This is to certify that the
thesis entitled

STRATEGIES FOR OPTIMIZING HYBRID ELECTRIC VEHICLE POWERTRAINS AND PROGRESSIVELY CRUSHING RAILS

presented by

NATHAN CHASE

has been accepted towards fulfillment
of the requirements for the

M.S. degree in Mechanical Engineering

Ron Averill

Major Professor's Signature

5/10/2010

Date

STRATEGIES FOR OPTIMIZING HYBRID ELECTRIC VEHICLE POWERTRAINS AND PROGRESSIVELY CRUSHING RAILS

By

Nathan Chase

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Mechanical Engineering

2010

ABSTRACT

STRATEGIES FOR OPTIMIZING HYBRID ELECTRIC VEHICLE POWERTRAINS AND PROGRESSIVELY CRUSHING RAILS

By

Nathan Chase

In application of automated design optimization methods to complex problems, often the biggest challenge is to identify an optimization problem statement that will lead to a design with the desired performance in an efficient and effective manner. In the current study, the optimization statements of two challenging problems are investigated and new approaches are introduced to improve the performance of the designs as well as the optimization process.

Fuel efficient and environmentally friendly ground vehicles require a careful balance among competing goals for fuel economy, performance, and emissions. Several optimization approaches are investigated for a series hybrid bus to find the best overall combination of engine size, battery pack, electric motor and generator with minimum fuel consumption under specified performance criteria. An alternative to a common optimization approach for HEV's is proposed and shown to yield results equivalent to other common approaches but with greater efficiency for the problem posed.

To increase robustness of the crush mode and to decrease repair costs after a crash, it is desirable for front and rear structural rails in an automotive vehicle to crush progressively - where crush initiates near the tip of the rail and progresses rearward in a controlled fashion. A new strategy is investigated here to achieve progressively crushing designs during an automated design optimization study. It is demonstrated that high performing designs with progressive crush can be obtained using the proposed approach.

To my parents for their encouragement, direction, and invaluable advice. And to my wife for her patience, support, and calming influence. Without them this success would surely have been unattainable.

ACKNOWLEDGEMENTS

My sincere appreciation to my advisor Dr. Ronald Averill for his guidance and vision in improving the way optimization is performed and in my understanding of it. Also I would like to thank my other thesis committee members Dr. Erik Goodman and Dr. Farhang Pourboghrat. Their support is much appreciated.

I also would like to thank Red Cedar Technology for their support in this endeavor and specifically Mr. Ranny Sidhu. His advice and aid greatly helped in the development of the strategies developed for this thesis.

Other key members in the development of this thesis include: Kaushik Sinha and his aid with the HEV models, as well as the High Performance Computing Center of Michigan State University (HPCC) and its staff. Without their help and resources this thesis would not be as complete.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Optimization is becoming an essential tool in computer aided engineering, specifically in the design process. The need for quicker, better solutions in the global economy that is the reality of today's world, has made it insufficient to rely solely on an engineer's intuition and experience. Optimization allows superior designs to be found quicker and more robustly in a realm that is not limited to the experience of a specific engineer working on a project, but rather the search capabilities of the optimization software in use, and the way the optimization problem is posed.

In the past, optimization has been slow to gain popularity in industry due to the perceived complexity associated with it. The methods used to perform the optimization search were largely dependent upon the problem at hand and it was common for one to need to be an expert in not only the problem being solved but also the search methods used to solve the problem. Today however, some of the commercial optimization software packages are improved enough, and user friendly enough, that being an expert in optimization is no longer a requirement. There are search methods available (e.g., SHERPA [29]) that require no knowledge by the user as to the details of the search strategy employed. Whereas in the past, the user had to specify values of tuning parameters for a given method, methods like SHERPA require no tuning parameters, only the number of evaluations allowed. In addition, modern hybrid and adaptive methods work on broad classes of problems, so the user does not need to know the nature of design landscape of the problem being solved a priori. These methods adapt to the

landscape of the problem being solved by modifying which search strategies are used, and how they are used, at various stages of the search process..

Today, it follows that the complexity of search algorithms and their use is no longer a primary issue. The issue at hand now becomes how to best define the optimization problem being solved in order to arrive at the desired results. For most problems this includes only a few decisions by the engineer. He must decide how best to setup the analysis model of the system he is trying to optimize; this may be deciding on how best to parameterize a CAD model or choosing the most appropriate settings for a numerical model, among other things. He then must decide what responses are of interest to him, and what performance criteria are to be used during the optimization. Finally, he must decide the allowable design space for his design variables, and what is considered an acceptable design.

For some applications, the above three decisions by the engineer regarding the optimization statement may not be sufficient. The way the optimization should be defined may not be clear to the engineer in that it may be defined in multiple ways. Each definition possibility may lead the optimization in a different direction and produce different optimal designs. Defining an optimization problem in one way may lead the optimization to a design that meets the requirements as defined, but that does not meet requirements that were undefined – ones that the user may not have thought of before the optimization was performed, or didn't know how to define. Once the engineer knows how to properly define the optimization problem for the specific application he is interested in, however, not only will he get superior optimal designs, but he also will have a better understanding of how to define optimization problems for similar applications.

It is the goal of this study to investigate optimization strategies for two specific applications for which the problem statement is not obvious, and for which multiple definitions are possible – each that give vastly different results. It is the hope that the optimization strategies posed here will provide engineers in the respective fields a "template-like" problem statement to be used in future optimization studies. In addition, it is hoped that this work will demonstrate the importance of optimization strategies for complex applications and provide the framework for more application problem definitions, perhaps even in other fields, to be explored in the future.

CHAPTER 2: INTRODUCTION TO HYBRID ELECTRIC VEHICLES

2.1 Background

Never before has there been such a concern over the future of the transportation industry throughout the world. The recent escalation in concerns over global warming and greenhouse gases, coupled with the depleting supply of the world's oil reserves and the growing instability in the regions of the world where these reserves exist, has caused a heightened awareness in the insufficiency of the standard internal combustion engine (ICE). It has never been as apparent as in today's modern world, that the current models for automobiles and other vehicles that rely on natural reserves are insufficient for a responsible, realistic future.

Currently there are many proposals as to the direction the world should take in changing the dependence on oil. Alternate fuel systems such as fuel cells and bio-diesels are strong contenders for the next wave of vehicles. However these vehicles, should they be implemented, would require a major overhaul in the economic and standard landscape the world has become accustomed to. The shifting agricultural landscape required for bio-diesels make them an unlikely alternative, while the instability and uncertainty of fuel cells also make them unlikely in the near future as a replacement for the standard IC engine. Electric vehicles (EV's) came into the mainstream of America over a decade ago. Their prompt failure and unpopularity make them an unlikely alternative for most of the world anytime soon.

Hybrid electric vehicles (HEV's) are perhaps the most promising substitute to conventional vehicles in the near future. HEV's utilize multiple energy sources; in terms of this thesis, these energy sources will refer to an energy storage system (such as a battery or ultra-capacitor), and also an engine. These multiple energy sources allow for the optimal operation of the engine, thus reducing fuel consumption and emissions. This proposed alternative to standard vehicles helps reduce the consumption of oil reserves, while also reducing the emissions of greenhouse gases and not drastically changing the current landscape of the transportation industry.

The design of hybrid electric vehicles (HEV's) requires complex optimizations to account for the intricate interactions among the components, control strategies and the resulting effects on performance. Due to the size of HEV models and the complexity of interactions among the many components, it is challenging to properly define an optimization problem statement that invokes a final design with the desired performance characteristics. The optimization problem statement definition greatly affects the performance of the optimized HEV design, as well as the implications of the design. While this may seem obvious, the proper problem statement for achieving a particular performance goal is not always so clear.

In the first section of this thesis, the main characteristics of two optimization strategies that have been shown to be efficient in optimizing hybrid electric vehicles are merged into a single and more robust strategy that is shown to have several advantages in terms of design performance. The main purpose here is to explore these various strategies and associated optimization problem statements and their effects on the resulting hybrid vehicle design. As a consequence of this study, a practical and efficient approach is

developed for sizing the components and defining the system control of HEV's to create efficient, reliable vehicles.

2.2 Literature Review

2.2.1 Overview

Hybrid electric vehicles (HEV's) include an engine as a fuel converter or irreversible prime mover, an electric prime mover, and sometimes a generator [1-2]. They are typically classified into three different categories: series, parallel, or combination. Series HEV's utilize an internal combustion engine (ICE) as an auxiliary power unit to extend the driving range of a purely electric vehicle. These vehicles use a generator to convert mechanical energy produced by the ICE into electricity that can either charge the battery or feed the drive motor directly (as shown by Figure 2.2.1). The use of a clutch in series HEV's is unnecessary. Parallel HEV's do require a clutch, and utilize both the ICE and electric motor to supply tractive power either individually or in combination (see Figure 2.2.2). Combined HEV's utilize both a mechanical and electric link, with two distinct electric machines (as shown by Figure 2.2.3 and Figure 2.2.4).

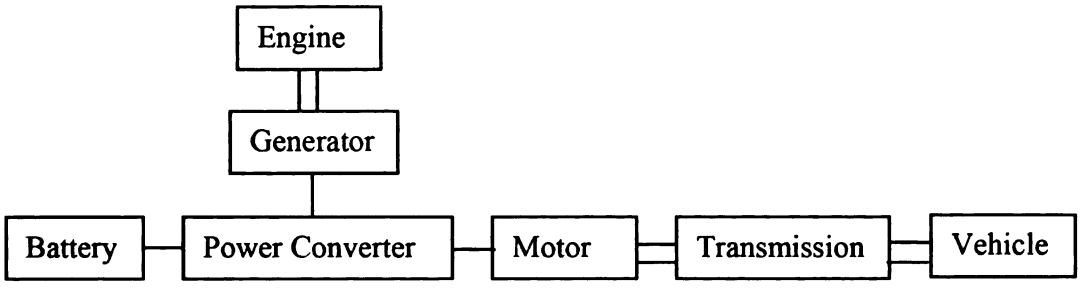


Figure 2.2.1. Series hybrid electric vehicle configuration.

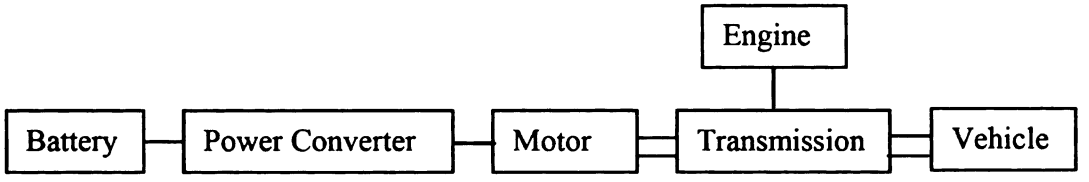


Figure 2.2.2. Parallel hybrid electric vehicle configuration.

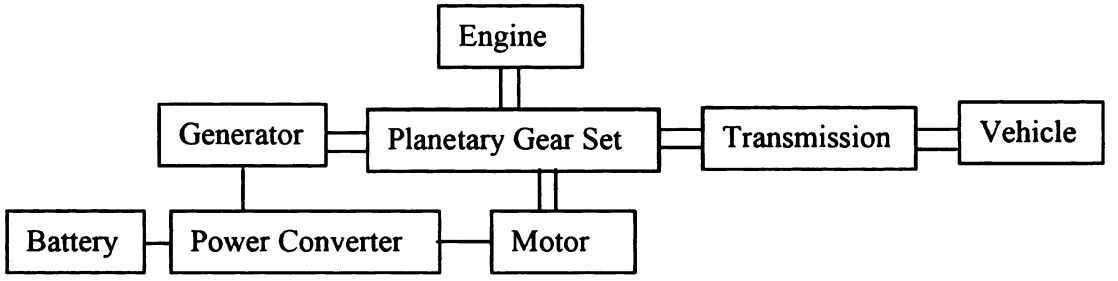


Figure 2.2.3. Combined hybrid electric vehicle configuration A.

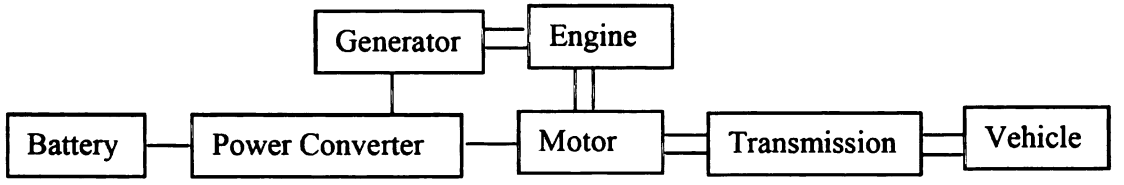


Figure 2.2.4. Combined hybrid electric vehicle configuration B.

Guzzella and Sciarretta [1] as well as Ehsani, Gao, and Miller [3] identify HEV's as the most promising alternatives to conventional vehicles. The high energy density of petroleum fuel and the convenience of current fueling systems make for a long vehicle operating range, easy refueling, and a smooth transition from conventional vehicles, while the design of the HEV components provide an improved fuel economy and reduced emissions. HEV's can utilize a downsized engine while still fulfilling the maximum power requirements of the vehicle. They possess the ability to recover energy during braking (regenerative braking), as opposed to dissipating it in conventional vehicles that use friction braking. HEV's can optimize the energy distribution between their prime movers for maximum efficiency, while also eliminating idle fuel consumption by turning off the engine when no power is required and eliminating clutching losses by engaging the engine only when the speeds match. In addition, due to the optimized operation of the internal combustion engine, the vehicles will need less maintenance than conventional vehicles currently require, as described by Chan [4].

While the benefits of HEV's over conventional vehicles are substantial, there are also some drawbacks as discussed by Chan [4]. Hybrid electric vehicles are typically 10-30% heavier than conventional vehicles, while costing more due to added motors, energy storage systems, and power converters. In addition, reliability and warranty related issues are of concern due to the lack of mechanic training in the electrical aspect of their design. Safety concerns with HEV's still need to be addressed due to the presence of high voltage devices, and the effect of electromagnetic fields on HEV's is not fully understood.

2.2.2 Modeling Tools

The need for accurate modeling and simulation of hybrid electric vehicles has been discussed in [5-7]. Gao, et al [5] present an overview of some currently available methods, and place special emphasis on four modeling tools: ADVISOR [8-9], PSAT [10], PSIM [11], and VTB [12]. ADVISOR is a steady state model that achieves fast computation but does not accurately resolve dynamic effects. PSAT is a quasi-steady state model. PSIM and VTB are dynamic models, in which time integration is performed. The tool QSS toolbox is introduced by Guzzella, and Amstutz [13] for optimizing and simulating hybrid powertrains. QSS toolbox uses a backward approach that is faster than typical approaches due to no feedback loops being used, and rather large time steps being utilized. There are significant assumptions in this tool, which the authors contend are small and have negligible effects over an entire drive cycle.

In the current study, ADVISOR was chosen as the modeling tool. An overview of ADVISOR (advanced vehicle simulator) is provided by Wipke, Cuddy, and Burch [8]. ADVISOR utilizes a combined forward/backward simulation approach to evaluate a vehicle's performance. It is used in conjunction with Matlab to simulate the behavior of Hybrid Electric Vehicles (HEV's) [9]. An ADVISOR model consists of multiple components that are evaluated in a 1-dimensional manner, often using look-up tables and efficiency maps to provide fast computation with the backward facing approach. The force required to accelerate the vehicle through a time step is calculated directly from the required speed trace. This force is then translated into torque, etc. This process is carried backward through the HEV drive train, component by component until the entire system is solved. This backward facing approach is typically more efficient than the forward

facing approach for force calculation. The forward facing aspect of the ADVISOR solution uses a driver model that considers the required speed and the present speed to develop appropriate throttle and brake commands, which are in turn translated into torque, etc. The forward facing approach better takes into account dynamic effects and maximum effort accelerations, and enables the development of vehicle controllers. By combining both of these approaches, ADVISOR is able to solve the problem efficiently, while achieving the desired accuracy. Chapter 3 provides more details on how ADVISOR works.

2.2.3 Optimization

Three types of HEV optimization are identified by Guzzella and Sciarretta [1]. The first of these is a structural optimization in which the best possible power train structure is sought. The second type is parametric optimization to find the best possible parameter values for a fixed power train structure (sizing of components). The third type of optimization focuses on the control system, where the best possible supervisory control strategy for a given structure is sought. While these three types of optimization studies are not necessarily independent, each one represents a meaningful and tractable suboptimization problem within the overall design process. The current study focuses primarily on the second type (sizing) while also addressing the role of the control strategy on the optimization of HEV's.

The commercial software HEEDS [29] was used as the optimization tool for the current study. The hybrid and adaptive search method SHERPA [29] was employed exclusively here because it has been shown to perform well on a wide variety of

problems, including multi-modal, noisy, discontinuous problems. Oh, et al [14] demonstrated that local search methods such as Sequential Quadratic Programming (SQP) can be used to find marginal improvement in HEV designs. More often, however, it has been found that local search methods are not ideal for the optimization of HEV's. Montazeri-Gh, et al [15], showed that HEV optimization problems are highly non-linear, discontinuous, and multi-modal (i.e., multiple local optima exist). Therefore gradient-based search methods, which require the existence of at least first-order derivatives at each design point, are often not the ideal choice for HEV design problems. This is supported by Piccolo, et al [16] who concluded that a genetic algorithm is an efficient search algorithm for identifying optimized control strategy parameters for a design that meets performance criteria while having low fuel consumption. Gao and Porandla [17] also identified genetic algorithms as a potentially efficient search algorithm for HEV optimization problems. While optimizing an HEV parallel powertrain with PSAT as the simulation tool, these investigators benchmarked the use of simulated annealing (SA), genetic algorithms (GA), and the DIRECT technique to perform the optimization. DIRECT is a modification to the Lipschitzian approach in which the design space is represented by a series of shrinking hypercubes for high performing regions of the design space. It was observed that simulated annealing outperformed the genetic algorithm and modified Lipschitzian approach in terms of performance for the optimization problem posed. The response functions associated with HEV design problems were found to be typically multi-modal and sometimes noisy and discontinuous. It was suggested that stochastic algorithms (such as SA and GA) and deterministic algorithms (such as

DIRECT) were insufficient in finding the global optimum as efficiently as a hybrid method that mixes stochastic and deterministic algorithms.

It was observed by Montazeri-Gh, et al [15] that very different optimized designs were obtained using different driving cycles. Three different driving cycles were explored for the same optimization problem, resulting in three very different optimized configurations for the control strategy of the HEV. The effects of drive cycle were also studied by Markel and Wipke [18-19], where it was found that the drive cycle greatly affected the optimized sizing of components in the HEV. It was found that more aggressive drive cycles yielded designs with smaller battery packs, while less aggressive drive cycles yielded optimized designs with larger battery packs. It was demonstrated that the drive cycle not only affects the component sizing of the optimized HEV designs, but also the energy management strategy of the HEV.

Assanis, et al [20] and Fellini, et al [21] integrated ADVISOR and Turbo Diesel Engine Simulation Program (TDES) to give greater accuracy in the engine maps being used for altered hybrid designs. Fellini, et al [21] described an optimization study using an Object Oriented-CORBA-based environment for the analysis, to optimally size the engine, motor, and battery of the vehicle. They concluded that optimized engine size is insensitive to variations in the drag coefficient, shell mass, battery efficiency, and drive ratio, whereas the battery size and optimized motor are sensitive to these parameters. This was shown through a sensitivity study following the attainment of optimized values via an optimization study.

Moore [22] suggested guidelines for specifying performance criteria when designing components and control strategies for HEV's. These guidelines specify that

the vehicle must be capable of: launching at a given grade road, accelerating to multiple speeds under given timeframes, maintaining gradability for a specified time and speed, achieving a maximum velocity, achieving a minimum fuel economy, meeting emissions criteria, and being capable of traveling a given distance in a mixed urban and highway driving environment. In addition, Moore suggested guidelines that are not necessarily pertinent to optimization of HEV's, but rather to HEV design in general. These guidelines include: having refueling times comparable to conventional vehicles, having noise, vibration, and harshness in travel less than current sedans, having the ability to be unattended for a given time below freezing temperatures, and meeting safety requirements.

A common approach to designing HEV's is to consider some type of battery state-of-charge (SOC) correction. SOC corrections are routines or methods designed to ensure that over the timeframe considered the initial SOC and final SOC are the same. This provides a meaningful way to compare designs and ensures that a design is less dependent upon the initial SOC. Wayne, et al [23], describe the importance of this correction in comparing hybrid vehicle performance to conventional vehicle performance, as described by the SAE Recommended Practice J27117 [24]. Pagerit, et al [25] describe how SOC corrections can be used within optimization studies for HEV's. Duoba [26] and Senger [27] discuss how a conversion for electrical energy usage can be used as an alternative to SOC correction, by converting the electrical energy that can be obtained from gasoline through the efficiency of an average power plant in the United States in 1996. SOC corrections in some form provide the basis for the vast majority of all optimization work with HEV's to date.

A review of the existing literature reveals that successful optimizations have been performed on HEV component, control strategy, and configuration design. However, despite the magnitude of attention given to the optimization of HEV powertrains, the roles the optimization problem statement and SOC correction have in the outcome of the optimized design has yet to be adequately addressed. In the current study, the HEV design found with an optimization strategy utilizing an SOC correction method is compared with the design found utilizing an alternative optimization strategy with no SOC correction. A comparison between the necessary optimization problem statements for each strategy and their benefits is presented, and the strategies are merged into a single and more robust strategy that is shown to have several advantages in terms of design performance.

