Title
PHEV Power Management Optimization Using Trajectory Forecasting and Fuzzy Logic

Permalink
https://escholarship.org/uc/item/83p636qq

Author
Garcia, Joseph Augusto

Publication Date
2019

Peer reviewed|Thesis/dissertation
UNIVERSITY OF CALIFORNIA,
IRVINE

PHEV Power Management Optimization Using Trajectory Forecasting and Fuzzy Logic

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Mechanical and Aerospace Engineering

by

Joseph Augusto Garcia

Thesis Committee:
Professor Gregory Washington, Chair
Professor Faryar Jabbari
Professor Jack Brouwer

2019
TABLE OF CONTENTS

List of Figures ........................................................................................................ iv

List of Tables ........................................................................................................... vii

Acknowledgements .................................................................................................. viii

Abstract of Thesis ................................................................................................. ix

1. Introduction ........................................................................................................... 1

2. PHEV Technology ............................................................................................... 3

   2.1. PHEV Vehicle Design .................................................................................. 3

   2.2. PHEV Energy Management Control Strategy Research ............................... 6

3. Control Strategy .................................................................................................. 10

   3.1. Part 1: Trajectory Forecasting Setup ............................................................. 10

   3.2. Part 2: Real-Time Trajectory Forecasting Control Strategy ......................... 21

   3.3. Rule-based Classification .............................................................................. 24

   3.4. Fuzzy Logic Control .................................................................................... 25

      3.4.1. Fuzzification .......................................................................................... 29

      3.4.2. Rule-Bases ............................................................................................ 30

      3.4.3. Inference Mechanism ............................................................................ 31

      3.4.4. Defuzzification ....................................................................................... 33

4. Simulation ........................................................................................................... 34

   4.1. Route Data .................................................................................................... 34
4.2. Simulation Parameters ................................................................. 36
4.3. Results .......................................................................................... 39
5. Summary and Conclusions ............................................................... 44
6. References ....................................................................................... 47
7. Appendix ......................................................................................... 55
7.1. ADVISOR Results Window Figures ............................................. 55
   7.1.1. Route 1 .................................................................................... 55
   7.1.2. Route 2 .................................................................................... 58
   7.1.3. Route 3 .................................................................................... 61
   7.1.4. Route 4 .................................................................................... 64
   7.1.5. Route 5 .................................................................................... 67
   7.1.6. Route 6 .................................................................................... 70
   7.1.7. Route 7 .................................................................................... 73
   7.1.8. Route 8 .................................................................................... 76
7.2. ADVISOR Code Additives ............................................................. 78
   7.2.1. Trajectory Forecasting Setup Code ......................................... 78
   7.2.2. Parallel TF Vehicle Control Block: Engine Shutoff for EV Only Routes Function Code 92
   7.2.3. Fuzzy TF Control Block: SOC Min Assignment Function Code ......................... 93
   7.2.4. Fuzzy TF Control Block: Control Strategy Function Code ............................... 93
List of Figures

Figure 1: Hybrid Electric Vehicle Design [10] .................................................................................... 4
Figure 2: Plug-in Hybrid Electric Vehicle Design [11] ......................................................................... 4
Figure 3: Typical Google Maps route [37] .................................................................................................. 12
Figure 4: Google Maps route [37]; modes assigned according to only traffic conditions ............. 12
Figure 5: Google Maps route [37]; modes assigned according to traffic and speed limits .......... 14
Figure 6: PHEV configuration snapshot from ADVISOR software [38] ................................................. 15
Figure 7: Plot for Amps per mph ........................................................................................................... 18
Figure 8: Plot for Amp-hours per mph .................................................................................................... 19
Figure 9: ADVISOR PHEV Simulink model; control strategy block (in red) [38] .............................. 22
Figure 10: ADVISOR control strategy block interior [38]; TF control strategy function block additive (in red); SOC min assignment function block additive (in blue) ............................................. 22
Figure 11: ADVISOR Route and PHEV battery parameters [38] ............................................................. 23
Figure 12: Fuzzy Logic Controller [43] ................................................................................................ 26
Figure 13: Membership functions for error and change in error ............................................................. 28
Figure 14: Membership functions for output K value ........................................................................... 29
Figure 15: Fuzzification example scenario ........................................................................................... 30
Figure 16: Google Maps route 1 [37] ..................................................................................................... 35
Figure 17: Route 1 drive cycle ................................................................................................................ 35
Figure 18: EPA created UDDS drive cycle [44] ..................................................................................... 35
Figure 19: Route 1 Drive Cycle ............................................................................................................... 39
Figure 20: Route 2 Drive Cycle ............................................................................................................... 39
Figure 21: Route 3 Drive Cycle ............................................................................................................... 40
Figure 22: Route 4 Drive Cycle ........................................................................................................ 40
Figure 23: Route 5 Drive Cycle ........................................................................................................ 41
Figure 24: Route 6 Drive Cycle ........................................................................................................ 41
Figure 25: Route 7 Drive Cycle ........................................................................................................ 42
Figure 26: Route 8 Drive Cycle ........................................................................................................ 42
Figure 27: Route 1 TF control strategy ADVISOR simulation results ............................................. 55
Figure 28: Route 1 CS control strategy ADVISOR simulation results ............................................. 56
Figure 29: Route 1 CD control strategy ADVISOR simulation results ............................................. 57
Figure 30: Route 2 TF control strategy ADVISOR simulation results ............................................. 58
Figure 31: Route 2 CS control strategy ADVISOR simulation results ............................................. 59
Figure 32: Route 2 CD control strategy ADVISOR simulation results ............................................. 60
Figure 33: Route 3 TF control strategy ADVISOR simulation results ............................................. 61
Figure 34: Route 3 CS control strategy ADVISOR simulation results ............................................. 62
Figure 35: Route 3 CD control strategy ADVISOR simulation results ............................................. 63
Figure 36: Route 4 TF control strategy ADVISOR simulation results ............................................. 64
Figure 37: Route 4 CS control strategy ADVISOR simulation results ............................................. 65
Figure 38: Route 4 CD control strategy ADVISOR simulation results ............................................. 66
Figure 39: Route 5 TF control strategy ADVISOR simulation results ............................................. 67
Figure 40: Route 5 CS control strategy ADVISOR simulation results ............................................. 68
Figure 41: Route 5 CD control strategy ADVISOR simulation results ............................................. 69
Figure 42: Route 6 TF control strategy ADVISOR simulation results ............................................. 70
Figure 43: Route 6 CS control strategy ADVISOR simulation results ............................................. 71
Figure 44: Route 6 CD control strategy ADVISOR simulation results ............................................. 72
Figure 45: Route 7 TF control strategy ADVISOR simulation results ........................................ 73
Figure 46: Route 7 CS control strategy ADVISOR simulation results ........................................ 74
Figure 47: Route 7 CD control strategy ADVISOR simulation results ........................................ 75
Figure 48: Route 8 TF control strategy ADVISOR simulation results ........................................ 76
Figure 49: Route 8 CS control strategy ADVISOR simulation results ........................................ 77
Figure 50: Route 8 CD control strategy ADVISOR simulation results ........................................ 78
List of Tables

Table 1: ADVISOR PHEV configuration component descriptions [38]........................................ 16
Table 2: Route 5 (20.1 miles) SOC estimation comparison .......................................................... 19
Table 3: Route 7 (17.1 miles) SOC estimation comparison .......................................................... 19
Table 4: Route 8 (6.5 miles) SOC estimation comparison .......................................................... 20
Table 5: Priority value table..................................................................................................... 25
Table 6: Approximated speed value table.................................................................................. 25
Table 7: Qualitative summary of input variable influence on output variable ......................... 28
Table 8: Rule Table.................................................................................................................. 31
Table 9: ADVISOR parallel hybrid control strategy parameters.............................................. 37
Table 10: ADVISOR parallel hybrid control strategy parameter values.................................. 38
Table 11: ADVISOR mpgge results for all route and control strategy combinations.............. 43
Acknowledgements

I would like to first thank Dr. Washington for all the help and guidance he has given me over the years. He has pushed and guided me throughout my academic career into becoming the student I am today. I owe a large portion of who I am as a researcher and engineer to him. I would also like to thank Professor Taha and Professor Torroja for their advice in the initial steps of my research. They assistance pushed me in the right direction. I thank Professor Jabbari and Professor Brouwer for serving on my thesis committee.

I would next like thank the many members of the ISSL lab. Joseph Bell and Vatche Donikian provided me with the resources to obtain the ADVISOR simulation software required for the basis of my research. Most notably, I would like to thank my lab mate and good friend Theron Smith. He helped me formulate a lot of my research’s main ideas, as well advise me in the development of my research’s control strategy.

Lastly, I would like to thank my family and many loved ones. Most important are my mother, Mirna Guardado, and sister, Karla Peña. They have sacrificed much throughout their lives to ensure my success and education. Next is Robin Jeffers, whose help in my undergraduate studies played a large role in leading me to my current position. For always being by my side, my remaining loved ones and friends: Karina Menendez, Juan Garcia, Abraham Martinez, Stephanie Martinez, Kevin Navarro, and Jasmin Garcia. Thank you to all of you for always supporting me and motivating me to continue forward.
Abstract of Thesis

PHEV Power Management Optimization Using Trajectory Forecasting and Fuzzy Logic

By

Joseph Augusto Garcia

Master of Science in Mechanical Engineering

University of California, Irvine, 2019

Professor Gregory Washington, Chair

In hopes of lessening the reliance on fossil fuels, Plug-in Hybrid Electric Vehicles (PHEVs) have become an attractive option as an alternative fuel vehicle due to their larger electric motors and energy storage systems (ESS). To improve their fuel efficiency, many studies have been done to investigate the use of a priori route information to optimize the use of a PHEV’s ICE and ESS. This study introduces a new control strategy that uses a priori knowledge of a PHEV’s pre-planned route to develop a battery charge usage plan that determines when the vehicle will use its different forms of propulsion. The PHEV can propel itself relying solely on its internal combustion engine (ICE), electric motor (EM), and or a hybrid of both. The strategy uses a route’s speed limits and states of traffic to estimate the consumption of charge and resulting decrease in SOC, and determine the optimal method of propulsion for the PHEV along its route. Fuzzy logic is then used to ensure that battery use during the times of hybrid propulsion is optimized. The control strategy is evaluated and compared to common PHEV control strategies such as Charge Sustaining (CS) and Charge Depletion (CD) using National Renewable Energy Laboratory’s vehicle simulator ADVISOR, with results showing possible increases fuel efficiency starting at about 1%-10% over long traffic heavy routes within this study.
1. Introduction

Daily transportation has become extremely dependent upon fossil fuels. Currently, the United States has less than 5% of the world’s population, but approximately one-fifth of the world’s automobiles [1]. Serving as the primary fuel source for this vast amount of transportation is petroleum. In 2017, the United States led the world in petroleum consumption at a rate of 19.88 million barrels per day and had a net petroleum import of 3.8 million barrels per day [2]. In 2018, the consumption of petroleum in the United States increased to a rate of 20.5 million barrels per day [3] while the net petroleum import decreased to 2.34 million barrels per day [4]. Even with a decreased net petroleum import, the current use of such fossil fuels forces the United States to heavily rely on them with the combustion of petroleum distillates from use as fuel in transportation leading to serious environmental issues from pollution.

When fossil fuels are burned, nitrogen oxides, hydrocarbons, particulate matter, and other pollutants are released into the atmosphere. These pollutants lead to harmful respiratory issues in people, the formation of smog, and function as a heat trapping greenhouse gas that worsen the effects of climate change [1]. In response to the problems from burning fossil fuels, many states have begun to implement legislation to promote the search of alternatives and solutions. California established its Renewables Portfolio Standard (RPS) in 2002 under Senate Bill 1078, requiring that electrical corporations increase their procurement of eligible renewable resources by 1% per year until 20% of its total retail sales are procured from renewable resources [5]. In 2015, the passing of Senate Bill 350 then required electrical retail sellers and publicly owned utilities in California to procure 50% of their electricity from renewable resources by 2030 [6]. California’s latest RPS goal, according to Senate Bill 100 passed in 2018, is now 60% by 2030 with all state’s electricity being required to come from carbon-free resources by 2045 [7].
To add to the list of practical solutions for lessening the effects of fossil fuels, alternative fuel vehicles have been identified as viable options. Currently, drivers can choose to drive vehicles that use alternative fuels such as biodiesel, electricity, ethanol, hydrogen, natural gas, or propane [8]. Increased interest in fuel efficiency over the past few decades has made them, specifically hybrid electric and plug-in hybrid electric vehicles (PHEVs), more accepted with increased attention leading to numerous technological improvements and cost reduction.
2. PHEV Technology

2.1. PHEV Vehicle Design

On the market today, two main types of hybrid electric vehicles (HEV) are sold; plug-ins and non-plug-ins. A non-plugin HEV, as shown in Figure 1, is propelled by a combination of both an internal combustion engine (ICE) and an electric motor (EM). The electricity that powers the EM can be generated by the vehicle’s own regenerative braking system and/or on-board generator. The use of regenerative braking is a process where the electric motor helps to slow the vehicle and converts the resulting kinetic energy into usable electricity [9]. Its on-board generator is powered by the ICE, producing electricity to power the EM and recharge the vehicle’s battery. A PHEV, as shown in Figure 2, is similar but has the added ability to have its battery re-energized by being plugged into an external electrical charging source. This allows PHEVs to carry larger electric motors and batteries, giving PHEVs’ electric vehicle (EV) mode an increased driving range with the ability to produce zero emissions. As a result, PHEVs can have higher overall miles per gallon (MPG) compared to vehicles that depend solely on an ICE. The added features and capabilities of PHEVs have made them a great area of focus, with companies and consumers looking for ways to further increase their electric driving range in between charges.

