the model and analyzing the effects of engine start-up on the fuel consumption.

Due to the availability of only engine data, the research in this thesis made use of a quasi-static model, and thus transient dynamics were not explicitly included, differing from the references described above. These references helped shape the direction of this thesis, and helped give an idea of what can be done with MPC and the ability to implement this type of control with many different models.

To help with creating a simulator model, references [25], [26], [27], and [28] were consulted. Reference [27] is a thesis discussing the modeling and simulation process for a hybrid electric vehicle. Areas of emphasis were architecture selection, power and component specification selection, and model formulation. The model was used to determine energy levels consumed in testing and to determine the required size and capacities of the components. This reference was consulted to select the electric motor and internal combustion engine for the current research in this thesis.

Reference [26] was used to formulate the references to test for this thesis. The work is a thesis itself that covers the implementation of a Fuzzy-Logic controller to regulate a hybrid electric powertrain. The operational modes used in the current thesis were derived from the modes of operation in [26].

In formulating a vehicle model designed for control implementation, [25] and [28] were very useful. Reference [25] is a paper dealing with the modeling of a hybrid electric vehicle, and the simulations involved in verifying the model. This paper was used in conjunction with the university's project competition, to build a plug-in hybrid electric vehicle. Reference [28] discussed the modeling of a plug-in hybrid electric vehicle, and analyzed the performance of different control strategies. One of those control strategies was the implantation of an optimal controller. The similarities between optimal control and Model Predictive Control were helpful in the practical implementation of MPC in this thesis.

3 Model Predictive Control

Model Predictive Control (MPC) was used in this research, mainly for its abilities to adapt to changes in the system. MPC is known for its accuracy, due to the prediction algorithm incorporated, along with the capability of disturbance rejection. Advanced control algorithms have been implemented in powertrains for some time now; the use of MPC pushes this boundary further, and will be important in the autonomy of vehicles in the near future, due to the ability to adjust the control law mid-execution. In this section, a basic explanation of MPC is presented to highlight the mathematics behind the controller. In the later sections, MPC will be applied to various powertrain configurations.

3.1 Introduction to Model Predictive Control

MPC is a predictive control law in which the control input of the current step is calculated using a prediction of model behavior over a set future horizon. This type of control reflects human behavior and how we make decisions based on what we think the outcome will be. An example is deciding when to brake based on an obstacle in the road ahead of you, or any time when a decision has to be made before an event happens. MPC tries to emulate this human thinking in anticipating changes and adjusting the control accordingly. The idea of predicting a set distance into the future is called the receding horizon concept. The algorithm is designed to continuously update information and replace past steps. For each iteration, the model looks one step further in the future, keeping the horizon at its fixed and predetermined length. As with any predictive control law, MPC must have an accurate model of the system, in order to determine future system behavior. One of the key aspects of MPC is the ability to reject disturbances and correct for uncertainties; thus, the prediction model does not need to be perfect. Another useful component of MPC is that the models can be linear or nonlinear, which is not the case for many controllers. Once the model predictions are calculated, the resulting error between the reference and prediction is minimized using a cost function and constraints. From here, the control input is determined. The process repeats itself each step. A block diagram representation of an MPC system is shown below:

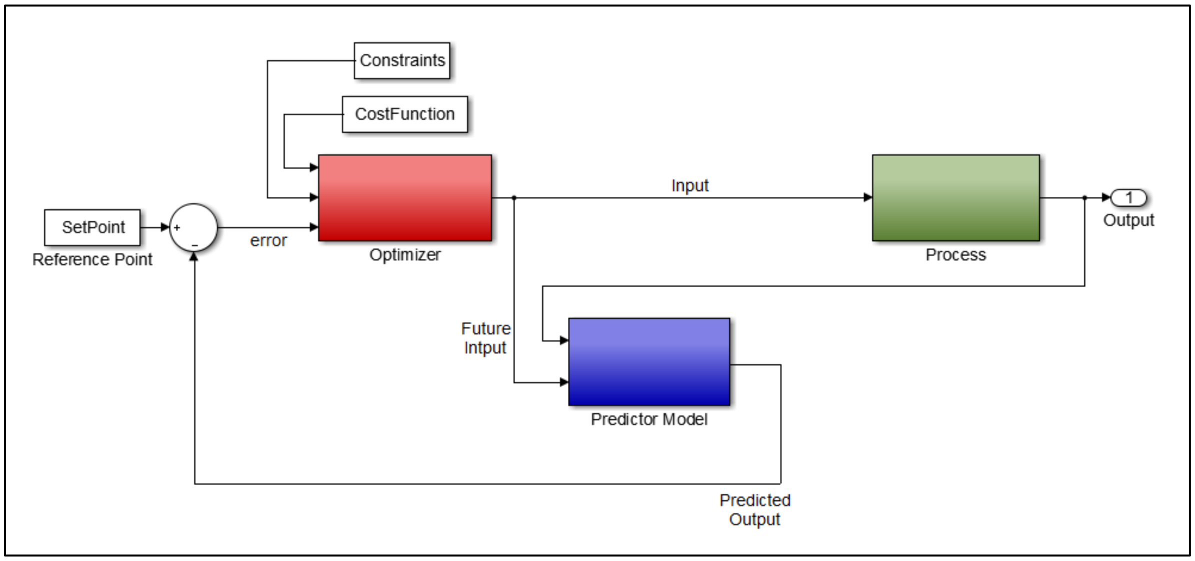


Figure 3: Schematic of MPC

3.2 Mathematical Analysis of MPC

In order to implement MPC, a discrete time model is needed. This can either be addressed by creating the system equations in the discrete z-domain, or by sampling and reconstruction around a time dependent model. Most dynamic systems are written in the continuous time frame, so the latter of the two methods will be discussed in this thesis. Consider a state space model of the form

$$\begin{aligned} \dot{x}(t) &= A_C x(t) + B_C u(t) \\ y(t) &= C_C x(t) + D_C u(t) \end{aligned} \tag{3.1}$$

where $A_c \in \mathbb{R}^{nxn}, B_c \in \mathbb{R}^{nxm}, C_c \in \mathbb{R}^{pxn},$ and $D_c \in \mathbb{R}^{pxm}$ are the system matrices in continuous time, corresponding to states $x \in \mathbb{R}^n$ and inputs $u \in \mathbb{R}^m$. This system can be sampled with a sampling time T to become discretized. The following system is created:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k + Du_k \end{aligned} \tag{3.2}$$

The new matrices, assuming A_C is stable, are

$$A = e^{A_C T}, B = (e^{A_C T} - I)A_C^{-1}B_C, C = C_C, D = D_C \tag{3.3}$$

with I specifying the identity matrix. These discretized matrices are found using the solution to (3.1) and the property of the matrix exponential:

$$e^{AT} = I + AT + \frac{AT^2}{2!} + \frac{AT^3}{3!} + \cdots \sum_{k=0}^{\infty} \frac{AT^k}{k!} \tag{3.4}$$

In the case that there is a disturbance in the system, system (3.2) becomes:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + d_k \\ y_k &= Cx_k + Du_k \end{aligned} \tag{3.5}$$

In order to convert from the discrete domain back to the continuous domain, a digital to analog conversion (DAC) is used such as the Zero-Order-Hold (ZOH), which holds the beginning value for the complete time step.

3.2.1 Prediction Model

A discrete system of the form (3.2) or (3.5) is used in MPC to construct the prediction model. The system equations will be manipulated over a certain receding horizon, which is N steps ahead [14]. In order to express these future steps, the forward looking matrix representations must be defined. For any vector v:

$$\underset{\rightarrow}{v}_k = \begin{bmatrix} v_{k+1} \\ v_{k+2} \\ \vdots \\ v_{k+N} \end{bmatrix}, \underset{\rightarrow}{v}_{k-1} = \begin{bmatrix} v_k \\ v_{k+1} \\ \vdots \\ v_{k+N-1} \end{bmatrix} \qquad [14](3.6)$$

Starting with the system defined in equation (3.5), the first three iterations are shown below:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + d_k & y_k &= Cx_k \\ x_{k+2} &= Ax_{k+1} + Bu_{k+1} + d_{k+1} & y_{k+1} &= Cx_{k+1} \\ x_{k+3} &= Ax_{k+2} + Bu_{k+2} + d_{k+2} & y_{k+2} &= Cx_{k+2} \end{aligned} \qquad (3.7)$$

The direct relation of the input to output, D, has been set to zero for the derivation. If one repeats this process N times and substitutes in the previous step, the Nth prediction will be

$$\begin{aligned} x_{k+N} &= A^N x_k + A^{N-1}Bu_k + A^{N-2}Bu_{k+1} + \cdots + Bu_{k+N-1} + A^{N-1}d_k + A^{N-2}d_{k+1} + \cdots + d_{k+N-1} \\ y_{k+N} &= C[A^N x_k + A^{N-1}Bu_k + A^{N-2}Bu_{k+1} + \cdots + Bu_{k+N-1} + A^{N-1}d_k + A^{N-2}d_{k+1} + \cdots + d_{k+N-1}] \end{aligned} \qquad (3.8)$$

Now, combining each step into one vector-matrix system, the system prediction over the complete horizon can be displayed as the following:

$$\underbrace{\begin{bmatrix} x_{k+1} \\ x_{k+2} \\ x_{k+2} \\ \vdots \\ x_{k+N} \end{bmatrix}}_{\underset{\rightarrow}{x}_k} = \underbrace{\begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^N \end{bmatrix}}_{P_{xx}} x_k + \underbrace{\begin{bmatrix} B & 0 & 0 & \cdots \\ AB & B & 0 & \cdots \\ A^2B & AB & B & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \cdots \end{bmatrix}}_{H_x} \underset{\rightarrow}{u}_{k-1} + \underbrace{\begin{bmatrix} I & 0 & 0 & \cdots \\ A & I & 0 & \cdots \\ A^2 & A & I & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ A^{N-1} & A^{N-2} & A^{N-3} & \cdots \end{bmatrix}}_{L_x} \underset{\rightarrow}{d}_{k-1} \qquad (3.9)$$

$$\underbrace{\begin{bmatrix} y_{k+1} \\ y_{k+2} \\ y_{k+2} \\ \vdots \\ y_{k+N} \end{bmatrix}}_{\underset{\rightarrow}{y}_k} = \underbrace{\begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^N \end{bmatrix}}_{P} x_k + \underbrace{\begin{bmatrix} CB & 0 & 0 & \cdots \\ CAB & CB & 0 & \cdots \\ CA^2B & CAB & CB & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ CA^{N-1}B & CA^{N-2}B & CA^{N-3}B & \cdots \end{bmatrix}}_{H} \underset{\rightarrow}{u}_k + \underbrace{\begin{bmatrix} C & 0 & 0 & \cdots \\ CA & C & 0 & \cdots \\ CA^2 & CA & C & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ CA^{N-1} & CA^{N-2} & CA^{N-3} & \cdots \end{bmatrix}}_{L} \underset{\rightarrow}{d}_k \tag{3.10}$$

This can be written in a more compact format:

$$\begin{gathered} \underset{\rightarrow}{x}_k = P_{xx} x_k + H_x \underset{\rightarrow}{u}_{k-1} + L_x \underset{\rightarrow}{d}_{k-1} \\ \underset{\rightarrow}{y}_k = P x_k + H \underset{\rightarrow}{u}_k + L \underset{\rightarrow}{d}_k \end{gathered} \tag{3.11}$$

The forward looking vector $\underset{\rightarrow}{u}_k$ for the control input will be found in the optimization step. The vector $\underset{\rightarrow}{d}_k$ is formed of the future disturbances; this can either be known disturbances or can be found by approximating the future disturbances by the previous step's error between the predicted and actual state [13]. The disturbance vector becomes:

$$\underset{\rightarrow}{d}_k = \begin{bmatrix} d_{k-1} & d_{k-1} & \cdots & d_{k-1} \end{bmatrix} \tag{3.12}$$

This completes the prediction process of MPC.

3.2.2 Optimization

With the predictions now available, the process can shift to finding the correct control input for the system. This is done by minimizing a cost function in order to obtain the best control input to drive the system towards the desired trajectory in the most optimal fashion. Since the goal of any control is to drive the system to a reference, a component of the cost function must be the error between the reference trajectory and the predicted system behavior. One must also put a limitation on the amount of control in each step, in order to not over-compensate for error in the system. The resulting cost function to be used is

$$J = \left\| \underset{\rightarrow}{R}_k - \underset{\rightarrow}{y}_k \right\|^2 + \lambda \left\| \underset{\rightarrow}{u}_k \right\|^2 \qquad (3.13)$$

The Euclidean norm is generally used in this algorithm. The resulting minimization problem becomes

$$\min_{\underset{\rightarrow}{u}_k} \left\| \underset{\rightarrow}{R}_k - \underset{\rightarrow}{y}_k \right\|^2 + \lambda \left\| \underset{\rightarrow}{u}_k \right\|^2 \qquad (3.14)$$

The predicted system output is substituted into (3.14) which gives

$$\min_{\underset{\rightarrow}{u}_k} \left\| \underset{\rightarrow}{R}_k - Px_k - H\underset{\rightarrow}{u}_k - Ld_k \right\|^2 + \lambda \left\| \underset{\rightarrow}{u}_k \right\|^2 \qquad (3.15)$$

The optimization of this cost function is achieved by taking the derivative with respect to the minimization variable $\underset{\rightarrow}{u}_k$ and setting it equal to zero [13]. Recalling the derivative of a 2-norm:

$$\frac{\partial}{\partial v}(\|w\|^2) = \frac{\partial}{\partial v}(w^T w) = 2(\frac{\partial w}{\partial v})^T w \qquad (3.16)$$

We can show that the derivative of the cost function is

$$\frac{\partial J}{\partial \underset{\rightarrow}{u}_k} = 2(-H)^T(\underset{\rightarrow}{R}_k - Px_k - H\underset{\rightarrow}{u}_k - Ld_k) + 2(\lambda)\underset{\rightarrow}{u}_k \qquad (3.17)$$

Setting this equal to zero and solving for $\underset{\rightarrow}{u}_k$, the optimal control input becomes

$$\underset{\rightarrow}{u}_k = (H^T H + \lambda I)^{-1}[H^T \underset{\rightarrow}{R} - H^T Px_k - H^T L\underset{\rightarrow}{d}] \qquad (3.18)$$

This optimization procedure has calculated the control input for the whole horizon; however only the first value is needed, since for each time step, the system will recalculate the control input. The input sent to the system plant is therefore

$$u_k = \begin{bmatrix} I & 0 & \cdots\cdots & 0 \end{bmatrix} \underset{\rightarrow}{u}_k \tag{3.19}$$

This value is sent to the process block of Figure 3. From here, the actual output is determined. In the following sections, this control algorithm will be applied to automotive powertrain systems to improve fuel efficiency and emissions of hybrid vehicles.

4 Hybrid Electric Vehicle Control Using MPC

There are many different classes of hybrid electric vehicles. This chapter will focus on the control of a through-the-road parallel hybrid vehicle powertrain, due to its ability to manage power in different configurations, such as electric motor propulsion, engine propulsion, a combination of the two, and battery charging. In the automotive industry, a parallel hybrid vehicle has a powertrain in which the internal combustion engine and the electric motor can each operate alone or in unison. The two propulsion devices may be coupled together to provide the torque to the driven axle, or a set of clutches can disengage one of the devices from the axles.

4.1 Operational Modes

The main concerns with internal combustion engines are inefficiencies and fuel consumption. In order to tackle these obstacles, the control algorithm implemented in this research aims to either maximize engine efficiency or limit fuel consumption [26]. These two methods are described in the sections below, followed by a complete description of the MPC algorithm applied to the system.

4.1.1 Efficiency Mode

Efficiency Mode is used for when the desired operating point of the engine is the point of maximum thermal efficiency. The goal in this mode is not necessarily fuel economy, but minimum power loss of the engine. Efficiency Mode is implemented by first calculating the required torque and speed of the ICE; then holding the engine speed constant, the optimal torque can be chosen for that engine speed. This request is sent to the controller which adjusts the input to the ICE. If the optimal torque produced by the engine is greater than the required torque, the excess will be sent to charge the batteries. Conversely, if the optimal torque produced by the engine is less than the required torque, the electric motor will provide the necessary torque to keep the vehicle at the desired speed.

For a given engine, the optimal efficiency can be found by experimental testing on a dynamometer. The engine is run at various operating points, while measurements are taken of many parameters such as engine speed, engine torque, fuel consumption, and emissions. The efficiency is then calculated using the following equation:

$$\eta_{eng} = \frac{P_{mechanical,out}}{P_{fuel}} \tag{4.1}$$

A typical efficiency plot is shown in Figure 4 below. The data is obtained from the Advanced Vehicle Simulator (ADVISOR) software in Matlab. The subject engine is a 1.9L Saturn ICE with a peak torque at 145 Nm. Using this data, one can determine the desired torque requested by choosing the optimal operating point, given a certain engine speed. The result is an optimal efficiency curve, which is overlaid on the efficiency contour in Figure 4.

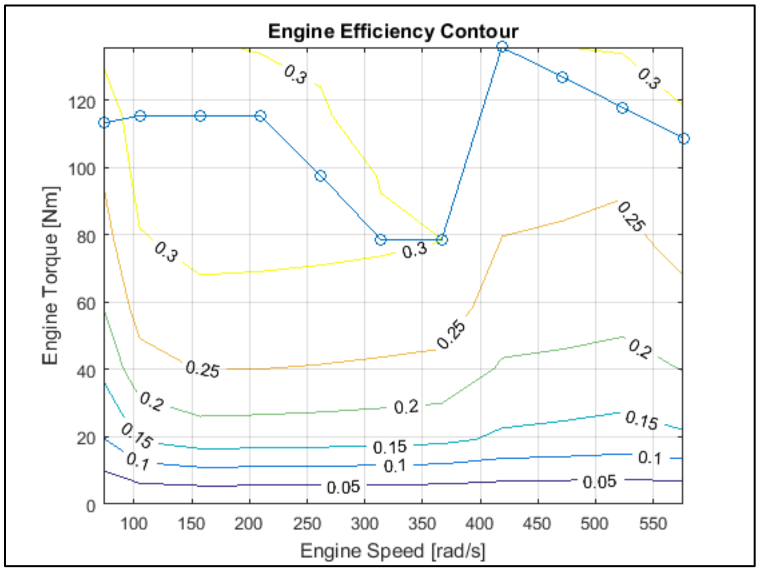


Figure 4: Engine Efficiency Contour, SI_63

4.1.2 Fuel Use Mode

Another strategy mentioned in reference [26] to improve fuel efficiency of a vehicle is the Fuel Use Mode. This mode limits the engine's fuel consumption by preventing the engine to operate above the 1 gram-per-second fuel use line. Once again, the independent variable used in this mode is the engine speed. At each engine speed, there is a certain engine torque with which the pair will result in a fuel consumption of 1 gram per second. Fuel consumption data is obtained experimentally from running an internal combustion engine in a test cell, and measuring the fuel use. This data is used to create fuel consumption contours such as the one in Figure 5. Each curve represents a certain amount of fuel consumption, in grams/sec.

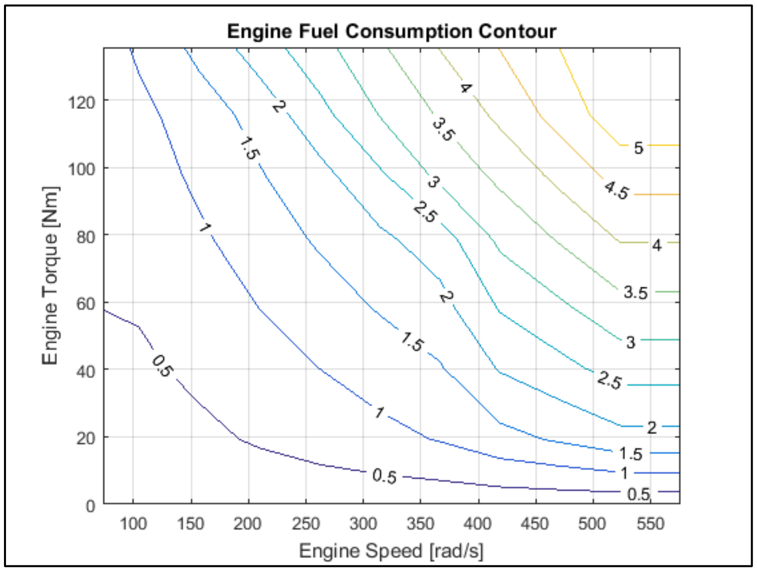


Figure 5: Engine Fuel Consumption Contour, SI_63

In order to operate the internal combustion engine around or below the 1 g/s fuel consumption line, the electric motor must be available to provide any additional torque needed. The 1 g/s line is used as a benchmark in fuel use mode, and the use will be studied further in this thesis.

There are tradeoffs evident when comparing the two operating modes. In Efficiency Mode, the engine is achieving the most power output from input fuel. This brings the thermal losses to a minimum; however as a consequence, the engine consumes more fuel. In Fuel Use Mode, the fuel consumption is being limited, which will improve fuel economy. However, this will be achieved at the cost of a lower thermal efficiency, meaning more power loss. These tradeoffs are decision variables in the design of a vehicle’s powertrain controller.

4.2 Model Formulation

The first part of a vehicle’s powertrain control implementation is software simulation. To perform these simulations, one must have a functioning model of the vehicle. In this research, Simulink (with MATLAB) was used for the formulation and simulation of the model. Simulink is the leading software in the automotive control field, since its visual interpretation of mathematical relations is easier to follow, and the ability to interface with hardware is readily available.

There are two different flow techniques to vehicle models, one being forward-facing and the other backward-facing. A forward-facing model uses the difference between the desired and actual velocity to provide a correction input to the model, from which the required torque is calculated. The chosen controller then determines the torque request to each of the power plants. The torque output of the power plants is then summed up and the model's vehicle speed is calculated along with the energies consumed. A visual representation of a forward-facing model is shown in Figure 6:

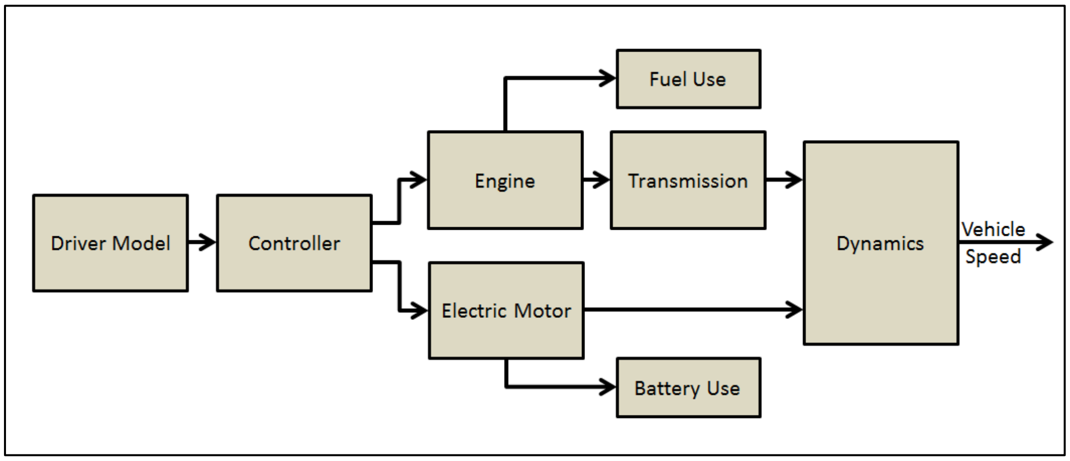


Figure 6: Forward-Facing Model Representation

A backward-facing model calculates the torque required assuming the desired vehicle speed is maintained, and moves through the model in a 'backwards' fashion where the desired output of the component is the input to its block, and the output of the block will be the required input to achieve that desired output. A representation of this type of model is shown below:

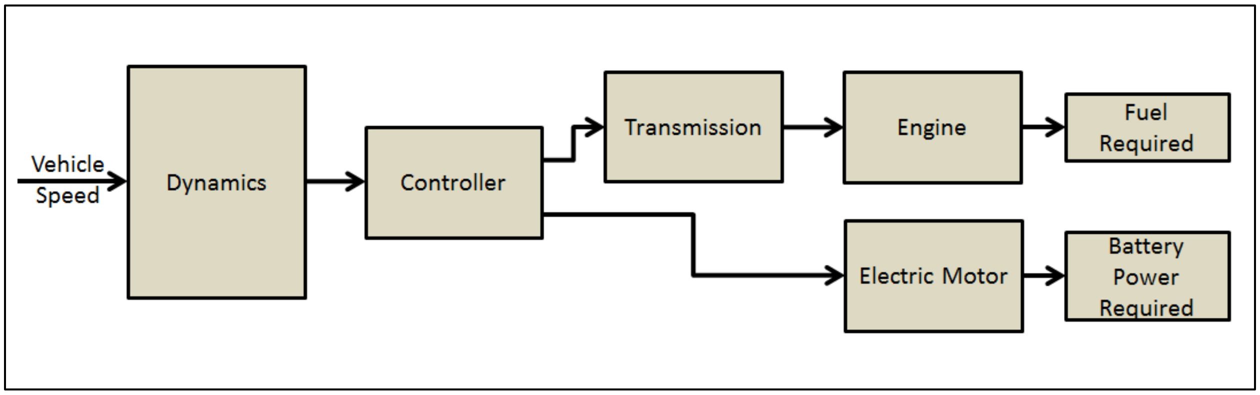


Figure 7: Backward-Facing Model Representation

A backward looking technique is best used for determining the system's energy requirements, since the only thing that matters is the output torque to achieve the speed. In contrast, a forward looking technique is the method best used for control implementation, due to its ability to prove that the control algorithm implemented does indeed achieve the desired vehicle speed [29]. This thesis uses a forward facing model to implement MPC. The complete model can be shown in Figure 8. The following sections describe in detail each of the subsystems used in the model.

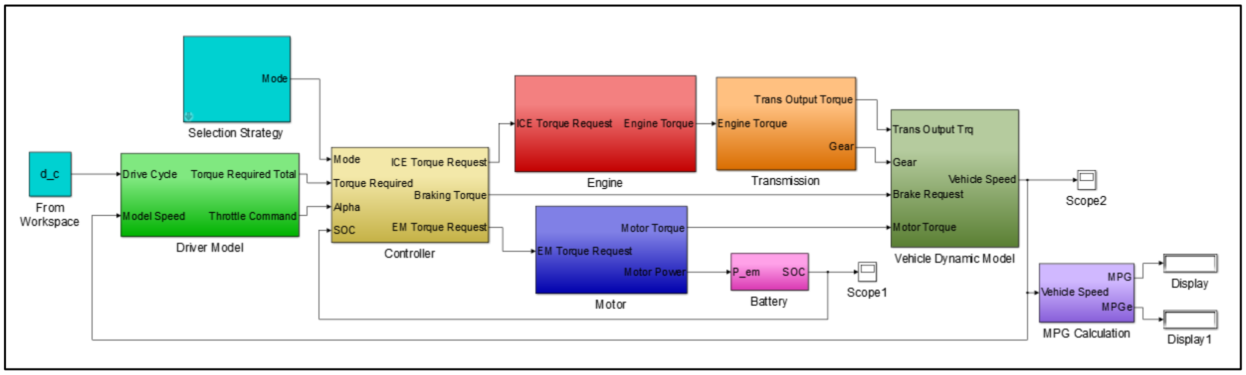


Figure 8: Complete Vehicle Model

4.2.1 Driver Model

The driver model dictates the total required torque of the powertrain given a specific drive cycle. The drive cycle is a predetermined velocity profile, where for each time step there is a chosen vehicle speed. In the model used for this research, the user can select from four drive cycles: Federal Urban Driving Schedule, US06 Drive Cycle, NEDC Drive Cycle, and Artemis Extra Urban Drive Cycle. These drive cycles are shown in Figure 9. The user can also choose how many times to repeat the cycle.