CHAPTER 3: OPTIMIZATION WITH HEEDS AND ADVISOR

3.1 Introduction to HEEDS

HEEDS (Hierarchical Evolutionary Engineering Design System) is a robust design exploration and optimization software package that automates the search for better and more robust solutions within a given design space and dramatically reduces design time [29]. HEEDS has a search algorithm SHERPA that is a hybrid adaptive search method proven to be one of, if not the best search algorithm available today. It has been demonstrated to outperform other methods on a wide variety of problems (both in terms of performance and robustness) [28]. For this reason, HEEDS and SHERPA have been chosen as the optimization tool and search algorithm for this thesis project. All optimizations performed in this first section utilize HEEDS and showcase the benefits and power of SHERPA as it relates to the optimization of hybrid electric vehicles.

3.2 Introduction to ADVISOR

3.2.1 Overview

The advanced vehicle simulator (ADVISOR) is a simulation tool that utilizes a combined forward/backward simulation approach to evaluate a vehicle's performance. It is used in conjunction with Matlab to solve difficult simulations involving Hybrid Electric Vehicles (HEV's).

ADVISOR uses Matlab .m files to define the HEV properties along with the actions to be taken to test the performance of the HEV. It has a graphical user interface

(GUI) that correlates all of the .m files needed to run a given problem. However to use HEEDS with ADVISOR, the GUI cannot be utilized while HEEDS is executing a design study. As a result the *no_gui* capabilities specific to ADVISOR must be used to define and alter the HEV and its performance characteristics during a HEEDS design study. The baseline model however can be built and tested in the ADVISOR GUI for validation before time is spent on the *no_gui* capabilities.

In the next sub-section of this chapter, the different .m files needed for a given ADVISOR analysis are introduced (for further detail, please refer to the ADVISOR User's Manual [9]). The *no_gui* capabilities are not generally discussed, but are discussed in detail as they relate to the example problem posed in the following section.

An example problem is presented to demonstrate how to use HEEDS with ADVISOR. It is representative of a specific problem and the *no_gui* utilization will likely need to be altered for other problems.

3.2.2 Matlab .m Files

To define an HEV and its performance actions in ADVISOR, there are 13 primary categories of .m files used. These files can be manipulated and managed to represent a specific HEV. The .m file categories are described below:

Vehicle Description: These .m files describe and load all of the other .m files needed for a given HEV configuration.

Vehicle: These .m files define all of the parameters associated with the vehicle such as the coefficient of aerodynamic drag, the

mass of the vehicle, etc.

Fuel Converter: These .m files define all of the parameters associated with the fuel converter or engine of the vehicle. Examples include the fuel and torque maps.

Exhaust Aftertreatment: These .m files define all emissions data and catalytic converter data associated with the vehicle.

Energy Storage System: These .m files define all of the data associated with the energy storage system such as the batteries, super capacitors, etc.

Motor: These .m files define all of the data regarding the electric motor.

Generator: These .m files define all of the data associated with generators in the vehicle and devices used for recharging.

Transmission: These .m files define the transmission for use in the vehicle.

Wheel/Axle: These .m files define the wheels and axles and the resistances associated with each.

Accessory: These .m files define the electrical accessories needed for the vehicle. Air conditioning is included in these .m files.

Control: These .m files define the control system used to define how the HEV operates.

Drive Cycle: These .m files define the drive cycle over which the vehicle is to be tested and evaluated.

Test Procedure: These .m files are an alternative category to the *Drive Cycle* .m files in that they define a specific test to be performed on the vehicle rather than a specific route the vehicle is to take.

3.3 Linking HEEDS to ADVISOR

3.3.1 ADVISOR Optimization Preparation

There are four files required with every ADVISOR optimization run. The main file defines the ADVISOR model to be used and contains the commands for writing the values of the design variables and responses to an output response file (*outdata_control.txt* in this example). In this example, the main file is named *optim_control_HEEDS.m*. Another file identifies the objectives for the run and stores their values following the simulation (*obj_fun_control.m* for this example). A third file identifies and stores the values of the constraints for the run (*con_fun_control.m* for this example). The fourth file contains the values of the design variables (*indata_control.m* for this example). These files are used by ADVISOR to load and define the entire problem setup for each HEEDS evaluation. They therefore are unique for every single optimization which utilizes ADVISOR as an analysis tool. Descriptions of the file contents follow as they relate to this particular example problem.

3.3.1.1 optim_control_HEEDS.m

For this example, the file associated with defining the ADVISOR model contains 10 primary sections (see Figure 3.3.1). Section 1 of the code creates a Matlab workspace where no variables are defined. Section 2 initiates a timer so that the length of the analysis can be monitored. Section 3 loads the vehicle description .m file *PARALLEL_defaults_in*. Section 4 alters the number of battery modules to 14 from the default value. Section 5 of the code scales the motor to 13.5kW from 75kW. Section 6 defines the variables and responses to be used in the optimization. Section 7 identifies the

file containing the design variable information and imports their values into the model. Section 8 identifies the files containing the objective and constraint variables and associates them with the model. Section 9 creates the output file for the analysis (*outdata_control.txt*) and writes the objective and constraint data to it. Finally, Section10 stops the timer since the analysis is now complete.

```
% Clear all the variables ------------------------------------------------ 1
clear all
% initiate timer ---------------------------------------------------------- 2
tic
% Load advisor saved vehicle ---------------------------------------------- 3
input.init.saved_veh_file='PARALLEL_defaults_in';
[error_code,resp]=adv_no_gui('initialize',input);
%Alteration to ESS ------------------------------------------------------- 4
input.modify.param={'ess_module_num'};
input.modify.value={14};
[error,resp]=adv_no_gui('modify',input);
%Alteration to Motor ----------------------------------------------------- 5
input.modify.param={'mc_trq_scale'};
input.modify.value={0.18};
[error,resp]=adv_no_gui('modify',input);
% Define variables -------------------------------------------------------- 6
dv_names={'cs_hi_soc','cs_lo_soc','cs_electric_launch_spd_lo','cs_electric_launch_spd_hi',
'cs_off_trq_frac','cs_min_trq_frac','cs_charge_trq'};
resp_names={'combined_mpgge','hc_gpm','co_gpm','nox_gpm',
'vinf.accel_test.results.time(1)','vinf.accel_test.results.time(2)','vinf.accel_test.results.time(3)',
'vinf.accel_test.results.max_speed','vinf.accel_test.results.max_rate',
'vinf.accel_test.results.dist_in_time','vinf.grade_test.results.grade'};
% load new design variables ----------------------------------------------- 7
indata_control
% Evaluate responses ------------------------------------------------------ 8
obj=obj_fun_control(X,dv_names,resp_names);
con=con_fun_control(X,dv_names,resp_names);
% write responses to file ------------------------------------------------- 9
fid=fopen('outdata_control.txt','w');
fprintf(fid,'Fuel Economy: %g',obj);
fprintf(fid,'\n');
fprintf(fid,'HC: %g',con(1));
fprintf(fid,'\n');
fprintf(fid,'CO: %g',con(2));
fprintf(fid,'\n');
fprintf(fid,'NOx: %g',con(3));
fprintf(fid,'\n');
fprintf(fid,'Accel 0-60: %g',con(4));
fprintf(fid,'\n');
fprintf(fid,'Accel 40-60: %g',con(5));
fprintf(fid,'\n');
fprintf(fid,'Accel 0-85: %g',con(6));
fprintf(fid,'\n');
fprintf(fid,'Max Speed: %g',con(7));
fprintf(fid,'\n');
fprintf(fid,'Max Accel: %g',con(8));
fprintf(fid,'\n');
fprintf(fid,'Distance in 5s: %g',con(9));
fprintf(fid,'\n');
fprintf(fid,'Grade: %g',con(10));
fprintf(fid,'\n');
fclose(fid);
%end timer ---------------------------------------------------------------- 10
toc
```

Figure 3.3.1. Main input file: optim_control_HEEDS.m

3.3.1.2 obj_fun_control.m

This file contains six primary sections (see Figure 3.3.2). Section 1 defines a function. Section 2 initializes the Matlab variables to be used. Section 3 updates the parameter settings defined by the design variables. Section 4 of the code runs the FTP driving cycle. Section 5 identifies the objective variable from the driving cycle (gas mileage). Finally, Section 6 identifies the other objective variables from the driving cycle (emissions).

```
function obj=obj_fun(x,varargin) ---------- 1
% initialize ---------- 2
error=0;
obj=0;
% update parameter settings ---------- 3
input.modify.param=varargin{1}; % parameter names are stored in the first optional argument
input.modify.value=num2cell(x); % assign corresponding values
[error,resp]=adv_no_gui('modify',input);
% run FTP driving cycle ---------- 4
if ~error
  input.cycle.param={'cycle.name'};
  input.cycle.value={'CYC_FTP'};
  [error,resp]=adv_no_gui('drive_cycle',input);
end
% assign objective value ---------- 5
if ~error
  obj=resp.cycle.mpgge;
end
% assign objective value ---------- 6
if ~error
  assignin('base','con',[resp.cycle.hc_gpm; resp.cycle.co_gpm; resp.cycle.nox_gpm])
end

return
```

Figure 3.3.2. Objective definition file: obj_fun_control.m

3.3.1.3 con_fun_control.m

This file has four primary sections (see Figure 3.3.3.). Section 1 defines a function. Section 2 enables additions to the vector where responses are stored. Section 3 runs and defines an acceleration test, as well as obtains constraints from it (acceleration times, max velocity, max acceleration, distance covered in 5s). Section 4 runs and defines a grade test, as well as obtains the constraint associated with the grade test (grade).

```
function [con, con_e]=con_fun(x,varargin)------------------------------------ 1
con=evalin('base','con');----------------------------------------------------- 2
offset=length(con);
% run acceleration test--------------------------------------------------------- 3
input.accel.param={'spds','gb_shift_delay','dist_in_time','max_speed_bool','max_rate_bool',};
input.accel.value={[0 60; 40 60; 0 85],0.2,5.0,1,1};
[error, resp]=adv_no_gui('accel_test',input);
if ~error&~isempty(resp.accel.times)
  con(offset+1,1)=resp.accel.times(1);
  con(offset+2,1)=resp.accel.times(2);
  con(offset+3,1)=resp.accel.times(3);
  con(offset+4,1)=resp.accel.max_speed;
  con(offset+5,1)=resp.accel.max_rate;
  con(offset+6,1)=resp.accel.dist;
else
  con(offset+1:offset+3,1)=100;
end
% run grade test---------------------------------------------------------------- 4
input.grade.param={'duration','speed'};
input.grade.value={1200,55};
[error, resp]=adv_no_gui('grade_test',input);
if ~error&~isempty(resp.grade.grade)
  con(offset+7,1)=resp.grade.grade;
else
  con(offset+7,1)=0;
end

con_e=0;
% ****

return
```

Figure 3.3.3. Constraint definition file: con_fun_control.m

3.3.1.4 indata_control.m

This file contains only one section which defines all the values for the design variables referenced in the file *optim_control_HEEDS.m* (see Figure 3.3.4).

```
X(1)=0.98
X(2)=0.96
X(3)=10
X(4)=10
X(5)=0.2
X(6)=0.4
X(7)=10
```

Figure 3.3.4. Input design variable file: indata_control.m

3.3.2 Running the ADVISOR Simulation from Batch Mode

There is an executable located within the *gui* folder of the ADVISOR files named *advisor_script.exe* that is created when ADVISOR is installed. It is this file that allows ADVISOR to be run from batch mode. To run this file from a command prompt, the format is:

advisor_script.exe WorkingDirectory ScriptFilename ADVISORRootDirectory SupportDirectory

Where for our optimization problem, this takes the form (if running from the C:\Temp directory and the ADVISOR root directory is set to C:\Program Files\AVL\ADVISOR\ADVISOR2004):

C:\Temp\advisor_script.exe C:\Temp optim_control_HEEDS ...
C:\PROGRA~1\AVL\ADVISOR\ADVISOR2004 C:\Temp

For use with HEEDS, a batch file needs to be created named *RUN_ADVISOR.bat* which contains this command, with the *advisor_script.exe* copied to the working directory.

3.3.3 Process Integration Between HEEDS and ADVISOR

HEEDS performs the optimization using ADVISOR as the simulation tool by iteratively performing simulations and searching the design space using the search method SHERPA. SHERPA is a hybrid adaptive search method that tunes itself for a given problem [29]. The process of optimizing using HEEDS with ADVISOR will be described in this sub-section.

As was discussed in Sub-Section 3.3.2, ADVISOR needs to be run from batch mode for HEEDS to be able to perform a design study. Sub-Section 3.3.1 detailed the *no_gui* capabilities of ADVISOR and the files associated with this. How HEEDS uses this capability is as follows. HEEDS inputs variable values into the input design variable file (*indata_control.m* for this example). These design variable values are read into the main file (*optim_control_HEEDS.m* for this example), along with data from the objective definition file (*obj_fun_control.m* in this example) and the constraint definition file (*con_fun_control.m* in this example). The executable *advisor_script.exe* then launches ADVISOR using the problem statement defined in the main file (*optim_control_HEEDS.m*). The ADVISOR simulation is then run and the output response file (*outdata_control.txt* in this example) is written out based upon the information in the main file. The response values are then read from this output response file by HEEDS. SHERPA then adapts and tunes its search based upon this information, in hopes of improving the design. In doing so, it creates new variable values that are again put into the input design variable file, and this iterative loop begins again. This loop is continued for a set number of evaluations or until the optimal design is found. Figure 3.3.5 depicts this process integration performed by HEEDS with ADVISOR.

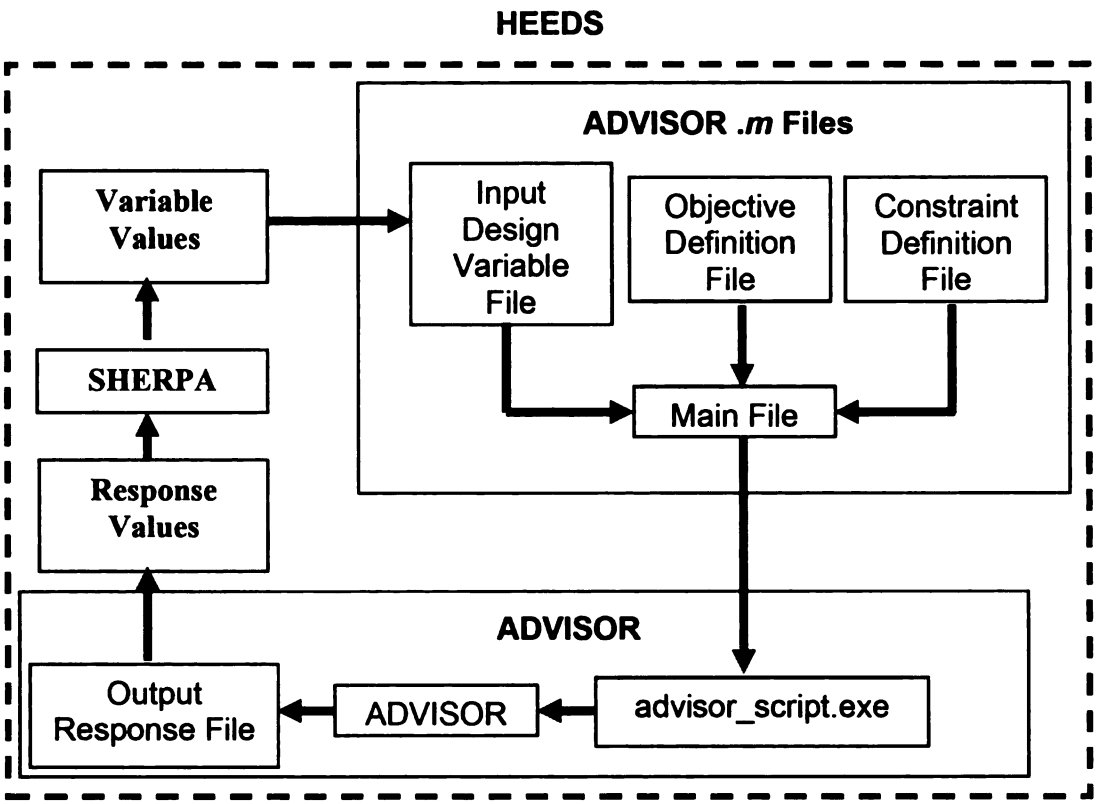


Figure 3.3.5. Process integration performed by HEEDS with ADVISOR.

3.4 Optimization Example: Small Vehicle Parallel HEV

3.4.1 Baseline Model

The vehicle powertrain to be studied in this example problem is that of a Parallel HEV (see Figure 3.4.1 for this configuration in ADVISOR). Its parameters are defined as follows (similar to the problem defined by M.Montazeri-Gh, A. Poursamad, and B. Ghalichi [15]):

- Body mass: 592 kg
- Rolling resistance coefficient: 0.009
- Body aerodynamic drag coefficient: 0.335
- Vehicle front area: $2m^2$

- Wheel radius: 0.282 m
- Gearbox: five speed manual gearbox with the following gear ratios: 2.84, 3.77, 5.01, 7.52, 13.45.
- IC engine: 41 kW SI engine
- Electric motor: 13.5 kW AC motor
- Energy storage system: 14 12V Hawker Genesis valve-regulated lead-acid battery
- Catalyst converter: close-coupled conventional converter for an SI engine
- Drive Cycle: FTP

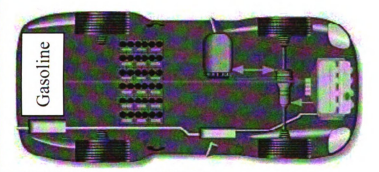


Figure 3.4.1. Parallel HEV Configuration in ADVISOR

This vehicle model is provided as a baseline model within the ADVISOR software. Slight modifications however, do need to be made to the default model (these modifications are described later). This problem definition correlates to the following pre-existing .m files:

Vehicle Description: PARALLEL_defaults_in.m

Vehicle: VEH_SMCAR

Exhaust Aftertreatment:	EX_SI
Energy Storage System:	ESS_PB25 with alterations necessary to the number of battery modules (shown in Sub-Section 3.3.1.1)
Motor:	MC_AC75 with alterations to the torque scaling to make the motor 13.5 kW instead of 75 kW (shown in Sub-Section 3.3.1.1)
Transmission:	TX_5SPD
Wheel/Axle:	WH_SMCAR
Drive Cycle:	CYC_FTP

3.4.2 Optimization Problem Statement

By altering the control system settings of the parallel HEV, this optimization study attempts to minimize both the fuel consumption and emissions of the vehicle, while maintaining the performance characteristics of the vehicle. This optimization problem can therefore be summarized as:

Minimize: $J(x) = \frac{1}{mpgge} + HC + CO + NOx$

where:

mpgge is the gas mileage,
HC is the hydrocarbon emissions,
CO is the carbon monoxide emissions,
NOx is the nitrous oxide emissions.

Subject to: $Grade \geq 6.5$
$Acc\_1 \leq 12$
$Acc\_2 \leq 5.3$
$Acc\_3 \leq 23.4$
$V\_max \geq 85.13$
$A\_max \geq 16.1$
$Dist\_5s \geq 140.1$

where:

Grade is the grade able to be achieved at 88.5 km/hr for 20 min,
Acc_1 is the time to accelerate from 0-60 mi/hr,
Acc_2 is the time to accelerate from 40-60 mi/hr,
Acc_3 is the time to accelerate from 0-85 mi/hr,
V_max is the maximum velocity achieved,
A_max is the maximum acceleration achieved,
Dist_5s is the distance achieved in 5 seconds.

By varying: $0 \leq DeltaSOC \leq 0.31$
$0 \leq VL \leq 15$
$0 \leq T\_off \leq 0.31$
$0.25 \leq T\_min \leq 0.88$
$4 \leq T\_ch \leq 35$

where:
DeltaSOC is the change in limits on the state of charge of the system,
VL is the vehicle speed below which the vehicle operates as a zero emissions vehicle (electric launch speed),
T_off is the minimum torque threshold when SOC > LSOC,
T_min is the minimum torque threshold when SOC < LSOC,
T_ch is the torque load on engine to recharge the battery pack when the engine is on.

3.4.3 HEEDS Setup

This section details the steps necessary to setup the optimization problem in HEEDS as posed in Sub-Section 3.4.2. Detailed step-by-step instructions are given in this section as to this setup.

3.4.3.1 Starting a New Project

1. Open the HEEDS Modeler.
2. Open the file menu and click **New** (or select **New** from the options displayed on the screen).
3. Navigate to your working directory.
4. Save the file as **parallelHEV.**

3.4.3.2 Defining the Evaluation Process

This problem only has one analysis, with 4 input files and one output file.

1. Select the **Processes** tab.

2. Select **Analysis_1** in the **Process Graph**. The **Analysis Manager** will appear below.
3. Click the **Browse** button next to the **Execution File** field and navigate to the working directory.
4. Select **RUN_ADVISOR.bat**. This is the file containing the commands and script necessary to run ADVISOR from batch mode (see above). Delete the path portion of the field and replace with ".\".
5. Select the **Input Files** button to display the **Input Files Manager.**
6. Right click anywhere in the **Input Files Manager** and **Add Input File.**
7. Add the input file **indata_control.m**.
8. Repeat for the input files **optim_control_HEEDS.m, obj_fun_control.m, con_fun_control.m**, and **RUN_ADVISOR.bat**.
9. Select the **Output Files** button to display the **Output Files Manager.**
10. Right click anywhere in the **Output Files Manager** and **Add Output File.**
11. Add the output file **outdata_control.txt**
12. Save the Project.

