ABSTRACT

WANG, RUI. A Real-time, Robust Control Strategy for Plug-in Hybrid Electric Vehicles. (Under the direction of Srdjan M. Lukic).

Plug-in hybrid electric vehicles (PHEVs) are hybrid electric vehicles (HEVs) that can be recharged by connecting to the electric grid. PHEVs can operate in all-electric mode or in a blended mode where the battery and the internal combustion engine work simultaneously to propel the vehicle. The state-of-the-art PHEV control uses the charge depleting (CD) - charge sustaining (CS) control strategy, which forces the battery energy to be utilized in priority. However this control approach does not guarantee optimal performance.

Dynamic programming (DP), an offline optimization tool, guarantees global optimal solution to a constrained cost function. Equivalent consumption minimization strategy (ECMS) is a real-time control strategy, which minimizes an equivalent cost function in form of $J = \dot{m}_{fuel} + \lambda * S\dot{O}C$ where $\dot{m}_{fuel}$ is fuel consumption rate, λ is equivalent factor and $S\dot{O}C$ is battery state of charge (SOC) variation rate. For PHEVs λ is sensitive to trip length since the energy stored in the battery should be used more aggressively on shorter trips, and vice versa.

In this thesis, we propose a novel control method for PHEVs that combines DP and ECMS to find the optimal λ value given remaining distance d_R and current SOC value. Using DP results, λ is determined; ECMS then calculates the engine and motor operating points based on its cost function. DP results are used as follows: given an optimal trajectory for a known drive cycle and the change in SOC when covering the remaining distance d_R, λ is obtained at that instance by linear regression of the DP results. Therefore, a look-up table of λ (referred as λ map) can be generated with respect to d_R and SOC for a given drive cycle.

With a λ map available, and trip length and target SOC known, ECMS can be implemented. Results show that the equivalent cost has an average of 5% gap to DP results over twelve standard drive cycles. This benchmarking exercise showed us that given a known driving pattern, the estimate of λ was near-optimal.

Next, we generated a universal map by merging the twelve λ maps into one. We find an increased 14% gap to DP results on average over the twelve drive cycles, but a 30% improvement over CD-CS strategy. For further verification, we simulate 4719 real-world drive cycles, proving controller robustness with an average 9.3% cost reduction compared to CD-CS strategy.

To further improve the performance of the proposed controller, we classify the λ maps into three patterns (city, suburban, and highway), and introduce an offset in the instantaneous value of λ based on the driving style. Fuzzy logic controllers are designed for driving pattern and driving style recognition. The twelve standard drive cycles are divided into two groups: six for training and six for test. Simulations show only a 5% gap to DP. Further test on real-world drive cycles shows an average 8% gap to DP results compared to 14% gap by ECMS with the universal λ map and 35% gap by CD-CS strategy.

This DP-ECMS combined optimization method is not limited to vehicle fuel economy optimization. Other factors such as emissions, can be considered in the cost function as well. More generally, for any optimization problem with multiple resources to meet a demand, a weighting factor can be assigned to each resource in the cost function and optimal weighting factors can be found using DP results.

A Real-time, Robust Control Strategy for Plug-in Hybrid Electric Vehicles

by
Rui Wang

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Electrical Engineering

Raleigh, North Carolina

2015

APPROVED BY:

_____	_____
Dr. Srdjan M. Lukic Committee Chair	Dr. Alex Q. Huang
_____	_____
Dr. Iqbal Husain	Dr. Joseph DeCarolis

BIOGRAPHY

Rui Wang was born in Qingdao, China in 1985. He received B.S. degree in Mechanical Engineering from Beijing Institute of Technology (BIT), Beijing, China in 2008. After receiving the B.S. degree he continued to study at BIT in a Ph.D. program in research area of hybrid electric vehicles (HEVs), and transferred to North Carolina State University (NCSU), USA in 2010. At NCSU, he continued the HEV research with FREEDM System Center, and received Ph.D. degree in Electrical Engineering in May, 2015. In 2013, he had a summer internship with Ford Motor Company in Dearborn, MI in the group of Battery Energy Control and Management, and joined the same group in 2015 to begin his career. His research interests include optimal control theory, dynamic programming, control strategy of HEVs and plug-in HEVs, battery management system, and electric vehicle fast charging.

ACKNOWLEDGMENTS

I am grateful for all the blessings from God towards the Ph.D. degree. I would never touch the degree without Him pointing me to the ideas and the people who made it possible.

To my ADVISOR, Dr. Srdjan Lukic, for his continuous, careful guidance and never satisfaction with my work such that I could continuously improve.

To my committee members, Dr. Alex Huang, Dr. Iqbal Husain, Dr. Joseph DeCarolis, for their precious comments and suggestions.

To all the research fellows, Heinrich Enslin, Kibok Lee, Changjian Hu, Feaza Hafiz, Mohammad Etemadrezaei, Chi Zhang, Xinyu Liang, for their support and friendship along my path towards the degree.

To all the faculties and staff members in FREEDM System Center for providing such a great research environment.

To John Gibeau, my manager at Ford, for the great internship opportunity and a promising job offer which transferred the potential time for job hunting into my research time.

To all my friends for the days, joyful or painful, we have spent together, and for the bright future we are heading to.

To China Scholarship Council and my ADVISORs at Beijing Institute of Technology, Fengchun Sun and Hongwen He, for their financial and technical support.

And lastly and foremost, to my parents Jingguo Wang and Jing Liu, my wife Luxiaofei Li, for their love, trust, and support.

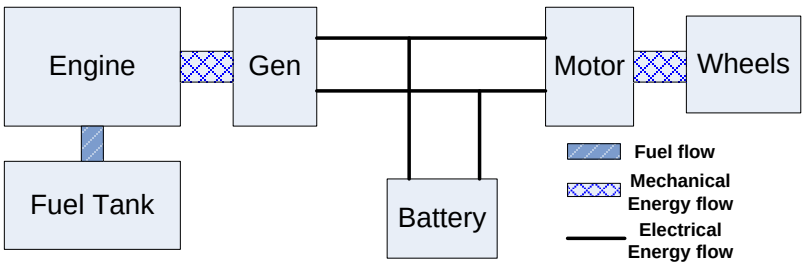
TABLE OF CONTENTS

LIST OF TABLES

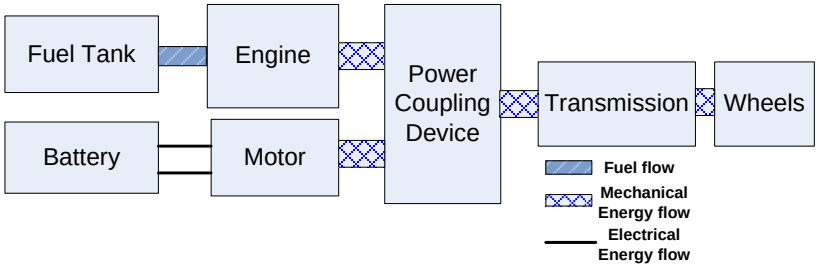
LIST OF FIGURES

Chapter 1 Introduction

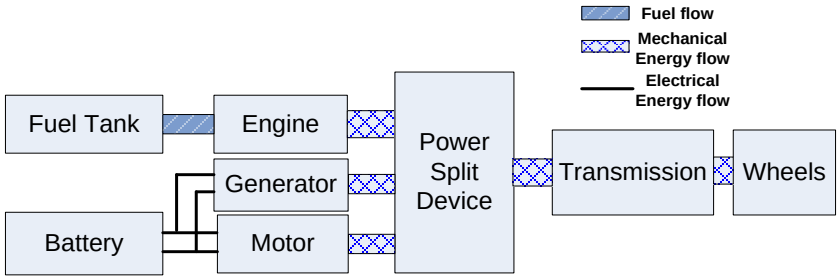
In recent years, increasing oil price and environmental pollution due to fossil fuel usage have attracted growing concerns all over the world. Increasingly stringent emission standards [1] and higher gas prices have forced the automobile manufacturers to accelerate the introduction of fuel-efficient vehicles, such as electric vehicles (EVs), hybrid electric vehicles (HEVs) and fuel cell powered vehicles (FCVs). FCVs typically use hydrogen as their fuel and product only water as the bi-product of the chemical reaction. However, since FCVs are still in experimental demonstration stages and hydrogen refueling station can hardly be found, the mass-market acceptance of FCVs is not expected in near future. EVs use battery energy instead of fossil fuels such that there is no emission from the vehicle, and running cost is a third of that for a regular gasoline vehicle. However, the range of an EV per charge varies from 100 km to 300 km [2], limited by the current battery technology. In addition, the charging time is much longer than that of typical refueling at a gas station. Even for fast charging with a 50kW charger, it takes up to 30 minutes to charge the battery to 80% of the battery capacity (around 100km range with 80% battery capacity) on a Nissan Leaf. Therefore, HEVs which use gasoline as fuel and has no range anxiety issue of EVs seem to be the most acceptable solution to reduce the fuel cost and emissions for the next decade [3]. Toyota Prius as the most popular HEV model worldwide which passed 3 million global cumulative sales in 2013, has a fuel economy of around 50 miles per gallon (MPG) gasoline (equivalent to 4.7 L/100km) in city drive; typical fuel economy on a similar conventional car is about 25 MPG in city drive.



a) Series powertrain



b) Parallel powertrain



c) Series-parallel powertrain

Figure 1.1 HEV powertrain topologies

Three topologies for HEV powertrains have been proposed: series, parallel, and series-parallel, as shown in Figure 1.1. In a series powertrain, the mechanical engine power is transferred to electrical power via a generator and then back to mechanical power via a motor, such that the transmission efficiency is low. However, the engine can be operated at arbitrary operating point (speed / torque) such that the efficiency of engine is maximized. In

addition, no mechanical transmission is needed and the layout of the components is simplified, which is a significant advantage for high power vehicles such as military vehicles and buses. In a parallel powertrain, engine is mechanically coupled to the transmission such that the engine speed is dependent on the vehicle speed, and a motor is used to adjust the engine torque and the motor usually serves as starter as well. The transmission efficiency is hence higher but the engine operating point cannot be fixed to the optimal region, which limits the fuel economy improvement. However, parallel powertrain is simplest if converting a conventional vehicle into a hybrid powertrain since they share most components. Series-parallel powertrain combines the advantages of both series and parallel powertrains with a compromise on cost and system complexity.

Plug-in hybrid electric vehicle (PHEV) combines the advantages of both EV and HEV. Different from HEVs, a PHEV has a larger battery pack that can be recharged by the grid, which can further reduce the running cost and emissions. Also, PHEV removes the range anxiety of EV drivers and can be refueled at any regular gas station.

The fuel economy of HEVs and PHEVs, however, depends on the control strategy to a great extent. The control strategy is the control algorithm in the vehicle controller, which manages the power from both engine and motor to meet the power demand from the driver while regulating the state of charge (SOC) of battery pack. The control strategies for an HEV are very mature but for a PHEV it is still in the early stage. The state-of-the-art charge depleting (CD) - charge sustaining (CS) control strategy forces the battery energy to be utilized in priority, but may be far from optimal when the trip length is greater than the remaining all electric range (AER). Based on simulation results in Chapter 5 and 6, the

potential of fuel economy improvement over CD-CS control strategy can be as high as 50% by an optimal control strategy on a PHEV, so the goal of this thesis is to develop a real-time, robust control strategy for PHEVs.

The remainder of this thesis is outlined as follows. In Chapter 2 and Chapter 3 the state of the art of control strategy for HEVs and PHEVs are reviewed, respectively. Chapter 4 introduces a global optimization method, dynamic programming (DP), and applies it to the three typical HEV powertrains. DP considers all feasible operating points at each step, and therefore guarantees an optimal solution. The result can be used as benchmark to evaluate a real-time control strategy; alternatively, DP results can guide the development of a real-time control strategy [4, 5]. Chapter 5 introduces an equivalent consumption minimization strategy (ECMS) [6-12]. ECMS is based on Pontryagin's Minimization Principle and minimizes an equivalent cost function in form of $J = \dot{m}_{fuel} + \lambda * S\dot{O}C$, where $\dot{m}_{fuel}$is fuel consumption rate, λ is equivalent factor and $S\dot{O}C$ is battery state of charge variation rate. This control method has been implemented to HEVs in successful field tests [7]. However, for PHEVs λ is sensitive to trip length and the implementation of ECMS to HEVs cannot be applied to PHEVs directly. Therefore, the focus of this thesis is to calculate the equivalent factor λ to reflect the driving conditions such as remaining driving distance, driving pattern (city, suburban, and highway) and driver's driving style. For PHEVs, this λ can be interpreted as a factor that converts electric energy into equivalent fuel in gallons, or unit price of electricity from grid over unit price of gasoline. By the first method the unit of the cost function is gallon and miles per gallon gasoline equivalent (MPGe) can be calculated, which is independent on the gasoline and electricity price. However, as the MPGe does not

reflect the actual cost, BYDe6 with 64 MGPge, for example, can cost more compared to Prius with 48 MPG if gasoline price is reduced to $3.2 per gallon [13]. Appropriate cost equivalence between fuel and electricity can be determined based on application. In this thesis the later method is favored and the unit of the cost function is dollar. In Chapter 6, DP combines with ECMS to find a λ value given remaining distance d_R and current SOC for a certain drive cycle. By this method, DP uses λ as control variable, and ECMS calculates the optimal engine operating point based on a given λ value and returns the fuel consumption and SOC variation to DP. The value of λ on a (d_R, SOC) node is obtained by linear regression from the λ values on the corresponding optimal SOC trajectory based on DP results. Therefore, a quasi-optimal look-up table of λ (referred as λ map) can be generated with respect to d_R and SOC for a given drive cycle. With a λ map available, ECMS can be implemented and the only external information is trip length at the beginning of a drive cycle. Simulations show the equivalent cost by ECMS with a quasi-optimal λ map has an average 5% gap to DP over twelve standard drive cycles. A universal map is then built for any drive cycle by fitting the twelve λ maps into one. The gap to DP results with this universal map is 14% on average over the twelve drive cycles, with 30% improvement over CD-CS strategy. 4719 drive cycles are generated from real-world driving diary and simulation results show great robustness of this universal map and an average 9.3% cost reduction compared to CD-CS strategy. Since the algorithm in Chapter 6 relies on the driving distance to the destination, Chapter 7 investigates the possibility to estimate the destination based on historical data. Also observed in Chapter 6 is that the driving pattern (city, suburban, and highway) is another decisive factor of λ, and quasi-optimal λ maps are similar for a same driving pattern.

In Chapter 8 a fuzzy logic controller is designed to recognize the current driving pattern using speed limit and average speed. In addition, another fuzzy logic controller is designed using current pattern and average power demand to compensate the aggressive driving styles. The twelve standard drive cycles are divided into two groups: six in training group and six in test group. The membership functions and rule bases of the both fuzzy logic controllers are trained by the training group, and the simulation results on the test group shows an additional 7% cost reduction on average over ECMS with the universal λ map, and the gap to ECMS with quasi-optimal λ maps is 2% on average. Further, tests on real-world drive cycles shows an average 8% gap to DP results compared to 14% gap by ECMS with the universal λ map and 35% gap by CD-CS strategy. Chapter 9 concludes the thesis and suggests the future work.

This DP-ECMS combined method is not limited to vehicle fuel economy optimization. If emissions are considered as well, the fuel consumption in the cost function can be replaced by both fuel consumption and emissions with a weighting factor for each. More generally, for any optimization problem with multiple resources to meet a demand, a weighting factor can be assigned to each resource in the cost function and optimal weighting factor trajectories can be found using DP results.

Chapter 2 Control Strategies for HEVs

Hybrid electric vehicles (HEVs) have become an effective solution to meet the tightening emission regulations and the need of more fuel-efficient vehicles. The performance of an HEV is, however, decided to a great extent by the power management strategy, also known as control strategy. With two power sources, fuel and electric energy, it is crucial to optimally meet the power demand using the internal combustion engine (ICE) and electric motor(s).

Many types of control strategies for HEVs have been proposed in recent years. In [14-16], the authors summarized the control strategies which have been successfully applied to HEV control. Typically, control strategies can be classified into two main categories: rule-based strategies and optimization-based strategies.

In this chapter, the state of the art of the proposed control strategies for HEVs is reviewed and their advantages and disadvantages are summarized. This will guide the development of a novel strategy in this thesis.

2.1 Rule based strategies

Rule based strategies, as the term suggests, means the strategies in which the control demands are decided by rules that have been determined from experiment results or engineering experience. The rules can be either deterministic rules[17] or fuzzy rules[18], which are typically rule tables or flow charts that are stored in the controller. The control demands are instantaneous and only related to the inputs at current instant, such that this method is inherently implementable in real-time.

2.1.1 Deterministic rule based control strategy

Deterministic rule based control strategy usually has certain operation modes such as "Stop Mode", "Pure Electric Mode", "Hybrid Mode", and "Regeneration Mode", under each of which the control algorithm is exclusive. The transition between the connected modes is decided by transition conditions, which are actually deterministic rules.

Figure 2.1 shows an example of modes and transition conditions for a series HEV. This is the simplest control algorithm trying to regulate the SOC within a desired range $[SOC_{min}, SOC_{max}]$. It has 4 modes and 3 inputs: power demand ($Power_{dmd}$), battery state of charge (SOC), and vehicle velocity (Velocity). The single output is the mode (Mode), and control algorithms can be further decided by the current mode. For example, under Pure Electric Mode engine is off and the power demand of the driver is met by electric motor, while under Hybrid Mode the engine is on and the power demand of the driver is met by both electric motor and the engine. More examples of deterministic rule-based strategy can be found in [6, 19, 20].

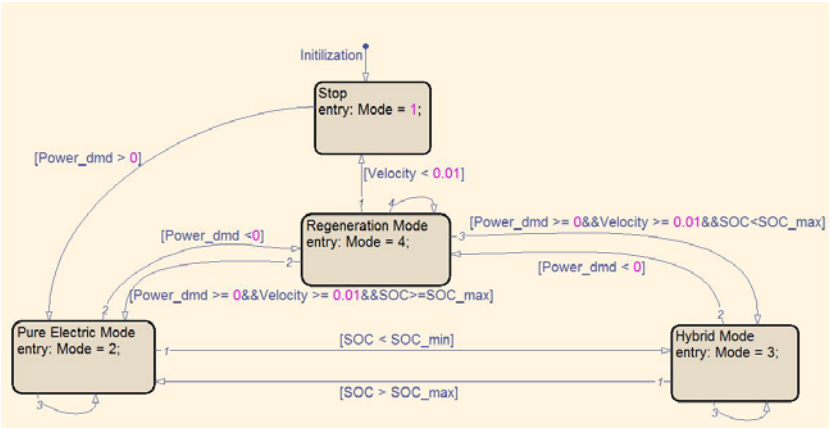


Figure 2.1 Example of modes and transition conditions for a series HEV

The advantages of deterministic rule based strategy are robustness, stability, and computational simplicity allowing easy implementation in real applications. Early commercial HEVs mostly used machine flowcharts [15]. However, the performance, in terms of fuel economy, of deterministic rule based strategy is limited by the finite number of modes and transition conditions. In addition, a fixed threshold for mode transition may lead to mode fluctuation, especially if the signals from the sensors have significant noise.

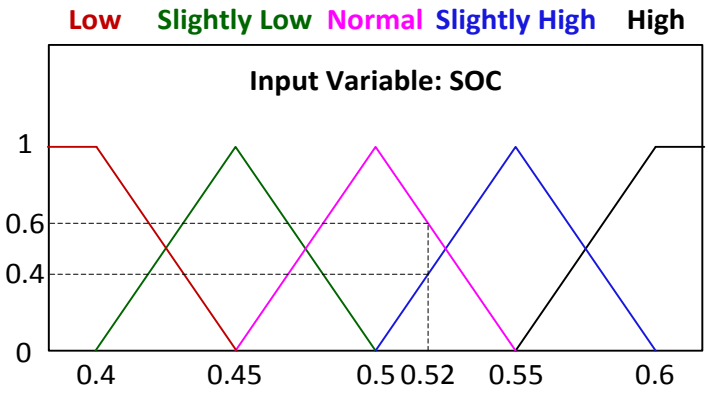
2.1.2 Fuzzy rule based control strategy

To address the mode fluctuation problem, fuzzy logic control is introduced [21]. Instead of a “true / false” type membership, fuzzy logic uses “partially true” type membership function, known as fuzzy sets. The border between two adjacent fuzzy sets is not a fixed threshold but rather a blurry area, which matches the way that humans sense and react. Still for a series HEV, an example of fuzzy sets of SOC is shown in Figure 2.2 a). If SOC is 0.52, for instance, it is 60% “Normal” and 40% “Slight High”. In the same way, fuzzy sets of driver’s power demand as the other input variable can be found in Figure 2.2 b) and the output, the engine power, can be found in Figure 2.2 c).

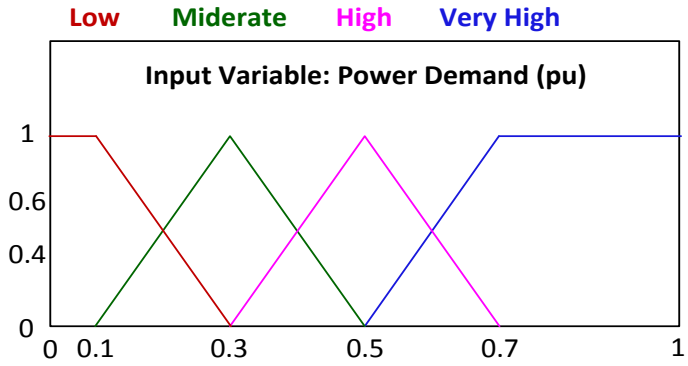
The rule sets are defined, such as

```
rule 1:  If SOC is High AND power demand is Low,
         THEN Engine power is Zero.
rule 2:  IF SOC is Low AND power demand is Low,
         THEN Engine power is Moderate.
   ⋮
```

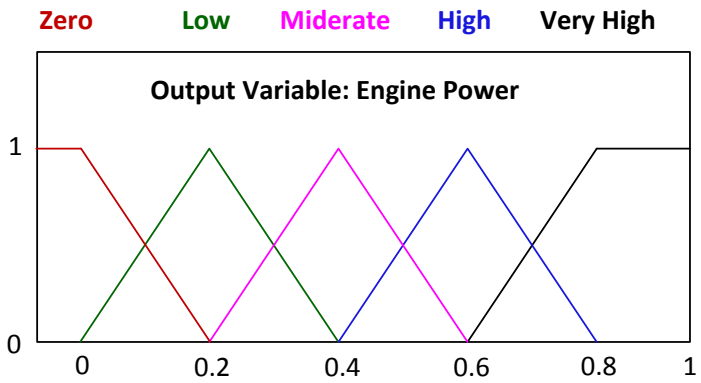
and detailed defuzzification method based on the membership (Normal or Slightly High) and weights (40% or 60%) can be found in [22]. Implementation of fuzzy logic on a parallel HEV can be found in [23].



a) Fuzzy sets of SOC



b) Fuzzy sets of power demand



c) Fuzzy sets of engine power

Figure 2.2 Fuzzy membership example

If we have M inputs and N fuzzy sets for each input, there could be as many as N^M rules. Increasing M and N can typically improves the performance of a fuzzy controller, but a faster microcontroller with larger memory is required to store and process the huge number of rules.

2.1.3 Summary of rule based control strategy

Rule based control strategy only relies on instantaneous inputs, which guarantees real-time implementability and robustness. Deterministic rule-based strategy requires least calculation and memory of microcontroller, but the limited number of modes limits the performance. Fuzzy logic control strategy can significantly increase the number of rules and hence improve the performance, with a compromise on complexity and cost of microcontroller.

The limited rules are designed from experiment results and engineering experience, which is never optimal in all the cases. For example, a set of rules can perform well in one or more driving patterns, but any set of rules cannot guarantee optimal performance in all driving patterns.

2.2 Optimization based strategies

Unlike rule based strategies, an optimization based strategy targets to minimize a cost function with constraints. The cost function is typically a numerical expression of fuel consumption, and can include different terms such as emission, gear changing, engine on/off switching, battery life, etc.[5]. If the drive cycle is completely known in prior, a global optimal solution can be found. As this approach requires the knowledge of past and future drive cycle information, it can only be implemented offline. However, it can either be the benchmark to evaluate the quality of a real-time strategy or guide the design of online rules

[4]. On the other hand, an instantaneous cost function can be defined and local optimal solution can be found by solving an instantaneous optimization problem. Although this approach doesn't guarantee a global optimal solution, real-time implementation can be achieved and the performance is usually comparable to a global optimization approach.

2.2.1 Global optimization

If a cost function is defined and the drive cycle is known, many approaches have been proved applicable to solve an optimization problem [15, 16]. Dynamic programming (DP) [4], optimal control theory [24], linear programming [25], genetic algorithm [26], neural network [27], and particle swarm optimization [28] are examples of global optimization methods that have been applied to control strategy of HEVs.

Among the methods mentioned above, DP is the most popular approach due to its global optimization nature: DP considers all the possible conditions such that an optimal solution is guaranteed. The disadvantage of DP is the heavy calculation burden which is usually not a problem for offline optimization. Detailed DP procedures will be introduced in Chapter 4.

2.2.2 Instantaneous optimization

The complete drive cycle is unknown in most of the cases, such that even if a much more powerful computer can overcome the calculation burden, global optimization methods may still be out of consideration as the control strategy in real applications. Instead, instantaneous optimization which solves an optimization problem for only current step is implementable[15, 16]. The examples include robust control [29], equivalent cost minimization strategy (ECMS) [10], and stochastic dynamic programming [30].

Among the methods mentioned above, ECMS converts the cost of battery energy into equivalent fuel cost and tries to find an optimal operating point of engine to minimize the total equivalent fuel cost, as shown in (2.1)

$$\arg\ min J = \dot{m}_{ice}(t, u(t)) + \lambda(t)\dot{SOC}(t, SOC(t), u(t)) \tag{2.1}$$

where $\dot{m}_{ice}$ donates fuel consumption rate, $\lambda(t)$ is the equivalent factor that converts electric energy cost into equivalent fuel cost, and u(t) includes the engine control parameter vector (torque and speed for example).

Thus, as long as the power demand is given, an optimal operating point of engine can be found without great computation intensity. This method is straightforward and effective, and the only parameter that needs to be tuned is the equivalent factor λ. Detailed procedures of solving an instantaneous optimization by ECMS will be shown in Chapter 5.

2.3 Prediction based strategies

As outlined above, the drive cycle is fundamental to both rule based and optimization based control strategies. An optimal control strategy for city drive where the cars have low speed and high frequency of start/stop may be totally different than that for a highway drive cycle. It is demonstrated in [31] that hybrid electric vehicle (HEV) is sensitive to drive cycle variations, and in [32] that with the information of future driving condition there is potential for significant fuel savings over two urban drive cycles. Thus, it is ideal if the drive condition can be predicted.

There are two main ways to recognize the current and predict the future driving conditions, which are global positioning system (GPS) or intelligent transportation systems (ITS) based techniques and statistic and clustering based analysis methods [33].

2.3.1 GPS/ITS based techniques

Taking advantage of GPS, driving information such as time, speed, trip distance, slope, acceleration, and deceleration could be easily acquired. ITS can be used to provide road condition, speed limit and traffic lights distribution with a higher accuracy. Using the historic data, the real-time monitoring and the traffic modeling techniques, ITS can predict the travel/driving profile with a much less uncertainty.

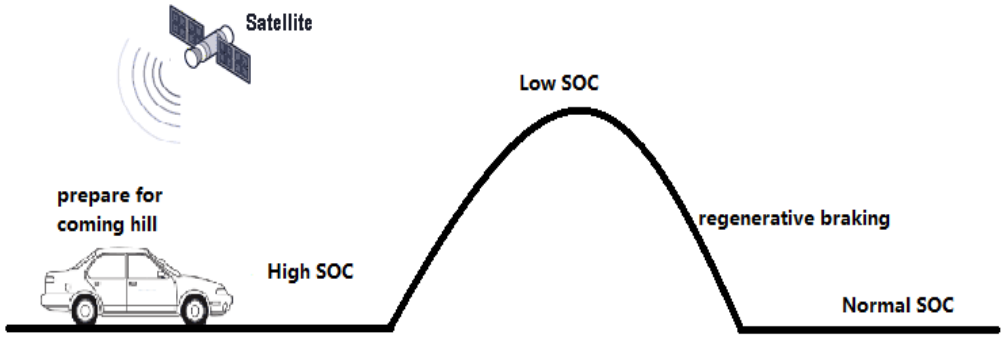


Figure 2.3 Energy management based on terrain prediction

Table 2.1 Fuzzy Rules by Predictive Control Strategy

Future State	**Increasing elevation**	**Constant elevation**	**Decreasing elevation**
Increasing traffic flow	No charging/ discharging	Normal discharging	Deep discharging
Constant traffic flow	Normal charging	No charging/ discharging	Normal discharging
Decreasing traffic flow	High charging	Normal charging	No charging/ discharging

Figure 2.3 shows an example of utilizing GPS to predict future driving condition, terrain information in this case [12]. If the origin and destination of a trip are already defined, GPS is an effective approach to supply terrain information. Table 2.1 shows an example how the traffic flow and terrain information by ITS can be utilized by a fuzzy logic controller [34].

2.3.2 Statistic and clustering based analysis methods

ITS signals are accessible on only few freeways, so it is not reliable at current stage. GPS can plan a route and show traffic conditions based on the average speed of all the drivers, however, the driver may drive slower or faster than the average speed. Therefore, external information has not been widely used. To take full advantage of the data available on board, statistic and clustering methods are preferable. The basic idea for this method is to collect the historical and current driving cycle parameters to analyze the previous driving pattern (city or highway), and assume the driving condition in the future will keep relatively consistent.

2.3.2.1 Data collected to recognize driving conditions

To recognize the driving condition, the representative parameters to be collected should be decided. If there are too many parameters, the computing burden is too heavy for real-time control implementation; on the other hand, too few parameters may not provide meaningful data about the driving pattern. Table 2.2 lists the parameters that have been used to recognize driving patterns in literatures.

Note that in Table 2.2, * means the parameter is used in the corresponding study, and (*) means this parameter is initially considered but later discarded after dominating parameters study.

Table 2.2 Summary of Parameters Used to Recognize Driving Patterns

Parameter	[31]	[32]	[33]	[35]	[36]	[37]	[38]
Average velocity	*	*	*	*			*
Average running velocity except stop	*						
Stop time/total time	*			*		(*)	
Positive acceleration kinetic energy change per unit mass per unit distance	*						
Average acceleration	*	*	*				
Average deceleration	*	*	*				
Average positive grade	*						
Average negative grade	*						
Positive grade time/total time	*						
Negative grade time/total time	*						
Number of stops per kilometer	*						
Average micro-trip time(from start to stop)	*						
Acceleration time/total time	*						
Deceleration time/total time	*						
Standard deviation of acceleration	*	*	*	(*)			
Standard deviation of deceleration	*		*				
Maximum velocity	*	*	*	(*)			
Standard deviation of velocity	*		*	(*)			
Average grade	*						
Maximum grade	*						
Minimum grade	*						
Standard deviation of grade	*						
Standard deviation of positive grade	*						
Standard deviation of negative grade	*						
Trip distance		*					
Maximum acceleration		*	*	*			
Minimum deceleration		*	*	*			
% of time in speed interval 0-15 km/h		*					
% of time in speed interval 15-30 km/h		*					
% of time in speed interval >110 km/h		*					
% of time in deceleration interval (-10)~(-2.5) m/s^2		*					
% of time in deceleration interval (-2.5)~(-1.5) m/s^2		*					
# of acceleration/deceleration shifts per 100m where the difference of adjacent local max-speed and min-speed was > 2 km/h		*					
% of time in speed interval 0-5 m/s			*				
% of time in speed interval 5-10 m/s			*				
% of time in speed interval 10-15 m/s			*				
% of time in speed interval > 15 m/s			*				
% of time in acceleration interval > 7 m/s^2			*				
% of time in acceleration interval 0~ 7 m/s^2			*				
% of time in deceleration interval -7~0 m/s^2			*				
% of time in deceleration interval < -7 m/s^2			*				

Table 2.2 Continued

Parameter	[31]	[32]	[33]	[35]	[36]	[37]	[38]
Maximum product of velocity and acceleration				(*)			
Minimum product of velocity and acceleration				(*)			
Average product of velocity and acceleration				(*)			
Standard deviation of product of velocity and acceleration				(*)			
Current velocity					*		
Driver power demand					*		
SOC					*		
Average positive power demand						*	
Average negative power demand						(*)	
Standard deviation of positive power demand						*	

2.3.2.2 Length of time window for data collection and processing

To realize real-time control, the historical data should be acquired online in a certain time window. If the time window is too short, the historical data may not reflect the driving cycle correctly; if the time window is too large, the computational burden may be too heavy for real-time control, and the data collected long time ago may not be relevant to current driving pattern. A time window of 300s is used in [31] and 150s in [35], and every 300s or 150s, the control strategy is updated based on past driving pattern.