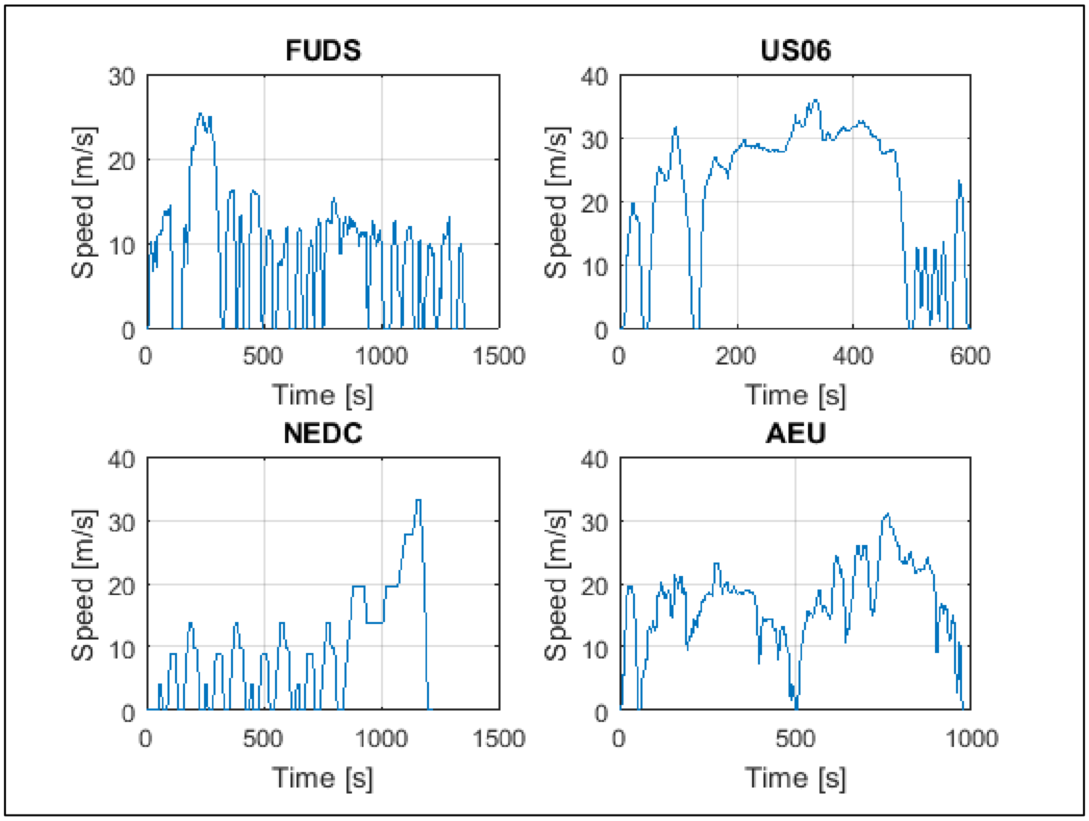


Figure 9: Drive Cycles Used

The driver block receives the desired speed from the drive cycle and the model calculated speed, and uses a PID controller to determine the required torque for the powertrain. In general, a PID controller has a form

$$u = K_p e + K_i \int e \cdot dt + K_d \frac{d}{dt}(e) \tag{4.2}$$

Where u is the control input, e is the error between desired and actual speed, and K_p, K_i, and K_d are the proportional, integral, and derivative gains, respectively. The visual representation of controller is displayed in Figure 10.

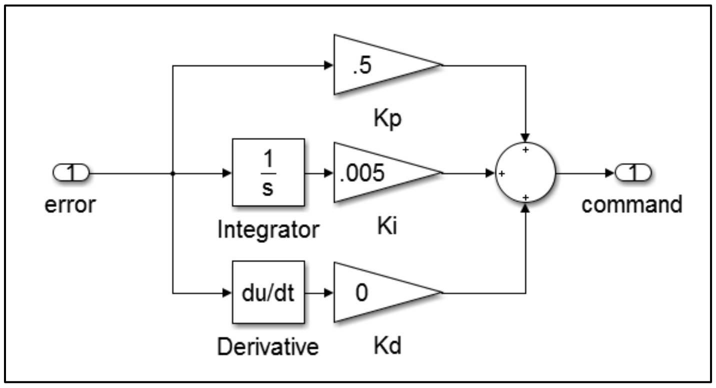


Figure 10: PID for Torque Estimation

In this controller, only proportional and integral gains were used, making it a PI controller. The command in this case is split up in to acceleration and deceleration decisions as shown in Figure 11. Acceleration, denoted in the automotive world as α, is a value between 0 and 1; conversely, deceleration, denoted as β, is a value between 0 and -1. This is done with the use of saturation blocks in Simulink. These values are multiplied by the maximum torque available which is found using data from the engine and motor submodels and the expression:

$$T_{\max,available} = T_{\max,ICE} \cdot GR_{trans} \cdot GR_{ICE,final} + T_{\max,EM} \cdot GR_{EM} \tag{4.3}$$

In (4.3), the variables $T_{\max,available}, T_{\max,ICE}$, and $T_{\max,EM}$ are the maximum torques for the total powertrain, the ICE, and the electric motor, respectively, while $GR_{trans}, GR_{ICE,final}$, and GR_{EM} are the gear ratios for the transmission, the ICE final drive, and the electric motor gear reduction, respectively. The total torque required to operate the vehicle at the drive cycle velocity is then determined by the following relation:

$$T_{required} = \begin{cases} \alpha T_{\max,available} & \alpha \geq 0 \\ \beta T_{\max,available} & \beta < 0 \end{cases} \tag{4.4}$$

Equation (4.4) is implemented using a switch block to send the correct signal to the output. This total vehicle torque required, measured in Newton-meters, is sent to the controller for the torque split decision making. The acceleration value is also sent to the controller to adjust for hard acceleration. This block simulates the response of a human driver attempting to achieve a certain speed while driving, with the α and β values representing the acceleration and brake pedal positions, respectively.

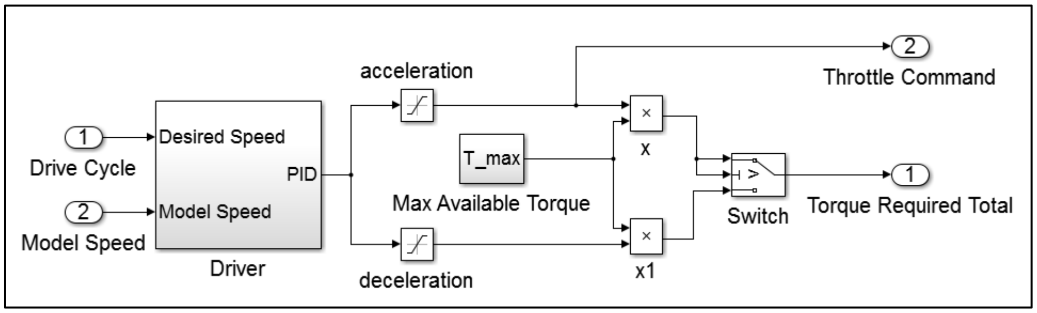


Figure 11: Required Torque Calculation

4.2.2 Selection Strategy

The Selection Strategy block is for the simulation user to select whether to run the model with Fuel Use Mode (mode 1) or Efficiency Mode (mode 0). Depending on which operation mode is chosen, the block will send either a (1) or (0) to the controller, in order to follow the corresponding control algorithm. This block also lets the user define the initial State of Charge (SOC) of the battery pack.

4.2.3 Controller

The controller block is the most important block of the model. This is where the torque commands are created for both the engine and electric motor. This subsystem receives the selected operational mode, the throttle pedal position, the battery state of charge, and the required torque of the powertrain, and determines the torque split between the propulsion systems. A diagram of the controller block is displayed in Figure 12:

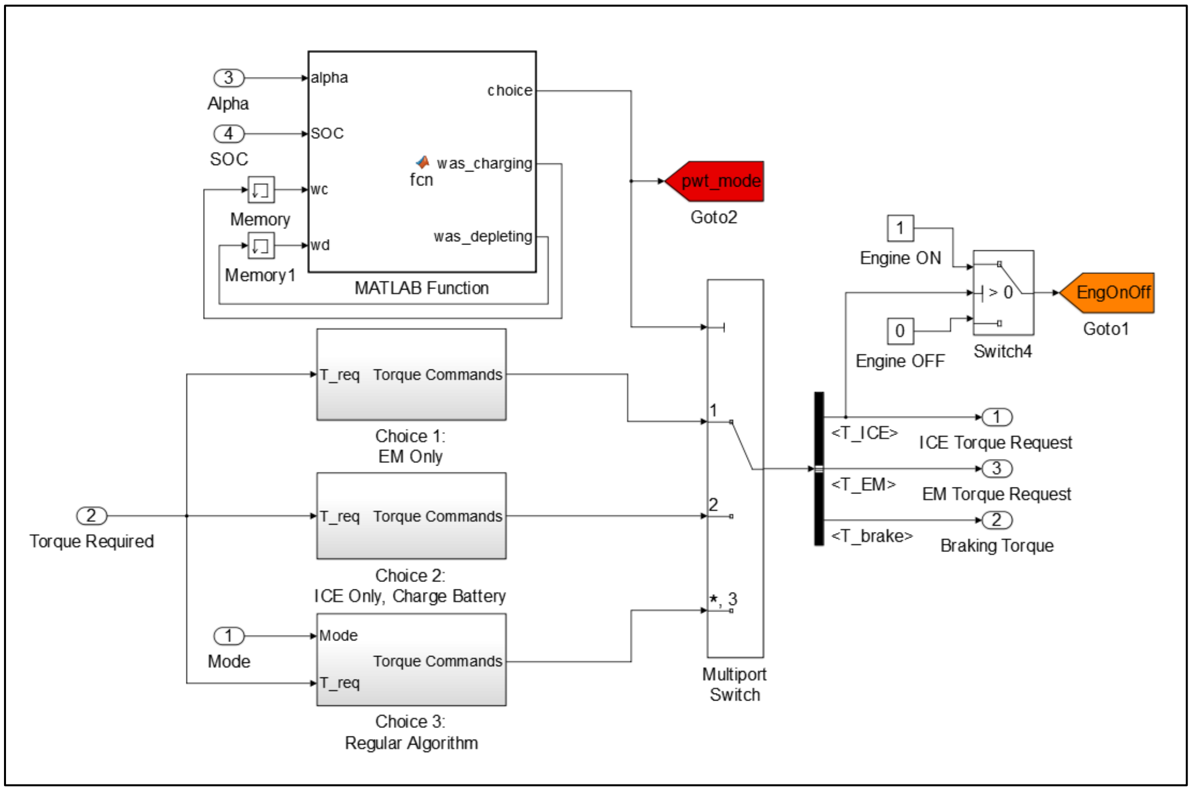


Figure 12: Controller Subsystem

There are many parameters to consider in the construction of a controller. In this research, the variables to consider are SOC, throttle position, and vehicle speed. A Matlab script was written in order to use those key parameters to choose from one of three propulsion cases: electric motor propulsion, ICE propulsion, and a torque split between the two. The pseudo code for the function is as follows (Matlab codes can be found in the Appendix):

- If the SOC is above the upper limit (80%) and there is no hard acceleration, shut off the engine and use electric motor propulsion only
- If the SOC is below the lower limit (20%) and there is no hard acceleration, enter ICE propulsion mode, and charge batteries
- If the SOC is between the acceptable limits, or there is hard acceleration, use the torque split mode
- If electric propulsion mode has been entered, remain in mode until the battery SOC reaches the depletion limit (75%)
- If ICE propulsion mode has been entered, remain in mode until the battery SOC reaches the charge limit (50%)
- If in ICE propulsion mode or in electric propulsion mode and a hard acceleration is needed, enter torque split mode but return to previous mode after hard acceleration is complete

Hard acceleration has been defined as 60% throttle. If there is a need for hard acceleration, both the motor and the engine need to be able to provide the sufficient power. The fourth and fifth bullet points are used so that the vehicle can enter the torque split mode at a reasonable SOC. If, for example, the battery SOC is at 18% and bullet point 5 is not enabled, the vehicle will enter ICE propulsion mode to charge the batteries, and once an SOC of 20% is reached, it will want to enter torque split mode immediately, and then will stay at the 20% line while it depletes and charges. This bullet point will prevent it from wavering around the charge limit, by keeping the controller in ICE mode until the battery SOC reaches 50%. The following paragraphs explain each powertrain mode in detail.

The first method to be described is the combined torque-split method, when the engine is run at the optimal torque as determined by the MPC algorithm, and the electric motor provides the remaining necessary torque. One decision made for this mode is to have the electric motor be the only power supply while the vehicle speed is 10 m/s or less, which equates to 22.4 mph. Electric take-off will help in saving fuel, and is common practice in the automotive controls industry. If the vehicle speed is greater than 10 m/s, the engine will be considered on, and the MPC algorithm torque will be commanded, which is calculated offline and discussed in a Section 4.3. The electric motor will make up for the difference between the total torque required and the engine commanded torque. This representation is displayed in Figure 13 below:

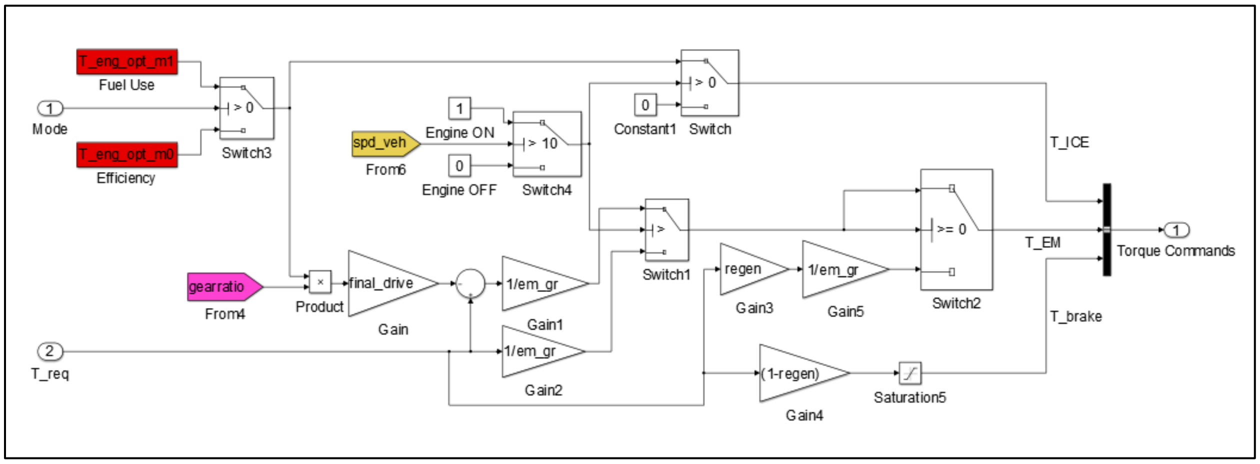


Figure 13: Torque Split Mode

Due to the engine needing a transmission and the electric motor needing a gear reduction, the equation to determine the electric motor torque when the engine is on is

$$T_{EM} = \frac{T_{requested,total} - T_{ICE,MPC} \cdot GR_{trans} \cdot GR_{ICE,final}}{GR_{EM}} \tag{4.5}$$

The transmission gear ratio is determined based on the engine speed further along the model; the other two gear ratios are fixed specifications of the vehicle. If the vehicle speed is 10 m/s or less, the engine will be considered off, and the torque will be routed completely to the electric motor. The required torque for the electric motor in this case is

$$T_{EM} = \frac{T_{requested,total}}{GR_{EM}} \tag{4.6}$$

If the torque requested from the driver block is negative, the model needs to execute a braking function. In hybrid vehicles, this is done by a combination of a mechanical brake and regenerative braking. Regenerative braking is when the vehicle's dynamic motion winds up the electric motor to act as a generator and charge the batteries. In turn, the vehicle will slow down. A percentage of the braking can be executed by the regenerative system, which for this research was chosen as 60%. The remaining braking torque required is provided by the mechanical braking system of the vehicle.

In the electric motor propulsion case, the engine torque is set to zero, and the electric motor provides all the torque to the wheels which makes it a charge-depleting mode. This is used when the battery state of charge is sufficiently high and it can be used to propel the vehicle without wasting gasoline. A diagram of this mode is shown in Figure 14. The braking command is executed in the same manner as the previous method.

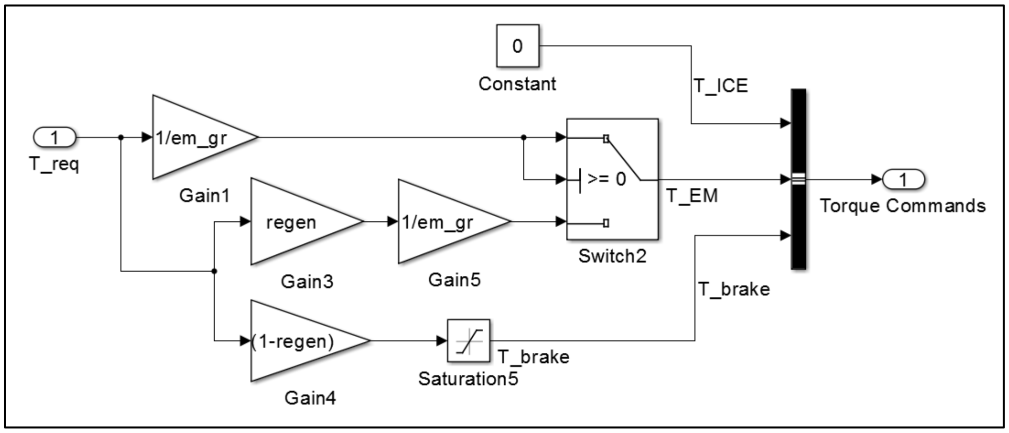


Figure 14: EM Propulsion Only

The final method is the ICE propulsion case. The engine provides the torque to the wheels, and charges the battery with any surplus torque that can be produced. The engine is run at maximum efficiency if possible; if the torque required is more than the engine's maximum efficiency torque, then the engine will run at maximum torque. This is used when the SOC is significantly low and the engine is needed to sustain the charge in the battery. When the vehicle comes to a stop, the engine will also turn off in order to save fuel. The visual representation of this type of operation is shown in Figure 15.

In all methods, the commanded torques of the engine, electric motor, and mechanical brake are sent out to the main controller block. If for any scenario, the engine torque is set to zero, a switch sends a signal to the rest of the system that the engine is off, as seen in Figure 12. The three torque requests are then output to their respective locations.

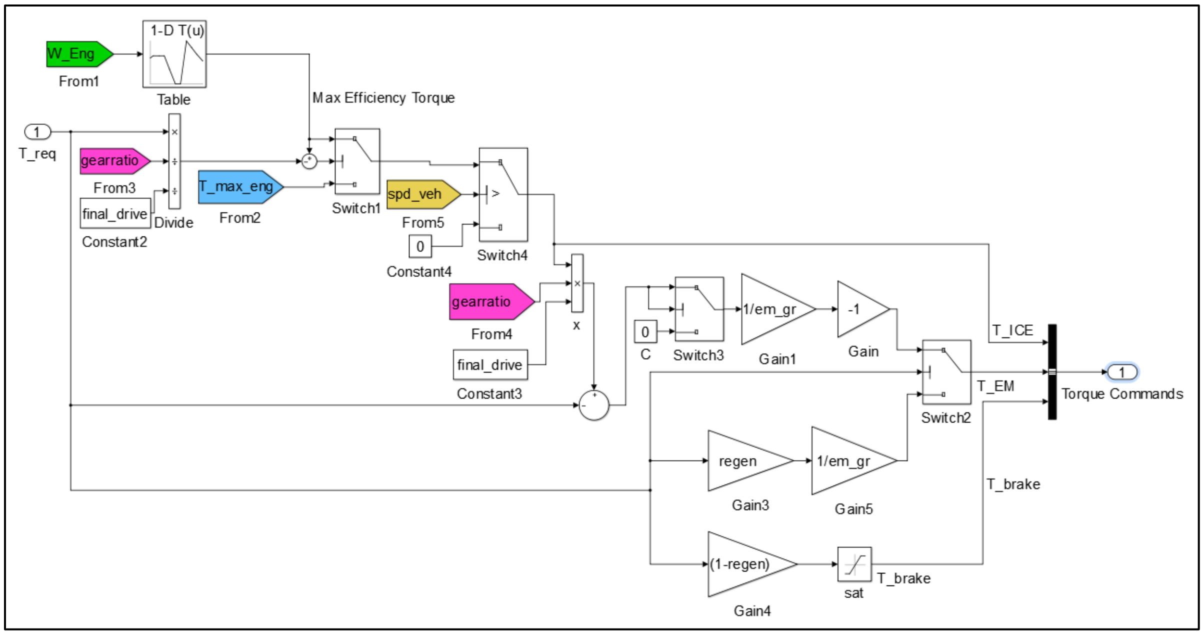


Figure 15: ICE Propulsion Only

4.2.4 Electric Motor

The electric motor model used in this simulator is a simplified model where torque output is equal to the torque request, while within the electric motor operating conditions. This is done since the focus of the research is the control construction and engine fuel consumption, as compared to analysis for each individual component. A saturation block is used in order to limit the electric motor torque to its capability at the current operating condition. The maximum and minimum torque values are determined by motor speed and a lookup table of electric motor data. This is considered a quasi-static model [4]. All electric motor data was obtained from ADVISOR, and the specifications are shown in Table 1 below. The power required by the electric motor from the battery is also calculated for means of battery analysis. For discharging, this is found by the following equation:

$$P_b = \frac{T_{EM}\,\omega_{EM}}{\eta_{EM}} \tag{4.7}$$

where ω_{EM} is the electric motor's angular speed and η_{EM} is the electric motor efficiency. When the electric motor torque is negative, the batteries are charging and the power gained is found as below:

$$P_b = T_{EM}\omega_{EM}\eta_{EM} \tag{4.8}$$

The motor efficiency is calculated by using a lookup table with motor speed and torque data. The electric motor subsystem is shown in Figure 16.

Table 1: Electric Motor Specifications

Manufacturer: Honda	Power Rating: 49kW cont.	Type: Permanent Magnet
Maximum Current: 400A	Minimum Voltage: 60V	Gearbox Ratio: 9.0

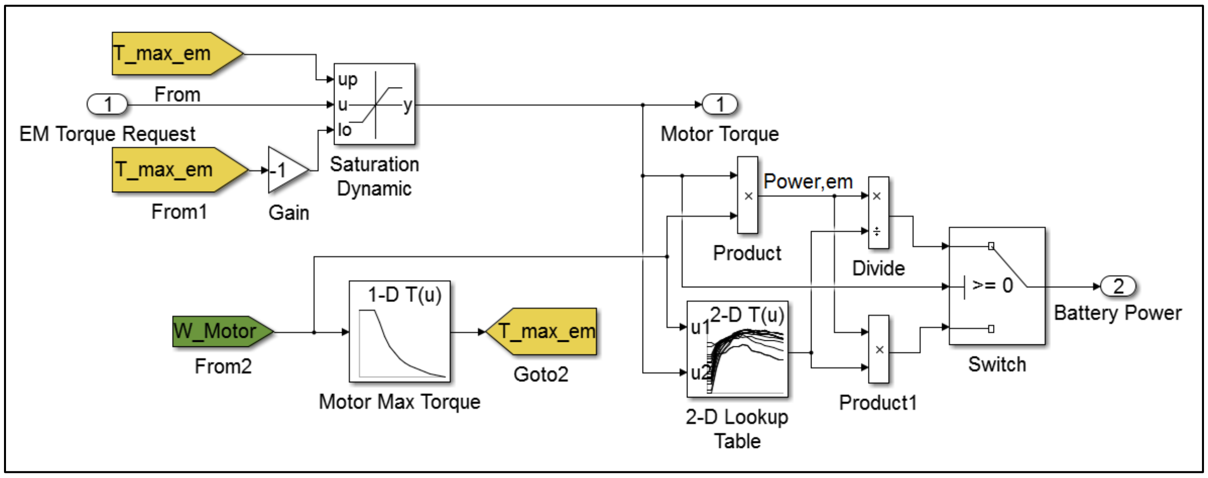


Figure 16: Electric Motor Subsystem

4.2.5 Battery

The battery subsystem accompanies the electric motor subsystem. This submodel is used to calculate the State of Charge (SOC) of the energy storage system (ESS) on the vehicle, and also to calculate the battery power required in order to calculate the equivalent MPG, more commonly known as MPGe. Figure 17 shows the visual layout of the subsystem.

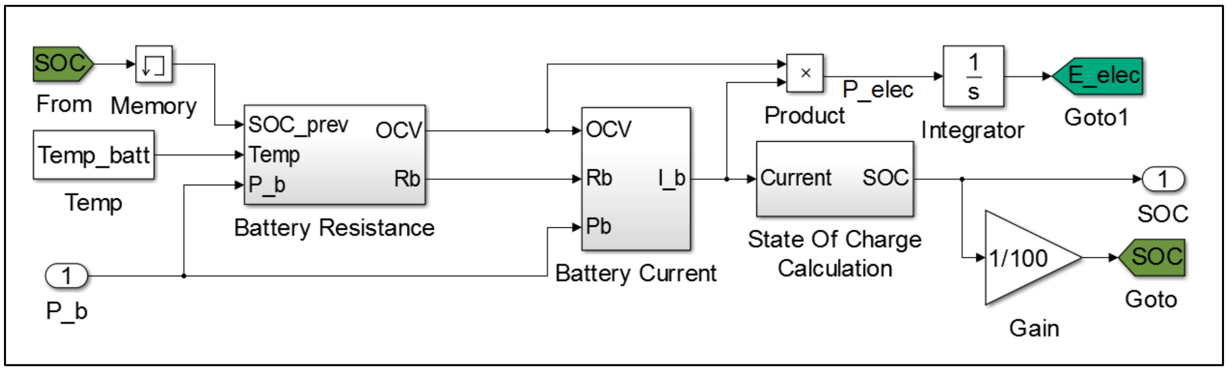


Figure 17: Battery Subsystem

The battery model used is an equivalent circuit model as obtained from reference [28]. This model uses the open current voltage of the battery pack in series with the battery's internal resistance. The governing equations for this system are:

$$I_b = \frac{V_{oc}}{2R_b} - \sqrt{\frac{V_{oc}^{\ 2} - 4R_b P_b}{4R_b^{\ 2}}} \tag{4.9}$$

$$S\dot{O}C = -\frac{I_b}{Q_b} \tag{4.10}$$

$$P_{elec} = V_{oc} I_b \tag{4.11}$$

The variables are defined in Table 2.

Table 2: Battery Model Variables

I_b = Battery Current	V_{OC} = Open Circuit Voltage	R_b = Internal Resistance
P_b = Battery Power	Q_b = Total Battery Capacity	P_{elec} = Battery Electrical Power

Equation (4.9) calculates the current in the battery. This is also displayed visually in Figure 18.

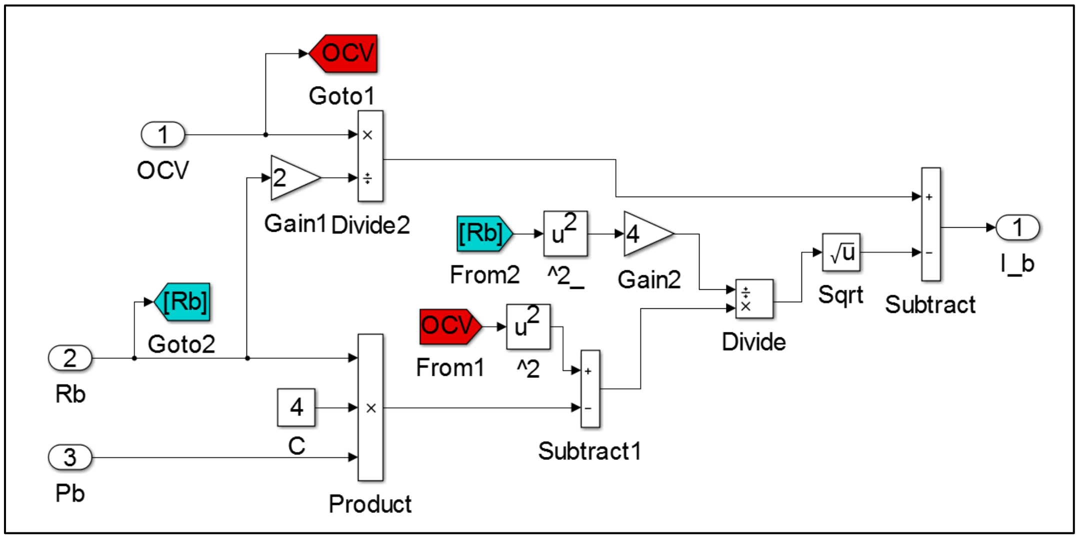


Figure 18: Battery Current Calculation

The battery power P_b is received as in input from the motor model. V_{OC}, the open current voltage, and R_b, the internal resistance, are found by using a lookup table from the battery data, as shown in Figure 19. The independent variables are SOC and battery temperature. In this research, the temperature was fixed at $25°C$ for simplicity.