3.4.3.3 Defining the Project Variables and Responses

3.4.3.3.1 Defining the Project Variables

1. Display the **Variables** tab.
2. Add 7 design variables and define them as follows:

   Name: DeltaSOC; Type: Continuous; Min: 0.0; Baseline: 0.1; Max: 0.31

   Name: HSOC; Type: Parameter; Baseline: 0.70

   Name: LSOC; Type: Dependent; Formula: HSOC-DeltaSOC

Name: VL; Type: Continuous; Min: 0.0; Baseline: 0.0; Max: 15

Name: T_off; Type: Continuous; Min: 0.0; Baseline: 0.0; Max: 0.31

Name: T_min; Type: Continuous; Min: 0.25; Baseline: 0.40; Max: 0.88

Name: T_ch; Type: Continuous; Min: 4; Baseline: 20; Max: 35

3. Add Internal Variables to the project (to define the path in the **RUN_ADVISOR.bat** file).

3.4.3.3.2 Defining the Project Responses

1. Add 13 project responses and define them as:

Name: mppge; Source: File

Name: inv_mppge; Source: File

Name: HC; Source: File

Name: CO; Source: File

Name: NOx; Source: File

Name: J_x; Source: Formula; Formula: inv_mppge+HC+CO+NOx

Name: Grade; Source: File

Name: Acc_1; Source: File

Name: Acc_2; Source: File

Name: Acc_3; Source: File

Name: V_max; Source: File

Name: A_max; Source: File

Name: Dist_5s; Source: File

2. Save the Project.

3.4.3.4 Tagging the Input and Output Files

3.4.3.4.1 Tagging the Input Files

1. Display the **Tagging** tab.
2. For this project, there are two input files that need tagging. The first dealing with the design variable tagging is **indata_control.txt**. Select this file in the **Processes** tree and the design variable **HSOC** in the **Project Variables** window.
3. Tag the file as detailed below (note VL is tagged twice since we want the upper and lower speeds for which it represents to be identical):

   Variable: HSOC; Location: Line 1, Field 3

   Variable: LSOC; Location: Line 2, Field 3

   Variable: VL; Location: Line 3, Field 3

   Variable: VL; Location: Line 4, Field 3

   Variable: T_off; Location: Line 5, Field 3

   Variable: T_min; Location: Line 6, Field 3

   Variable: T_ch; Location: Line 7, Field 3

4. The file **RUN_ADVISOR.bat** must have the path tagged within it. Select this input file in the **Processes** tree and the internal variable **Design_Path**. Tag the file as described below:

   Internal Variable: Design Path; Location: Line 1, Field 2

3.4.3.4.2 Tagging the Output File

1. Select the output file **outdata_control.txt** from the **Processes** tree and the response **mppge** from the **Project Responses** window.
2. Tag the file as described below:

   Response: mppge; Location: Line 1, Field 3

   Response: HC; Location: Line 2, Field 2

   Response: CO; Location: Line 3, Field 2

   Response: NOx; Location: Line 4, Field 2

   Response: Acc_1; Location: Line 5, Field 3

   Response: Acc_2; Location: Line 6, Field 3

   Response: Acc_3; Location: Line 7, Field 3

   Response: V_max; Location: Line 8, Field 3

   Response: A_max; Location: Line 9, Field 3

   Response: Dist_5s; Location: Line 10, Field 4

   Response: Grade; Location: Line 11, Field 2

3. Save the Project

3.4.3.5 Assembling the Problem Definition

1. Select the **Assembly** tab.
2. Select the **OPT_Agent_1** button and assign **Process_1** to the Agent.
3. Select the **Methods** button and select **SHERPA** as the optimization technique.
4. Click the **Next** button and change the number of evaluations to 200.
5. Select the **Variables** button. Leave the resolution at the default value of 101.

6. Select the **Responses** button. Define the optimization problem as that defined below:

   Response: mppge; Type: Prerequisite

   Response: HC; Type: Prerequisite

   Response: CO; Type: Prerequisite

   Response: NOx; Type: Prerequisite

   Response: Grade; Type: Constraint ≥ 6.5

   Response: Acc_1; Type: Constraint ≤ 12

   Response: Acc_2; Type: Constraint ≤ 5.3

   Response: Acc_3; Type: Constraint ≤ 23.4

   Response: V_max; Type: Constraint ≥ 85.13

   Response: A_max; Type: Constraint ≥ 16.1

   Response: Dist_5s; Type: Constraint ≥ 140.1

   Response: inv_mppge; Type: Prerequisite

   Response: J_x; Type: Objective, Minimize, 1

7. Save the project.

3.4.4 Running the Project

To run the HEEDS project, save the project and click on the *Run* tab and the *Run* button in the HEEDS Modeler. The project should run with a HEEDS window popping up along with a Matlab window. The Matlab window will close after each evaluation and a new one pop up with each new evaluation.

3.4.5 Results and Discussion

The baseline model for this run is defined in Table 3.4.1. This baseline was chosen somewhat arbitrarily from [15], but was believed to have been a good design to begin with based upon the paper this problem was adapted from. The resulting response values for this design are:

mppge	= 40.19840
HC	= 0.438372
CO	= 1.929040
NOx	= 0.36759
grade	= 8.21989
Acc_1	= 10.3826
Acc_2	= 5.3204
Acc_3	= 22.0176
Vmax	= 108.788
Amax	= 16.2423
Dist_5s	= 170.814
J_X	= 2.759884

Table 3.4.1. Baseline model design variable values

Name	Baseline
HSOC	0.70
DeltaSOC	0.10
LSOC	0.60
VL	0
T_off	0
T_min	0.4
T_ch	20

HEEDS was capable of finding within the 200 evaluations specified, a design that had the following response values:

mppge	= 43.75
HC	= 0.4266190
CO	= 1.910740
NOx	= 0.362577
grade	= 8.015360
Acc_1	= 10.3536

Acc_2 = 5.27818
Acc_3 = 21.75
Vmax = 110.045
Amax = 16.2424
Dist_5s = 169.719
J_X = 2.722793

HEEDS found a new design with an 8.84% increase in fuel economy, while decreasing hydrocarbon emissions by 2.75%, carbon monoxide emissions by 0.96% and nitric oxide emissions by 1.383% over the baseline design. Also, the design HEEDS found satisfied all the performance constraints placed upon the problem as defined. This design corresponds to the values for design variables seen in Table 3.4.2

Table 3.4.2. Optimized design variable values.

Name	Value
HSOC	0.70
DeltaSOC	0.0206666
LSOC	0.6793333
VL	1.071429
T_off	0.0206666
T_min	0.4735484
T_ch	4

CHAPTER 4: BASELINE ADVISOR MODEL FOR A HYBRID ELECTRIC BUS

4.1. Introduction

In attempting to develop an optimization strategy for HEV's, it was necessary to develop an ADVISOR model that was realistic and representative of a real-world application. A hybrid bus model was chosen as the baseline model for all work in regards to optimization. Hybrid buses are becoming commonplace, and work in optimizing their performance is becoming very important.

An accurate simulation model therefore, had to be created and validated for a baseline design configuration of a hybrid bus. As mentioned previously, the simulation software ADVISOR has been chosen for the current study. Therefore, the baseline bus configuration had to be accurately modeled in ADVISOR by defining the necessary parameters used in the hybrid electric powertrain model. The baseline bus was modeled according to early bus specifications and data associated with a joint project between Michigan State University, Kettering University, and TransTeq [30] with support from the Michigan Economic Development Corporation. After the baseline model was validated, the optimization software HEEDS was used to perform sizing optimization of the powertrain so as to develop an optimization strategy for hybrid vehicles. The goal of this chapter is to define the simulation model for the baseline configuration.

4.2. ADVISOR Vehicle Model Description

Since ADVISOR was selected as the simulation tool for this project, each of the vehicle systems had to be described in detail in terms of the parameters that defined that system within ADVISOR.

A Series Hybrid Electric Vehicle configuration was chosen for the baseline simulations of the bus. ADVISOR has a pre-existing model for a bus known as the Orion VI Low-Floor Transit bus. This vehicle model was used for the simulations, with the parameters modified to represent the early specifications for the bus in the joint project discussed briefly above.

While many parameters could be measured or obtained directly, some had to be estimated based upon assumptions. The height of the vehicle center of gravity (*veh_cg_height*) was one of these parameters. To obtain this vehicle center of gravity, the bus had to be broken down into components based upon its weight. The curb weight of the vehicle (the bus weight without passengers) was 26,750 lbs. The tires, axles, and transmission weight were 4000 lbs, distributed at a location 18.8 inches above the ground (rolling radius of the wheel). The engine and engine compartment weighed roughly 1300 lbs and had a center of gravity located about 30.8 inches above the ground. The air conditioning unit was located 80 inches above the ground and weighs 500 lbs. The remaining weight of the bus (21,150 lbs) was that without components (*veh_glider_mass*). The roof of the bus could be estimated as accounting for 15% of the remaining weight (3,172 lbs) located at 124 inches. The floor accounted for roughly 35% of the 21,150 lbs (or 7,402 lbs) located at 10.5 inches. The walls accounted for the rest of the 21,150 lbs, but had the weight evenly distributed so their effect on the center of

gravity was insignificant. Assuming 40 passengers at 150 lbs each gave the cargo mass (*veh_cargo_mass = 6000 lbs*). If 23 of these passengers were seated, their center of mass could be assumed to be 2 ft. The other 17 passengers could be assumed to have a center of mass of 3 ft since they were standing. Standard calculations for the center of gravity therefore yielded the center of gravity of the bus to be 35.80 inches.

4.2.1. Vehicle Level Parameters

Table 4.2.1. Vehicle parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
veh_gravity	9.81 m/s^2	32.185 ft/s^2	Reference value	Acceleration of gravity
veh_air_density	1.23 kg/m^3	0.0768 lb/ft^3	Reference value	Air density
veh_CD	0.79	0.79	Transit Bus Standard	Coefficient of aerodynamic drag
veh_FA	9.4732 m^2	101.97 ft^2	Specific for Flint Buses	Frontal area of bus
veh_cg_height	0.908304 m	2.98ft	Specific for Flint Buses	Height of vehicle center of gravity
veh_front_wt_frac	0.2785	0.2785	Specific for Flint Buses	Fraction of total vehicle mass supported by front axle when at rest
veh_wheelbase	7.443m	24.42 ft	Specific for Flint Buses	Distance between front and rear axles
veh_glider_mass	9593.48 kg	657.362 slugs	Specific for Flint Buses	Mass of bus without components
veh_cargo_mass	2,721.55 kg	186.486 slugs	Specific for Flint Buses	Vehicle cargo mass (assuming 40 people at 150 lbs per person)

4.2.2. Fuel Converter Parameters

The fuel converter data corresponded to that of a Detroit Diesel Corp. Series 50 8.5 (205kW) diesel engine. Many of the fuel converter parameters are given as maps. These maps are defined below in SI units defined in Table 4.2.2.

Map 4.2. 1
fc_map_spd = [125.66, 141.37, 157.08, 172.79, 188.50, 219.91]

Map 4.2.2
fc_map_trq =[123.02, 246.04, 369.07, 492.09, 615.11, 738.13, 861.15, 984.18, 1107.20]

Map 4.2.3
fc_fuel_map_gpkWh =
[242, 220, 207, 201.5, 198.5, 197, 195.5, 194.5, 193.5;
261, 220, 207, 200, 195, 192.5, 190.5, 189.5, 189.5;
273, 224, 207, 198, 194.5, 192, 190.5, 189.5, 189.5;
288, 228, 209, 200.5, 196, 193.5, 191.5, 191.5, 191.5;
288, 238, 217, 207.5, 203, 199.5, 197.5, 197.5, 197.5;
290, 244, 222, 210.5, 204.5, 202.5, 199.5, 199.5, 199.5]

Table 4.2.2. Fuel converter parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
fc_fuel_type	Diesel	Diesel	ADVISOR Default Value	Type of fuel
fc_disp	8.5 L	2.245 gal	ADVISOR Default Value	Engine size (cylinder displacement)
fc_map_spd	Map 4.2.1 (rad/s)	Map 4.2.1	ADVISOR Default Value	Speed range of the engine
fc_map_trq	Map 4.2.2 (N-m)	Map 4.2.2	ADVISOR Default Value	Engine torque range
fc_fuel_map_gpkWh	Map 4.2.3 (g/kWh)	Map 4.2.3	ADVISOR Default Value	Fuel use map of engine
fc_idle_speed	650 rad/s	650 rad/s	ADVISOR Default Value	Idle speed of the engine

4.2.3. Exhaust Aftertreatment Parameters

The aftertreatment catalyst parameters defined were those for a diesel-powered CI engine. All of the catalyst parameters are given as maps. These maps are defined below in SI units defined in Table 4.2.3.

Map 4.2. 4
ex_cat_tmp_range = [-40, 0, 220, 240, 310, 415, 475, 550, 650, 1200]

Map 4.2. 5
ex_cat_hc_frac = [0, 0, 0.04, 0.08, 0.20, 0.57, 0.80, 0.90, 0.91, 0.91]

Map 4.2.6
ex_cat_co_frac = [0, 0, 0.06, 0.20, 0.50, 0.80, 0.90, 0.95, 0.95, 0.95]

Map 4.2.7
ex_cat_nox_frac = [0, 0, 0.01, 0.15, 0.45, 0.30, 0.20, 0.10, 0.01, 0]

Map 4.2.8
ex_cat_pm_frac = [0, 0, 0.04, 0.08, 0.20, 0.30, 0.35, 0.40, 0.40, 0.40]

Map 4.2.9
ex_cat_lim = [1.25, 17.0, 2.0, 0.4]

Table 4.2.3. Exhaust aftertreatment parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
ex_cat_tmp_range	Map 4.2.4 (°C)	Map 4.2.4	ADVISOR Default Value	Catalyst temperature range
ex_cat_hc_frac	Map 4.2.5	Map 4.2.5	Automotive Eng. 10/95 p. 42 (Reference)	Catalyst HC conversion efficiency map (dependent on catalyst temperature)
ex_cat_co_frac	Map 4.2.6	Map 4.2.6	Johnson Matthey Corp.	Catalyst CO conversion efficiency (dependent on catalyst temperature)
ex_cat_nox_frac	Map 4.2.7	Map 4.2.7	ADVISOR Default Value	Catalyst NO_x conversion efficiency (dependent on catalyst temperature)
ex_cat_pm_frac	Map 4.2.8	Map 4.2.8	ADVISOR Default Value	Catalyst PM conversion efficiency (dependent on catalyst temperature)
ex_cat_lim	Map 4.2.9	Map 4.2.9	ADVISOR Default Value	Breakthrough limit of converter (5 times the Tier 1 g/mi limits)

4.2.4. Energy Storage System Parameters

The energy storage system was based upon an Ovonic 90Ah NiMH battery. Many of the catalyst parameters are given as maps. These maps are defined below in SI units defined in Table 4.2.4.

Map 4.2.10
ess_soc = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]

Map 4.2.11
ess_tmp = [0, 22, 40]

Map 4.2.12
ess_max_ah_cap = [90, 90, 90]

Map 4.2.13
ess_r_dis =

0.01*[1.167, 0.631, 0.594, 0.552, 0.541, 0.530, 0.523, 0.536, 0.574, 0.615, 0.585;
1.167, 0.631, 0.594, 0.552, 0.541, 0.530, 0.523, 0.536, 0.574, 0.615, 0.585;
1.167, 0.631, 0.594, 0.552, 0.541, 0.530, 0.523, 0.536, 0.574, 0.615, 0.585]

Map 4.2.14

ess_voc =

[12.5, 12.8, 13.1, 13.3, 13.4, 13.4, 13.5, 13.6, 13.7, 13.9, 14.2;
12.5, 12.8, 13.1, 13.3, 13.4, 13.4, 13.5, 13.6, 13.7, 13.9, 14.2;
12.5, 12.8, 13.1, 13.3, 13.4, 13.4, 13.5, 13.6, 13.7, 13.9, 14.2]

Table 4.2.4. Energy storage system parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
ess_soc	Map 4.2.10	Map 4.2.10	ADVISOR Default Value	Range of the state of charge of the battery
ess_tmp	Map 4.2.11 (ºC)	Map 4.2.11	ADVISOR Default Value	Temperature range over which data is defined.
ess_max_ah_cap	Map 4.2.12 (A-h)	Map 4.2.12	ADVISOR Default Value	Maximum capacity at C/5 rate. (dependent on temperature)
ess_r_dis	Map 4.2.13 (Ohms)	Map 4.2.13	ADVISOR Default Value	Module's resistance to being discharged (dependent on temperature {vertically} and SOC {horizontally})
ess_voc	Map 4.2.14 (Volts)	Map 4.2.14	ADVISOR Default Value	Module's open-circuit voltage (dependent on temperature {vertically} and SOC {horizontally})
ess_min_volts	9.135 volts	9.135 volts	ADVISOR Default Value	Storage system minimum voltage (based upon 10 cells and a 105% safety factor)
ess_max_volts	15.675 volts	15.675 volts	ADVISOR Default Value	Storage system maximum voltage (based upon 10 cells and a 95% safety factor)
ess_module_mass	16.7 kg	1.144313 slugs	ADVISOR Default Value	Mass of a single 12 V module
ess_module_volume	.006597 m^3	0.232971 ft^3	ADVISOR Default Value	Volume of a single module
ess_module_number	25	25	ADVISOR Default Value	Number of modules

4.2.5. Motor Parameters

The motor defined was that of a 58 kW permanent magnet motor/controller. Many of the motor parameters are given as maps. These maps are defined below in SI units defined in Table 4.2.5.

Map 4.2.15
mc_map_spd =
[0, 26.2, 52.4, 78.5, 104.7, 130.9, 157.1, 183.3, 209.4, 235.6, 261.8, 288, 314.2, 340.3, 366.5, 392.7, 418.9]

Map 4.2.16
mc_map_trq = [0, 25, 50, 75, 100, 125, 150, 175, 200, 225, 250, 275, 300, 325, 350, 375, 400]

Map 4.2.17
mc_eff_map =
[0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3;
0.3, 0.7, 0.75, 0.75, 0.75, 0.73, 0.71, 0.7, 0.7, 0.68, 0.65, 0.63, 0.62, 0.61, 0.6, 0.59, 0.58;
0.3, 0.75, 0.8, 0.82, 0.82, 0.82, 0.82, 0.82, 0.81, 0.8, 0.78, 0.77, 0.76, 0.76, 0.75, 0.73, 0.72;
0.3, 0.75, 0.82, 0.84, 0.86, 0.86, 0.85, 0.85, 0.85, 0.84, 0.84, 0.83, 0.82, 0.81, 0.8, 0.77, 0.77;
0.3, 0.75, 0.83, 0.86, 0.87, 0.88, 0.88, 0.87, 0.87, 0.87, 0.86, 0.86, 0.85, 0.84, 0.84,0.83, 0.82;
0.3, 0.75, 0.84, 0.87, 0.88, 0.88, 0.89, 0.89, 0.88, 0.88, 0.88, 0.87, 0.87, 0.86, 0.86, 0.85, 0.84;
0.3, 0.75, 0.84, 0.87, 0.89, 0.90, 0.90, 0.90, 0.89, 0.89, 0.89, 0.89, 0.88, 0.88, 0.88, 0.88, 0.88;
0.3, 0.75, 0.84, 0.88, 0.89, 0.9, 0.90, 0.90, 0.90, 0.9, 0.90, 0.89, 0.89, 0.89, 0.89, 0.89, 0.89;
0.3, 0.75, 0.84, 0.88, 0.90, 0.9, 0.90, 0.91, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90;
0.3, 0.75, 0.84, 0.88, 0.9, 0.90, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91;
0.3, 0.75, 0.85, 0.88, 0.9, 0.91, 0.91, 0.92, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91;
0.3, 0.75, 0.85, 0.88, 0.9, 0.90, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91, 0.91;
0.3, 0.75, 0.84, 0.88, 0.89, 0.9, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90;

0.3, 0.74, 0.82, 0.87, 0.89, 0.89, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9;
0.3, 0.7, 0.82, 0.86, 0.88, 0.89, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9;
0.3, 0.7, 0.81, 0.86, 0.88, 0.89, 0.89, 0.89, 0.89, 0.89, 0.89, 0.89, 0.89, 0.89, 0.89, 0.89, 0.892;
0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3]

Map 4.2.18

mc_max_trq=[340, 375, 402, 403, 401, 400, 348, 300, 250, 227, 202, 190, 175, 170, 150, 148, 0]

Table 4.2.5. Motor parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
mc_map_spd	Map 4.2.15 (rad/s)	Map 4.2.15	ADVISOR Default Value	Speed range of the motor
mc_map_trq	Map 4.2.16 (N-m)	Map 4.2.16	ADVISOR Default Value	Torque range of the motor
mc_eff_map	Map 4.2.17	Map 4.2.17	ADVISOR Default Value	Motor efficiency map (dependent on speed range of the motor {vertically} and torque range of the motor {horizontally})
mc_max_trq	Map 4.2.18 (N-m)	Map 4.2.18	ADVISOR Default Value	Max Torque Curve
mc_max_crrnt	480 Amps	480 Amps	ADVISOR Default Value	Maximum current allowed by the controller and motor
mc_inertia	0.0235 kg-m^2	0.017333 slugs-ft^2	ADVISOR Default Value	Rotor's rotational inertia
mc_mass	70 kg	4.80 slugs	ADVISOR Default Value	Mass of motor and controller
mc_min_volts	120 volts	120 volts	ADVISOR Default Value	Minimum voltage allowed by the controller and motor

4.2.6. Generator Parameters

Some of the generator parameters were given as maps. These maps are defined below in SI units defined in Table 4.2.6. For these maps, colons define them. For example, the torque range is defined by [0:5:200]. This means that the range is from 0 to 200 in increments of 5.