In [32] and [37], although a window of 150s is used, the control strategy is updated every 3 and 5 seconds, respectively. This method can be summarized as

$$\begin{cases} \vec{D}(k) = \vec{f}\left(\vec{P}(t)\middle| t\epsilon[(k-\Delta w), k]\right) \\ \vec{C}(t|t\epsilon[k,(k+\Delta t)) = \vec{G}(\vec{D}(k)) \end{cases} \tag{2.2}$$

where Δw is the time window for data collection, $\vec{P}(t)$ is the data collected, such as velocity, at each time step, $\vec{f}$ is a vector of functions, such as max, min, and average, to process the

data, $\vec{D}(k)$ is a vector of parameters to be used in driving cycle recognition, $\vec{G}$ is a set of functions to decide the current vector of control parameters $\vec{C}$, such as ICE throttle angle and motor current, and Δt is the time period for current control strategy to last. In particular, if $\Delta w = \Delta t$, (2) represents the data collection and processing method in [31] and [35]. Specially, [32] studied how the length of Δw and Δt affect prediction accuracy, pointing out that with larger Δw and shorter Δt, the prediction accuracy will increase.

2.3.2.3 Methods for data analyzing

To define different driving cycles, the characteristic parameters should be extracted from the known driving cycles and be used to train the classification tools or methods. Neural network is the most accepted algorithm [31-33, 36].

2.4 Conclusions

In this chapter, the state of the art of the proposed control strategies for HEVs is reviewed, including rule based control strategies, optimization based control strategies, and prediction based control strategy, which are summarized in Table 2.3. Although rule based strategies may not achieve optimal fuel economy, they can be implemented in real time such that they are the main strategies for current applications. Global optimization methods, on the other hand, can evaluate and guide the development of online rules and improve the performance of rule based strategies. An instantaneous optimization method combines the advantages of both rule based strategies and optimization strategies and will be considered in this thesis. However, not a set of rules is optimal over multiple driving patterns, such that prediction of driving pattern is necessary.

Table 2.3 Summary of Control Strategies in literature

Category	Subcategory	Literature
Rule based	Deterministic Rule based	[17], [6, 19, 20]
	Fuzzy Rule based	[18], [23]
Optimization based	Global optimization	[4] [24] [25] [26] [27]
	Instantaneous optimization	[29] [10] [30]
Prediction based	GPS / ITS	[12], [35]
	Statistics	[31-33, 36-38]

Chapter 3 Control Strategies for PHEVs

3.1 Introduction

Different from HEVs, PHEVs have larger battery capacity. PHEV is a combined technology of EV and HEV, which further improves fuel economy over HEV while removes the range anxiety of EV drivers. The vehicle can operate in two modes: charge depleting (CD) and charge sustaining (CS). In CD mode, a PHEV operates as an EV and the range that can be covered by EV mode is defined as all-electric range (AER) in this thesis. When battery SOC is close to its lower limit, engine is engaged and the vehicle is operated in charge sustaining (CS) mode until the next destination where the battery can be recharged by the grid. By 2014, the AER of the commercial PHEVs available in the United States varies from 11 miles to 38 miles [39]. Ideally this AER is greater than the distance of daily commutes (one way) and the vehicle can be recharged at work place, such that the vehicle is operated as an EV for most of the time to reduce the cost and emissions.

If the distance is greater than the AER, applying the charge depleting (CD) mode followed by charge sustaining (CS) mode will force the battery energy to be utilized in priority. However, this control strategy may be far from optimal when the trip range is greater than the AER [40]. The optimal algorithm has been demonstrated to be a blended control where the engine operates throughout the trip and SOC reaches the lower limit at the end of the trip [40-42].

Unlike HEVs, a PHEV often has a final SOC lower than the initial SOC after a trip. Therefore the control strategies for HEVs, which maintain a small SOC window, cannot be directly implemented to PHEVs. In addition, the distance to the destination is a decisive

factor for a PHEV controller and not for an HEV controller. For example, if a vehicle has an AER of 10 miles, the optimal control strategy of the vehicle for a 5-mile trip should be totally different than that for a 20-mile trip. In this chapter, the state of the art of the proposed control strategies specifically for PHEVs is reviewed. This will also guide the development of a novel strategy in Chapter 6.

3.2 CD-CS and rule based strategies

CD-CS strategy for a PHEV is considered as the state of the art [40]. If SOC is higher than a threshold, engine is not engaged and the power demand is met only by electric motor, and the PHEV operates in CD mode like an EV; otherwise, PHEV is operated in CS mode and the control strategy is very similar to that for an HEV. The mode transition is usually unidirectional, which means if the vehicle is in CS mode, the control strategy will regulate the SOC in a small window and avoid charging the battery to a high SOC level.

This strategy will force the battery energy to be utilized in priority. However, when the trip range is greater than the AER, the optimal algorithm has been demonstrated to be a blended control where the engine operates throughout the trip and SOC reaches the lower limit at the end of the trip [40-42], as shown in Figure 3.1. The fuel economy improvement can be as much as 30% over UDDS drive cycle [5].

Similar to rule based strategies for HEVs, fuzzy logic technique can still be utilized. A fuzzy logic controller is developed in [43] to regulate SOC level of a series PHEV. This controller prevents the battery to be over discharged, but the vehicle still follows CD-CS strategy. Control strategy of a PHEV is sensitive to trip distance unlink an HEV, so it is difficult to design a set of fuzzy rules to meet all the conditions for a PHEV.

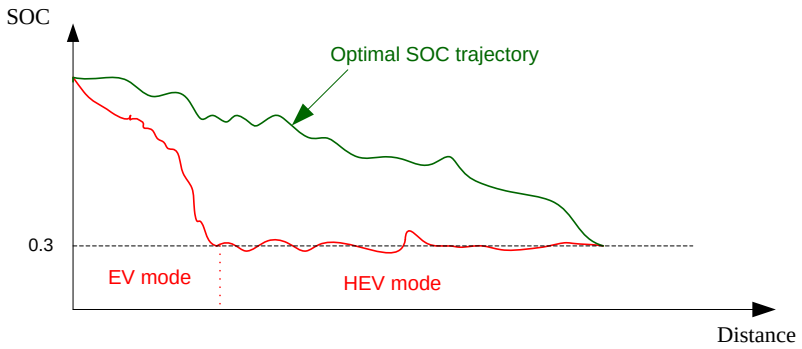


Figure 3.1 SOC trajectory: CD-CS strategy compared to optimal control

3.3 Optimization based strategies

Many optimization methods have been proposed for PHEVs. Although the control strategy for a PHEV can be quite different than that for an HEV, similar optimization methods and tools can be shared. Still, a cost function with constraints can be defined and global optimization or instantaneous optimization can be implemented.

3.3.1 Global optimization

Global optimization methods have been summarized in [44], including particle swarm optimization (PSO) [45], dynamic programming (DP) [46], and game theory [47]. However, methods outlined in [44-47] still require a-priori knowledge of the driving profile, and therefore cannot be implemented in real time.

3.3.2 Instantaneous optimization

Stochastic dynamic programming proposed in [48] can be implemented in real time based on a static map obtained offline [30]. This method is trip independent, and hence does not guarantee that the SOC reaches the lower limit at the destination; in addition, the static map needs a lot of data to initialize.

ECMS is still favored for PHEV control. Different from ECMS for HEVs, the equivalent factor λ which converts electric energy consumption into equivalent fuel cost is no longer a simple function of driving pattern, and more information such as SOC and the remaining driving distance to the destination where the vehicle can be recharged are also important. In [49] a 3-D look-up table of λ is built with respect to current SOC and remaining distance and this algorithm ensures real-time implementation. However, this look-up table is designed based on a simplified equation which does not guarantee convergence over arbitrary drive cycles.

3.4 Prediction based strategies

Driving pattern recognition methods have been summarized in Chapter 2. As a PHEV control strategy is sensitive to not only the driving pattern but also the distance, the trip information is crucial. For trip length information, GPS is still the ideal technology to supply such information as long as the destination is known. When GPS is not accessible or destination is not indicated by the driver directly, statistic methods are still workable to predict the destination and calculate the distance, though the fidelity is lower than GPS.

Statistic methods take the historical data as reference, and the basic assumption is that a driver goes to certain destinations frequently. This assumption is reasonable according to 2009 national household travel survey [50]: at least 60%~70% of the trips are for work, school, church, social, shopping, dining and recreation which have relatively fixed destinations. If the destinations of interest are stored in advance, the probability distribution for the next destination is available given the current location. Hidden Markov model is a useful tool towards destination prediction problem [51, 52], and the probability distribution

of possible destinations given current location can be built based on Bayes' Rule [53, 54]. Network technique is utilized in [55, 56], and in addition to the destination information, time of day, day of week, route being taken, and vehicle velocity can be included in a network. What can be expected is that with more information available to the network, the prediction is more accurate but with heavier computation burden. For example, the learning progress in [56] requires data from the entire vehicle population, and takes up to 5 days to calculate.

With the predicted destination, the remaining distance, which is the information that a PHEV controller requires, can be easily calculated.

3.5 Conclusions

In this chapter, the state of the art of the PHEVs control strategies is reviewed. PHEVs operating principles are quite different from that of HEVs requiring a different set of control strategies and optimization criteria. The controller of a PHEV is very sensitive to the trip length, which is not the case for HEVs. Although PHEVs can follow simple CD-CS control strategy, the performance can be far from optimal when the trip length is greater that the AER. Global optimization tools are quite similar for HEVs and PHEVs and the results are useful to guide online rule development or for benchmarking. However, instantaneous optimization for PHEVs can be more complicated when SOC and trip length are considered. ECMS is still favored due to its real-time implementable nature. The detailed procedures to optimize a look-up table of equivalent factor λ as a function of SOC and remaining distance is introduced in Chapter 6.

Chapter 4 Dynamic Programming for HEVs and PHEVs: Global Optimization

4.1 Introduction

Before a real-time implementable control strategy is developed, a benchmark is necessary to evaluate any control strategy. To find the optimal fuel economy given a drive cycle, a global optimization control algorithm is needed, and the options are outlined in Chapter 2 and Chapter 3. Dynamic programming (DP) technique is an effective way that guarantees a global optimal solution for HEVs given a drive cycle.

In this chapter, DP is respectively applied to three typical HEV powertrains – series, parallel, and series-parallel – and the detailed procedures are given via three flow charts. The case study selects Toyota Prius model since it is currently the most popular commercial HEV model. The simulation results of UDDS drive cycle shows a 30% potential fuel economy improvement over simple rule based strategy. The rule based control algorithm is then optimized based on DP results, and the improvement of fuel economy is around 27%, which is very close to the optimal results.

To implement DP to PHEVs, the procedures for HEVs can be followed. The only difference is that a PHEV optimization will have a different final SOC than initial SOC, which is not the case for an HEV.

4.2 Dynamic Programming principle

Dynamic programming deals with the situations in which decisions are made at each step, with the objective to minimize (maximize) a mathematical expression of cost function [57]. Specifically, a deterministic discrete-state and finite-state system can be expressed as

$$x_{k+1} = f(x_k, u_k) \tag{4.1}$$

and the cost function is of the form

$$J = g_N(x_N) + \sum_{k=0}^{n-1} g_k(x_k, u_k) \tag{4.2}$$

The dynamic optimization problem is over the controls $u_0, u_1, \dots, u_{N-1}$, to minimize/maximize cost function J subject to constraints. Dynamic programming is an effective tool to solve general dynamic optimizing problems mentioned above. It can handle the constraints and obtain a globally optimal solution for nonlinear systems.

The DP technique is based on Bellman's Principle of Optimality [58], stating that the optimal policy can be obtained if single-stage sub-problem involving only the last stage is solved first, and then sub-problem involving last two stages, last three stages, etc. until the entire problem is solved step by step. In this case, the overall dynamic optimization problem, taking minimizing problem for example, can be expressed as

$$J_N(x_N) = g(x_N) \tag{4.3}$$

$$J_k(x_k) = min_{u_k \in U_k}[g_k(x_k, u_k) + J_{k+1}(x_{k+1})] \quad k = N-1, N-2, \dots, 1 \tag{4.4}$$

The preceding expression proceeds backwards in time. However, a deterministic finite-state problem is equivalent to a shortest path problem and can be solved by either backward or forward DP algorithm [57]. The forward DP can be expressed as

$$J_0(x_0) = g(x_0) \tag{4.5}$$

$$J_k(x_k) = min_{u_k \in U_k}[g_k(x_k, u_k) + J_{k-1}(x_{k-1})] \quad k = 1, \dots, N \tag{4.6}$$

At step k+1, $J_{k+1}(x_{i,k+1})$ is the minimum cost at state $x_{i,k+1}$ among the costs for every admissible route from $x_{j,k}$, which can be explained in Fig. 4.1. In Fig. 4.1, each dot

represents an SOC node with an exclusive SOC value. After $J_{k+1}(x_{i,k+1})$ for every $x_{i,k+1}$ is calculated, the DP continues to the next step k+2, k+3, … until step N.

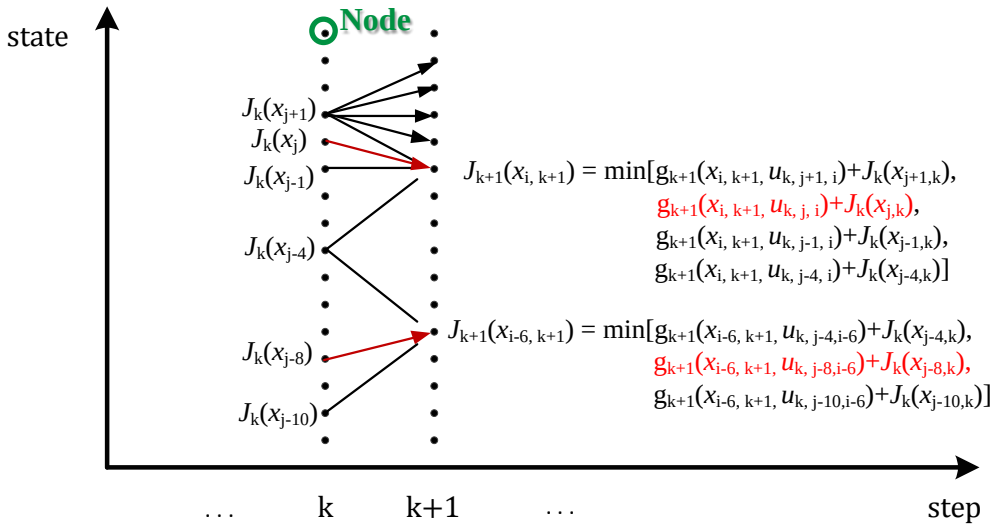


Figure 4.1 Forward dynamic programming

4.3 DP in HEV Optimization

DP has been widely used in HEV optimization. The model of an HEV is identical to (4.1) where x_k is the state vector including vehicle speed, SOC, engine speed, motor speed, etc. while u_k is the control vector that can cover but is not limited to gear number, generator torque, engine torque, motor torque, etc. Note that the parameters mentioned above as state vector can be control parameters sometimes, depending on the control algorithm. The optimization goal is to find the control u_k at each step to minimize a cost function (4.7). Constraints as (4.8) are necessary during the optimization, and it is explained in Section 4.4 how they are met.

$$J_0(x_0) = J_0$$

$$J_k(x_k) = min_{u_k \in U_k}[fuel(x_k, u_k) + J_{k-1}(x_{k-1})] \quad k = 1, \dots, N \tag{4.7}$$

subject to

$$\omega_{e\,min} \leq \omega_e(k) \leq \omega_{e\,max}$$

$$T_{e\,min}(\omega_e(k)) \leq T_e(k) \leq T_{e\,max}(\omega_e(k))$$

$$T_{m\,min}(\omega_m(k), SOC(k)) \leq T_e(k) \leq T_{m\,max}(\omega_m(k), SOC(k))$$

$$SOC_{min} \leq SOC(k) \leq SOC_{max} \tag{4.8}$$

Multiple optimization objects can be considered in the optimization. For example emissions and gear changing can be also considered in the cost function by adding more variables multiplied by weights. For a PHEV, it is better to minimize the overall cost rather than simply fuel cost while the battery peak power is limited to improve the battery state of health (SOH) over time. The overall cost function can be expressed in (4.9)

$$J_k(x_k) = \min_{u_k \in U_k} [fuel(x_k, u_k) + \sum \alpha_i \cdot Emn_i(k) + \beta \cdot I_{bat}^2(k) + \gamma \cdot \Delta gear + \cdots + J_{k-1}(x_{k-1})] \quad k = 1, \dots, N-1$$

$$J_k(x_k) = min_{u_k \in U_k} [fuel(x_k, u_k) + \sum \alpha_i \cdot Emn_i(k) + \beta \cdot I_{bat}^2(k) + \gamma \cdot \Delta gear + \cdots + J_{k-1}(x_{k-1})] + \varepsilon \cdot \Delta SOC \quad k = N \tag{4.9}$$

where $\mathrm{Emn_i(k)}$ relates to the emissions, such as NO_x and HC, at step k with the weight α_i, $I_{bat}(k)$ is battery current at step k, Δgear equals to absolute value of gear number at step k-1 minus gear number at step k. At step N, one additional term $\varepsilon\Delta$SOC is added where ΔSOC equals to initial SOC_0 minus final SOC_N. The coefficient ε converts electricity cost to equivalent fuel cost. Note that this is only for a PHEV, and for a standard HEV, ε should be substantially large and the penalty term should be $\varepsilon|SOC_N^* - SOC_N|$ where SOC_N^* is the desired final SOC.

4.4 Discussion of penalty term $\varepsilon\Delta SOC$

At the final step of DP, to compare the cost on different SOC nodes on which ΔSOC values are different, the electric energy cost should be converted comparable to other costs by the weighting factor ε. For PHEVs, this ε can be interpreted as a factor that converts electric energy into equivalent fuel in gallons, or unit price of electricity from grid over unit price of gasoline.

By the first method the unit of the cost function is gallon and miles per gallon gasoline equivalent (MPGe) can be calculated [13]. MPGe is a measure of the average distance that can be traveled by consuming per unit of energy, which is independent on the gasoline and electricity price. It calculates as a constant over the wall-to-wheel electrical energy consumed per mile. The comparison of MPGe rated by the U.S. Environmental Protection Agency (EPA) for EVs for the U.S. market as of October 2014 can be found in [13], and the fuel economy varies from 64 MPGe to 111 MPGe.

However, the MPGe does not reflect the actual cost, it is indirect for users to compare the operating cost. BYDe6 with 64 MPGe, for example, costs $1.62 to drive 25 miles compared to Prius hybrid with 48 MPG which costs $1.74 per 25 miles. The MPGe and MPG are calculated based on 45% highway and 55% city driving, and the actual cost in dollar has very small difference while the difference of MPG(e) is much larger. The cost is calculated based on the electricity cost of $0.12/kw-hr and regular gasoline price of $3.49 per gallon [13]. If gasoline price is reduced to $3.2 per gallon, the MPGe and MPG values remain the same, while Prius costs less than BYDe6. In this thesis the later method is favored and the unit of the cost function is in dollars. It is also important to keep the unit of cost function consistent

for evaluating real-time strategies, and in this thesis equivalent operating cost in dollars is evaluated.

4.5 Detailed DP procedures for different powertrain topologies

In this section, detailed procedures of DP for three typical HEV powertrain topologies are introduced. For each topology, a different method of engine modeling is adopted, based on the available data type. Also, two different types of control parameters, power and torque/speed, are discussed in this section.

4.5.1 Parallel HEV Model and DP Procedures

Let's consider a typical parallel powertrain, an integrated motor assistant (IMA) system. For a parallel HEV with an IMA system, the engine couples to the wheels whenever the car runs. Hence, the speed of engine/motor set ω(rad/s) can be taken as a known parameter by (4.10) as long as transmission ratio and vehicle speed v (m/s) are given. In (4.10), v is the vehicle speed in m/s, i_g is the gear ration of transmission, i_o is the gear ration of final drive, and r is the radius of the wheels.

$$\omega = \frac{v}{i_g i_o r} \tag{4.10}$$

The simplified IMA system model can be described as below.

4.5.1.1 Engine model

The engine is modeled as a black box with inputs of engine/motor speed ω(rad/s) and engine torque T_e(Nm), and output of fuel mass flow rate $\dot{m}_f$(kg/s) [59]. A simplified engine model can be described by (4.11) and (4.12). For a parallel HEV, T_e is a control variable as vehicle speed is dependent on vehicle speed. The fuel consumption rate is usually from the engine map as a look-up table. Also, it can be expressed using an equation in (4.11).

$$\dot{m}_f(\omega, T_e) = a_1\omega + a_2\omega^2 + a_3\omega^3 + a_4\omega T_e + a_5\omega^2 T_e + a_6\omega T_e^2 \quad (4.11)$$

$$T_{e\,max}(\omega) = b_1\omega + b_2\omega^2 + b_3\omega^3 \quad (4.12)$$

This polynomial expression of fuel consumption rate saves memory of controller and eliminated the error of interpolation, but more calculation is expected. Also, the accuracy of this modeling method is dependent on the engine map, and sometimes weights a_i and b_j are difficult to find.

4.5.1.2 Battery model

The battery is modeled as a voltage source V_{oc} in series with an internal resistance R_{int} and a terminal resistance R_t, each of which is a function of SOC, and the equivalent circuit of battery model is shown in Fig. 4.2. SOC is updated based on Coulomb integral method, as shown in (4.12), where I is the current. A discrete model in (4.13) calculates the SOC at each step by torque T_m(Nm) and speed ω of motor, where Q_b is the capacity of the battery pack, η_m is the efficiency of motor.

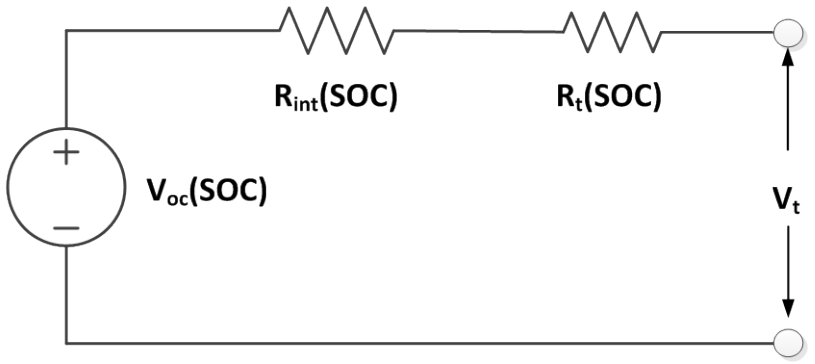


Figure 4.2 Battery circuit model

$$\Delta \mathrm{SOC} = \frac{\int Idt}{Q_b} \tag{4.12}$$

$$SOC_{k+1} = SOC_k - \frac{V_{oc} - \sqrt{V_{oc}^2 - 4(R_{int}+R_t)\cdot T_m \cdot \omega \cdot \eta_m^{-sgn(T_m)}}}{2(R_{int}+R_t)\cdot Q_b} \tag{4.13}$$

4.5.1.3 Vehicle dynamics

Force balance equation (4.14) is utilized here to calculate the torque demand on engine and motor at each step, where v(k) is vehicle velocity at step k, f is rolling resistance coefficient, τ is rotating mass coefficient, M is vehicle mass, g is gravity acceleration, C_D is aerodynamic coefficient, A is frontal area of vehicle, $\rho_a$is air density, i_o is transmission ratio on final drive, η is the mechanical transmission efficiency, θ_k is the slope at step k, r is wheel radius, and where $\bar{v} = \frac{v(k+1)+v(k)}{2}$.

$$\frac{[T_e(k)+T_m(k)]i_o\eta}{r} = \tau M[v(k+1) - v(k)] + fMg\cos\theta_k + \frac{C_D A \rho_a}{2}\bar{v}^2 + Mg\sin\theta_k \tag{4.14}$$

The procedures for implementing DP to a parallel powertrain can be summarized in Fig. 4.3. There are two control variables, T_e and gear ratio of transmission, such that this control method is considered as torque/speed control.

The advantage of the algorithm above is that it calculates only limited number of iterations in the inner loop for each SOC_k. If 1 Nm increment of engine torque is considered for each iteration, the number of interactions is the value of T_{e_max}. For example, if $T_{e\_max} =$ 100 Nm, then there are 100 iterations in the inner loop. The compromise, however, is the sacrifice of accuracy because the calculated values of admissible SOC_{k+1} may not fall exactly on a SOC node and quantization and interpolation are needed.

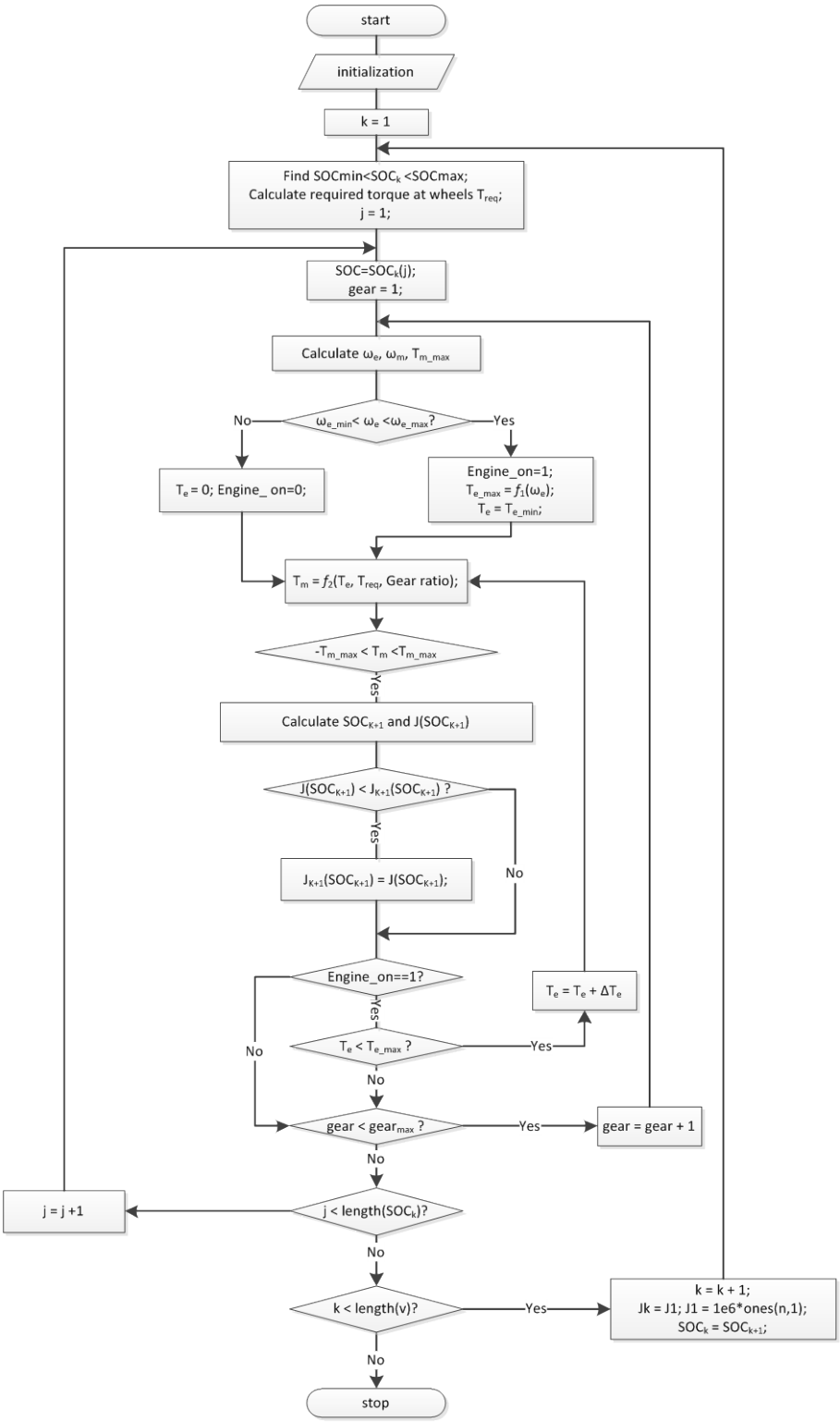


Figure 4.3 Flow chart of DP procedures for parallel powertrain

To address this issue the increment of SOC is selected substantially small (0.01% in this thesis) such that the error can be neglected. One improvement can be achieved by inversely calculating T_m for each admissible pair of SOC_k and SOC_{k+1} using (4.13) while T_m and T_e are both within the feasible range. However, as the density of SOC grid is much larger, at each step many more combinations should be considered such that the calculation burden is much heavier. For example, if we still consider a 0.01% increment of SOC, there are 8000 nodes at each step (20%~99.99%), and 8000^2 combinations are considered compared to 8000×100 iterations using T_e as variable.

4.5.2 Series HEV Model and DP Procedures

In a typical series HEV powertrain, the engine is mechanically decoupled with the powertrain. As a result, both engine speed and torque are controllable at each step, and given a power requirement the best operation point can be found out to minimize fuel consumption (Fig. 4.4). Thus, engine power P_e can be selected as the control variable, which differs from T_e and gear ratio for parallel HEV. The simplified series HEV model can be described as below.

4.5.2.1 Engine model

On the engine map (Fig. 4.4), power contour lines are drawn, and the red numbers are the engine power in kW. For each single $P_e \epsilon [P_{e\,min},\ P_{e\,max}]$, the minimum fuel consumption point can be found, which are marked as green '+'. Then the fuel consumption rate can be simply a function of P_e in form of (4.16).

$$\dot{m}_f = c_1 P_e^2 + c_2 P_e + c_3 \tag{4.16}$$

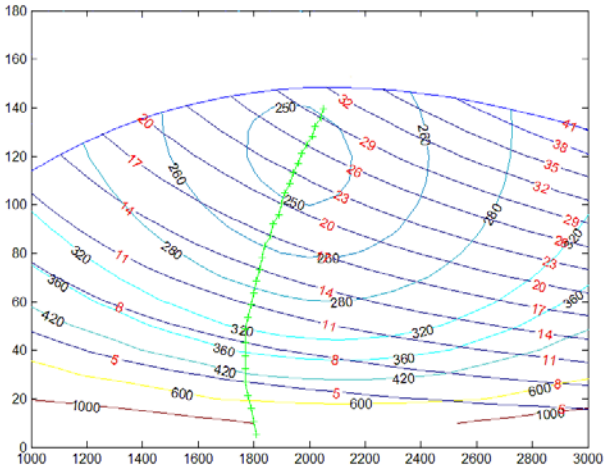


Figure 4.4 Minimum fuel consumption line for a given power demand on the engine map

4.5.2.2 Battery model

As power control can be adopted for series HEV, the battery model should use battery power P_b as input.

$$SOC_{k+1} = SOC_k - \frac{V_{oc} - \sqrt{V_{oc}^2 - 4(R_{int} + R_t) \cdot P_b}}{2(R_{int} + R_t) \cdot Q_b} \quad (4.17)$$

4.5.2.3 Vehicle dynamics:

As power control instead of torque control is used, the power balance equation is preferable for vehicle dynamics.

$$[P_e(k) \cdot \eta_1 + P_b(k)] \cdot \eta_2 \eta = \tau M[v(k+1) - v(k)] \cdot \bar{v} + fMg\cos\theta_k \cdot \bar{v} + \frac{C_D A \rho_a}{2} \bar{v}^3 + Mg\sin\theta_k \cdot \bar{v} \quad (4.18)$$

where η_1 represents the efficiency of generator and power electronics, and η_2 represents the efficiency of motor and power electronics. The procedures for implementing DP to a series powertrain can be summarized in Fig. 4.5.

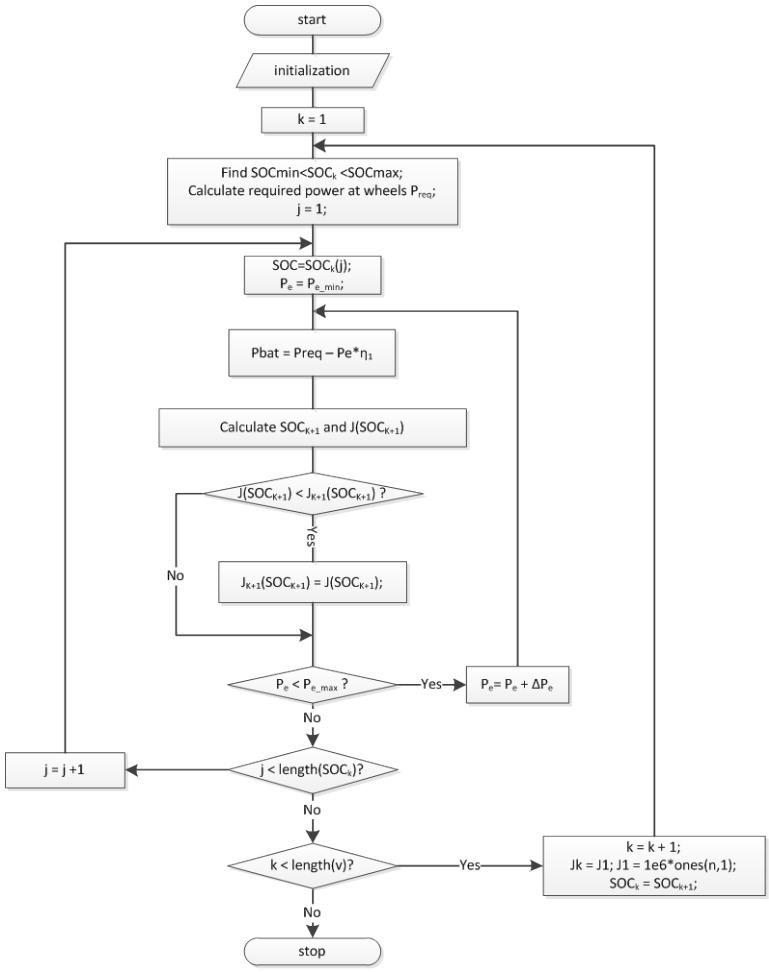


Figure 4.5 Flow chart of DP procedures for series HEV powertrains

The advantages and disadvantages are similar to those discussed in section 4.4.1. The improvement can also be to inversely calculate P_m for each admissible pair of SOC_k and SOC_{k+1} using (4.17) while P_b and P_e are within the feasible range, with a cost of computation burden.