PHEVs can have a series, parallel, or a series-parallel drivetrain configuration. In a series configuration, a PHEV runs solely on the EM with an ICE only being used to power a generator and recharge the vehicle’s battery. In a parallel configuration, a PHEV can run on an ICE and electric motor individually or in a blended mode. The series-parallel configuration allows the PHEV to behave with a series or parallel configuration depending on which is more efficient at the given speed and torque request. A PHEV with the ability to run solely powered by its ICE, EM, or a combination of both is categorized as a full hybrid, or strong hybrid. In comparison, non-
plugin hybrids are typically categorized as mild hybrids, which comprise vehicles with limited hybrid technology, as they generally cannot run as a full EV due to the limited size of their batteries and electric motors.

Figure 1: Hybrid Electric Vehicle Design [10]

Figure 2: Plug-in Hybrid Electric Vehicle Design [11]
A larger battery will increase the all-electric range of a PHEV compared to an HEV, decreasing the fuel consumption over a given distance, thus leading to improved tank to wheel fuel economy for the vehicle and decreased harmful emissions [12]. This has resulted in sales rising over the years with 72,885 sold in 2016 as reported by the Office of Energy Efficiency and Renewable Energy. However, it is important to keep in consideration that additional battery weight decreases the attainable efficiency in miles per kWh and miles per gallon for a PHEV [13]. Therefore, an optimum in tank to wheel fuel economy can be reached when larger battery capacity and addition battery weight are both properly taken into consideration.

From data taken from 2015 modeled light duty passenger vehicles, on-road fuel economy was averaged at about 31 mpg [14]. Compared to a traditional HEV, a PHEV has the capability of alternating between using an electric motor, the internal combustion engine, and a mixture of both depending upon which driving mode is chosen. The modes include electric vehicle (EV) mode, charge depletion (CD) mode, charge sustaining (CS) mode, and internal combustion engine (ICE) mode. EV mode is when the PHEV runs solely on the battery and electric motor until it completes a predefined cycle or reaches a predefined minimum state of charge (SOC) in the vehicle’s storage system [15]. CD mode is when the PHEV runs primarily using the electric motor with a net decrease in SOC, with the ICE turning on when the power demand is too high for the electric motor to handle or if the SOC drops too low [15]. CS mode is when the PHEV is propelled by the electric motor, ICE, or a combination of both, with the constraint of maintaining a constant SOC in the battery [15]. ICE mode is where the PHEV runs solely on the ICE to propel itself. These modes allow maximum flexibility in reducing overall fuel consumption when coupled to a supervisory control that optimally chooses the driving mode that is most efficient for vehicle operation from those available. For example, in stop-and-go urban driving EV mode will be more efficient as
internal combustion engines are less efficient at low engine speed, usually characterized by revolutions per minute (RPM).

2.2. PHEV Energy Management Control Strategy Research

To further optimize and expand the all-electric range that a PHEV can travel, an onboard computer can use specific energy management strategies to determine which of its various modes to run in. With a proper control strategy, a PHEV can autonomously decide when and to what extent to use its two energy sources (battery energy or fuel) to increase its overall efficiency over a given drive cycle. This feature allows power management systems to strongly influence and increase PHEV fuel efficiency [16, 17, 18]. Giving some insight into the energy management strategies, switching between a PHEV’s various modes can be controlled automatically as a function of battery SOC, vehicle speed, engine speed, engine torque, environment temperature, battery temperature, and air conditioning need [19]. It is important to note that these decision factors all deal with the state of the PHEV itself in real-time. Therefore, having accurate readings of these factors proves to be very important. A battery’s SOC has proven to be the most difficult to measure with extensive research into improving measurement accuracy through open circuit voltage and coulomb counting (current integration) measurement techniques. Fortunately, research has shown that with proper evaluation of a battery’s state of health (SOH) at recharging and discharging, estimation error for SOC can be reduced to 1% at the operating cycle [20].

In many actual cases a PHEV’s driving mode can also be manually selected by the driver based upon which mode they desire to use at a given moment. It is this manual control option that has led to the consideration of a new energy management strategy that considers factors beyond the confines of the PHEV. Drivers of PHEVs have been known to manually switch between operating modes based on road conditions, such as traffic, that they anticipate facing on their
commutes. Choosing to run in EV mode as often as possible in instances of medium traffic or forcing the vehicle to run in CS mode in small instances of stop-and-go traffic, are strategies that consider a driver’s knowledge of their route to further improve their vehicle’s fuel economy. Given that modern vehicles have on-board global positioning systems (GPS) with included traffic, weather, road grade and hazard information, supervisory controllers like the one outlined in this research can optimize which driving mode will be utilized at what time to optimize fuel economy over the whole route.

The most examined strategies used to enhance PHEV fuel efficiency over a particular route use optimal control and optimization [21]. From the perspective of optimal control, many have considered the use of rule-based control strategies, such as fuzzy logic controls (FLC), driving mode classification, and dynamic feedback control [21]. Studies in [21, 22, 23] have shown that FLC type techniques make controllers easier to implement as operation merely requires matching immediate driving conditions to different prearranged scenarios for which to adjust the power contributions from the electric motor and ICE. In a similar fashion, driving mode classification techniques rely upon different parameters obtained from past and current driving conditions to characterize real-time driving patterns and adjust driving control strategies accordingly [24]. Studies investigating the benefits of using a combination of both FLC and driving mode classification techniques have been presented in the past by Langari and Won [25, 26]. As discussed in [27] and [28], dynamic feedback control approaches solve for the control strategies based on current and previous operations, which are easier for real-time implementation. Unfortunately, these algorithms are not able to reach global optimality in terms of power distribution over an entire route [21], thus investigations have then been made into dynamic programming (DP).
DP is a common optimization technique that has been used to obtain global optimality [29, 30, 31, 32, 33]. However, it uses a distinct few standardized dive cycles from the U.S. Department of Transportation to optimize power management for various other routes, serving more as a reference than an exact solution [21]. Fortunately, with the development and accessibility of trip prediction and modeling such as intelligent transportation systems (ITS), geographical information systems (GIS), and global positioning systems (GPS), models for individual trips can be accessed a priori [21]. These models can include information such as speed limits, traffic flow, and road grade from one location to another. In combination with a priori drive cycle knowledge, DP has been studied as a near globally optimized power management approach reinforcing the charge-depletion approach [21, 34]. While proven to efficiently optimize PHEV power management, especially with the integration of advanced route modeling, DP is a very computationally expensive technique that would require the optimization be performed offline [21]. Studies have been conducted to reduce the algorithm’s computational load to implement it in real-time. A two-scale dynamic programming approach solves for a globally optimized state of charge (SOC) model offline on a macro-level and then adapts the model on a micro-level in real-time onboard the vehicle [35].

This thesis will focus on implementing an optimal control technique that uses a combination of classification and FLC type rule-based controls to plan and execute an optimal SOC distribution for any given route based upon a priori knowledge (of the route and initial SOC) to optimize battery usage and improve fuel economy. This study is done to investigate the possibility of using a more easily implementable and less computationally heavy optimal control strategy over an entire route, using available route information to drive its solution to global optimality. Similar studies done in [36] and [16] show the possibility of using optimal control with
a priori route information to optimize fuel efficiency in PHEVs. While these studies only focus on using a route’s distance to distribute a PHEV’s SOC optimally, this study will use a route’s distance along with speed limits and traffic flow in a process called trajectory forecasting (TF) to drive its solution closer to global optimality.
3. Control Strategy

A PHEV’s driving mode is chosen based on internal factors, such as engine power output, needed acceleration, and SOC, usually not considering the impacts that outside factors also have on the fuel economy. Driving routes can have numerous factors, such as traffic conditions and physical road conditions, that affect a PHEV’s entire drive cycle and required power output. While many energy management strategies have focused more on a PHEVs’ internal factors for improving fuel economy, this study’s main contribution is its use of TF in its two-part control strategy. Many PHEV drivers manually control the modes of operation of their PHEVs based on information they have regarding their commute. In this study, the term TF will be used to enhance the range of operation of a PHEV by using information of a predetermined route to decide the best time to run in either all-electric (EV) mode, hybrid mode, or ICE mode. The first part of the control strategy will use TF to develop an optimal SOC distribution that uses both charge depletion (CD) and charge sustaining (CS) operation, while the second part will implement the distribution using the developed control strategy.

3.1. Part 1: Trajectory Forecasting Setup

In the first part of the strategy (corresponding MATLAB code is presented in the appendix section as Trajectory Forecasting Setup Code) the process initializes by taking the speed limits and traffic flow along the entire distance of a desired route as inputs. This is information that is easily accessible because of the widely available trip prediction and modeling services mentioned earlier. Examples of such a service, the service used as a reference for the route data used in this study, is Google Maps. Here we present a full traffic flow breakdown of a 64.4-mile route from Irvine, CA in Orange County to Panorama City, CA in Los Angeles County, shown in Figure 3. Due to the long distance of the route, the battery capacity of a PHEV would not be able to sustain the use of
EV and hybrid mode through the entirety of it. Thus, a decision must be made for when and how to use the battery so that an improved fuel efficiency for the overall route rather than instantaneous efficiency can be achieved. Using the traffic information, a driver can formulate a general plan for which PHEV propulsion mode to use during different flows of traffic. Shown in Figure 4, the following example explains the reasoning for assigning driving modes along the first quarter of the route.

The planned route takes a PHEV from an apartment to an uncrowded freeway known as the 73 freeway for a few miles, merging onto the well-known traffic heavy 405 freeway and continuing for a long distance. Little to no traffic, symbolized by the blue colored route portions, from the starting location in Irvine to the merging point between the two freeways indicates that the PHEV can run in ICE mode, hybrid mode, or EV mode with high efficiency. Large amounts of traffic congestion, shown by the red, in following section of the route resulting from the merging of cars between the freeways can then cause extremely reduced speeds and stops. Such stop-and-go traffic is handled more efficiently by a PHEV in EV and/or hybrid mode when charge from the battery is available for use. Experiencing medium traffic, as shown in orange a slight distance after the merging, hybrid mode and EV mode can both efficiently handle any remaining speed variations and reductions when charge is available. In consideration of overall route efficiency, conserving battery charge in sections of little to no traffic by using ICE mode would enable the use of EV and/or hybrid mode in later sections that contain higher traffic congestion when the amount of charge in the vehicle’s battery is not enough to support their consistent use.
Figure 3: Typical Google Maps route [37]

Figure 4: Google Maps route [37]; modes assigned according to only traffic conditions
Solely relying on traffic flow information, however, is not enough to optimize mode uses in PHEVs. Speed limits must also be considered to properly decide the best times to switch between propulsion modes, hence the use of both traffic flow and speed limits as inputs to the first part of the strategy. The route is then divided into evenly distributed segments by markers, with a rule-based classification strategy assessing and assigning a suggested priority driving mode at each marker according to the combination of current speed limit and traffic flow. Depending upon the total route distance, these markers separating the segments are spaced out every 0.05 miles for distances less than 11 miles, every 0.15 miles for a distance between 11 and 26 miles, and every 0.25 miles for distances greater than 26 miles. This is equivalent to setting a proper sampling rate based on a signal’s period. The suggested priority driving mode and route features at each marker are then assumed to be constant through a segment up until the next marker where the classification assessment changes. Consecutive segments with the same classification are then grouped together into sections. The classifications are set as priority 3, priority 2, and priority 1. A section classified as priority 3 is suggested to be run in EV mode. A section classified as priority 2 is suggested to be run in hybrid mode. A section classified as priority 1 is suggested to be run in ICE mode. The details of the rule-based classification used for this section of the study’s control strategy will be explained in further detail in Section 3.3. The resulting plan for the example route is shown in Figure 5.
The algorithm is then completed by approximating how much SOC is allocated to the different sections based upon their assigned priority classification, battery type, initial SOC level, max SOC level allowed, and minimum SOC level allowed. In this study, the vehicle simulator ADVISOR, developed by the National Renewable Energy Laboratory, is used to perform advanced vehicle simulations. ADVISOR is a simulation program developed to perform rapid analyses of the performance and fuel economy of conventional, electric, and hybrid vehicles, providing support for detailed simulations and studies of user-defined vehicle components [38]. Given a required/desired speed input, ADVISOR determines the drivetrain torques, speeds, and power requirements needed to meet the required/desired speed input [38]. This flow of information back through the drivetrain, from the tire to the axle to the gearbox and so on, makes it a backward-
facing vehicle simulation type program [38]. For all simulations, a common test parallel hybrid vehicle with the configuration shown in Figure 6 and described in Table 1 was used as the PHEV.