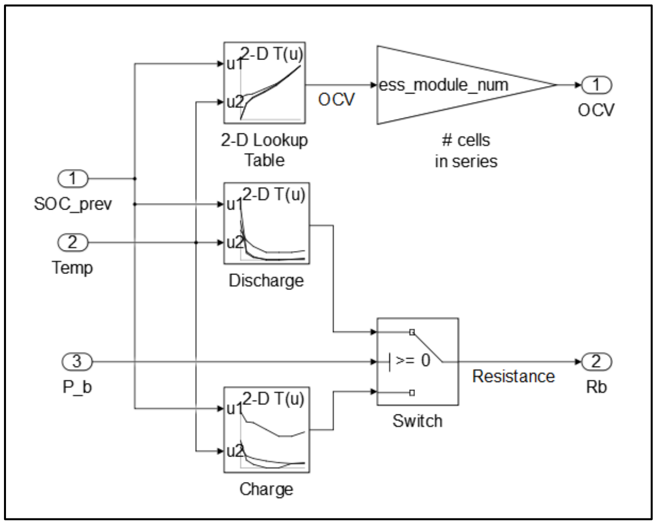


Figure 19: V_{oc} and R_b Calculation

The battery pack used in this model is a Lithium-Ion battery pack obtained from ADVISOR. The characteristics of the battery are shown in Table 3. The open current voltage from the ESS data must be multiplied by the number of modules in series. The internal resistance depends on whether the battery is discharging or charging; thus a switch is used depending on if the power drawn from the motor is positive (discharging) or negative (charging). Once the battery current is calculated, the state of charge can be obtained by integrating Equation (4.10). The battery charge capacity is denoted by Q_b, which is the battery capacity of all modules in parallel. One must note that capacity is given in units of Amp-hours, and must be converted to Amp-seconds in order for SOC to be calculated. The SOC calculation, which is an integration of the rate of state of charge, can be seen in Figure 20. An initial state of charge is required for the model. This is varied for simulation analysis and explained in further sections.

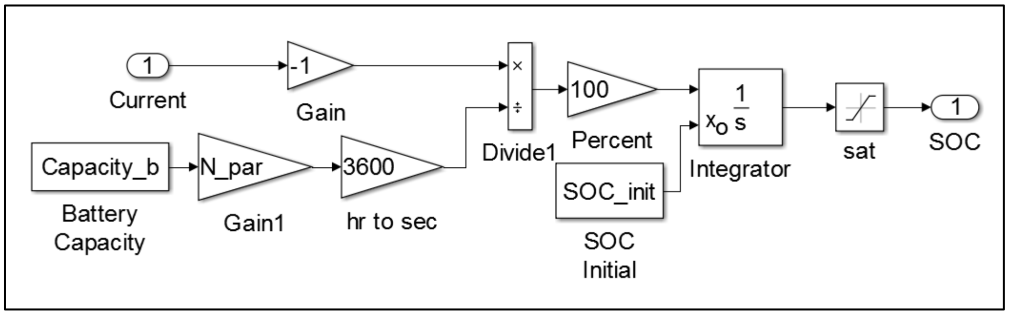


Figure 20: SOC Calculation

The final component to the battery subsystem is the battery's electrical power consumption calculation. This calculation, which can be seen in Figure 17, is displayed in Equation (4.11). This is then integrated to get the electrical energy consumed in Joules, and is sent to the MPG calculation block.

Table 3: Li-Ion Battery Specifications

Battery Capacity: 7.035 [Ah]	No. of Modules in Series: 25	Min Module Voltage: 6 [V]
Mass of Module: 1.1347 [kg]	No. of Modules in Parallel: 2	Max Module Voltage: 11.7 [V]

4.2.6 Internal Combustion Engine

The basis of the ICE subsystem is similar to that of the electric motor; however a few additions are present: correction for uncertainties and fuel calculations. This can be seen in Figure 21. The engine torque is determined with the same lookup table process as the EM. Then the torque and speed are sent to the fuel and efficiency calculation block to determine the fuel consumption and the thermal efficiency of the ICE in an implementation of quasi-static modeling [4]. These are also done with lookup tables of engine data. All data for the engine was obtained from ADVISOR, and the specifications are shown in Table 4:

Table 4: Internal Combustion Engine Specifications

Type: Saturn 1.9L SOHC	Mass: 200.54kg	Peak Torque: 145Nm @2000 RPM
Fuel Density: 749g/L	LHV: 4.26kJ/g	Max Power: 63kW @5500 RPM

After obtaining the fuel consumption data, the amount of gallons used can be calculated for the purpose of obtaining fuel economy of the vehicle. The relationship between gallons and fuel consumption is

$$gal_{ICE} = \int_{t_1}^{t_2} L2G \frac{fc}{\rho_{fuel}} dt \tag{4.12}$$

where fc is defined as the fuel consumption in grams/second, ρ_{fuel} is the fuel density of the petrol in grams/L, and L2G is a conversion factor from liters to gallons. The engine model also calculates the maximum available engine torque, to limit the torque output.

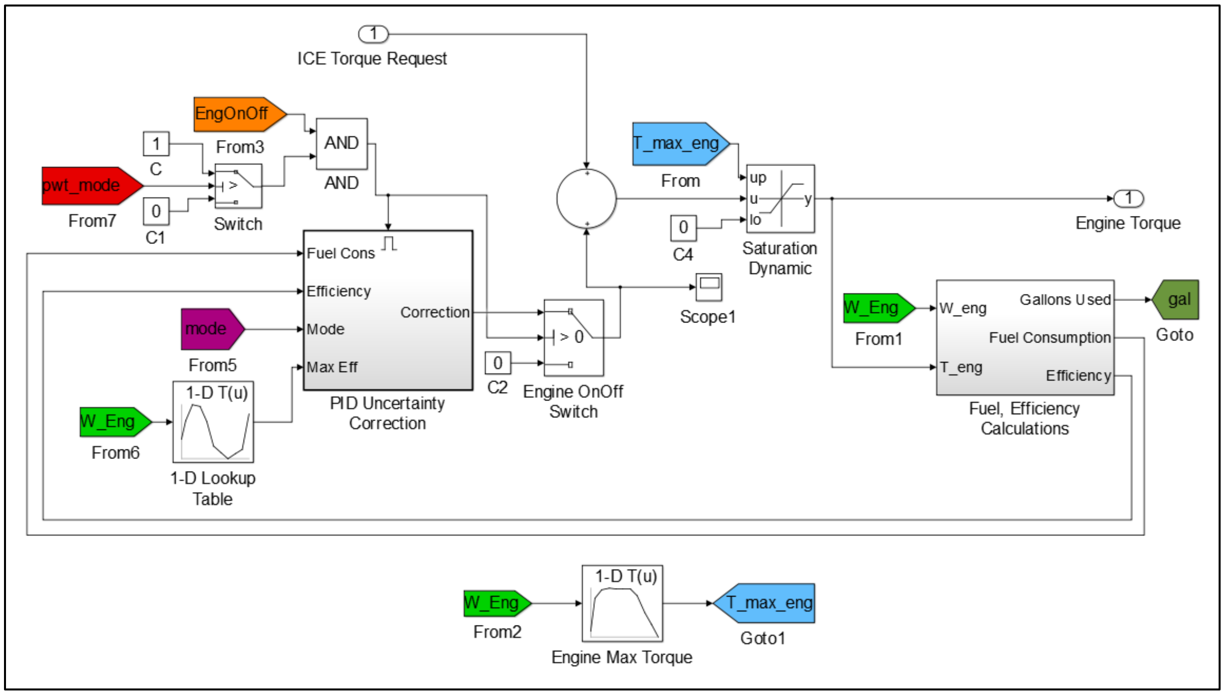


Figure 21: Internal Combustion Engine Subsystem

If, in the control strategy block, the engine is determined to be off, the fuel consumption will be set to zero for that instance. The diagram for the fuel consumption and efficiency calculation can be seen below in Figure 22.

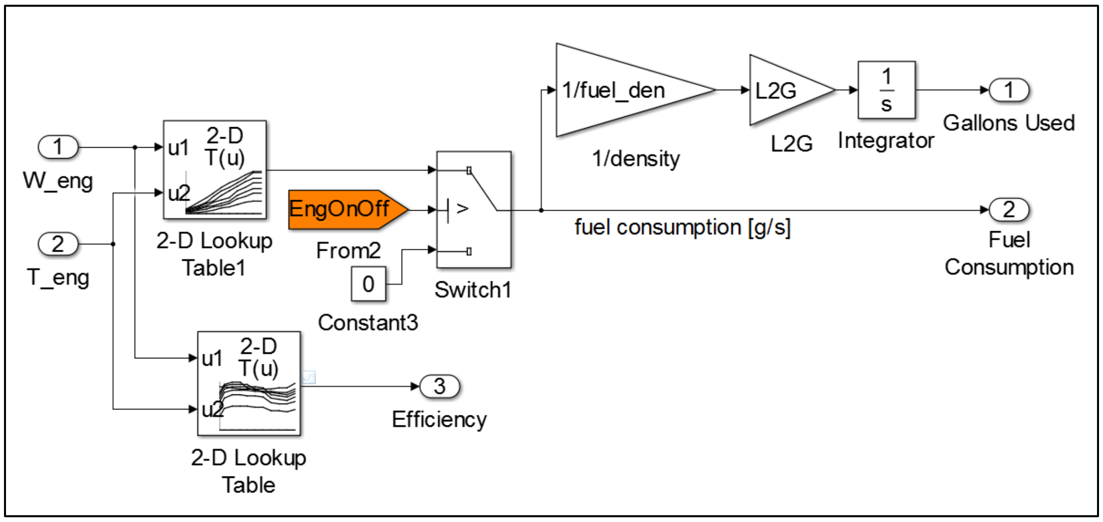


Figure 22: Fuel Calculations

Once the fuel consumption and efficiency are calculated, they are fed into the uncertainty correction subsystem. This block is part of the practical implementation of MPC. In the theory described in Chapter 3, the difference between the actual system output and the predicted output is fed back into the model as a disturbance. Using this same idea, in Fuel Use Mode, the previous step's fuel consumption is compared to 1 gram/second and using PID, a correction term is added to the ICE torque request. In Efficiency mode, the previous step's efficiency is compared to the maximum efficiency at the previous engine speed, and PID is used to provide the corrective term to the torque. These corrections can be seen in Figure 23. This keeps the model attempting to continually obtain 1 gram/second fuel consumption or maximum efficiency, regardless of uncertainties. It is important to note this correction is only added if the engine is considered on and in torque-split mode. This is done with the use of an Enabled Subsystem in Simulink, which only executes when the switch value sent to it is positive, as seen in Figure 21.

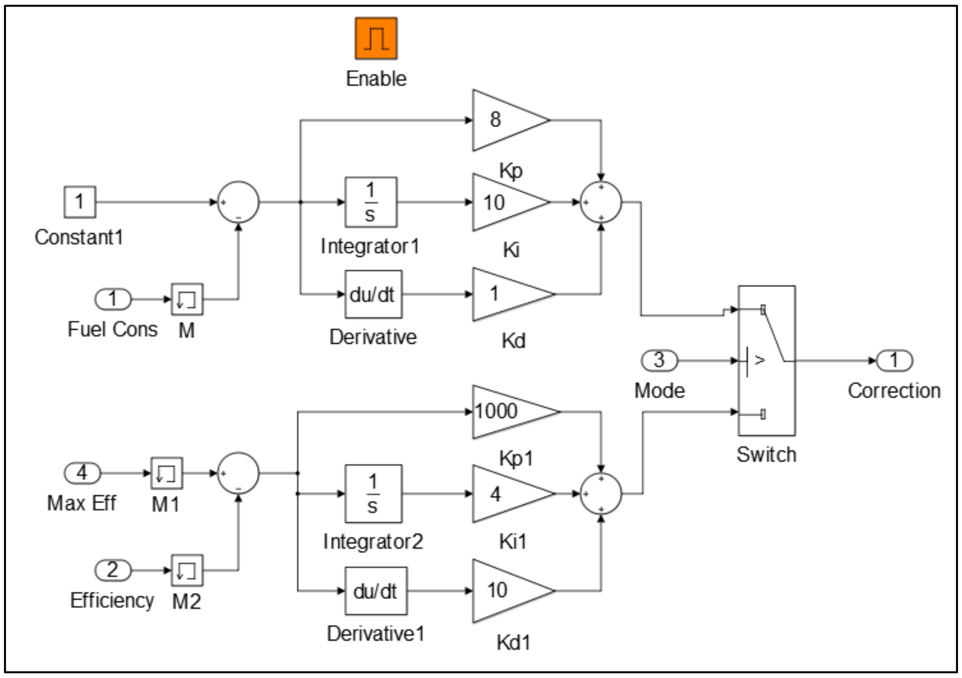


Figure 23: PID Uncertainty Correction

4.2.7 Transmission

The transmission block is used to transfer the engine output torque to the axle, by multiplying the torque by the transmission gear. The transmission gear is determined with a user-created Matlab function. The pseudo code is as follows:

- If the engine speed is greater than the upper limit speed, shift up a gear.
- If the engine speed is less than the lower limit speed, shift down a gear.
- If the engine speed is between the limits, stay in the current gear.
- If the desire is to shift up, but the current position is the top gear, do nothing.
- If the desire is to shift down, but the current position is first gear, do nothing.

The Matlab script can be found in the Appendix. The upper limit used in this algorithm is 350 rad/s (~3342 RPM) and the lower limit used is 125 rad/s (~1194 RPM). After the correct gear is determined, a lookup table is used to obtain the corresponding gear ratio. This gear ratio multiplies with the engine torque to provide the transmission output torque to the vehicle system:

$$T_{trans,output} = T_{ICE} \cdot GR_{trans} \tag{4.13}$$

The configuration can be seen in Figure 24. The transmission selected was a 5 speed transmission from ADVISOR. The following table gives the gear ratios:

Table 5: Transmission Gear Ratios

Gear	1st	2nd	3rd	4th	5th	Final Drive
Gear Ratio	3.25	1.81	1.21	0.86	0.64	4.06

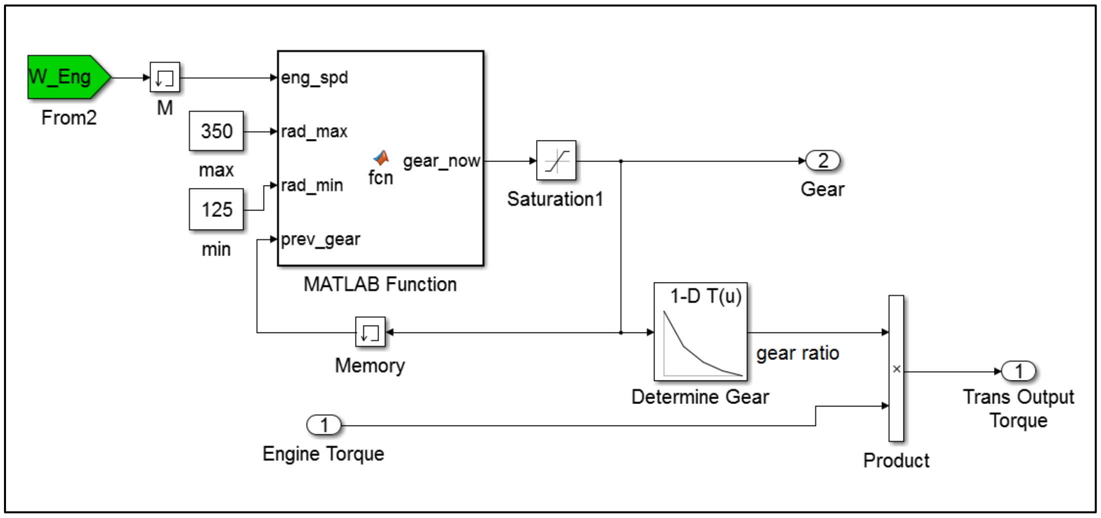


Figure 24: Transmission Subsystem

4.2.8 Vehicle Dynamic Model

The Vehicle Dynamic Model is representative of the vehicle system in relation to the road. This subsystem receives the transmission output torque (provided by the ICE), the electric motor torque, and the braking torque and calculates the speed of the vehicle. The dynamic model is shown in Figure 25:

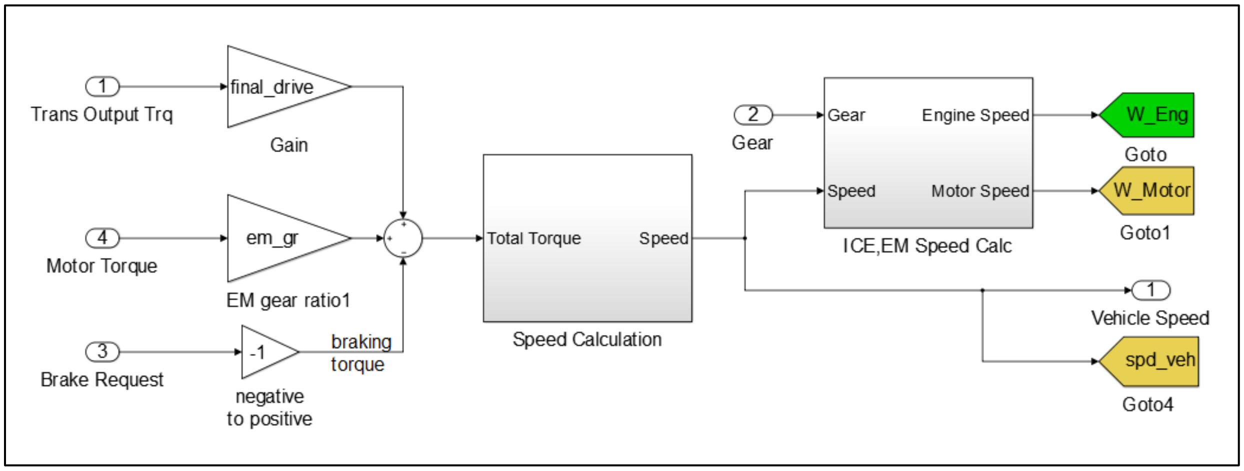


Figure 25: Vehicle Dynamic Model Subsystem

The total axle torque that is used to compute the vehicle speed is a summation specified by:

$$T_{tot} = T_{trans,out} \cdot GR_{ICE,\,final} + T_{EM} \cdot GR_{EM} - T_{brake} \tag{4.14}$$

This torque is then the input to the fundamental vehicle dynamic equation [3] as shown in equation (4.15).

$$\frac{T_{tot}}{r_w} = F_{tr} = \frac{1}{2}\rho C_d A V^2 + Mgc_{rr} + M_i \frac{dV}{dt} \tag{4.15}$$

The variables and their units are in Table 6. This equation holds assuming there is no slope in the road.

Table 6: Vehicle Parameters

T_{tot} = total wheel torque [Nm]	r_w = wheel radius = 0.324 [m]	F_{tr} = tractive force [N]
ρ = density of air 1.2 [kg/m^3]	C_d = drag coefficient	A = frontal area [m^2]
V = vehicle speed [m/s]	M = vehicle mass = 2050 [kg]	g = gravity = 9.8 [m/s^2]
c_{rr} = rolling coefficient = 0.01	M_i = inertial mass = 2132 [kg]	$\frac{dV}{dt}$ = acceleration [m/s^2]

The drag coefficient and the frontal area are grouped together in this thesis for a value of $0.76m^2$. The values shown in Table 6 were obtained from reference [27]. Appearing in the form of a differential equation, the vehicle speed can be calculated from (4.15) by using a Simulink feedback loop as shown in Figure 26.

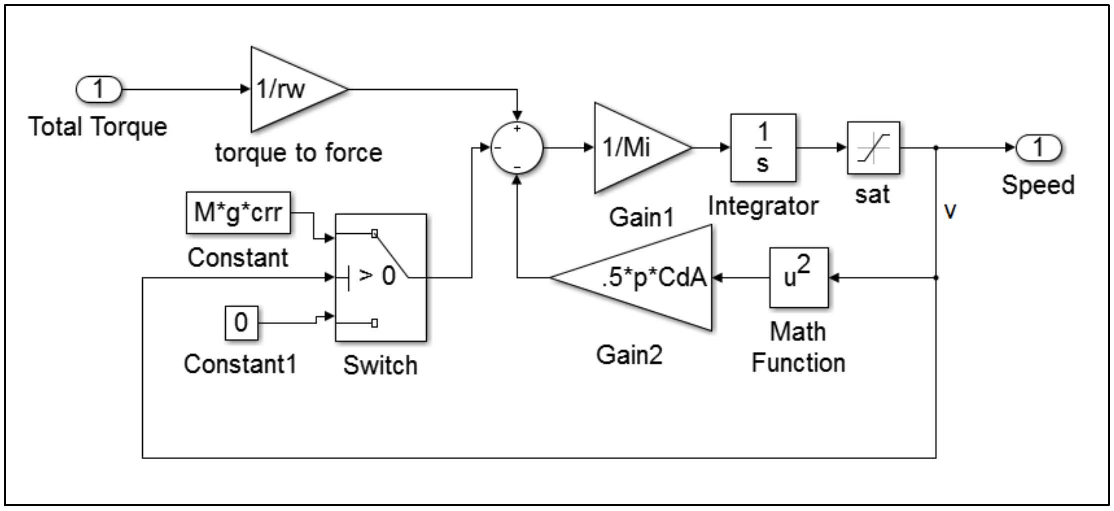


Figure 26: Speed Calculation Subsystem

Once the vehicle speed is calculated, both the engine speed and the motor speed can be determined. This is shown in Figure 27:

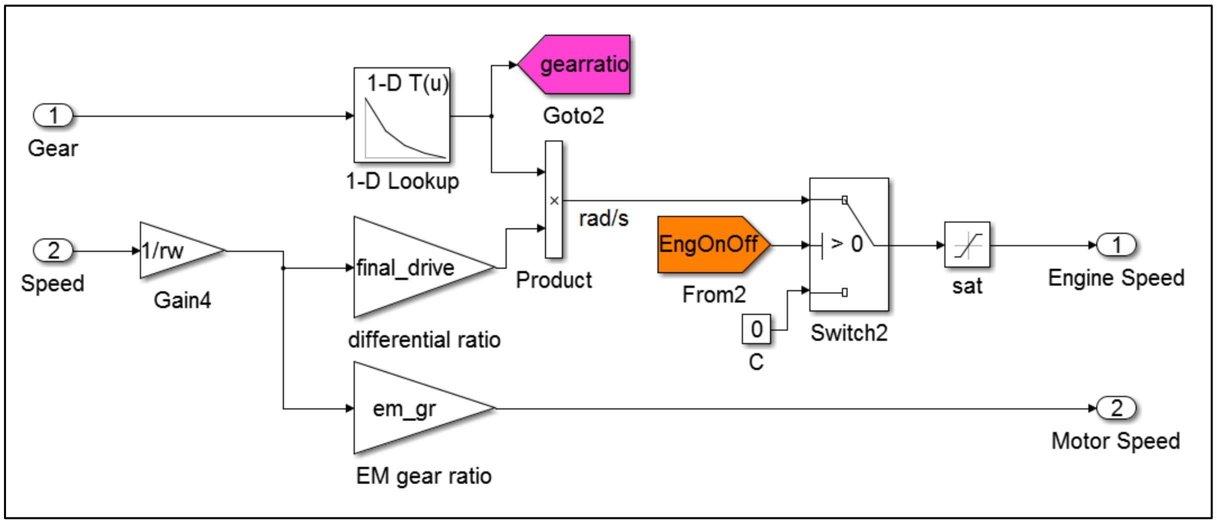


Figure 27: ICE and EM Speed Calculations

There must be knowledge of what gear the transmission is in. A lookup table is used to determine the gear ratio corresponding to that gear. Meanwhile, the vehicle speed is converted to axle speed with the following relation:

$$\omega_{axle} = \frac{V}{r_w} \tag{4.16}$$

The units are in radians per second. From here, the engine speed can be determined as

$$\omega_{ICE} = \omega_{axle} \cdot GR_{ICE,\,final} \cdot GR_{trans} \tag{4.17}$$

To obtain the angular speed of the electric motor, the axle speed is multiplied by the electric motor's fixed gearbox ratio:

$$\omega_{EM} = \omega_{axle} \cdot GR_{EM} \tag{4.18}$$

The three speeds calculated in the Vehicle Dynamic Model subsystem are then sent to other parts of the Simulink simulator by use of 'Goto' blocks.

4.2.9 MPG Calculation

One of the purposes of this simulator is to observe changes in fuel economy due to changes in the control algorithm. Thus, the final block to cover in this description of the model is the MPG Calculation subsystem, shown in Figure 28:

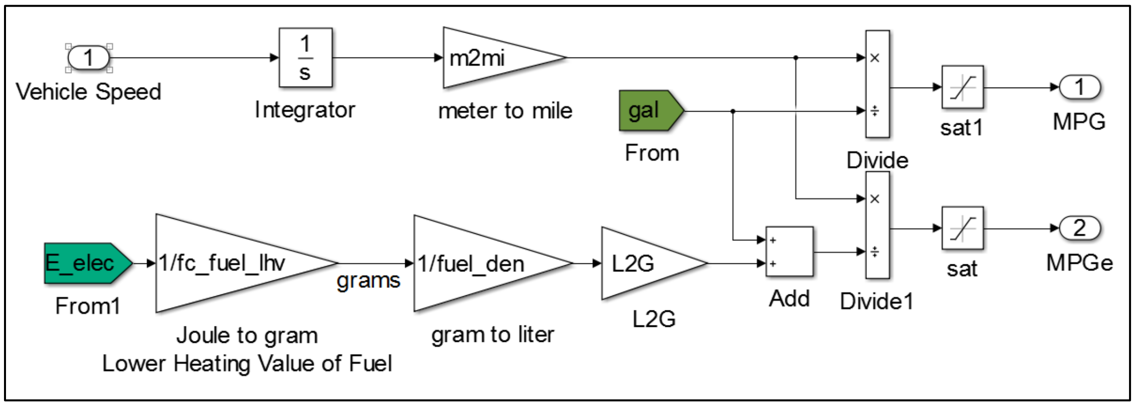


Figure 28: MPG Calculation Subsystem

The inputs to the subsystem are the vehicle's speed, gallons used by the engine, and energy consumed by the ESS. The vehicle speed is integrated to get the distance traveled during the drive cycle, as in Equation (4.19).

$$d = \int V dt \tag{4.19}$$

The units are in meters; however, in order to compute MPG of the system, a gain block is needed to convert from meters to miles traveled. The calculation of the model MPG is a simple division shown below:

$$MPG = \frac{d}{gal_{ICE}} \tag{4.20}$$

An important addition to fuel economy calculations in hybrid vehicles is the equivalent MPG, or MPGe. This is calculated by making an equivalent fuel consumption term for the batteries. This will convert the energy consumed by the battery into grams of fuel consumed if the same energy was used by the engine. The following relation is used to convert electrical energy into equivalent gallons of fuel:

$$gal_{EM,eq} = \frac{E_{elec}}{LHV} \frac{1}{\rho_{fuel}} L2G \tag{4.21}$$

With this equivalent term calculated, the MPGe of the system can be described as

$$MPGe = \frac{d}{gal_{ICE} + gal_{EM,eq}} \tag{4.22}$$

During a simulation of the model, these fuel parameters will provide the instantaneous fuel economy, but the final values, after the drive cycle is complete, are sent to the Matlab workspace to analyze.

4.3 Implementation of MPC Control Algorithm

The mathematical representation of MPC is shown in Chapter 3; however, in order to implement in this form of research, a practical approach must be used. This section will lay out the different MPC variables used and then describe the MPC process, which is completed offline in order to use less computation time, since drive cycles are readily available before simulation.

4.3.1 MPC Parameters

When implementing MPC in a system, one important variable is the receding horizon length, denoted as N. Changing the horizon will change the results of the simulation. For this simulator, the step size is one second, so each step forward, the horizon recedes one second further. Different horizons were tested and compared, and discussed in Section 5.1.1.

As with the mathematical model, this implementation of MPC requires a reference for the prediction model to follow. Due to two different operational modes (Fuel Use Mode and Efficiency Mode), there must be two different references. For Fuel Use Mode, the reference chosen was the 1 gram-per-second line; this means the MPC controller will be formulated to get the internal combustion engine's torque as close to the 1 gram-per-second fuel consumption line as possible. This is slightly different than how reference [26] used Fuel Use Mode, but is adequate in showing the improvement of an MPC controller as compared to a conventional vehicle. In mathematical representation, the cost function becomes

$$J = \sum_{i=1}^{N} [(fc_i - r)^2 + \lambda_1 T_{ICE}{}^2] \tag{4.23}$$

Where fc_i is the fuel consumption of the engine at step 'i', r is the reference, 1 gram/second, and T_{ICE} is the control input, the engine requested torque. This torque is multiplied by λ_1, a scaling factor. Minimizing the cost in (4.23) will make the engine torque follow the 1 gram-per-second line over the whole horizon, which will improve the fuel economy of the vehicle. In the case of Efficiency Mode, the goal is to operate the engine at its maximum efficiency over the horizon specified. In this case, the reference becomes the maximum efficiency point of the engine. Mathematically, the cost function becomes

$$J = \sum_{i=1}^{N} [(eff_i - eff_{\max,i})^2 + \lambda_0 T_{ICE}{}^2] \tag{4.24}$$

Where eff_i is the efficiency of the engine at each step, and $eff_{\max,i}$ is the maximum efficiency for the given engine speed at the given time step. Similar to Fuel Use Mode, there is also a cost term on the control. Minimizing the cost over the drive cycle in Efficiency Mode would bring the engine's efficiency close to the maximum engine efficiency for the drive cycle.

The last parameter to mention is the control input. As Chapter 3 showed above, the cost function is minimized over the control, in order to get the next control input to the system. Since the goal in both operational modes is to make the engine follow a certain rule, the obvious choice for the control input to the system is the engine torque request. The MPC will be implemented on the engine model alone.