4.5.3 Series-parallel HEV Model and DP Procedures

Series-parallel powertrain combines the advantages of both series and parallel powertrains, with compromise on manufacturing cost and control complexity. In this section,

a typical series-parallel powertrain, Toyota Hybrid System (THS) for Prius (Fig. 4.6), is discussed.

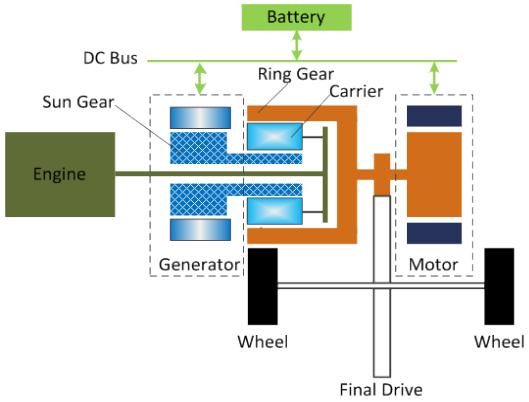
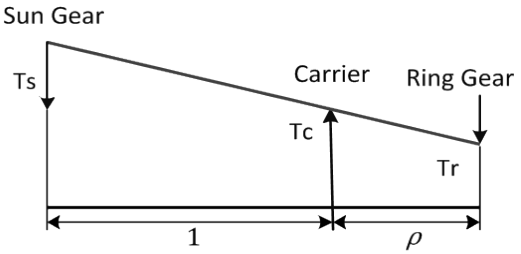
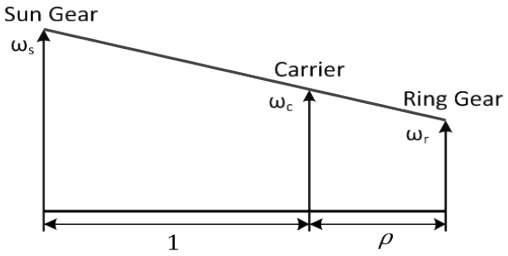


Figure 4.6 Prius powertrain



a) lever model for torque analysis



b) lever model for speed analysis

Figure 4.7 Lever model for plenary gear set

The simplified series-parallel HEV model can be described as below.

4.5.3.1 Planetary gear set

Planetary gear set can be modeled as an ideal "lever" [60] as shown in Fig. 4.7, where $\rho = \frac{Z_{sun}}{Z_{ring}}$ (ρ<1) is the gear ratio, Z_{sun} and Z_{ring} are teeth number of sun gear and ring gear, respectively.

$$\begin{cases} T_e = (1+\rho)\left(T_{req} - T_m\right) \\ T_g = \frac{\rho}{(1+\rho)} T_e \\ \omega_e = \frac{1}{(1+\rho)}\omega_m + \frac{\rho}{(1+\rho)}\omega_g \end{cases} \tag{4.19}$$

T_{req} and ω_m at each step are known for a given drive cycle, so there are five unknowns ($T_e, T_g, T_m, \omega_e, \omega_g$) and three equations in (19) with degree of freedom equals to 2.

On the other hand, power control is still admissible if mechanical transmission efficiency is neglected.

$$\begin{cases} P_e = P_r + P_g \\ P_m = P_g + P_b \\ P_{req} = P_m + P_r \end{cases} \tag{4.20}$$

P_{req} at each step is known for a given drive cycle, so there are five unknowns (P_e, P_r, P_g, P_m, P_b) and three equations in (20) with degree of freedom equals to 2.

4.5.3.2 Engine model

Engine model is identical to either the one in section 4.4.1 for speed/torque control or that in section 4.4.2 for power control. Also, using a look-up table based engine fuel consumption map for interpolation is always feasible.

4.5.3.3 Battery model

Battery model for speed/torque control is identical to (4.13) except that the term $T_m \cdot \omega \cdot \eta_m^{-sgn(T_m)}$ is replaced by $T_m \cdot \omega_m \cdot \eta_m^{-sgn(T_m)} - T_g \cdot \omega_g \cdot \eta_g^{-sgn(T_g)}$. For power control, it is identical to (4.16) except that the term P_b is replaced by $(P_b - P_g)$.

4.5.3.4 Vehicle dynamics

For speed/torque control force balance equation (4.21) is used while for power control power balance equation (4.22) is used.

$$\frac{[T_e(k)/(1+\rho)+T_m(k)]i_o\eta}{r} = \tau M[v(k+1)-v(k)] + fMgcos\theta_k + \frac{C_D A \rho_a}{2}\bar{v}^2 + Mgsin\theta_k \quad (4.21)$$

$$[P_e(k) - P_g(k) + P_m(k)] \cdot \eta = \tau M[v(k+1)-v(k)] \cdot \bar{v} + fMgcos\theta_k \cdot \bar{v} + \frac{C_D A \rho_a}{2}\bar{v}^3 + Mgsin\theta_k \cdot \bar{v} \quad (4.22)$$

The procedures for implementing DP to a series-parallel powertrain are similar to that in 4.5.1 or 4.5.2 based on the control method. For speed/torque control, the control variables can be any pair of the five unknowns in (4.19) with one torque variable and one speed variable. As the cost function relates to engine speed and torque directly, T_e and ω_e can be good choices. For power control, the control variables can be any pair of the five unknowns in (4.20). P_e and P_b can be good choices. Since (4.20) cannot include detailed efficiency of each component, this model is not as accurate as the model in (4.19). The flow chart of speed/torque control based DP procedures for series-parallel powertrain is shown in Fig. 4.8.

4.6 Case Study

In this section DP is implemented to gen-I Prius model (model year 2001-2003), which is obtained from software ADVISOR [61], to find the potential of fuel economy improvement.

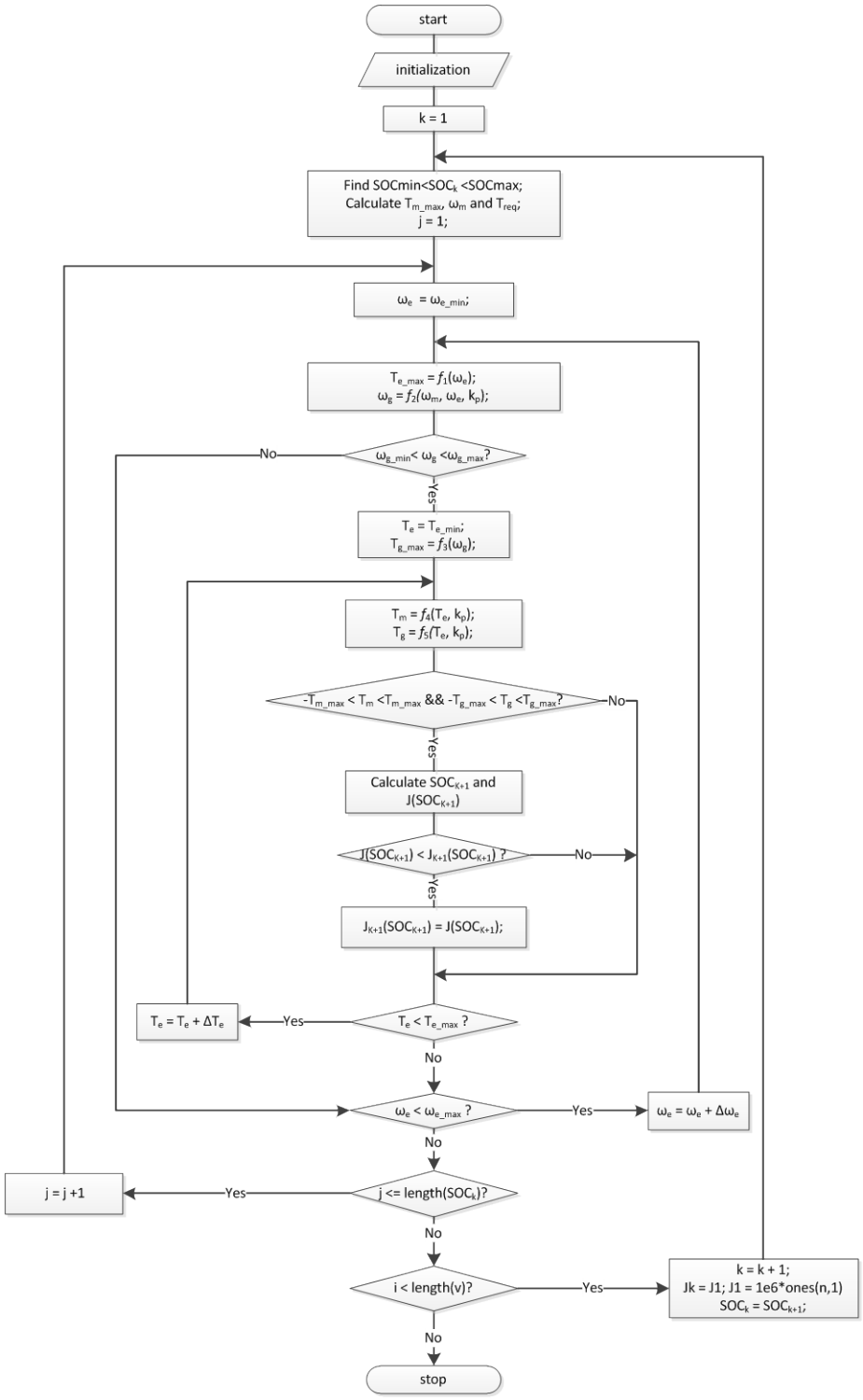


Figure 4.8 Flow chart of speed/torque control based DP procedures for series-parallel HEV powertrains

4.6.1 Vehicle Parameters

The vehicle parameters are shown in Table 4.1, where M is the mass of the vehicle, R_w the wheel radius, i_o the gear ratio of final drive, ρ the gear ratio of planetary gear set, f_r rolling friction coefficient, S the area of frontal surface, C_d the air drag coefficient, ρ_a the density of air, and Q_b the battery maximum capacity. All these parameter are obtained from the Prius model in ADVISOR.

4.6.2 Vehicle Model

The engine is modeled as a static map with inputs engine speed and engine torque and output fuel consumption. The engine map is obtained from ADVISOR. In the same way, the generator and motor are respectively modeled as static maps with input torque and speed and output efficiency. Note that the efficiency here is the overall efficiency considering the efficiency of power electronics. The battery model is identical to (4.13) and the parameters of resistance and capacity are from ADVISOR. The vehicle dynamics is identical to (4.19) and (4.21).

4.6.3 Model Validation

As we take the simulation results in ADVISOR as reference, the simplified vehicle model (SVM) is validated as follows: 1) run the Prius model in ADVISOR over UDDS drive cycle, 2) obtain the operating points at each step, and 3) implement the same operating points in SVM to compare the simulation results.

The fuel consumption of the engine over UDDS cycle by SVM is 427.9416g (48 MPG) compared to 436.7761g (47 MPG) given by ADVISOR, with an error of 0.2%. The error is

due to the cold start correction in ADVISOR model and not in SVM. However, the error is small and can be neglected.

Table 4.1 Vehicle Model Parameters

Parameter	value	unit
M	1365	kg
R_w	0.287	m
i_o	3.93	-
ρ	30/78	-
f_r	0.015	-
S*Cd	1.746*0.3	m^2
ρ_a	1.2	Kg/ m^3
Q_b	6	Ah

The electric system model, which contains generator, motor and battery, is validated via the input power of generator and motor, which is calculated from the operating points (speed and torque) of the generator and motor in ADVISOR. Columbic effect is considered when battery is charged and the comparison of battery power is shown in Fig. 4.9. The battery performance in SVM is similar to that in ADVISOR.

Thus, we are confident that the two models are comparable.

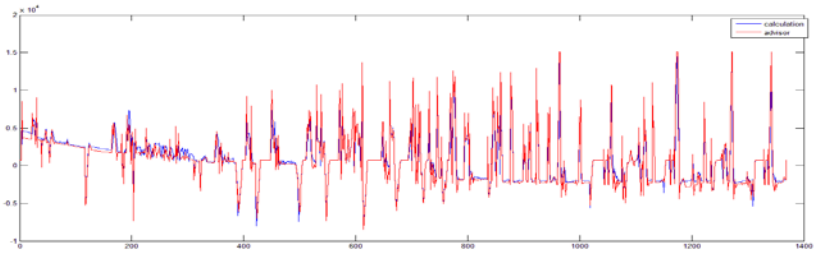
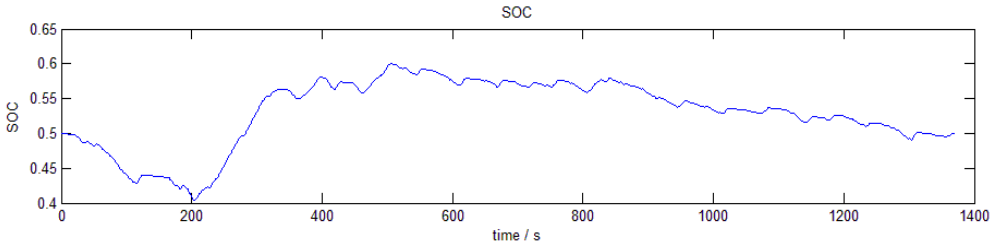


Figure 4.9 The battery model comparison

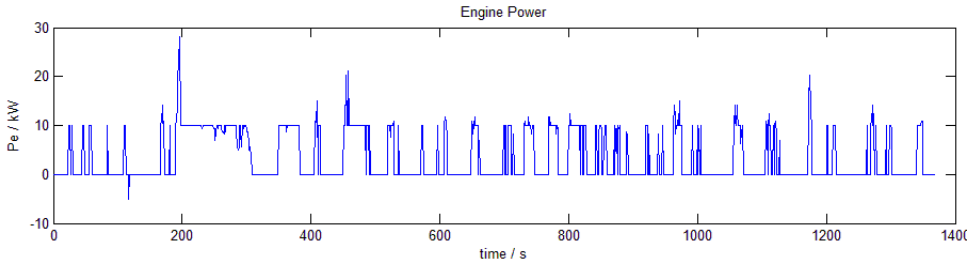
4.6.4 Simulation Results and DP Algorithm Improvement

4.6.4.1 Refinement of cost function:

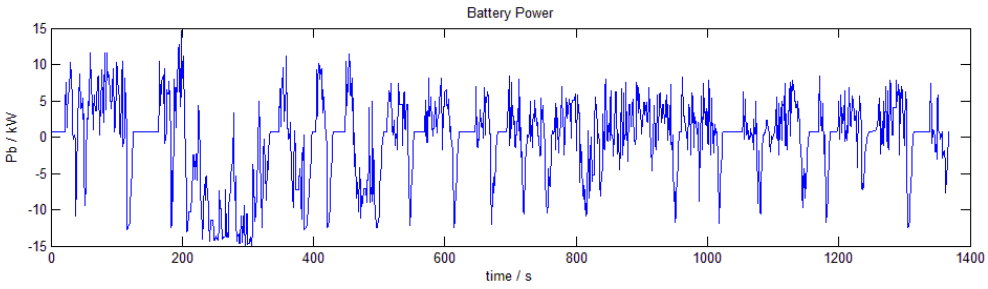
The UDDS drive cycle is used to evaluate the algorithms mentioned in Section 4.4.3. The initial and final SOC are set to 0.5, and at first only the fuel economy is optimized. The results are shown in Fig. 4.10.



a) Battery SOC

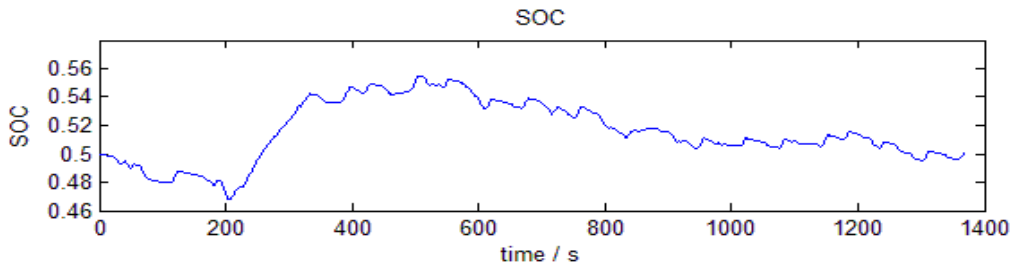


b) Engine power

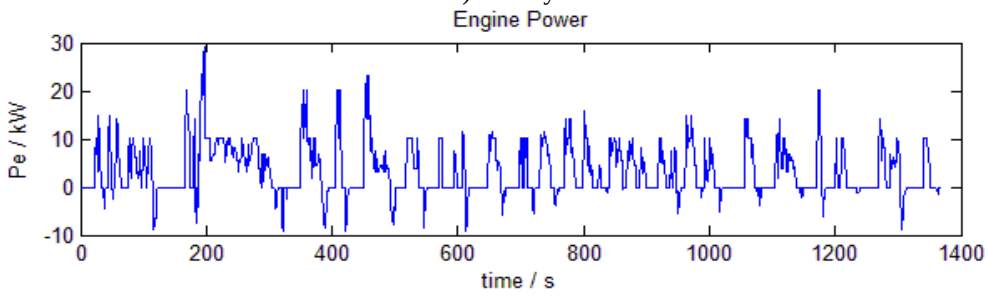


c) Battery power

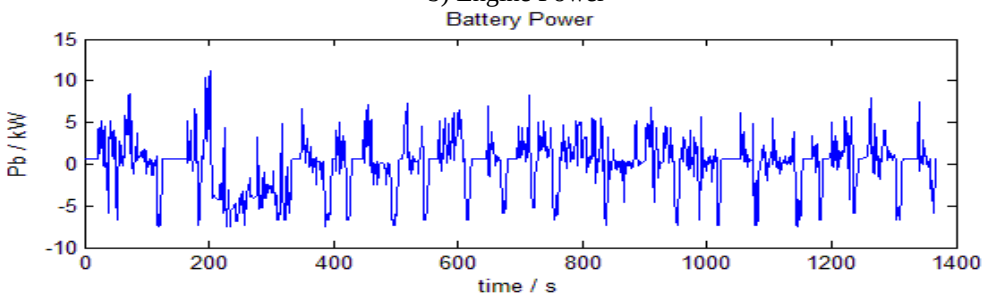
Figure 4.10 Simulation results on Prius powertrain



a) Battery SOC



b) Engine Power



c) Battery power

Figure 4.11 Simulation results with updated cost function

It is observed that the engine turns on and off very frequently, the battery peak power is large, and the brake energy is 100% regenerated. To make the model more feasible, the penalty terms for engine on/off and battery current are added and the engine brake is considered. The cost function is updated as

$$J_k(x_k) = \min_{u_k \in U_k} [fuel(x_k, u_k) + \lambda \cdot I^2 + \gamma \cdot \Delta Engine\_on/off + J_{k-1}(x_{k-1})] \qquad k = 1, \dots, N \tag{4.22}$$

where I is the battery current and $\Delta Engine\_on/off$ is the absolute value of current engine on/off status (1 for 'on' and 0 for 'off') minus that of last step. After many trails, the

parameter of λ is set to 0.0003 and γ is set to 0.1. Then the simulation results are shown in Fig. 4.11.

4.6.4.2 On-line Control Strategy based on DP Results

As discussed in Chapter 2 and Chapter 3, DP is an off-line simulation and requires a known drive cycle which is unknown in real cases. However, with the optimal operating points obtained via DP, an implementable online control strategy can be developed with quasi-optimal performance.

In [62] the author uses a simple power-split algorithm for a parallel HEV. The optimal power split ratio (PSR), which equals engine power over demanded power, is a function of demanded torque at the input shaft of transmission. In this case, for any given demanded torque the optimal PSR can be found and hence the optimal engine power. However, this study is for a parallel powertrain where the engine is mechanically coupled to the wheels and the engine speed is dependent on the vehicle velocity and gear ratio of the transmission. In a series-parallel powertrain the engine speed is independent and there is not a strong relationship between the engine power and demanded power (or torque), as shown in Fig. 4.12 and Fig. 4.13.

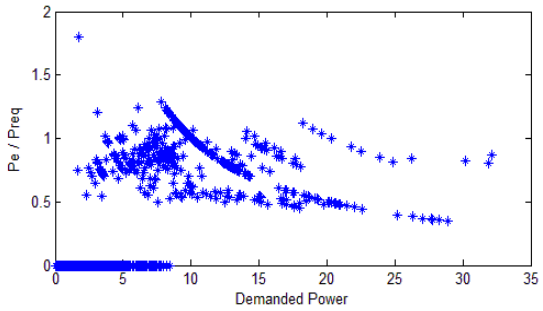


Figure 4.12 Relationship between PSR and demanded power in DP results

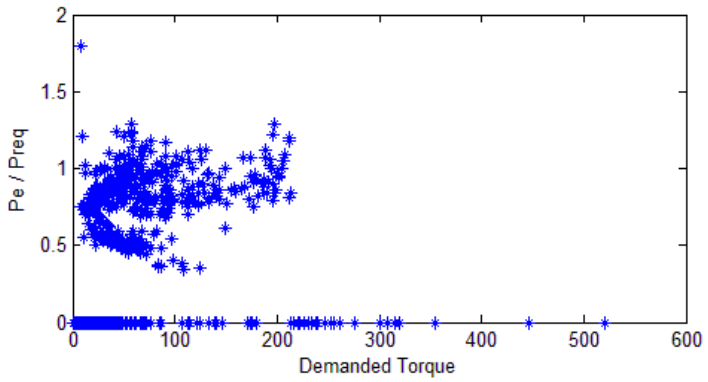


Figure 4.13 Relationship between PSR and demanded torque in DP results

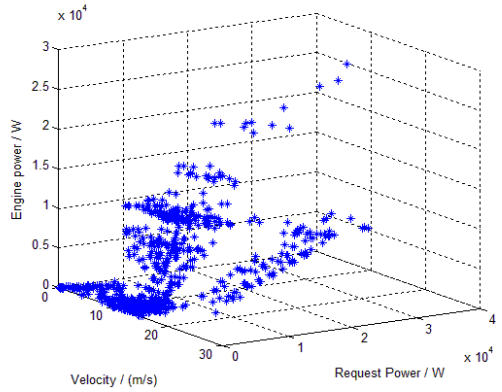


Figure 4.14 Relationship among Pe, Preq and v in DP results

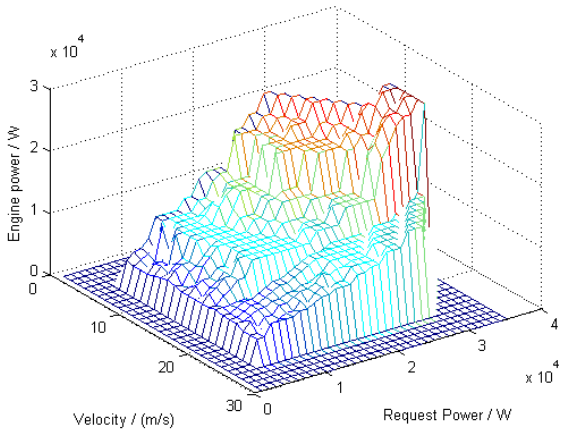
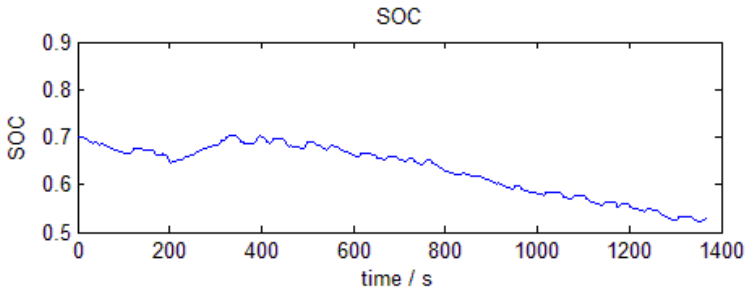
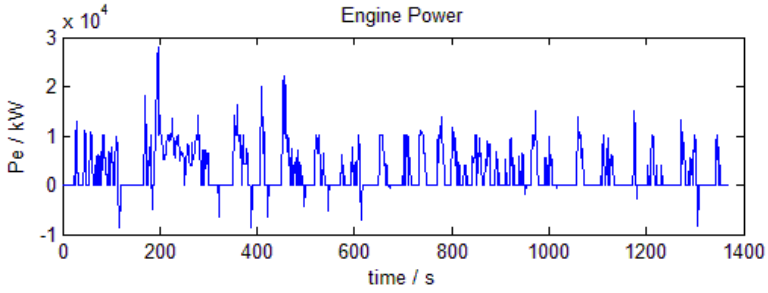


Figure 4.15 2D lookup table based on DP results

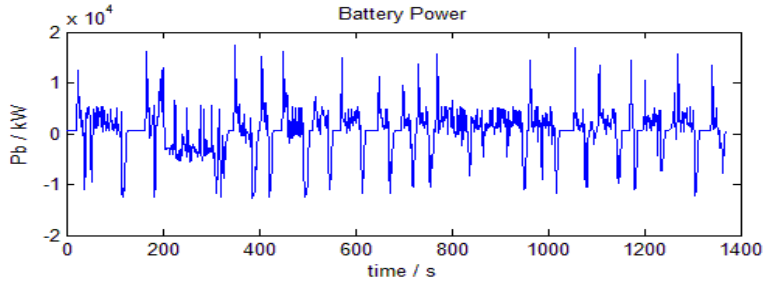
If one more variable, the vehicle velocity, is added, the 3D plot is shown in Fig. 4.14. The discrete operating points are not able to build a 2D lookup table, so linear interpolation is used and the lookup table is shown in Fig. 4.15. This lookup table in Fig. 4.15 can be considered as a set of 30×35 "*if* … *then* …" rules, which can be implemented in real-time control. The simulation results are shown in Fig. 4.16 and Table 4.2.



a) Battery SOC



b) Engine power



c) Battery power

Figure 4.16 Simulation results with new rule-based control algorithm

Table 4.2 Simulation Result Comparisons

	MPG	increase	Total cost	decrease
ADVISOR model	48.4	0	$0.4884	0
New rule-based	80.66	66.65%	$0.3524	27.85%
DP	87	79.8%	$0.3382	30.75%

From Table 4.2, the updated rule based control algorithm is pretty close to the optimal results obtained by DP. However, this algorithm works fine enough for real time control, though some more constraints are needed.

In summary, the rules for finding optimal engine power are more complicated in series-parallel powertrains than in parallel ones as in [62]. This study shows that 2D lookup table achieves close-to-optimal performance in one particular drive cycle but may not be optimal in some others. Also, a more general algorithm should take SOC into consideration, and the optimal real-time algorithm can be better designed. This study has proven that this method (from DP to real-time algorithm) is admissible and effective if the rules in the look-up table are wisely designed.

4.7 Conclusions

Dynamic programming is a powerful tool to obtain global minimum of a cost function, which can be used to evaluate an existing control strategy for an HEV. In this chapter, the procedures of DP for three typical powertrain topologies are introduced in detail, and based on the flow charts the readers without a background in DP can implement it easily.

The case study on Prius model shows that the optimal fuel consumption results have some factors, such as frequent engine on/off switching and large battery discharge current,

that are detrimental to vehicle life. Hence the cost functions with consideration of SOH and engine on/off penalty is suggested. Even with the penalty terms, the potential of overall cost improvement over rule based strategy on UDDS drive cycle is still above 30%.

DP results can guide the development of rule based strategies for a given drive cycle. In this chapter a new rule-based control algorithm is proposed based on the optimal operating points determined from DP. The simulation results show a very close performance to the optimal case with only 3% difference.

Chapter 5 ECMS for PHEVs: Instantaneous Optimization

5.1 Introduction

Global optimization is not implementable in real time, and the rule based strategy that is optimized by optimal results from global optimization tools achieves quasi-optimal performance only on similar drive cycles. In this chapter, an instantaneous optimization method ECMS is introduced, which has been successfully applied to HEVs.

ECMS is based on Pontryagin's minimum principle [8], and it has a cost function in a form similar to Hamiltonian function in an optimal control problem. In its simplest form, the cost function can be defined as: $J = \dot{m}_{fuel} + \lambda * S\dot{O}C$ where $\dot{m}_{fuel}$ is the fuel consumption rate, $S\dot{O}C$ is battery state of charge variation rate and λ is the equivalent factor. ECMS is able to optimize this equivalent cost function in real time, and field tests have been performed on HEVs with ECMS [7]. The key to ECMS is to determine the equivalent factor λ in the cost function. For a PHEV, however, additional information of trip length is required because of a limited all electric range (AER). It is not difficult to understand that, even for a certain driving pattern (for example city pattern), the control strategy for a 5-mile trip is different than that for a 50-mile trip if AER is, say, 20 miles by this driving pattern. Also, the AER is a function of SOC, such that SOC is another important factor that decides the λ.

In this chapter, the principle of ECMS is introduced, the state of the art is reviewed, and a new method for generating a look-up table of λ for PHEVs with respect to remaining driving distance and SOC is discussed. Simulation results show the robustness of this map and substantial fuel economy improvement over multiple drive cycles.

5.2 Pontryagin's minimum principle and ECMS

ECMS is based on Pontryagin's minimum principle which is used in optimal control problems. In this section Pontryagin's minimum principle is introduced and then it is explained how ECMS is developed based on this principle.

5.2.1 Pontryagin's minimum principle

Pontryagin's minimum principle is developed by the Russian mathematician Lev Semenovich Pontryagin and then applied to optimal control problems [63]. The minimum principle provides a set of necessary conditions for optimality, and candidate optimal trajectories can be found by using the minimum principles [64].

For an optimal control problem as shown in (5.1), the Hamilton function can be written as in (5.2). To simplify the expression, the variables in the brackets are omitted, and the necessary conditions along the optimal trajectory can be found in (5.3) [65]. The equation (5.3-4) deals with the terminal conditions: 1) if t_f is free, then $\delta t_f \neq 0$ and terminal condition is $\left(H + \frac{\partial h}{\partial t}\right)|_{t_f} = 0$; 2) if x_f is free, then $\delta x_f \neq 0$ and terminal condition is $\left(\frac{\partial H}{\partial x} - p\right)|_{t_f} = 0$; 3) if both of them are free, then both conditions should be met; 4) if both of them are fixed, then no terminal condition should be met.

$$\min J = h\left(x\left(t_f\right), t_f\right) + \int_{t_0}^{t_f} g(x(t), u(t), t)dt$$

$$\text{subject to} \qquad \dot{x}(t) = f(x(t), u(t), t) \tag{5.1}$$

$$H(x(t), u(t), p(t), t) = g(x(t), u(t), t) + \lambda^T(t) \cdot f(x(t), u(t), t) \tag{5.2}$$

1) $\dot{\lambda} = -\frac{\partial H}{\partial x}$

2) $\frac{\partial H}{\partial u} = 0$

3) $\dot{x} = f$

4) $\left(\frac{\partial H}{\partial x} - \lambda\right)\delta x_f + \left(H + \frac{\partial h}{\partial t}\right)\delta t_f = 0$ (5.3)

5.2.2 ECMS for HEVs

ECMS solves an optimal control problem as in (5.1), and the goal is to find an optimal trajectory of control inputs $u^*(t)$ throughout the drive cycle to minimize the fuel cost. As there is no terminal cost, the term of $h\left(x\left(t_f\right), t_f\right)$ in (5.1) is 0.

If we take $SOC(t)$ as state $x(t)$, the fuel consumption rate $\dot{m}_{ice}$ as g function which is independent on $SOC(t)$, engine torque and speed as $u(t)$, and from Pontryagin's minimum principles, the Hamiltonian function can be expressed as

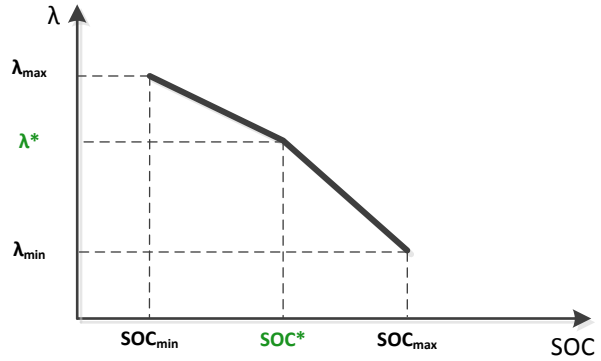
$$H(SOC(t), u(t), p(t), t) = \dot{m}_{ice}(u(t), t) + \lambda^T(t) \cdot \dot{SOC}(SOC(t), u(t), t) \quad (5.4)$$

subject to $SOC\left(t_f\right) = SOC_{desired}$ where $\lambda(t)$ is the co-state and can be taken as the equivalent factor that converts battery energy consumption to equivalent fuel consumption.