Figure 6: PHEV configuration snapshot from ADVISOR software [38]
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicle</td>
<td>VEH_SMCAR</td>
<td>Defines road load parameters for a hypothetical small car, roughly based on a 1994 Saturn SL1 vehicle</td>
</tr>
<tr>
<td>Fuel Converter</td>
<td>FC_SI41_emis</td>
<td>1991 Geo Metro 1.0L SI engine with maximum power of 41 kW @ 5700 rpm and peak torque of 81 Nm @ 3477 rpm</td>
</tr>
<tr>
<td>Exhaust After Treatment</td>
<td>EX_SI</td>
<td>Defines exhaust aftertreatment catalyst parameters for hypothetical vehicle equipped with a gasoline-powered SI engine (Masses, areas, etc. are scaled based on engine peak power)</td>
</tr>
<tr>
<td>Energy Storage</td>
<td>ESS_PB25</td>
<td>Parameters describe the Hawker Genesis 12V 26Ah 10EP sealed valve-regulated lead-acid (VRLA) battery</td>
</tr>
<tr>
<td>Motor</td>
<td>MC_AC75</td>
<td>Westinghouse, 75 kW, AC Induction motor with efficiency/loss data appropriate for a 320 V system</td>
</tr>
<tr>
<td>Transmission</td>
<td>TX_5SPD</td>
<td>Defines a 5-speed gearbox by defining gear ratios and gear number, and calling TX_VW to define loss characteristics</td>
</tr>
<tr>
<td>Torque Coupling</td>
<td>TC_DUMMY</td>
<td>Defines lossless belt drive with a motor-to-engine speed ratio that ensures the motor is at top speed when the engine is at top speed</td>
</tr>
<tr>
<td>Wheel Axle</td>
<td>WH_SMCAR</td>
<td>Defines tire, wheel, and axle assembly parameters for use of a hypothetical small car</td>
</tr>
<tr>
<td>Accessory</td>
<td>ACC_HYBRID</td>
<td>Defines standard accessory load data for use with a hybrid in ADVISOR</td>
</tr>
<tr>
<td>Powertrain Control</td>
<td>PTC_PAR</td>
<td>Defines all powertrain control parameters, including gearbox, clutch, hybrid and engine controls, for a parallel hybrid using a multi-speed gearbox</td>
</tr>
</tbody>
</table>

Table 1: ADVISOR PHEV configuration component descriptions [38]
For the purpose of approximating the SOC profile along the route, the program’s configured PHEV simulation was performed at different constant speeds in EV mode, 0 mph, 20 mph, 40 mph, 55 mph, and 60 mph, and the average current drawn by the selected motor for the test PHEV at each constant speed was recorded to form an equation for Amps per mph, shown in Figure 7 and Equation 1. To obtain an equation to understand the number of Amp-hours used per mile driven at a specific speed, Equation 2 is formed by taking the derivative of Equation 1. This approximation is for determining the PHEV’s battery usage in Amp-hours over the sections’ distance, as shown in Figure 8. The number of Amp-hours needed to run in EV mode for each segment are calculated and then used to determine the total needed for larger sections. Taking these values and dividing them by the total Amp-hour capacity of the chosen battery, we use the known initial SOC and determine the preferred total initial SOC profile for the route sections. This application of SOC profile approximation using only current from and to the battery and corresponding vehicle speed aims at using a linear version of the relation between SOC estimation and battery current draw found in coulomb counting. Coulomb counting is an efficient SOC estimation method currently used and researched in many battery applications that, with the pre-known capacity of the battery, calculates SOC by integrating the charging and discharging currents over the operating periods of the vehicle [39].

To address any issues of over or under approximating the total charge needed, an adjustment coefficient (AC) in terms of a percentage is added to the equation to increase or decrease the approximation and make it more fail-safe, giving Equation 3. To determine the value
\[ y = AC(0.0198x + 0.1689) \quad (3) \]

of AC, the 3 shortest routes of the 8 used in this thesis, routes 5, 7, and 8 whose characteristics are found in the route data section, are run with the test PHEV as a zero emissions vehicle and the SOC profile for every section along the routes is then recorded. The complete SOC profile recorded for each section priority type of the 3 routes is then compared to the amount approximated using Equation 3, shown in Table 2, Table 3, and Table 4. In comparing the actual SOC profile to the approximated SOC profile, it becomes noticeable that using different AC values for the different speed ranges of the three types of section priorities is more efficient. An AC value is chosen for each of the three section priorities based upon use of the value that results in the expected SOC profile that best matches the actual SOC profile across the multiple sections of the same priority. For the purpose of this study, the SOC profile is estimated using this method to simplify the non-linear dynamic nature of the vehicle’s EM and electronics. These AC values are only valid for the specific PHEV configuration shown earlier that will be consistently used throughout the study. To approximate the SOC profile for any other PHEV EM and electronics setup, another equation fit will have to be done.

![Figure 7: Plot for Amps per mph](image)
Using $AC = 0.87$ for priority 3 sections, $AC = 0.42$ for priority 2 sections, and $AC = 0.34$ for priority 1 sections, the preferred SOC profile for each section and possible SOC profile allowed by the available charge in the battery are used to assess the viability of the suggested modes of each section. The comparison of the SOC profile using the AC values for the three routes can be seen in Table 2, Table 3, and Table 4. Depending upon what the estimated SOC profile along the entire route and the current battery SOC is, four different proposed high efficiency driving scenarios for how to switch between the PHEV’s driving modes along the route can occur.

### Table 2: Route 5 (20.1 miles) SOC estimation comparison

<table>
<thead>
<tr>
<th>Section Priority Type</th>
<th>Estimated SOC Use</th>
<th>Actual SOC Use</th>
<th>% Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority 1</td>
<td>0.2415</td>
<td>0.2259</td>
<td>6.906</td>
</tr>
<tr>
<td>Priority 2</td>
<td>0.2053</td>
<td>0.2041</td>
<td>0.5879</td>
</tr>
<tr>
<td>Priority 3</td>
<td>0.2027</td>
<td>0.1991</td>
<td>1.808</td>
</tr>
</tbody>
</table>

### Table 3: Route 7 (17.1 miles) SOC estimation comparison

<table>
<thead>
<tr>
<th>Section Priority Type</th>
<th>Estimated SOC Use</th>
<th>Actual SOC Use</th>
<th>% Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority 1</td>
<td>0.2361</td>
<td>0.2216</td>
<td>6.543</td>
</tr>
<tr>
<td>Priority 2</td>
<td>0.1882</td>
<td>0.1753</td>
<td>7.359</td>
</tr>
<tr>
<td>Priority 3</td>
<td>0.1376</td>
<td>0.1431</td>
<td>-3.843</td>
</tr>
</tbody>
</table>

Figure 8: Plot for Amp-hours per mph
<table>
<thead>
<tr>
<th>Section Priority Type</th>
<th>Estimated SOC Use</th>
<th>Actual SOC Use</th>
<th>% Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority 1</td>
<td>0.0897</td>
<td>0.091</td>
<td>-1.429</td>
</tr>
<tr>
<td>Priority 2</td>
<td>0.0672</td>
<td>0.0765</td>
<td>-12.16</td>
</tr>
<tr>
<td>Priority 3</td>
<td>0.0535</td>
<td>0.062</td>
<td>-13.71</td>
</tr>
</tbody>
</table>

Table 4: Route 8 (6.5 miles) SOC estimation comparison

A newly proposed construct for achieving high efficiency is broken down with the following four driving scenarios:

1) within +0.125 SOC of the PHEV’s all-electric range, EV mode will be used to drive all sections of the route

2) enough SOC to drive suggested EV and hybrid mode sections in EV mode, and drive suggested ICE mode sections in hybrid mode using the leftover SOC

3) enough SOC to drive suggested EV sections in EV mode with the leftover SOC being used to drive suggested hybrid mode sections in hybrid mode, and remaining ICE suggested sections driven in ICE mode

4) enough SOC to drive some suggested EV mode sections in EV mode and all remaining sections of the route in ICE mode

According to the resulting driving scenario, the current SOC of the battery is distributed among all the sections according to highest priority, attempting to fulfill the sections’ SOC needs. Among multiple sections with the same priority, a sublevel priority is given based on length of sections where longer sections have a higher sublevel priority. In driving scenarios where there are sections driven in hybrid mode, the remaining SOC is taken and distributed between all hybrid mode driven sections according to the percentage of the total hybrid mode section distance each one contains, Equation 4. Finalizing the SOC distribution, the created plan dictates how much of the PHEV’s total SOC can be used by each section, setting a profile of minimum SOC levels the battery can
drain to during each categorized section of the route. This part of the control strategy then outputs the SOC minimum profile, the priority levels of the sections, and preferred high efficiency driving scenario.

\[ \text{Hybrid Section SOC} = \text{Remaining SOC} \times \frac{\text{Hybrid Section Distance}}{\sum_{i=1}^{n} \text{Hybrid Section Distance}_i} \]  

(4)

3.2. Part 2: Real-Time Trajectory Forecasting Control Strategy

For part two of the control strategy, the outputs from part one are sent as variables to the ADVISOR Simulink PHEV model and TF control strategy block, shown in Figure 9 and Figure 10, respectively. The PHEV simulation is then initialized with ADVISOR’s default battery parameters and the set route from part 1 in Section 3.1, both shown in Figure 11. Detailed descriptions of ADVISOR’s default battery parameters can be found within its documentation [38]. Once the simulation begins, the PHEV’s drivetrain experiences a desired torque load request that tries to be met every iteration of the simulation to properly drive at the requested speed along the route. The first step in the TF control strategy block is setting the current SOC minimum, done within the SOC min assignment function block boxed in blue within Figure 10. It takes the vehicle’s current distance traveled as an input and uses it to output what SOC minimum from the SOC distribution plan it should follow. The TF control strategy block, boxed in red within Figure 10, then takes the ICE’s torque load request as an input and gives a percentage of the ICE’s available torque load as an output to satisfy the request. The amount of desired torque that is left to be satisfied is left for the electric motor (EM) to fulfill, represented by Equation 5. The code used in the control strategy function block, boxed in red within Figure 10, and the SOC min assignment function block can be found in the appendix section of this study. A block by block walk-through of the ADVISOR Simulink model with corresponding descriptions can be found in
the online ADVISOR documentation [38] and the study of Intelligent Control of Parallel Hybrid Electric Vehicles performed by Glenn [40].

\[ T_{EM} = T_{LOAD} - T_{ICE} \quad (5) \]

Figure 9: ADVISOR PHEV Simulink model; control strategy block (in red) [38]

Figure 10: ADVISOR control strategy block interior [38]; TF control strategy function block additive (in red); SOC min assignment function block additive (in blue)
Considering the propulsion modes in a PHEV, the control strategy can attempt to satisfy the desired torque load with the ICE and no contribution from the EM, with the EM and no contribution from the ICE, or with contributions from both. EV and ICE modes are implemented by setting the output of the TF control strategy block to 100% or 0% of the ICE’s available torque output respectively. In an EV mode driven route section, the PHEV will run only using the battery until the assigned SOC minimum is reached or it reaches the end of the route section. Hybrid mode is implemented by using the heuristic control approach, FLC, because of its easy implementation, low computational cost, and ability to work on complex non-linear models. The use of FLC will work to ensure that the specific amount of SOC distributed to each of the hybrid sections will be used evenly along each of their lengths by determining what percentage of the ICE’s available torque output will be used. The exact breakdown of the FLC method specific to this study’s control strategy will be explained in further detail in a later section. For driving scenarios 3 and 4, the
sections along a route that require only the use of the ICE may require safety conditions that ensure any SOC recovered through regenerative braking is accounted for and used. This is done by having the torque contribution from the ICE be only 85% or 90% of its available load if the amount of SOC regenerated puts the overall battery SOC at 0.01 or 0.005, respectively, above the section’s assigned SOC minimum. The remaining 10% or 15% of the requested torque load will then be fulfilled by the EM. During the use of EV mode, if the available power and torque output from the EM is slightly insufficient to allow the PHEV to achieve a desired speed and the vehicle’s SOC is above the current minimum, the control strategy will prioritize efficiency by keeping the ICE off and not permit it to assist in providing the remaining needed torque. This will cause a slight throttling of the PHEV’s requested speed in return for less fuel burning, where the requested speeds and resulting accelerations are not fully met by the vehicle. The methodology behind this study’s control strategy will allow the PHEV’s fuel efficiency to be optimized from a global standpoint.

3.3. Rule-based Classification

The rule-based classification in this study mimics the use of if-then statements for its logic, where explicit outcomes are triggered from different combinations of input parameters. When implementing, we consider the traffic flow and speed limits along a route as the input parameters to the logic. Traffic flow can be broken down into three categories, zero-to-light traffic, medium traffic, and heavy traffic, similarly to how GPS services define traffic flow. What these categories represent is how close to the speed limit a driver can drive their vehicles, setting zero-to-light traffic as driving at 100% of the speed limit, medium traffic as driving at 65% of the speed limit, and heavy traffic as driving at 25% of the speed limit. For speed limits, we consider a total of 7 different possibilities based on different speed limits a driver can encounter, from parking lots to school areas, to residential streets, to rural streets, to business districts, to different highways.
proposed combinations of the different traffic flow and speed limits, along with their triggered outcomes created in this study for use as desired input parameters for the control strategy can be expressed in Table 5 and Table 6. Table 5 shows the resulting priority the different input parameter combinations trigger, while Table 6 shows an approximated speed a car would drive at the given traffic and speed limit combination. The values from Table 6 are used as approximated speed inputs into Equation 3 for approximating the SOC for the PHEV during different sections of the route.

<table>
<thead>
<tr>
<th>Speed Limit</th>
<th>Zero/Light Traffic</th>
<th>Medium Traffic</th>
<th>Heavy Traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>25</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>35</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>45</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>55</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>65</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>75</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 5: Priority value table

<table>
<thead>
<tr>
<th>Speed Limit</th>
<th>No/Light Traffic</th>
<th>Medium Traffic</th>
<th>Heavy Traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>15</td>
<td>9.75</td>
<td>3.75</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
<td>16.25</td>
<td>6.25</td>
</tr>
<tr>
<td>35</td>
<td>35</td>
<td>22.75</td>
<td>8.75</td>
</tr>
<tr>
<td>45</td>
<td>45</td>
<td>29.25</td>
<td>11.25</td>
</tr>
<tr>
<td>55</td>
<td>55</td>
<td>35.75</td>
<td>13.75</td>
</tr>
<tr>
<td>65</td>
<td>65</td>
<td>42.25</td>
<td>16.25</td>
</tr>
<tr>
<td>75</td>
<td>75</td>
<td>48.75</td>
<td>18.75</td>
</tr>
</tbody>
</table>

Table 6: Approximated speed value table

3.4. Fuzzy Logic Control

FLC is a heuristic form of control logic that focuses on using a practical method to produce a solution not guaranteed to be the optimal solution, but rather a solution that is enough for the
immediate goal. It relies on the concept of partial truth, or the degree of truth, to determine which user defined rules are active in calculation of the desired solution [41]. The basics of fuzzy logic can be explained by breaking it down into four main parts: fuzzification, rule-base, inference mechanism, and defuzzification. A more comprehensive breakdown of fuzzy logic control can be found in Fuzzy Control [42].