4.3.2 MPC Process

The procedure to calculate the optimal torque request at each time step is described below. It was derived with the help of reference [28] and its corresponding Simulink model. However, the calculations in this research are all done offline, using Matlab scripts and functions to complete the task in an ordered manner. First the engine data and drive cycle data must be loaded. In order to compute the necessary engine torques, the engine speed must be calculated. This is done with the same process as the ICE speed calculation in section 4.2.8, and specifically Figure 27. However, in this case, the transmission gear is not known. A Matlab function was created in order to determine the transmission gear and the corresponding engine speed for each time step. The function, which is displayed in the Appendix, uses the same process as described in Section 4.2.7 to find the transmission gear for the simulator. Once the engine speed is back-calculated from the transmission speed and gear ratio, the MPC setup can begin. The maximum engine torque at each speed is found using interpolation from engine data. This is used to create an engine torque vector at each time step, ranging from zero to maximum engine torque in increments of 0.1 (11 total positions). This provides a range of all possible torques over the whole drive cycle, which will be used for prediction purposes. The fuel consumption and the efficiency are calculated at each position of each step by interpolating engine data to obtain a fuel consumption matrix and an efficiency matrix. Next, at each time step, all possible combinations of sums of fuel consumption and efficiency over the receding horizon are added, in order to make a set different outputs; this is constitutes the prediction part of the MPC algorithm. Because the torque vector ranges from zero to maximum for each time, the combinations of sums of data provide all possible predictions over the time horizon. From here, the cost function is made. This vector of fuel consumptions (or efficiencies in Efficiency Mode) is compared to the reference as shown in equations (4.23) and (4.24), and the control factor is also applied. Now the minimum of this vector is taken, which is effectively minimizing the cost function over the horizon. The control input—engine torque—is assigned as with the index corresponding to this minimum of the cost. This provides the optimal engine torque for the current time step, similar to (3.19). The process is performed in a loop in order to obtain the optimal engine torque for each time step, for each mode.

Once the optimal torque inputs for each drive cycle are calculated, the user can select which cycle and which operational mode to use and simulate the system. The PID feedback described in Section 4.2.6 is a way to implement uncertainty correction in the model, to force the system towards the reference.

When using Model Predictive Control, certain parameters can be changed in order to improve the performance. These consist of the receding horizon length and the weighting factor in the cost functions. Changing these values will change the outcome of the system, and trial and error brings around the best set of parameters. These are discussed in the next section.

5 Parameter Selection and Simulation Results

Due to the different variables available to change in this simulator, each one had to be studied individually to determine the optimum set. This chapter will first discuss the parameters related to MPC as well as to the control algorithm used in the simulator; then, simulation results for each drive cycle will be provided and compared.

5.1 MPC Parameter Selection

As stated in Section 4.3, the parameters to vary when using MPC are the receding horizon length, the weights in the cost functions, and the PID parameters for uncertainty correction in the engine algorithm. Changes to all three are studied in the following sections. In order to simplify the process, only one drive cycle was used to compare the horizons, and that is the US06 drive cycle. The initial battery SOC was set at 70%, so that the vehicle would remain in MPC operating mode.

5.1.1 Receding Horizon

The receding horizon length will depend on your system, and what type of changes happen along the reference path. In this research, each step into the horizon is one second further into the drive cycle. Horizon lengths of 2, 3, and 5 were analyzed, by looking at how well the engine torques followed the fuel consumption or efficiency rules. The results are shown below for the US06 drive cycle, in Fuel Use Mode. Figure 29 shows the engine operating points with a horizon of 5 seconds.

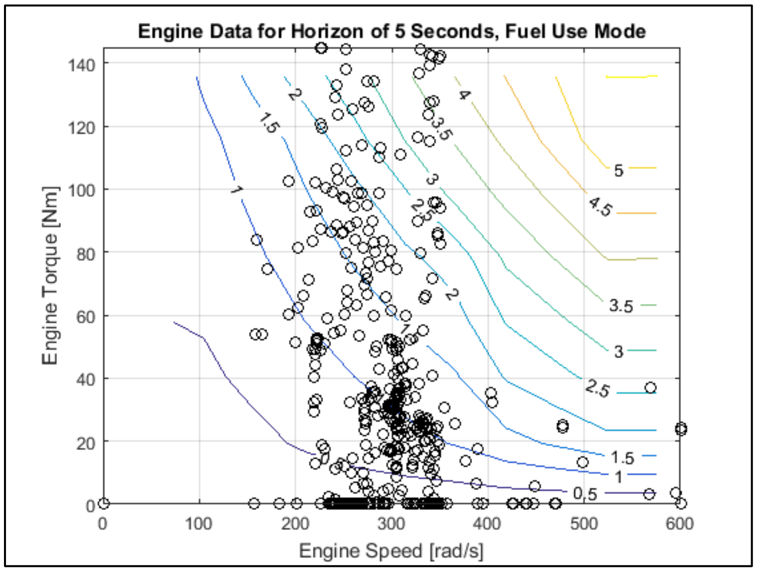


Figure 29: ICE Operating Points for N=5

The result shows that there are too many outlier points that are not close to following the line. The fuel economy of this case was 48.69 mpg. The total error from 1 gram per second fuel consumption over the whole cycle is 275.91, which shows that much improvement can be done to bring the engine points closer to the reference line. The error is calculated using the equation below:

$$e = \sum_{i=1}^{t_{sim}} (fc_i - 1)^2 \tag{5.1}$$

The next plot shows the results for a horizon of 3 seconds.

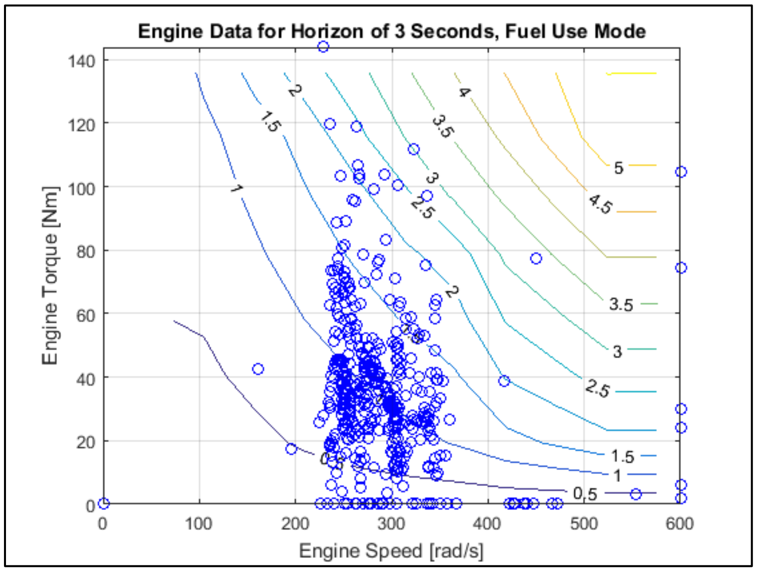


Figure 30: ICE Operating Points for N=3

This seemingly is an improvement, as more points are gravitating toward the reference line than the previous case. The fuel economy for this run was 48.90 mpg. The total error from the reference line is 109.08. In an attempt to improve the precision, a horizon of 2 seconds was considered, which is shown in Figure 31.

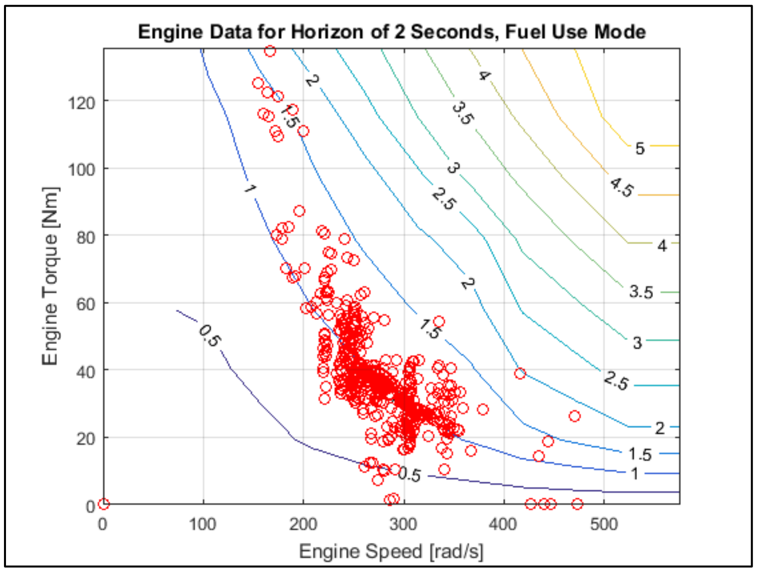


Figure 31: ICE Operating Points for N=2

This is the best of the three horizons, as it has the most data points around the 1 gram-per-second reference, and the ones that are not on it are much closer to it than in previous cases. The total error from reference for this case is 16.72, which is a substantial improvement in performance. The vehicle obtained 46.94 mpg. While the error is getting smaller, the fuel economy is as well. This makes it seem as though the shorter horizons result in worse fuel economy; however, the reason for this decrease in fuel economy is that the engine points operate closer to the reference, and there are fewer outliers producing a small or zero torque resulting in very low fuel consumption, which would be falsely improving fuel economy. The result shows that the best horizon to choose would be 2 seconds, and this will be used from now on in this research. The trend seen in the three plots above is that the further the horizon extends, the worse the fuel consumption results show. This can be explained by the fact that the drive cycles used for vehicle analysis change quickly, and looking too many steps ahead will cause the controller to compensate for long term actions and miss some upcoming actions. It is important to note that a horizon of 1 second was not chosen, since this could be achieved with a simple lookup table and no control. This can be attributed to the quasi-static nature of the engine model being used, in contrast to a dynamic model.

5.1.2 Cost Function Weight

Included in the cost function is the input control. This puts a constraint on the control input, in order to satisfy the optimization problem with minimum control. The control term is multiplied by a weight λ, which is used to dictate how much emphasis will be put on the control term as compared to the error term. Different values of the weights were studied and analyzed in this research in order to pick the best selection.

For Fuel Use Mode, the values tested were $\lambda_1 = 0.001, \lambda_1 = 0.0001$, and $\lambda_1 = 0.00001$. For each case, the optimal torques were calculated offline and the process was simulated with the complete system, with all other parameters fixed. The engine operating points for $\lambda_1 = 0.001$ are shown in Figure 32.

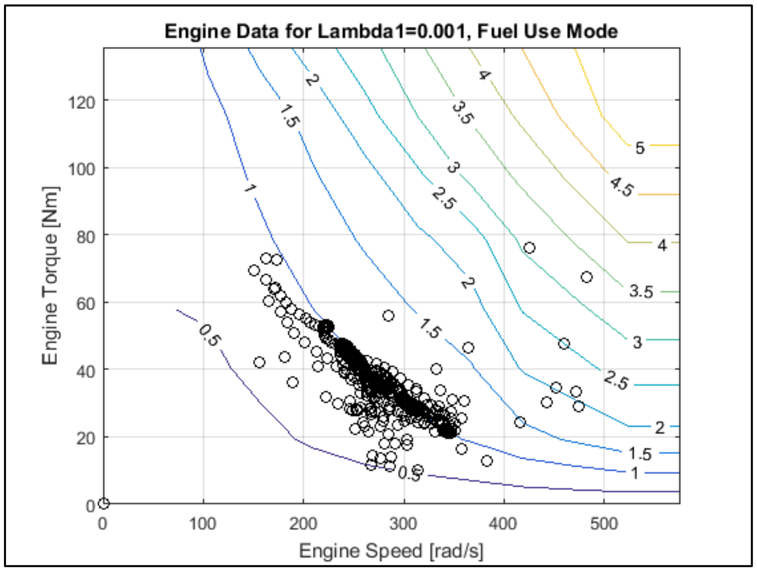


Figure 32: ICE Operating Points for Lambda1=0.001

The fuel economy in this case is 46.94 mpg. The error from the reference line over the complete cycle is 20.33. The data seems low in fuel consumption, but visually it can be improved to a closer fit of the reference. The second choice was a value of 0.0001, which is shown in Figure 33:

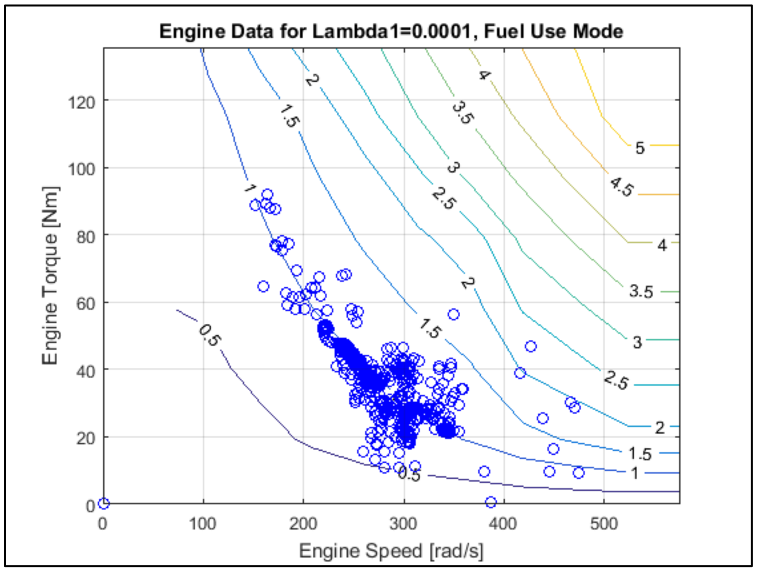


Figure 33: ICE Operating Points for Lambda1=0.0001

Figure 33 above shows an improvement from the results in Figure 32, and this is verified by the total fuel consumption error being only 12.35 over the whole cycle. The fuel economy for this case is 46.85 mpg, which is a slight decrease but acceptable since the control law is followed better. The last value to test for this operating mode is 0.00001. The results are in Figure 34:

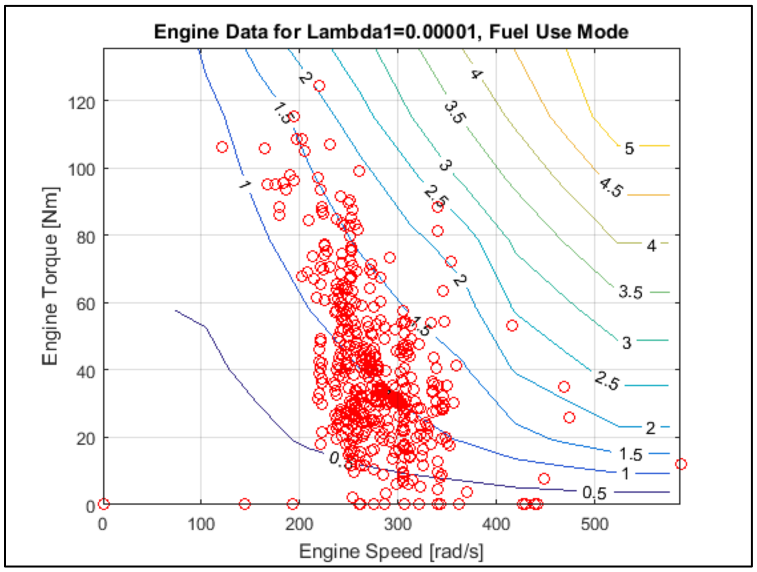


Figure 34: ICE Operating Points for Lambda1=0.00001

Visually, this seems like the worst of the three weights. Mathematically this is confirmed, as the total fuel consumption error is 54.39. The fuel economy is 46.85 mpg. Based on the analysis of these three weight values, the best value for the weight is $\lambda_1 = 0.0001$. This will be used for the rest of the simulations.

In the case of Efficiency Mode, where the goal is to have the engine operate on or near maximum thermal efficiency, the cost function weight was tested with values $\lambda_0 = 0.00001, \lambda_0 = 0.000001,$ and $\lambda_0 = 0.0000001$. The first value provided the engine points in the plot of Figure 35:

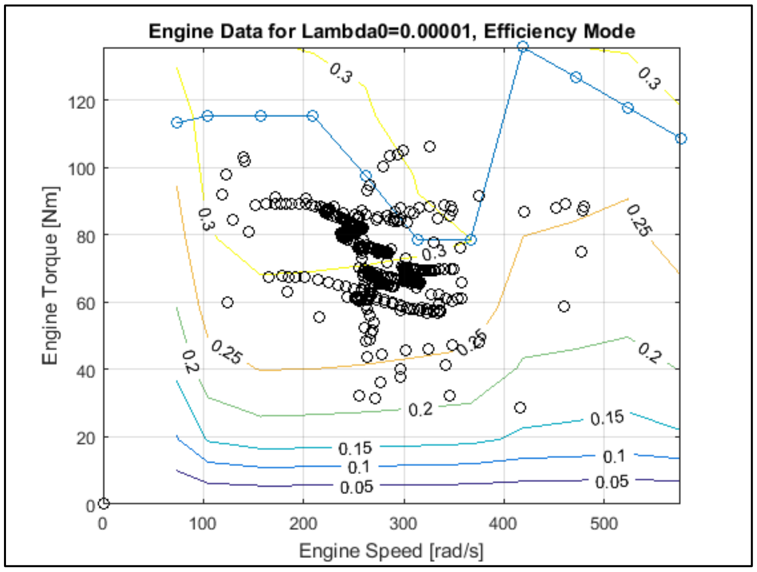


Figure 35: ICE Operating Points for Lambda0=0.00001

It is clear visually that this value for λ_0 is not sufficient, as many engine points do not fall near the maximum efficiency line. The error from maximum efficiency as calculated by

$$e = \sum_{i=1}^{t_{sim}} (eff_i - eff_{\max,i})^2 \tag{5.2}$$

In this case, the error for the cycle is 0.3491. The next choice, 0.000001, is shown in Figure 36:

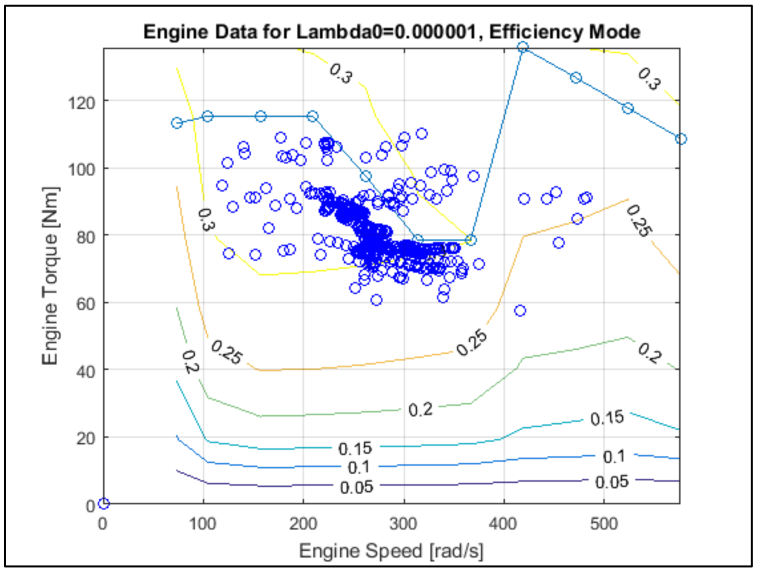


Figure 36: ICE Operating Points for Lambda0=0.000001

Clearly, this value for the weight is more successful at operating the engine close to the maximum efficiency. The error over the cycle here is 0.1088, which is much smaller. To try to achieve better performance, the next value was tested:

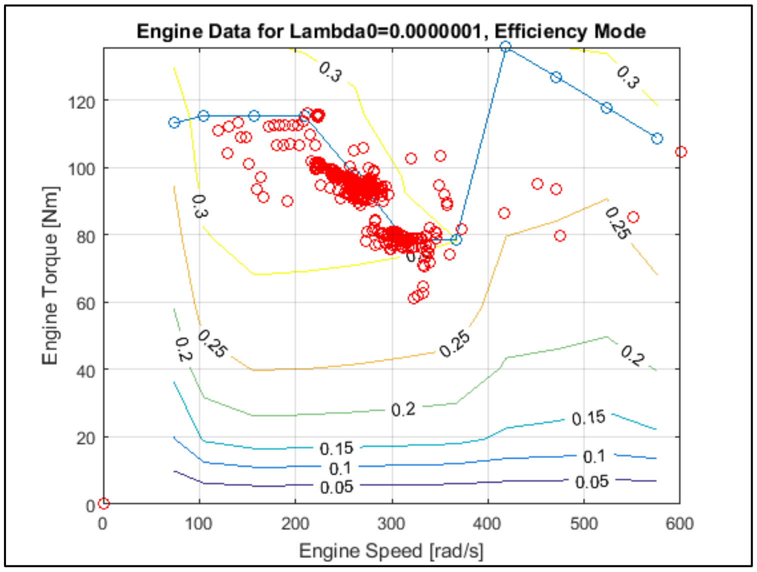


Figure 37: ICE Operating Points for Lambda0=0.0000001

This case follows the maximum efficiency line much better than the previous two. Mathematically this is verified with the error being only 0.0283. Based on this analysis, the optimal value for the Efficiency Mode weight is $\lambda_0 = 0.0000001$. This value is used for the duration of the thesis.

5.1.3 PID Uncertainty Correction

In the internal combustion engine subsystem, PID is used as a form or uncertainty correction for the MPC algorithm. Since the control input is calculated offline and with a prediction model, there will be some error and uncertainty in the system when implementing the control on the engine. The PID values used must be tuned for the best performance. Since there are two operational modes for the simulator, there are two sets of PID gains to tune. Both performances will be discussed.

When operating in Fuel Use Mode, the error is between a fuel consumption of 1 gram/sec and the fuel consumption of the model at the previous time step similar to the disturbance described in Section 3.2.1. The proportional, integral, and derivative gains were varied in order to find the best response and tracking. The starting values for each gain were set at 1. The engine data obtained are shown below:

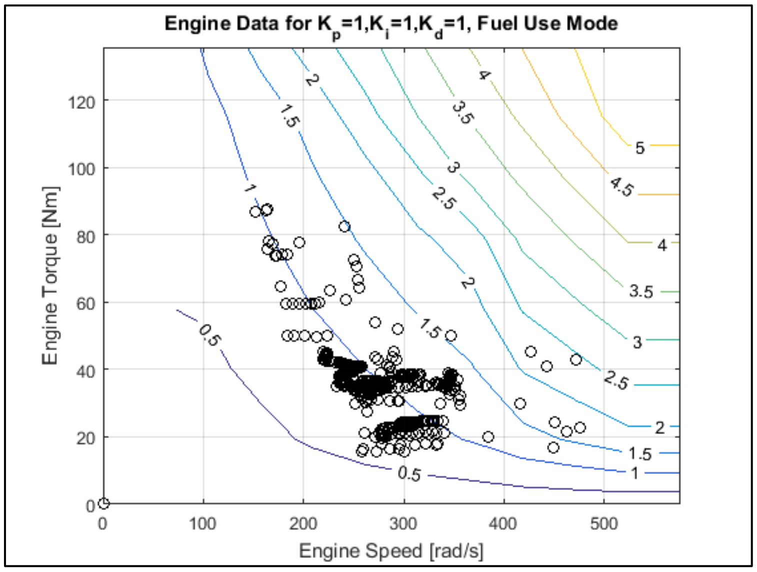


Figure 38: ICE Operating Points for Kp=1,Ki=1,Kd=1, Fuel Use Mode

Clearly there is room for improvement, as many of the data points are scattered. Table 7 shows different simulation runs with different gain values and the corresponding errors in fuel consumption.

Table 7: PID Uncertainty Correction Gain Selection, Fuel Use Mode

K_p	K_i	K_d	Error	MPG
1	1	1	16.42	47.40
0.001	1	1	16.67	47.39
10	1	1	15.04	47.51
10	0.001	1	21.06	54.08
10	10	1	11.36	46.89
10	10	0.001	11.39	46.89
10	10	10	13.00	46.8
10	20	1	11.88	46.89
15	10	1	11.62	46.90
5	10	1	11.50	46.86
8	10	1	11.35	46.88

As the gains were adjusted, different values were corresponding to an increase or decrease in the error. The selected values for the PID correction in Fuel Use Mode are $K_p = 8, K_i = 10,$ and $K_d = 1$. The engine data for the final run is shown in Figure 39.

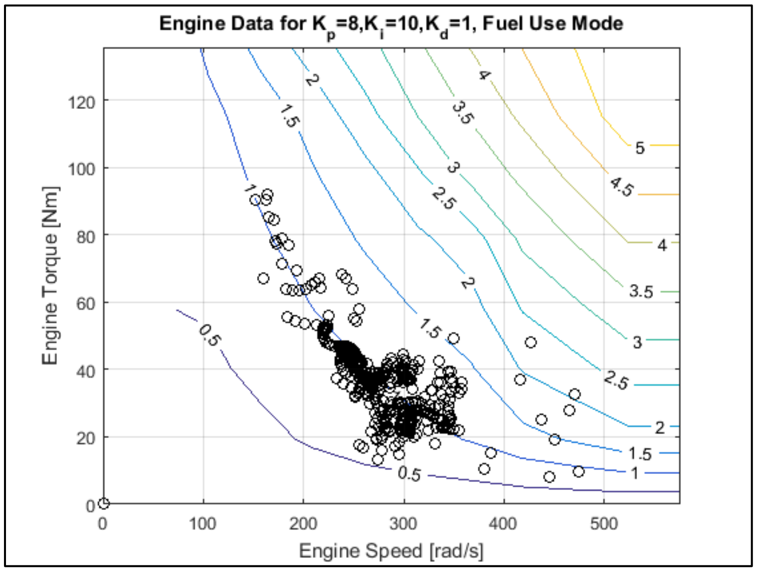


Figure 39: ICE Operating Points for Kp=8,Ki=10,Kd=1, Fuel Use Mode

The reference tracking in this case is very good, as the data points run close to the 1 gram/sec line of the engine's fuel consumption plot. There are very few values that land far away from the reference line. These gain values are used for Fuel Use Mode for the rest of the thesis.

The same process is followed for Efficiency Mode, to move the engine operating points towards the maximum efficiency line. Starting at 1 for each gain value, the results are shown in Figure 40:

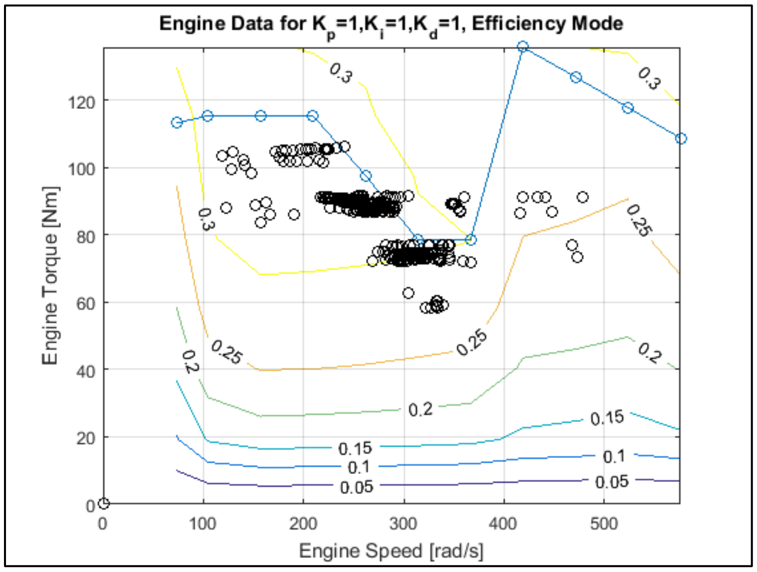


Figure 40: ICE Operating Points for Kp=1,Ki=1,Kd=1, Efficiency Mode

The majority of data points are in the range of maximum efficiency, and for the most part near the region of 30% efficiency. It is worth exploring different values of PID gains to improve the performance. Many sets of values were tested, and a table of the results is provided:

Table 8: PID Uncertainty Correction Gain Selection, Efficiency Mode

K_p	K_i	K_d	Error	MPG
1	1	1	0.0656	26.83
0.001	1	1	0.0657	26.84
10	1	1	0.0649	26.81
10	0.001	1	0.0882	27.44
10	10	1	0.0755	26.31
10	1	0.001	0.0649	26.81
10	1	10	0.0648	26.81
100	1	10	0.0589	26.63
500	1	10	0.0488	26.75
1000	1	10	0.0331	27.90
1000	4	10	0.0290	27.43

Based on the analysis above, the best gain values prove to be $K_p = 1000, K_i = 4,$ and $K_d = 10$. This set of values results in the smallest error from maximum efficiency over the duration of the cycle. The engine data for this set is shown in Figure 41.