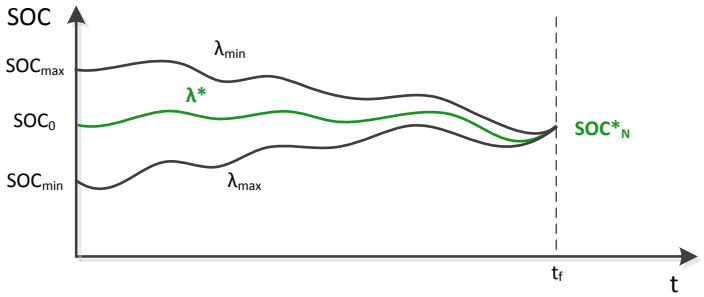
However, this method is not directly solvable by (5.3) as $\dot{m}_{ice}$ and $\dot{SOC}$ are highly nonlinear functions dependent on future power demands. Hence, some simplification is required. For an HEV, the SOC varies in a very small window, which is typically 0.45~0.55, such that $\dot{SOC}$ can be a function of only $u(t)$ and t. In this case, from (5.3-1)

$$\dot{\lambda}(t) = -\frac{\partial H}{\partial SOC} = -\frac{\partial \dot{m}_{ice}(u(t),t)}{\partial SOC} - \lambda^T(t)\frac{\partial \dot{SOC}(u(t),t)}{\partial SOC} = 0 \quad (5.5)$$

This is very important because the equivalent factor λ is then a time-invariant constant value. The authors in [12] tune the equivalent factor until $SOC(t_f) = SOC(t_0)$ for a given drive cycle. However, a constant λ can't regulate the SOC for an arbitrary drive cycle even if λ can be a function of SOC to increase the robustness [49] as show in Figure 5.1a), because the curve in Figure 5.1a) can be very different from one driving patterns to another.



a) Equivalent factor



b) Definition of λ_{min}, λ_{max}, and λ^*

Figure 5.1 Equivalent factor λ as a function of SOC

In Figure 5.1a), λ_{max} is the equivalent factor in the case that the drive cycle starts at $SOC_0 = SOC_{max}$ and ends at SOC_N^* while λ_{min} is the equivalent factor in the case that the drive cycle starts at $SOC_0 = SOC_{min}$ and ends at SOC_N^* , and λ^* is the equivalent factor in the case that $SOC_0 = SOC_N^*$, which is shown in Fig. 5.1b).On the other hand, this method is very effective for HEVs as long as the driving pattern can be estimated, and it can be implemented in real time with no requirement on the complete drive cycle. An adaptive ECMS has been proposed in [9] that tunes the λ based on driving pattern. When the λ is available, all $u(t) \in U$ are substituted to (5.4) and the minimum H at current step can be easily found, where U is the space of feasible control inputs at current step.

5.2.3 ECMS for PHEVs: equivalent factor

As stated above, the key to ECMS is to find an appropriate equivalent factor λ. The controllers for PHEVs cannot simply copy the successful implementation of ECMS for HEVs, because the λ is no longer a function of simply driving pattern. For a PHEV, the SOC varies in a large window which is typically 0.2~0.99, such that (5.5) does not hold. In addition, as discussed in Chapter 3 the trip length is another important factor that influences the λ.

To address this problem, the authors in [49] build a 3-D look-up table for λ by (5.6) ~ (5.8).

$$x_e = X_e \cdot (SOC - SOC_{min})/(SOC_{max} - SOC_{min}) \tag{5.6}$$

$$k = min(1, \frac{x_e}{x_r}) \tag{5.7}$$

$$\lambda = \lambda_e + \sqrt{1 - k^2}(\lambda_0 - \lambda_e) \tag{5.8}$$

where X_e is the AER when fully charged, x_e is the remaining AER, and x_r is the remaining driving distance to the destination where the vehicle can be recharged. By such method, $\lambda = \lambda_0$ holds at $x_e = 0$ ($SOC = SOC_{min}$), and $\lambda = \lambda_e$ when $x_e \geq x_r$. So we can consider that λ_0 is the equivalent factor in charge sustaining mode in which the remaining AER equals to 0 and can be calculated by (5.9) ~ (5.11), while λ_e is the equivalent factor for pure electric drive mode when the remaining AER is equal to or greater than the remaining driving distance which can be calculated by (5.12).

$$\lambda_{dis} = \frac{1}{\bar{\eta}_{dis} \cdot \bar{\eta}_{eng}} \tag{5.9}$$

$$\lambda_{chg} = \frac{\bar{\eta}_{chg}}{\bar{\eta}_{eng}} \tag{5.10}$$

$$\lambda_0 = \sqrt{\lambda_{dis} \cdot \lambda_{chg}} \tag{5.11}$$

$$\lambda_e = \frac{Pr_e/\eta_{g2v}}{Pr_g/H_{LHV}} \tag{5.12}$$

Table 5.1 Parameters for building the map of λ

Parameter	Value	Unit
$\bar{\eta}_{dis}, \bar{\eta}_{chg}$	0.9	-
$\bar{\eta}_{eng}$	0.3	-
Pr_e	0.15	$/kWh
η_{g2v}	0.95	-
Pr_g	3.3	$/gallon
H_{LHV}	63.7785 (44.4)	kWh/gallon (MJ/kg)

In (5.9) ~ (5.11), $\bar{\eta}_{dis}$ and $\bar{\eta}_{chg}$ are the average electric circuit efficiencies for discharge and charge, respectively; $\bar{\eta}_{eng}$ is the average efficiency of internal combustion engine. In

(5.12) Pr_e is the electricity price in \$/kWh, η_{g2v} is the efficiency of charging the battery pack by the grid, Pr_g is the gas price in \$/gallon, and H_{LHV} is the lower heating value of gas in kWh/gallon. If we consider the parameters in Table 5.1, the map of equivalent factor λ with respect to SOC and remaining driving distance x_r is shown in Fig. 5.2.

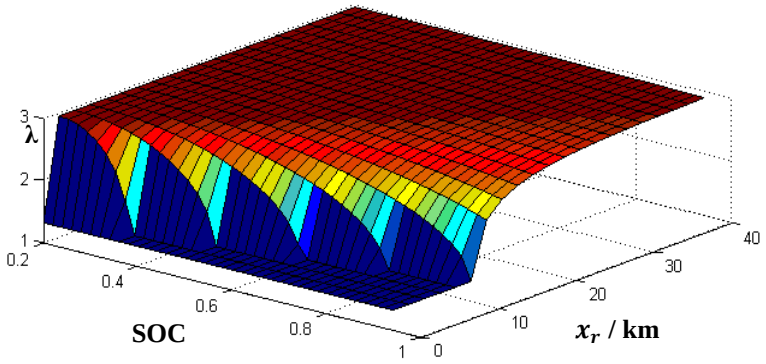


Figure 5.2 Equation based λ map with respect to SOC and x_r

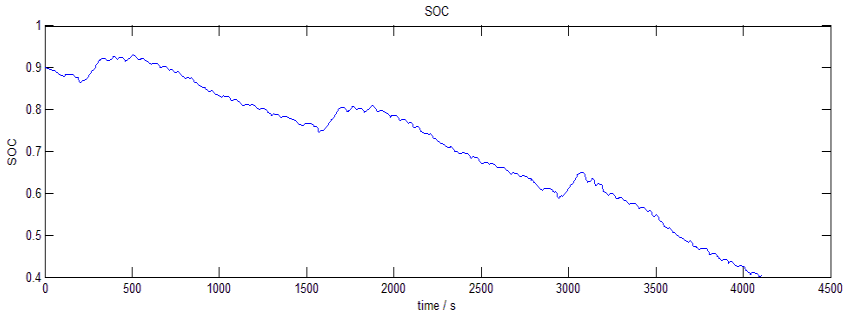
5.3 Case study

In this section the map of λ in Fig. 5.2 is examined by multiple drive cycles, and improvement of (5.6) is discussed. The vehicle model is still Toyota Prius, the same model as the model for case study in Chapter 4, and the only difference is the battery capacity which is increased to 8 Ah.

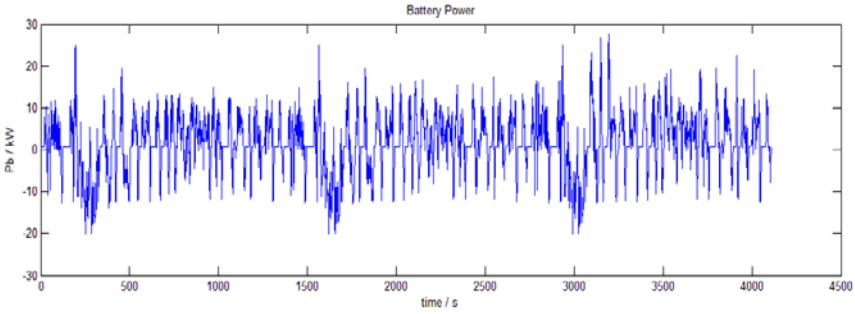
5.3.1 Performance test by the equation based λ map

If all the parameters in Table 5.1 are taken as constant to avoid the drive condition prediction, and suppose the remaining distance estimation is perfect, the simulation results for a triple-UDDS drive cycle are shown in Fig. 5.3. The fuel economy from ECMS is very

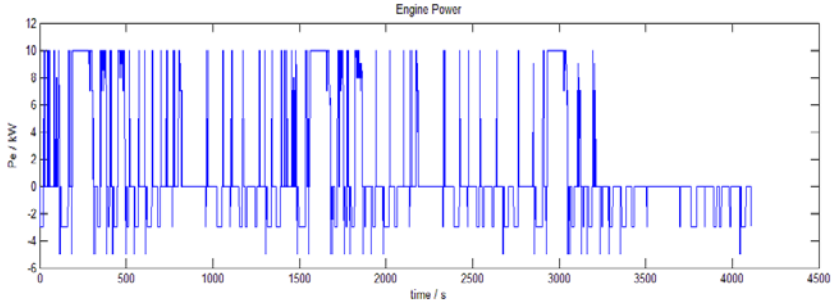
close to that by DP (4% gap) and much better than that by CD-CS control (43% improvement) shown in Fig. 5.4.



a) SOC

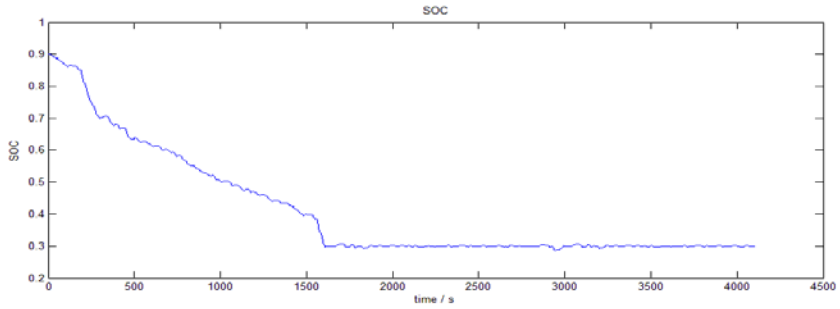


b) Battery Power

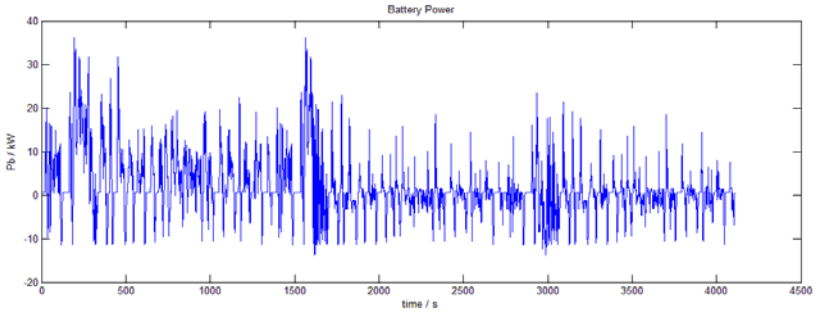


c) Engine power

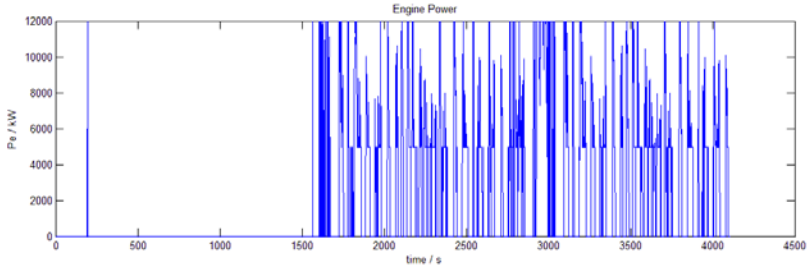
Figure 5.3 Simulation results by ECMS with λ map in Fig. 5.2



a) SOC



b) Battery Power



c) Engine Power

Figure 5.4 Simulation results by CD-CS strategy

5.3.2 Improvement of the equation based λ map

Although the performance is very good in triple-UDDS drive cycle, we have enough reason to doubt if this map works well in other drive cycles, too. The λ map in Fig. 5.2 is built based on constant values of parameters in Table 5.1, which is not the case in different drive cycles. To test the sensitivity of this map to different drive cycles,

UDDS+HWFET+UDDS (UHU) drive cycle is simulated. The simulation results show that SOC keeps increasing during highway drive, HWFET cycle, until it exceeds the higher limit. That's due to high equivalent factor values when the distance remaining is greater than AER while the engine efficiency is high on highway, such that to charge the battery costs less than to use the battery energy. From Fig. 5.2, when the remaining distance is large, λ doesn't decrease much as SOC increases. To avoid the case that SOC keeps increasing, the equivalent factor values at high SOC are brought down by (5.13), which is an improvement of (5.8).

$$\lambda = \lambda_e + (1-k)(\lambda_0 - \lambda_e)(SOC_{target}/SOC) \tag{5.13}$$

$$x_e = X_e \cdot (SOC - SOC_{target})/(SOC_{max} - SOC_{target}) \tag{5.14}$$

This equation brings down the λ values at high SOC, and $\lambda = \lambda_0$ still holds at $x_e = 0$ ($SOC = SOC_{target}$), but the average value of λ is brought down, which means the electric energy is favored to be used in priority. Another modification is the calculation of x_e as shown in (5.14). SOC_{min} that is set to 0.2 in (5.6) is replaced by SOC_{target} that is set to 0.3, which means in practical the SOC is actually allowed to be lower than SOC_{target} but the values of λ must be very high in this case to charge the battery back to SOC_{target}. This leads to negative values of k, and the new map of λ is shown in Fig. 5.5.

The simulation results are summarized in Table 5.2. Although the performance by the improved map of λ varies from drive cycle to drive cycle, the performance is still close to that by DP. What's more, this map shows great robustness by multiple drive cycles, and in none of the simulation the SOC is out of the range of [SOC_{min}, SOC_{max}].

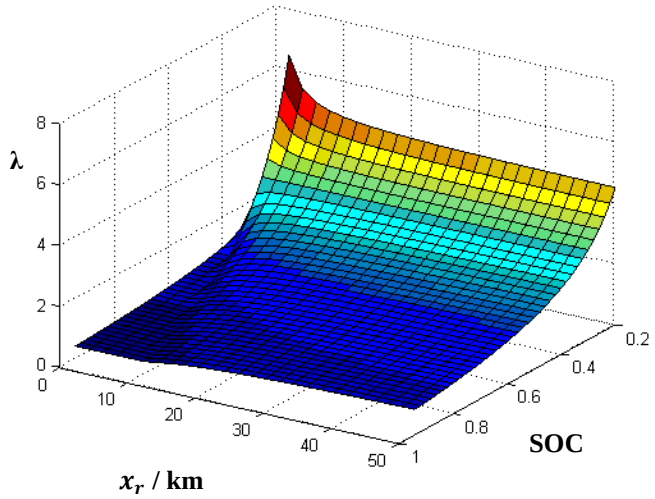


Figure 5.5 Improved λ map

Table 5.2 Summary of simulation results by improved ECMS

Cycle	**MPG increase compared to CD-CS**	
	Improved ECMS	*DP*
Triple UDDS	43.3%	48.5%
UHU	51.4%	68.1%
UUH	28.9%	39.4%
UHHU	39.8%	42.4%

5.4 Conclusions

This chapter presents a real-time control strategy for PHEVs by ECMS. The theoretical basis of ECMS, Pontryagin's minimization principles, is introduced. For ECMS implementation to HEVs, a key assumption is a small SOC window which is true in most

cases. However, this assumption does not hold for PHEVs, such that the way that ECMS is implemented to HEVs cannot be copied to PHEVs.

In this chapter, it is pointed out that a 3D map of equivalent factor λ is necessary and the map decides the performance and robustness of ECMS. Referring to previous works [49], the equivalent factor λ as a function of current SOC and remaining driving distance has been better designed with better robustness. If at the beginning of a trip the trip length is known, the fuel economy can be improved with our proposed algorithm. Multiple standard driving cycle tests have shown great improvement on fuel economy over the traditional CD-CS control, and the gap to the fuel economy obtained by DP is small.

Chapter 6 DP and ECMS combined strategy for PHEVs

6.1 Introduction

In the previous chapter a robust, real-time implementable ECMS with an improved λ map is developed. With this λ map, fuel economy can be significantly improved by ECMS over CD-CS strategy. However, the control algorithm has been tested on a limited set of drive cycles, which does not guarantee an improvement on fuel economy over arbitrary drive cycles.

The performance of ECMS with a λ map as PHEV control strategy entirely relies on the λ map. In this chapter, a systemic approach of generating a robust λ map using DP is proposed. Twelve standard drive cycles are used as a training set and a quasi-optimal λ map is built for each of the twelve cycles. Simulation results with quasi-optimal maps show only a 2%~6% gap in terms of equivalent cost compared to DP results. These twelve drive cycles are selected such that all the driving patterns (city, suburban, and highway) are included. Then, a universal λ map that can be used over arbitrary drive cycles is built based on the twelve quasi-optimal λ maps. With the universal λ map, a robust, real-time control strategy is achieved and the only external information of the trip is the trip length at the beginning of the trip which is commonly available. Simulation results with this universal λ map show a significant fuel economy improvement (18%-60%) over the 12 cycles in the training set. To further demonstrate the robustness of the proposed approach, we validate the convergence of the control strategy by testing our controller over real-world drive cycles generated by MATSim [66, 67] based on the driving log from 290 drivers in two German cities. 4719

drive cycles are simulated with arbitrary initial SOC and the results show a great robustness and an average of 9.3% cost reduction compared to CD-CS strategy.

6.2 Determining a Universal, Robust Equivalent Factor Map by DP

As introduced in Chapter 4, DP which guarantees an global optimal solution is well accepted as the optimization tool for an off-line simulation for (P)HEVs [4, 5, 68]. Although this technique cannot be implemented in real time due to heavy calculation burden and the requirement for a-priori knowledge of detailed future drive cycle, the global optimal solution can guide online rules or serve as benchmark to evaluate real-time strategies. Similar approach is followed in this chapter by using DP results to obtain a universal, robust λ map, which is built from all the quasi-optimal maps determined by DP.

6.2.1 DP using λ as control variable

Detailed DP procedures for different powertrain topologies are summarized in Chapter 4. In this section, DP is combined with ECMS, and the control variable is selected as the equivalent factor λ rather than torque/speed or engine power as in typical DP in Chapter 4. ECMS is based on Pontryagin's minimum principle and the detailed procedures can be found in Chapter 5. In brief, ECMS solves an instantaneous local optimization problem as shown in (6.1) where $\dot{m}_{ice}$ is fuel consumption rate of internal combustion engine, u is control parameter vector (engine torque T_e and engine speed ω_e in this chapter). Also as discussed in Chapter 4, penalty terms can be added. In this chapter the battery current penalty and engine on/off penalty are considered and the discrete expression can be found in (6.2). In (6.2) $\Delta Engine\_on/off$ is 1 if engine on/off status is changed from previous step and 0 if not. The variable k represents a discrete time step with a step size δ of 1 second in this chapter.

$$\arg\ min\, g(t) = \dot{m}_{ice}\big(t, u(t)\big) + \lambda(t)\dot{SOC}(t, SOC(t), u(t)) \tag{6.1}$$

$$\arg\ min\, g_k = \dot{m}_{ice}(u_k) + \lambda_k \dot{SOC}(SOC_k, u_k) + \alpha \cdot \Delta Engine\_on/off \tag{6.2}$$

Due to the limited storage capacity of the on-board batteries in a PHEV, the equivalent cost of electricity will change depending on the remaining driving distance d_R and battery SOC. The goal of this section is to determine the near-optimal value of λ with respect to d_R and SOC. To achieve this goal, we first use DP to determine the optimal value of λ over a number of drive cycles. We will then use these results to arrive at a generic λ map that provides near-optimal performance over any drive cycle.

In DP calculations, the equivalent factor λ is considered as the control variable, rather than engine operating point (T_e, ω_e) as in Chapter 4. The state variable is still selected as SOC. Finite number of λ values are considered, where $\lambda \in \Lambda = \{\lambda_{min}, \lambda_{min} + \Delta\lambda,\ \lambda_{min} + 2\Delta\lambda, \dots, \lambda_{max}\}$; likewise, $T_e \in \mathcal{T} = \{T_{e_{min}}, T_{e_{min}} + \Delta T_e,\ T_{e_{min}} + 2\Delta T_e, \dots, T_{e_{max}}\}$ and $\omega_e \in \Omega = \{\omega_{e_{min}}, \omega_{e_{min}} + \Delta\omega_e,\ \omega_{e_{min}} + 2\Delta\omega_e, \dots, \omega_{e_{max}}\}$ are considered, such that engine operating points belong to a finite discrete space $\mathcal{U} = \Omega^T \mathcal{T}$. Also, a finite discrete SOC space is created and the variable is expressed by SOC_m, $m \in \{1, 2, \dots, 10000\}$. SOC_m represents an SOC value from 0% (SOC_1) to 99.99% (SOC_{10000}) with an increment of 0.01%. Therefore, for any $\in \Lambda$, we can find an optimal, feasible engine operating point in $\mathcal{U}$ by solving the cost function (6.2) where $\dot{m}_{fuel}(k)$ is determined from the engine map and the $\dot{SOC}$ term can be calculated by combining (4.19), (4.21), and (6.3). The feasible engine operating points are those that can meet the power demand at the current step while maintaining the battery power

and SOC constraints. In this way, a one-to-one mapping from λ to $(\dot{m}_{fuel}, \dot{SOC})$ can be created for a given SOC value at each step of a drive cycle.

$$SOC_{k+1} = SOC_k - \frac{V_{oc} - \sqrt{V_{oc}^2 - 4(R_{int}+R_t)\cdot T_m \cdot \omega_m \cdot \eta_m^{-sgn(T_m)} - T_g \cdot \omega_g \cdot \eta_g^{-sgn(T_g)}}}{2(R_{int}+R_t)\cdot Q_b} \tag{6.3}$$

The process of finding optimal λ map by DP is shown in Fig. 6.1. Each black dot in Fig. 6.1 represents an SOC node, with a resolution of 0.01%. To clearly distinguish the SOC nodes at step k-1 from that at step k, what follows in this section uses SOC_m to represent one of all the SOC nodes at step k-1, and SOC_i to represent one of all the SOC nodes at step k. In Fig. 6.1 $J_k(SOC_i)$ is the minimum accumulated cost from step 0 to step k, $\in \{1, 2, 3, \dots, N\}$, on the node SOC_i, $i \in \{1, 2, \dots, 10000\}$. In this paper the cost is considered as fuel cost plus engine on/off penalty as described in Chapter 4.

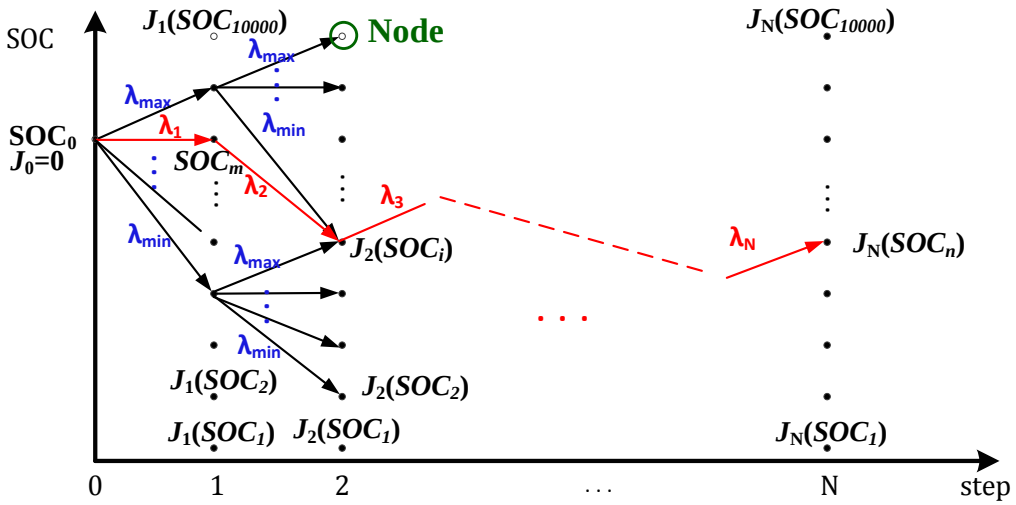


Figure 6.1 Find optimal λ at each step by DP

At the first step k = 1, DP starts with an initial SOC = SOC_0 and $J_0 = 0$. For each $\lambda \in \Lambda$, the corresponding $(\dot{m}_{fuel}, S\dot{O}C)$ are available from the one-to-one mapping method outlined above. $J_1(SOC_i)$ which represents the cost at step1 on the node $SOC_i = SOC_0 + S\dot{O}C \cdot \delta$ is updated by the fuel consumption such that $J_1(SOC_i) = \dot{m}_{fuel} \cdot \delta$. If more than one λ values lead to a same $S\dot{O}C$, then on the node of SOC_i the minimum fuel consumption value is retained.

At step k, k $\in \{2, 3, 4, \dots, N\}$, apply all $\lambda \in \Lambda$ respectively to each of the SOC nodes at step k-1, and each λ value on each SOC node SOC_m is mapped to unique $(\dot{m}_{fuel}, S\dot{O}C)$ values as outlined above. This $S\dot{O}C$ creates a path from SOC_m at step k-1 to a new node $SOC_i = SOC_m + S\dot{O}C \cdot \delta$, i $\in \{1, 2, \dots, 10000\}$ at step k, and the cost at current step as $g_{k-1}(SOC_m)$ in (6.4) is calculated as $g_{k-1}(S\dot{O}C_m) = m_{fuel} \cdot \delta$. One or more paths from all the nodes at step k-1 will find a same SOC_i node at step k as in Fig. 6.1, and the one with minimum total cost J_k calculated by (6.4) is picked up as the optimal path from step k-1 to the node SOC_i at step k, as expressed by the red arrow in Fig. 6.1. After all $\lambda \in \Lambda$ being applied respectively to all the SOC nodes in step k-1 which means all the possible paths to the node SOC_i have been investigated, on the node SOC_i in Fig. 6.1 the following information is retained: the previous step SOC node SOC_m on the optimal path (red arrow), accumulated minimum cost $J_k(SOC_i)$, and optimal λ (λ_2 in Fig 6.1) on the optimal path, and engine on/off status. In this way, the J_k on each node, which compares the cost on all possible paths, is guaranteed to be the minimum accumulated cost by step k. This algorithm is repeated until the final step N. Finally for each node at step N, an optimal path with the

minimum cost all the way to SOC_0 can be retrieved step by step backward, as indicated by the red line in Fig. 6.1. The λ values on this path at each step constitute an optimal equivalent factor set.

$$J_k(SOC_m) = arg\ min_{i\epsilon 1,2,..,M}\ g_{k-1}(SOC_i) + J_{k-1}(SOC_i) + \alpha \cdot \Delta Engine\ on/off \quad (6.4)$$

6.2.2 Build quasi-optimal λ map for a given drive cycle

In this section a quasi-optimal λ map is established from the DP results over a given drive cycle. The goal is to find a unique λ value for each given ΔSOC and remaining driving distance d_R and establish an s-map which represents the chosen drive cycle. ΔSOC is defined in (6.5) where SOC* is the target final SOC at the end of a trip.

$$\Delta SOC = SOC - SOC^* \quad (6.5)$$

ΔSOC is considered instead of SOC in this paper, because the target final SOC may vary from case to case. With a given drive cycle, it is not difficult to find the step index k_d at which the driving distance is d km from the first step, $d \in \{1, 2, \dots, M\}$ where M is the round-down length of the entire given drive cycle. For example if at step k= 1000, d = 5km, then k_5 = 1000. This d can also be taken as the remaining driving distance d_R at the beginning of the drive cycle if the drive cycle ends at k_d. At each index k_d, for each ΔSOC value, an optimal SOC trajectory can be found step by step backward to the original point (k = 0, SOC_0 = 90%) by retained DP results as shown in Fig. 6.2, and the optimal λ values on each trajectory can also be retrieved from the DP results. Fig. 6.3 gives an example of optimal λ values (blues line) on an optimal SOC trajectory with $\Delta SOC = 60\%$ and d_R = 35km over NYCC drive cycle. As shown in Fig. 6.3, the optimal λ values on the optimal SOC trajectory will vary

substantially due to the power demand, which guarantees the minimal cost. Also seen in Fig. 6.3, the trend is for λ to reduce as the distance to destination reduces. To catch this trend and increase the accuracy of the λ map, the λ value on a certain node (d_R , ΔSOC) is calculated using linear regression from the optimal λ values on the corresponding optimal trajectory. For example, if the linear regression model on a node is $\hat{\lambda} = a \cdot n + b$ where n is the step variable, then $\hat{\lambda}|_{n=0}$ is the estimated λ value for that node as shown in Fig. 6.3. In Fig. 6.3, the red line is the regression model, and the green circle shows the estimated λ value for the node ($d_R = 35km$, ΔSOC =60%).

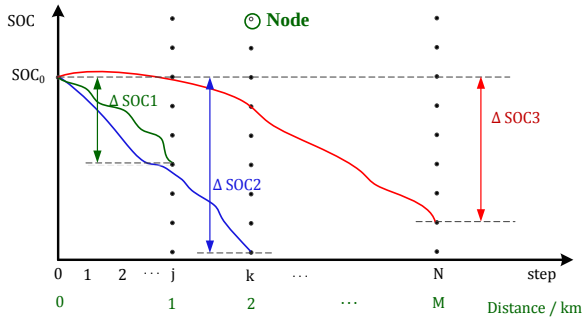


Figure 6.2 The optimal SOC trajectories from a given node to the original point

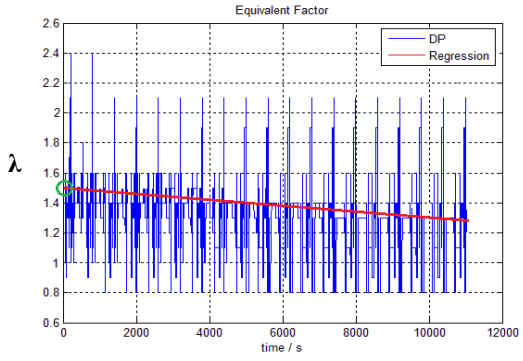


Figure 6.3 Find λ from the λ values on the optimal trajectory

In such a way, one λ value can be found for each (d_R , ΔSOC) node, and λ values for all the nodes finally constitute a quasi-optimal λ-map for each of the drive cycles. Some examples of λ maps for different drive cycles are shown in Fig. 6.4, where SOC* is set to 30%. When SOC > SOC*, λ increases as distance increases. However, when SOC < SOC*, λ increases as distance decreases to encourage the engine to charge the battery to SOC* sooner.