Before the step of fuzzification, we take the input and output control variables, referred to as linguistic variables, $u_i$, and categorize them into areas called membership functions. For each hybrid mode section in the route, their given amount of SOC is used up along their distance according to a rate, SOC per mile, found initially at the beginning of the section. This desired rate is found using Equation 6, taking the initial SOC at the start of the hybrid section, subtracting the minimum SOC value it can drain the battery to in the section, and dividing it by the hybrid section’s length. The current rate is calculated every iteration along the section in a similar way using Equation 7, but instead uses the difference in SOC between the previous simulation iteration and

$$\text{Desired Rate} = \frac{\text{Initial Section SOC} - \text{Section SOC Min}}{\text{Section Distance}} \tag{6}$$

$$\text{Current Rate} = \frac{\text{Previous SOC} - \text{Current SOC}}{\text{Traveled Distance}} \tag{7}$$
the current simulation iteration, dividing it by the distance traveled between the them. Its value changes according to how much of the desired torque load is met by the EM each iteration. How much higher or lower the current rate is compared to the desired rate is taken as a percentage and referred to as the error, Equation 8. From one iteration to the next, we determine the change in error to understand the speed at which the error increases or decreases, Equation 9.

\[
e(t) = \frac{\text{Desired Rate} - \text{Current Rate}}{\text{Desired Rate}} \quad (8)
\]

\[
\Delta e(t) = \text{Current Error} - \text{Previous Error} \quad (9)
\]

The error and change in error are the input linguistic variables that will be categorized into membership functions with corresponding linguistic values, shown in Figure 13. Membership functions are chosen to be a set of shapes, overlapping triangles for the purpose of this study, that break up the domain of the linguistic variables into smaller sections. For the input linguistic variables, each triangular membership function corresponds to a linguistic value, ranging from 1 to 7 for the purpose of this study. These values are descriptors for how a human would describe the size of the input linguistic variables. The linguistic values represent the size of the variables as follows:

1 to represent “large negative”
2 to represent “medium negative”
3 to represent “small negative”
4 to represent “zero”
5 to represent “small positive”
6 to represent “medium positive”
7 to represent “large positive”
The output linguistic variable, defined as $K(t)$, will be a coefficient that regulates the percentage of ICE torque used from what is available at the current engine speed. The possible linguistic values it can take, and corresponding membership functions, will be formed similarly to those of the input linguistic variables. A difference will be that for the $K(t)$ output, the positive linguistic values will refer to numerical values greater 0.4, the negative linguistic values will refer to values less than 0.4, and the zero linguistic value will refer to values of about 0.4, as shown in Figure 14. Cases showing the general effects of the input variables’ values on the output variable value can be seen in Table 7.

<table>
<thead>
<tr>
<th>Case</th>
<th>$e(t)$</th>
<th>$\Delta e(t)$</th>
<th>$K(t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&lt;0</td>
<td>&lt;0</td>
<td>&gt;0.4</td>
</tr>
<tr>
<td>2</td>
<td>&lt;0</td>
<td>&gt;0</td>
<td>&gt;0.4</td>
</tr>
<tr>
<td>3</td>
<td>&gt;0</td>
<td>&lt;0</td>
<td>&lt;0.4</td>
</tr>
<tr>
<td>4</td>
<td>&gt;0</td>
<td>&gt;0</td>
<td>&lt;0.4</td>
</tr>
</tbody>
</table>

Table 7: Qualitative summary of input variable influence on output variable

Figure 13: Membership functions for error and change in error
3.4.1. Fuzzification

It is possible for a linguistic variable to be categorized as more than one linguistic value due to the overlapping of membership functions. The plots within Figures 13 and 14 are of a function $\mu_i$ versus the numerical values of the linguistic variables. The function $\mu_i$ quantifies the certainty, or degree of truth, that a linguistic variable can be classified as a specific linguistic value and can range from 0 to 1. With the numerical value of a linguistic variable possibly falling into one or two membership functions, there will be a certainty value, $\mu_i$, given to each membership function’s corresponding linguistic value. A set of numerical values that can be described by $\mu_i$ being a distinct linguistic value is called a fuzzy set and is denoted by $A_i$. Membership functions can define a fuzzy set of $A_i$ for a linguistic variable of $u_i$ in the form of Equation 10 [42].

$$\mu_{A_i}(u_i) = \text{certainty} \quad (10)$$

For example, Figure 15 shows an instant in time during the drive cycle when the error input is 0.025 and the change in error is 0.039. For the error input, the line crosses the 4 membership function at 0.5 and crosses the 5 membership function at 0.5. The fuzzification of the input variable says that there is a 50% certainty that the error input is a 4, meaning about “zero” error, and a 50% certainty that error input is a 5, meaning a “small positive” error. For the change in error input, the line crosses the 5 membership function at 0.44 and crosses the 6 membership function at 0.56. The fuzzification of the input variable says that there is a 44% certainty that the change in error input
is a 5, meaning a “small positive” change in error, and a 56% certainty that change in error input is a 6, meaning a “medium positive” change in error.

![Fuzzification example scenario](image)

\[
\mu_{\text{zero}}(e(t)) = \mu_{\text{small pos}}(e(t)) = 0.5 \\
\mu_{\text{small pos}}(\Delta e(t)) = 0.44, \mu_{\text{med pos}}(\Delta e(t)) = 0.56
\]

### 3.4.2. Rule-Bases

Linguistic values can be used to specify a set of rules that capture an expert’s knowledge about how to control a system and its dynamics, a rule-base [42]. The rules take the following general form,

\[
\text{If premise, Then consequent.} \quad (11)
\]

where the premise is associated with the fuzzy linguistic inputs and the consequent is associated with the resulting linguistic output. A generic rule form for two inputs and one output that will be used for the purpose of this study is
If the error is "" and the change in error is " ",

Then the percentage of ICE torque, K, used is " ".

(12)

With two inputs and seven linguistic values for each of them, there are at most \(7^2 = 49\) possible rules within the fuzzy logic of this study. A tabular representation referred to as a rule table, is used to properly list all possible rules in a convenient way, as shown in Table 8.

<table>
<thead>
<tr>
<th>Error</th>
<th>Change in Error</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 8: Rule Table

3.4.3. Inference Mechanism

The inference mechanism, consisting of two steps, represents the decision-making process of an expert. The first step is referred to as matching, where it is determined which rules apply to the current situation or are considered on. Continuing with the example from the fuzzification step, the following rules are determined to be on:

1. If the error is “zero” and the change in error is “small positive”, Then the percentage of ICE torque, K, used is “small negative”.

2. If the error is “zero” and the change in error is “medium positive”, Then the percentage of ICE torque, K, used is “medium negative”.

31
3. **If** the error is “small positive” and the change in error is “small positive”, **Then** the percentage of ICE torque, \( K \), used is “medium negative”.

4. **If** the error is “small positive” and the change in error is “medium positive”, **Then** the percentage of ICE torque, \( K \), used is “large negative”.

Before the second step, a method for quantifying the overall linguistic premise of each active rule must be agreed upon. The main idea behind this is agreeing on how to quantify the logical “and” operation that combines the individual linguistic input variables into the premise [42]. This value represents the certainty that the rule to which the premise belongs applies to the current situation and is denoted by \( \mu_{\text{premise}(i)} \). Common methods for doing this are known as the minimum and the product. The minimum method uses the minimum of the two membership function certainties, while the product method multiplies them together [40]. The minimum method is used for the purpose of this study, as shown in Equation 13.

\[
\mu_{\text{premise}(i)} = \min\{\mu_{A_i}(e(t)), \mu_{A_i}(\Delta e(t))\} \quad (13)
\]

\[
\mu_{\text{premise}(1)} = \min\{0.5, 0.44\} = 0.44
\]
\[
\mu_{\text{premise}(2)} = \min\{0.5, 0.56\} = 0.5
\]
\[
\mu_{\text{premise}(3)} = \min\{0.5, 0.44\} = 0.44
\]
\[
\mu_{\text{premise}(4)} = \min\{0.5, 0.56\} = 0.5
\]

The second step then establishes the conclusion for each active rule. The membership function for the consequent reached by each rule quantifies how certain the rule is that the output variable should take on a certain linguistic value, Equation 14. These membership functions now define the implied fuzzy sets. The justification for using the minimum operator in Equation 14 to
quantify the certainty is that we can be no more certain about our consequent than our premise [42].

\[
\mu_i(K(t)) = \min\{\mu_{\text{premise}(i)}, \mu_{A_i}(K(t))\} \quad (14)
\]

\[
\begin{align*}
\mu_1(K(t)) &= \min\{0.44, \mu_{\text{small neg}}(K(t))\} = 0.44 \\
\mu_2(K(t)) &= \min\{0.5, \mu_{\text{med neg}}(K(t))\} = 0.5 \\
\mu_3(K(t)) &= \min\{0.44, \mu_{\text{med neg}}(K(t))\} = 0.44 \\
\mu_4(K(t)) &= \min\{0.5, \mu_{\text{large neg}}(K(t))\} = 0.5
\end{align*}
\]

3.4.4. Defuzzification

The defuzzification process operates on the implied fuzzy sets produced by the inference mechanism and combines their effects to provide the most certain control output \(\mu^{\text{crisp}}\) [42]. For the scope of this study, this output will be achieved using the most popular method, the “center of gravity” (COG) method. The method is defined by Equation 15 as:

\[
K = \mu^{\text{crisp}} = \frac{\sum_{i=1}^{n} b_i \int \mu_i}{\sum_{i=1}^{n} \int \mu_i} \quad (15)
\]

where \(b_i\) is the center of the output membership for the consequent of rule \(i\) and \(\int \mu_i\) is the area underneath the output membership function “chopped off” at a height of \(\mu_i(K(t))\) for the consequent of rule \(i\). The calculation gives the K coefficient that will best regulate the ICE torque output so that the desired charge consumption rate and resulting preferred SOC profile can be achieved in the hybrid section.

\[
K = \frac{(0.2667 \times 0.0915) + (0.1333 \times 0.1000) + (0.1333 \times 0.0915) + (0 \times 0.1000)}{0.0915 + 0.1000 + 0.0915 + 0.1000} = 0.1303
\]
4. Simulation

4.1. Route Data

The data used in this study is a set of 8 different routes planned out using Google Maps services. Distances are varied between every route, from 6.5 miles to 70 miles. The traffic flow breakup from Google Maps, e.g. route 1 shown in Figure 16, is based on historical and real-time data gathered from traffic sensors and cellphone users. Each Google Maps route is used to portion out the changing traffic sections of the route data for the purpose of testing. Traffic is a rough representation of the speed relative to the speed limit a vehicle is likely to experience when on the road. This usage of traffic information and different speed limits according to road type along the route allows for the use of Table 6, shown earlier, to approximate the speed a vehicle would have at different points along a route. However, using the constant values that each combination in Table 6 provides would result in the vehicle unrealistically traveling at constant speeds during every portioned-off section of the route. To produce more realistic route data, randomized variations are added to the approximated speed values. The variations reduce the speeds presented in Table 6 an additional 0%-5% for each type of traffic. This results in new drive cycles, e.g., shown in Figure 17 derived from Figure 16, that appears to be more realistic as compared to the EPA developed Urban Dynamometer Driving Schedule (UDDS) used for light duty vehicle emissions and fuel economy testing, as shown in Figure 18 [44].
Figure 16: Google Maps route 1 [37]

Figure 17: Route 1 drive cycle

Figure 18: EPA created UDDS drive cycle [44]
4.2. Simulation Parameters

The results shown in this study will be for the parallel hybrid PHEV control strategies in ADVISOR, charge depletion (CD) and charge sustaining (CS), and the trajectory forecasting (TF) control strategy of this study. The numerical values usable for comparing the performance of the three control strategies are the “Fuel Economy” (FE) value in miles per gallon (mpg) and the “Gas Equivalent” (GE) value in miles per gallon of gasoline equivalent (mpgge) given in the results window of the ADVISOR simulations. The FE value is calculated using Equation 16 and the GE value is calculated using Equation 17. The GE value gives a better understanding of the PHEV’s fuel efficiency because it considers the consumption of both the liquid fuel and electric power source in the vehicle. In instances where the vehicle’s route can be driven nearly all in EV mode, with only a small portion left to be driven in ICE or Hybrid mode, the amount of liquid fuel that would be used would significantly smaller numerically than the length of route. These scenarios would then result in very skewed FE values that would seem unrealistic. Thus, the FE value will be ignored, and the GE value will serve as the key value for quantifying and comparing control strategy performance. The parameters of the parallel hybrid control strategies in ADVISOR are defined as shown in Table 9.

\[
\text{Fuel Economy} = \frac{\text{Total Route Distance}}{\text{Total Gasoline Consumption}} \quad (16)
\]

\[
\text{Gas Equivalent} = \frac{\text{Total Route Distance}}{\text{Total Gasoline Consumption} + \frac{\text{kWh Needed to Recharge to Initial SOC}}{\text{kWh per Gallon of Gasoline}}} \quad (17)
\]
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs_hi_soc</td>
<td>highest desired SOC, used as initial SOC for every case</td>
</tr>
<tr>
<td>cs_lo_soc</td>
<td>lowest desired battery SOC</td>
</tr>
<tr>
<td>cs_electric_launch_spd_lo</td>
<td>speed below which vehicle operates as ZEV at low SOC</td>
</tr>
<tr>
<td>cs_electric_launch_spd_hi</td>
<td>speed below which vehicle operates as ZEV at low SOC</td>
</tr>
<tr>
<td>cs_off_trq_frac</td>
<td>required fraction of max torque when SOC &lt; cs_lo_soc below which engine shuts off</td>
</tr>
<tr>
<td>cs_min_trq_frac</td>
<td>torque as a fraction of max torque engine puts out when required is below this value, when SOC &lt; cs_lo_soc</td>
</tr>
<tr>
<td>cs_charge_trq</td>
<td>accessory-like torque load on engine that goes to recharging the batteries whenever the engine is on</td>
</tr>
<tr>
<td>cs_charge_deplete_bool</td>
<td>charge depleting hybrid strategy flag, 1=&gt; use charge deplete strategy, 0=&gt; use charge sustaining strategy</td>
</tr>
<tr>
<td>cs_electric_decel_spd</td>
<td>speed above which no engine shut down occurs due to low torque requests</td>
</tr>
</tbody>
</table>

Table 9: ADVISOR parallel hybrid control strategy parameters

The cs_charge_deplete_bool parameter is set to 0 to enable the CS strategy and set to 1 to enable the CD strategy. The parameter of interest for running these simulations is cs_electric_launch_spd_hi. For the CS strategy, the parameter is set to 11.18 m/s (≈ 25.0 mph), shown in Table 10, to allow EV mode to be used when the vehicle is at or below 25 mph and has adequate SOC. This allows the normal parallel hybrid PHEV control strategy to save electrical energy, represented by the level of SOC, at higher speeds where the ICE and hybrid modes are more efficient, leaving more usable SOC for EV mode at low speeds. The CS strategy will result in the PHEV using its battery sparingly, making it available for use during large amounts of the route. Using the CD strategy, the parameter is set to 33.53 m/s (≈ 75.0 mph), shown in Table 10, to allow EV mode to be used when the vehicle is at or below 75 mph and has adequate SOC. The max speed achievable in any of the routes is 75 mph, allowing the PHEV to use the EV mode regardless of speed provided there is adequate SOC. The PHEV will start the beginning a route in EV mode and continue until the vehicle’s usable SOC is depleted, sustaining its SOC at the
minimum allowable level until the end of the route. The CD strategy will have the PHEV always attempt to use up all its SOC first regardless of current and future route conditions.