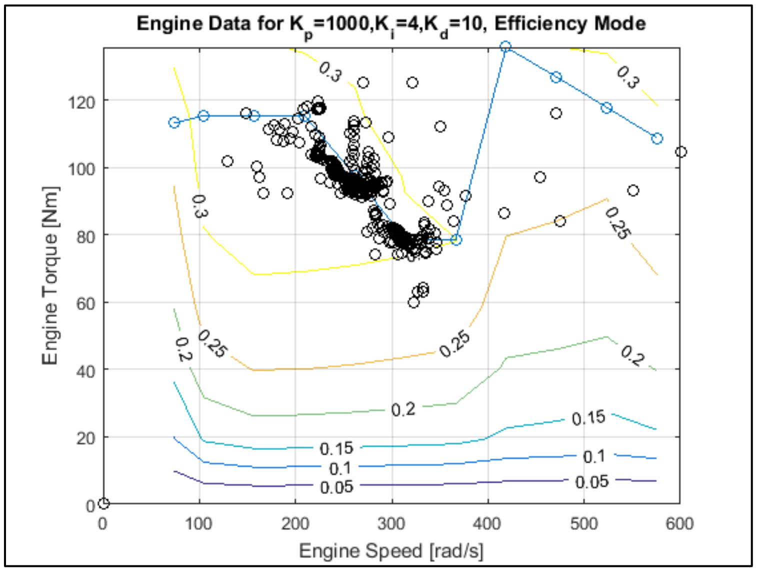


Figure 41: ICE Operating Points for Kp=1000,Ki=4,Kd=10, Efficiency Mode

This is clearly an improvement from the initial values, as the engine points are closer to the maximum efficiency line. There are a few points that are off from the efficiency line, but this is the best performance of the system through testing. These values are used for the rest of the Efficiency Mode calculations in this thesis.

5.2 Simulation Results

With all the Model Predictive Control parameters chosen, and the ICE torque requests calculated offline, the vehicle model can be simulated and analyzed. The simulations were done for two drive cycles: US06 and FUDS. The US06 Supplemental Federal Test Procedure is a high speed cycle with some hard acceleration, while the Federal Urban Driving Schedule is used to test city driving. For each drive cycle, the initial states of charge of 85%, 50%, and 15% were tested, for both Fuel Use Mode and Efficiency Mode. The metrics to analyze for each simulation run are:

- How well the model speed follows the drive cycle speed

- The final MPG and MPGe of the model
- Engine data points for fuel consumption and efficiency
- The powertrain operation modes based on SOC
- SOC charging and discharging characteristics

These metrics will be discussed for the drive cycles specified, and then compared between the drive cycles themselves.

5.2.1 Fuel Use Mode Results

The goal for Fuel Use Mode is to limit the engine's fuel consumption in order to improve fuel economy. This is done, as explained in previous sections, by forcing the engine to operate on or near the 1 gram-per-second fuel consumption line. There are instances where this is temporarily put on hold, due to the need for the engine to charge the battery pack. The results below show the different ways the powertrain will operate due to different states of charge.

For the US06 drive cycle, the velocity profile and the model's vehicle speed are shown in Figure 42 on the next page.

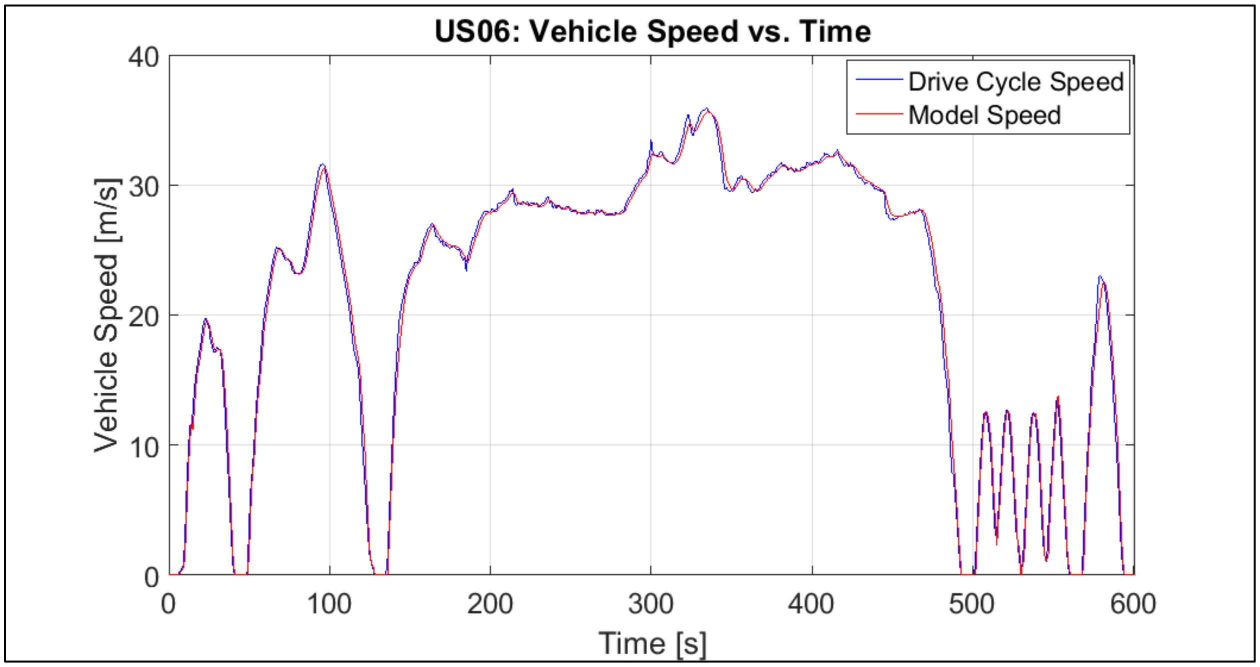


Figure 42: US06 Drive Cycle Tracking for Fuel Use Mode

The model speed follows the commanded drive cycle speed very closely, with slight errors at some drastic speed changes. The metrics to discuss for Fuel Use Mode are the fuel consumption data points, battery SOC, and vehicle fuel economy. A plot of the engine operating points for 85% initial SOC is shown in Figure 43:

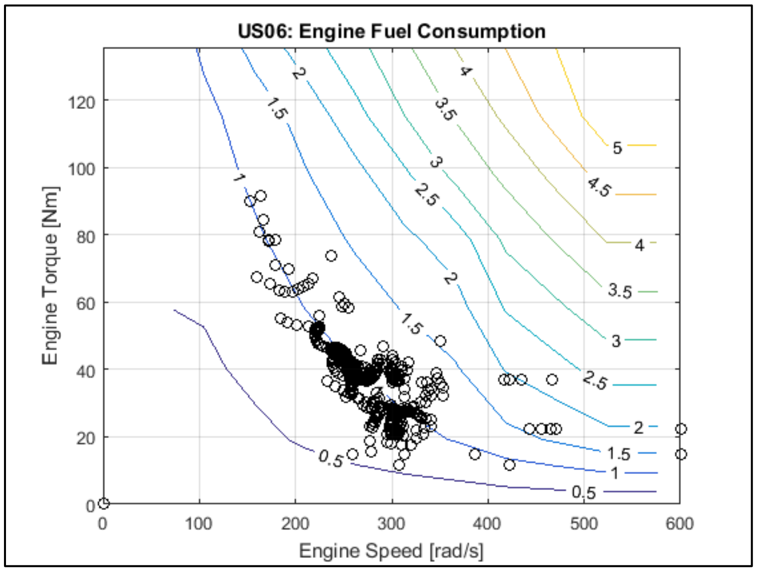


Figure 43: ICE Points for US06 with Initial SOC 85%, Fuel Use Mode

The engine operating points clearly follow the reference line. This shows that the control algorithm applied is successful. In order to analyze the ability of the powertrain to make use of the battery, the state of charge is plotted for the same run, along with the powertrain operating modes:

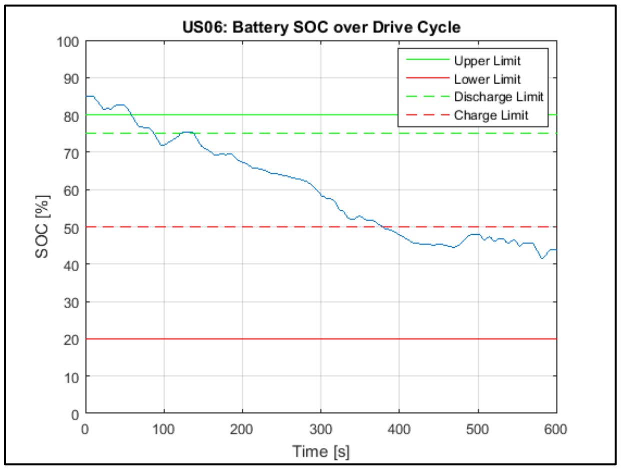


Figure 44: State of Charge Path for US06 with Initial SOC 85%, Fuel Use Mode

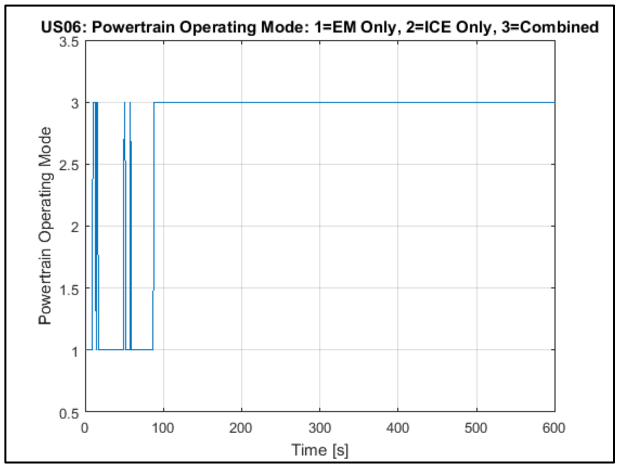


Figure 45: Powertrain Modes for US06 with Initial SOC 85%, Fuel Use Mode

The battery SOC goes from 85% down to a final SOC of 43.9%. Since the initial charge is above the threshold of 80% as described in Section 4.2.3, the powertrain starts with electric propulsion only. The powertrain switches to the combined torque-split mode whenever there is a hard acceleration, then switches back to EM only, until the SOC reaches the discharge limit of 75% at which time the combined mode is activated in full. The battery depletes for the most part of the cycle, as there is not much braking done to regain energy.

The next initial SOC tested was 50%. This is right in the middle of the battery limits, so the powertrain mode would start as a torque split. The engine points can be seen in Figure 46:

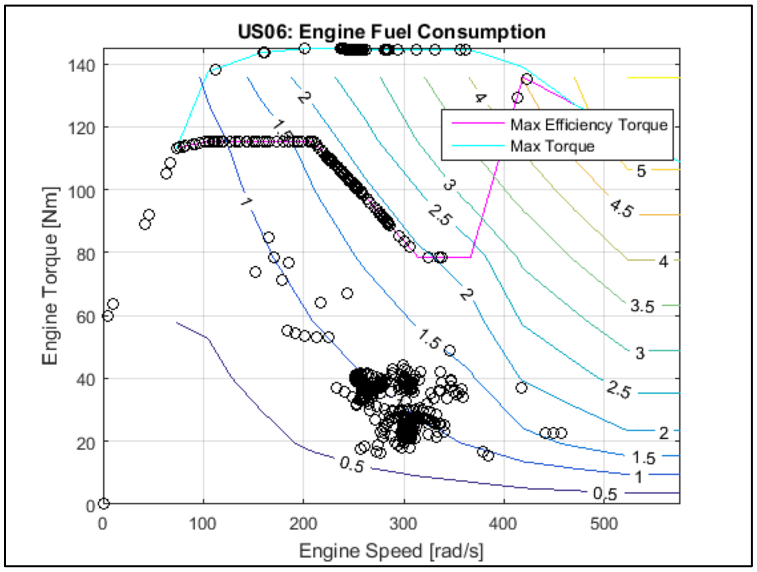


Figure 46: ICE Points for US06 with Initial SOC 50%, Fuel Use Mode

The SOC and powertrain modes are displayed next:

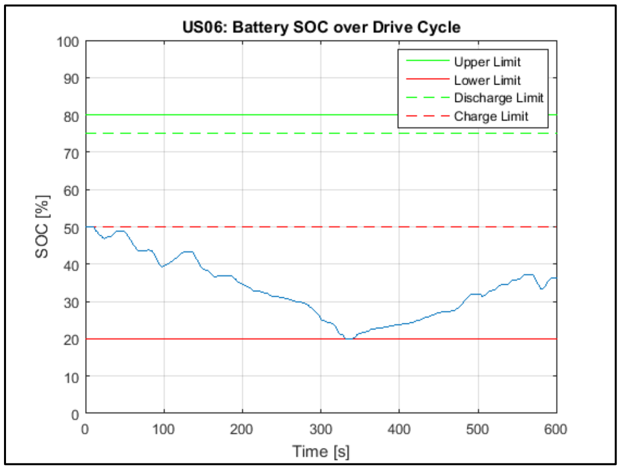


Figure 47: State of Charge Path for US06 with Initial SOC 50%, Fuel Use Mode

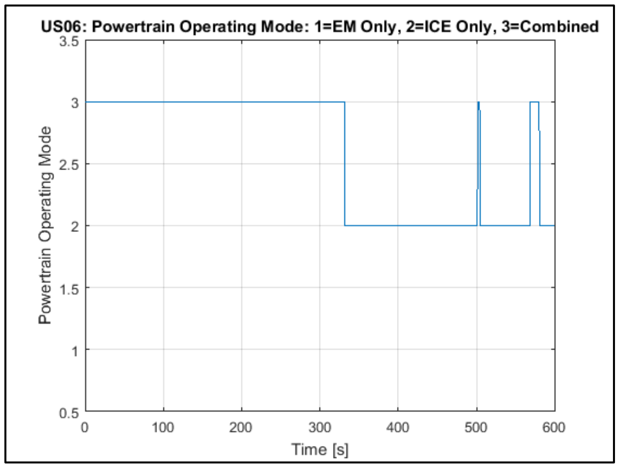


Figure 48: Powertrain Modes for US06 with Initial SOC 50%, Fuel Use Mode

When looking at the operating points of the ICE in Figure 46, a group of them are right along the 1 gram/sec line as desired. However, there are points that follow the maximum efficiency curve, and a few on the maximum torque line as well. Looking at Figure 47 and Figure 48, it is shown that the SOC hits 20%; at this point, the powertrain enters internal combustion engine mode, in order to charge the battery pack. The group of data points that follow the maximum efficiency and maximum torque lines are from this section of the drive cycle. The charging mode executes as planned; since the SOC never returns to above 50%, the powertrain does not revert to regular MPC operation.

The final simulation for US06 in Fuel Use Mode was a starting charge of 15%. This is below the lower limit of 20%, so the powertrain starts in charging mode. The engine operating points can be seen in Figure 49:

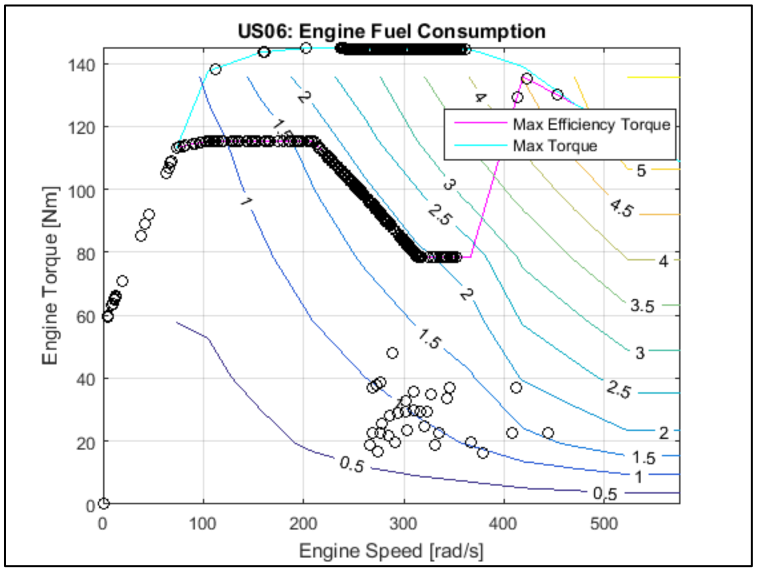
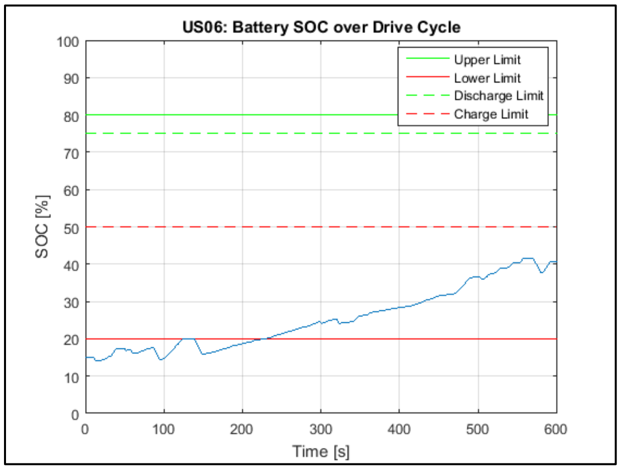


Figure 49: ICE Points for US06 with Initial SOC 15%, Fuel Use Mode

Here, there are few data points following the reference line, and many others following maximum efficiency torque or maximum torque. This can be explained through the plots for the SOC and powertrain mode:



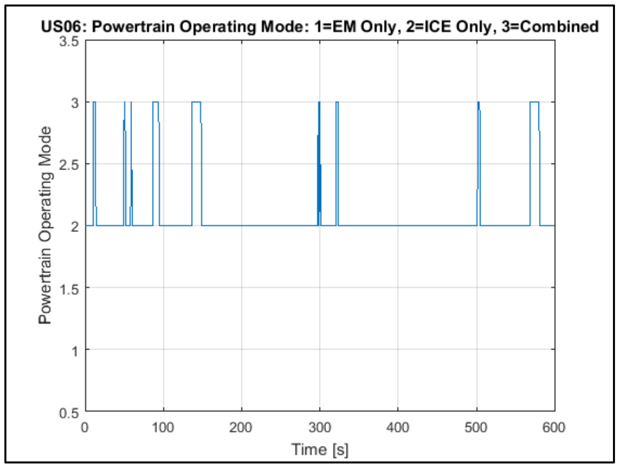


Figure 50: State of Charge Path for US06 with Initial SOC 15%, Fuel Use Mode

Figure 51: Powertrain Modes for US06 with Initial SOC 15%, Fuel Use Mode

As the internal combustion engine charges the battery pack, the SOC increases. During this time, the engine is being operated at its maximum efficiency, or if more torque is required to propel the vehicle, at maximum torque. This corresponds to Mode 2 of Figure 51, and the engine points on the maximum efficiency and maximum torque lines of Figure 49. The only cases of rapid switch between operational modes are when there is hard acceleration from the driver model. If you compare these jumps from Mode 2 to Mode 3 in Figure 51, they correspond to areas of strong acceleration in the drive cycle diagram of Figure 42. After the quick jumps to the combined mode, the powertrain returns to ICE only mode in order to finish charging the ESS. The battery SOC never reaches the charge-up value of 50% to fully enter MPC operation. This signifies a strong hybrid, since the vehicle can operate in the full range of battery SOC.

To show the effects of the initial state of charge on the performance of the vehicle, the fuel economy for each case must be calculated and compared. This is done with the table below:

Table 9: Performance Comparison for US06 with Different Initial SOC, Fuel Use Mode

Initial SOC	Final SOC	MPG	MPGe
85%	43.9%	52.1	39.9
50%	36.3%	30.6	28.9
15%	40.8%	20.3	21.9

When the state of charge is high, specifically above the upper limit, the fuel economy of the vehicle is the highest; this is because for a portion of the drive cycle, the ICE is shut off and the electric motor is the only propulsion device. The MPGe value is 12.2 lower than the conventional MPG value, and this is because the electrical energy used by the battery is adding to the equivalent fuel used by the system. The difference in MPG and MPGe is less for the second case, and this is because the ICE is taking more of the load during the complete cycle. The fuel economy for this case is lower than that of the 85% SOC case, due to the fact that the powertrain is in torque split mode for roughly half of the cycle, then there is a time of the drive simulation where the ICE switches to operating in charging mode, which will increase fuel consumption. In the final simulation run, since the system starts at a low SOC, the battery gets charged up and the final SOC is actually higher than the initial. The tradeoff to this is that the fuel economy of the system decreases drastically, since the engine is operating at higher fuel consumption rates for the majority of the drive cycle. In this case, the MPGe is slightly higher than the MPG, due to the fact that the continuous charging of the batteries is providing a negative current to the ESS, thus offsetting some of the equivalent fuel used.

The next drive cycle to be studied in Fuel Use Mode is the Federal Urban Drive Schedule, or FUDS. This urban driving cycle has lots of stop-and-go, which simulates city driving. The velocity profile along with the model's trace can be found in Figure 52:

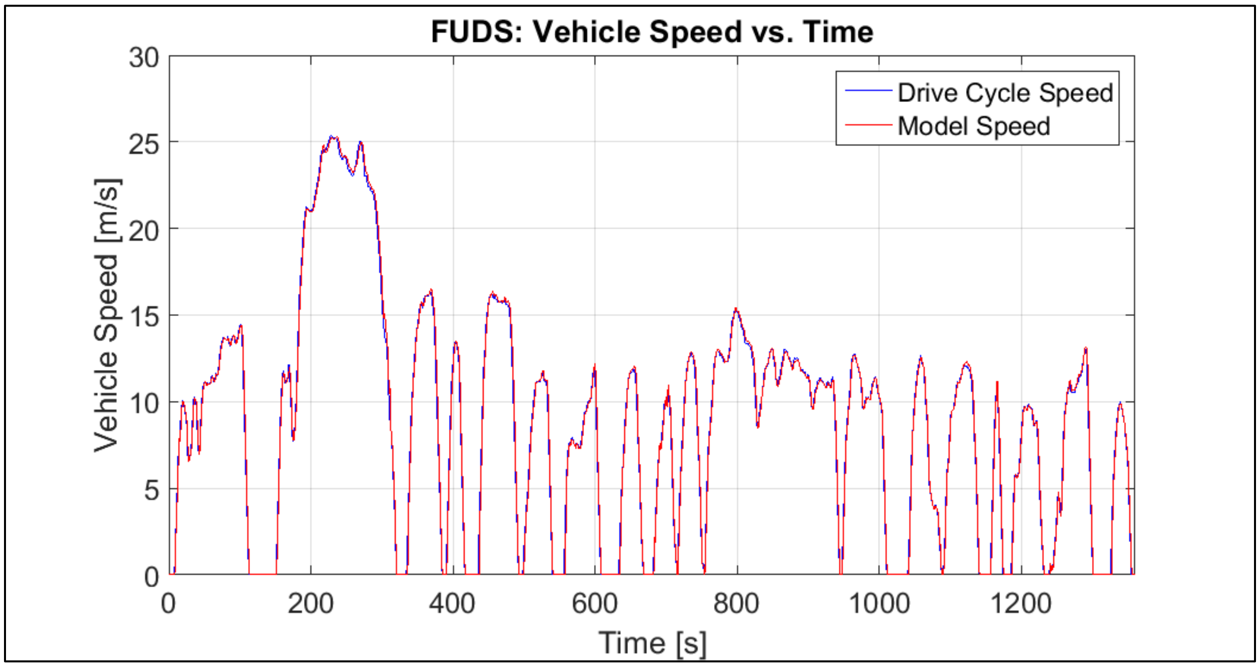


Figure 52: FUDS Drive Cycle Tracking for Fuel Use Mode

The Simulink model does an excellent job at following the drive schedule. The simulation metrics to check are the same as the US06 case above. For the initial SOC of 85%, the engine points obtained are shown:

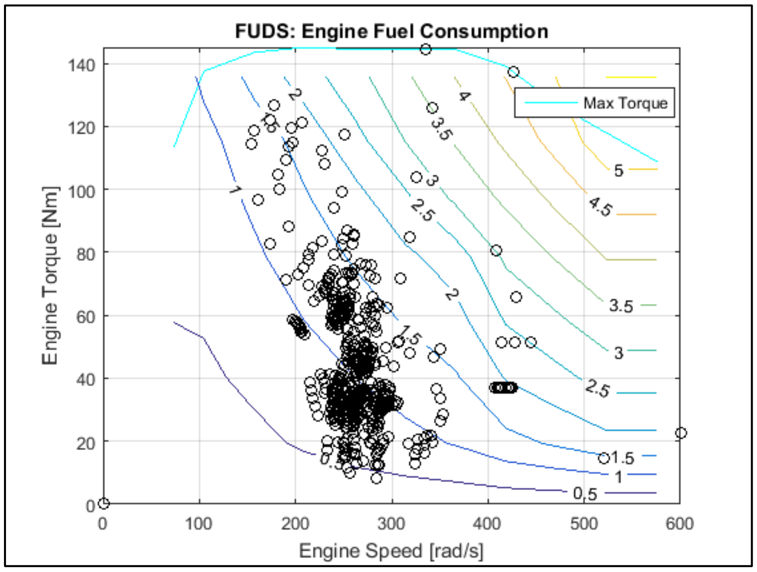


Figure 53: ICE Points for FUDS with Initial SOC 85%, Fuel Use Mode

For this cycle, the points are still gathered around the 1 gram/sec line, but there are much more scattered points. This is due to the fact that this cycle has a lot of stop-and-go action, which means there are lots of drastic changes in speed and acceleration, therefore making it more difficult to control the fuel use. The state of charge and mode of operation plots are displayed next:

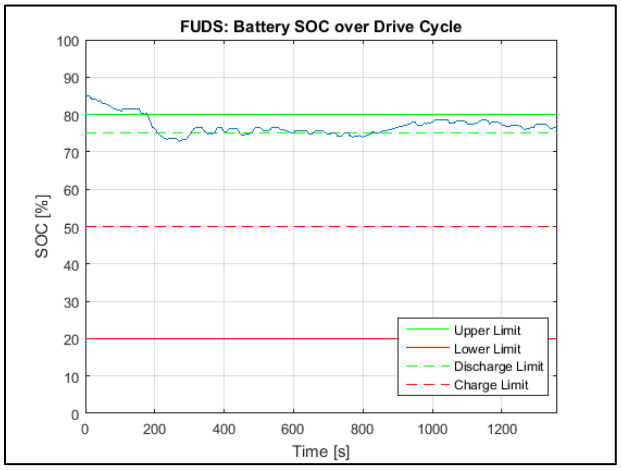


Figure 54: State of Charge Path for FUDS with Initial SOC 85%, Fuel Use Mode

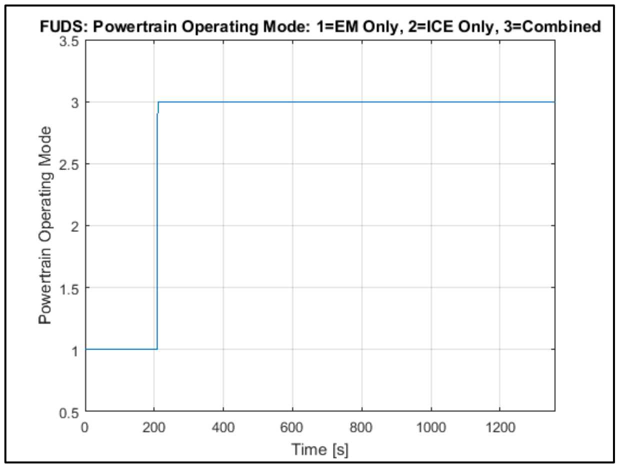


Figure 55: Powertrain Modes for FUDS with Initial SOC 85%, Fuel Use Mode

This cycle is shown to do well at sustaining charge at a high limit, since the drastic braking commands result in regenerative braking producing electrical generation. Even when the powertrain enters combined propulsion mode, the SOC increases and stays above 75%.