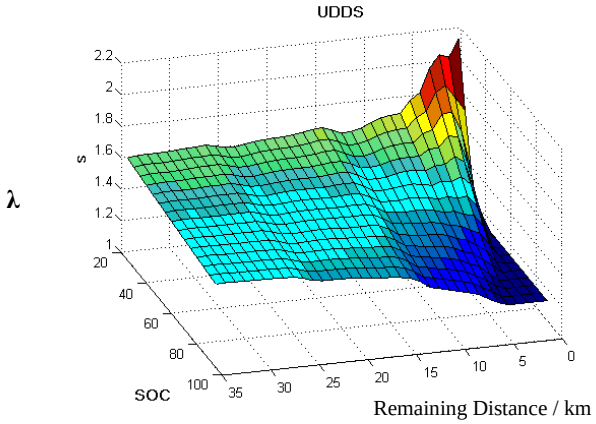


Figure 6.4 Quasi-optimal λ map obtained by DP for UDDS drive cycles

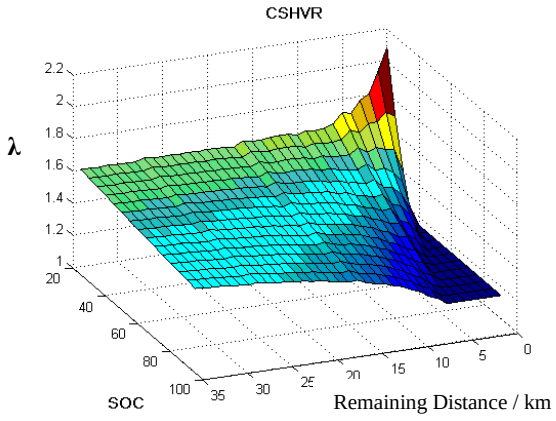


Figure 6.5 Quasi-optimal λ map obtained by DP for CSHVR drive cycles

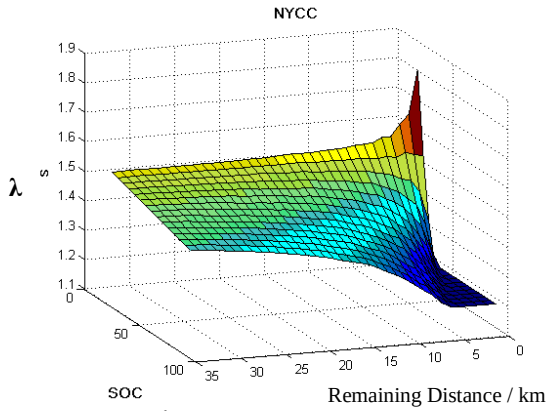


Figure 6.6 Quasi-optimal λ map obtained by DP for NYCC drive cycles

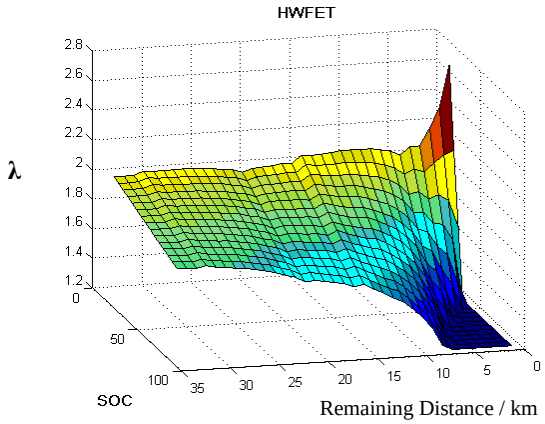


Figure 6.7 Quasi-optimal λ map obtained by DP for HWFET drive cycles

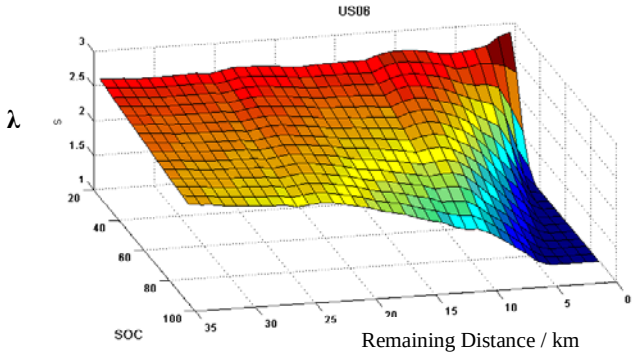


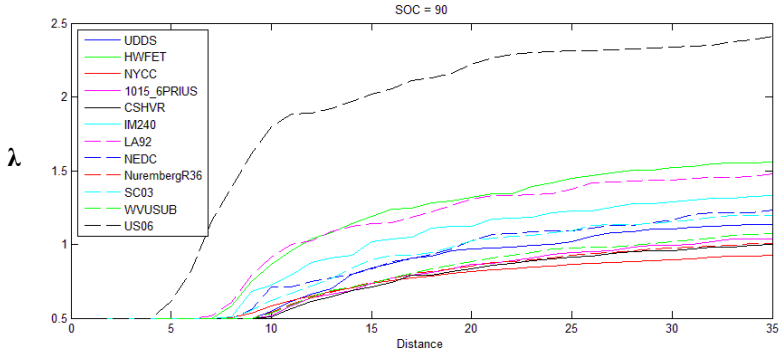
Figure 6.8 Quasi-optimal λ map obtained by DP for US06 drive cycles

6.2.3 Map analysis

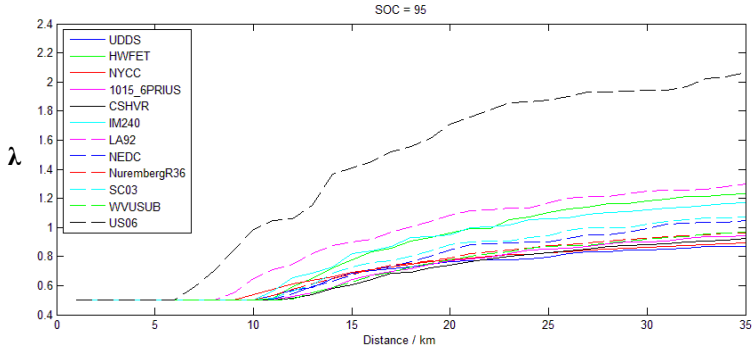
In total, twelve maps are generated for all the drive cycles that listed in Table 6.1. These cycles include all driving patterns: city, suburban, and highway. At a fixed SOC level, the factor λ is a function of only remaining driving distance. Fig. 6.5 shows examples when SOC = 90% and SOC = 95%. From the figures, the equivalent factor λ remains minimal for all-electric drive and increases as the remaining diving distance to the destination increases, which means the closer to the destination is, the more electric energy usage is favored. It is noticed that the pure electric range varies significantly from cycle to cycle. It is up to 10km for CSHVR cycle and only 4 km for US06 cycle. Note that a short all electric range (AER) is intentionally chosen by scaling down the PHEV battery pack described in Chapter 5 to reduce the total distance of simulation. However, the trend remains the same: when the distance is between AER and 3*AER, the λ is very sensitive to the distance; if the distance is more than 3*AER, λ increases slowly and converges to the λ value for CS mode in this drive cycle, which is a function of driving pattern.

6.2.4 Build a universal λ map for all cycles

From Fig. 6.5, it is clear that driving pattern is still significant for an optimal λ value even with a known distance to destination. Ideally, if the future driving pattern (for example, city or highway) is known or accurately predictable, a corresponding map can be used for real time control. Since the map is generated from DP which guarantees optimality, the performance can be expected to be close to the global optimal solution. However, even if the driving pattern cannot be predicted, it is still possible to develop a robust control strategy with a universal λ map.



a) At SOC = 90%



b) At SOC = 95%

Figure 6.9 Average equivalent factor λ as a function of distance

The twelve λ maps are utilized as the training set to generate a universal λ map. To ensure a stable SOC range, at high SOC end a small λ value is preferred, such that SOC never exceeds the upper limit for all the cycles. In the same way, at low SOC a larger λ value is considered to make sure the SOC is never below lower limit even with the most aggressive driving pattern. At each SOC level, a figure of λ as a function of distance can be plotted as in Fig. 6.5, and each λ curve represents a drive cycle. When $SOC = 85\sim95\%$ the λ curves with smallest values is selected (for example, UDDS curve in Fig. 6.5 b) and NYCC in Fig. 6.5 a)), and at $SOC = 25\sim35\%$ the λ curves with largest values (US06) is selected. In

between, for every 5% SOC increment we pick a λ curve with smaller values. For example, at SOC = 40% we pick the second largest value curve, at SOC = 45% the third largest, at SOC = 50% the fourth largest, … , and at SOC = 80% we pick the 11^{th} largest value curve which is the second smallest curve. The map works as a look-up table, which is shown in Fig. 6.6. The performance of the generated map is dependent on the standard drive cycles that are selected. To make this map more representative, the standard drive cycles that are selected should cover all driving patterns, and in this thesis 4 city drive cycles, 6 suburban drive cycles, and 2 highway drive cycles are selected.

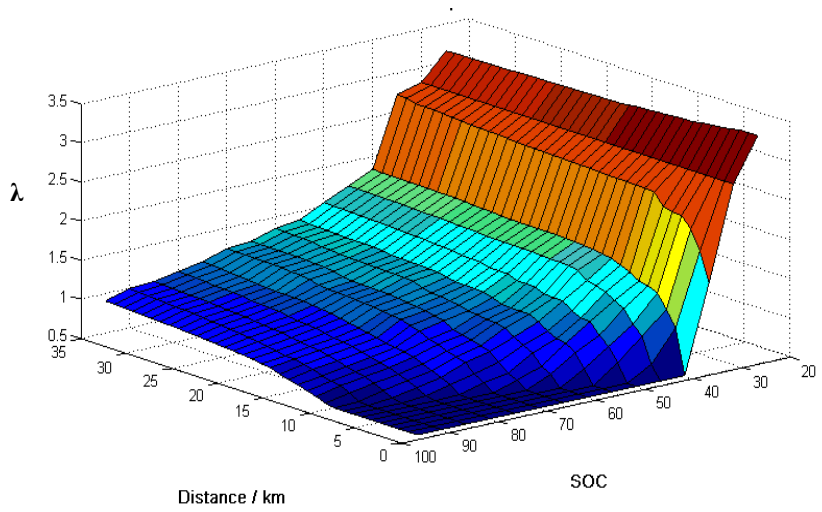


Figure 6.10 Universal λ map for the proposed robust control strategy

With such map available and the distance to the destination known at the beginning of the trip, ECMS can be implemented in real time. At each step, the controller calculates the distance travelled and distance remaining, and then the equivalent factor λ can be found by

the remaining distance and current SOC. Then the optimal engine power and battery power combination can be calculated by solving (6.2).

6.3 Simulation Results

6.3.1 Simulation results from standard drive cycles

The simulation results by using the proposed universal λ map over the twelve standard drive cycles are shown in Table 6.1. Because the final SOC values are different between drive cycles, equivalent cost in dollars rather than MPG is calculated by converting both the fuel consumption and electric energy consumption into cost as described in Chapter 4. As the powertrain is identical to the Prius model in ADVISOR, the results are compared to those by ADVISOR which we consider as the performance by the-state-of-the-art CD-CS control strategy. The only changes that are made in ADVISOR are the battery capacity, initial and target SOC, which are set to the same values as the proposed strategy.

The simulation results are shown in Table 6.1 and Fig. 6.7. The improvements and gaps are calculated by (6.5) and (6.6), where $COST_{CD-CS}$ is the equivalent cost by CD-CS strategy, $COST_{ECMS}$ by ECMS, and $COST_{DP}$ by DP. From the results, 16% ~ 38% improvement over CD-CS strategy can be achieved for all of the drive cycles by the universal λ map in Fig. 6.6. In addition, each of the simulations ends up with a stable SOC level and the over charge or discharge of battery is avoided. If the corresponding optimal λ map is used, the cost can be further reduced, which is also shown in Table 6.1. From the results in Table 6.1, additional 7% ~14% improvement over ECMS using universal λ map can be achieved by using the optimal λ maps, and the gap to DP results is only 4.6% on average. Fig. 6.8 shows examples of the

SOC trajectories by ECMS with optimal λ map and by DP, and a very similar battery usage pattern can be observed.

$$Improvement = \frac{COST_{CD-CS} - COST_{ECMS}}{COST_{CD-CS}} \tag{6.5}$$

$$Gap = \frac{COST_{DP} - COST_{ECMS}}{COST_{CD-CS}} \tag{6.6}$$

Table 6.1 Simulation results and comparison

drive cycle	Universal λ Map		Quasi-optimal λ map	
	Improvement over CD-CS strategy	Gap to DP results	Improvement over CD-CS strategy	Gap to DP results
UDDS × 3	31.49%	-9.89%	38.67%	-2.71%
HWFET × 3	15.83%	-15.07%	25.74%	-5.16%
NYCC × 21	37.58%	-11.96%	45.67%	-3.87%
1015_6PRIUS × 9	35.40%	-13.31%	43.06%	-5.65%
CSHVR × 4	35.02%	-14.27%	43.69%	-5.60%
CYC_IM240 × 13	23.61%	-14.43%	34.03%	-4.01%
LA92 × 3	26.79%	-16.57%	40.87%	-2.49%
NEDC × 4	29.58%	-12.66%	39.00%	-3.24%
NurembergR36 × 9	35.53%	-12.77%	43.21%	-5.10%
SC03× 7	28.37%	-16.50%	38.89%	-5.98%
WVUSUB × 4	32.29%	-13.07%	40.06%	-5.30%
US06 × 3	25.40%	-15.01%	34.20%	-6.21%

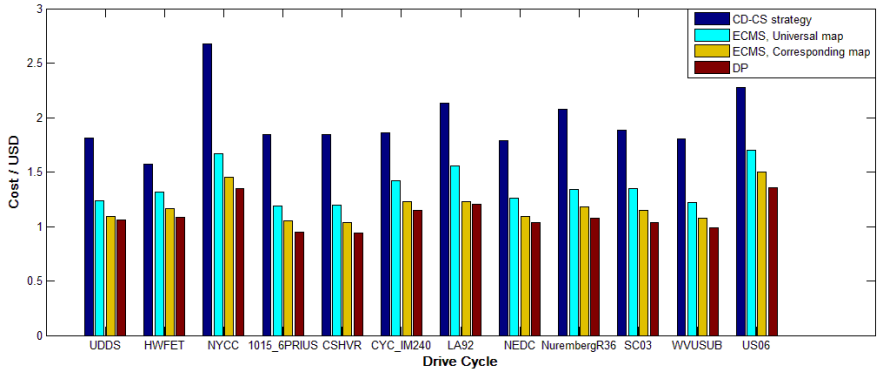
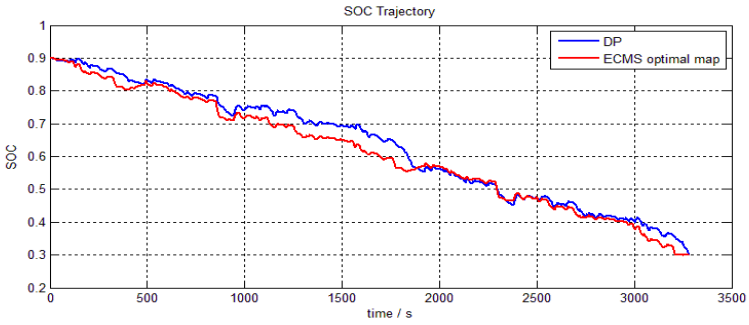
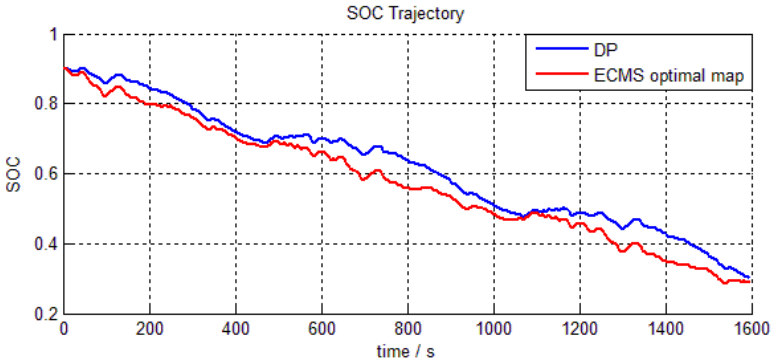


Figure 6.11 Cost comparison by different strategies over multiple drive cycles



a) LA92 drive cycle



b) US06

Figure 6.12 Trajectories of SOC on LA92 cycle and US06 cycle

6.3.2 Simulation results on real-world drive cycles

The robustness and performance are further examined by the drive cycles from real-world driving missions. The recourse is from Institute for Transport Planning and Systems at ETH, Switzerland. A driving diary was completed by more than 300 drivers in two German cities: Karlsruhe and Halle. This diary recorded the driving information over 6 weeks and finally 290 drivers reported meaningful driving information. The information that is of interest in

this study includes the departure location (GPS coordinates), arrival location, and duration of each trip. Then MATSim [67] is utilized to find out the drive cycle of each trip.

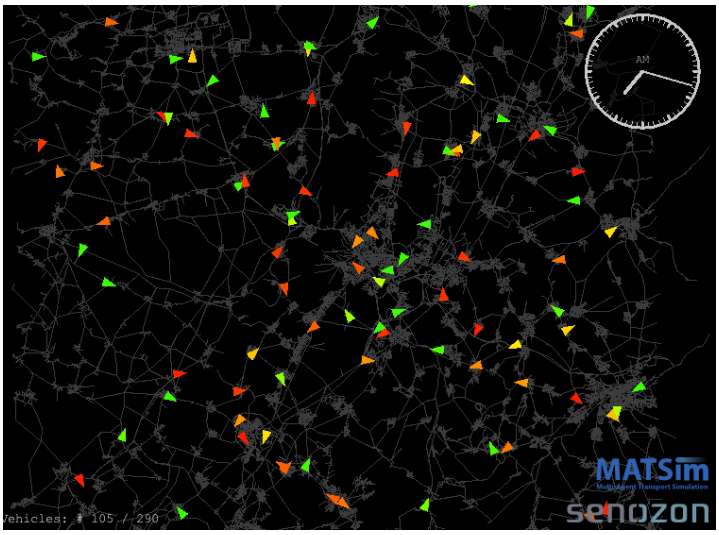


Figure 6.13 Screenshot of agent-based simulation by MATSim

MATSim is an open-source large-scale agent-based transportation simulation tool, which is shown in Fig. 6.9. In the figure, each line represents a link (road) and each arrow represents a vehicle, and the color shows the vehicle speed. The software tries to find out the most efficient path for each driver with a travelling plan in terms of departure location, departure time and destination. However, drive cycle is not the focus of this macro-simulation tool and it simulates the vehicle speed in a pretty simple way: if the number of vehicles on a certain road does not exceed the capacity limit, each vehicle follows the speed limit of this road; otherwise the speed of each vehicle will reduce linearly as the number of vehicles increases. Also, a vehicle will not reduce speed or stop at intersections.

To generate meaningful drive cycles, some additional features have been added to the open-source software. First, at each intersection the vehicle is forced to wait to enter the next link. The waiting time follows normal distribution with $\mu_t = 0$ and $\sigma_t = 15s$, which means in 99.7% of the cases the waiting time is within 45 seconds at traffic lights. If the waiting time is equal or smaller than 0 the vehicle does not decelerate or stop. This assumption is based on a small town traffic condition, and this parameter should be tuned for metropolitan cities with a larger σ_t value. Second, the driving speed also follows normal distribution with $\mu_v = \text{speed limit} \text{and } \sigma_v = 5\% \times \text{speed limit}$, such that on a link with speed limit of 65mph 99.7% of the generated drive cycles have the average speed in range of 55~75mph. In addition, 1/5 Hz random disturbance is added such that the vehicle doesn't travel with constant speed within a same link. Third, the maximum acceleration (deceleration) between two successive links follows normal distribution, too. This parameter actually determines the driver's driving style: calm, normal or aggressive. From the statistics of standard drive cycles in the US, Europe, and Japan, and also from the real-world drive cycles collected in [69], $\mu_{a_{max}} = 2\ \mathrm{m/s^2}$ and $\sigma_{a_{max}} = 0.5\ \mathrm{m/s^2}$ are considered. Note that the distribution is for maximum acceleration a_{max} that is identical for all the trips from a same driver. For each individal trip, each acceleration mission follows Extreme Value Distribution [70] with $\mu_a = 0.9 \times a_{max}$ and $\sigma_a = 0.05 \times a_{max}$. The probability density function is shown in Fig. 6.10 with $a_{max} = 2\ \mathrm{m/s^2}$. At high speed, however, a maximum acceleration does not make sense as the transmission ratio is small, such that the actual acceleration decreases linearly until 0 at 150kmph.

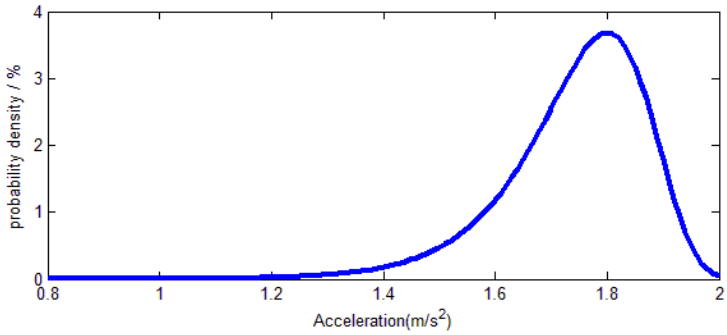


Figure 6.14 Probability density function of extreme value distribution

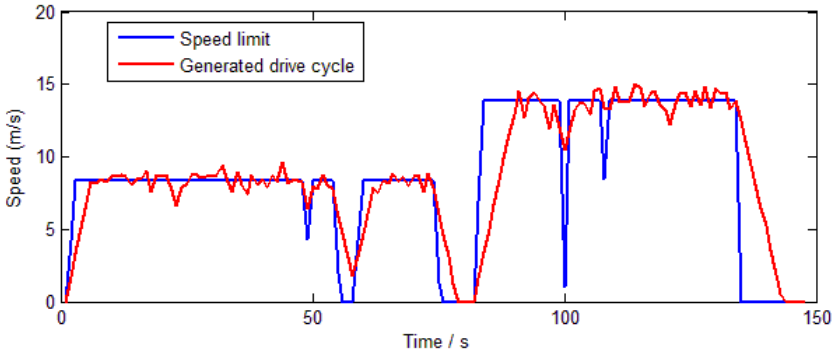


Figure 6.15 Example of a section of a generated drive cycle

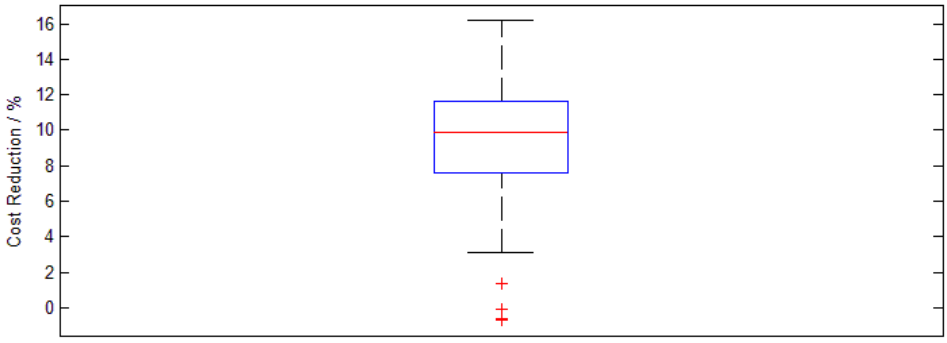


Figure 6.16 Box plot of the cost reduction rate for all drivers

With such a method, in total 24921 drive cycles have been generated and a section of one of the drive cycles can be found in Fig. 6.11 as an example. To reduce the simulation time, the maximum number of drive cycles from each single driver is limited to 50, and the drive cycles with length smaller than 2km are discarded. Finally, 4719 drive cycles are simulated by ECMS with the λ map in Fig. 6.6. For each simulation, an initial SOC is randomly selected in the range of 0.3~0.9 such that all the conditions can be included, and trip range is assumed a known parameter for each simulation. Simulation results by proposed DP-ECMS combined strategy and by ADVISOR's strategy are compared in terms of cost, and the statistics are shown in Fig. 6.12. Note that for the drive cycles within the AER, CD-CS strategy in ADVISOR may achieve lower cost because the vehicle always operates in pure electric mode, while the proposed strategy limits the battery current as suggested in (6.2). However, an average cost reduction of 9.3% can still be expected from all the 264 drivers. 99.24% of the drivers (262 out of 264) end up with an improved fuel economy, and the median is 9.8% which means 50% of the drivers can improve it by at least 9.8%. In addition, in all the simulations, the proposed strategy holds a steady SOC level and SOC is never above 95% or below 25%, which shows great robustness of this algorithm.

6.4 Conclusions

This paper presents a robust, real-time control strategy for a plug-in hybrid vehicle. A unique λ-map is generated from twelve quasi-optimal λ maps obtained by DP, and this universal λ map serves as a look-up table with the inputs of SOC and remaining driving distance to the destination and the output is a λ value which is fed to the cost function of ECMS. Detailed procedures for generating this universal λ map are introduced in this paper

and can be applied to different PHEV models. The only necessary information at the beginning of a trip is the distance to the destination, which is often readily available. Simulation results show 16% to 38% cost reduction compared to state-of-the-art CD-CS strategy over the twelve drive cycles in the training set, and a potential of additional 7%~14% cost reduction can be achieved if a corresponding quasi-optimal λ map is used. In the second part of this paper, a novel real-world drive cycle generation method by MATSim is introduced and 24921 drive cycles are generated based on a driving diary by more than 300 drivers. 4719 of those drive cycles are simulated and results show a great robustness of the universal λ map and an average of 9.3% cost reduction. It is noted that the quasi-optimal λ maps of driving cycles belonging to a same driving pattern (such as city, highway) are similar, which means if driving pattern can also be predicted accurately, the fuel economy can be further improved by using corresponding λ map for such driving pattern.

Chapter 7 Destination Estimation based on Driving History

7.1 Introduction

As outlined in the preceding Chapter, the main challenge to the implementation of a robust, real-time control strategy to PHEVs is the need for trip length information. In Chapter 6 all the simulation results are obtained by assuming a known trip length to determine the appropriate equivalent factor λ, which is not always the case.

Many have looked at algorithms for predicting the vehicle destination, given limited knowledge of the vehicle past behavior. In [53] the authors divide a map of a city into 1600 (40 by 40) cells with each side of 1km, and the cell of destination is predicted by a probability distribution based on Bayes' Rule. Bayes' Rule is combined with maximum entropy technique in [54] to improve the prediction accuracy, and the probability distribution of the routes taken to each destination is required in the model. Network technique applied to destination prediction can be found in [56, 71] in which a hierarchical dynamic Bayesian network and a hybrid dynamic mixed network are introduced, respectively. In Project Lachesis [51], the authors present a clustering algorithm and the probability of transitions between destinations is predicted with a hidden Markov model (HMM) which includes both position and time-of-day information. A time-independent Markov model is utilized in [52] and the order is discussed. By order the authors mean the number of previous destinations to be used to predict the next one, and 2nd order is implemented in [52] with 13 candidates of destination. However, higher order leads to highly sparse matrix for Markov model with more candidates of destination, such that the prediction is highly data driven. Additional

information such as time of day, day of week, route being taken, and vehicle velocity can be included in a network, and the algorithm is hence more complex. For example, the learning progress in [56] is off-line, requires data from the entire vehicle population, and takes up to 5 days to calculate. As expected, more information leads to more accurate prediction and heavier computation burden.

The goal of this chapter is to develop a destination estimation algorithm without the destination directly indicated by the driver is proposed. The algorithm should be able to collect the coordinates where the vehicle parks in sequence, and cluster the collected data into a destination transition matrix. As real-time feature is essential, this study begins with 1st order Markov model to establish the probabilities of transition from one destination to another, such that the least information is required. The transition probabilities are used only as the initial values when the vehicle starts a trip with low fidelity, such that it is crucial to adjust the probabilities as the vehicle moves through the transportation network to the destination. In this chapter distance variation based method is considered with a simple assumption that the driver will drive the vehicle closer and closer to the destination. However, if the vehicle is going to a new destination, the estimation error may be large, and a contingency algorithm should be developed to ensure that the prediction information is not used.

7.2 Markov Model based Destination Estimation

7.2.1 Transition probability matrix for relevant destinations

According to 2009 national household travel survey [50], more than 50% of the trips are made for work, school, church, social and recreation that are of relatively fixed destinations.

The remaining 40% of trips are for personal errands, such as dinning and shopping, that may also be at fixed destinations. As a result, we can postulate that at least 60%~70% destinations will be visited multiple times. If the destinations of interest are recognized and stored in the memory, the probability distribution for the next destination is available given the current location. We first define the terminology used in defining a destination to help formalize this insight.

Location – physical coordinates where the car is located

Stop – the event that the car stays static for a time interval

Cluster – a group of stops indicating the same destination

Destination center - the Euclidean center of a cluster

If we consider the destinations occur in series as a Markov process [72], which means the estimation of the next destination is based only on current location, the conditional probability can be expressed as

$$P(X_i = x_i | X_{i-1} = x_{i-1},\ X_{i-2} = x_{i-2}, \cdots,\ X_1 = x_1) = P(X_i = x_i |\ X_{i-1} = x_{i-1}) \qquad (7.1)$$

where X_i denotes the i[th] destination that has been recorded and x_i represents the coordinates of one of the possible destinations. The probability distribution can be generated by a hidden Markov model (HMM) [30]

$$P(D = x_j | C = x_i\ ) = p_{ij} \qquad (7.2)$$

where D represents destination location, C current location, p_{ij} the probability of the transition from x_i to x_j , and $\sum_j p_{ij} = 1$. p_{ij} is calculated by the observation data as

$$p_{ij} = \frac{m_{ij}}{m_i} \qquad (7.3)$$

where m_{ij} is the number of times that the trips start from x_i and end at x_j, and m_i is the total number of times that the trips start from x_i. A matrix of transition probability among all the destinations can then be established.

7.2.2 Data collecting and processing

The raw data is collected by GPS logger *LandAirSea 3100* from 6 drivers for 3 months. A threshold of stopping interval is required to determine a stop. In [52] the authors show that smaller threshold leads to more stops. In this study a threshold of 10 minutes is decided. In other words, whenever an interval of 10 minutes with speed less than 3 mph was detected, we take it as a stop. This threshold guarantees that temporary stops, such as stops at traffic lights or stop signs, will not be considered as destinations. Then the first coordinates of the data in the interval was stored as the coordinates of the corresponding stop. Note that a speed of less than 3 mph rather than exact 0 is considered here to eliminate the GPS signal noise.

After N stops are detected and stored, an initial transition probability matrix mentioned in section 7.2.1 is built. The value of N is chosen in such way that the resulting probability matrix is statistically significant and the initialization time is limited. N = 50 is considered as the threshold in this chapter, which will be better designed by statistical significance study in the future. As the car is not parked exactly at the same location even for a same destination, a clustering algorithm is then necessary. The relationship between the cluster radius and the number of clusters is discussed in [52] and in this study a threshold of 300m is adopted as the cluster radius. The destination center is calculated as the centroid of the stops in that cluster, and the coordinates are calculated by squared Euclidean distances as shown in Fig. 7.1. The red dots are the actual locations of stops (the bigger the dot, the more times the car is parked

there), the yellow dot stands for the destination center which may not be a feasible parking position but will be used for distance calculation, and the purple dot is the office building.

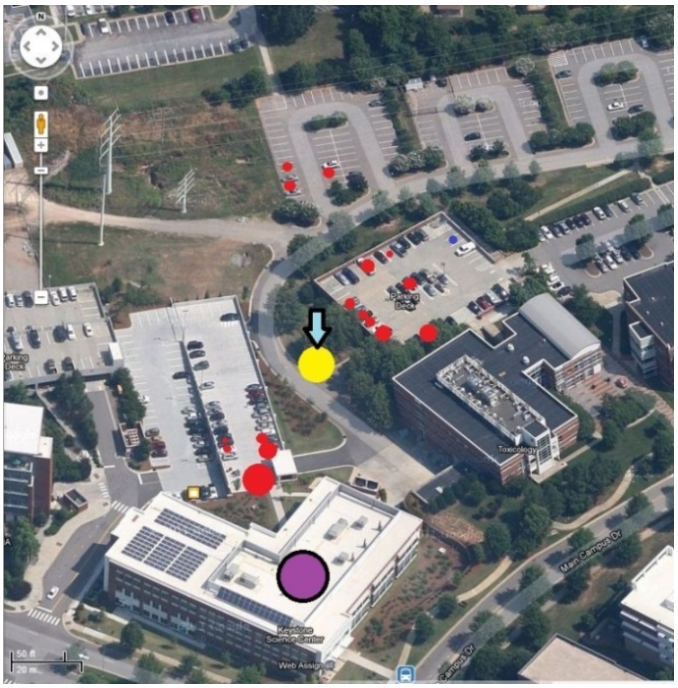


Figure 7.1 The centroid of a cluster (yellow dot) of stops (red dots)

The initial 50 stops are clustered and the destination centers are recorded as $x_1,\ x_2, \dots, x_n$. If a new stop at x_{new} is detected, the distance between x_{new} and $x_1,\ x_2, \dots, x_n$ is respectively compared to the threshold of 300m. If the distance between x_{new} and $x_i, i \in 1, 2, \dots, n$ is smaller than the threshold, x_{new} is clustered into the ith cluster, and the center of ith destination is updated; otherwise, if none of the distances is within the threshold, x_{new} is considered as the center of the (n+1)th destination x_{n+1}.

7.2.3 Forgetfulness algorithm

To account for the fact that people's habits change, and that the vehicle driver may change, there is a need to discard the destination that are no longer relevant. Therefore, a forgetfulness algorithm is a good supplement to destination prediction, which improves the accuracy of the prediction and saves memory. Two forgetfulness algorithms are considered in this study, time-based and dimension-based.

In a time-based algorithm only the destinations that have been visited within a time threshold are utilized for prediction. This algorithm is straight forward and in accordance with the way people forget things. However, the optimal time threshold is difficult to define: longer time thresholds may keep destinations that are no longer visited for long time, while shorter time thresholds may lead to data loss if the vehicle is not in use for a period. In addition, the dimension of destination matrix is uncontrollable.