Map 4.2.19
gc_map_trq = [0:5:200]

Map 4.2.20
gc_map_spd = [0:26.18:733.04]

Table 4.2.6. Generator parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
gc_map_trq	Map 4.2.19 (N-m)	Map 4.2.19	ADVISOR Default Value	Torque range of the generator
gc_map_spd	Map 4.2.20 (rad/s)	Map 4.2.20	ADVISOR Default Value	Speed range of the generator
gc_max_crrnt	480 Amps	480 Amps	ADVISOR Default Value	Maximum current draw for the motor/controller set
gc_min_volts	120 volts	120 volts	ADVISOR Default Value	Minimum voltage for the motor/controller set
gc_inertia	0.01 kg-m^2	0.007376 slugs-ft^2	ADVISOR Default Value	Rotor's rotational inertia

4.2.7. Transmission Parameters

The transmission in the model was based upon an Allison automatic transmission for a school bus/transit bus. Some of the transmission parameters are given as maps. These maps are defined below in SI units defined in Table 4.2.7.

Map 4.2.21
gb_ratio=[3.49, 1.86, 1.41, 1.00, 0.75, 0.65]

Map 4.2.22
tx_map_spd=[0, 60.85]

Map 4.2.23
tx_map_trq=[-1254, 1254]

Map 4.2.24
tx_eff_map= [0.90, 0.90;
0.90, 0.90]

Table 4.2.7. Transmission parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
gb_ratio	Map 4.2.21	Map 4.2.21	Specific for Flint Buses	Ratios in gearbox
gb_gears_num	6	6	Specific for Flint Buses	Number of discrete gear choices in gearbox
tx_map_spd	Map 4.2.22 (rad/s)	Map 4.2.22	Specific for Flint Buses	Speed of transmission shaft output (wheel-side of transmission)
tx_map_trq	Map 4.2.23 (N-m)	Map 4.2.23	Specific for Flint Buses	Torque of transmission shaft output (wheel-side of transmission)
tx_eff_map	Map 4.2.24	Map 4.2.24	ADVISOR Default Value	Transmission efficiency (dependent on torque of transmission shaft output {vertically} and speed of transmission shaft output {horizontally})
fd_ratio	4.037	4.037	ADVISOR Default Value	Final Drive Ratio (5.34 is also sited as a good value for transit buses)
fd_inertia	0 kg-m^2	0 slugs-ft^2	ADVISOR Default Value	Rotational inertia of final drive
gb_mass	283.447 kg	19.42 slugs	ADVISOR Default Value	Mass of gearbox
fd_mass	90.703 kg	6.215 slugs	ADVISOR Default Value	Mass of final drive

4.2.8. Wheel Parameters

The wheel parameters were based upon those for a heavy truck (representative of a transit bus). Some of the wheel parameters are given as maps. These maps are defined below in SI units defined in Table 4.2.8.

Map 4.25
wh_axle_loss_mass=[0, 5000, 10000, 15000, 30000]

Map 4.26
wh_slip_force_coeff=[0, 0.3913, 0.6715, 0.8540, 0.9616, 1.0212]

Map 4.27
wh_axle_loss_trq=[0, 15, 30, 45, 90]

Map 4.28
wh_slip=[0.0, 0.025, 0.050, 0.075, 0.10, 0.125]

Map 4.29
wh_fa_dl_brake_frac=[0, 0, 0.5, 0.8, 0.8]

Map 4.30
wh_fa_fric_brake_frac=[0.8, 0.8, 0.4, 0.1, 0.1]

Table 4.2.8. Wheel parameters for the baseline bus model.

ADVISOR Parameter	Parameter Value (SI)	Parameter Value *(English)*	Source	Description
wh_axle_loss_mass	Map 4.2.25 (kg)	Map 4.2.25	ADVISOR Default Value	Vehicle test mass vector
wh_slip_force_coeff	Map 4.2.26	Map 4.2.26	ADVISOR Default Value	Coefficient defined as the tractive force on the front tires divided by the weight on the front axle
wh_axle_loss_trq	Map 4.2.27	Map 4.2.27	ADVISOR Default Value	Front brake and axle bearing drag torque
wh_slip	Map 4.2.28	Map 4.2.28	ADVISOR Default Value	Wheel slip of drive wheels
wh_radius	0.45 m	1.4764 ft	Specific for Flint Buses	Rolling radius of wheels
wh_fa_dl_brake_frac	Map 4.2.29	Map 4.2.29	ADVISOR Default Value	Fraction of braking done by driveline via front axle
wh_fa_fric_brake_frac	Map 4.2.30	Map 4.2.30	ADVISOR Default Value	Fraction of braking done by front axle friction brakes
wh_1st_rrc	0.00938	0.00938	Reference Value	1st coefficient of rolling resistance
wh_2nd_rrc	0	0	Reference Value	2nd coefficient of rolling resistance

4.3. Summary

A baseline hybrid bus model was developed in ADVISOR based upon early specifications for a bus in development with a joint project between Michigan State University, Kettering University, and TransTeq. This was important for early validation purposes for the project, as well as ensuring that the optimization strategies developed and explored in this thesis for HEV's would be valid for real-world hybrid applications. The development of the baseline simulation model based upon an existing bus also ensured that the results obtained for simulations were realistic. This chapter gave all of the details associated with the baseline model.

CHAPTER 5: OPTIMIZATION STRATEGY FOR HEV'S USING ADVISOR AS A SIMULATION TOOL

5.1 Introduction

The design of hybrid electric vehicles (HEV's) requires complex optimizations to account for the intricate interactions among the components, control strategies and the resulting effects on performance. Due to the size of HEV models and the complexity of interactions among the many components, it is challenging to properly define an optimization problem statement that invokes a final design with the desired performance characteristics. The optimization problem statement definition greatly affects the performance of the optimized HEV design, as well as the implications of the design. While this may seem obvious, the proper problem statement for achieving a particular performance goal is not always so clear.

In this chapter, the main characteristics of two optimization strategies that have been shown to be efficient in optimizing hybrid electric vehicles are merged into a single and more robust strategy that is shown to have several advantages in terms of design performance. The main purpose here is to explore these various strategies and associated optimization problem statements and their effects on the resulting hybrid vehicle design. As a consequence of this study, a practical and efficient approach is developed for sizing the components and defining the system control of HEV's to create efficient, reliable vehicles.

5.2 Hybrid Electric Bus Model

5.2.1 Component Sizing Procedure

For the purpose of the current study, we considered a series configuration hybrid electric bus as shown in Figure 5.2.1, with the baseline components described in Chapter 4, summarized in Table 5.2.1. The main components of interest for the optimization were the generator, electric motor, engine, and battery. During optimization, a scaling method was used to represent the size of each component relative to the baseline size.

By altering these scaling factors (which acted as the design variables during optimization), different size motors, engines, and generators could be evaluated. ADVISOR uses the scaling factor specified for a given component to linearly scale the relative internal parameters of that component. This linear scaling assumption makes for an easy way to change the size of a component without needing detailed experimental data for multiple sized components, so long as linear scaling is appropriate for the different size components in question. This is an approximation for a given component, but is a common and accepted way to size components of HEV's [9].

Values of the scaling parameters less than one corresponded to smaller sized devices, while values greater than one corresponded to larger sized devices. The baseline model was thus represented by the following parameter values:

Generator Speed Scale (*GC SPEED*):	1.0
Generator Torque Scale (*GC TORQUE):*	1.0
Motor Speed Scale (*MC SPEED*):	1.0
Motor Torque Scale (*MC TORQUE)*:	1.0

Engine Speed Scale (*FC SPEED*): 1.0

Engine Torque Scale (*FC TORQUE*): 1.0

Similarly, the battery size was altered using a scaling factor for the capacitance. In addition, the number of modules for the battery was altered during the study. The baseline battery was represented as follows:

Battery Capacitance Scale (*ESS CAP*): 1.0

Number of Battery Modules (*ESS MOD NUM*): 25

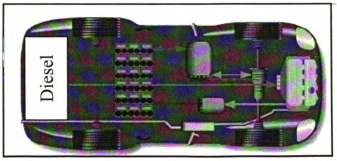


Figure 5.2.1. Powertrain configuration of the HEV bus (as it appears in ADVISOR).

Table 5.2.1. Baseline components of the series hybrid bus for optimization.

Component	Characteristics
Engine	Detroit Diesel Corp. Series 50 8.5 (205kW) Diesel Engine
Motor	UQM 150 kW motor/controller
Generator	UQM 150 kW generator
Battery	NIMH 90Ah Ovonic, 25 battery modules
Wheel and Axle	ACCURIDE, wh_radius = 0.45
Transmission	Single gear, TX_1SPD_BUS with overall ratio (8.074:1).
Accessory	Mechanical and electrical power scaled of ~ 21.4 kW (29 hp) with A/C

5.2.2 Control Strategy

A thermostat control strategy was used in the current study. This type of control strategy allows the user to specify a lower/upper limit for the state of charge (SOC) of the battery, below/above which the engine turns on/off and the generator charges/discharges the battery. In a thermostat control strategy, when the engine is on it is run at a constant speed (ideally that for which the engine operates most efficiently). For all optimization studies, the lower and upper limits on the state of charge for this thermostat control strategy were defined as:

Lower Limit on SOC:	0.52
Upper Limit on SOC:	0.68

These values were selected in accordance with an assumed requirement that the battery of the bus must operate with a state of charge between 52% and 68% in practice.

5.2.3 Driving Cycle

Two different driving cycles were considered the most viable to gauge the performance of a hybrid electric bus: the Federal Test Provisions Driving Cycle (FTP Driving Cycle) and Urban Dynamometer Driving Schedule (UDDS). The most demanding of the two driving cycles was judged to be the Urban Dynamometer Driving Schedule (UDDS), and therefore it was chosen for all comparative studies. It has the characteristics of Figures 5.2.2 and 5.2.3.

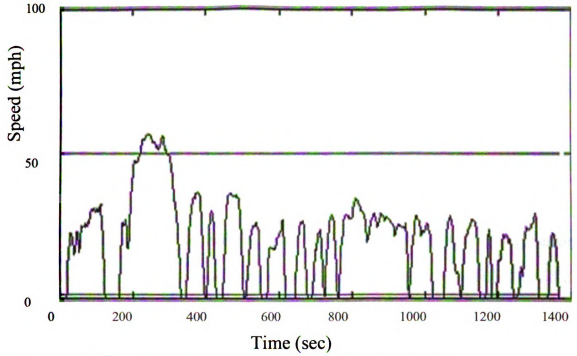


Figure 5.2.2. Driving cycle speed/elevation vs. time for the UDDS driving cycle.

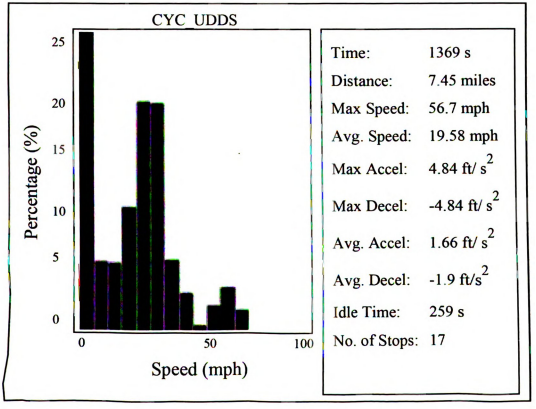


***Figure 5.2.3.* UDDS driving cycle characteristics.**

5.3. Baseline Designs

5.3.1 Conventional Bus Design

For comparison and validation purposes, an existing baseline conventional (non-hybrid) bus model was used. Starting with the validated conventional bus model, a baseline hybrid bus model was developed and used to gauge the performance of the different optimization strategies.

The conventional bus had the same basic design characteristics (e.g., engine, wheel and axle, transmission, and accessories) as the hybrid bus described in Table 5.2.1. The conventional bus had the following performance characteristics:

fuel economy: 4.7 mpgge

missed_trace: < 5 mph

t_{60} : 68.6 s

t_{30}: 13.3 s

where:

fuel economy is measured in miles per gallon gasoline equivalent (mpgge),

missed_trace is a gauge of how close the bus is to meeting the drive demands of the cycle (Figure 5.2.2),

t_{60} is the time required for the bus to accelerate from 0-60 mph, and

t_{30} is the time required for the bus to accelerate from 0-30 mph.

5.3.2 Baseline Hybrid Bus Design

The validated conventional bus model was used as a starting place in designing a hybrid bus. The baseline hybrid bus model had all of the characteristics of Table 5.2.1, and the following performance:

Average fuel economy: 3.71 mpgge

Corrected fuel economy: 4.07 mpgge

missed_trace: 15.7 mph

t_{60} : 52.67 s

t_{30}: 21.22 s

where:

Average fuel economy is a measure of the average fuel economy of the design evaluated with multiple initial SOC's over multiple run cycles of the UDDS with no SOC correction (measured in miles per gallon gasoline equivalent),

Corrected fuel economy is a measure of the fuel economy over one run cycle of the UDDS with an SOC correction routine as described by [23-27] (measured in miles per gallon gasoline equivalent),

missed_trace is a gauge of how close the bus is to meeting the drive demands of the cycle (Figure 5.2.2),

t_{60} is the time required for the bus to accelerate from 0-60 mph, and

t_{30} is the time required for the bus to accelerate from 0-30 mph.

Note that the baseline hybrid bus did not meet the target for missed trace, and acceleration times, and is thus considered an infeasible design. These performance targets will subsequently be enforced as constraints during optimization. The fuel economies of this baseline hybrid design will be used for assessing new designs based on the proposed optimization strategies.

5.4. Modeling Concepts for HEV Optimization

5.4.1. Introduction

For the various optimization studies performed in this work, the optimization goal was to size the various components of interest in a manner that minimized fuel consumption while meeting performance characteristics for a given drive cycle. One of the more challenging and interesting aspects of HEV optimization is identifying an appropriate definition of fuel economy and subsequently minimizing its sensitivity to various assumptions in the model and to the drive cycle used during the simulation for each design. In this section, an overview of why these issues arise is presented as a prelude to the detailed investigation described in the next section.

5.4.2 SOC Corrections

SOC corrections in some form are used in the vast majority of optimization work with HEV's to date. SOC corrections are important in the design of HEV's for multiple reasons. They provide a meaningful way to compare HEV performance to conventional vehicle performance [23-24], as well as reduce the impact the initial state of charge has on the performance of the design [9, 25]. By ensuring initial and final SOC over a drive

cycle are the same (or nearly the same), SOC corrections alleviate the need for multiple drive cycles to be performed to see how performance changes with varying initial SOC. This provides an easy and seemingly robust way to compare designs during optimization [25].

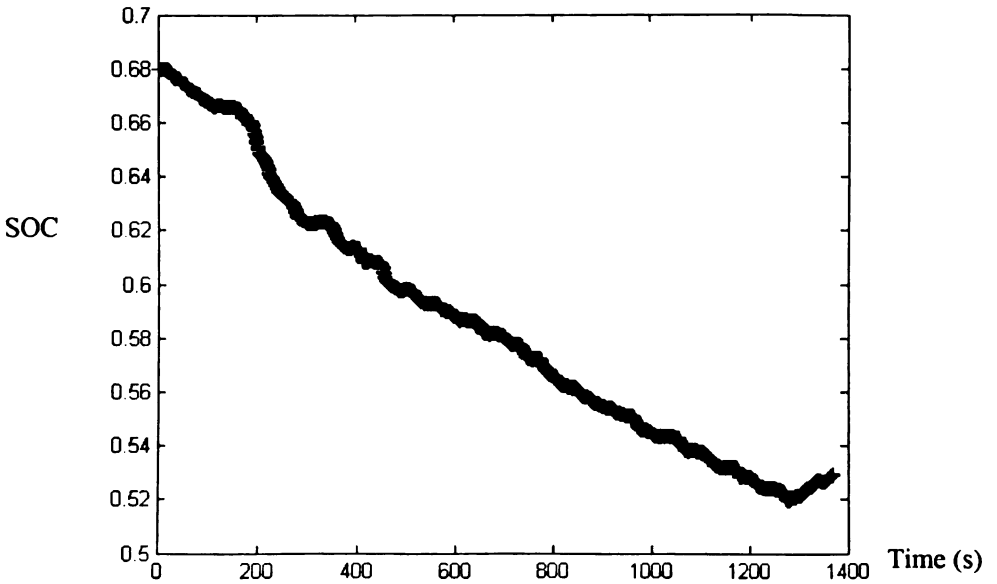
5.4.3 Multiple Initial SOC's

It was desired to see the effects on the results obtained from optimization if an SOC correction was not utilized. A preliminary optimization was therefore performed that sized the engine, generator, motor, and battery, such that the fuel economy of the engine was maximized and certain performance characteristics were met. This optimization run had an initial SOC of 0.68 and did not utilize SOC correction.

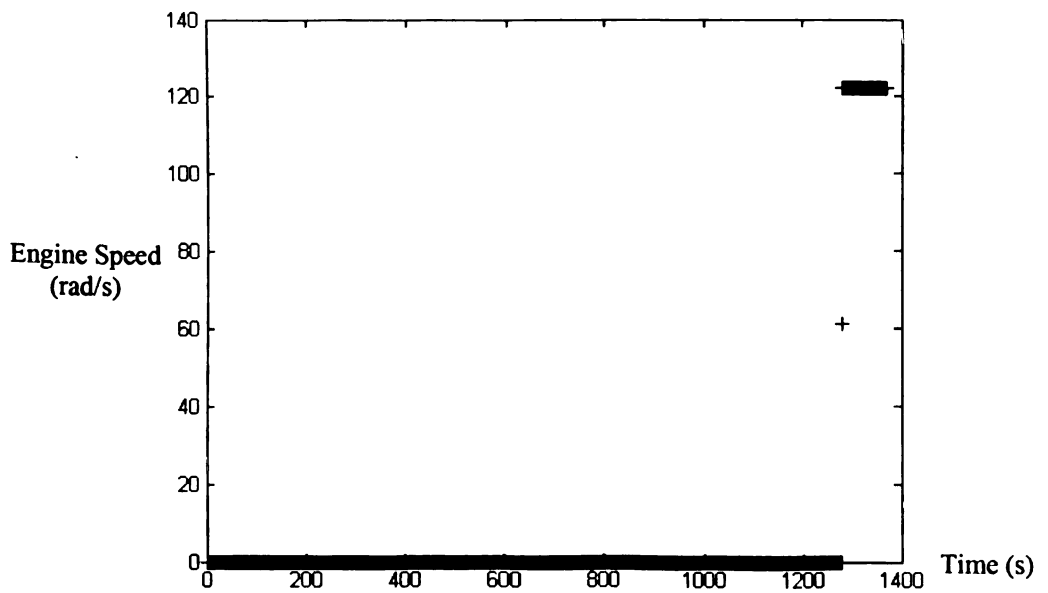
The optimized design found had the SOC and engine speed history plots shown in Figure 5.4.1. The components for this design were sized such that the design acted almost entirely as an electric vehicle (EV) over the drive cycle, with the engine not needing to turn on until the end of the drive cycle. This EV behavior gave a very high performance during optimization since no battery losses were accounted for. These results are consistent with the trends seen by Markel and Wipke [18-19].

If however, the optimized design had an initial SOC less than 0.68, it would not perform as an EV over the drive cycle. Figure 5.4.2 shows the SOC and engine speed history plots for the same design found as optimal during the optimization, re-evaluated *in ADVISOR* with an initial SOC of 0.60 instead of 0.68. Clearly the design now does not act as an EV over the drive cycle, and therefore its performance will be drastically *lower than* that found during the optimization where it did act as an EV. It was concluded

therefore, that if an SOC correction routine is not utilized, multiple initial SOC's should be considered when evaluating an HEV's performance since it is not guaranteed that a vehicle will always start with a high initial SOC.

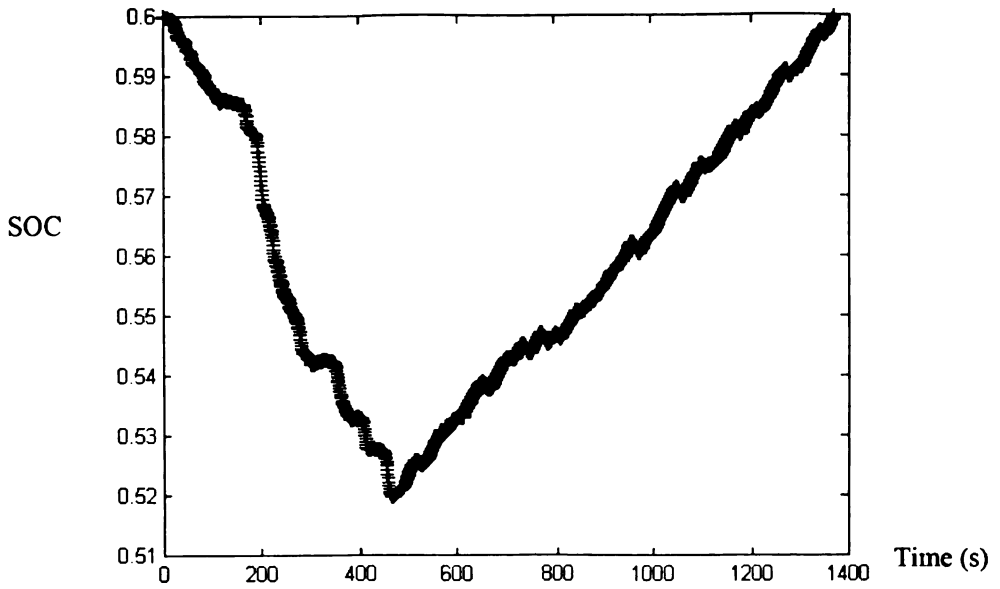


(a) SOC vs. time.