The model is simulated again with the initial SOC changed to 50%. This is the middle of the battery charge region, and will start with the combined propulsion mode. The engine data points are displayed in Figure 56:

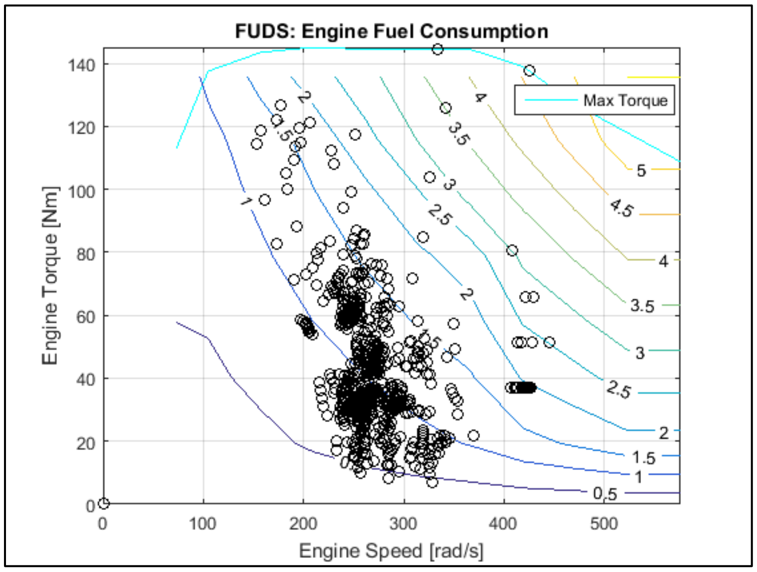


Figure 56: ICE Points for FUDS with Initial SOC 50%, Fuel Use Mode

This plot looks similar to the fuel consumption plot of Figure 53 for the initial SOC of 85%; the difference is that here there are more points, since the powertrain never enters the electric propulsion mode, as shown in Figure 58. The majority of the points are situated around the reference line. The SOC for this system has an interesting trajectory:

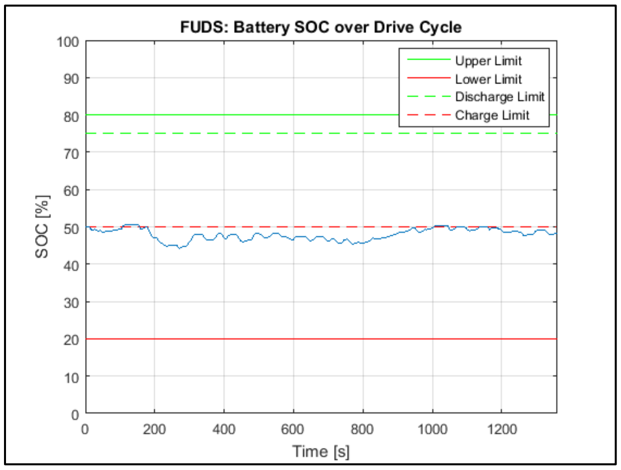


Figure 57: State of Charge Path for FUDS with Initial SOC 50%, Fuel Use Mode

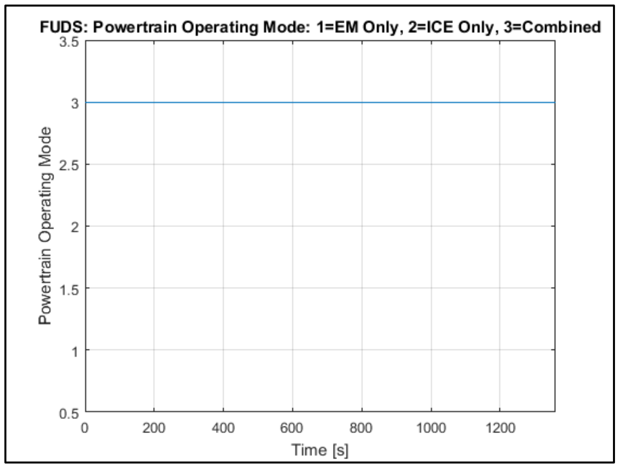


Figure 58: Powertrain Modes for FUDS with Initial SOC 50%, Fuel Use Mode

The charge stays relatively consistent below 50% for the beginning of the drive cycle. Then, around 800 seconds, the SOC increases and stays around 50% for the remaining time. Looking at the drive cycle at around 800 seconds, there portion where the vehicle speed is relatively constant with a slight decrease, and no hard acceleration. Situations like this is where the battery charges well since there is not much motor torque demand. Since the SOC stays in the nominal operating range, the powertrain remains in the combined mode with the MPC algorithm. It is important to note that having the powertrain in combined mode specifies the ability for the powertrain to operate with both the electric motor and the ICE; this does not mean that both propulsion devices are being used at all times. As stated in Section 4.2.3, in combined mode the vehicle will be in electric operation when the speed is under 10 m/s.

The operation of FUDS with a starting charge of 15% is slightly different than the previous runs:

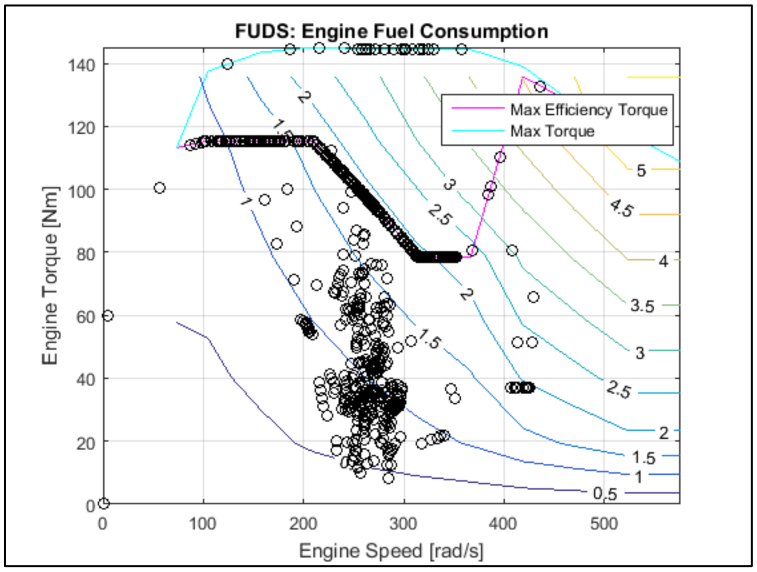


Figure 59: ICE Points for FUDS with Initial SOC 15%, Fuel Use Mode

Based on the fuel consumption plot, it is easy to see that the engine operates around the 1 gram/sec line but also on the maximum efficiency line, with some data points on the maximum torque line. Due to the algorithm implemented in this thesis, this must mean that the engine has been charging the battery pack. This is verified with the SOC trajectory and the powertrain mode plot:

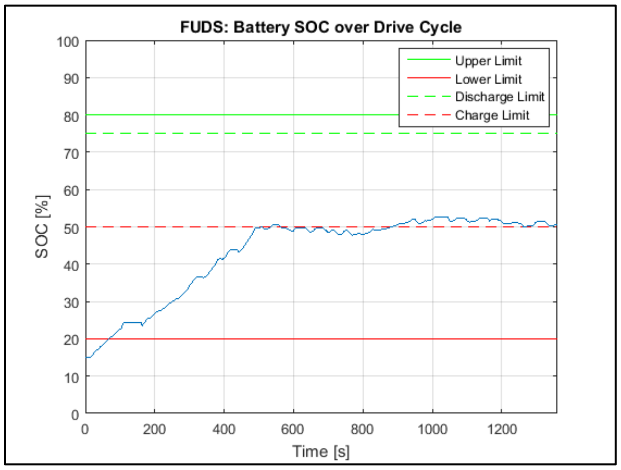


Figure 60: State of Charge Path for FUDS with Initial SOC 15%, Fuel Use Mode

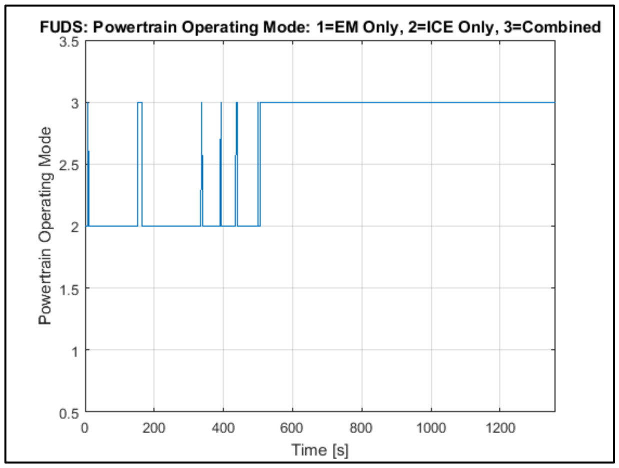


Figure 61: Powertrain Modes for FUDS with Initial SOC 15%, Fuel Use Mode

Since the ESS starts with an initial charge of 15%, the powertrain starts in Mode 2, which is the charging mode. However, the engine charges the ESS and the controller eventually shifts to Mode 3, when the state of charge becomes 50%. After this instant, the battery stays within the normal operating mode, due to the regenerative braking not allowing the ESS to deplete more than a few percent.

To compare the fuel economy performance for each of these three simulations, a table is created with the results from all three simulations:

Table 10: Performance Comparison for FUDS with Different Initial SOC, Fuel Use Mode

Initial SOC	Final SOC	MPG	MPGe
85%	76.6%	35.6	33.9
50%	48.2%	29.6	29.4
15%	50.5%	19.3	21.5

The fuel economy for the first case is higher than other two, since there are portions of the drive cycle that the ICE is off. As the initial state of charge decreases, Table 10 shows that the MPG decreases. This is because the SOC is directly correlated to how much torque the engine provides in Fuel Use Mode. When in the combined powertrain operation, due to the many starts and stops, the ICE fuel efficiency is much lower than that of US06. When in the low SOC range, the engine fuel efficiency is low due to the fact that the battery needs to be charged. In the case of an initial SOC of 50%, the conventional and equivalent fuel economies are practically the same; this is due to the fact that the SOC does not drop much, so the amount of equivalent fuel lost is mostly gained back. In the last case of Table 10, the equivalent fuel economy is higher than the conventional, since the battery is getting charged in many instances of the FUDS cycle.

By looking at the two tables of simulation performance, one can determine that the US06 drive cycle yields higher MPG for all cases. This can be attributed to the highway driving profile, as the majority of the drive cycle is spent on a higher speed with few drastic changes in acceleration. That being said, the FUDS drive cycle performs much better at charging up the battery. The largest state of charge loss in the US06 cycle was 41.1%, coming from an initial SOC of 85%. The largest loss for FUDS was 8.4%, also for an initial SOC of 85%. The other two simulation runs for FUDS limited the loss in charge, and charged the battery to a higher SOC than at the start of the cycle. This can be attributed to the many instances where there is hard braking, which will charge up the battery over the drive cycle. Since Fuel Use Mode is most concerned with the limiting of fuel consumption, the US06 is considered as the more successful drive cycle.

5.2.2 Efficiency Mode

The main goal for the system in Efficiency Mode is to have the engine operate at or near its maximum efficiency curve. This is the curve shown in Figure 4. Unlike in Fuel Use Mode, the MPG in this analysis will not be the determining factor of whether the results are useful; when operating at maximum engine efficiency, the engine will consume more fuel, making the fuel efficiency go down. The importance here is more on sustaining the charge of the battery.

The first drive cycle to analyze is once again the US06 drive cycle, simulating highway driving. The vehicle response in Figure 62 shows that the model can follow the drive cycle in Efficiency Mode just as well as in Fuel Use Mode.

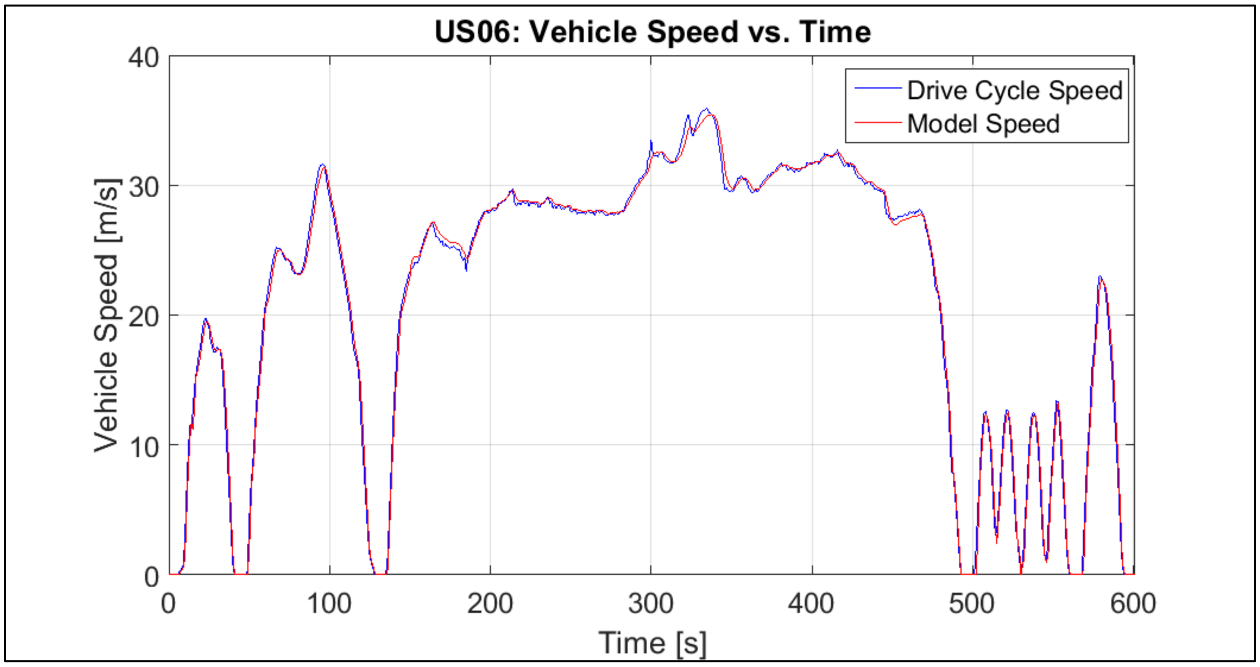


Figure 62: US06 Drive Cycle Tracking for Efficiency Mode

Once again, the first test performed was for an initial SOC of 85%. Since for this operating mode the idea is to operate the engine at its peak efficiency, the engine points are overlaid on the efficiency contour of the engine:

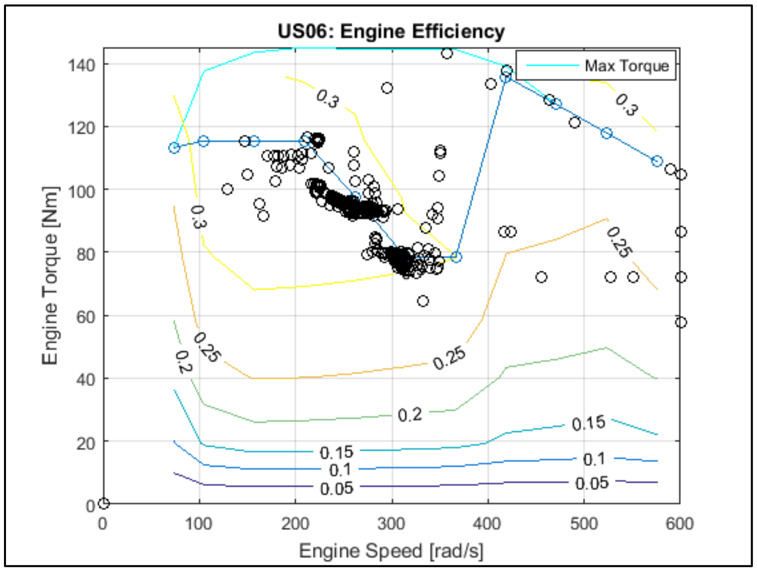


Figure 63: ICE Points for US06 with Initial SOC 85%, Efficiency Mode

Figure 63 above shows that the engine is following the desired control law rather well. The majority of points are clustered between 200 and 400 rad/s, which correspond to approximately 1910 and 3820 RPM, respectively. The plots for the SOC of the system and the powertrain decision making are shown in Figure 64 and Figure 65:

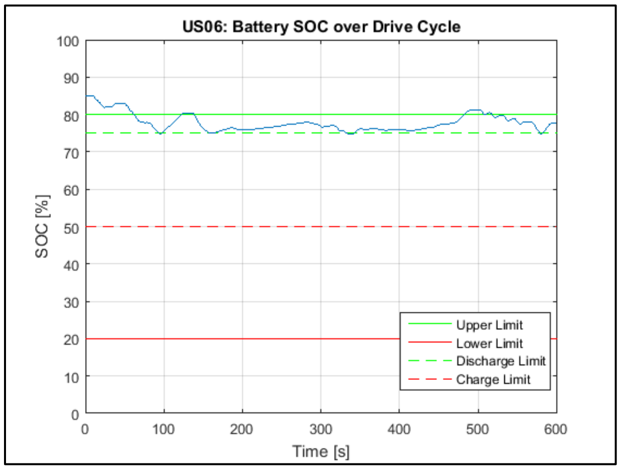


Figure 64: State of Charge Path for US06 with Initial SOC 85%, Efficiency Mode

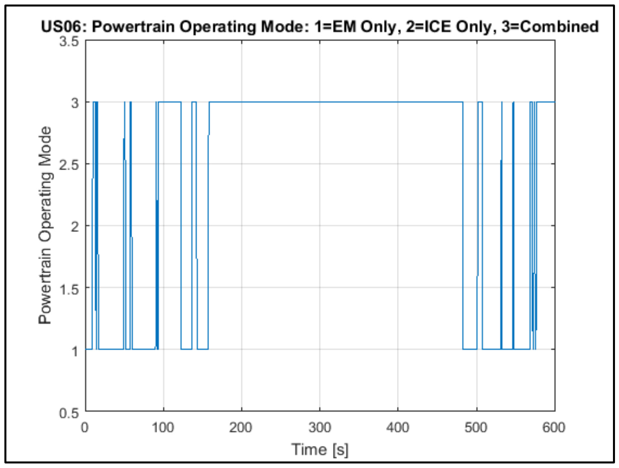


Figure 65: Powertrain Modes for US06 with Initial SOC 85%, Efficiency Mode

The state of charge, while starting at 85%, depletes to the discharge limit, but sustains the charge between 80% and 75% for duration of the cycle. Analyzing Figure 65 shows that the powertrain depletes charge and then charges back up in Efficiency Mode only to reach the limit and deplete the charge again. The heavy acceleration at the beginning and end of the US06 drive cycle constitute a need for a combined torque split mode momentarily, for a few instances while in electric only mode.

When the initial SOC is set to 50%, the engine does a great job of charging the energy storage system while following the efficiency curve. The engine data points can be seen in Figure 66:

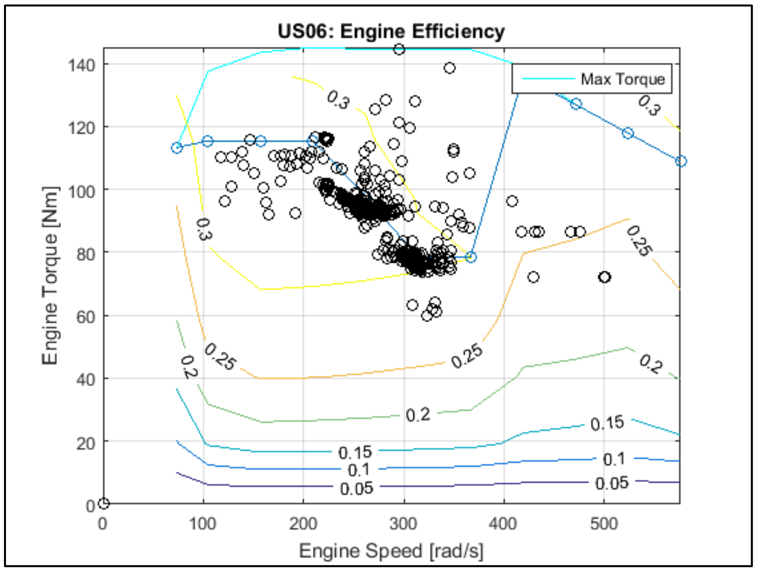


Figure 66: ICE Points for US06 with Initial SOC 50%, Efficiency Mode

The data points, while not as compact as the previous case, still follow the maximum efficiency trend. There are more data points that are scattered, because the powertrain operates solely with the MPC algorithm in Mode 3, as can be seen in Figure 68. This relates to the fact that for the beginning and end of the drive cycle, there are instances of hard acceleration and deceleration; when the initial SOC was 85%, these times operated in electric only mode, so the engine was off and the data points were plotted at the origin. However, in the current simulation run, the engine was operating at those instances, and during those starts and stops, it is more difficult for the engine to follow the maximum efficiency curve.

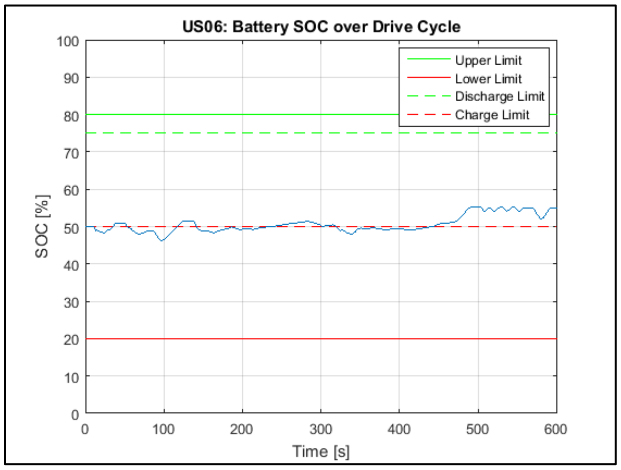


Figure 67: State of Charge Path for US06 with Initial SOC 50%, Efficiency Mode

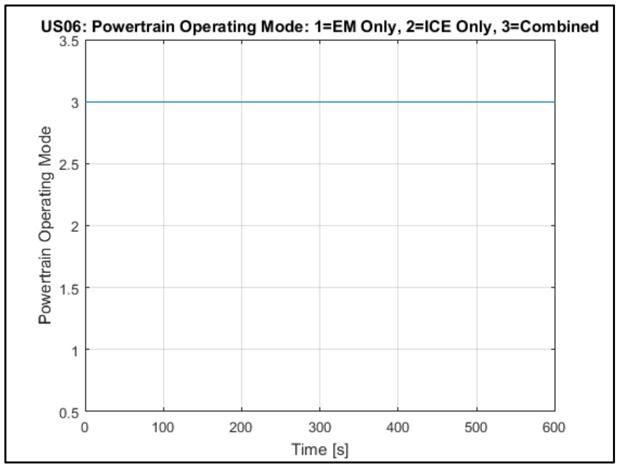


Figure 68: Powertrain Modes for US06 with Initial SOC 50%, Efficiency Mode

The state of charge for this case increases as the cycle goes on. This is expected since this is a highway cycle; the engine operating efficiently will mean there is less need for the electric motor to provide torque to the axle to assist the engine. This can be seen on the plot of electric motor torque over the duration of the drive cycle:

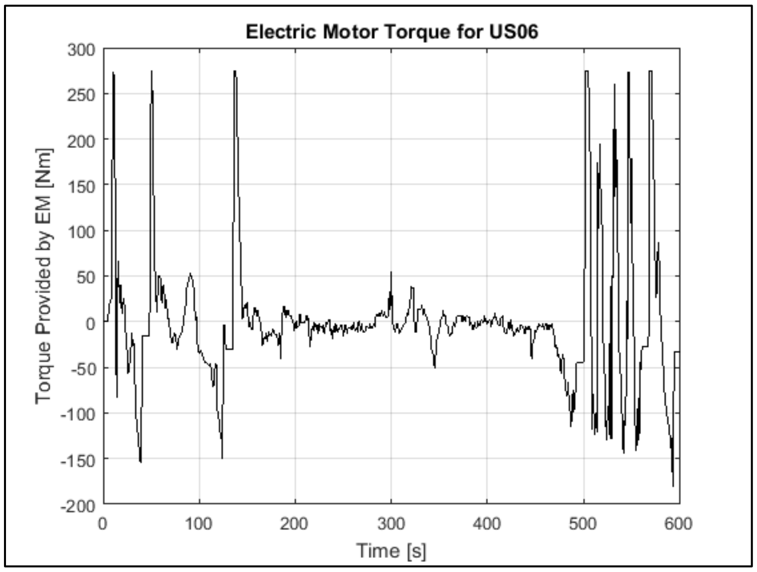


Figure 69: EM Torque for US06 Drive Cycle with Initial SOC 50%, Efficiency Mode

As is shown above, the middle of the drive cycle—the highway conditions—require very little to no torque assist from the electric motor. This means when the vehicle continues along the path, the momentary braking will result in a net flow of charge into the battery, which increases the SOC.

The final case to test for the current drive cycle is an initial ESS charge of 15%. The engine efficiency points are plotted in Figure 70.

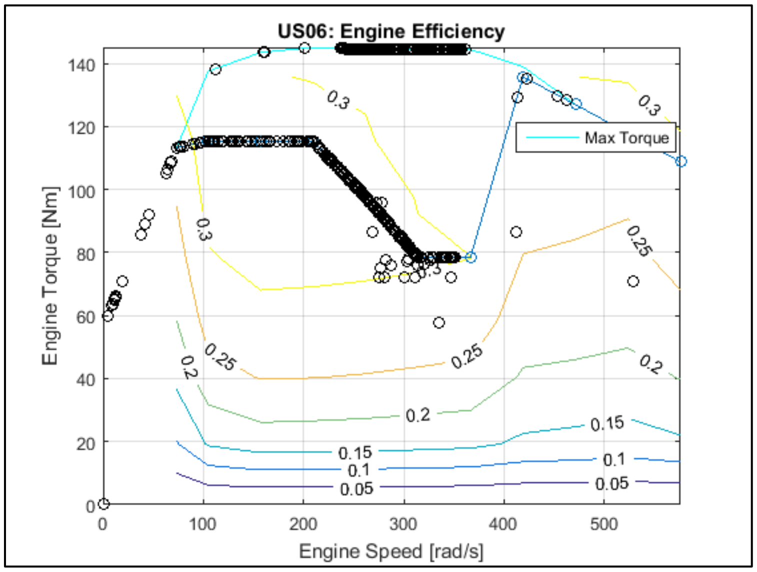


Figure 70: ICE Points for US06 with Initial SOC 15%, Efficiency Mode

There seems to be many points on the maximum efficiency line and others on the maximum torque line, signifying that the powertrain was in charging mode for a large part of the cycle. This is verified with the SOC and powertrain decision plots shown next:

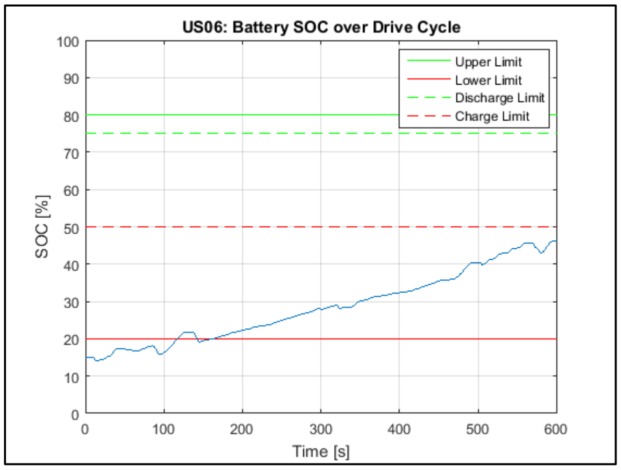


Figure 71: State of Charge Path for US06 with Initial SOC 15%, Efficiency Mode

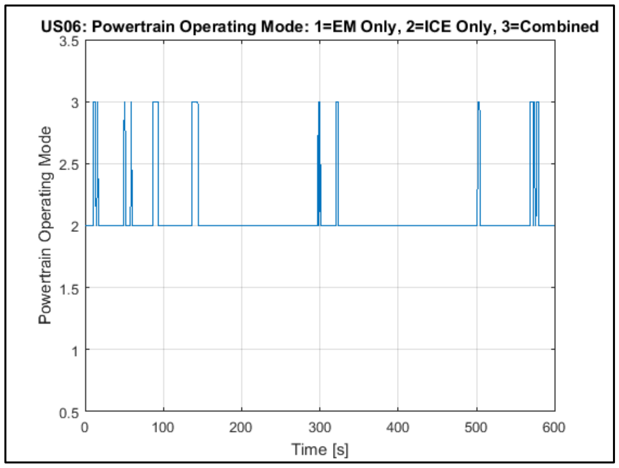


Figure 72: Powertrain Modes for US06 with Initial SOC 15%, Efficiency Mode

Since the ESS started with a charge of 15%, the powertrain started in charging mode. The charge gradually increases, but it never reaches the charge limit of 50%. The only times the powertrain operates in the torque-split mode is in instances of hard acceleration. Three cases can be compared in the following table:

Table 11: Performance Comparison for US06 with Different Initial SOC, Efficiency Mode

Initial SOC	Final SOC	MPG	MPGe
85%	77.7%	31.3	30.3
50%	55.0%	25.1	25.5
15%	46.1%	19.6	21.3

When looking at the changes in SOC for the US06 cycle, the smallest (and only) decrease in SOC is 7.3%, while the largest increase is 31.1%. As stated previously, this is the important metric to judge whether efficiency mode is working correctly; the charge is sustained very well in this drive cycle. The fuel economy is lower than the cases in Fuel Use Mode, but as explained, maximizing MPG is not the priority in this implementation of Efficiency Mode. One thing to note is that the equivalent fuel economy, MPGe, is higher than the conventional fuel economy for the last two cases, which shows that the battery is gaining more charge than it is using. The influx of current to the battery actually lowers the equivalent fuel consumption of the system due to it subtracting away from the energy consumed by the engine.