For testing the TF control strategy, its ADVISOR parameters, shown in Table 10, will match the parameters used for the CS control strategy in all except the cs_electric_launch_spd_hi and cs_electric_decel_spd. These parameter values will change according to one of the four high efficiency driving scenarios explained earlier in the methodology. The value for cs_charge_deplete_bool will not have any effect on the strategy’s performance due to its Simulink additives, therefore it will be left as 0.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>CS values</th>
<th>CD values</th>
<th>TF values</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs_hi_soc</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>cs_lo_soc</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>cs_electric_launch_spd_lo</td>
<td>0 m/s</td>
<td>0 m/s</td>
<td>0 m/s</td>
</tr>
<tr>
<td>cs_electric_launch_spd_hi</td>
<td>11.18 m/s</td>
<td>33.53 m/s</td>
<td>11.18 m/s (driving scenarios 3 &amp; 4)</td>
</tr>
<tr>
<td>cs_off_trq_frac</td>
<td>0.35</td>
<td>0.35</td>
<td>0.35</td>
</tr>
<tr>
<td>cs_min_trq_frac</td>
<td>0.48</td>
<td>0.48</td>
<td>0.48</td>
</tr>
<tr>
<td>cs_charge_trq</td>
<td>0.25*min(fc_max_trq)</td>
<td>0.25*min(fc_max_trq)</td>
<td>0.25*min(fc_max_trq)</td>
</tr>
<tr>
<td>cs_charge_deplete_bool</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>cs_electric_decel_spd</td>
<td>11 m/s</td>
<td>33 m/s</td>
<td>11 m/s (driving scenarios 3 &amp; 4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24 m/s (driving scenario 2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>33 m/s (driving scenario 1)</td>
</tr>
</tbody>
</table>

Table 10: ADVISOR parallel hybrid control strategy parameter values
4.3. Results

The route 1 drive cycle, shown in Figure 19, is 62 miles long with the TF control strategy resulting in the best GE value, 53.3 mpgge. Its GE value is 4.7% greater than that produced by the CS control strategy, 50.9 mpgge, and 3.5% greater than that produced by the CD control strategy, 51.5 mpgge.

![Figure 19: Route 1 Drive Cycle](image1)

The route 2 drive cycle, shown in Figure 20, is 65 miles long with the TF control strategy resulting in the best GE value, 52.8 mpgge. Its GE value is 10.2% greater than that produced by the CS control strategy, 47.9 mpgge, and 10.2% greater than that produced by the CD control strategy, 47.9 mpgge.

![Figure 20: Route 2 Drive Cycle](image2)
The route 3 drive cycle, shown in Figure 21, is 70.7 miles long with the TF control strategy resulting in the best GE value, 51.1 mpgge. Its GE value is 6.2% greater than that produced by the CS control strategy, 48.1 mpgge, and 8.3% greater than that produced by the CD control strategy, 47.2 mpgge.

![Figure 21: Route 3 Drive Cycle](image1)

The route 4 drive cycle, shown in Figure 22, is 52.2 miles long with the TF control strategy resulting in the best GE value, 50.5 mpgge. Its GE value is 0.60% greater than that produced by the CS control strategy, 50.2 mpgge, and 2.4% greater than that produced by the CD control strategy, 49.3 mpgge.

![Figure 22: Route 4 Drive Cycle](image2)
The route 5 drive cycle, shown in Figure 23, is 20.1 miles long with the TF control strategy resulting in the best GE value, 79.1 mpgge. Its GE value is 49.81% greater than that produced by the CS control strategy, 52.8 mpgge, and 10.47% greater than that produced by the CD control strategy, 71.6 mpgge.

![Figure 23: Route 5 Drive Cycle](image)

The route 6 drive cycle, shown in Figure 24, is 40.7 miles long with the TF control strategy resulting in the best GE value, 53.1 mpgge. Its GE value is 0.19% greater than that produced by the CS control strategy, 53 mpgge, and 2.7% greater than that produced by the CD control strategy, 51.7 mpgge.

![Figure 24: Route 6 Drive Cycle](image)
The route 7 drive cycle, shown in Figure 25, is 17.1 miles long with the TF control strategy resulting in the best GE value, 93.7 mpgge. Its GE value is 84.45% greater than that produced by the CS control strategy, 50.8 mpgge, and 41.98% greater than that produced by the CD control strategy, 66 mpgge.

![Figure 25: Route 7 Drive Cycle](image)

The route 8 drive cycle, shown in Figure 26, is 6.5 miles long with the TF control strategy resulting in the best GE value, 88.3 mpgge. Its GE value is 84.5% greater than that produced by the CS control strategy, 105.35 mpgge, and 87.9% greater than that produced by the CD control strategy, 47 mpgge.

![Figure 26: Route 8 Drive Cycle](image)
<table>
<thead>
<tr>
<th>Route</th>
<th>Control Strategy</th>
<th>Gas Equivalent (mpgge)</th>
<th>Distance (miles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CS</td>
<td>50.9</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>51.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>53.3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CS</td>
<td>47.9</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>47.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>52.8</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CS</td>
<td>48.1</td>
<td>70.7</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>47.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>51.1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CS</td>
<td>50.2</td>
<td>52.2</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>49.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>50.5</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>CS</td>
<td>52.8</td>
<td>20.1</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>71.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>79.1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>CS</td>
<td>53</td>
<td>40.7</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>51.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>53.1</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>CS</td>
<td>50.8</td>
<td>17.1</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>93.7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>CS</td>
<td>43</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>88.3</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: ADVISOR mpgge results for all route and control strategy combinations

Figures showing the ADVISOR results window for each route simulation for each of the three control strategies can be found in the appendix section of this study. These figures will show graphs of the vehicle’s drive cycle, SOC profile, emissions output, and actual torque output of the engine.
5. Summary and Conclusions

In all the routes tested, the TF-based control strategy achieved greater fuel efficiency. This is important because PHEV fuel economy enhancements can lead to a tremendous reduction in fuel consumption for the nation and possibly shorter payback time for customers in terms of vehicle investment [35]. The TF-based control strategy proved to be the most beneficial in routes 2, 3, 5, 7, and 8. In routes 1, 4, and 6, the TF-based control strategy gave results that were only slightly better than the other control strategies.

Routes 2 and 3 are long distance with roughly half of their distances requiring the test PHEV to travel at speeds at or below 25 mph. These priority 3 categorized sections for the routes require all the PHEV’s available SOC and more to drive them in the most efficient propulsion mode, EV mode, putting them in driving scenario 4. To assure this is done, the PHEV’s control strategy must prevent its SOC from reaching zero in all other sections, as the priority 3 sections are spread throughout the route. Typical CS and CD control strategies are unable to do this as they cannot determine the optimal time to use or preserve a PHEV’s SOC. In contrast, the developed TF-based control strategy looks at the traffic flow and speed limits along a route to make these decisions for the vehicle.

Routes 1, 4, and 6 are approximated to require the majority of the PHEV’s SOC to run priority 3 sections in EV mode, while priority 2 sections use the remaining SOC available to run in hybrid mode and the priority 1 sections run solely using the ICE. The routes therefore fall into driving scenario 3. Like routes 2 and 3, the TF control strategy has an advantage in knowing the most efficient times to use EV mode. It keeps the lead in fuel efficiency when compared to the CS and CD control strategies due to the implemented FLC algorithm that determines the level of hybridization in priority 2 sections. Safety protocols enabling a level of hybridization to be used
in ICE only sections, priority 1 sections, when regenerative braking recovers enough SOC also keep the TF control strategy efficient.

Routes 5 and 7 are medium distance, requiring more SOC to run them in EV mode than is available in the PHEV’s battery. However, the amount of extra SOC the TF control strategy approximates is needed resides within a range of 0-0.125, putting them in driving scenario 1. Within this range of needed SOC, the developed strategy determines that the most efficient course of action is using EV mode from the routes’ starting point until the battery is depleted and ICE mode is the only option for propulsion that can be used for the remaining small amount of distance. The TF control strategy’s ability to know a route’s conditions and approximate SOC allows it to make full use of a PHEV’s battery between charges, unlike the CS control strategy that always has the goal of sustaining the SOC of the battery [45]. The key difference in performance between the CD and TF control strategies is that the CD strategy still allows the ICE to provide any unmet torque request during times when the PHEV’s SOC is still above the allowed minimum value and attempted to be driven in EV mode. This causes more gas consumption than in the TF strategy where some speed throttling is allowed in return for less fuel consumption.

Route 8 is a short distance well within the all-electric range of the PHEV, making it a driving scenario 1. Thus, the TF control strategy drives the PHEV entirely in EV mode with its speed throttling ensuring that the ICE does not consume any fuel along the route. The CS control strategy uses a mix of the ICE and EM to ration the vehicle’s SOC to last the length of a driving route, information that it does not have access to. Comparable to the TF control strategy, the CD control strategy also attempts to focus on using only the PHEV’s battery to drive the entire route but uses the ICE in moments when the power and torque from the EM is insufficient to fully satisfy
the requested speed. The results are unnecessary fuel consumption in return for SOC rationing and prioritization of speed request fulfillment creating less fuel-efficient outcomes.

One downfall of the TF control strategy is its inability to account for sudden unforeseen changes in \textit{a priori} route information once the vehicle has already begun traveling on its designed drive cycle. Future work for this strategy could attempt to resolve this issue using machine learning to recognize sudden changes in a vehicle’s route, correctly adjusting its drive cycle and use of propulsion modes. Additionally, more input parameters such as urgency of personal travel needs and the effects of weather on the road will also be considered in future development of the TF control strategy. To further reach global optimality, a control strategy must be able to consider as many influential input features as necessary.

Overall, the approach presented in this study proves to be a viable preferred global control strategy for deciding on when and how to ideally use a PHEV’s battery along a preplanned route. The TF control strategy achieves its objective of producing a more fuel-efficient drive cycle while remaining computationally inexpensive, robust, and easily implementable in real-time on board a vehicle.
6. References


[40] B. C. Glenn, "Intelligent Control of Parallel Hybrid Electric Vehicles," The Ohio State University, 1999.


7. Appendix

7.1. ADVISOR Results Window Figures

7.1.1. Route 1

Figure 27: Route 1 TF control strategy ADVISOR simulation results
Figure 28: Route 1 CS control strategy ADVISOR simulation results
Figure 29: Route 1 CD control strategy ADVISOR simulation results
7.1.2. Route 2

Figure 30: Route 2 TF control strategy ADVISOR simulation results
Figure 31: Route 2 CS control strategy ADVISOR simulation results
Figure 32: Route 2 CD control strategy ADVISOR simulation results
7.1.3. Route 3

Figure 33: Route 3 TF control strategy ADVISOR simulation results
Figure 34: Route 3 CS control strategy ADVISOR simulation results
Figure 35: Route 3 CD control strategy ADVISOR simulation results
7.1.4. Route 4

Figure 36: Route 4 TF control strategy ADVISOR simulation results
Figure 37: Route 4 CS control strategy ADVISOR simulation results
Figure 38: Route 4 CD control strategy ADVISOR simulation results
7.1.5. Route 5

Figure 39: Route 5 TF control strategy ADVISOR simulation results
Figure 40: Route 5 CS control strategy ADVISOR simulation results
Figure 41: Route 5 CD control strategy ADVISOR simulation results
7.1.6. Route 6

Figure 42: Route 6 TF control strategy ADVISOR simulation results
Figure 43: Route 6 CS control strategy ADVISOR simulation results
Figure 44: Route 6 CD control strategy ADVISOR simulation results
7.1.7. Route 7

Figure 45: Route 7 TF control strategy ADVISOR simulation results
Figure 46: Route 7 CS control strategy ADVISOR simulation results
Figure 47: Route 7 CD control strategy ADVISOR simulation results
7.1.8. Route 8

Figure 48: Route 8 TF control strategy ADVISOR simulation results
Figure 49: Route 8 CS control strategy ADVISOR simulation results
Figure 50: Route 8 CD control strategy ADVISOR simulation results