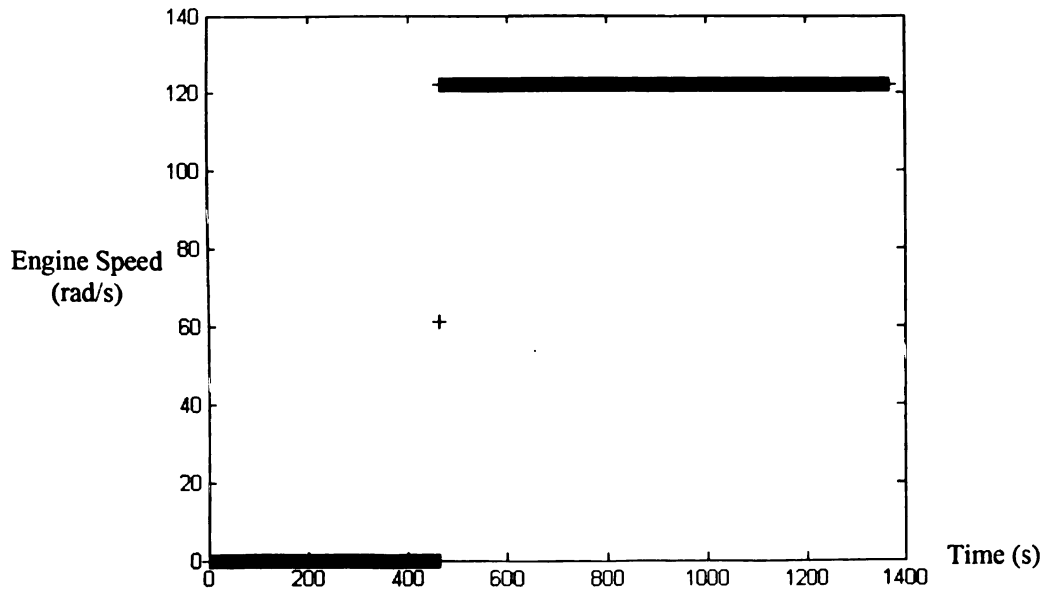


(b) ω vs. time.

Figure 5.4.1. Optimized design characteristics for one drive cycle of the UDDS (SOC=state of charge; ω=engine speed) with an initial SOC of 0.68.



(a) SOC vs time.



(b) ω vs. time.

Figure 5.4.2. Optimized design characteristics (SOC=state of charge; ω=engine speed) if re-evaluated with an initial SOC of 0.60.

For a given drive cycle, an HEV performs differently depending upon its initial SOC [9]. A design's performance therefore will be skewed depending upon the initial

SOC that it is evaluated for (performing well at one initial SOC and poor at another). A method to overcome the sensitivity to initial SOC is to evaluate a given design for multiple initial SOC's and average the performance. One issue with this approach is selecting which initial SOC's to consider in evaluating the average performance.

5.4.4 Multiple Cycle Repeats

If an SOC correction routine is not utilized, evaluating designs based upon the average performance from multiple initial SOC's over a single drive cycle may be inaccurate. Utilizing the optimized design from Sub-Section 5.4.3, a study was performed to see the effect the number of consecutive UDDS drive cycles has on the responses. The optimized design was altered so that the initial SOC was 0.60 instead of 0.68 so that it wouldn't run as an EV. This design was then run for 1, 3, and 20 drive cycles. When performing multiple drive cycles, a periodic behavior was observed in the SOC and engine speed time histories, as shown in Figures 5.4.3 and 5.4.4. Thus, as the number of drive cycle repetitions is increased, the dependency on the initial SOC decreases while the evaluation time increases. Since the period of the SOC time history for the evaluated bus appeared to be slightly less than three drive cycles (as seen in Figure 5.4.3), this was the number of drive cycle repeats used herein whenever an optimization strategy required multiple drive cycles.

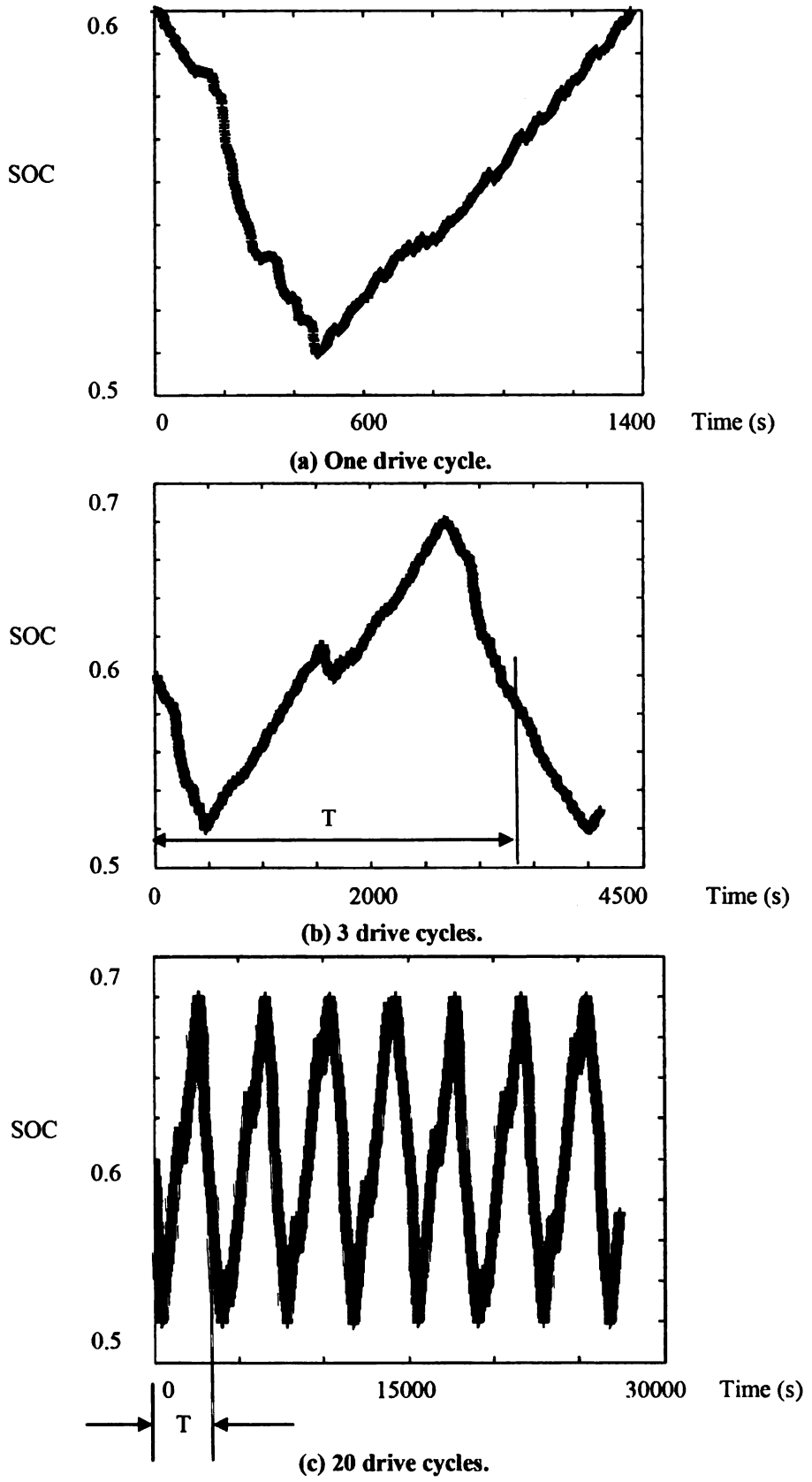
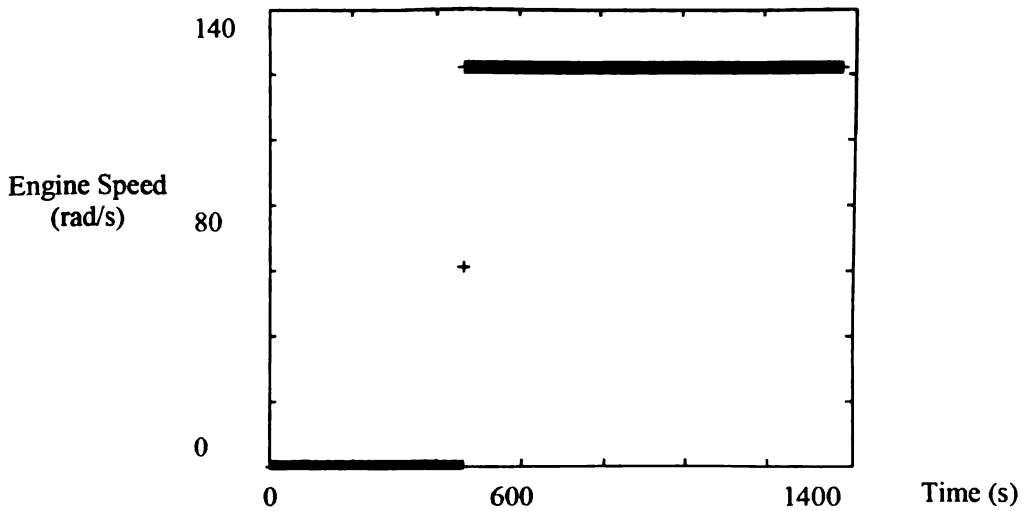
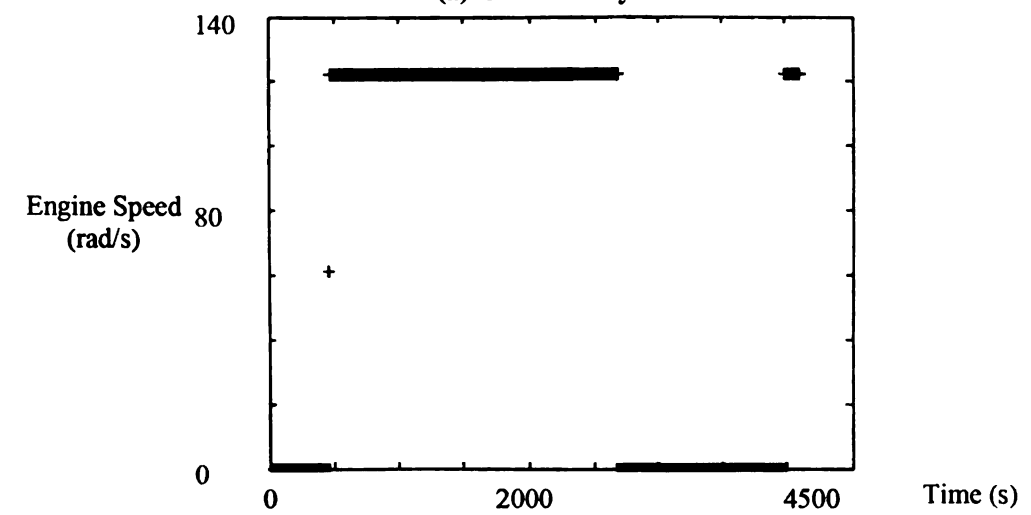


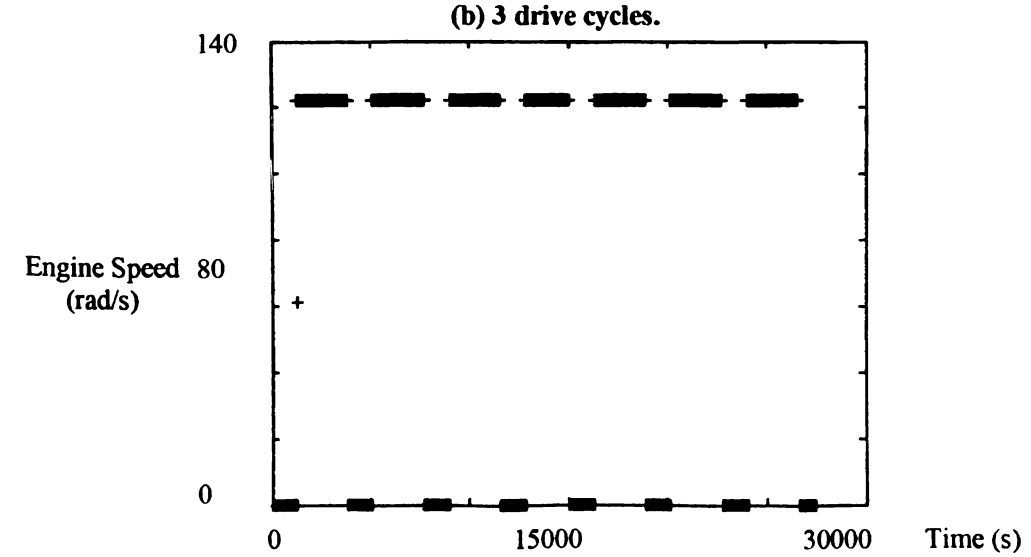
Figure 5.4.3. SOC history plots for an optimized design evaluated for different number of drive cycles. The period (T) of the cycle corresponds with just less than three cycle repetitions.



(a) One drive cycle.



(b) 3 drive cycles.



(c) 20 drive cycles.

Figure 5.4.4. Engine speed history plots for an optimized design evaluated for different number of drive cycles.

Due to the periodic nature of the SOC history plots with drive cycle repetitions, a design should be evaluated during optimization for multiple cycle repetitions at multiple initial SOC's, in the hopes that this methodology decreases the dependence of a design's performance on its initial SOC.

5.4.5 Corrected Fuel Economy

Duoba [26] and Senger [27] discuss how a conversion for electrical energy usage can be used as an alternative to SOC correction, by converting the electrical energy that can be obtained from gasoline through the efficiency of an average power plant in the United States in 1996. The fuel economy values used with the concepts discussed above base the fuel calculations on the actual volume of fuel used. But in a hybrid vehicle, some amount of electric energy is also used during the driving cycle. Therefore one could postulate a 'corrected fuel economy' measure as an alternative objective, in which the equivalent fuel energy of electrical energy used is added to actual fuel energy used. This approach provides a more consistent comparison among designs and effectively results in 'perfectly charge sustaining' vehicle behavior.

5.5 Optimization Strategies and Problem Statements

Three optimization strategies for optimizing HEV's have been developed and compared using the concepts discussed in Sub-Section 5.4. The definition of the strategies, as well as the formal optimization problem statement utilized with each are discussed in this sub-section.

5.5.1 Strategy 1-Multiple SOC's and Multiple Drive Cycle Repetitions

The first strategy utilizes multiple initial SOC's and multiple drive cycle repetitions to evaluate the performance of a given design. No state of charge correction is utilized for this strategy, and a measure of the actual volume of fuel used instead of the corrected fuel economy. Three analyses are performed for each design, each with a different initial state of charge (SOC). Two of the analyses use an initial state of charge at the extremes of the SOC range specified for the thermostat control strategy (*High SOC* and *Low SOC*), while the third analysis uses an initial SOC that is the average of the extreme SOC's. By using multiple drive cycle repetitions (three repetitions utilized in this study) in three different analyses, each with different initial SOC's, the performance of a given design should not be highly dependent upon the initial SOC used.

Strategy 1, thus has the following fuel economy performance criterion:

$$f = \frac{\sum_{i=1}^{N} f_i}{N} \tag{5.1}$$

where:

f is the *composite fuel economy* (average fuel economy over all analyses),

f_i is the fuel economy for a given analysis, and

N is the number of analyses performed (3 here: each with a different initial SOC).

The optimization problem statement for Strategy 1 as used in this study is formally expressed as:

Maximize: *composite fuel economy* (average miles per gallon of gasoline (mpg) as defined by Equation (5.1))

Subject to: *missed_trace (mph)* $\leq$ 5.0 (how close the bus was to meeting the drive demands of the cycle)
minimum SOC $\geq$ 0.5175 (lowest state of charge over all initial SOC loadcases)
$t_{60} \leq 42.0$ s (time to accelerate to 60 mph from start)
$t_{30} \leq 10.0$ s (time to accelerate to 30 mph from start)

By varying: *Number of battery modules* = {15,16, .. ,50}
0.3 $\leq$ *Battery capacitance Scale* $\leq$ 1.5
0.5 $\leq$ *Engine Speed Scale* $\leq$ 1.5
0.5 $\leq$ *Engine Torque Scale* $\leq$ 1.5
0.5 $\leq$ *Generator Speed Scale* $\leq$ 1.5
0.5 $\leq$ *Generator Torque Scale* $\leq$ 1.5
0.5 $\leq$ *Motor Speed Scale* $\leq$ 3.0
0.5 $\leq$ *Motor Torque Scale* $\leq$ 3.0

With: *Upper SOC Limit* = 0.68
Lower SOC Limit = 0.52
Initial SOC = {0.68, 0.60, and 0.52} (multiple initial SOC load cases)

5.5.2 Strategy 2 – SOC Correction with Corrected Fuel Economy

The second strategy utilizes a SOC correction as well as considers the corrected fuel economy as opposed to the actual volume of fuel used. Within ADVISOR, there is a "Zero-Delta SOC correction routine" available [9]. This routine can be set to iterate on the initial SOC until a tolerance on SOC is satisfied. The routine adjusts the initial SOC until the simulation run yields either a zero change in SOC +/- a tolerance (making the initial SOC more or less equal to the final SOC), or until the routine runs for the maximum number of iterations specified by the user. The default ADVISOR tolerance and maximum iterations were utilized in this study: 0.5% and 15, respectively. The algorithm for this routine is described in the following steps (as described in the ADVISOR User's Manual [9]) and by Figure 5.5.1 below:

1. An initial SOC is guessed that is the average of the high and low limits:

$$SOC_{initial,0} = \frac{SOC_{high} + SOC_{low}}{2} \tag{5.2}$$

2. The new initial SOC ($SOC_{initial,t+1}$) is guessed based upon the previous initial SOC ($SOC_{initial,t}$), the change in SOC achieved (ΔSOC_t), and a weighting factor (q). The weighting factor is chosen so that the charge neutral point (zero ΔSOC) is bracketed quickly.

$$SOC_{initial,t+1} = SOC_{initial,t} + q * \Delta SOC_t \quad (5.3)$$

3. Once the charge neutral state is bracketed (values for the metrics are known on both the positive ΔSOC and the negative ΔSOC, the new initial SOC is linearly interpolated.

$$SOC_{initial,t+1} = interp[SOC(\Delta SOC_{min,(+)}), SOC(\Delta SOC_{min,(-)})] \quad (5.4)$$

4. The simulation is then run and the ΔSOC evaluated, the bracket size shrinks, and step 3 is repeated until the ΔSOC is within the tolerance band or the maximum iterations are run.

5. Final simulation iteration is run at the initial SOC that gives ΔSOC within the tolerance, and only those results are reported from the ADVISOR simulation.

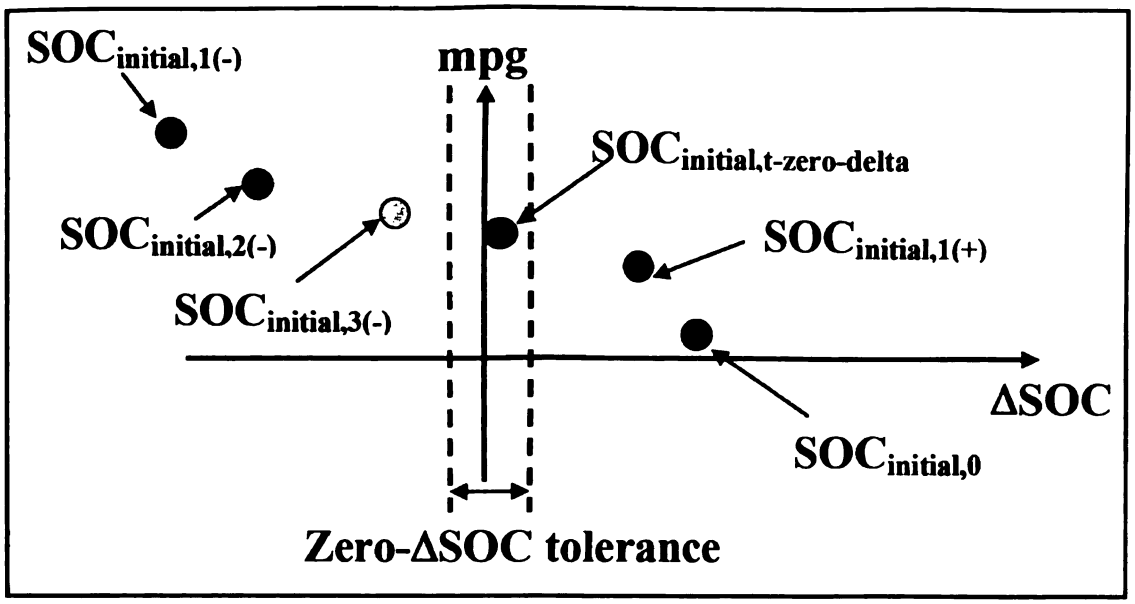


Figure 5.5.1. Zero-Delta SOC correction routine. An initial SOC guess is evaluated ($SOC_{initial,0}$) in terms of ΔSOC. The initial bounds around the zero-ΔSOC tolerance are then found ($SOC_{initial,1(-)}$ and $SOC_{initial,1(+)}$) and linearly interpolated to get $SOC_{initial,2(-)}$. This bound ($SOC_{initial,2(-)}$ and $SOC_{initial,1(+)}$) is then linearly interpolated and the process repeated until an initial SOC is found that gives a ΔSOC within the tolerance ($SOC_{initial,t\text{-}zero\text{-}delta}$) [9].