In a dimension-based algorithm a fixed number of stops are stored in memory on a first-in first-out basis. For example, M stops $y_1, y_2, \dots, y_M$ are stored in memory in chronological order for prediction. When a new stop y_{M+1} is detected, the first stop y_1 is discarded and the prediction is then based on $y_2, y_3, \dots, y_{M+1}$. The center(s) of the destination cluster(s) to which y_1 and y_{M+1} belong is (are) updated. This algorithm limits the maximum dimension of the relevant destination matrix mentioned in Section 7.2.2, and the dimension is reduced if some stops are clustered as the same destination. Also, this algorithm records only the sequence of stops regardless of the time of each stop, which is more programming-friendly.

Based on the considerations noted above, we chose to use the dimension-based forgetfulness algorithm. According to 2009 National Household Travel Survey [50], the

average number of person trips per day is 3.79 (around 110 per month). Therefore, the consideration of past 110 stops should give statistically significant samples while modeling for changing habits.

7.3 Distance Estimation with Dynamic Probability Updating

7.3.1 Distance prediction from probabilities

If the trip begins from x_i, we consider a simple mean to calculate the remaining distance of the trip as

$$d_p = \sum_{j=1}^{m} p_j d_{cj}$$

$$\text{s.t.} \quad \sum_j p_j = 1 \tag{7.4}$$

where d_p is the predicted distance to the destination given current position $x_{current}$, m is the amount of destinations that have been visited from x_i, p_j is the probability that the final destination position is x_j, and d_{cj} is the linear distance between $x_{current}$ and destination x_j. At the beginning of a trip, $x_{current}$ equals to x_i. The outcome of this equation d_p will be used by the control strategy.

7.3.2 Dynamic updating of probabilities during vehicle mission

With the probability distribution generated by (7.3), and with the vehicle starting at a location pre-stored as the last destination location x_i, the next likely destinations can be determined. However, as the vehicle leaves x_i, the likeliness of reaching a specific destination changes. For example, if the vehicle is moving away from a specific destination, the probability of arriving there is decreased, and vice versa. To capture this fact, we dynamically update the probability of reaching a destination x_j as follows:

$$p_j = p_{ij}\left(\frac{d_{ij}}{d_{cj}}\right)^n \tag{7.5}$$

where d_{ij} is the linear distance between initial position x_i and possible destination x_j, and n is a coefficient allowing the probability update to converge faster to the final results. The reason of introducing n is to reduce the interference of the destinations that are far away if using (7.4) to calculate the distance. Larger *n* leads to faster convergence if the prediction is correct, but also leads to larger error if prediction is incorrect, such as the vehicle goes to a new destination. The estimated distance is updated every time when the vehicle position is updated by GPS. This update allows probabilities to become more precise in most cases as the vehicle is approaching the destination.

7.3.3 Results of destination prediction

To illustrate the algorithm mentioned above, 5 scenarios are studied, and the trip and possible destinations are shown in Fig. 7.2~7.6.

- Scenario 1: the vehicle starts at initial point A; two possible destinations B and C are at opposite directions and have initial 50% probability each.
- Scenario 2: the vehicle starts at initial point A; two possible destinations B and C form a triangle with A and have initial 50% probability each.
- Scenario 3: the vehicle starts at initial point A, and the final destination is far from A with low initial probability.
- Scenario 4: The vehicle starts at initial point A, passes one of the possible destinations C and reaches the destination B.
- Scenario 5: One of the possible destinations is far away and it is not the final destination.

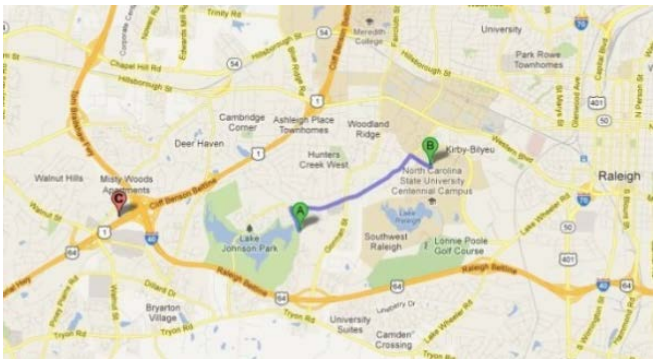


Figure 7.2 The trips and destinations, Scenario 1

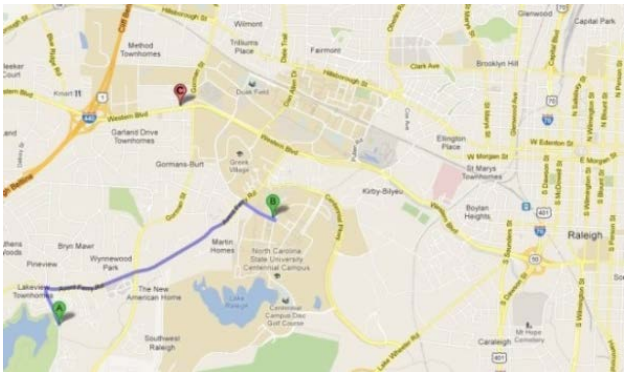


Figure 7.3 The trips and destinations, Scenario 2

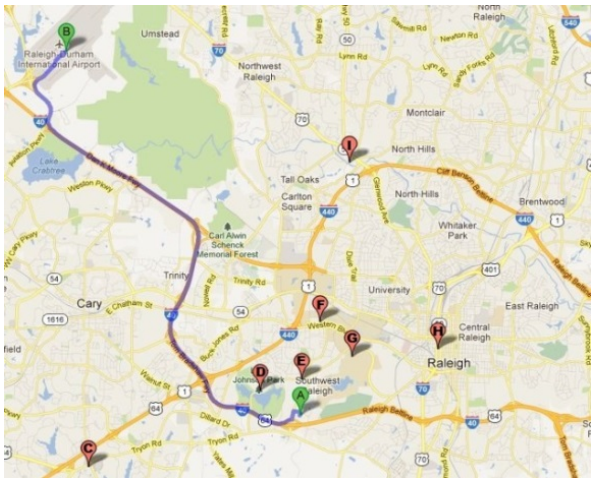


Figure 7.4 The trips and destinations, Scenario 3

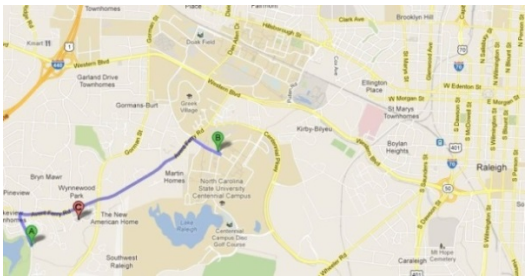


Figure 7.5 The trips and destinations, Scenario 4

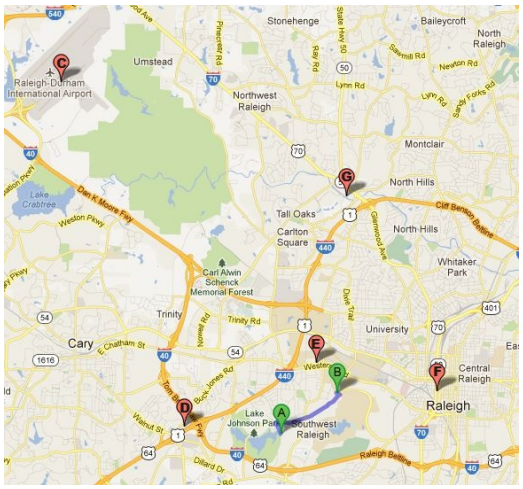


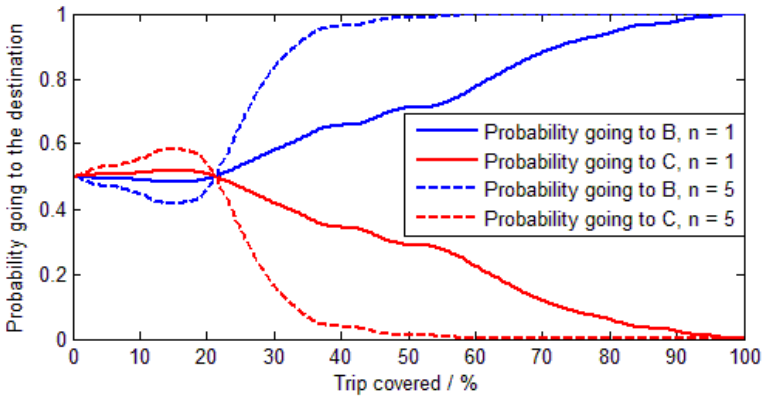
Figure 7.6 The trips and destinations, Scenario 5

Fig. 7.7~7.11 shows the results of dynamic probability updating as the vehicles moves and the comparison of estimated distance to the actual distance to the destination for the 5 scenarios mentioned above. The difference in the results by introducing a power factor n is also shown. In this study, n is increased until the distance estimation converges by 50% of each trip when n = 5. It is considered a convergence when the estimated distance error is within $\pm$10% or less than 10% of the trip length and remains in those ranges till the end of

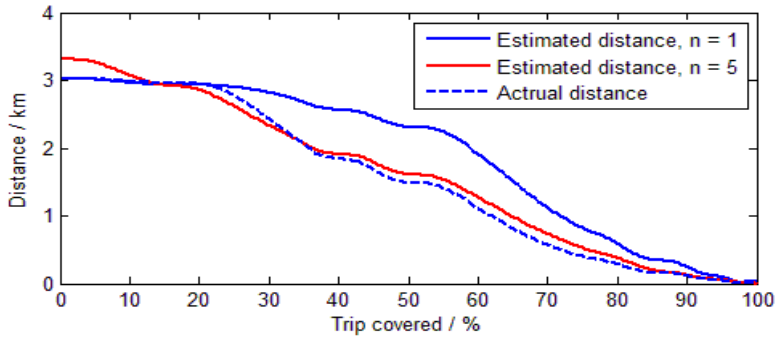
the trip. From Fig. 7.7~7.11, by introducing a power factor n = 5, the estimated distance converges to the actual distance faster.

For the scenario that the two destinations are at opposite directions shown in Figure 7.2, the convergence is fast as shown in Fig. 7.7(b). That is because as the vehicle moves, the distance to B (the destination) decreases while that to C increases. For Scenario 2, although the probability distribution does not indicate the destination B until around 70% of the trip, the distance estimation converges fast (at around 30% of the trip). That is because for the first half of the trip, the distances to B and C are similar such that the probability does not change the estimated distance really much by (7.4). For Scenario 3, as the destination B is of a low initial probability and the vehicle passes through some other possible destinations before going to B, the estimation does not converge before 50% of the trip. However, the distance estimation converges for the second half of the trip from Fig. 7.9(b) with n = 5. For Scenario 4, the probability of going to C increases to nearly 100% even though the initial probability going to C is pretty low. That brings a big error around 30% of the trip from Fig. 7.10(b). However, the probability going to C decreases fast and the distance estimation is accurate after the vehicle passes C. In this scenario, the accuracy depends on where C located between A and B. If the trip is made from B to A, for example, the prediction does not converge until 70% of the trip as shown in Fig. 7.10(c). To sum up, for Scenario 4 the distance estimation does not converge until the vehicle passes the possible destination in the middle of a trip, and the accuracy is dependent on where C is located. However, the estimated remaining distance never exceeds the actual distance such that battery energy is favored just as CD-CS strategy, so the performance cannot be worse than CD-CS strategy.

For Scenario 5, although the initial probability to the actual destination B is highest, the estimation still does not converge until 50% of the trip. That is because by (7.4), although the probability going to a possible destination really far away (C in Fig. 7.6) is really low, the big distance times a low probability is still substantial.

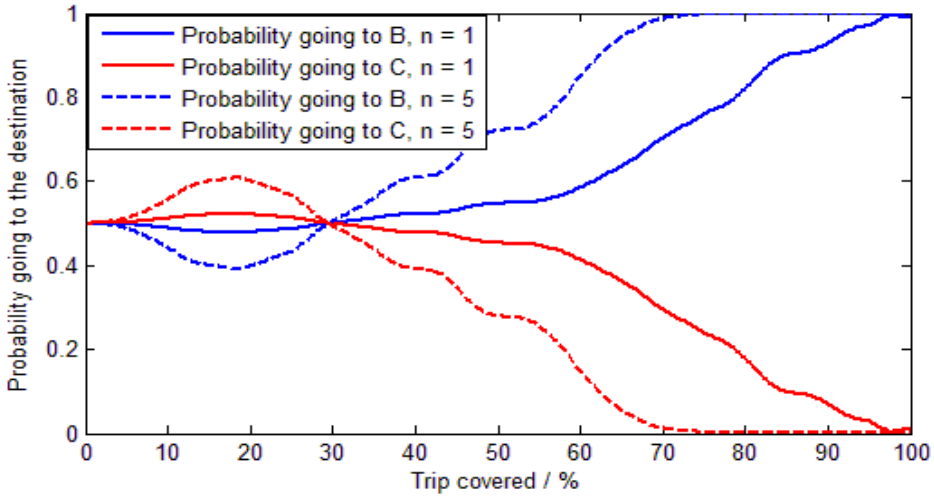


a) Probability updating as the vehicle moves

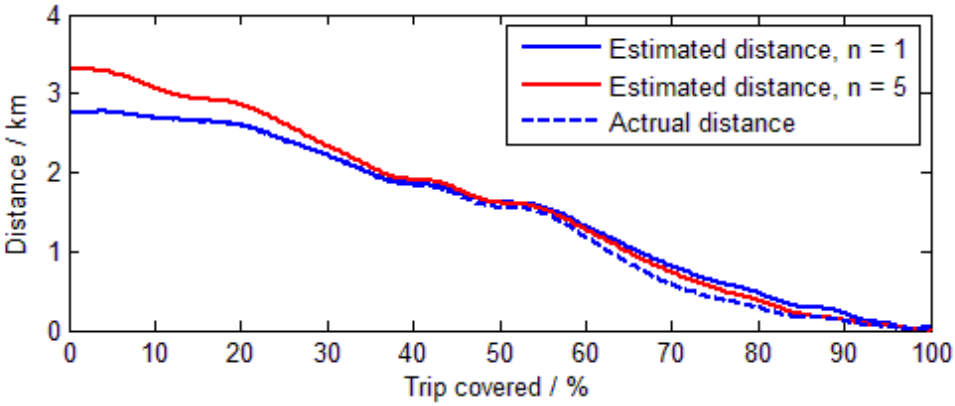


b) Estimated vs. actual distance as the vehicle moves

Figure 7.7 The results of distance estimation with dynamic probability updating (Scenario 1)

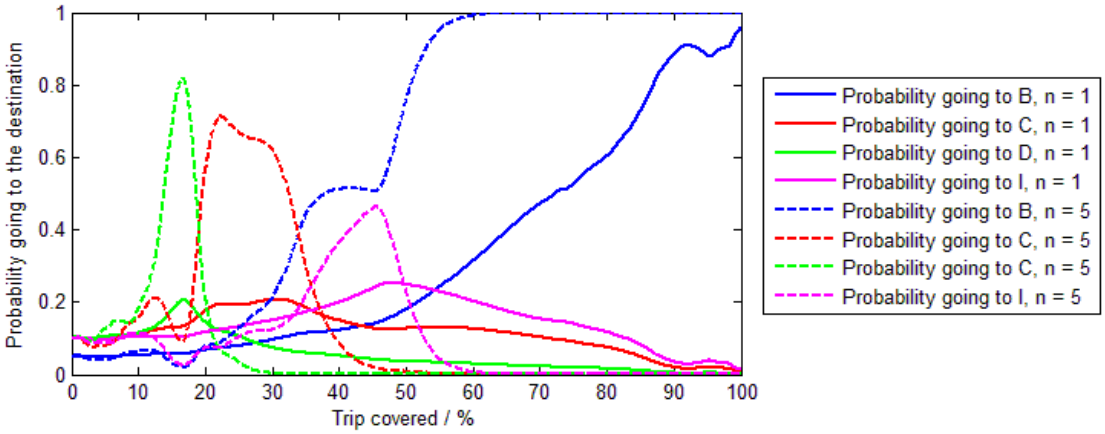


a) Probability updating as the vehicle moves

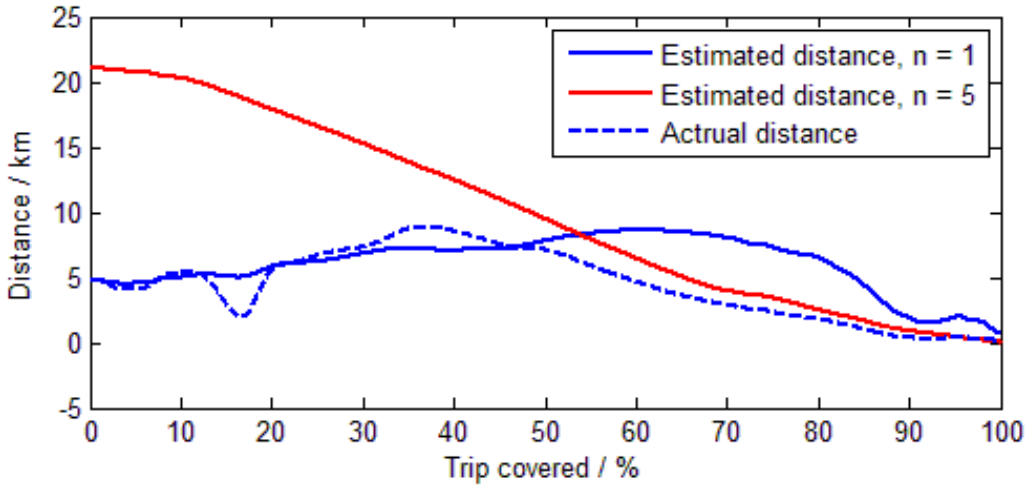


b) Estimated vs. actual distance as the vehicle moves

Figure 7.8 The results of distance estimation with dynamic probability updating (Scenario 2)

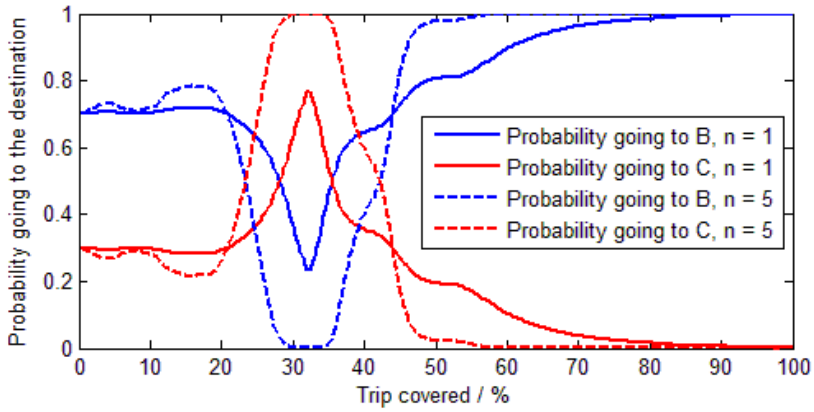


a) Probability updating as the vehicle moves

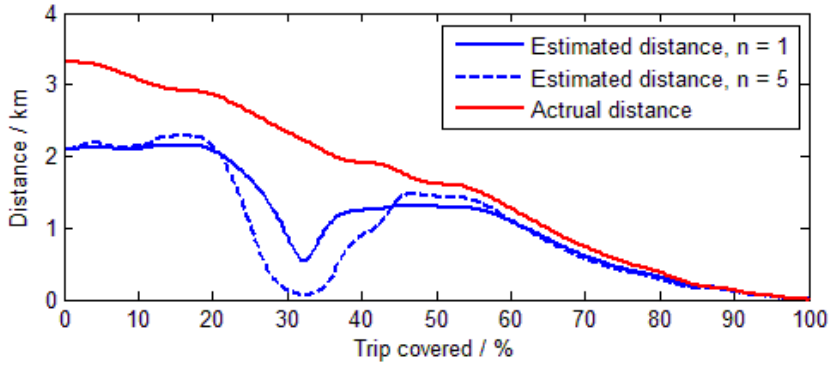


b) Estimated vs. actual distance as the vehicle moves

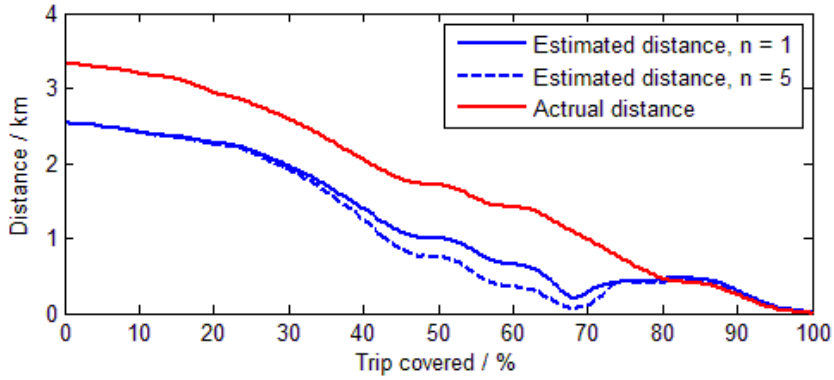
Figure 7.9 The results of distance estimation with dynamic probability updating (Scenario 3)



a) Probability updating as the vehicle moves

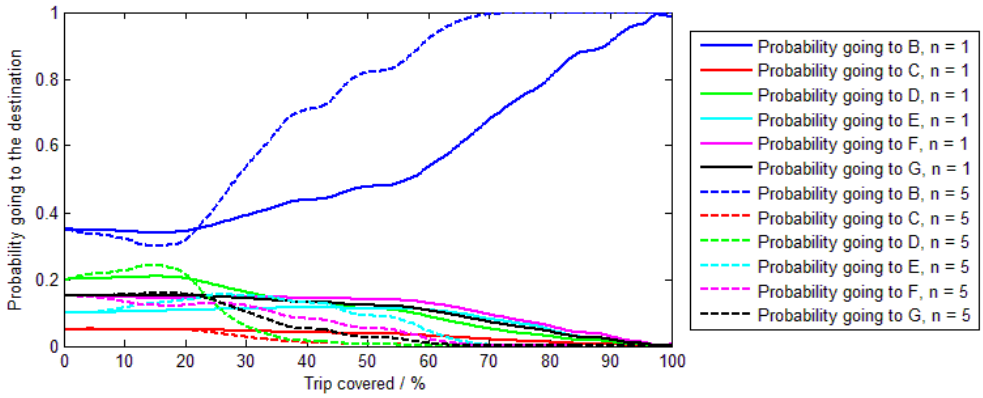


b) Estimated vs. actual distance as the vehicle moves (Scenario 4-1)

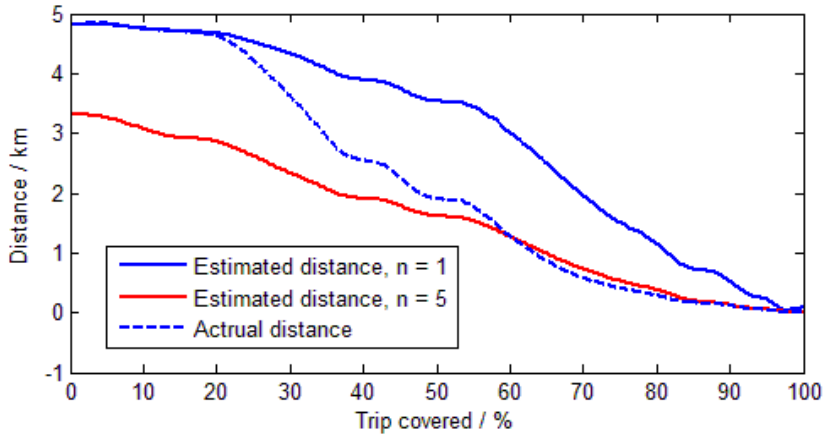


c) Estimated vs. actual distance as the vehicle moves (Scenario 4-2)

Figure 7.10 The results of distance estimation with dynamic probability updating (Scenario 4)



a) Probability updating as the vehicle moves (Scenario 5)



b) Estimated vs. actual distance as the vehicle moves

Figure 7.11 The results of distance estimation with dynamic probability updating (Scenario 5)

7.4 Analysis of error in distance estimation

In Section 7.3, although 5 typical scenarios are studied, they are still not able to represent for all the possible cases in real world. To analyze more generally the performance of this algorithm when the distance estimation goes wrong, two typical cases are analyzed: the destination is within (Case 1) or beyond (Case 2) the AER. For each case, 6 subcases are studied:

1. The distance to the destination is over estimated all along the drive cycle;
2. The distance to the destination is over estimated for the first half of the drive cycle, then estimation is close to the actual distance;
3. The estimation is close to the actual distance for the first half of the drive cycle, then the distance to the destination is over estimated;
4. The distance to the destination is under estimated all along the drive cycle;
5. The distance to the destination is under estimated for the first half of the drive cycle, then estimation is close to the actual distance;
6. The estimation is close to the actual distance for the first half of the drive cycle, then the distance to the destination is under estimated.

Here 'over estimated' means the estimation distance is twice of the whole trip length, and 'under estimated' means the estimation distance is exactly 1 km along the trip except for the last 1km of the trip when the estimation is 200m. These error subcases already cover all the five Scenarios in Section 7.3 and the errors go more extreme.

The drive cycle shown in Figure 7.4 is used for both cases, and the initial SOC is set to 70% for Case 1 and 40% for Case 2. The results are shown in Table 7.1. As the final SOC may not be the same for the two cases, the equivalent cost (in dollars) is considered for comparison.

From Table 7.1, the distance estimation based control algorithm is substantially less economic than CD-CS strategy only when the destination is within the AER (Case 1) and the distance is over estimated (subcase 1). For the subcases in which the distance is under estimated for Case 1, the performances by two strategies are close. On the other hand, if the

destination is beyond the AER (Case 2), this algorithm can tolerate the error and the estimation during the second half of a trip is more important. This also explains the reason why the factor n is introduced in (7.4) for faster convergence.

Table 7.1 Simulation Results for Error Tolerance Tests

Case	Subcase	Improvement over CD/CS
1	1	-13.3%
	2	-7.7%
	3	-9.0%
	4	-1.8%
	5	-1.8%
	6	-1.8%
2	1	10.0%
	2	12.0%
	3	5.6%
	4	9.8%
	5	9.5%
	6	4.6%

7.5 Conclusions

In Chapter 6 all the simulation results are based on a known destination which may not be available directly from the driver. This Chapter presents a real-time destination prediction strategy based on which the distance to the destination can be estimated, and this distance is necessary for the DP-ECMS combined strategy in Chapter 6. The procedures of establishing a Markov model based probability distribution of the possible destinations have been presented, and the algorithm of updating the probability in real time has been explained. A factor of *n* has been imported to make the convergence faster. This algorithm works very

well if the destination is pre-stored, which means it does not guarantee a better fuel economy for a new destination, especially if it is within the all-electric range.

The estimation error analysis shows that in most cases especially the destination is beyond AER, even with a low fidelity of distance estimation, by the proposed DP-ECMS strategy an improvement of fuel economy over CD-CS strategy can be still expected.

Chapter 8 ECMS-DP Combined Strategy with Driving Pattern and Driving Style Recognition

8.1 Introduction

The preceding chapter discussed the possibility of drive distance estimation. However, this historical data based method can never accurately estimate the distance to the destination if the driver is going to a new location that is never visited. Actually, with the continuous development of science and technology, drivers nowadays are able to 'talk' to the vehicle about the destination via voice command [73-76]. It is assumed in this chapter that the driving distance information is available from a known destination.

From Fig. 6.4, the optimal λ values at same SOC and remaining distance are quite different from driving pattern to driving pattern, which means the driving distance information itself cannot guarantee a near-optimal performance of the control strategy. From Table 6.1, the potential of further improvement with driving pattern information is 7% ~ 14% on the standard driving cycles in the table. The term driving pattern refers to the type of drive cycles, and three main patterns are considered in this chapter: city, suburban, and highway. Even though the on-board GPS can calculate the exact driving distance to the destination from the destination indicated by the driver, the driving pattern along the trip remains uncertain. GPS estimates the arriving time based on the speed limit of each section of the planned trip, but drivers never absolutely follow the speed limits, especially due to traffic condition. It should be noted that driving pattern is not necessarily decided by the type of road the driver is driving on. If there is congestion on highway, it can be similar to suburban

or even city driving pattern; if it is midnight in a city, the driving pattern on arterial roads can be close to suburban pattern if all traffic lights along the trip are green.

The methods that have been implemented towards driving pattern recognition have been summarized in Chapter 2. Statistics based method is the most popular approach towards driving pattern recognition [31] [32] [33] [35] [36] [37] [38]. The literatures using statistics based method exclusively consider some characteristic parameters in a recent time window, such as average speed and max acceleration, to recognize the driving pattern in that time window, and assume that this pattern is identical to current pattern. The length of time window should be large enough to catch the characteristic parameters as discussed in [32], which concludes that a larger time window can improve the accuracy of recognition. However, with a large time window the driving pattern cannot be updated promptly if there is a change of pattern.

Another approach is to use external data, typically from GPS or ITS. The authors in [34] estimate the future traffic flow variation trend (increasing or decreasing) and elevation variation by GPS, and implement fuzzy logic controller to decide the battery discharge / charge rate for a series HEV. However, the traffic flow variation trend cannot fully represent the driving pattern. For example, if increasing traffic flow is expected, that can be a change of driving pattern either from city to suburban or from suburban to highway. In [41, 68], the authors use the planned trip by GPS and estimate the drive cycle based on the speed limit of each section. However, the speed limit can represent the driving pattern on its own only when the vehicles are moving freely without traffic.

Main mapping apps such as *Google Maps* and *Apple Maps* are able to show the traffic condition by different colors. Take *Google Maps* for example, on highway roads the color is determined by speed only and on city roads the color is determined by congestion health which is calculated by (8.1), and the thresholds are shown in Table 8.1[77]. In Equation (8.1), CH represents the congestion condition, $\bar{v}$ is the average speed of all vehicles on the segment, v_{lim} is the speed limit of the segment, O_{max} is the max occupancy of the segment and O_{act} is current reported occupancy. In this chapter, the term segment is defined as a portion of the drive cycle on which the speed limit doesn't change.

$$CH = \frac{1}{2}[\frac{\bar{v}}{v_{lim}} + (1 - \frac{O_{act}}{O_{max}})] \times 100\% \tag{8.1}$$

Table 8.1 Thresholds that determine the color of the roads on a map showing traffic

Color	Highway	City
Green	speed is over 50 mph	congestion health is between 80% - 100%
Yellow	speed is between 25 - 50 mph	congestion health is between 60% - 80%
Red	speed is between 15 - 25 mph	congestion health is between 40% - 60%
Black	speed is between 0 - 15 mph	congestion health is between 0% - 40%
Grey	no sufficient data is available	no sufficient data is available

Showing the traffic by colors is straightforward and apprehended at a glance. However, it is not ideal for PHEV control strategies based on driving pattern recognition. If the speed is close to the boundaries, for instance, the estimation error can be large. Average speeds of 49

mph and 51 mph lead to different colors on highway roads, yet the actual driving patterns are similar. To solve this issue, fuzzy logic controller is preferred in this chapter. In addition, the color on the map only shows the average speed of all vehicles on a certain segment. However, a driver may drive slower or faster than the average traffic flow, so the average speed of the specific driver rather than the average speed of all drivers is of more importance to the vehicle controller.

On the other hand, aggressive driving style has significant contribution to poor fuel economy of the vehicle (as large as 60%) [78]. To address the aggressiveness of a driver, driver's driving style recognition is just as important. In [79] the parameters that have been considered to indicate the driving style are summarized. However, neither the acceleration [80, 81] nor the jerk information(second deviation of speed) [82] can independently represent the driving style without the driving pattern information. On a congested road, for example, even calm drivers have to accelerate and decelerate frequently; on a smooth highway, however, even aggressive drivers may maintain relatively constant speed while driving fast.

In this chapter a new method for driving pattern recognition is introduced, using both speed limit information and traffic condition. A known destination is assumed, such that the speed limits along the trip are available from GPS, and the traffic condition is determined by the actual speed of the driver compared to the speed limit. A fuzzy logic controller is developed to decide the current driving pattern. Also introduced in this chapter is a new method for driving style recognition method using power requirement and current driving pattern, and another fuzzy logic controller that recognizes the driver's driving style is

developed. Finally, an equivalent factor is calculated based on the quasi-optimal λ maps in Chapter 6 for different driving patterns, current driving pattern, and the driver's driving style.

8.2 Driving patterns and driving styles

Before the driving pattern recognition algorithm, it is necessary to define the driving patterns by their exclusive characteristics. From the US Environmental Protection Agency (EPA), standard drive cycles are given to represent different driving patterns including stop-and-go, urban, highway, and aggressive highway driving pattern for light duty vehicles [83].

Referring to EPA driving pattern classification, the following driving patterns in Table 8.2 are considered in this chapter, which is slightly different than EPA patterns. Table 8.2 also defines the characteristics of each driving pattern.

Table 8.2 Driving Patterns and Characteristics in this thesis

Driving pattern	Characteristics	Representative driving cycle
Highway	High speed, no interval of stopping	HWFET
Suburban	Moderate speed, moderate interval of stopping	UDDS
Urban / city	Low speed, frequent interval of stopping	NYCC

Driving pattern can be roughly represented by the speed limit on a road. For example, the speed limit is typically above 55 mph (90 km/h) on highways and below 30 mph (50 km/h) on the roads in urban area [84]. Actually when the speed limits are designed, some factors such as 85th percentile speed is considered, such that the speed limit is set at a speed such that 85% ~ 90% of the free-moving vehicles on that segment are with an average speed

below the speed limit [85]. In another words, if the vehicles are moving freely, the speed limit can represent the driving pattern with a confidence level of 85%~90%. Due to the traffic condition, however, the average speed of vehicles widely fluctuates around the speed limit. In this chapter, a speed ratio is considered to represent traffic condition, which is defined as the average speed of a vehicle on a segment divided by the speed limit of the segment.