For the case of the FUDS drive cycle, once again the Efficiency Mode is capable of following the velocity profile:

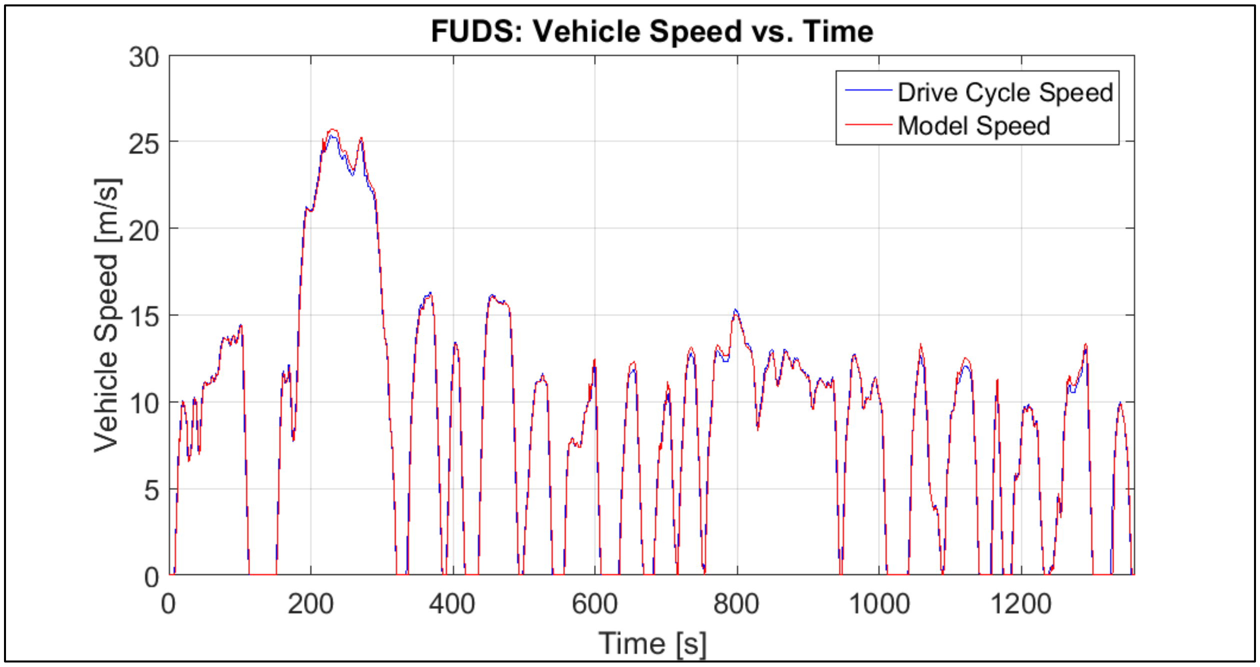


Figure 73: FUDS Drive Cycle Tracking for Efficiency Mode

Simulating the model with an initial charge of 85%, the following engine data is obtained:

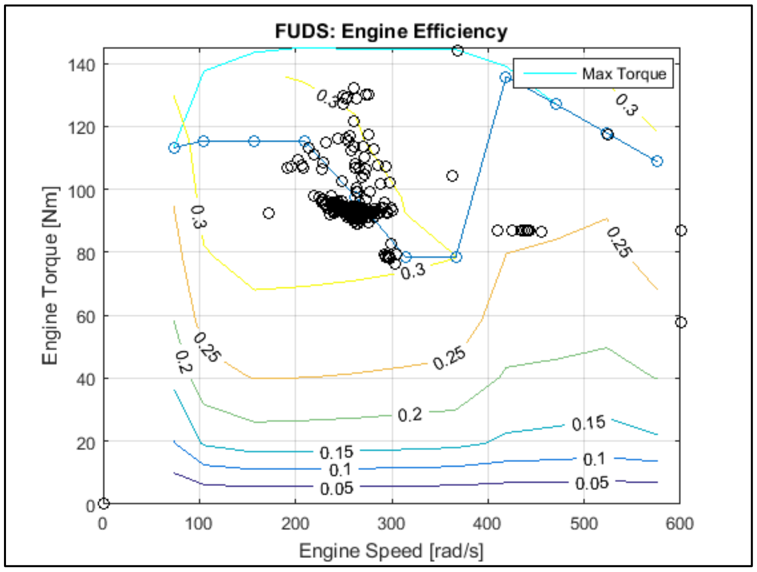
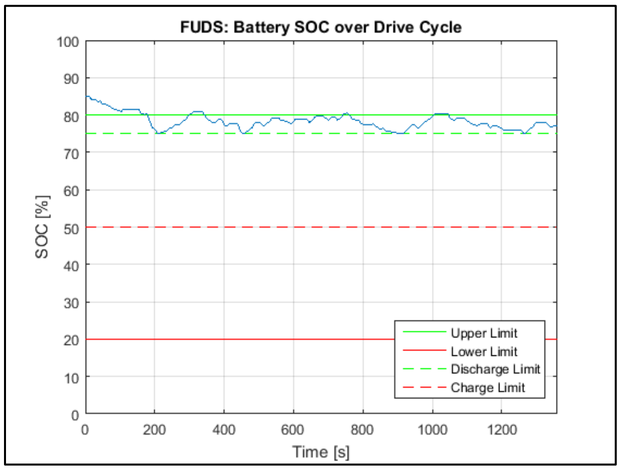


Figure 74: ICE Points for FUDS with Initial SOC 85%, Efficiency Mode

Most of the points follow the maximum efficiency curve as desired. A group of the points is above the line. This scattering, as seen in the previous cases, is a cause of quick changes in acceleration or deceleration. However, the majority of the points stay within the 30% efficiency contour area. The SOC and powertrain modes are displayed below:



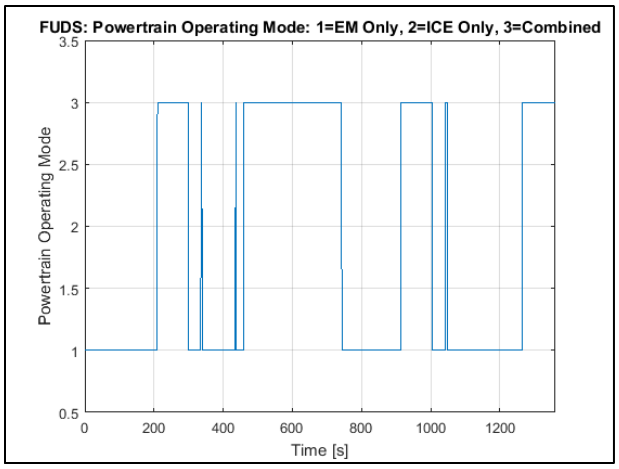


Figure 75: State of Charge Path for FUDS with Initial SOC 85%, Efficiency Mode

Figure 76: Powertrain Modes for FUDS with Initial SOC 85%, Efficiency Mode

Efficiency Mode does an excellent job at sustaining the SOC, as can be seen in Figure 75. Starting at 85%, the vehicle is in electric propulsion mode until 75%, where it enters the combined mode, from where it returns to the electric propulsion mode soon after due to the ability to charge up quickly in Efficiency Mode. Looking at Figure 76, it can be seen that the powertrain bounces back and forth from the electric mode to combined mode throughout the simulation run.

When simulating from an initial SOC of 50%, the engine efficiency data is as shown in Figure 77:

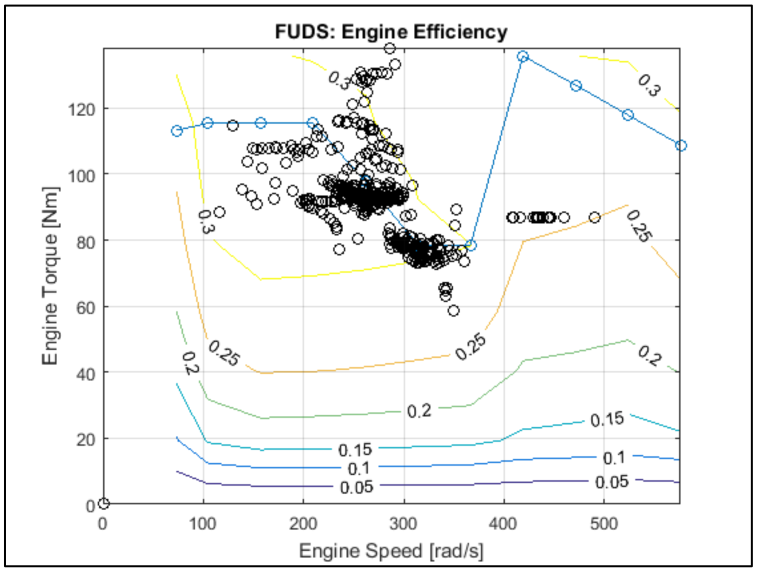


Figure 77: ICE Points for FUDS with Initial SOC 50%, Efficiency Mode

The engine data follows the efficiency line for the most part of the cycle, but there are more instances where the data does not follow the control law well, as seen at about 250 rad/s where there is a group of points at a higher torque than the efficiency line, or the group of points between 400 and 500 rad/s. Looking at the vector of data points, this happens each time the vehicle is in the normal operation mode, and the speed goes from 0 to a high speed, which happens many times in the FUDS drive cycle. The engine has to provide the torque needed to keep the car moving. This can be described with the throttle position over the duration of the cycle:

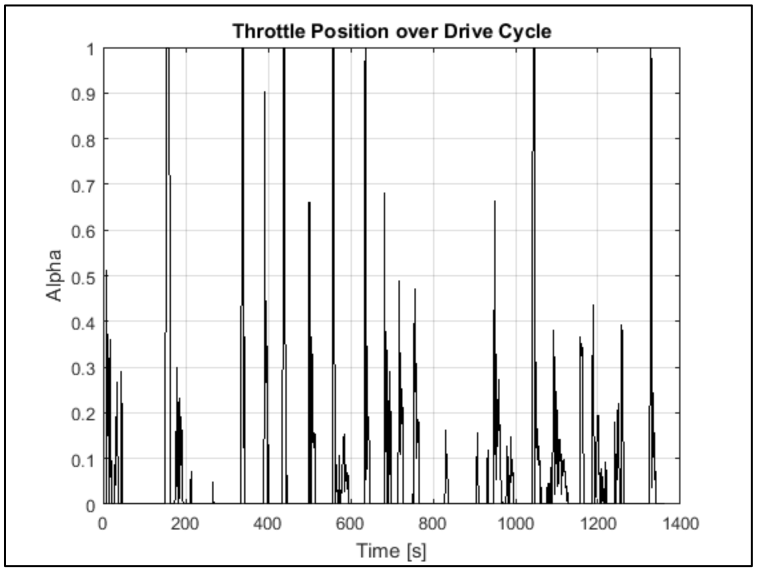


Figure 78: Throttle Request for FUDS with Initial SOC 50%, Efficiency Mode

At points of a quick start up from stopped position, the throttle request from the Driver Model is very high, and in some cases even maximum. This creates the engine points in the 400 rad/s range. The next step creates the large torque request from that engine speed, creating the engine points in the 250 rad/s range explained above. The state of charge and powertrain mode plots can be seen in Figure 79 and Figure 80.

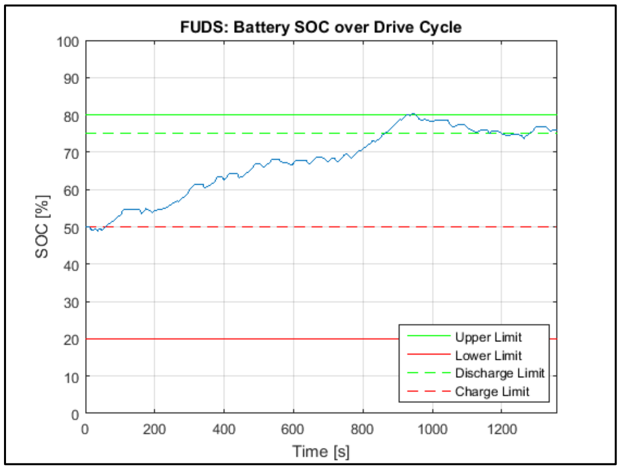


Figure 79: State of Charge Path for FUDS with Initial SOC 50%, Efficiency Mode

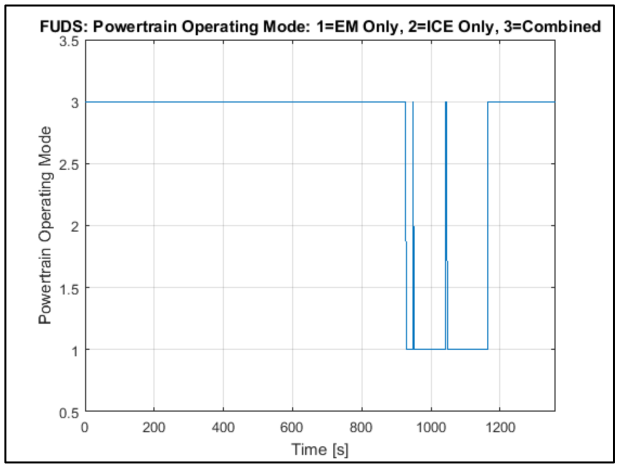


Figure 80: Powertrain Modes for FUDS with Initial SOC 50%, Efficiency Mode

These plots demonstrate the ability for the ESS to charge up in Efficiency Mode. The SOC starts at 50% as commanded, and then ends up charging to the upper limit, which switches the powertrain into electric mode. Eventually, the SOC discharges enough to re-enter the combined mode. Of all the cases tested, this is the sole simulation that charges the battery pack enough to enter electric propulsion mode.

To complete the simulation set, the model was executed with an initial SOC of 15%. This starts the powertrain off in charging mode, which will initially operate the engine on the maximum efficiency or maximum torque line if needed. The data points show the following trend:

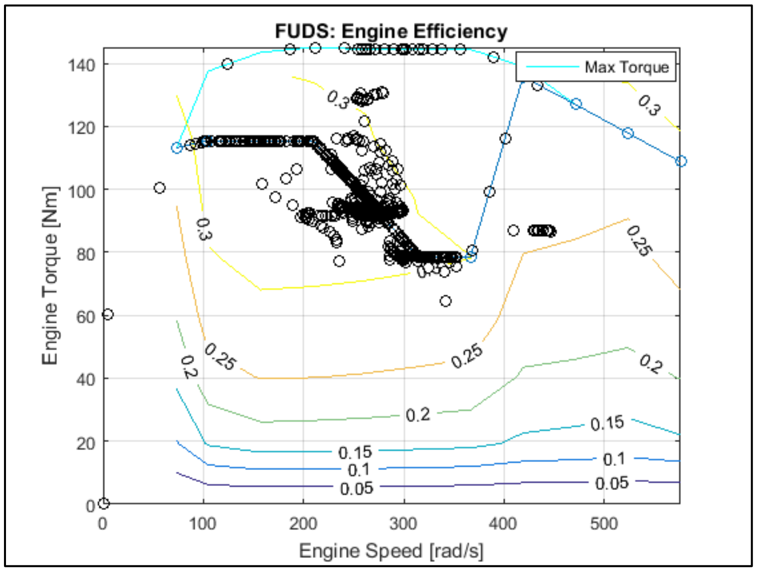


Figure 81: ICE Points for FUDS with Initial SOC 15%, Efficiency Mode

A large portion of the data points are either on the maximum efficiency line or on the maximum torque line, insinuating ICE propulsion only. The SOC and powertrain modes can be used to analyze this simulation run as well:

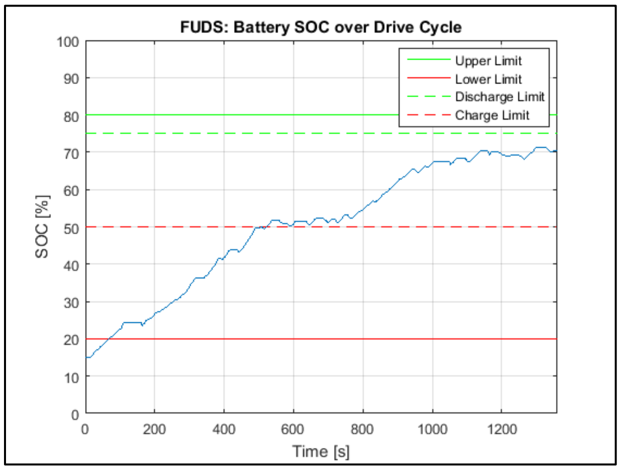


Figure 82: State of Charge Path for FUDS with Initial SOC 15%, Efficiency Mode

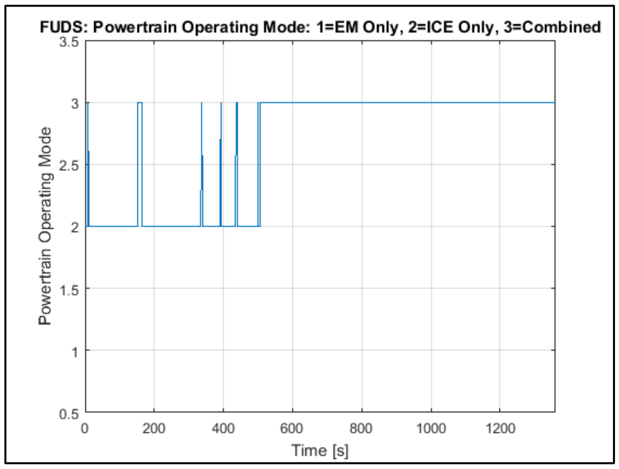


Figure 83: Powertrain Modes for FUDS with Initial SOC 15%, Efficiency Mode

The engine charges up the battery steadily, and the controller exits charging mode in about 500 seconds. From then on, the engine operates with the MPC control strategy for the remaining time. The SOC has an upward trend for the complete cycle, and ends up close to the upper limit. This once again shows how efficiency mode is useful in charging the energy storage system.

All three of these simulations can be compared to each other in terms of performance metrics. The table below shows the difference in vehicle performance parameters as a result of the initial SOC:

Table 12: Performance Comparison for FUDS with Different Initial SOC, Efficiency Mode

Initial SOC	Final SOC	MPG	MPGe
85%	77.2%	39.0	37.2
50%	76.1%	19.9	21.6
15%	70.5%	15.2	17.4

The first case yields a high fuel economy of 39.0 MPG and 37.2 MPGe, due to part of the process being in electric only mode, thus saving the engine fuel. The other two runs prove to not have the best fuel economy, which as described previously, is expected. The MPGe for those cases is however higher than the conventional MPG, which shows that this is a mode and drive cycle which benefits from the higher operational torque of the engine. This is backed up by the SOC readings in the same table. The first case, which starts in a charge depletion mode, only yields a net loss of 7.8% SOC. The other two simulation runs prove to have immense SOC increases.

5.2.3 Comparison over Repeated Cycles

In order to compare the performance of Fuel Use Mode and Efficiency Mode, the model was run on repeated cycles of each drive schedule. The result to look for was the ability for the operational mode to hold a state of charge. The US06 and FUDS drive cycles were each run for 3 continuous cycles with a starting initial SOC of 50% to study response over a longer distance. The drive cycle response and SOC results for US06 are shown below:

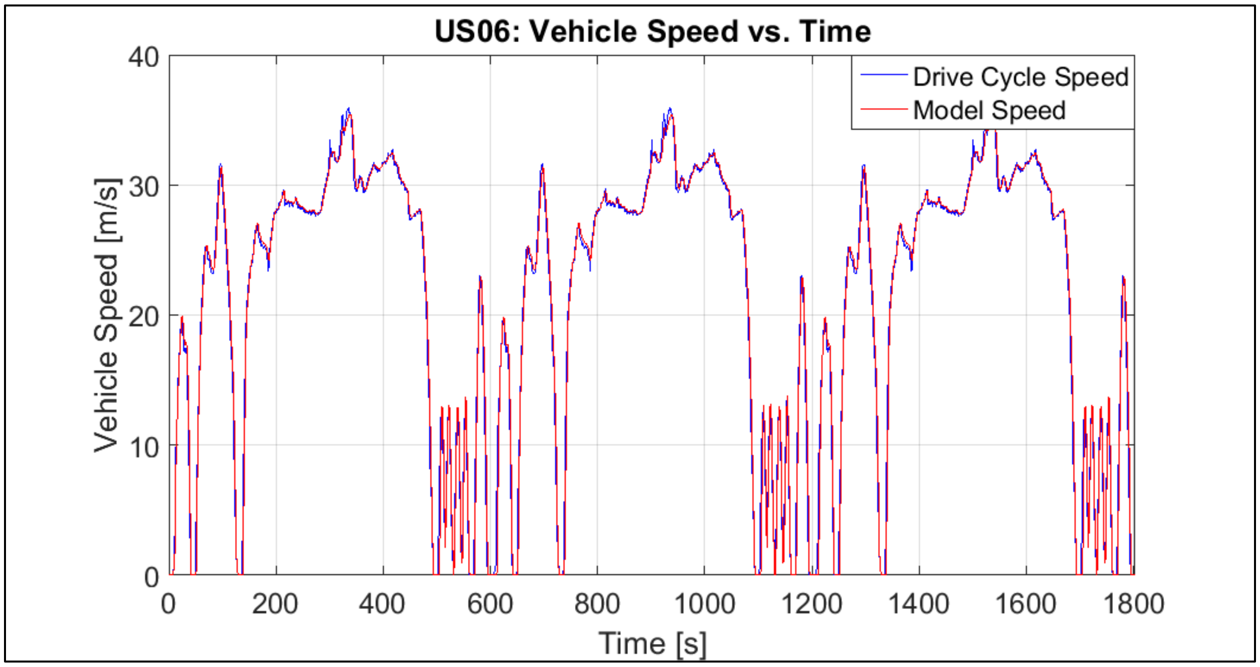


Figure 84: US06 Response for Three Continuous Cycles

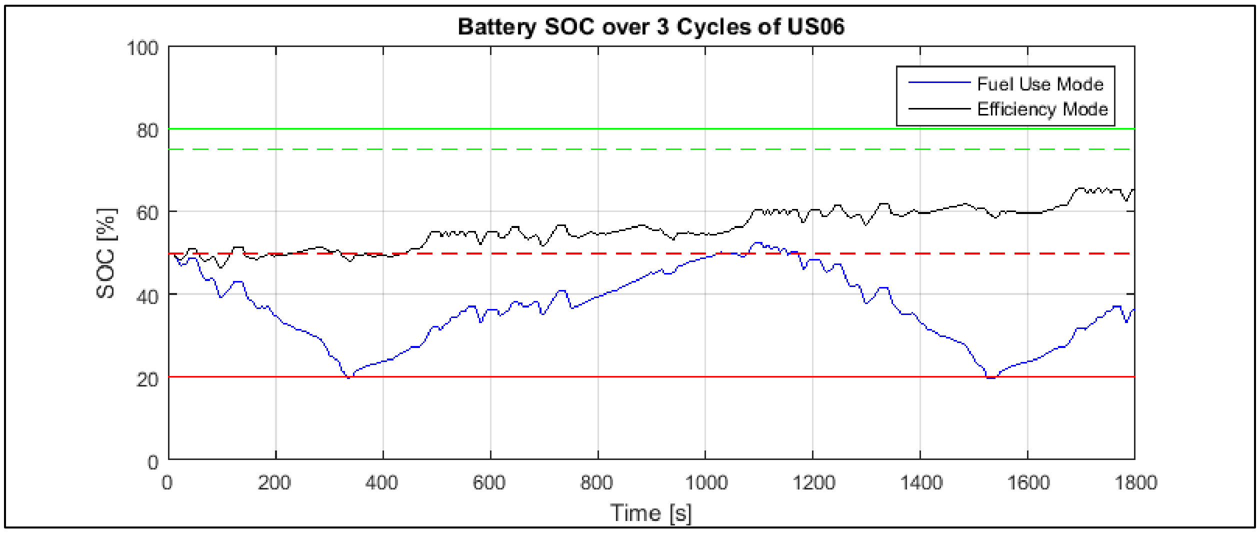


Figure 85: SOC Comparison for Fuel Use Mode and Efficiency Mode, US06

As the cycle repeats itself three times as in Figure 84, the SOC plot in Figure 85 shows that the battery depletes and recharges in Fuel Use Mode, while it maintains and even increases charge in Efficiency Mode. This is intuitive, since in Fuel Use Mode more power is commanded from the battery due to the limitation of fuel use of the engine. Similar results can be show for running three continuous cycles of FUDS:

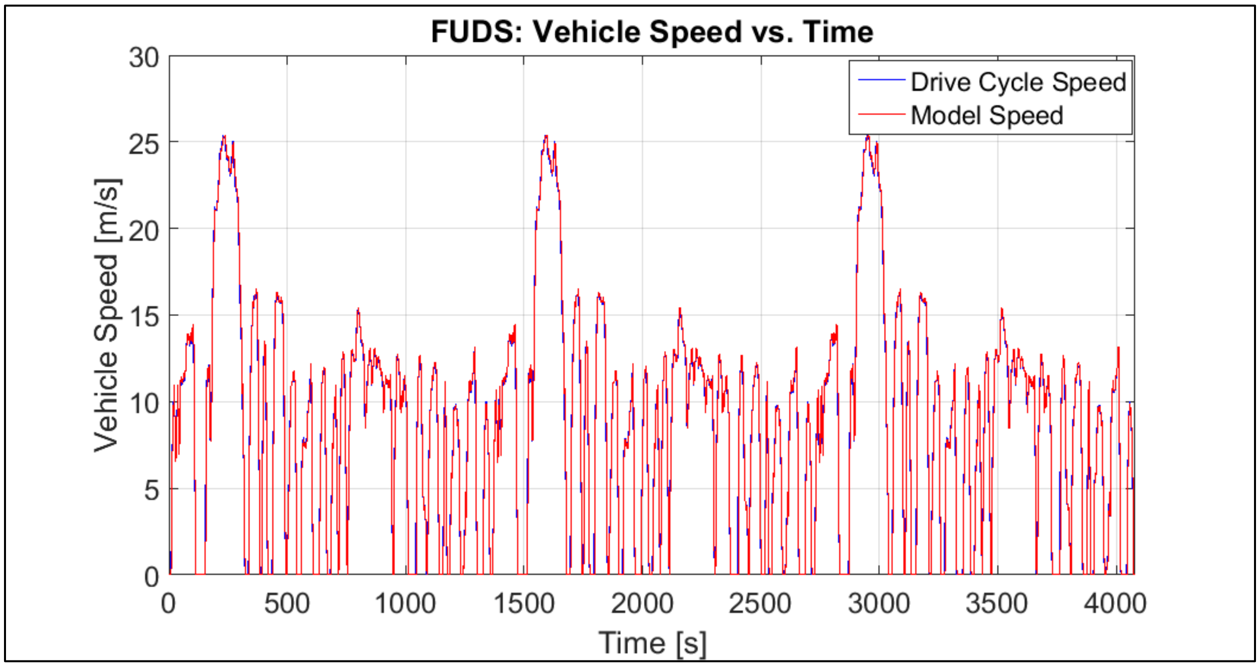


Figure 86: FUDS Response for Three Continuous Cycles

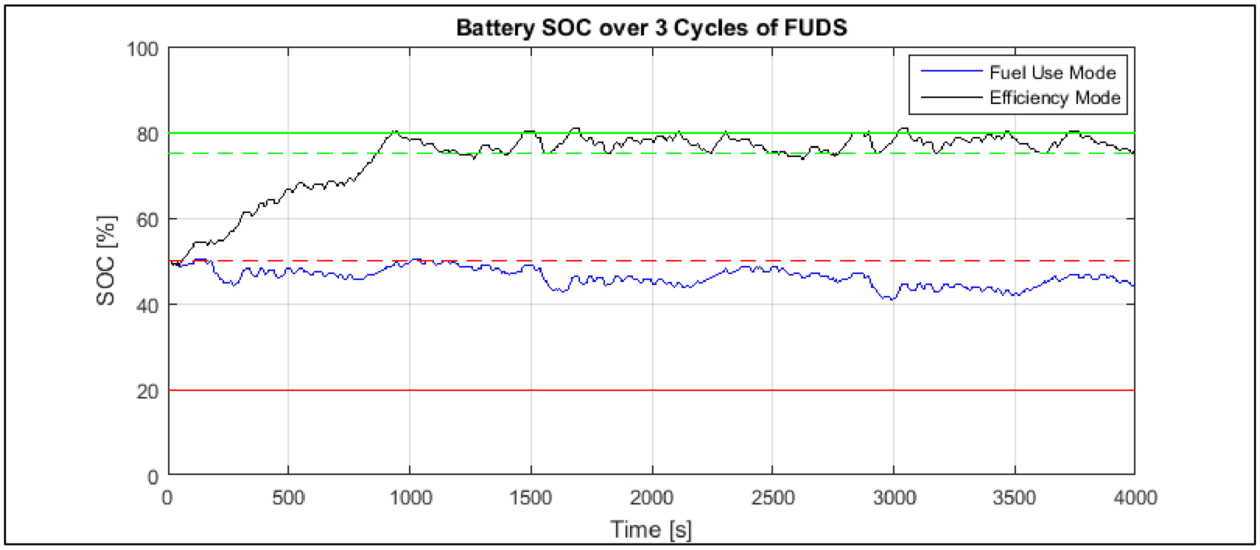


Figure 87: SOC Comparison for Fuel Use Mode and Efficiency Mode, FUDS

For both Fuel Use Mode and Efficiency Mode, it is evident from Figure 87 that the SOC is well sustained for repeated cycles of FUDS. In Fuel Use Mode, the battery drains less than 10% over the duration of the simulation. In Efficiency Mode, the battery charges to the upper limit during the first cycle, and then the powertrain switches between charge depleting and MPC operation for the remaining time.