7.2. ADVISOR Code Additives

7.2.1. Trajectory Forecasting Setup Code

```matlab
%% Route Setup Version 5 %%
clear; clc;

%%Inputs
dist = input('Length of route in miles: ');

if dist < 11
    interval_dist = 0.05;
elseif (dist >= 11) && (dist < 26)
    interval_dist = 0.15;
else
    interval_dist = 0.25;
end
```
fprintf('\n')
while 1
    speed_lim_per = input('Percentages of route speed limit sections: ')/100;
    if (sum(speed_lim_per)-1) > 0.01
        disp('Percentages must add to 100%')
    else
        break
    end
end

fprintf('\n')
while 1
    speed_lim_cond = input('Corresponding speed limit conditions: ');
    if length(speed_lim_cond) ~= length(speed_lim_per)
        disp('Number of speed limit conditions must match number of speed limit sections')
    else
        break
    end
end

fprintf('\n')
while 1
    traffic_per = input('Percentages of route traffic sections: ')/100;
    if (sum(traffic_per)-1) > 0.01
        disp('Percentages must add to 100%')
    else
        break
    end
end

fprintf('\n')
while 1
    traffic_cond = input('Corresponding traffic conditions: ');
    if length(traffic_cond) ~= length(traffic_per)
        disp('Number of traffic conditions must match number of traffic sections')
    else
        break
    end
end

fprintf('\n')
while 1
    slope_per = input('Percentages of route slope sections: ')/100;
    if (sum(slope_per)-1) > 0.01
        disp('Percentages must add to 100%')
    else
        break
    end
end

fprintf('\n')
while 1
    slope_cond = input('Corresponding slope conditions: ');
    if length(slope_cond) ~= length(slope_per)
disp('Number of slope conditions must match number of slope sections')
else
    break
end
end

fprintf('
')
Slope_Option = input('Slope consideration, 1 for On or 2 for Off: ');

%%Route Data (miles)
if (rem(dist,interval_dist) ~= 0)
    segments = (dist-rem(dist,interval_dist))/interval_dist + 1;
else
    segments = round(dist/interval_dist);
end

route = zeros(1,(segments+1));
route(end) = dist;

for i = 1:(segments)
    route(i) = (i-1)*interval_dist;
end

%%Speed Limit Data (mph)
speed_lim = zeros(1,segments+1);
speed_lim_marker = 0;
Car_speed = zeros(1,segments+1);

for i = 1:length(speed_lim_per)
    for j = 1:length(speed_lim)
        if i == 1
            if route(j) <= round((dist*speed_lim_per(1)),2)
                speed_lim(j) = speed_lim_cond(1);
                    if j > 1
                        if speed_lim(j) == 1
                            Car_speed(j) = 15;
                        elseif speed_lim(j) == 2
                            Car_speed(j) = 25;
                        elseif speed_lim(j) == 3
                            Car_speed(j) = 35;
                        elseif speed_lim(j) == 4
                            Car_speed(j) = 45;
                        elseif speed_lim(j) == 5
                            Car_speed(j) = 55;
                        elseif speed_lim(j) == 6
                            Car_speed(j) = 65;
                        elseif speed_lim(j) == 7
                            Car_speed(j) = 75;
                        end
                    end
            else
                speed_lim(j) = speed_lim_per(1);
                    if j > 1
                        if speed_lim(j) == 1
                            Car_speed(j) = 15;
                        elseif speed_lim(j) == 2
                            Car_speed(j) = 25;
                        elseif speed_lim(j) == 3
                            Car_speed(j) = 35;
                        elseif speed_lim(j) == 4
                            Car_speed(j) = 45;
                        elseif speed_lim(j) == 5
                            Car_speed(j) = 55;
                        elseif speed_lim(j) == 6
                            Car_speed(j) = 65;
                        elseif speed_lim(j) == 7
                            Car_speed(j) = 75;
                        end
                    end
            end
        end
    end
end
else
if (route(j) > (speed_lim_marker)) && (route(j) <= round((speed_lim_marker + dist*speed_lim_per(i)),2))
    speed_lim(j) = speed_lim_cond(i);
    if speed_lim(j) == 1
        Car_speed(j) = 15;
    elseif speed_lim(j) == 2
        Car_speed(j) = 25;
    elseif speed_lim(j) == 3
        Car_speed(j) = 35;
    elseif speed_lim(j) == 4
        Car_speed(j) = 45;
    elseif speed_lim(j) == 5
        Car_speed(j) = 55;
    elseif speed_lim(j) == 6
        Car_speed(j) = 65;
    elseif speed_lim(j) == 7
        Car_speed(j) = 75;
    end
end

speed_lim_marker = speed_lim_marker + dist*speed_lim_per(i);
end

%%Traffic Data
traffic = zeros(1,segments+1);
traf_marker = 0;

for i = 1:length(traffic_per)
    for j = 1:length(traffic)
        if i == 1
            if route(j) <= round((dist*traffic_per(1)),2)
                traffic(j) = traffic_cond(1);
                if traffic(j) == 1
                    Car_speed(j) = Car_speed(j)*(randi([95,100],1,1)/100);
                elseif traffic(j) == 2
                    Car_speed(j) = Car_speed(j)*(randi([60,65],1,1)/100);
                else
                    Car_speed(j) = Car_speed(j)*(randi([20,25],1,1)/100);
                end
            end
        else
            if (route(j) > (traf_marker)) && (route(j) <= round((traf_marker + dist*traffic_per(i)),2))
                traffic(j) = traffic_cond(i);
                if traffic(j) == 1
                    Car_speed(j) = Car_speed(j)*(randi([95,100],1,1)/100);
                elseif traffic(j) == 2
                    Car_speed(j) = Car_speed(j)*(randi([60,65],1,1)/100);
                else
                    Car_speed(j) = Car_speed(j)*(randi([20,25],1,1)/100);
                end
            end
        end
        traf_marker = traf_marker + dist*traffic_per(i);
%%Slope Data
slope = zeros(1,segments+1);
slope_marker = 0;
road_slope = zeros(1,segments+1);
for i = 1:length(slope_per)
    for j = 1:length(slope)
        if i == 1
            if route(j) <= round((dist*slope_per(1)),2)
                slope(j) = slope_cond(1);
                if slope(j) == 1
                    road_slope(j) = -0.05;
                elseif slope(j) == 2
                    road_slope(j) = 0;
                else
                    road_slope(j) = 0.05;
                end
            end
        else
            if (route(j) > (slope_marker)) && (route(j) <= round((slope_marker + dist*slope_per(i)),2))
                slope(j) = slope_cond(i);
                if slope(j) == 1
                    road_slope(j) = -0.05;
                elseif slope(j) == 2
                    road_slope(j) = 0;
                else
                    road_slope(j) = 0.05;
                end
            end
        end
    end
    slope_marker = slope_marker + dist*slope_per(i);
end

%%Time Calculation
Time = zeros(1,segments+1);
for i = 2:length(Time)
    Time(i) = Time(i-1) + interval_dist/((1/3600)*((Car_speed(i)+Car_speed(i-1))/2));
end
cyc_mph = [Time' Car_speed'];
cyc_slope = [1609.34*route' road_slope'];
save('C:\Users\Joseph\Documents\advisor\data\drive_cycle\CYC_TrajFore_Fuz.mat', 'cyc_mph', 'cyc_slope', 'Slope_Option')
save('route.mat')

%%Plots
figure
plot(route,Car_speed)
title('Car Speed vs. Distance')
xlabel('Distance (miles)')
ylabel('Car Speed (mph)')

figure
plot(Time,Car_speed)
title('Car Speed vs. Time')
xlabel('Time (secs)')
ylabel('Car Speed (mph)')

figure
plot(route,road_slope)
title('Grade vs. Distance')
xlabel('Distance (miles)')
ylabel('Grade (mph)')

%% Path Identifier Version 5 %%
clear; clc;

fprintf('
')
Bat_Type = input('1 if Pb battery, 2 if Li battery: '); fprintf('
')
Ah_Cap = input('Battery Capacity (ah): '); fprintf('
')
Max_SOC = input('Max SOC: '); fprintf('
')
Initial_SOC = input('Initial SOC: '); fprintf('
')
Lowest_SOC = input('Minimum SOC: '); fprintf('
')
SOC_Coeff = input('SOC Coefficient Adjustment: '); fprintf('
')
Drive_Type = input('1 for EV/ICE capability, 2 EV/Blended/ICE capability: '); fprintf('
')
load('route.mat')

%%%Assigning Section Priority
Ant_Spd_Matrix = [15 9.75 3.75; % used numbers in routeV4 code that give the
carspeed profile
25 16.25 6.25 % 1 0.65 0.25
35 22.75 8.75 %15 - ---- ----
45 29.25 11.25 %25 - ---- ----
55 35.75 13.75 %35 - ---- ----
65 42.25 16.25 %45 - ---- ----
75 48.75 18.75];%55 - ---- ----
 %65 - ---- ----
 %75 - ---- ----

Priority_Matrix = [3 3 3; % 1- least priority (ICE); 2- medium priority
(Blended); 3- highest priority (EV)
3 3 3; % v <= 25 is EV Mode, 25 < v < 55 Blended Mode, v
>= 55 ICE Mode
2 3 3;
2 2 3;
1 2 3;
1 2 3;
1 2 3];

Matrix = [route; traffic; speed_lim; slope; zeros(size(route));
zeros(size(route))];

for i = 1:length(route)
    if Matrix(4,i) == 3

83
Matrix(5,i) = 1; % high road grade means this should be of least priority
else
    Matrix(5,i) = Priority_Matrix(Matrix(3,i),Matrix(2,i));
end

% Calculating Section Size and Distance
Section_Size = 0;

j = 0;
for i = 2:length(Matrix(5,:))
    if Matrix(5,i) == Matrix(5,i-1)
        j = j + 1;
    else
        Section_Size = [Section_Size, j]; %#ok<AGROW>
        j = 1;
    end
end
Section_Size = [Section_Size(2:end), (i-sum(Section_Size)-1)];
Section_Dist = interval_dist*Section_Size;

% SOC Needed per Section
ampHrs_per_Segment = zeros(size(route));
for i = 2:length(ampHrs_per_Segment)
    if Matrix(5,i) == 3 % SOC equation for priority 3
        if Matrix(4,i) == 3
            ampHrs_per_Segment(i) = interval_dist*2*0.87*(0.0198*(Ant_Spd_Matrix(Matrix(3,i),Matrix(2,i))) + 0.1689); % amount required double for uphill
        else
            ampHrs_per_Segment(i) = interval_dist*0.87*(0.0198*(Ant_Spd_Matrix(Matrix(3,i),Matrix(2,i))) + 0.1689);
        end
    elseif Matrix(5,i) == 2 % SOC equation for priority 2
        if Matrix(4,i) == 3
            ampHrs_per_Segment(i) = interval_dist*2*0.42*(0.0198*(Ant_Spd_Matrix(Matrix(3,i),Matrix(2,i))) + 0.1689); % amount required double for uphill
        else
            ampHrs_per_Segment(i) = interval_dist*0.42*(0.0198*(Ant_Spd_Matrix(Matrix(3,i),Matrix(2,i))) + 0.1689);
        end
    else
        if Matrix(4,i) == 3 % SOC equation for priority 1
            ampHrs_per_Segment(i) = interval_dist*2*0.34*(0.0198*(Ant_Spd_Matrix(Matrix(3,i),Matrix(2,i))) + 0.1689); % amount required double for uphill
        else
            ampHrs_per_Segment(i) = interval_dist*0.34*(0.0198*(Ant_Spd_Matrix(Matrix(3,i),Matrix(2,i))) + 0.1689);
        end
    end
ampHrs_per_Segment = ampHrs_per_Segment(2:end);

Section_AmpHrs = zeros(size(Section_Size));

for i = 1:length(Section_AmpHrs)
    if i == 1
        Section_AmpHrs(i) = sum(ampHrs_per_Segment(1:Section_Size(i)));
    else
        Section_AmpHrs(i) = sum(ampHrs_per_Segment((1+sum(Section_Size(1:(i-1)))):sum(Section_Size(1:i))));
    end
end

if Bat_Type == 1
    Section_SOC = Section_AmpHrs/(Ah_Cap*0.55);
else
    Section_SOC = Section_AmpHrs/Ah_Cap;
end

%%Sorting Sections by Priority and Length
Section_Priority = size(Section_Size);

for i = 1:length(Section_Size)
    Section_Priority(i) = Matrix(5,(1+sum(Section_Size(1:i))));
end

Section_Sort = [Section_Priority; Section_Dist; Section_SOC]';
[Section_Sort,index] = sortrows(Section_Sort,'descend');

%%Setting Section SOC_min
Section_SOC_Mins = zeros(1,length(index));
if Initial_SOC == Max_SOC
    Initial_SOC = Initial_SOC - 0.005;
end

allowed_SOC = Initial_SOC - Lowest_SOC;

%% Determine settings for determining sections' SOC_Min
if Drive_Type == 1
    SOC_setting = 1;
else
    if allowed_SOC >= (round(sum(Section_Sort(:,3)),3) - 0.125) %Enough for whole route in all electric mode; launch speed that includes all section speeds
        SOC_setting = 1;
    else
        k_EV = length(find(Section_Sort(:,1) == 3));
        k_B1 = length(find(Section_Sort(:,1) == 2));
        if (allowed_SOC < (round(sum(Section_Sort(:,3)),3)) - 0.125) && (allowed_SOC >= round(sum(Section_Sort(1:(k_EV+k_B1),3)),3)) %Enough for EV
            SOC_setting = 1;
        else
            SOC_setting = 1;
        end
    end
end

end
end

%%%%%%%%%%%%%%%%%%%%%%
ampHrs_per_Segment = ampHrs_per_Segment(2:end);

Section_AmpHrs = zeros(size(Section_Size));

for i = 1:length(Section_AmpHrs)
    if i == 1
        Section_AmpHrs(i) = sum(ampHrs_per_Segment(1:Section_Size(i)));
    else
        Section_AmpHrs(i) = sum(ampHrs_per_Segment((1+sum(Section_Size(1:(i-1)))):sum(Section_Size(1:i))));
    end
end

if Bat_Type == 1
    Section_SOC = Section_AmpHrs/(Ah_Cap*0.55);
else
    Section_SOC = Section_AmpHrs/Ah_Cap;
end

%%Sorting Sections by Priority and Length
Section_Priority = size(Section_Size);

for i = 1:length(Section_Size)
    Section_Priority(i) = Matrix(5,(1+sum(Section_Size(1:i))));
end

Section_Sort = [Section_Priority; Section_Dist; Section_SOC]';
[Section_Sort,index] = sortrows(Section_Sort,'descend');

%%Setting Section SOC_min
Section_SOC_Mins = zeros(1,length(index));
if Initial_SOC == Max_SOC
    Initial_SOC = Initial_SOC - 0.005;
end

allowed_SOC = Initial_SOC - Lowest_SOC;