Table 8.3 Power requirements of selected standard drive cycles

Cycle	$\overline{V}$ / mph	$\overline{Preq+}$ / kW
NYCC	11	4.18
NurembergR36	12.7	4.6
CSHVR	17.5	4.38
1015_6PRIUS	18.8	4.83
WVUSUB	21.66	4.8
UDDS	24.4	6.56
NEDC	26.24	6.23
SC03	26.4	7.73
LA92	29.14	10.2
IM240	30.5	9.53
HWFET	48	9.38
US06	52.9	18.8

As for the driving style, three typical styles are considered in this chapter: Calm, Normal, and Aggressive. As outlined in Section 8.1, power demand is combined with driving pattern for driving style recognition. According to the standard driving cycles in Table 8.3, average power demand and average speed of a drive cycle have a positive correlation. In Table 8.3 $\overline{V}$

is average non-stop speed, and $\overline{\mathrm{Preq}+}$ is average positive power requirement. The color of the rows represents the driving pattern: light orange is city pattern; light purple is between city pattern and suburban pattern; light blue is suburban pattern, and light green is highway pattern. From the table, even the calm style on highway drive cycle HWFET has a large power demand, such that the driving style recognition algorithm using power requirements should consider the current driving pattern.

8.3 Fuzzy logic controllers for Driving pattern and driving style recognition

As there is not a clear boundary between different driving patterns, fuzzy logic controller is favored for driving pattern recognition in this chapter. The 12 standard driving cycles in Table 8.3 are divided into two groups: training group and test group. Both the driving pattern and driving style recognition fuzzy controllers are designed based on the training group, which contains all the driving patterns and styles as shown in Table 8.4.

Table 8.4 Training group of driving cycles for fuzzy logic controllers

Number	Cycle
1	NYCC
2	1015_6PRIUS
3	UDDS
4	NEDC
5	LA92
6	US06

8.3.1 Driving pattern fuzzy logic controller

As discussed in Section 8.2, two parameters, speed limit v_{lim} and speed ratio v_r, are selected as inputs of the fuzzy logic controller of driving pattern recogition. Speed limit v_{lim}

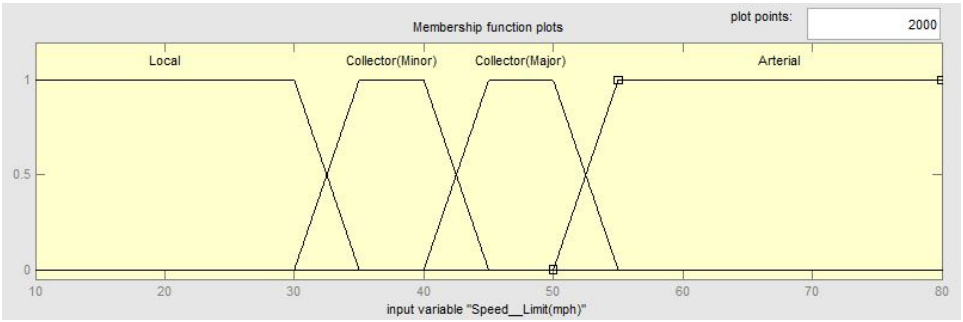
can be an estimation of road types, and v_r represents the traffic condition. The output of fuzzy logic controller is a number that represents the driving pattern. Speed ratio v_r can be either smaller or greater than 100%, which is defined as

$$v_r = \bar{v}_{actual}/v_{lim} \times 100\% \tag{8.1}$$

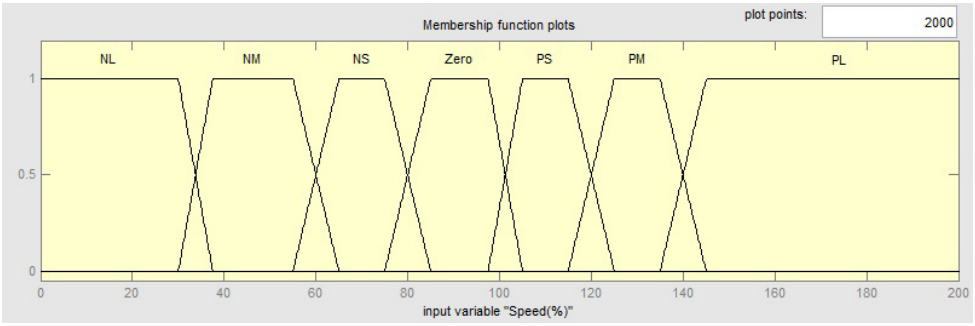
where $\bar{v}_{actual}$ is the actual average speed of a specific vehicle.

To compare $\bar{v}_{\text{actual}}$ and v_{lim}, a trip is divided into segments according to speed limit, and $\bar{v}_{\text{r}}$ is calculated one segment by another. In this way, if the driving pattern switches (the driver merges onto a highway for instance), the average speed on previous driving pattern should not disturb the driving pattern recognition on the new segment. Another key factor is the time window for averaging the historical speed such that it can reflect the current traffic condition. As mentioned in [79], if the average speed is calculated based on a large recent time window, the accumulated historical data cannot promptly reflect a sudden traffic condition change. A good example is an accident on highway. The driving pattern can be city pattern for up to several miles before the scene of the accident, and once the vehicle passes the scene of the accident it resumes to highway pattern immediately. If the time window is too short, on the other hand, the historical data may not be sufficient to reflect the current pattern and the mode fluctuation can be expected. In this chapter, $\bar{v}_{actual}$ is calculated considering all short (10s), median (30s) and long (60s) time windows by (8.2). In (8.2), $\bar{v}_i$ is the average non-zero speed in the corresponding time window i, $i \in \{10, 30, 60\}$, and ω_i, are the weights. In this chapter $\omega_{10} = 0.5$, $\omega_{30} = 0.3$, and $\omega_{60} = 0.2$ are considered. Higher weights are given to the shorter time windows to address the fact that the more recent data can better reflect the current condition.

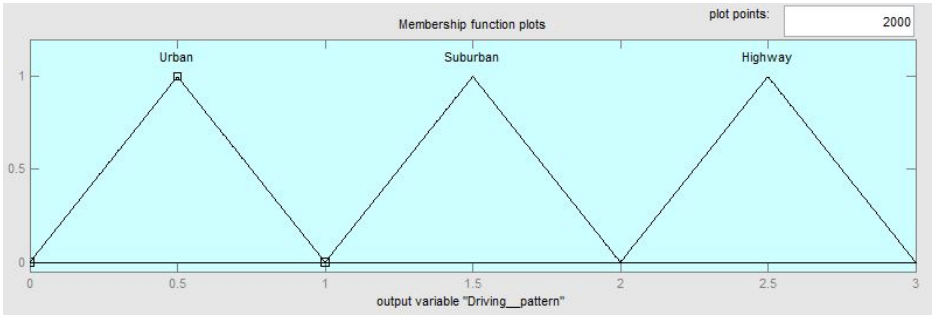
$$\bar{v}_{actual} = \sum_i \omega_i * \bar{v}_i \tag{8.2}$$



a) Input 1: Speed limit



b) Input 2: Speed difference



c) Output: Driving pattern

Figure 8.1 Membership functions of the fuzzy logic controller for driving pattern recognition

Based on how the speed limit is determined for different road types [84], all the roads are classified into 4 types in this chapter: Local, Collector (minor), Collector (major) and Arterial.

The membership functions and rules are trained by the training group of drive cycles in Table 8.4. As the standard driving cycles don't have speed limit v_{lim} information, speed limits must be assigned to each drive cycle. To ensure the robustness of this driving pattern recognition algorithm, each time a constant speed limit $v_{lim} \in \{25, 35, 45, 55, 65\}$ is assigned to each of the drive cycles. Ideally, the recognized driving pattern should be independent of the speed limit. The input and output membership functions are designed as shown in Fig. 8.1.

The rule base for the fuzzy logic controller in this chapter is presented as follows:

1.	IF	v_{lim} = Local	AND	v_r =	NL	THEN	Driving Pattern = Urban
2.	IF	v_{lim} = Local	AND	v_r =	NM	THEN	Driving Pattern = Urban
3.	IF	v_{lim} = Local	AND	v_r =	NS	THEN	Driving Pattern = Suburban
4.	IF	v_{lim} = Local	AND	v_r =	ZERO	THEN	Driving Pattern = Suburban
5.	IF	v_{lim} = Local	AND	v_r =	PS	THEN	Driving Pattern = Suburban
6.	IF	v_{lim} = Local	AND	v_r =	PM	THEN	Driving Pattern = Suburban
7.	IF	v_{lim} = Local	AND	v_r =	PL	THEN	Driving Pattern = Highway
8.	IF	$v_{\lim}$ = C_Minor	AND	v_r =	NL	THEN	Driving Pattern = Urban
9.	IF	$v_{\lim}$ = C_Minor	AND	v_r =	NM	THEN	Driving Pattern = Urban
10.	IF	$v_{\lim}$ = C_Minor	AND	v_r =	NS	THEN	Driving Pattern = Suburban
11.	IF	$v_{\lim}$ = C_Minor	AND	v_r =	ZERO	THEN	Driving Pattern = Suburban
12.	IF	$v_{\lim}$ = C_Minor	AND	v_r =	PS	THEN	Driving Pattern = Suburban
13.	IF	$v_{\lim}$ = C_Minor	AND	v_r =	PM	THEN	Driving Pattern = Highway
14.	IF	$v_{\lim}$ = C_Minor	AND	v_r =	PL	THEN	Driving Pattern = Highway
15.	IF	$v_{\lim}$ = C_Major	AND	v_r =	NL	THEN	Driving Pattern = Urban
16.	IF	$v_{\lim}$ = C_Major	AND	v_r =	NM	THEN	Driving Pattern = Suburban
17.	IF	$v_{\lim}$ = C_Major	AND	v_r =	NS	THEN	Driving Pattern = Suburban
18.	IF	$v_{\lim}$ = C_Major	AND	v_r =	ZERO	THEN	Driving Pattern = Suburban
19.	IF	$v_{\lim}$ = C_Major	AND	v_r =	PS	THEN	Driving Pattern = Highway
20.	IF	$v_{\lim}$ = C_Major	AND	v_r =	PM	THEN	Driving Pattern = Highway
21.	IF	$v_{\lim}$ = C_Major	AND	v_r =	PL	THEN	Driving Pattern = Highway
22.	IF	$v_{\lim}$ = Arterial	AND	v_r =	NL	THEN	Driving Pattern = Urban
23.	IF	$v_{\lim}$ = Arterial	AND	v_r =	NM	THEN	Driving Pattern = Suburban
24.	IF	$v_{\lim}$ = Arterial	AND	v_r =	NS	THEN	Driving Pattern = Highway
25.	IF	$v_{\lim}$ = Arterial	AND	v_r =	ZERO	THEN	Driving Pattern = Highway
26.	IF	$v_{\lim}$ = Arterial	AND	v_r =	PS	THEN	Driving Pattern = Highway

27. IF v_{lim} = Arterial AND v_r = PM THEN Driving Pattern = Highway
28. IF v_{lim} = Arterial AND v_r = PL THEN Driving Pattern = Highway

where

C_Minor: Collector (minor);

C_Major: Collector (major);

PL: positive large;

PM: positive medium;

PS: positive small;

NL: negative large;

NM: negative medium;

NS: negative small.

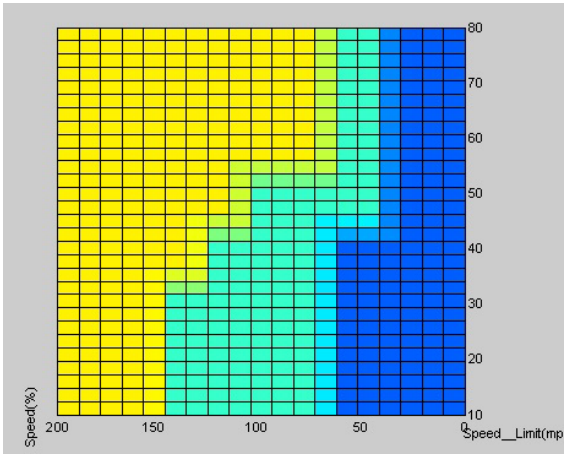
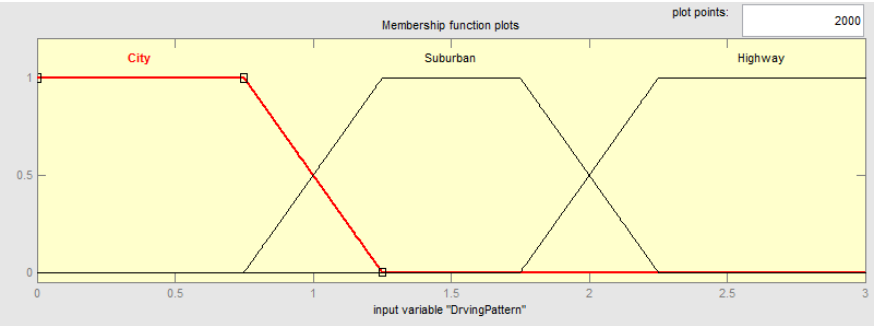
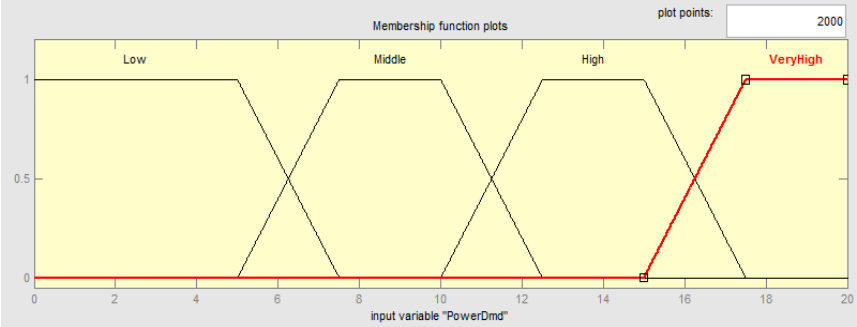


Figure 8.2 The surface view of the fuzzy rules for driving pattern recognition

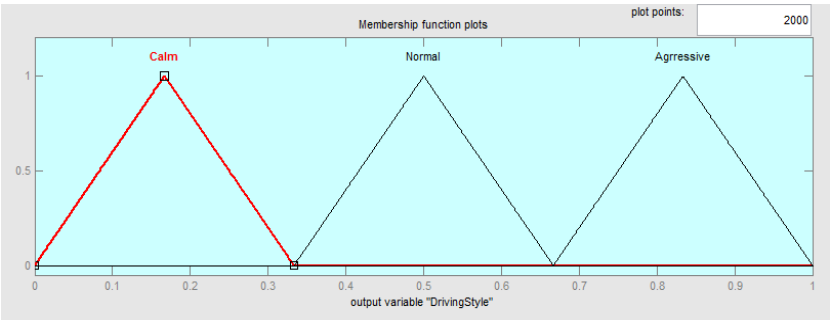
Based on the rules, the output value with respect to the two inputs is shown in Fig. 8.2, where yellow squares are for highway pattern, cyan for suburban, and blue for city. The output of the fuzzy logic controller is a number that represents the driving pattern.



a) Input 1: Driving Pattern



b) Input 2: Average Power Demand



c) Output: Driving Aggressiveness

Figure 8.3 Membership functions of the fuzzy logic controller for driving style recognition

8.1.1 Driving style fuzzy logic controller

As discussed in Section 8.2, the average positive power demand is a better estimation of a driver's driving style than acceleration, and it must integrate the driving pattern information. Therefore, the fuzzy logic controller for driving style recognition has two inputs: current driving pattern and average power demand. Current driving pattern is estimated in 8.3.1, and based on the average power demand for the standard drive cycles in the training group, the membership functions are shown in Fig. 8.3. The output of the fuzzy logic controller is an number that represents the aggressiveness of a driving style, and the surface view of the rules is shown in Fig. 8.4 where yellow squares represent aggressive driving style, cyan for normal style, and blue for calm style.

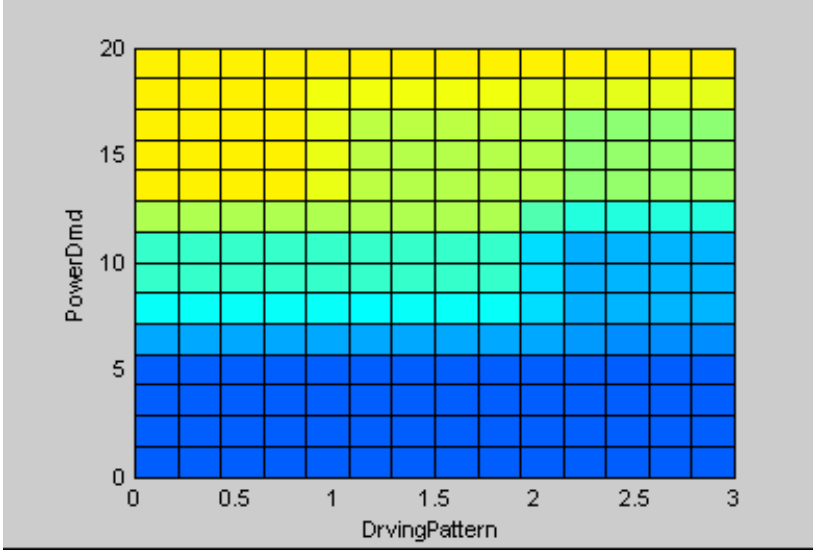


Figure 8.4 The surface view of the fuzzy rules for driving style recognition

8.2 Fuzzy logic implementation

The outputs of the fuzzy logic controllers are not directly implementable by ECMS; additional work is needed to convert the driving pattern to the equivalent factor λ. The simulation logic is shown in Fig. 8.5. The standard drive cycles are classified into city, suburban, and highway patterns as shown in Table 8.3, and for each drive cycle one quasi-optimal λ map with respect to SOC and remaining distance has already been built in Chapter 6. Then a λ map that represents a certain driving pattern is built by averaging all the quasi-optimal λ maps belonging to the corresponding driving pattern, and finally one λ map is built for each of the three driving patterns. At each step of a drive cycle, an equivalent factor λ is available from each of the three λ maps, and based on the recognized driving pattern, linear interpolation is used to find the λ value at current step. Driving style compensation is designed as a multiplier to increase the λ value for aggressive driving style, which is the final λ value for ECMS to find the optimal engine power and battery power combination as described by (6.2).

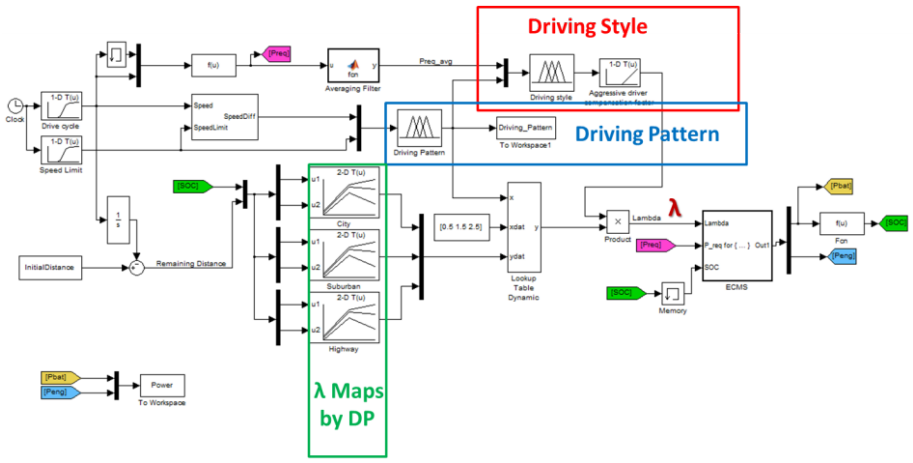


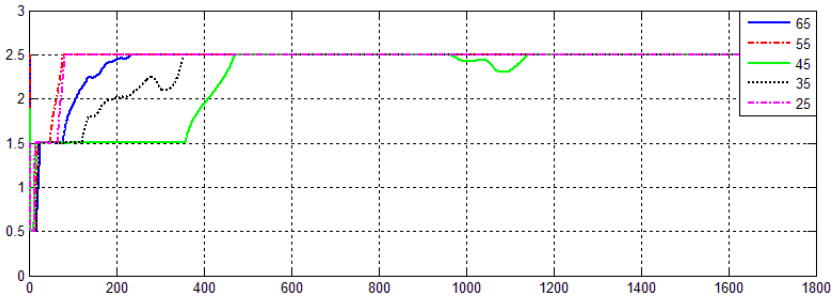
Figure 8.5 Simulation model with driving pattern and driver's driving style recognition

8.2.1 Simulation with standard driving cycles

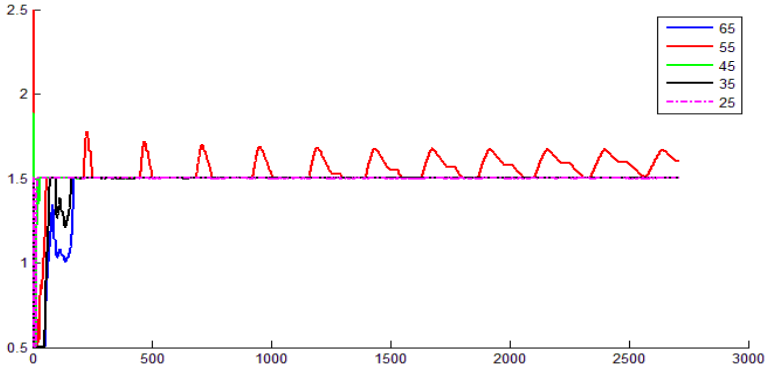
Test group of standard drive cycles in Table 8.5 are simulated. Again, a speed limit $v_{\text{lim}} \in \{25, 35, 45, 55, 65\}$ is assigned to each of the drive cycles at a time. Fig. 8.6 shows examples of the performance of the fuzzy logic controller for driving pattern recognition. In Fig. 8.6 the driving pattern is shown as a number: 0.5 as city pattern, 1.5 as suburban pattern, and 2.5 as highway pattern. Examples of SOC trajectories by ECMS with driving pattern and driving style recognition are shown in Fig. 8.7. The cost by ECMS with driving pattern and driving style recognition is compared to ECMS with corresponding quasi-optimal λ maps and ECMS with universal λ map as in Chapter 6, and the results are shown in Table 8.5.

Table 8.5 Simulation results on the test group of driving cycles

Drive Cycle	Gap to ECMS with Quasi-optimal λ maps / %						Improvement over ECMS with universal λ map / %					
	Speed Limit / mph						Speed Limit / mph					
	25	35	45	55	65	mean	25	35	45	55	65	mean
NurembergR36	1.59	1.59	1.59	1.59	1.59	1.59	6.09	6.09	6.09	6.09	6.09	6.09
CSHVR	2.24	2.65	2.23	2.48	2.65	2.45	6.43	6.02	6.44	6.18	6.02	6.22
WVUSUB	1.43	1.09	1.42	1.55	1.24	1.35	6.34	6.68	6.35	6.22	6.53	6.42
SC03	2.47	2.52	2.47	2.41	2.54	2.48	8.05	8.00	8.05	8.11	7.99	8.04
IM240	3.21	3.10	3.14	2.81	3.07	3.07	7.21	7.32	7.28	7.61	7.35	7.35
HWFET	0.87	1.95	3.46	0.73	1.98	1.80	9.05	7.96	6.46	9.19	7.93	8.12
Average						2.12						7.04



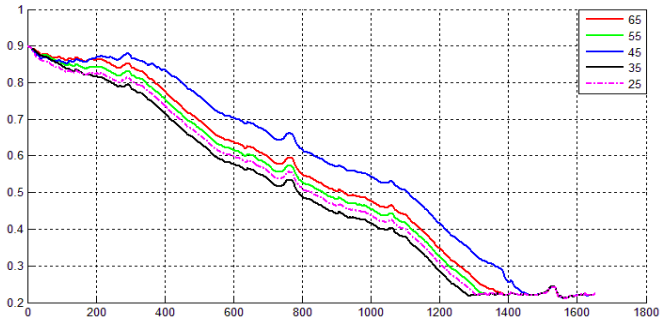
a) HWFET



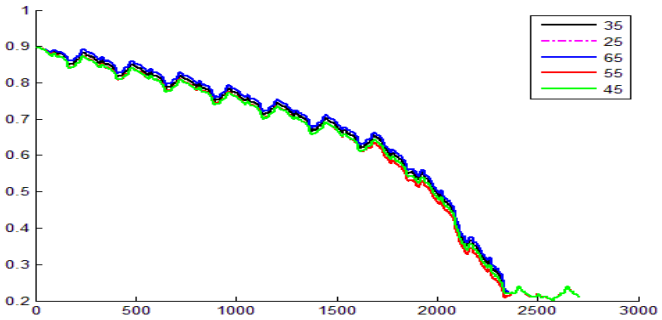
b) IM240

c) SC03

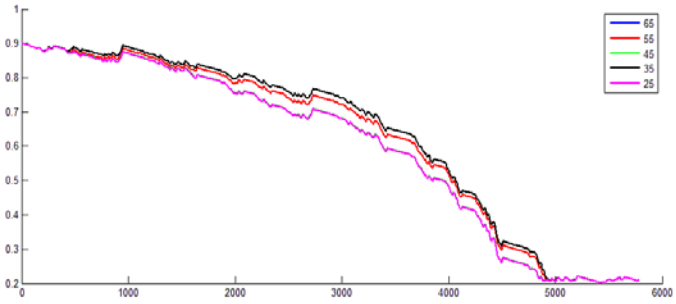
Figure 8.6 Examples of driving pattern recognition results



a) HWFET



b) IM240



c) CSHVR

Figure 8.7 Examples of SOC trajectories with different speed limit

In Table 8.5 the gap to the ECMS with quasi-optimal maps G_{opt} is calculated by (8.3) and the improvement over ECMS with universal λ map Im_{Umap} is calculated by (8.4), where $COST_{\mathrm{Umap}}$ is the equivalent cost by the universal λ map, $COST_{recg}$ is the equivalent cost with driving pattern recognition in this chapter, and $COST_{\mathrm{opt}}$ is the equivalent cost by the corresponding quasi-optimal λ map.

$$G_{opt} = \frac{COST_{recg} - COST_{opt}}{COST_{CD-CS}} \times 100\% \tag{8.3}$$

$$Im_{Umap} = \frac{COST_{Umap} - COST_{recg}}{COST_{CD-CS}} \times 100\% \tag{8.4}$$

From the simulation results, with the driving pattern recognition algorithm in this chapter, an additional 6.6%~ 7.9% average cost reduction compared to using the universal λ map can be expected, and the gap to the results with quasi-optimal λ maps is only 1.6% ~3%.

8.2.2 Simulation with real world driving cycles

The driving pattern and driving style recognition algorithm is further tested by real world driving cycles with actual speed limits. The data was collected by multiple vehicle models in North Carolina Triangle area [86], and 8 routes including both highway, suburban, and city driving patterns are designed as shown in Fig. 8.8. The 8 routes are Local_AB, Local_BA, Local_BC, Local_CB, Highway_AB, Highway_BA, Highway_BC, and Highway CB. The 'Highway' routes follow highway whenever possible while the 'Local' routes avoid driving on highway completely, and the sequence of letter 'ABC' shows the direction. Local_AB, for example, takes local routes from A to B.

As the Prius model in ADVISOR is adopted, only the drive cycles collected in [86] by 2013 Toyota Prius are simulated in this chapter. Examples of the driving cycles with driving

pattern recognition results are shown in Fig. 8.9. Simulation results are shown in Table 8.6 in terms of the equivalent cost gap to DP results G_{DP}, which is calculated by (8.5). In (8.5) $Cost_{DP}$ is the cost by DP and $COST_i$ is the equivalent cost by corresponding control algorithms including CD-CS, ECMS with universal λ map, and ECMS with driving pattern and style recognition. Examples of SOC trajectories by different control algorithms are shown in Fig. 8.10. From the simulation results, even the distance of the driving cycles are substantially shorter than the standard driving cycles simulated in 8.4.1 (above 35km), 3%~7% additional cost reduction can still be achieved by ECMS with the driving pattern and driving style recognition algorithm compared to ECMS with universal λ map, and the gap to the DP results shrinks to 8% in average compared to 14% by ECMS with universal λ map and 35% by CD-CS strategy.

$$G_{DP} = \frac{COST_i - COST_{DP}}{COST_{CD-CS}} \times 100\% \tag{8.5}$$

Table 8.6 Simulation results on real world drive cycles

Drive cycles	Distance / km	CD-CS	ECMS with universal λ map	ECMS with driving pattern and style recognition	Improvement
Highway_AB	17.9	38.75%	10.71%	5.43%	5.28%
Highway_BA	17.5	38.04%	12.67%	8.30%	4.37%
Highway_BC	25.8	25.77%	16.25%	10.67%	5.58%
Highway_CB	25.7	26.50%	16.53%	10.89%	5.64%
Local_AB	16.4	44.30%	14.20%	9.36%	4.84%
Local_BA	15.8	40.62%	7.93%	4.86%	3.07%
Local_BC	27.8	38.08%	15.65%	8.89%	6.76%
Local_CB	27.7	36.16%	16.18%	10.34%	5.85%

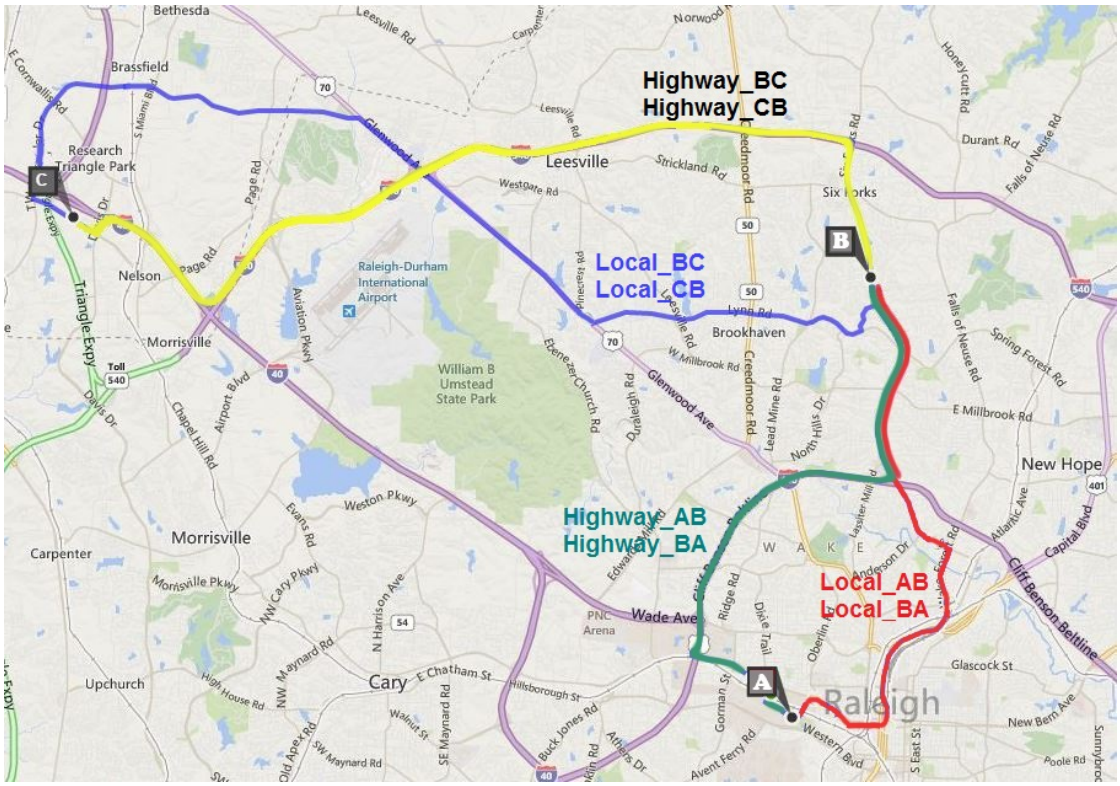
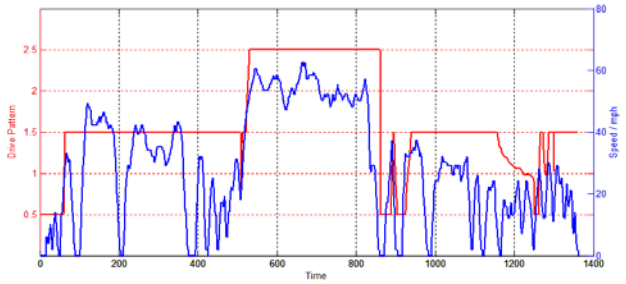
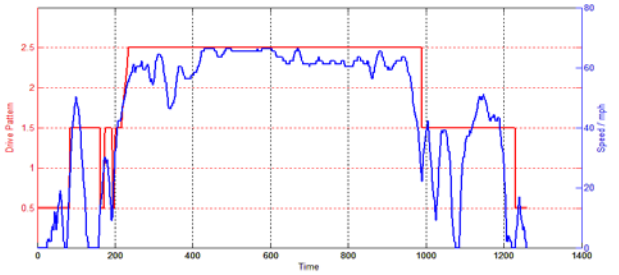


Figure 8.8 Driving routes for real world drive cycle collection

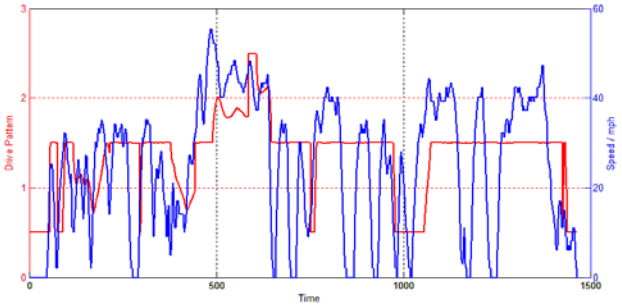
The λ maps for all the driving patterns have a fixed maximum distance, which is about 4 times of AER. If the remaining trip length is greater than the maximum distance of a λ map, the maximum distance is considered. From Fig. 6.5, λ converges to a constant value beyond 4 times of AER, such that this assumption is valid. Simulation result on the drive cycle of Highway_ABC followed by Local_CBA, which has a total trip length of 87.9km, still holds a stable SOC range as shown in Fig. 8.11.