All in all, the simulations performed show that the MPC algorithm used is capable of yielding a high fuel economy in Fuel Use Mode and limit the net battery consumption in Efficiency Mode. At a high initial state of charge, both operational modes and all cycles prove to be capable of yielding a high fuel economy. When the initial SOC is in the middle range, the highway cycle US06 with Fuel Use Mode provides the best fuel economy. This makes sense since on highways the engine can operate at a more consistent and steady path without the need of torque assist from the motor. When the state of charge starts low, the best mode to charge it is Efficiency Mode, and the FUDS drive cycle showed the greatest increase in charge. When running multiple cycles in succession, it can be seen that Efficiency Mode does a better job at sustaining and increasing the charge than Fuel Use Mode. These simulations show that MPC can be used to dictate the operation of the engine, and based on the velocity profile and battery charge, Fuel Use Mode and Efficiency Mode can be equally effective.

6 Implementation

This research done for this thesis focuses on the design and simulation of a hybrid powertrain. The next step in the process would be to implement this control algorithm on a vehicle. In order to do so, some changes would need to be made in order to accommodate for the use of Model Predictive Control in an actual vehicle. The problem is that MPC needs an idea of the vehicle cycle's characteristics in order to predict the output and apply the control input. In this research, this was done offline using Matlab scripts and functions. However, in order to use this algorithm on a vehicle, the MPC process must be done on board. One area to explore would be using a Global Positioning System (GPS) and path planning to determine a path for the vehicle and create a drive cycle based on this, and update the schedule whenever necessary, as detailed in [30]. MPC can then be used to determine for that cycle what engine torque to command.

An addition to the possibility for onboard implementation would be to refresh the prediction process every few steps of this newly defined drive cycle. This would consist of recalculating the drive cycle and predicting the output once again, and then refreshing the torque commands. This can be done at stop lights in an urban environment, or at steady state operating conditions in a highway environment. A process like this could incorporate changes in traffic into the control algorithm. The result of this would be a dynamic implementation of MPC.

7 Conclusion and Future Work

The goal of this thesis was to use Model Predictive Control to dictate an internal combustion engine’s torque in a series-parallel hybrid powertrain. MPC was chosen as the control strategy due to its ability to track a reference while predicting changes in the system response and handle uncertainties and disturbances in the system. In order to show the effects of the MPC algorithm on the complete vehicle system, two references were chosen: one to limit fuel consumption and one maximize engine efficiency. Limiting the fuel consumption was done by forcing the engine to operate on its 1 gram-per-second fuel consumption curve, which in this thesis was called Fuel Use Mode. To maximize engine efficiency, the engine was forced to operate at the torque corresponding to its maximum thermal efficiency, which was referred to as Efficiency Mode.

The mathematics of MPC was described in detail. This included the discretization of a continuous-time model, the construction of a prediction of system outputs over a specified time horizon, and the optimization of a cost function to determine the best control input to minimize the cost. The cost function used was a combination of the error in output and a limit on the control input. Further in the thesis, the practical implementation was described. The process paralleled the mathematical procedure; however, due to the quasi-static nature of the engine model used, a dynamic model was not available, and so the prediction had to be calculated in a different manner. This consisted of calculating combinations of fuel consumptions over the horizon, for a vector of torques ranging from zero to maximum engine torque at the current engine speed, then taking the torque request that resulted in the smallest cost.

A large part of the research process was the formulation of a working vehicle model. This consisted of implementing different subsystems for each of the major vehicle components, including a driver model representing the acceleration and brake pedal commands, a powertrain controller subsystem which sends torque commands to the engine and motor, the models for the engine and motor themselves, a subsystem representing the dynamics of a battery pack, a transmission model, and a vehicle dynamic model to translate the torques into vehicle motion and speed. Vehicle components were chosen from the ADVISOR simulator data, which were in line with specifications obtained in [27]. The controller was designed to implement the described MPC algorithm, and also incorporate responses to battery state-of-charge changes and hard acceleration. If the SOC was above a certain limit, the controller navigated the powertrain into electric propulsion only. If the SOC was below a certain limit, the engine was commanded to produce extra torque to charge the battery pack. In between the range, the MPC algorithm was implemented. In the event of hard acceleration, the MPC algorithm was implemented regardless of SOC, in order to provide the necessary torque from a combination of ICE and electric motor propulsion. Regenerative braking was also incorporated into the controller, in order to regain some lost energy in the braking process.

Once the model was completed, the MPC parameters had to be selected in order to provide the best performance of the system. The necessary parameters to choose were the receding horizon length, the cost function weights, and the PID gains for uncertainty correction in the engine model. This selection process was done by creating an error function and changing the parameters until the minimum error was found for each operational mode.

To conclude the thesis, simulations were run for both FUDS and US06 drive cycles. The simulations showed the effects of the starting SOC on the performance of the powertrain. When the initial charge was high, the vehicle obtained high fuel efficiency. In contrast, when the initial charge was low, the vehicle obtained low fuel efficiency but a better ability to charge the batteries. It was also determined that Fuel Use Mode is best for obtaining a high fuel economy, while Efficiency Mode is best or sustaining charge, or limiting the net loss of charge.

Future work was described in the implementation section; this work will include incorporating the controller onto an actual vehicle. The process will involve modifications, including coming up with a way to obtain the optimal torque requests on board in real time. An area to explore will be the use of GPS and path planning to determine a drive cycle, and update such drive cycle every few iterations in order to better adjust to real time traffic and road conditions.

Works Cited

[1] "Reduce Climate Change." [Online]. Available: https://www.fueleconomy.gov/feg/climate.shtml#collapseOne. [Accessed: 22-May-2016].

[2] M. Salman, N. J. Schouten, and N. A. Kheir, "Control Strategies for Parallel Hybrid Vehicles," in American Control Conference, 2000. Proceedings of the 2000, 2000, vol. 1, pp. 524–528 vol.1.

[3] Mehrdad Ehsani et al., Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. CRC Press, 2004.

[4] L. Guzzella and A. Sciarretta, *Vehicle Propulsion Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[5] J. Meisel, W. Shabbir, and S. A. Evangelou, "Evaluation of the Through-the-Road Architecture for Plug-In Hybrid Electric Vehicle Powertrains," in *Electric Vehicle Conference (IEVC), 2013 IEEE International*, 2013, pp. 1–5.

[6] R. Finesso, E. Spessa, and M. Venditti, "Optimization of the Layout and Control Strategy for Parallel Through-the-Road Hybrid Electric Vehicles," 2014.

[7] D. K. Mehr et al., "Optimization and Testing of a Through the Road Parallel, Hybrid-Electric, Crossover Sports Utility Vehicle," 2009.

[8] H. Moghbeli, A. H. Niasar, and N. Fallahi, "Fuzzy Energy Control Strategy of Through-to-Road Hybrid Electric Vehicle," in 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE), 2014, pp. 1660–1665.

[9] M. Young et al., "The Design and Development of a Through-the-Road Parallel Diesel Electric Hybrid," in 2007 IEEE Vehicle Power and Propulsion Conference, 2007, pp. 511–518.

[10] S. M. Saiful A Zulkifli, "Operation and Control of Split-Parallel, Through-theroad Hybrid Electric Vehicle with In-Wheel Motors," Int. J. Automot. Mech. Eng., vol. 11, no. 1, pp. 2793–2808, 2015.

[11] A. Panday and H. O. Bansal, "A Review of Optimal Energy Management Strategies for Hybrid Electric Vehicle," *Int. J. Veh. Technol.*, vol. 2014, pp. 1–19, 2014.

[12] G. Paganelli et al., "General Supervisory Control Policy for the Energy Optimization of Charge-Sustaining Hybrid Electric Vehicles," JSAE Rev., vol. 22, no. 4, pp. 511–518, Oct. 2001.

[13] V. Neelakantan, "Modeling, Design, Testing and Control of a Two-Stage Actuation Mechanism Using Piezoelectric Actuators for Automotive Applications," The Ohio State University, 2005.

[14] J. A. Rossiter, *Model-Based Predictive Control: A Practical Approach*. CRC Press, 2003.

[15] B. Saerens et al., "Model Predictive Control of Automotive Powertrains - First Experimental Results," in 47th IEEE Conference on Decision and Control, 2008. CDC 2008, 2008, pp. 5692–5697.

[16] D. Hrovat et al., "The Development of Model Predictive Control in Automotive Industry: A Survey," in 2012 IEEE International Conference on Control Applications (CCA), 2012, pp. 295–302.

[17] H. Chen et al., "Applying Model Predictive Control in Automotive," in 2012 10th World Congress on Intelligent Control and Automation (WCICA), 2012, pp. 6–7.

[18] A. Santucci, A. Sorniotti, and C. Lekakou, "Model Predictive Control for the Power-Split between Supercapacitor and Battery for Automotive Applications," in *Electric Vehicle Conference (IEVC), 2013 IEEE International*, 2013, pp. 1–7.

[19] A. Chaudhari, A. Plianos, and R. Stobart, “Development of Model Predictive Controller for SOFC-IC Engine Hybrid System,” *SAE Int. J. Engines,* vol. 2, no. 1, pp. 56–66, Apr. 2009.
[20] R. Beck et al., “Model Predictive Control of a Parallel Hybrid Vehicle Drivetrain,” in 44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC ’05, 2005, pp. 2670–2675.
[21] L. He et al., “A Model-Predictive-Control-Based Torque Demand Control Approach for Parallel Hybrid Powertrains,” IEEE Trans. Veh. Technol., vol. 62, no. 3, pp. 1041–1052, Mar. 2013.
[22] G. Ripaccioli et al., “A Stochastic Model Predictive Control Approach for Series Hybrid Electric Vehicle Power Management,” in Proceedings of the 2010 American Control Conference, 2010, pp. 5844–5849.
[23] F. Yan, J. Wang, and K. Huang, “Hybrid Electric Vehicle Model Predictive Control Torque-Split Strategy Incorporating Engine Transient Characteristics,” *IEEE Trans. Veh. Technol.*, vol. 61, no. 6, pp. 2458–2467, Jul. 2012.
[24] C. H. Mayr, A. Fleck, and S. Jakubek, “Hybrid Powertrain Control using Optimization and Cycle Based Predictive Control Algorithms,” in 2011 9th IEEE International Conference on Control and Automation (ICCA), 2011, pp. 937–944.
[25] K. Bovee et al., “Plant Modeling and Software Verification for a Plug-in Hybrid Electric Vehicle in the EcoCAR 2 Competition,” 2015.
[26] B. C. Glenn, “Intelligent Control of Parallel Hybrid Electric Vehicles,” The Ohio State University, 1999.
[27] J. C. King, “Model-Based Design of a Plug-In Hybrid Electric Vehicle Control Strategy,” 27-Sep-2012. [Online]. Available: https://theses.lib.vt.edu/theses/available/etd-09072012-124321/. [Accessed: 22-May-2016].
[28] A. Sciarretta et al., “A Control Benchmark on the Energy Management of a Plug-In Hybrid Electric Vehicle,” Control Eng. Pract., vol. 29, pp. 287–298, Aug. 2014.
[29] T. Markel et al., “ADVISOR: A Systems Analysis Tool for Advanced Vehicle Modeling,” J. Power Sources, vol. 110, no. 2, pp. 255–266, Aug. 2002.
[30] A. Rajagopalan et al., “Development of Fuzzy Logic and Neural Network Control and Advanced Emissions Modeling for Parallel Hybrid Vehicles The National Renewable Energy Laboratory (NREL) Golden, CO,” ResearchGate, Oct. 2002.

Appendix: Matlab Script Files

Controller Mode Determination (for Simulink)

```
function [choice,was_charging,was_depleting] = fcn(alpha,SOC,wc,wd)
up_lim=80;          %upper limit of battery SOC
lo_lim=20;          %lower limit of battery SOC
deplete_lim=75;     %depletion draining sufficient level
charge_lim=50;      %charging sufficient level
hard_accel=0.6;     %hard acceleration for throttle

was_charging=wc;
was_depleting=wd;

if (was_depleting==1 && SOC>deplete_lim && alpha<hard_accel)
    %stay in electric only until battery drains sufficient amount
    choice=1;
    was_depleting=1;
elseif (was_charging==1 && SOC<charge_lim && alpha<hard_accel)
    %stay in ICE only until battery charges sufficient amount
    choice=2;
    was_charging=1;
elseif (was_charging==1 && SOC>=charge_lim)
    %leave ICE only mode
    was_charging=0;
    choice=3;
elseif (was_depleting==1 && SOC<=deplete_lim)
    %leave electric only mode
    was_depleting=0;
    choice=3;
elseif (SOC>=up_lim && alpha<hard_accel)
    %electric only
    choice=1;
    was_depleting=1;
elseif (SOC<lo_lim && alpha<hard_accel)
    %ICE only, charge batteries
    choice=2;
    was_charging=1;
else
    %regular algorithm
    choice=3;
end

end
```

Transmission Gear Selection (for Simulink)

```
function gear_now = fcn(eng_spd,rad_max,rad_min,prev_gear)
% input is the engine speed, this will determine gear

if eng_spd > rad_max
    %shift up
        if prev_gear==5  %can't shift up if in last gear
           gear_now=prev_gear;
        else
           gear_now=prev_gear+1;
        end
elseif eng_spd < rad_min
    %shift down
        if prev_gear==1    %can't shift down if in first gear
           gear_now=prev_gear;
        else
           gear_now=prev_gear-1;
        end
    else
        gear_now=prev_gear;
end

end
```

MPC Torque Calculation (Offline)

```
%Reference Creation from Drive Cycles for MPC controller use
cycle='US06';   % choose from: US06, FUDS, AEU, NEDC

load (['Drive_Cycles/' cycle '.mat'])
load ('ICE_data.mat')

%% MPC Parameters for Both Modes
N=2;    %receding horizon
lambda1=0.0001;     %control weight in cost, mode 1 (fuel use)
lambda0=0.0000001;    %control weight in cost, mode 0 (efficiency)

%% Initialization
%Vehicle characteristics:
%constants needed
M   =2050;   %test mass [kg]
Mi  =1.04*M; %inertial mass (including rotating bodies) [kg]
CdA =0.76;   %drag coefficient * frontal area [m^2]
crr =0.01;   %coefficient of rolling resistance
rw  =0.324;  %tire radius [m]
g   =9.81;   %gravity [m/s^2]
p   =1.2;    %density of air [kg/m^3]

%gear ratio callouts
shifts=[0 10 25 35 45]; %speed at which to shift
gears=[1 2 3 4 5];      %gear corresponding to shift speed
gear_ratios=[3.25 1.81 1.21 0.86 0.64]; %5 spd transmission
final_drive=4.06;       %differential gear ratio
em_gr=9;    %instead of the final drive for EM

%drive cycle specs
v_ms=v_cyc*0.277778;
d_c=[t_cyc v_ms];   %drive cycle array for from workspace block
axle_spd=(v_ms)/rw;
trans_out_spd=axle_spd*final_drive;

%find engine speed and transmission gear for each point
[eng_spd_rad,position]=trans_gear_calc(t_cyc,gears,gear_ratios,trans_out_spd)
;
eng_spd_rpm=eng_spd_rad*60/(2*pi);      %for me to analyze

%find max engine torque for given speed
T_eng_max=interp1(fc_map_spd,fc_max_trq,eng_spd_rad);   %use engine data
ind=find(isnan(T_eng_max));     %find the entries that have NaN in eng_Trq
T_eng_max(ind)=0;               %change the NaN to zero

%create engine torque vector from 0 to max
T_eng_vec=zeros(size(T_eng_max,1),11);
for i=1:length(T_eng_max)
    T_eng_vec(i,1:11)=[0:.1:1].*T_eng_max(i);
end

%% Mode 1: Fuel Use Mode
```

```
disp('Fuel Use Mode')
fuel_cons=zeros(size(T_eng_vec));
for j=1:size(T_eng_vec,2)       %calculate fuel consumption of each operating
point

fuel_cons(:,j)=interp2(fc_map_spd,fc_map_trq,fc_fuel_map',eng_spd_rad,T_eng_v
ec(:,j),'spline');
end
ind=find(isnan(fuel_cons));     %find the entries that have NaN in fuel_cons
fuel_cons(ind)=0;               %change the NaN to zero

%MPC Optimization Step
min_cost=zeros(size(fuel_cons,1),1);
Topt_m1=zeros(size(fuel_cons,1),1);
for i=1:size(fuel_cons,1)
    if (size(fuel_cons,1)-i)>=N
        Tsum=matrixsum(T_eng_vec(i:i+N-1,:));       %sum of torques, for cost
function
        fuel_sum=matrixsum(fuel_cons(i:i+N-1,:));
        cost=(fuel_sum-N*1).^2 + lambda1*(Tsum.^2);    %cost function, 1g/s
times 5(horizon length)
        [min_val,min_ind]=min(cost);    %minimize cost function
        min_cost(i)=fuel_sum(min_ind);
        Topt_m1(i)=T_eng_vec(i,ceil(min_ind/(size(T_eng_vec,2)^(N-1))));
    else
        Tsum=matrixsum(T_eng_vec(i:size(fuel_cons,1),:));       %sum of
torques, for cost function
        fuel_sum=matrixsum(fuel_cons(i:size(fuel_cons,1),:));
        cost=(fuel_sum-(size(fuel_cons,1)-i)*1).^2 + lambda1*(Tsum.^2);
%cost function, 1g/s times 5(horizon length)
        [min_val,min_ind]=min(cost);    %minimize cost function
        min_cost(i)=fuel_sum(min_ind);
        Topt_m1(i)=T_eng_vec(i,ceil(min_ind/(size(T_eng_vec,2)^(N-1))));
    end
    i
end

%% Mode 0: Efficiency Mode
disp('Efficiency Mode')

%Calculate max efficiencies
fc_eff_max=max(fc_eff,[],2);   %maximum efficiency at each map speed
max_eff=interp1(fc_map_spd,fc_eff_max,eng_spd_rad);
ind=find(isnan(max_eff));      %find the entries that have NaN in max_eff
max_eff(ind)=0;                %change the NaN to zero

max_eff_sum=zeros(size(t_cyc));
for j=1:length(t_cyc)           %create sum at each point of max effiencies
for cost
    if (j+N)<=(length(t_cyc)+1)
        max_eff_sum(j) = sum(max_eff(j:j+N-1));
    else
        max_eff_sum(j) = sum(max_eff(j:end));
    end
end
```

```
ice_eff=zeros(size(T_eng_vec));
for j=1:size(T_eng_vec,2)       %calculate ICE efficiency of each operating
point

ice_eff(:,j)=interp2(fc_map_spd,fc_map_trq,fc_eff',eng_spd_rad,T_eng_vec(:,j)
,'spline');
end
ind=find(isnan(ice_eff));     %find the entries that have NaN in ice_eff
ice_eff(ind)=0;               %change the NaN to zero

%MPC Optimization Step
min_cost=zeros(size(ice_eff,1),1);
Topt_m0=zeros(size(ice_eff,1),1);
for i=1:size(ice_eff,1)
    if (size(ice_eff,1)-i)>=N
        Tsum=matrixsum(T_eng_vec(i:i+N-1,:));       %sum of torques, for cost
function
        eff_sum=matrixsum(ice_eff(i:i+N-1,:));
        cost=(eff_sum-max_eff_sum(i)).^2 + lambda0*(Tsum.^2);    %cost
function, sum of the efficiencies-sum of max efficiencies
        [min_val,min_ind]=min(cost);    %minimize cost function
        min_cost(i)=eff_sum(min_ind);
        Topt_m0(i)=T_eng_vec(i,ceil(min_ind/(size(T_eng_vec,2)^(N-1))));
    else
        Tsum=matrixsum(T_eng_vec(i:size(ice_eff,1),:));       %sum of
torques, for cost function
        eff_sum=matrixsum(ice_eff(i:size(ice_eff,1),:));
        cost=(eff_sum-max_eff_sum(i)).^2 + lambda0*(Tsum.^2);    %cost
function, sum of the efficiencies-sum of max efficiencies
        [min_val,min_ind]=min(cost);    %minimize cost function
        min_cost(i)=eff_sum(min_ind);
        Topt_m0(i)=T_eng_vec(i,ceil(min_ind/(size(T_eng_vec,2)^(N-1))));
    end
    i
end

hzn=num2str(N);
save(['Toptimals/Topt_' cycle '_N' hzn '.mat'], 'Topt_m1','Topt_m0')
```

Transmission Gear Calculation (for MPC)

```
function
[eng_spd,position]=trans_gear_calc(t_cyc,gears,gear_ratios,trans_out_spd)
rad_max=350;    %max engine limit
rad_min=125;    %min engine limit

position=zeros(size(t_cyc));    %initialize vectors
position(1)=gears(1);
eng_spd=zeros(size(t_cyc));

for i=1:length(t_cyc)
    eng_spd(i)=trans_out_spd(i)*gear_ratios(position(i));
    %calculate engine speed in loop
    if eng_spd(i) > rad_max
        if position(i)==gears(end)  %can't shift up if in last gear
           position(i+1)=position(i);
        else
           position(i+1)=gears(position(i)+1);
        end
    elseif eng_spd(i) < rad_min
        if position(i)==gears(1)    %can't shift down if in first gear
           position(i+1)=position(i);
        else
           position(i+1)=gears(position(i)-1);
        end
    else
        position(i+1)=position(i);
    end
end

end
```

Matrix Summation (for MPC)

```
function sums=matrixsum(y)
%matrix summation: sum all different possible combinations of indices

[m,n]=size(y);  %define dimensions to reference
p=n^m;

A=zeros(m,p);   %create the different rows to be summed
for i=1:m
    for j=1:n
        A(i,1+(j-1)*((p)/(n^i)):j*((p)/(n^i)))=ones(1,(p)/(n^i))*y(i,j);
    end
    B=A(i,1:j*((p)/(n^i)));
    if i~=1
        D=B;
        for k=1:(n^(i-1)-1)
                C=cat(2,B,D);
                B=C;
        end
        A(i,:)=C;
    end
end
sums=sum(A,1);  %sum up the rows to get the output
end
```

Model Initialization

```
%Initialization script for simulation model

clear all;
close all;
clc;

%load engine variables
FC_SI63_emis;   %run advisor file to get data
save('ICE_data.mat','fc_map_spd','fc_map_trq','fc_fuel_map','T','w', ...
    'fc_fuel_lhv','fc_max_trq','fc_fuel_den')
clear all;

%load electric motor variables
MC_PM49;        %run advisor file to get data
save('EM_data.mat','mc_map_spd','mc_max_trq','mc_map_trq','mc_eff_map')
clear all;

%load energy storage system (ESS) variables (battery)
ESS_LI7_temp;   %fun advisor file to get data
save('ESS_data.mat','ess_soc','ess_tmp','ess_max_ah_cap','ess_r_dis',...
    'ess_r_chg','ess_voc','ess_module_num','ess_module_mass');
clear all;
load 'ESS_data.mat';

% Vehicle constants needed:
M   =2050; %test mass [kg], including batteries
Mi  =1.04*M; %inertial mass (including rotating bodies) [kg]
CdA =0.76;   %drag coefficient * frontal area [m^2]
crr =0.01;   %coefficient of rolling resistance
rw  =0.324;  %tire radius [m]
g   =9.81;   %gravity [m/s^2]
p   =1.2;    %density of air [kg/m^3]

% Transmission characteristics:
gears=[1 2 3 4 5];
gear_ratios=[3.25 1.81 1.21 0.86 0.64];
final_drive=4.06;
em_gr=9;    %instead of the final drive for EM

% Electric Motor Constants
regen=0.6;  %percentage of braking that can be done by regenerative

% Battery Constants
Temp_batt=25;   %constant temp, degrees C
N_par=2;        %number of packs in parallel
Capacity_b=interp1(ess_tmp,ess_max_ah_cap,Temp_batt);   %battery capacity
%SOC_init=70;    %inital state of charge [%]
M_batt=ess_module_num*ess_module_mass*N_par; %mass of battery pack

% MPC Conditions
N=2;    %set the receding horizon used to 2 seconds
```

```
% Conversions:
m2mi=0.000621;  %meters to miles conversion
L2G=0.264;      %Liter to gallon conversion

% Load Engine Data
load ICE_data.mat
%efficiency:
    fc_eff=(T.*w)./(fc_fuel_map*fc_fuel_lhv);
    save('ICE_data.mat','fc_map_spd','fc_map_trq','fc_fuel_map','T','w', ...
    'fc_fuel_lhv','fc_max_trq','fc_fuel_den','fc_eff')

%fuel use mode 1:
    %find 1 g/s fuel consumption line, plot the contour
    figure();
    [C h]=contour(fc_map_spd,fc_map_trq,fc_fuel_map',[1 1]); %plots only the
1g/s fuel use line
    clabel(C,h);
    grid on
    xlabel('Engine Speed [rad/s]')
    ylabel('Engine Torque [Nm]')
    title('Engine 1 g/s Line')
    one_gs_spd=C(1,2:end);  %obtain contour data for 1g/s line
    one_gs_trq=C(2,2:end);
%efficiency mode 0:
    %get the torque with most efficiency at the current speed
    [fc_max_eff fc_max_eff_ind]=max(fc_eff');
    for i=1:length(fc_max_eff)
        trq_max_eff(1,i)=fc_map_trq(fc_max_eff_ind(i));
    end
    trq_max_eff=min(trq_max_eff,fc_max_trq);    %don't let the efficiency
torque exceed max torque at that speed

    %create efficiency contour
    figure()
    [C h]=contour(fc_map_spd,fc_map_trq,fc_eff');
    clabel(C,h);
    grid on
    hold on
    xlabel('Engine Speed [rad/s]')
    ylabel('Engine Torque [Nm]')
    title('Engine Efficiency Contour')
    plot(fc_map_spd,trq_max_eff,'-o')

%note: fc_fuel_den = density, grams/Liter
fuel_den=fc_fuel_den;

%load Electic Motor Data
load EM_data.mat;

t_sample=1; %sample time
```

Post-Simulation Plots

```
% Post-Simulation Script for Forward Model
% Create the plots from the simulation run
close all;

% Vehicle Speed and Model Speed:
figure('position', [50, 50, 1050, 550])
plot(model_time,model_v_dc,'b',model_time,model_v_car,'r')
grid on
xlabel('Time [s]')
ylabel('Vehicle Speed [m/s]')
title([selection ': Vehicle Speed vs. Time'])
legend('Drive Cycle Speed','Model Speed')
axis([0 model_time(end) 0 40])
set(gca,'fontsize',18)

% MPG:
figure();
plot(model_time,model_MPG)
grid on
xlabel('Time [s]')
ylabel('Fuel Economy [MPG]')
title([selection ': Fuel Economy vs. Time'])

% Fuel Consumption Plot:
figure();
plot(fc_map_spd,trq_max_eff,'m')    %max efficiency line
hold on
plot(fc_map_spd,fc_max_trq,'c')     %max torque line

[C h]=contour(fc_map_spd,fc_map_trq,fc_fuel_map');
clabel(C,h);
grid on
xlabel('Engine Speed [rad/s]')
ylabel('Engine Torque [Nm]')
title([selection ': Engine Fuel Consumption'])
hold on
plot(model_W_eng,model_T_eng,'ko')  %model data

% Efficiency Plot
figure();
plot(fc_map_spd,fc_max_trq,'c')
hold on
[C h]=contour(fc_map_spd,fc_map_trq,fc_eff');
clabel(C,h);
grid on
hold on
xlabel('Engine Speed [rad/s]')
ylabel('Engine Torque [Nm]')
title([selection ': Engine Efficiency'])
plot(fc_map_spd,trq_max_eff,'-o')
hold on
plot(model_W_eng,model_T_eng,'ko')  %model data
```

```
% Powertrain Operating Mode
figure();
plot(model_time,model_pwt_mode);
xlabel('Time [s]')
ylabel('Powertrain Operating Mode')
title([selection ': Powertrain Operating Mode: 1=EM Only, 2=ICE Only,
3=Combined'])
grid on
axis([0 model_time(end) 0.5 3.5])

% Battery SOC
figure();
plot(model_time,80*ones(size(model_time)),'g',model_time,20*ones(size(model_t
ime)),'r')
hold on;
plot(model_time,75*ones(size(model_time)),'g--
',model_time,50*ones(size(model_time)),'r--')
legend('Upper Limit','Lower Limit','Discharge Limit','Charge Limit')
plot(model_time,model_SOC)
xlabel('Time [s]')
ylabel('SOC [%]')
title([selection ': Battery SOC over Drive Cycle'])
grid on
axis([0 model_time(end) 0 100])
```