While the "Zero-Delta SOC correction routine" attempts to ensure that the energy input to the battery is the same as the energy output from the battery over the drive cycle, this is not always the case due to the tolerance and maximum iterations specified for the routine. For this reason, Strategy 2 also converts the difference in battery energy into gasoline equivalent energy to obtain the corrected fuel economy. This is done using a value for the electrical energy that can be converted to gasoline through the efficiency of an average power plant in the United States (3.11 kWh/liter in 1996). With this method, a 'fuel penalty' term (*e*) is calculated based upon the gasoline equivalence described above [26-27]. The fuel penalty *e* is defined as:

$$e = \frac{(ess_{out} - ess_{in})*0.000278}{3.11} *0.264172 \quad (5.5)$$

where:

ess_{out} is the useful energy leaving the batteries over the drive cycle (units of kJ),

ess_{in} is the energy into the storage system over the drive cycle (units of kJ),

0.000278 is the conversion factor from kJ to kWh,

3.11 is the electrical energy that can be converted to gasoline through efficiency of an average power plant (units of kWh/L),

0.264172 is the conversion factor from L to gallons, and

e is the fuel penalty (units of gallons).

A positive value for the fuel penalty ($e > 0$) indicates that electrical energy is used to propel the vehicle and thus acts as a penalty to the fuel economy of the vehicle. A negative value for the fuel penalty ($e < 0$) indicates that fuel has been used to charge the battery and thus acts as additional fuel economy to the vehicle (boosting its performance value). Figure 5.5.2 shows the logic-loop that evaluates the corrected fuel economy for a given design using the fuel penalty calculated for that design. For Strategy 2:

- *e* will be referred to as the fuel penalty; it is considered as the fuel used to operate the battery
- *mpgge* is the drive cycle mi/gal gasoline equivalent calculated by ADVISOR (the same as the fuel economy used in Strategy 1)

- *dist* will represent the distance traveled over the simulation
- *f* is the *corrected fuel economy*

This strategy can also be thought of as calculating the consumed fuel (that of the actual fuel used during the run (*mpgge*)), a fuel penalty (*e*), as well as a corrected fuel economy (*f*) (the distance traveled (*dist*) per consumed fuel).

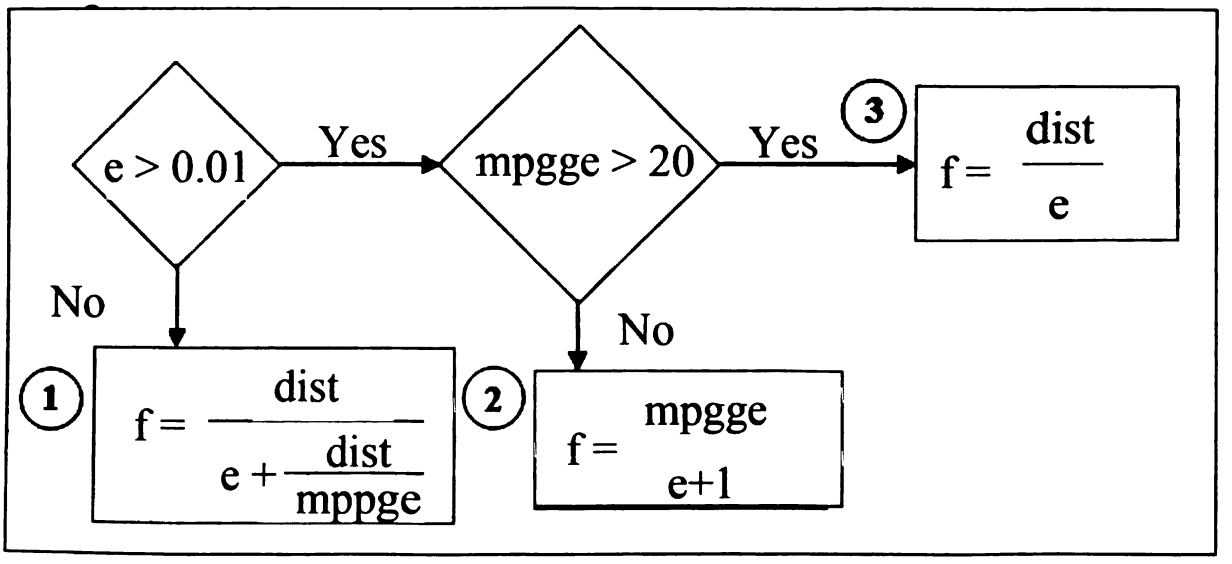


Figure 5.5.2. Logic used to calculate the corrected fuel economy. Scenario 1 occurs when the fuel penalty is small (or negative) and makes the corrected fuel economy equal to: the drive cycle mi/gal gasoline equivalent for small positive values, and greater than the drive cycle mi/gal gasoline equivalent for negative values. Scenario 2 occurs when the battery equivalent is large but the drive cycle mi/gal gasoline equivalent is not too large, thus reduces the corrected fuel economy to account for the fuel penalty during the run. Scenario 3 occurs when the vehicle operates almost entirely as an electric vehicle (EV) and therefore treats the corrected fuel economy as if the vehicle were completely EV with all of its fuel used coming from the fuel penalty.

The optimization problem statement for Strategy 2 as used in this study is formally expressed as:

Maximize:	*corrected fuel economy* (accounting for total energy usage with miles per gallon gasoline equivalent (mpgge) as defined by Figure 5.5.2 and Equation (5.5))
Subject to:	*missed_trace (mph)* $\leq$ 5.0 (how close the bus was to meeting the drive demands of the cycle)
	minimum SOC $\geq$ 0.5175 (lowest state of charge achieved)
	$t_{60} \leq 42.0$ s (time to accelerate to 60 mph from start)
	$t_{30} \leq 10.0$ s (time to accelerate to 30 mph from start)
By varying:	*Number of battery modules* = {15,16, .. ,50}
	0.3 $\leq$ *Battery capacitance Scale* $\leq$ 1.5
	0.5 $\leq$ *Engine Speed Scale* $\leq$ 1.5
	0.5 $\leq$ *Engine Torque Scale* $\leq$ 1.5
	0.5 $\leq$ *Generator Speed Scale* $\leq$ 1.5
	0.5 $\leq$ *Generator Torque Scale* $\leq$ 1.5
	0.5 $\leq$ *Motor Speed Scale* $\leq$ 3.0
	0.5 $\leq$ *Motor Torque Scale* $\leq$ 3.0
With:	*Upper SOC Limit* = 0.68
	Lower SOC Limit = 0.52
	Zero-Delta SOC correction routine utilized

5.5.3 Strategy 3 – A Hybrid Approach

The third strategy utilizes multiple initial SOC's and multiple drive cycle repetitions to evaluate the performance of a given design, along with the corrected fuel economy. No state of charge correction is utilized for this strategy. As in Strategy 1, three analyses are performed for each design, each with a different initial state of charge (SOC). Two of the analyses use an initial state of charge at the extremes of the SOC range specified for the thermostat control strategy (*High SOC* and *Low SOC*), while the third analysis uses an initial SOC that is the average of the extreme SOC's. In addition,

the strategy utilizes multiple drive cycle repetitions (three repetitions utilized in this study) for each initial SOC analysis, as is done with Strategy 1.

Whereas Strategy 1 utilizes the actual volume of fuel used, Strategy 3 utilizes the corrected fuel economy as defined by Equation (5.5) and Figure 5.5.2. While Strategy 2 utilizes a SOC correction routine, Strategy 3 does not. Rather than trying to find the design with the greatest average fuel economy or the design with the greatest corrected fuel economy, Strategy 3 tries to find the design with the greatest *average corrected fuel economy*:

$$f = \frac{\sum_{i=1}^{N} f_i}{N} \tag{5.6}$$

where:

f is the *average corrected fuel economy* over all analyses,

f_i is the corrected fuel economy for a given analysis utilizing Equation (5.5) and Figure 5.5.2,

N is the number of analyses performed (3 here: each with a different initial SOC).

The optimization problem statement for Strategy 3 as used in this study is therefore formally expressed as:

Maximize:	*average corrected fuel economy* (accounting for total energy usage with miles per gallon gasoline equivalent (mpgge) as defined by Equation (5.6), Equation (5.5), and Figure 5.5.2)
Subject to:	*missed_trace (mph)* $\leq$ 5.0 (how close the bus was to meeting the drive demands of the cycle)
	minimum SOC $\geq$ 0.5175 (lowest state of charge over all initial SOC loadcases)
	$t_{60} \leq 42.0$ s (time to accelerate to 60 mph from start)
	$t_{30} \leq 10.0$ s (time to accelerate to 30 mph from start)
By varying:	*Number of battery modules* = {15,16, .. ,50}
	0.3 $\leq$ *Battery capacitance Scale* $\leq$ 1.5
	0.5 $\leq$ *Engine Speed Scale* $\leq$ 1.5
	0.5 $\leq$ *Engine Torque Scale* $\leq$ 1.5
	0.5 $\leq$ *Generator Speed Scale* $\leq$ 1.5
	0.5 $\leq$ *Generator Torque Scale* $\leq$ 1.5
	0.5 $\leq$ *Motor Speed Scale* $\leq$ 3.0
	0.5 $\leq$ *Motor Torque Scale* $\leq$ 3.0
With:	*Upper SOC Limit* = 0.68
	Lower SOC Limit = 0.52
	Initial SOC = {0.68, 0.60, and 0.52} (multiple initial SOC load cases)

5.6 Results of the Optimization Studies

5.6.1 Strategy Comparisons

The commercial software HEEDS [29] was used as the optimization tool for all three strategies, and the hybrid adaptive search algorithm SHERPA [29] was employed. Three independent optimization runs were performed for each strategy to lessen the effect of the random starting conditions of the search on the optimized design found. Similar to a genetic algorithm, SHERPA begins with a set of quasi-random initial designs (unless these designs are otherwise provided) that depend loosely on the value of an arbitrary number, called a random seed. When the same or similar results are obtained from multiple optimization runs with different random seeds, there is greater confidence (but no guarantee) that the results found are at or near a global optimum. In the current study, each optimization run was allowed to continue for an extended period of time, up to a maximum of 1000 evaluations. Though all the optimization runs may not have been fully converged (something that can never be guaranteed due to the nature of the design landscapes for these problems), by 1000 evaluations the rate of improvement was very low in all cases, so the runs were considered to be very nearly converged (see a typical example in Figure 5.6.1).

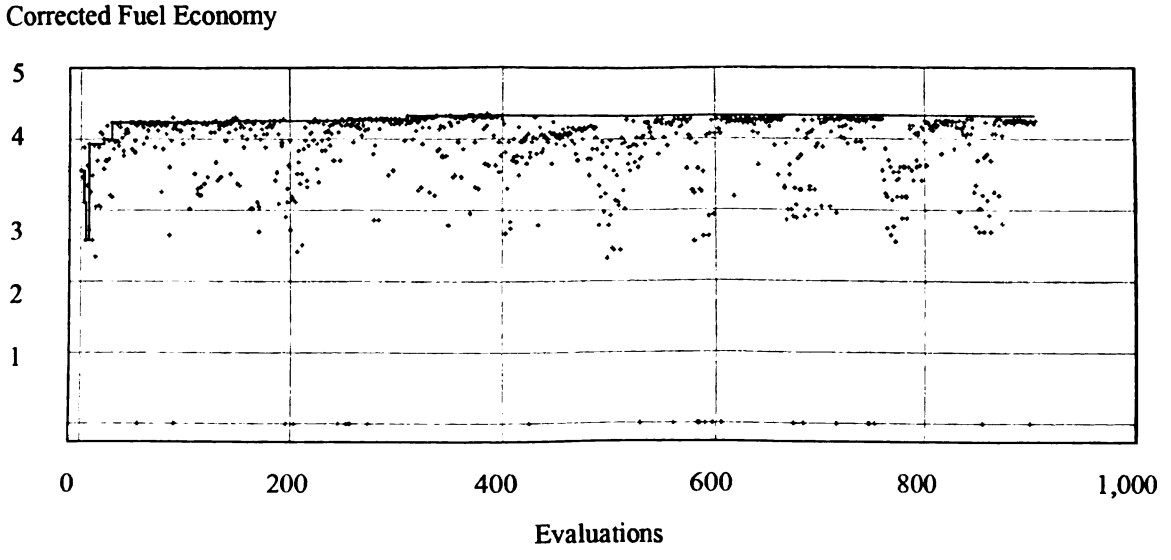


Figure 5.6.1. Typical convergence plot for the HEV optimizations. The shown optimization was nearly converged by ~300 evaluations.

The optimized designs from all three strategies are collected in Table 5.6.1 and Table 5.6.2. Recall that Strategy 1 maximizes the fuel economy from the engine only (miles per gallon of gasoline), while Strategies 2 and 3 maximize the total energy usage (miles per gallon gasoline-equivalent). Because the objective functions associated with each strategy are different, it is difficult to directly compare the solutions obtained from each strategy. Therefore, the optimized designs from Strategy 1 and Strategy 2 were re-analyzed using the criteria from the opposite strategy to facilitate this comparison. In other words, the optimized designs from Strategy 1 were re-analyzed with a single drive cycle utilizing a SOC correction so that their corrected fuel economy could be calculated. Likewise the optimized designs from Strategy 2 were re-analyzed with no SOC correction but rather multiple drive cycles for multiple initial SOC values so that their average fuel economy could be calculated. Strategy 3 calculates the average fuel economy in terms of the corrected fuel economy; therefore its optimized designs did not need to be re-analyzed for comparative purposes.

The fact that each strategy found three different optimized designs with similar response values is an indication of the difficult design landscapes associated with HEV design. As Montazeri-Gh, et al [15] showed, HEV optimization problems are typically highly non-linear, discontinuous, and multi-modal (i.e., multiple local optima exist). Adding discrete variables to the problem statement as was done in this study, only complicates the optimization search further. The three different optimized designs for a given strategy would seem to indicate that the optimization is not fully converged. To check further for the global optima, local search methods could be used around the optimized designs to see if further improvement could be attained. Local search methods have been shown to find marginal improvement in HEV designs as shown by Oh, et al [14].

The similarity in responses among the three designs for a given strategy could have multiple explanations. It is possible that multiple local minima with similar performances exist in the design landscape (see Figure 5.6.2 for an example), or that many designs with similar performances exist in a flat region in the design space (see Figure 5.6.3 for an example), among other things. Determining the cause for the seemingly un-converged similar solutions is beyond the scope of this study.

When considering total energy usage, the optimized designs from Strategies 2 and 3 have a significantly greater corrected fuel economy than the optimized designs from Strategy 1. Strategy 3 produced designs with a marginally improved corrected fuel economy over the optimized designs from Strategy 2. Strategies 2 and 3 have consistently high performing designs for all three optimization runs. Compared to Strategy 2, Strategy 3 has the added advantage of improved run-time efficiency. In the current study Strategy

3 runs are about 10% more efficient than Strategy 2 runs. This speedup will vary based on the number of iterations required to perform the SOC correction within Strategy 2. The speedup would also be affected by the number of initial SOC load cases and drive cycle repeats utilized within Strategy 3. Overall, Strategy 3 appears to be an effective and efficient alternative to the commonly used Strategy 2.

When considering only the energy usage from the engine, Strategy 1 produces designs with the highest average fuel economy, significantly outperforming Strategies 2 and 3. This can be attributed to the very high fuel economies achieved when the initial SOC is high, corresponding to the 0.68 initial SOC runs. For the three initial SOC's chosen for this study (0.68, 0.6, and 0.52) the optimizations with Strategy 1 sized the components such that they operated almost entirely as an electric vehicle (EV) when the initial SOC was 0.68. This strategy's optimized designs had high values for battery capacitance, a high number of battery modules, and also had less powerful engines than the optimized designs from Strategies 2 and 3, as shown in Table 5.6.2. Since the average fuel economy is the average of performance over all initial SOC load cases, the EV nature of the optimized Strategy 1 designs when the initial SOC is 0.68 gives a very high overall average fuel economy as well as a high standard deviation. These designs have a lower performance than those of Strategies 2 and 3 when the relevant measure is overall energy usage. Nevertheless, in those limited cases where a fixed route can be guaranteed (as may be the case for some buses) and the fuel usage of the engine is the primary concern, then Strategy 1 may be very efficient for optimizing such HEV's.

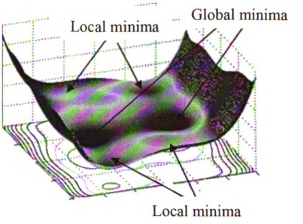


Figure 5.6.2. Three-dimensional contour plot of the local region surrounding the global minima locations of the Six Hump Camel Back Function. Note the multiple local minima and multiple global minima with identical response values.

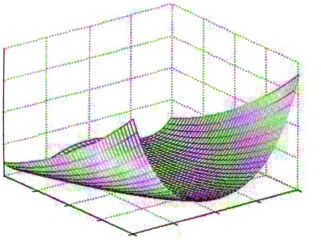


Figure 5.6.3. Three-dimensional contour plot of the Rosenbrock's Valley function for two variables. Note the similar response values along the valley close to the global optima value.

Table 5.6.1. Optimization results obtained from all three strategies. Where three results are reported for a run, each corresponds with the results from a given initial SOC load case (0.68, 0.60, 0.52). The highlighted results correspond with the objective value for a given strategy.
***Strategy 2 results were re-analyzed with no SOC correction and run with the multiple drive cycle iterations and multiple initial SOC analyses used for Strategy 1.**
****Strategy** 1 results were re-analyzed with the SOC correction settings of Strategy 2 for one drive cycle.

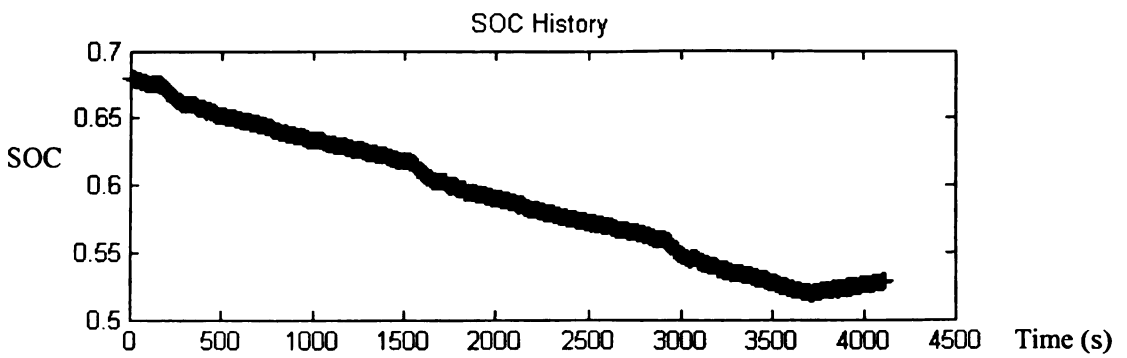
		Strategy 1				Strategy 2				Strategy 3			
Response	*Initial SOC*	*Run 1*	*Run 2*	*Run 3*	*Strategy Avg.*	*Run 1*	*Run 2*	*Run 3*	*Strategy Avg.*	*Run 1*	*Run 2*	*Run 3*	*Strategy Avg.*
Fuel Econ. {mpg}	0.68	30.94	28.28	30.63	29.95	4.03*	4.50*	3.86*	4.13*	4.24	4.23	3.90	4.12
	0.60	5.53	5.29	5.49	5.44	4.11*	3.95*	3.69*	3.92*	3.82	4.00	3.77	3.86
	0.52	3.21	3.07	3.19	3.16	3.55*	3.64*	3.54*	3.58*	3.45	3.88	3.55	3.63
Avg. Fuel Econ. {mpg} (Std. Dev)	-	13.23 (15.38)	12.22 (13.96)	13.10 (15.22)	12.85	3.90* (0.30*)	4.03* (0.44*)	3.70* (0.16*)	3.88*	3.84 (0.39)	4.04 (0.18)	3.74 (0.18)	3.87
Corrected Fuel Econ. {mpgge}	0.68									4.39	4.30	4.42	4.37
	0.60	4.01**	3.96**	3.86**	3.94**	4.30	4.33	4.30	4.31	4.41	4.30	4.42	4.38
	0.52									4.43	4.30	4.43	4.39
Avg. Corrected Fuel Econ. {mpgge} (Std. Dev)	-	4.01	3.96	3.86	3.94	4.30	4.33	4.30	4.31	4.41 (0.02)	4.30 (0)	4.42 (0.006)	4.38
Missed Trace{mph}	0.68	2.12	2.12	2.12	2.12					2.45	2.46	3.08	2.66
	0.60	2.12	2.12	2.12	2.12	3.82	2.12	3.99	3.31	2.50	2.43	3.13	2.69
	0.52	2.12	2.12	2.12	2.12					2.45	2.41	3.08	2.65
Minimum SOC	0.68	0.519	0.520	0.519	0.519					0.519	0.518	0.518	0.518
	0.60	0.520	0.520	0.520	0.520	0.518	0.540	0.519	0.526	0.518	0.520	0.518	0.519
	0.52	0.515	0.516	0.515	0.515					0.519	0.520	0.518	0.519
Average SOC	0.68	0.591	0.591	0.591	0.591					0.610	0.603	0.614	0.609
	0.60	0.545	0.547	0.545	0.546	0.590	0.611	0.591	0.597	0.595	0.605	0.601	0.600
	0.52	0.537	0.542	0.538	0.539					0.603	0.60	0.608	0.604
Accel. 0-60 {s}	-	17.64	17.68	17.65	17.66	37.60	24.71	38.84	33.72	25.43	27.92	28.83	27.39
Accel. 0-30 {s}	-	6.61	6.62	6.61	6.61	8.11	6.77	8.26	7.71	6.97	7.20	7.32	7.16

Table 5.6.2. Design variable values corresponding with the optimized designs from each strategy.