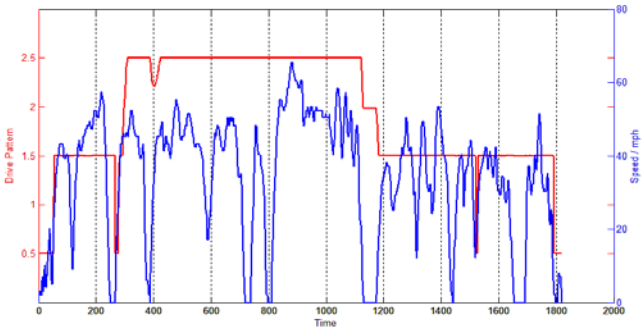
a) Highway_BA



b) Highway_CB

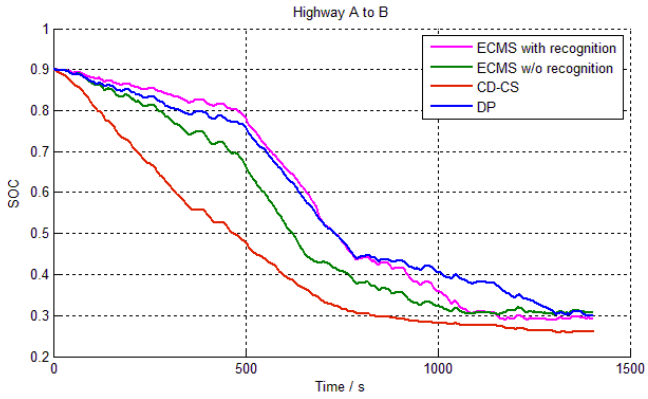


c) Local_AB

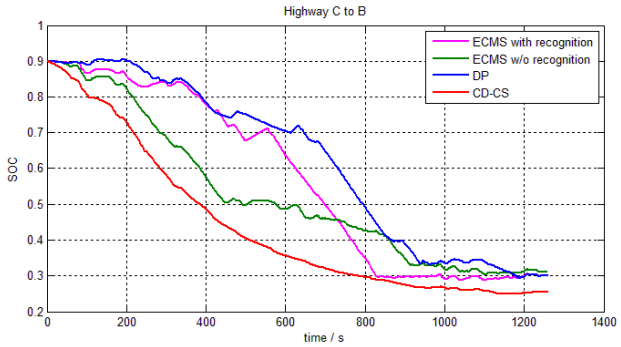


d) Local_CB

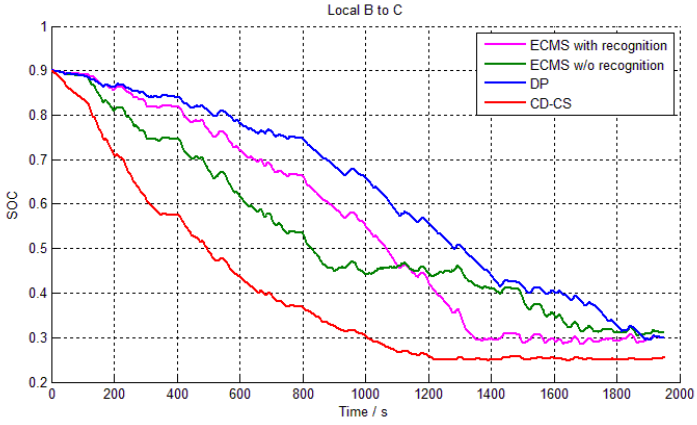
Figure 8.9 Drive cycles and diving pattern recognition results



a) Highway_AB



b) Highway_CB



c) Local_BC

Figure 8.10 SOC trajectories by different control algorithm

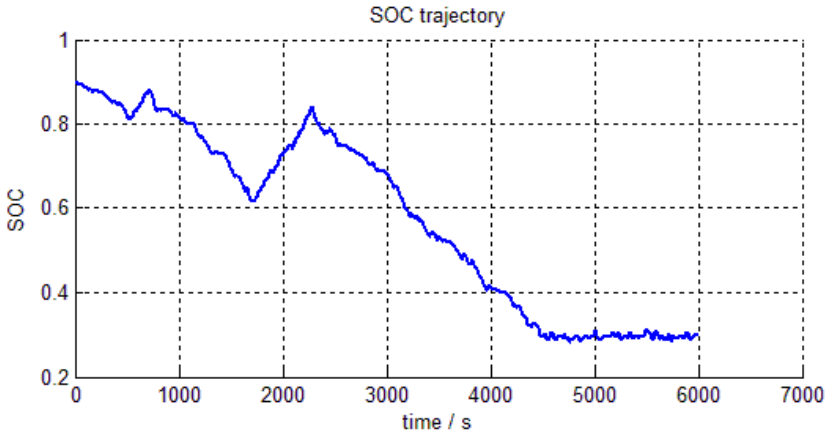


Figure 8.11 SOC trajectory of long trip simulation

8.3 Conclusions

Based on the conclusions in Chapter 6, driving pattern is another important factor that affects the performance of ECMS controller. In this chapter, three patterns (city, suburban, and highway) are considered and a fuzzy logic controller (Fuzzy controller 1) is designed to recognize the driving pattern. In this chapter a known destination is assumed, such that the route and the speed limits on the route to the destination are easily accessible by GPS. The speed limit information is utilized to increase the fidelity of the recognition algorithm. Another input of the fuzzy logic controller is the weighted average speed to represent the traffic condition. In addition, the driving style of a driver is also considered in this chapter and another fuzzy logic controller (Fuzzy controller 2) is designed with inputs of driving pattern and average power demand. The twelve standard driving cycles in Chapter 6 are divided into training group and test group, and the membership functions and rules of the two fuzzy logic controllers are trained by the training group.

The output of fuzzy controller 1 is a number which represents the driving pattern, and additional work is necessary to convert this driving pattern into the λ value which is required by ECMS. From the results in Chapter 6, a quasi-optimal λ map is already available for each of the twelve standard driving cycles, and the 12 cycles are grouped by their driving patterns. Then, a λ map is generated by averaging the quasi-optimal λ maps in the corresponding driving pattern group, and three λ maps are such generated to represent the three driving patterns, respectively. At a certain step, with a known SOC and remaining driving distance to the destination, a λ value is available from each of the three λ maps, and linear interpolation method is adopted based on the output of fuzzy controller 1 to find the λ value in current driving pattern. The output of fuzzy controller 2 is another number that represents the aggressiveness of the driver, which works as a multiplier to compensate the aggressive driving style. After the multiplier, the λ value for ECMS to implement is finally calculated.

Simulation on the test group of standard driving cycles shows an additional 7% cost reduction in average over ECMS without driving pattern and diving style recognition, and the gap to ECMS results with quasi-optimal λ maps is only 2% in average. This algorithm is further tested by real world driving cycles, and 3%~7% additional cost reduction can be achieved by the driving pattern and driving style recognition algorithm compared to ECMS without recognition. The gap to the DP results is 8% in average compared to 14% by ECMS without recognition and 35% by CD-CS strategy.

Chapter 9 Conclusions and Future Work

Plug-in hybrid electric vehicles (PHEVs) are playing an increasingly important role in the real world by further improving the fuel economy and reducing the emissions in comparison with hybrid electric vehicles (HEVs), especially for daily urban commutes. Applying the charge depleting (CD) mode followed by charge sustaining (CS) mode will force the battery energy to be utilized in priority. However, this state-of-the-art control approach may be far from optimal when the trip length is greater than the remaining all-electric range (AER). The optimal control approach has been demonstrated by dynamic programming (DP) in this thesis as a blended control where engine operates throughout a trip and battery state of charge (SOC) reaches the low limit right at the end of the trip, and simulation results show up to 60% cost difference between results by DP and results by CD-CS approach.

To improve fuel economy, optimization algorithms suited for to PHEVs are summarized in Chapter 2 and Chapter 3 of this thesis. Rule-based strategies can be implemented in real time, but a rule based controller cannot guarantee performance over any drive cycles.

DP is a well-accepted offline global optimization tool which guarantees a global optimal solution to a cost function, (see Chapter 4). DP can be applied to determine the rules for rule-based online control strategies. Such look-up tables show very good performance over drive cycles similar to the training drive cycle; however a more general approach needs to be investigated if the control approach is to be applicable to any driving pattern. On the other hand, DP results can be used to evaluate an existing control strategy; in this thesis DP results are used as a benchmark.

Chapter 5 introduces an equivalent cost minimization strategy (ECMS), which is an optimal control based strategy and can be implemented in real time. The key to ECMS is to find an equivalent factor λ in its cost function, which can be interpreted either as a factor for calculating miles per gallon gasoline equivalent (MPGe) or as unit price of electricity from grid over unit price of gasoline. From the discussion in Chapter 4, MPGe is an indirect measure of vehicle operating cost, so direct cost is considered in this thesis. For HEVs, λ is a function of driving pattern (city, suburban, and highway) and driving style (calm, normal, and aggressive), and the goal is to regulate the battery SOC while reducing the fuel consumption. For PHEVs, this λ is a function of driving distance, battery SOC, driving pattern, and driving style. Previous works [49] have looked into the influence of driving distance and SOC by building a λ map with respect to these two factors using a simplified equation. This equation-based λ map is tested with multiple drive cycles and it is not stable over all tested drive cycles. The term 'stable' here refers that the SOC is regulated in the range of 20%~100%. Improved equation is proposed in Chapter 5 and stable SOC regulation is achieved. However, this simplified method needs to be improved in a more systematic may.

In Chapter 6, a method is proposed that combines DP and ECMS to find a quasi-optimal λ value with given remaining distance d_R and current SOC for a certain drive cycle. By this method, DP uses λ as the control variable, and ECMS calculates the optimal engine operating point based on the λ value from DP and returns fuel consumption rate and SOC variation to DP. Then DP updates the accumulated cost on the corresponding node (a point representing a remaining distance and an SOC value) at the next step. After DP finishes all the steps of a given drive cycle, from each of the (d_R, SOC) nodes an optimal trajectory to the origin can be

found step by step backwards, and λ values on this optimal SOC trajectory are retained. Using linear regression, a quasi-optimal value of λ on each of the nodes is available. By this method, a quasi-optimal λ map with respect to d_R and SOC can be built for a given drive cycle. Twelve drive cycles are selected as a training set in Chapter 6, and twelve maps are such built respectively for the twelve standard drive cycles. If a quasi-optimal λ map from a drive cycle is applied to the same drive cycle, the gap to DP results in term of equivalent cost is 5% on average over the twelve standard drive cycles. However, as drive cycle as a 'speed vs. time' profile is not predictable, a universal map is built by fitting the λ maps into one map which can be used for arbitrary drive cycles. The gap to DP results by this universal map is 14% on average over the 12 drive cycles, with 30% improvement over CD-CS strategy. Large scale simulation is adopted to test the robustness and performance of ECMS with this universal map and 4719 drive cycles are generated from real-world driving diary. The results show great robustness and average 9.3% cost reduction compared to CD-CS strategy.

In all the simulation in Chapter 6, a known distance to the known destination is assumed. Chapter 7 discusses the possibility of predicting the destination using historical data and distance variation information. Scenario study shows the destination can be predicted after 50% of the drive cycle unless there are multiple destination candidates on the same driving direction. Another issue of this algorithm is that it cannot predict a new destination, so this meth od is limited to the drivers who have regular daily routine and can estimate frequent destinations like home and work.

Also observed in Chapter 6 is that the λ maps are similar for a same driving pattern while the drive cycle profiles remain quite different. If the driving pattern can be recognized, the

performance of ECMS with the universal λ map can be further improved. In chapter 8, a fuzzy logic controller is designed for driving pattern recognition using speed limit and average speed information. A known destination is assumed, and a route can be planned by GPS which also knows the speed limit information on each segment of the route. Another fuzzy logic controller is designed to recognize aggressive drivers and compensate accordingly. The twelve standard drive cycles in Chapter 6 are divided into two groups: six in training group and the other six in test group. The fuzzy logic membership functions and rules of both of the fuzzy logic controllers are trained by the training group, and the simulation results on the test group shows an additional 7% cost reduction in average over ECMS with the universal λ map, and the gap to ECMS with quasi-optimal λ maps is only 2% on average. Further test on real world drive cycles that include all the driving patterns shows a 6% additional cost reduction compared to ECMS without the universal λ map. The gap to the DP results is 8% in average compared to an average 14% gap by ECMS with the universal λ map and an average 35% gap by CD-CS strategy.

Future work can be carried on in the following areas:

- Include the battery degradation to the cost function. In some conditions such as high temperature, it may not be ideal to drain the battery energy at each destination due to the accelerated battery degradation. The potential cost of battery replacement should be considered. In [87] mathematical models of battery life degradation due to temperature and daily depth of discharge are given, and the challenge is to integrate the cost of battery life degradation into the cost function of ECMS.

- Consider the impact of dynamic charging to the equivalent factor λ. Inductive power transfer is an emerging technology and has already been applied to vehicle charging [88-90]. If the vehicle is able to charge while it's being driven, the cost function λ will change dramatically. The question then becomes how to optimally utilize the dynamic charging: is it better to charge the battery than to sustain the SOC with dynamic charging? To answer this question the battery degradation model is again necessary.

- Consider the application of this DP-ECMS combined method to other optimization problems. For example, more and more vehicles are charge enabled, increasing the demand for charging stations. With multiple power sources (grid, energy storage, solar and wind) in play, the optimization problem becomes similar that of the (P)HEV and the key is to define a meaningful cost function. Even more generally, for a smart grid where multiple power sources are available and the load command can be estimated, λ maps can be generated for different load patterns using DP results.

REFERENCES

[1] Pi-Innovo. (2014). Engine Emission Control. Available: http://www.pi-innovo.com/engineering/engine-emission-control

[2] Wikipedia. (2014). Electric car. Available: http://en.wikipedia.org/wiki/Electric_car

[3] M. Ehsani, Y. Gao, and A. Emadi, Modern electric, hybrid electric, and fuel cell vehicles: fundamentals, theory, and design: CRC press, 2009.

[4] C. C. Lin, H. Peng, J. W. Grizzle, and J. M. Kang, "Power management strategy for a parallel hybrid electric truck," Ieee Transactions on Control Systems Technology, vol. 11, pp. 839-849, Nov 2003.

[5] R. Wang and S. M. Lukic, "Dynamic programming technique in hybrid electric vehicle optimization," in Electric Vehicle Conference (IEVC), 2012 IEEE International, 2012, pp. 1-8.

[6] P. Pisu and G. Rizzoni, "A comparative study of supervisory control strategies for hybrid electric vehicles," Control Systems Technology, IEEE Transactions on, vol. 15, pp. 506-518, 2007.

[7] G. Paganelli, M. Tateno, A. Brahma, G. Rizzoni, and Y. Guezennec, "Control development for a hybrid-electric sport-utility vehicle: strategy, implementation and field test results," in American Control Conference, 2001. Proceedings of the 2001, 2001, pp. 5064-5069.

[8] L. Serrao, S. Onori, and G. Rizzoni, "ECMS as a realization of Pontryagin's minimum principle for HEV control," in Proceedings of the 2009 conference on American Control Conference, 2009, pp. 3964-3969.

[9] C. Musardo, G. Rizzoni, Y. Guezennec, and B. Staccia, "A-ECMS: An adaptive algorithm for hybrid electric vehicle energy management," European Journal of Control, vol. 11, pp. 509-524, 2005.

[10] G. Paganelli, S. Delprat, T.-M. Guerra, J. Rimaux, and J.-J. Santin, "Equivalent consumption minimization strategy for parallel hybrid powertrains," in Vehicular Technology Conference, 2002. VTC Spring 2002. IEEE 55th, 2002, pp. 2076-2081.

[11] A. Sciarretta, M. Back, and L. Guzzella, "Optimal control of parallel hybrid electric vehicles," Control Systems Technology, IEEE Transactions on, vol. 12, pp. 352-363, 2004.

[12] C. Zhang, A. Vahidi, P. Pisu, X. Li, and K. Tennant, "Role of terrain preview in energy management of hybrid electric vehicles," Vehicular Technology, IEEE Transactions on, vol. 59, pp. 1139-1147, 2010.

[13] Wikipedia. (2014). Miles per gallon gasoline equivalent. Available: http://en.wikipedia.org/wiki/Miles_per_gallon_gasoline_equivalent

[14] F. R. Salmasi, "Control Strategies for Hybrid Electric Vehicles: Evolution, Classification, Comparison, and Future Trends," Vehicular Technology, IEEE Transactions on, vol. 56, pp. 2393-2404, 2007.

[15] Y. Gurkaynak, A. Khaligh, and A. Emadi, "State of the art power management algorithms for hybrid electric vehicles," in Vehicle Power and Propulsion Conference, 2009. VPPC'09. IEEE, 2009, pp. 388-394.

[16] A. Sciarretta and L. Guzzella, "Control of hybrid electric vehicles," Control systems, IEEE, vol. 27, pp. 60-70, 2007.

[17] C. Shumei, Z. Wei, and T. Likun, "Control Strategy for Double Shaft Parallel Hybrid Electric Vehicle," in Vehicle Power and Propulsion Conference, 2006. VPPC'06. IEEE, 2006, pp. 1-4.

[18] M. Mohebbi, M. Charkhgard, and M. Farrokhi, "Optimal neuro-fuzzy control of parallel hybrid electric vehicles," in Vehicle Power and Propulsion, 2005 IEEE Conference, 2005, pp. 26-30.

[19] M. Mohammadian and M. Bathaee, "Motion control for hybrid electric vehicle," in Power Electronics and Motion Control Conference, 2004. IPEMC 2004. The 4th International, 2004, pp. 1490-1494.

[20] A. M. Phillips, M. Jankovic, and K. E. Bailey, "Vehicle system controller design for a hybrid electric vehicle," in Control Applications, 2000. Proceedings of the 2000 IEEE International Conference on, 2000, pp. 297-302.

[21] L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," Systems, Man and Cybernetics, IEEE Transactions on, pp. 28-44, 1973.

[22] Wikipedia. (2014). Fuzzy control system. Available: http://en.wikipedia.org/wiki/Fuzzy_control_system

[23] H. D. Lee and S. K. Sul, "Fuzzy-logic-based torque control strategy for parallel-type hybrid electric vehicle," Ieee Transactions on Industrial Electronics, vol. 45, pp. 625-632, Aug 1998.

[24] S. Delprat, J. Lauber, T. M. Guerra, and J. Rimaux, "Control of a parallel hybrid powertrain: Optimal control," Ieee Transactions on Vehicular Technology, vol. 53, pp. 872-881, May 2004.

[25] E. D. Tate and S. P. Boyd, "Finding ultimate limits of performance for hybrid electric vehicles," SAE transactions, vol. 109, pp. 2437-2448, 2000.

[26] A. Piccolo, L. Ippolito, and A. Vaccaro, "Optimisation of energy flow management in hybrid electric vehicles via genetic algorithms," in Advanced Intelligent Mechatronics, 2001. Proceedings. 2001 IEEE/ASME International Conference on, 2001, pp. 434-439.

[27] J. Moreno, M. E. Ortúzar, and L. Dixon, "Energy-management system for a hybrid electric vehicle, using ultracapacitors and neural networks," Industrial Electronics, IEEE Transactions on, vol. 53, pp. 614-623, 2006.

[28] J. Wu, C.-H. Zhang, and N.-X. Cui, "PSO algorithm-based parameter optimization for HEV powertrain and its control strategy," International Journal of Automotive Technology, vol. 9, pp. 53-59, 2008.

[29] P. Pisu and G. Rizzoni, "H∞ control for hybrid electric vehicles," in Decision and Control, 2004. CDC. 43rd IEEE Conference on, 2004, pp. 3497-3502.

[30] C.-C. Lin, H. Peng, and J. Grizzle, "A stochastic control strategy for hybrid electric vehicles," in American Control Conference, 2004. Proceedings of the 2004, 2004, pp. 4710-4715.

[31] S. I. Jeon, S. T. Jo, Y. I. Park, and J. M. Lee, "Multi-mode driving control of a parallel hybrid electric vehicle using driving pattern recognition," Journal of Dynamic Systems Measurement and Control-Transactions of the Asme, vol. 124, pp. 141-149, Mar 2002.

[32] Y. L. Murphey, Z. Chen, L. Kiliaris, J. Park, M. Kuang, A. Masrur, et al., "Neural learning of driving environment prediction for vehicle power management," in Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, 2008, pp. 3755-3761.

[33] T. Yi, Z. Xin, Z. Liang, and Z. Xinn, "Intelligent energy management based on driving cycle identification using fuzzy neural network," in Computational Intelligence and Design, 2009. ISCID'09. Second International Symposium on, 2009, pp. 501-504.

[34] M. Hajimiri and F. Salmasi, "A fuzzy energy management strategy for series hybrid electric vehicle with predictive control and durability extension of the battery," in Electric and Hybrid Vehicles, 2006. ICEHV'06. IEEE Conference on, 2006, pp. 1-5.

[35] X. Huang, Y. Tan, and X. G. He, "An Intelligent Multifeature Statistical Approach for the Discrimination of Driving Conditions of a Hybrid Electric Vehicle," Ieee Transactions on Intelligent Transportation Systems, vol. 12, pp. 453-465, Jun 2011.

[36] Z. Chen, L. Kiliaris, Y. L. Murphey, and M. Masrur, "Intelligent power management in SHEV based on roadway type and traffic congestion levels," in Vehicle Power and Propulsion Conference, 2009. VPPC'09. IEEE, 2009, pp. 915-920.

[37] C.-C. Lin, S. Jeon, H. Peng, and J. Moo Lee, "Driving pattern recognition for control of hybrid electric trucks," Vehicle System Dynamics, vol. 42, pp. 41-58, 2004.

[38] M. Montazeri-Gh, A. Ahmadi, and M. Asadi, "Driving condition recognition for genetic-fuzzy HEV Control," in Genetic and Evolving Systems, 2008. GEFS 2008. 3rd International Workshop on, 2008, pp. 65-70.

[39] Wikipedia. (2014). Plug-in hybrid. Available: http://en.wikipedia.org/wiki/Plug-in_hybrid

[40] A. Rousseau. (2008). PHEV vehicle level control strategy summary. Available: http://www.transportation.anl.gov/pdfs/HV/552.pdf

[41] C. Zhang and A. Vahid, "Real-time optimal control of plug-in hybrid vehicles with trip preview," in American Control Conference (ACC), 2010, 2010, pp. 6917-6922.

[42] C. Zhang, A. Vahidi, X. Li, and D. Essenmacher, "Role of trip information preview in fuel economy of plug-in hybrid vehicles," in ASME 2009 Dynamic Systems and Control Conference, 2009, pp. 253-258.

[43] S. G. Li, S. Sharkh, F. C. Walsh, and C.-N. Zhang, "Energy and battery management of a plug-in series hybrid electric vehicle using fuzzy logic," Vehicular Technology, IEEE Transactions on, vol. 60, pp. 3571-3585, 2011.

[44] S. G. Wirasingha and A. Emadi, "Classification and review of control strategies for plug-in hybrid electric vehicles," vehicular Technology, IEEE Transactions on, vol. 60, pp. 111-122, 2011.

[45] H. Banvait, X. Lin, S. Anwar, and Y. Chen, "Plug-in hybrid electric vehicle energy management system using particle swarm optimization," World Electric Vehicle Association Journal, vol. 3, 2009.

[46] Q. Gong, Y. Li, and Z.-R. Peng, "Trip based power management of plug-in hybrid electric vehicle with two-scale dynamic programming," in Vehicle Power and Propulsion Conference, 2007. VPPC 2007. IEEE, 2007, pp. 12-19.

[47] M. J. Gielniak and Z. J. Shen, "Power management strategy based on game theory for fuel cell hybrid electric vehicles," in Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th, 2004, pp. 4422-4426.

[48] S. J. Moura, H. K. Fathy, D. S. Callaway, and J. L. Stein, "A stochastic optimal control approach for power management in plug-in hybrid electric vehicles," Control Systems Technology, IEEE Transactions on, vol. 19, pp. 545-555, 2011.

[49] C. Zhang and A. Vahidi, "Route preview in energy management of plug-in hybrid vehicles," Control Systems Technology, IEEE Transactions on, vol. 20, pp. 546-553, 2012.

[50] A. Santos, N. McGuckin, H. Y. Nakamoto, D. Gray, and S. Liss, "Summary of travel trends: 2009 national household travel survey," 2011.

[51] R. Hariharan and K. Toyama, "Project Lachesis: parsing and modeling location histories," in Geographic Information Science, ed: Springer, 2004, pp. 106-124.

[52] D. Ashbrook and T. Starner, "Using GPS to learn significant locations and predict movement across multiple users," Personal and Ubiquitous Computing, vol. 7, pp. 275-286, Oct 2003.

[53] J. Krumm and E. Horvitz, "Predestination: Inferring destinations from partial trajectories," in UbiComp 2006: Ubiquitous Computing, ed: Springer, 2006, pp. 243-260.

[54] B. D. Ziebart, A. L. Maas, A. K. Dey, and J. A. Bagnell, "Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior," in Proceedings of the 10th international conference on Ubiquitous computing, 2008, pp. 322-331.

[55] L. Liao, D. J. Patterson, D. Fox, and H. Kautz, "Learning and inferring transportation routines," Artificial Intelligence, vol. 171, pp. 311-331, Apr 2007.

[56] V. Gogate, R. Dechter, B. Bidyuk, C. Rindt, and J. Marca, "Modeling transportation routines using hybrid dynamic mixed networks," arXiv preprint arXiv:1207.1384, 2012.

[57] D. P. Bertsekas, Dynamic programming and optimal control, third edition: Athena Scientific Belmont, MA, 2005.

[58] I. Rosu, "The Bellman Principle of Optimality," Availiable at: http://faculty. chicagogsb. edu/ioanid. rosu/research/notes/bellman. pdf, 2002.

[59] L. Del Re, F. Allgöwer, L. Glielmo, C. Guardiola, and I. Kolmanovsky, Automotive model predictive control: models, methods and applications: Springer, 2010.

[60] A. Szumanowski, Hybrid electric vehicle drives design: Publishing and Printing House of the Institute for Sustainable Technologies-NRI, 2006.

[61] T. Markel, A. Brooker, T. Hendricks, V. Johnson, K. Kelly, B. Kramer, et al., "ADVISOR: a systems analysis tool for advanced vehicle modeling," Journal of power sources, vol. 110, pp. 255-266, 2002.

[62] C.-C. Lin, H. Peng, J. W. Grizzle, and J.-M. Kang, "Power management strategy for a parallel hybrid electric truck," Control Systems Technology, IEEE Transactions on, vol. 11, pp. 839-849, 2003.

[63] Wikipedia. (2014). Pontryagin's minimum principle. Available: http://en.wikipedia.org/wiki/Pontryagin's_minimum_principle

[64] S. M. LaValle, Planning algorithms: Cambridge university press, 2006.

[65] D. E. Kirk, Optimal control theory: an introduction: Courier Dover Publications, 2012.

[66] M. D. Galus, R. A. Waraich, F. Noembrini, K. Steurs, G. Georges, K. Boulouchos, et al., "Integrating power systems, transport systems and vehicle technology for electric mobility impact assessment and efficient control," Smart Grid, IEEE Transactions on, vol. 3, pp. 934-949, 2012.

[67] MATSim-T. (2008). Multi Agent Transportation Simulation Toolkit. Available: http://www.matsim.org

[68] Q. Gong, Y. Li, and Z.-R. Peng, "Trip-Based Optimal Power Management of Plug-in Hybrid Electric Vehicles," Vehicular Technology, IEEE Transactions on, vol. 57, pp. 3393-3401, 2008.

[69] I. M. Berry, "The effects of driving style and vehicle performance on the real-world fuel consumption of US light-duty vehicles," Masters, Massachusetts Institute of Technology, 2010.

[70] MathWorks. (2014). Extreme Value Distribution. Available: http://www.mathworks.com/help/stats/extreme-value-distribution.html

[71] L. Liao, D. J. Patterson, D. Fox, and H. Kautz, "Learning and inferring transportation routines," Artificial Intelligence, vol. 171, pp. 311-331, 2007.

[72] J. Dai, "Isolated word recognition using Markov chain models," Speech and Audio Processing, IEEE Transactions on, vol. 3, pp. 458-463, 1995.

[73] M. Ford, "Voice Activated Navigation," ed, 2013. Available at https://www.youtube.com/watch?feature=player_embedded&v=YHFAlsp89s0.

[74] P. BMW, "BMW Voice Command - Navigation," ed, 2011.

[75] D. Newcomb, "Kia Soul's Voice Activation System," ed, 2013.

[76] H. Hyundai, "2013 Hyundai Tucson"s Voice Activated Navigation System - Hallmark Hyundai - Franklin, TN," ed, 2012.

[77] How is Speed Calculated? Available: http://www.cotrip.org/helpSpeedCalculation.htm

[78] S. Zorrofi, S. Filizadeh, and P. Zanetel, "A simulation study of the impact of driving patterns and driver behavior on fuel economy of hybrid transit buses," in Vehicle Power and Propulsion Conference, 2009. VPPC '09. IEEE, 2009, pp. 572-577.

[79] R. Wang and S. M. Lukic, "Review of driving conditions prediction and driving style recognition based control algorithms for hybrid electric vehicles," in Vehicle Power and Propulsion Conference (VPPC), 2011 IEEE, 2011, pp. 1-7.

[80] K. Igarashi, C. Miyajima, K. Itou, K. Takeda, F. Itakura, and H. Abut, "Biometric identification using driving behavioral signals," in Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on, 2004, pp. 65-68.

[81] R. Langari and J.-S. Won, "Intelligent energy management agent for a parallel hybrid vehicle-part I: system architecture and design of the driving situation identification process," Vehicular Technology, IEEE Transactions on, vol. 54, pp. 925-934, 2005.

[82] Y. L. Murphey, R. Milton, and L. Kiliaris, "Driver's style classification using jerk analysis," in Computational Intelligence in Vehicles and Vehicular Systems, 2009. CIVVS'09. IEEE Workshop on, 2009, pp. 23-28.

[83] U. S. E. P. Agent. Dynamometer Drive Schedules. Available: http://www.epa.gov/nvfel/testing/dynamometer.htm

[84] R. F. B. Tony S. Abbo, Leanna M. Belluz, Robert Bucholc, etc, "Methods and Practices for Setting Speed Limits: An Informational Report," FHWA-SA-12-004, 2012.

[85] M. R. P. Associates, "Synthesis of Speed Zoning Practices," FHWA/RD-85/096.

[86] H. C. Frey, B. Boroujeni, B. Liu, J. Hu, W. Jiao, B. Graver, et al., "Field Measurements of 1996 to 2013 Model Year Light Duty Gasoline Vehicles," in

Proceedings, 106th Annual Conference and Exhibition, Air & Waste Management Association, Chicago, Illinois, 2013.

[87] A. Hoke, A. Brissette, D. Maksimovic, A. Pratt, and K. Smith, "Electric vehicle charge optimization including effects of lithium-ion battery degradation," in Vehicle Power and Propulsion Conference (VPPC), 2011 IEEE, 2011, pp. 1-8.

[88] H. H. Wu, A. Gilchrist, K. D. Sealy, and D. Bronson, "A high efficiency 5 kW inductive charger for EVs using dual side control," Industrial Informatics, IEEE Transactions on, vol. 8, pp. 585-595, 2012.

[89] M. Etemadrezaei and S. M. Lukic, "Optimization of foil conductor layout in inductive power transfer system resonators," in Energy Conversion Congress and Exposition (ECCE), 2014 IEEE, 2014, pp. 876-883.

[90] J. Boys, G. Covic, and A. W. Green, "Stability and control of inductively coupled power transfer systems," IEE Proceedings-Electric Power Applications, vol. 147, pp. 37-43, 2000.