%% Determine settings for determining sections' SOC_Min
if Drive_Type == 1
    SOC_setting = 1;
else
    if allowed_SOC >= (round(sum(Section_Sort(:,3)),3) - 0.125) %Enough for whole route in all electric mode; launch speed that includes all section speeds
        SOC_setting = 1;
    else
        k_EV = length(find(Section_Sort(:,1) == 3));
        k_B1 = length(find(Section_Sort(:,1) == 2));
        if (allowed_SOC < (round(sum(Section_Sort(:,3)),3)) - 0.125) && (allowed_SOC >= round(sum(Section_Sort(1:(k_EV+k_B1),3)),3)) %Enough for EV
            SOC_setting = 1;
        else
            SOC_setting = 1;
        end
    end
end

85
and Blended suggested sections in all electric mode; launch speed that includes EV and Blended section speeds

SOC_setting = 2;
elseif
(allowed_SOC < round(sum(Section_Sort(1:(k_EV+k_Bl),3)),3)) &&
(allowed_SOC >= round(sum(Section_Sort(1:k_EV,3)),3)) %Enough for just EV
suggested sections in all electric mode; launch speed that includes EV section
speeds
SOC_setting = 3;
elseif
allowed_SOC < round(sum(Section_Sort(1:k_EV,3)),3) %SOC will
run out in the EV suggested sections; launch speed that includes EV section
speeds
SOC_setting = 4;
end
end

%%% Different SOC settings for choosing SOC_Min values
if SOC_setting == 1
flip_index = fliplr(index');
for i = 1:(length(Section_SOC_Mins) - 1)
if
(allowed_SOC - sum(Section_SOC(index(1:(end-i))))) <= 0
Section_SOC_Mins(flip_index(i)) = 0;
else
Section_SOC_Mins(flip_index(i)) = allowed_SOC -
sum(Section_SOC(index(1:(end-i))));
allowed_SOC = sum(Section_SOC(index(1:(end-i))));
end
end
Section_SOC_Mins(index(1)) = allowed_SOC;
Allowed_SOC = Section_SOC_Mins;
Section_SOC_Mins = Lowest_SOC*ones(size(Section_SOC_Mins));
elseif SOC_setting == 2
flip_index = fliplr(index');
for i = 1:(length(Section_SOC_Mins) - 1)
if
(allowed_SOC - sum(Section_SOC(index(1:(end-i))))) <= 0
Section_SOC_Mins(flip_index(i)) = 0;
else
Section_SOC_Mins(flip_index(i)) = allowed_SOC -
sum(Section_SOC(index(1:(end-i))));
allowed_SOC = sum(Section_SOC(index(1:(end-i))));
end
end
Section_SOC_Mins(index(1)) = allowed_SOC;
residual = sum(Section_SOC_Mins(sort(index((k_EV+k_Bl+1):end))));
blend_ratios =
Section_Dist(sort(index((k_EV+k_Bl+1):end)))/sum(Section_Dist(sort(index((k_EV+
_k_Bl+1):end))));
Section_SOC_Mins(sort(index((k_EV+k_Bl+1):end))) = residual*blend_ratios;
Allowed_SOC = Section_SOC_Mins;
for i = 1:length(Section_SOC_Mins)
    if i == 1
        Section_SOC_Mins(i) = Initial_SOC - Section_SOC_Mins(i);
    else
        Section_SOC_Mins(i) = Section_SOC_Mins(i-1) - Section_SOC_Mins(i);
    end
end

blended_Section_SOC_Mins = Section_SOC_Mins(sort(index((k_EV+k_Bl+1):end)));
blended_Section_distances = Section_Dist(sort(index((k_EV+k_Bl+1):end)));

elseif SOC_setting == 3
    flip_index = fliplr(index');
    for i = 1:(length(Section_SOC_Mins) - 1)
        if (allowed_SOC - sum(Section_SOC(index(1:(end-i))))) <= 0
            Section_SOC_Mins(flip_index(i)) = 0;
        else
            Section_SOC_Mins(flip_index(i)) = allowed_SOC - sum(Section_SOC(index(1:(end-i))));
            allowed_SOC = sum(Section_SOC(index(1:(end-i))));
        end
    end
    Section_SOC_Mins(index(1)) = allowed_SOC;
residual = sum(Section_SOC_Mins(sort(index((k_EV+1):(k_EV+k_Bl))));
blend_ratios = Section_Dist(sort(index((k_EV+1):(k_EV+k_Bl))))/sum(Section_Dist(sort(index((k_EV+1):(k_EV+k_Bl)))))
    Section_SOC_Mins(sort(index((k_EV+1):(k_EV+k_Bl)))) = residual*blend_ratios;
    Allowed_SOC = Section_SOC_Mins;
    for i = 1:length(Section_SOC_Mins)
        if i == 1
            Section_SOC_Mins(i) = Initial_SOC - Section_SOC_Mins(i);
        else
            Section_SOC_Mins(i) = Section_SOC_Mins(i-1) - Section_SOC_Mins(i);
        end
    end

    blended_Section_SOC_Mins = Section_SOC_Mins(sort(index((k_EV+1):(k_EV+k_Bl))));
    blended_Section_distances = Section_Dist(sort(index((k_EV+1):(k_EV+k_Bl))));

elseif SOC_setting == 4
    flip_index = fliplr(index');
    for i = 1:(length(Section_SOC_Mins) - 1)
        if (allowed_SOC - sum(Section_SOC(index(1:(end-i))))) <= 0
            Section_SOC_Mins(flip_index(i)) = 0;
        else
            Section_SOC_Mins(flip_index(i)) = allowed_SOC - sum(Section_SOC(index(1:(end-i))));
            allowed_SOC = sum(Section_SOC(index(1:(end-i))));
        end
    end
    Section_SOC_Mins(index(1)) = allowed_SOC;
residual = sum(Section_SOC_Mins(sort(index((k_EV+1):(k_EV+k_Bl))));
blend_ratios = Section_Dist(sort(index((k_EV+1):(k_EV+k_Bl))))/sum(Section_Dist(sort(index((k_EV+1):(k_EV+k_Bl)))))
    Section_SOC_Mins(sort(index((k_EV+1):(k_EV+k_Bl)))) = residual*blend_ratios;
    Allowed_SOC = Section_SOC_Mins;
    for i = 1:length(Section_SOC_Mins)
        if i == 1
            Section_SOC_Mins(i) = Initial_SOC - Section_SOC_Mins(i);
        else
            Section_SOC_Mins(i) = Section_SOC_Mins(i-1) - Section_SOC_Mins(i);
        end
    end

    blended_Section_SOC_Mins = Section_SOC_Mins(sort(index((k_EV+1):(k_EV+k_Bl))));
    blended_Section_distances = Section_Dist(sort(index((k_EV+1):(k_EV+k_Bl))));
Section_SOC_Mins(flip_index(i)) = 0;
else
    Section_SOC_Mins(flip_index(i)) = allowed_SOC -
    sum(Section_SOC(index(1:(end-i))));
    allowed_SOC = sum(Section_SOC(index(1:(end-i))));
end
end

Section_SOC_Mins(index(1)) = allowed_SOC;
Allowed_SOC = Section_SOC_Mins;

for i = 1:length(Section_SOC_Mins)
    if i == 1
        Section_SOC_Mins(i) = Initial_SOC - Section_SOC_Mins(i);
    else
        Section_SOC_Mins(i) = Section_SOC_Mins(i-1) -
        Section_SOC_Mins(i);
    end
end

%%SOC Throughout Route
SOC_Mins = zeros(1,(length(route)-1));
j = 1 + length(Section_SOC_Mins);

for i = 1:length(Section_SOC_Mins)
    if i == 1
        SOC_Mins(1:Section_Size(i)) =
        Section_SOC_Mins(i)*ones(1,Section_Size(i));
    else
        SOC_Mins((1+sum(Section_Size(1:(i-1)))):sum(Section_Size(1:i))) =
        Section_SOC_Mins(i)*ones(1,Section_Size(i));
    end
end

SOC_Mins = [SOC_Mins(1) SOC_Mins];
Matrix(6,:) = SOC_Mins;
Section_SOC_Min_Dist = [Section_SOC_Mins; Section_Dist];
if SOC_setting == 1 || SOC_setting == 4
    blended_SOC_and_dist = [0;0];
else
    blended_SOC_and_dist = [blended_Section_SOC_Mins;
    blended_Section_distances];
end

save('C:\Users\Joseph\Documents\advisor\models\Traj_Fore_Fuzzy_SOC_Variables.\mat','Section_SOC_Min_Dist','Matrix','SOC_setting','blended_SOC_and_dist')
save('Traj_Fore_Fuzzy_SOC_Variables.mat')

disp(Section_Dist)
disp(Section_Priority)
disp(Section_SOC)
disp(SOC_setting)
disp(Allowed_SOC)
disp(Section_SOC_Mins)
disp(blended_SOC_and_dist)

%% Fuzzy Block Controller Setup Version 5 %%
if Drive_Type == 2
    clear; clc

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
%%Meant for Two Inputs and One Output with same # of membership functions
for all%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Number of Membership Functions
fprintf('n')
while 1
    tri = input('Choose 5 or 7 membership functions for variables: ');
    if ((tri == 5) || (tri == 7))
        break
    else
        disp('Choose either 5 or 7. ')
    end
end
if (tri == 5)
    rule = [5 5 5 4 3;
         5 5 4 3 2;
         5 4 3 2 1;
         4 3 2 1 1;
         3 2 1 1 1];
elseif (tri == 7)
    rule = [7 7 7 7 6 5 4;
         7 7 7 6 5 4 3;
         7 7 6 5 4 3 2;
         7 6 5 4 3 2 1;
         6 5 4 3 2 1 1;
         5 4 3 2 1 1 1;
         4 3 2 1 1 1 1];
end

%%Error Input Triangles
fprintf('n')
sat1 = input('Upper/Lower point of saturation for error range: ');
b1 = (2*sat1)/(((tri+1)/2)-1);
limit1 = sat1 + (b1/2);
x1 = zeros(2*tri,10);
y1 = zeros(2*tri,10);
for i = 1:(tri+1)
\( x_1(i,:) = \text{linspace}((-\text{limit}1 + ((i-1) \ast (b_1/2))), (-\text{limit}1 + (i \ast (b_1/2))), 10); \)

\( x_1((\text{tri}+2):2*\text{tri}, :) = x_1(2:\text{tri}, :); \)

for \( i = 1:(2*\text{tri}) \)
  if \( ((i == 1) || (i == (\text{tri}+1))) \)
    \( y_1(i,:) = \text{ones}(1, 10); \)
  else
    if \( (\text{rem}(i, 2) \neq 0) \)
      \( y_1(i,:) = (2/b_1) \ast (x_1(i,:) - x_1(i,1)); \)
    else
      \( y_1(i,:) = -(2/b_1) \ast (x_1(i,:) - x_1(i,1)) + 1; \)
  end
end

%%d(Error)/dt Triangles Variables
\text{fprintf}('n')
\text{sat}2 = \text{input}('Upper/Lower point of saturation for change in error range: ') ;
\b2 = (2*\text{sat}2)/(((\text{tri}+1)/2)-1);
\text{limit}2 = \text{sat}2 + (\b2/2);

\( x_2 = \text{zeros}(2*\text{tri}, 10); \)
\( y_2 = \text{zeros}(2*\text{tri}, 10); \)

for \( i = 1:(\text{tri}+1) \)
  \( x_2(i,:) = \text{linspace}((-\text{limit}2 + ((i-1) \ast (b_2/2))), (-\text{limit}2 + (i \ast (b_2/2))), 10); \)
end

\( x_2((\text{tri}+2):2*\text{tri}, :) = x_2(2:\text{tri}, :); \)

for \( i = 1:(2*\text{tri}) \)
  if \( ((i == 1) || (i == (\text{tri}+1))) \)
    \( y_2(i,:) = \text{ones}(1, 10); \)
  else
    if \( (\text{rem}(i, 2) \neq 0) \)
      \( y_2(i,:) = (2/b_2) \ast (x_2(i,:) - x_2(i,1)); \)
    else
      \( y_2(i,:) = -(2/b_2) \ast (x_2(i,:) - x_2(i,1)) + 1; \)
  end
end

%%Output Triangles Variables
\text{sat}3 = 0.4; % hard set so x-axis allows [-0.4 0.4] for possible K output
\b3 = (2*\text{sat}3)/(((\text{tri}+1)/2)-1);
\text{limit}3 = \text{sat}3 + (\b3/2); %input('Upper/Lower limit for output range: ')
\( x_3 = \text{zeros}(2*\text{tri}, 10); \)
\( y_3 = \text{zeros}(2*\text{tri}, 10); \)
for i = 1:(tri+1)
    x3(i,:) = linspace((-limit3+((i-1)*(b3/2))),(-limit3+(i*(b3/2))),10);
end

x3((tri+2):(2*tri),:) = x3(2:tri,:);

x3 = x3 + sat3; %shift possible K output to [0 0.8]

for i = 1:(2*tri)
    if (rem(i,2) ~= 0)
        y3(i,:) = (2/b3)*(x3(i,:) - x3(i,1));
    else
        y3(i,:) = -(2/b3)*(x3(i,:) - x3(i,1)) + 1;
    end
end

out_midpoint = zeros(1,tri);

for i = 1:tri
    out_midpoint(i) = -limit3 + i*(b3/2);
end

out_midpoint = out_midpoint + sat3; %shift possible K output to [0 0.8]

fprintf('
')
while 1
    Prem_Method = input('Choose either Minimum or Product for premise certainty: ','s');
    if (Prem_Method == 'Minimum')
        Prem_Method = 1;
        break
    elseif (Prem_Method == 'Product')
        Prem_Method = 2;
        break
    else
        disp('Must choose Minimum or Product.')
    end
end
fprintf('
')
while 1
    Defuzz_Method = input('Choose either COG or C_A defuzzification method: ','s');
    if (Defuzz_Method == 'COG')
        Defuzz_Method = 1;
        break
    elseif (Defuzz_Method == 'C_A')
        Defuzz_Method = 2;
        break
    else
        disp('Must choose COG or C_A.')
    end
end
save('C:\Users\Joseph\Documents\advisor\models\Traj_Fore_Fuzzy_Calc_Variables.mat')
save('Traj_Fore_Fuzzy_Calc_Variables.mat')