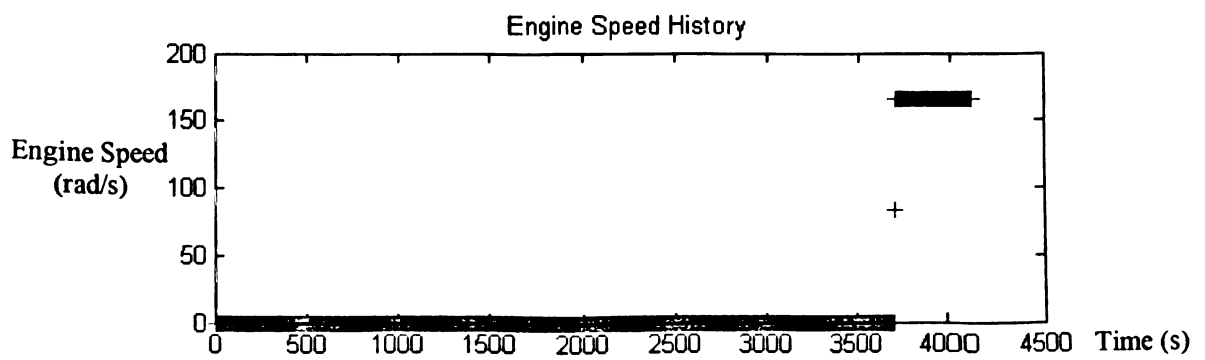
	Strategy 1			Strategy 2			Strategy 3		
Variable	*Run 1*	*Run 2*	*Run 3*	*Run 1*	*Run 2*	*Run 3*	*Run 1*	*Run 2*	*Run 3*
ESS CAP	1.5	1.5	1.5	0.3	0.3	0.3	0.3	1.128	0.312
ESS MOD NUM	50	50	50	29	42	28	40	31	35
FC SPEED	1.15	0.6	0.77	1.14	1.4	0.83	1.28	1.38	1.18
FC TORQUE	0.65	1.31	0.98	0.96	0.69	1.47	1.18	0.76	1.28
GC SPEED	1.32	0.5	0.70	1.05	0.63	0.61	1.05	0.97	0.87
GC TORQUE	0.89	1.5	1.44	1.22	1.39	1.45	0.96	1.3	1.5
MC SPEED	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
MC TORQUE	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0

Figures 5.6.4-5.6.6 display the SOC and engine speed history plots of a typical optimized design from Strategy 1 from this study (the result from Run 1 is shown). Figure 5.6.7 shows the SOC and engine speed history plots of the same optimized design when it is re-analyzed with the SOC correction of Strategy 2. Conversely, Figures 5.6.8 displays the SOC and engine speed history plots for the optimized design of Run 1 from Strategy 2, wherein the objective was to maximize total energy instead of engine fuel usage only, and Figures 5.6.9-5.6.11 show the same optimized design if it is re-analyzed with no SOC correction and using the settings from Strategy 1.The trends in Figures 5.6.4-5.6.7 versus those in Figures 5.6.8-5.6.11 are significantly different. When comparing engine fuel usage only with Figures 5.6.4-5.6.6 and Figures 5.6.9-5.6.11, the Strategy 1 design runs primarily as an EV for high initial SOC's whereas the Strategy 2 design does not. The components of the Strategy 1 solution have been designed such that

the engine runs continually for long durations of time in order to recharge a nearly depleted battery. Conversely, the Strategy 2 design has a more evenly distributed load balance between the engine and electric motor. When comparing total energy usage in Figure 5.6.7 and Figure 5.6.8, the Strategy 1 design has the engine run around 50% longer than the Strategy 2 design.

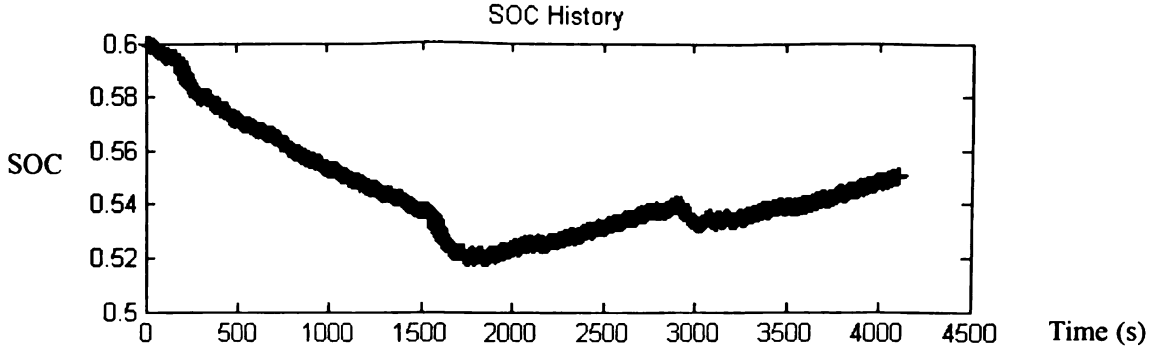


(a) SOC history with initial SOC = 0.68.

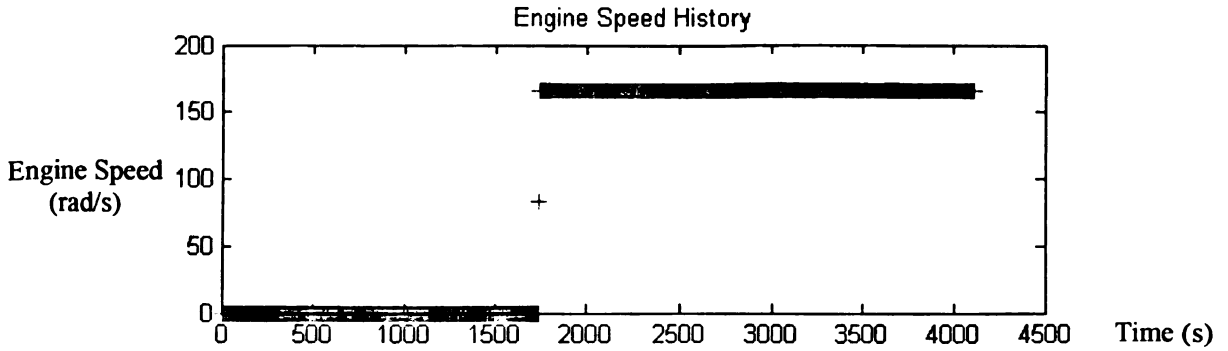


(b) Engine speed history with initial SOC=0.68.

Figure 5.6.4. Comparison of the SOC history and engine speed history plots for a typical optimized design from the Strategy 1 optimizations (Run 1 shown here). With an initial SOC of 0.68 these optimized designs can run primarily as electric vehicles (EV's).

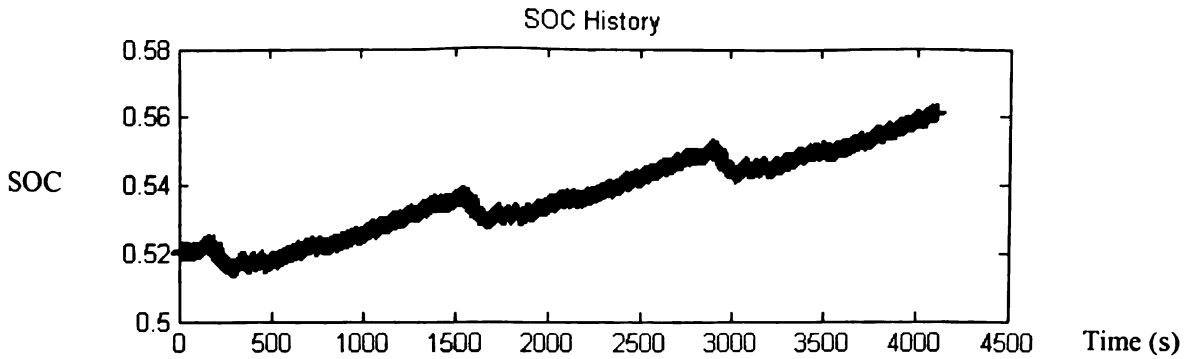


(a) SOC history with initial SOC = 0.60.

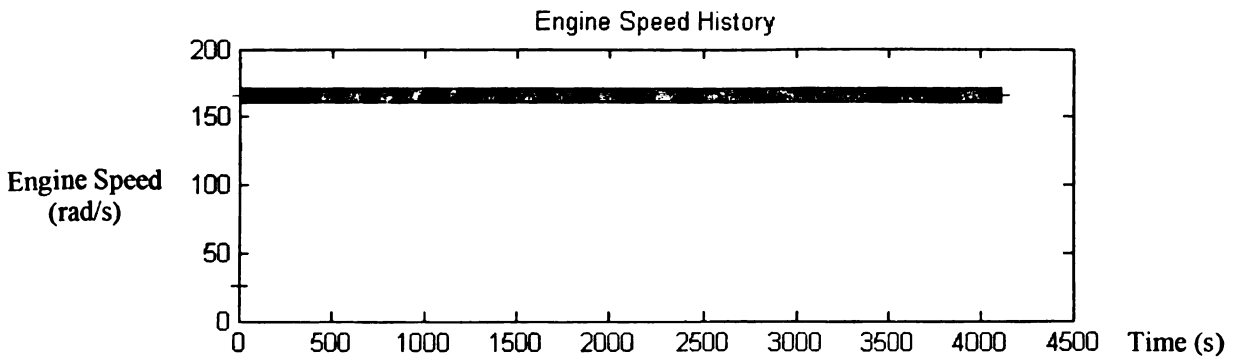


(b) Engine speed history with initial SOC=0.60.

Figure 5.6.5. Comparison of the SOC history and engine speed history plots for a typical optimized design from the Strategy 1 optimizations with initial SOC = 0.60 (Run 1 shown here).

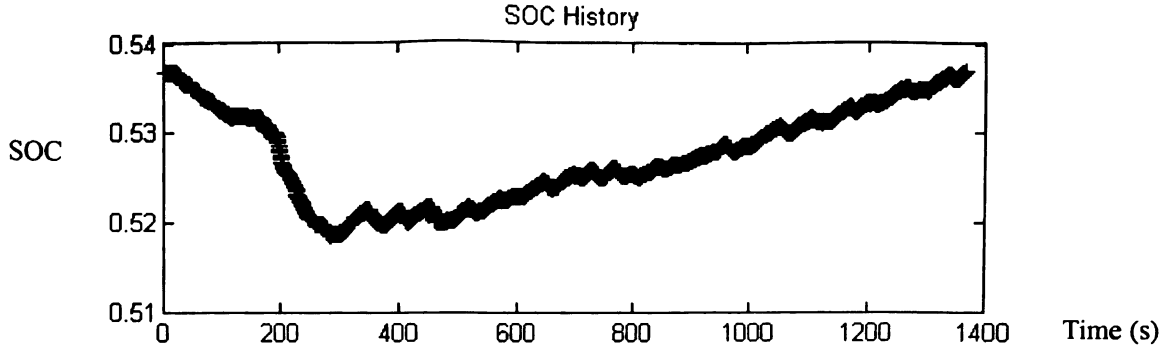


(a) SOC history with initial SOC = 0.52.

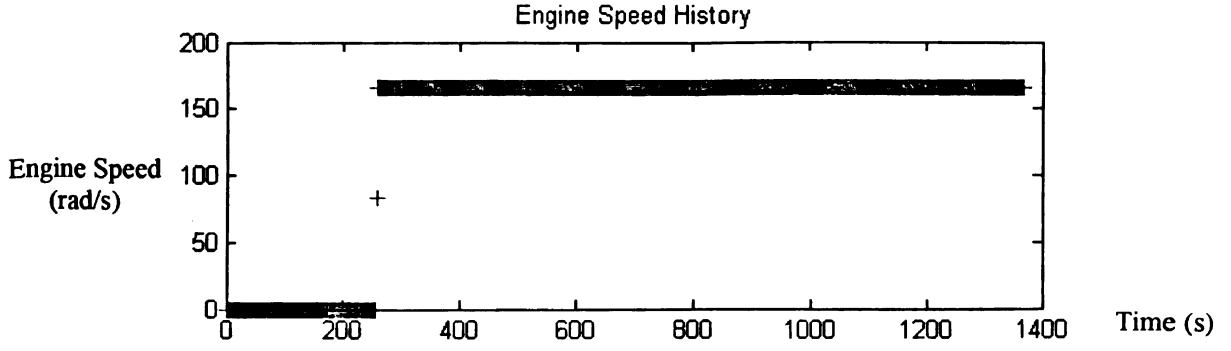


(b) Engine speed history with initial SOC=0.52.

Figure 5.6.6. Comparison of the SOC history and engine speed history plots for a typical optimized design from the Strategy 1 optimizations with initial SOC = 0.52 (Run 1 shown here).

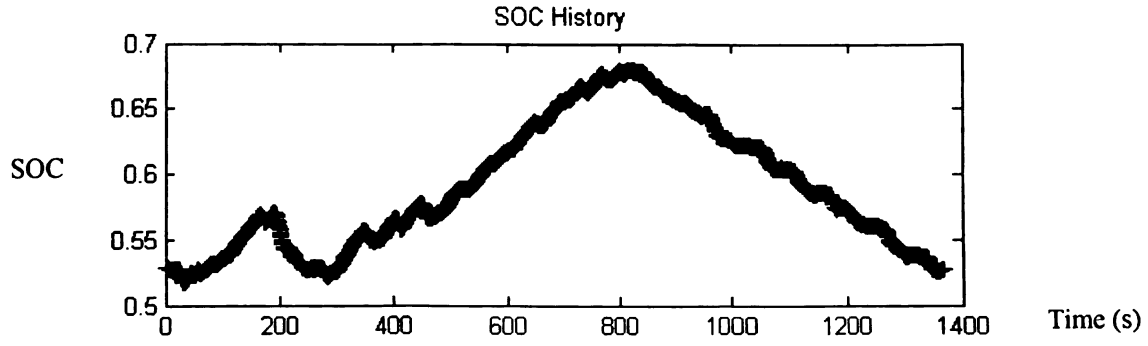


(a) SOC history.

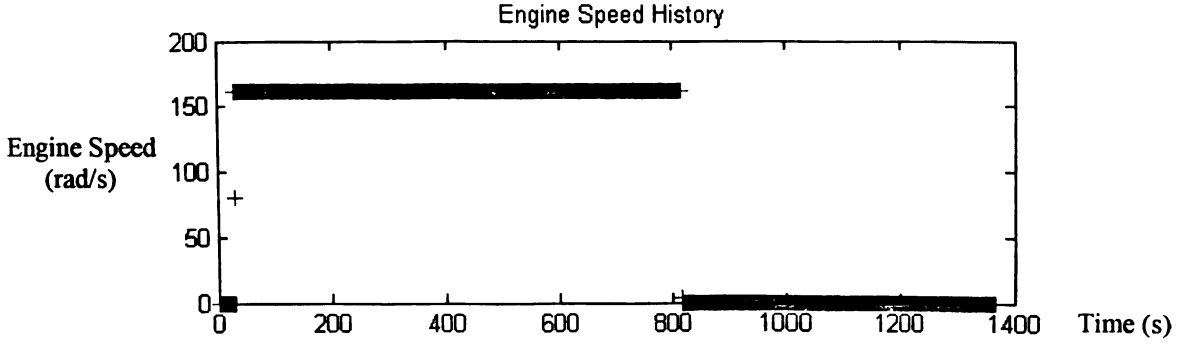


(b) Engine speed history.

Figure 5.6.7. Typical SOC history and engine speed history plots for the Strategy 1 optimized designs re-analyzed with SOC correction settings from Strategy 2 for a single drive cycle (Run 1 shown here). Note that due to the Zero-Delta SOC routine, the initial and final SOC's are very similar.

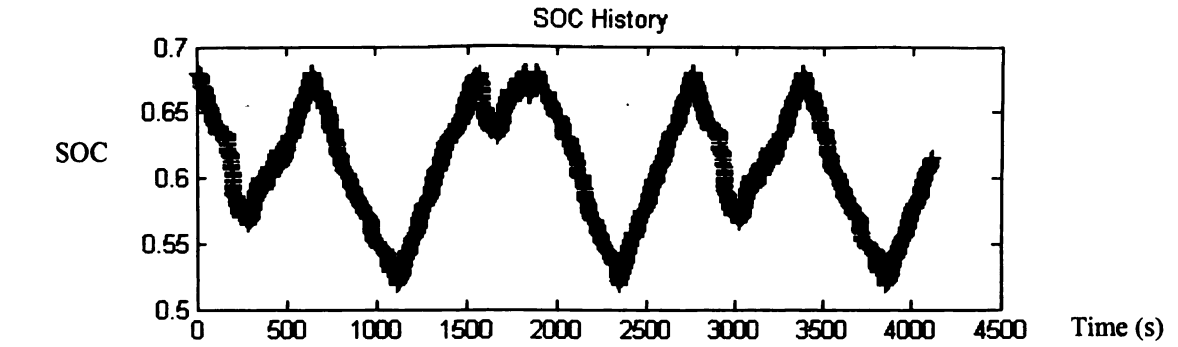


(a) SOC history.

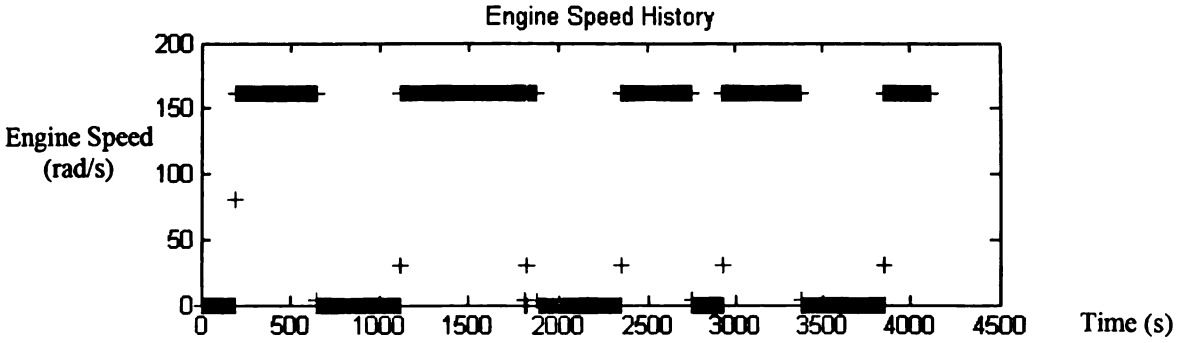


(b) Engine speed history.

Figure 5.6.8. SOC history and engine speed history plots for typical optimized designs where maximizing total energy usage is the objective (Run 1 from Strategy 2 is shown). Note that due to the Zero-Delta SOC routine, the initial and final SOC's are very similar.

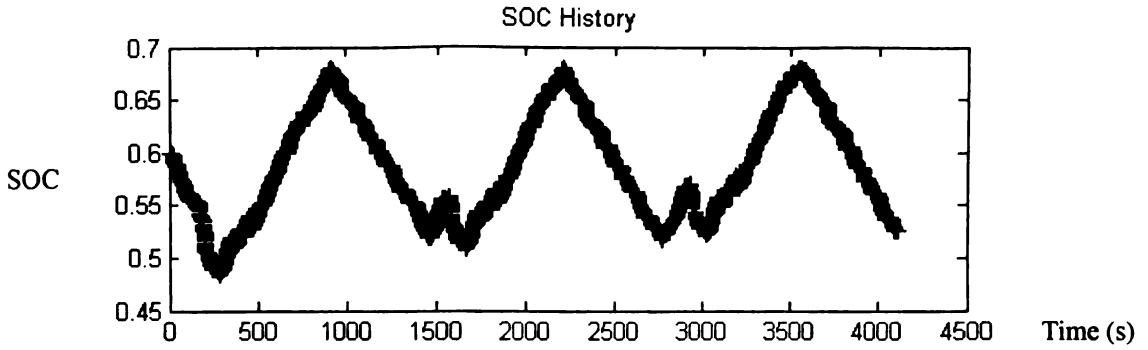


(a) SOC history with initial SOC = 0.68.

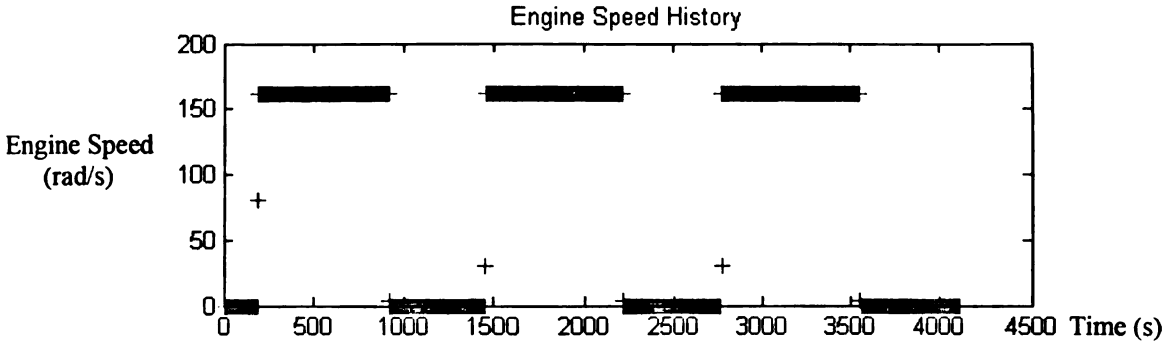


(b) Engine speed history with initial SOC=0.68.