%%Plots
figure
subplot(3,1,1)
for m = 1:1:(2*tri)
    plot(x1(m,:),y1(m,:),'b')
    hold on
end
grid on
ylabel('Certainty')
xlabel('e(t)')
ylim([0 1])

subplot(3,1,2)
for m = 1:1:(2*tri)
    plot(x2(m,:),y2(m,:),'b')
    hold on
end
grid on
ylabel('Certainty')
xlabel('delta e(t)')
ylim([0 1])

subplot(3,1,3)
for m = 1:1:(2*tri)
    plot(x3(m,:),y3(m,:),'b')
    hold on
end
grid on
ylabel('Certainty')
xlabel('u(t)')
ylim([0 1])
clear;
end
clear;
advisor;

7.2.2. Parallel TF Vehicle Control Block: Engine Shutoff for EV Only Routes Function

Code

function y = fcn(u, SOC)
FSC = load('Traj_Fore_Fuzzy_SOC_Variables.mat');
SOC_setting = FSC.SOC_setting;

if (SOC_setting == 1 && SOC >= 0.25)
7.2.3. Fuzzy TF Control Block: SOC Min Assignment Function Code

```matlab
def function [y,soc_min] = fcn(u,distance)
FSC = load('Traj_Fore_Fuzzy_SOC_Variables.mat');
Matrix = FSC.Matrix;

distance = distance*0.000621371;
if distance == 0
    soc_min = Matrix(6,1);
else
    a = find(Matrix(1,:) > distance);
    if sum(a) == 0
        soc_min = Matrix(6,end);
    else
        soc_min = Matrix(6,(a(1)-1));
    end
end
y = u;
```

7.2.4. Fuzzy TF Control Block: Control Strategy Function Code

```matlab
def function
[SOC_init_cur,Dist_init_cur,K,avail_trq,element_cur,priority_cur,desired,error]
= fcn(SOC_init_prev,Dist_init_prev,req_trq,distance,element_prev,SOC,priority_prev,
SOC_goal,error_prev)

%% Initialize Variables
a = 0;
element_cur = 0;
desired = 0;
K = 0;
Switch = 0;
error = 0;
SOC_init_cur = 0;
Dist_init_cur = 0;

%% Fuzzy SOC Variables C:\Users\Joseph\Documents\advisor\models
FSC = load('Traj_Fore_Fuzzy_SOC_Variables.mat');
SOC_setting = FSC.SOC_setting;
Matrix = FSC.Matrix;
blended_SOC_and_dist = FSC.blended_SOC_and_dist;

%% Fuzzy Calculation Variables C:\Users\Joseph\Documents\advisor\models
FV = load('Traj_Fore_Fuzzy_Calc_Variables.mat');
b1 = FV.b1;
```
sat1 = FV.sat1;
x1 = FV.x1;
y1 = FV.y1;
b2 = FV.b2;
sat2 = FV.sat2;
x2 = FV.x2;
y2 = FV.y2;
b3 = FV.b3;
% x3 = FV.x3;
% y3 = FV.y3;
tri = FV.tri;
Prem_Method = FV.Prem_Method;
rule = FV.rule;
Defuzz_Method = FV.Defuzz_Method;
out_midpoint = FV.out_midpoint;

%% Determine Priority Value
distance = distance*0.000621371;

if distance == 0
   priority = Matrix(5,1);
   SOC_Min = Matrix(6,1);
else
   a = find(Matrix(1,:) > distance);
   if sum(a) == 0
      priority = Matrix(5,end);
      SOC_Min = Matrix(6,end);
   else
      priority = Matrix(5,(a(1)-1));
      SOC_Min = Matrix(6,(a(1)-1));
   end
end

%% Determine Which Blended Section and SOC goal
if SOC_setting == 1
   element_cur = 0;
   % z = 0;
   K = 0; %Consider using K = 1
   % Switch = 0;
elseif SOC_setting == 2
   if priority == 1
      if distance == 0 && element_prev == 0
         element_cur = element_prev + 1;
         desired = (SOC - blended_SOC_and_dist(1,element_cur))/blended_SOC_and_dist(2,element_cur);
         SOC_init_cur = SOC;
         Dist_init_cur = distance;
      elseif distance == 0 && priority_prev == 1
         element_cur = element_prev + 1;
         desired = (SOC - blended_SOC_and_dist(1,element_cur))/blended_SOC_and_dist(2,element_cur);
         SOC_init_cur = SOC;
         Dist_init_cur = distance;
      elseif distance ~= 0 && (a(1)-2) > 0

94
% if priority_prev ~= 1 \%Matrix(5,(a(1)-2)) ~= 1
%     w = element + 1;
%     z = (SOC -  
blended_SOC_and_dist(1,w))/blended_SOC_and_dist(2,w);
%     end
else
    element_cur = element_prev;
    desired = SOC_goal;
    SOC_init_cur = SOC_init_prev;
    Dist_init_cur = Dist_init_prev;
end

Switch = 1;
else
    element_cur = element_prev;
    % z = 0;
    K = 0; %Consider using K = 1
    %
end
else
    Switch = 0;
end
elseif SOC_setting == 3
    if priority == 1
        element_cur = element_prev;
        % z = 0;
        if ((SOC - SOC_Min) >= 0.01)
            K = 0.85;
        elseif ((SOC - SOC_Min) >= 0.005)
            K = 0.9;
        else
            K = 1;
        end

        % Switch = 0;
else
    priority == 2
    if distance == 0 && element_prev == 0
        element_cur = element_prev + 1;
        desired = (SOC -  
blended_SOC_and_dist(1,element_cur))/blended_SOC_and_dist(2,element_cur);
        SOC_init_cur = SOC;
        Dist_init_cur = distance;
    elseif distance ~= 0 && priority_prev ~= 2
        element_cur = element_prev + 1;
        desired = (SOC -  
blended_SOC_and_dist(1,element_cur))/blended_SOC_and_dist(2,element_cur);
        SOC_init_cur = SOC;
        Dist_init_cur = distance;
    elseif distance ~= 0 && (a(1)-2) > 0
    elseif priority_prev ~= 2 \%Matrix(5,(a(1)-2)) ~= 2
        w = element + 1;
        % z = (SOC -  
blended_SOC_and_dist(1,w))/blended_SOC_and_dist(2,w);
    % end
else
    element_cur = element_prev;
    desired = SOC_goal;
    SOC_init_cur = SOC_init_prev;
Dist_init_cur = Dist_init_prev;
end

Switch = 1;
else
    element_cur = element_prev;
    %
    z = 0;
    K = 0; %Consider using K = 1
    %
    Switch = 0;
end
elseif SOC_setting == 4 %possibly combine with SOC_setting 3 with if
statement for EV section K depending on setting 3 or 4
    element_cur = 0;
    %
    z = 0;
    %
    if priority == 1 || priority == 2
        if ((SOC - SOC_Min) >= 0.01)
            K = 0.85;
        elseif ((SOC - SOC_Min) >= 0.005)
            K = 0.9;
        else
            K = 1;
        end
    else
        K = 0;
    end
end
if (SOC <= SOC_Min)
    K = 1;
    Switch = 0;
end

%%% Determine Available Torque K Coefficient
if Switch == 1
    if distance == 0 || distance == Dist_init_cur
        error = 0;
        del_error = error;
    else
        error = (desired - (SOC_init_cur-SOC)/(distance-
Dist_init_cur))/desired;
        del_error = error - error_prev;
    end

%%%%%%%%%%%%%%%%FUZZY LOGIC%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%%%Meant for Two Inputs and One Output with same # of membership functions for all%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%% Controller (Finding Input Certainties and Triangle Numbers)
i = [0 0]; % Error triangle numbers, i(1) for odd and i(2) for even
j = [0 0]; % Change-in-Error triangle numbers, j(1) for odd and j(2) for even

mf_cer1 = zeros(1,2); % Error triangle certainties, mf_cer1(1) for odd and mf_cer1(2) for even
mf_cer2 = zeros(1,2); % Change-in-Error triangle certainties, mf_cer2(1) for odd and mf_cer2(2) for even

for k1 = 1:(2*tri) % Checks saturation and greater; checks for when certainties are less than 1
    if (error <= -sat1)
        mf_cer1(1) = 1;
i(1) = 1;
    elseif (error >= sat1)
        mf_cer1(1) = 1;
i(1) = tri;
    elseif ((error > x1(k1,1)) && (error < x1(k1,end)) && (abs(error) < sat1))
        if (rem(k1,2) ~= 0)
            if (mf_cer1(1) == 0)
                mf_cer1(1) = (2/b1)*(error - x1(k1,1));
                if (k1 <= (tri + 1))
                    i(1) = k1;
                elseif (k1 > (tri + 1))
                    i(1) = k1 - tri;
                end
            else
                mf_cer1(2) = (2/b1)*(error - x1(k1,1));
                if (k1 <= (tri + 1))
                    i(2) = k1;
                elseif (k1 > (tri + 1))
                    i(2) = k1 - tri;
                end
            end
        else
            if (mf_cer1(1) == 0)
                mf_cer1(1) = -(2/b1)*(error - x1(k1,1)) + 1;
                if (k1 <= (tri + 1))
                    i(1) = k1 - 1;
                elseif (k1 > (tri + 1))
                    i(1) = k1 - tri - 1;
                end
            else
                mf_cer1(2) = -(2/b1)*(error - x1(k1,1)) + 1;
                if (k1 <= (tri + 1))
                    i(2) = k1 - 1;
                elseif (k1 > (tri + 1))
                    i(2) = k1 - tri - 1;
                end
            end
        end
    end
end
if ((rem(error, (b1/2)) == 0) && (abs(error) < sat1))%Checks when certainties are one
mf_cer1(1) = 1;
for k1 = 1:(2*tri)
  if ((error == x1(k1,1)) && (1 == y1(k1,1))
    if (k1 <= (tri + 1))
      i(1) = k1 - 1;
    elseif (k1 > (tri + 1))
      i(1) = k1 - tri - 1;
    end
  end
end
end

for k1 = 1:(2*tri) %Checks saturation and greater; checks for when certainties are less than 1
if (del_error <= -sat2)
  mf_cer2(1) = 1;
  j(1) = 1;
elseif (del_error >= sat2)
  mf_cer2(1) = 1;
  j(1) = tri;
elseif ((del_error > x2(k1,1)) && (del_error < x2(k1,end)) && (abs(del_error) < sat2))
  if (rem(k1,2) == 0)
    if (mf_cer2(1) == 0)
      mf_cer2(1) = (2/b2)*(del_error - x2(k1,1));
      if (k1 <= (tri + 1))
        j(1) = k1;
      elseif (k1 > (tri + 1))
        j(1) = k1 - tri;
      end
    else
      mf_cer2(1) = -((2/b2)*(del_error - x2(k1,1))) + 1;
      if (k1 <= (tri + 1))
        j(1) = k1 - 1;
      elseif (k1 > (tri + 1))
        j(1) = k1 - tri - 1;
      end
    end
    end
  else
    if (mf_cer2(1) == 0)
      mf_cer2(1) = (2/b2)*(del_error - x2(k1,1));
      if (k1 <= (tri + 1))
        j(1) = k1;
      elseif (k1 > (tri + 1))
        j(1) = k1 - tri;
      end
    end
  end
else
  if (mf_cer2(1) == 0)
    mf_cer2(1) = -((2/b2)*(del_error - x2(k1,1))) + 1;
    if (k1 <= (tri + 1))
      j(1) = k1 - 1;
    elseif (k1 > (tri + 1))
      j(1) = k1 - tri - 1;
    end
  else
    mf_cer2(2) = -((2/b2)*(del_error - x2(k1,1))) + 1;
    if (k1 <= (tri + 1))
      j(2) = k1 - 1;
    elseif (k1 > (tri + 1))
      j(2) = k1 - tri - 1;
    end
  end
end
end
if ((rem(del_error, (b2/2)) == 0) && (abs(del_error) < sat2)) % Checks when certainties are one
  mf_cer2(1) = 1;
  for k1 = 1:(2*tri)
    if ((del_error == x2(k1,1)) && (1 == y2(k1,1))
      if (k1 <= (tri + 1))
        j(1) = k1 - 1;
      elseif (k1 > (tri + 1))
        j(1) = (k1 - (tri + 1));
      end
    end
  end
end

%%%% Controller (Determining Premise and Conclusion Certainty)
prem = zeros(2,2);
for k1 = [1 2]
  for k2 = [1 2]
    if (Prem_Method == 1)
      prem(k1,k2) = min(mf_cer1(k1),mf_cer2(k2));
    elseif (Prem_Method == 2)
      prem(k1,k2) = (mf_cer1(k1)*mf_cer2(k2));
    end
  end
end
leng = length(find(prem~=0));
u = zeros(leng,2);
k = 1;
for k1 = [1 2]
  for k2 = [1 2]
    if prem(k1,k2) ~= 0
      u(k,:) = [prem(k1,k2) rule(i(k1),j(k2))];
      k = k + 1;
    end
  end
end
Denom = zeros(1,leng);
Num = zeros(1,leng);
if (Defuzz_Method == 1)
  if (Prem_Method == 1)
    Denom = b3.*(u(:,1)' - 0.5.*(u(:,1)').^2));
  elseif (Prem_Method == 2)
    Denom = 0.5.*b3.*u(:,1)';
  end
  Num = out_midpoint(u(:,2)).*Denom;
\begin{verbatim}
elseif (Defuzz_Method == 2)
    Denom = u(:,1)';
    Num = out_midpoint(u(:,2)).*Denom;
end
    K = sum(Num)/sum(Denom);

if K < 0
    K = 0;
end

avail_trq = K*req_trq;
priority_cur = priority;
\end{verbatim}