Figure 5.6.9. Comparison of typical SOC history and engine speed history plots (SOC = 0.68) for the Strategy 2 optimized design of Figure 5.6.8 re-analyzed with no SOC correction and the settings of Strategy 1 (multiple drive cycle iterations and multiple initial SOC analyses). Note that the optimized designs do not act as EV's.

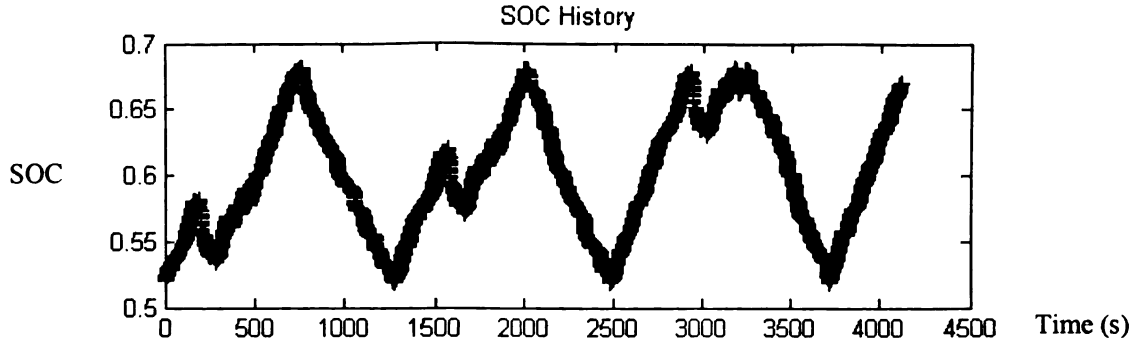


(a) SOC history with initial SOC = 0.60.

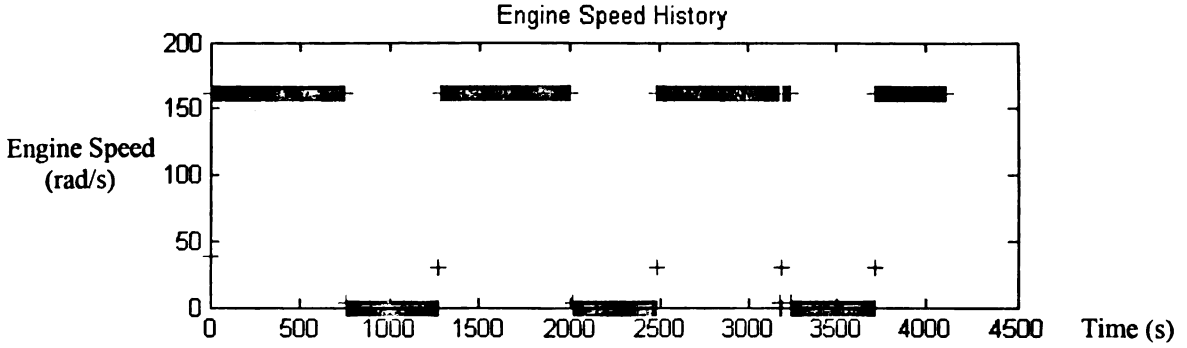


(b) Engine speed history with initial SOC=0.60.

Figure 5.6.10. Comparison of typical SOC history and engine speed history plots (SOC = 0.60) for the Strategy 2 optimized design of Figure 5.6.8 re-analyzed with no SOC correction and the settings of Strategy 1 (multiple drive cycle iterations and multiple initial SOC analyses).



(a) SOC history with initial SOC = 0.52.



(b) Engine speed history with initial SOC=0.52.

Figure 5.6.11. Comparison of typical SOC history and engine speed history plots (SOC = 0.52) for the Strategy 2 optimized design of Figure 5.6.8 re-analyzed with no SOC correction and the settings of Strategy 1 (multiple drive cycle iterations and multiple initial SOC analyses).

5.6.2 Baseline Hybrid Bus Comparisons

The optimized designs from Strategy 1 significantly outperformed the baseline non-hybrid bus in terms of engine fuel usage (measured in miles per gallon of gasoline), as shown in Table 5.6.3. However, when total energy usage is considered (measured in miles per gallon gasoline equivalence), the optimized designs from Strategy 1 are sub-par. Again, this is caused by the EV nature of the designs when they start with a high initial SOC. This demonstrates further the importance of this strategy being utilized only

when a fixed route is guaranteed and the starting initial SOC is known to be very high frequently.

Table 5.6.3. Comparison between the optimized designs from Strategy 1 and the baseline hybrid bus model.

	Average Fuel Economy (mpg)	Average Fuel Econ. Improvement over Baseline	Corrected Fuel Economy (mpgge)	Corrected Fuel Econ. Improvement over Baseline
Baseline Design	3.71	NA	4.07	NA
Optimized Design1	13.23	256 %	4.01	-1.5 %
Optimized Design 2	12.22	229 %	3.96	-2.7 %
Optimized Design 3	13.10	253 %	3.86	-5.1 %

The Strategy 2 optimized designs perform better than the baseline non-hybrid bus design in terms of total energy usage and engine fuel usage. With Strategy 2 (summarized in Table 5.6.4), all three optimized designs are a significant improvement in corrected fuel economy, while two of the three optimized designs are an improvement in the average fuel economy measure (with the third optimized design being equal).

Table 5.6.4. Comparison between the optimized designs from Strategy 2 and the baseline hybrid bus model.

	Average Fuel Economy (mpg)	Average Fuel Econ. Improvement over Baseline	Corrected Fuel Economy (mpgge)	Corrected Fuel Econ. Improvement over Baseline
Baseline Design	3.71	NA	4.07	NA
Optimized Design1	3.90	5.12 %	4.30	5.65 %
Optimized Design 2	4.03	8.68 %	4.33	6.38 %
Optimized Design 3	3.70	-0.27 %	4.30	5.65 %

As with the Strategy 2 optimized designs, the Strategy 3 optimized designs perform better than the baseline non-hybrid bus design in terms of total energy usage and engine fuel usage. With Strategy 3 (summarized in Table 5.6.5), all three optimized designs are a significant improvement in average corrected fuel economy, while marginally better in terms of average fuel economy.

Table 5.6.5. Comparison between the optimized designs from Strategy 3 and the baseline hybrid bus model.

	Average Fuel Economy (mpg)	**Average Fuel Econ. Improvement over Baseline**	**Avg. Corrected Fuel Economy** (mpgge)	**Corrected Fuel Econ. Improvement over Baseline**
Baseline Design	3.71	NA	4.07	NA
Optimized Design1	3.84	3.50 %	4.41	8.35 %
Optimized Design 2	4.04	1.09 %	4.30	5.65 %
Optimized Design 3	3.74	0.81 %	4.42	8.60 %

5.7

used

robu

and

for

gua

for

and

Stra

cor

Stra

tim

Stra

5.7 Conclusions

The main characteristics of two optimization strategies that have frequently been used in optimizing hybrid electric vehicles have been merged into a single and more robust strategy that is shown to have some advantages in terms of design performance and solution efficiency.

If reducing fuel usage is the primary goal, then Strategy 1was shown to work well for optimizing vehicles with a fixed route, and where a high initial SOC can be guaranteed a large portion of the operating time. This strategy found optimized designs for the current problem that run mostly electric in cases where the initial SOC is high.

If total energy usage is more important than fuel consumption, then Strategies 2 and 3 were shown to be superior to Strategy 1 for the problem posed in this study. Strategy 3 significantly outperformed Strategy 1 in terms of optimized designs with high corrected fuel economy and marginally outperformed Strategy 2 on this measure. Strategy 3 was also observed to have an added advantage over Strategy 2 in terms of run-time efficiency. Strategy 3 has been shown to be a viable alternative to the common Strategy 2 for optimization of hybrid electric vehicles.

CHAPTER 6: INTRODUCTION TO PROGRESSIVE CRUSH

6.1 Preliminary Information

Axial crush rails are one of the key energy absorbers during frontal and rear crash of automotive and other transportation vehicles. These components not only absorb a significant amount of kinetic energy, but also control the maximum forces that are transmitted to the remaining structure. When designing crush rails, a common goal is to identify the geometry, gage thickness and material of the rail so as to maximize its energy absorption while satisfying constraints on mass and the allowable axial force. Using such a statement within an automated design optimization study can lead to lightweight designs that absorb large amounts of energy. However, these designs may not necessarily absorb energy in a progressive manner, in which crush initiates near the tip of the rail and then progresses rearward in a controlled fashion. Progressive crush here is defined as crush occurring sequentially through a rail beginning at the front of the rail where impact occurs and propagating through to the rear of the rail. More formally, ideal progressive crush would occur when at any place where crush is occurring, there is no place to the rear of that location where crush has occurred. The propagation of progressive crush can be thought of as that of an accordion that is fixed at one end and pushed together from the opposite end. Figure 6.1.1 shows an example of progressive crush, while Figure 6.1.2 shows an example of non-progressive crush. Note that the rail in Figure 6.1.1 crushes

from front

exaggerate

Pro

A

robust cra

both in te

crash only

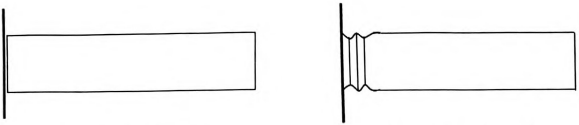
limited to

from front-to-back, whereas the rail in Figure 6.1.2 does not. While this is clearly exaggerated, it demonstrates the point.

Progressive crush is important for several reasons, but the primary ones are

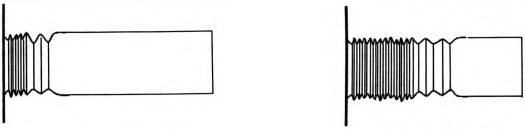
- to achieve a robust and repeatable crush mode, and
- to decrease repair costs after a minor crash event.

A rail design with a robust, repeatable crush mode is a key ingredient to achieving robust crashworthy behavior of the entire vehicle during frontal and rear crash events, both in terms of crash modality and total energy absorbed. In addition, during a minor crash only a small portion of the rail may experience crush. If this crushed region is limited to the tip of the rail, then repair costs can be minimized.



(a) Rail with no impact yet occurring.

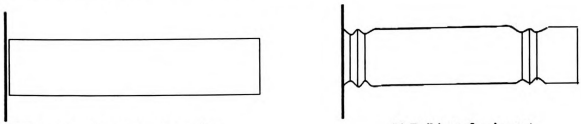
(b) Rail just after impact.



(c) Rail midway through crush

(d) Rail after crush.

Figure 6.1.1. Example of a rail undergoing progressive crush.



(a) Rail with no impact yet occurring.

(b) Rail just after impact.

(c) Rail midway through crush

(d) Rail after crush.

Figure 6.1.2. Example of a rail undergoing non-progressive crush.

6.2 Literature Review

DiPaolo et al. [31] describe three major response characteristics for axial energy absorbing devices. Depending upon such factors as plastic deformation, friction, and fracture, an energy absorbing device can either undergo axial crush (buckling), inversion, or splitting when undergoing axial compressive loading. For the purpose of this thesis, axial crush will be the primary focus.

Extensive research has been performed on the progressive buckling of columns under axial impact (see Jones [32] for a general overview of the subject). The two primary collapse modes in columns are global bending and progressive short-column buckling (folding) (Karagizova and Jones [33]). The latter mode absorbs significantly more energy than does collapse in bending, or global buckling. The introduction of a trigger (e.g., a local change in shape or material property) can improve the energy absorbing characteristics of a column by promoting the initiation of progressive buckling.

When progressive buckling occurs, the force-displacement curve typically follows a characteristic path (as shown by Guillow et al. [34] as well as Zarei and Kroger [35]) (see Figure 6.2.1). The axial load rises until a first buckle is formed, after which the force decreases significantly. Then folds begin to form, with a small peak in the force-displacement curve corresponding to each new fold. The values of the force required to initiate these folds are often significantly less than the maximum force observed at initiation of the first buckling event. Ideally, the formation of folds occurs sequentially from the front of the rail progressing toward the rear during progressive crush.

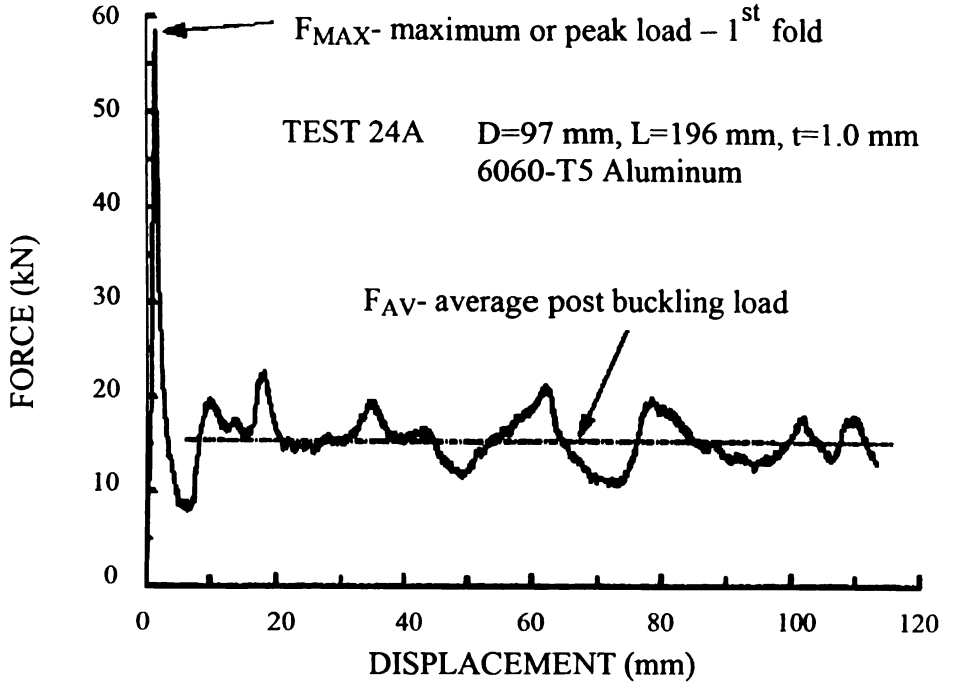


Figure 6.2.1. Quasi-static axial compression of thin-walled circular aluminum tubes (from Guillow et al. [34]).

Ku et al. [36] identify three key aspects in judging how well a column progressively crushes: geometric efficiency, load efficiency, and energy efficiency. Geometric efficiency is defined as the effectiveness of the deformation during the collapse (i.e., the ratio of compression to original length). Load efficiency is a measure of the load fluctuations between folds in the force-displacement curves. Energy efficiency is the total amount of energy absorbed.

Many formal definitions have been used to characterize axial crush (DiPaolo et. al [31]). The specific energy absorption is defined as the energy absorbed per unit mass of an axial crush. Similarly, the energy dissipation density is defined as the energy absorbed per unit volume of material. The structural effectiveness is sometimes used as a means to compare different materials' crushing capabilities and is defined as the specific energy

divided by the specific true ultimate tensile strength of the material. A load uniformity parameter is defined as the ratio between maximum and mean crushing forces seen during crush and indirectly gives an indication of the energy absorbing capabilities. In addition, there are common characteristics related to the collapse efficiency during crush. The crush efficiency is defined as the ratio between the maximum displacement and the total length. The load efficiency is defined as the inverse of the load uniformity parameter (or as the ratio of the mean load to the maximum load). Finally, the energy efficiency is defined as the energy absorbed divided by the product of the maximum load magnitude and the total length of the energy absorbing device.

Gumruk and Karadeniz [37] discuss a common characteristic to progressively buckling designs of top hat thin-walled sections. When comparing the rigid wall crush force profiles of progressively buckling designs, designs with mixed modes of deformation, and globally buckling designs, distinct differences can be seen. For progressively buckling designs, sharp peaks and valleys are present in the profile. The mixed modes of deformation designs have the same peaks and valleys present, however they are much smoother. Finally, global buckling designs have the size of the peaks decrease substantially.

Through optimization of hexagonal thin-walled columns undergoing axial crush, Hou et al. [38] show the effect of a constraint limiting the maximum peak force on the optimal design obtained during optimization. By performing identical optimizations of maximizing specific energy absorption with and without constraints imposed upon the maximum peak force seen, the authors were able to demonstrate the necessity of the constraint in crashworthy optimization of rails. Without the constraint, design variables

determining the thickness of the columns were seen to increase to the extremes of their allowable values, causing higher energy absorption, weight, specific energy absorption, and maximum peak forces. Most notably, the peak forces reached values considered detrimental to passenger safety and vehicle crashworthiness when the reaction force constraint was not imposed.

Kim [39] performed an optimization of multi-cell aluminium profiles for columns in which maximizing the specific energy absorption was the objective. The author defined a unique approach to force progressive crush in his designs by defining a constraint for the stable progressive collapse of square corner elements in his profile (based upon a theory for the stable deformation mode of a thin-walled prismatic square column presented by Abramowicz and Jones [40]). By analytically calculating whether the profile would progressively buckle based upon the thickness, length, and width of the corner elements in the profile using this theory, the author was able to enforce the desired collapse mode and eliminate designs as feasible that had a high specific energy but didn't progressively buckle. For simple column buckling this was shown to be an effective way to optimize for progressive buckling, however for complicated energy absorbing devices such as front rails where the geometry prevents a simple analytical solution as to whether progressive crush will occur, the approach is invalid.

The majority of the studies on crashworthy optimization have focused on total energy absorbed as opposed to progressive crushing. For example, Yamazaki and Han [41] performed tube optimizations in which they attempted to maximize the total energy absorbed while constraining the axial reaction force and mass of the tube. Zarei and Kroger [35] used a weighted multi-objective problem statement which attempted to

maximi

satisfyi

optimiz

inequal

problen

total en

optimi

optimi

constr

Kurtar

forces

absorp

al. [4

maxim

in the

force

attem

cautio

an ex

levels

uniform

[49] d

maximize the sum of the total energy and the specific energy absorbed by tubes while satisfying a constraint on the axial reaction force. Sidhu et al. [42], [43] used a crush optimization statement that maximized total energy absorption in a front rail with inequality constraints on the peak rigid wall force and mass. Cho et al. [44] used a similar problem statement for optimization of a frontal frame where the goal was to maximize total energy absorption with constraints on reaction force and volume. A crashworthiness optimization on an S-frame was performed by Kim et al. [45] in which they utilized an optimization problem statement that attempted to minimize the mass of the frame while constraining not only the energy absorption, but also the sectional bending stiffness. Kurtaran et al. [46] presented an optimization statement that attempted to minimize the forces in the occupant compartment (or the intrusion), while constraining the energy absorption.

A slightly different crush optimization problem statement was used by Avalle et al. [47]. They used the load uniformity parameter (LU) defined as the ratio between the maximum and mean crushing forces. This load uniformity parameter was then minimized in the optimization problem statement. By encouraging the design to have an average force as close as possible to the maximum force, it was assumed the optimization was attempting to maximize the total energy absorbed. Karbhari and Haller [48] however, caution that optimizing based upon the load uniformity parameter can give designs with an extremely good LU at very low overall load levels, and hence at low corresponding levels of energy absorption. This needs to be considered when determining if the load uniformity parameter is alone appropriate as an objective for optimization. Hanssen et al. [49] define a parameter called the total efficiency as the product of the load efficiency

(which they call the crush force efficiency) and the relative deformation. This characteristic accounts for the concern presented by Karbhari and Haller and could be used in optimization as an alternative to the load uniformity parameter.

None of these common optimization approaches guarantees progressive buckling of complex structures such as front rails because the buckling failures could initiate anywhere in the rail and progress in a non-sequential manner.

CHAPTER 7: OPTIMIZATION STRATEGY DEFINITION

7.1 Introduction

A new strategy for achieving progressively crushing designs during an automated design optimization study is presented and explored in this chapter. Independent crush zones are defined along the length of the rail. The main goal of the optimization is then to encourage maximum energy absorption in a given crush zone to occur prior to any energy absorption in rearward crush zones. In other words, crush should initiate in the tip and progress from zone to adjacent zone toward the rear of the rail. Compared to traditional optimization statements that focus only on total energy, it is demonstrated that the current approach can produce designs with similar total energy absorption but with a dominant progressive mode.

7.2 Formal Definition

Traditional crush optimization strategies produce designs that absorb a lot of energy, but not necessarily progressive crush. The new strategy considered here attempts to find optimal designs that not only absorb a lot of energy, but also exhibit progressive crush behavior. The strategy utilizes a measure of the amount and timing of energy absorbed in different regions of the rail to promote progressive crush.

Figure 7.2.1 displays a rail profile in which different crush zones have been defined. In an ideal progressive crush scenario, zone N-1 will crush almost entirely before zone N begins crushing (for $N > 1$).

We define a measure called progressive energy absorbed (PEA) as:

$$PEA = \sum_{N=1}^{NZ-1} PEA_N - EA_{NZ} \tag{7.1}$$

where PEA_N is the progressive energy absorbed in Zone N, NZ is the total number of zones defined in the rail, and EA_{NZ} is the total energy absorbed by the last zone in the rail. We define the zone-wise progressive energy absorbed, PEA_N, as:

$$PEA_N = EA_N(U_N) - EA_N(U_{N-1}) \tag{7.2}$$

where $EA_N(U_N)$ is the total energy absorbed so far by Zone N when the impactor has displaced an amount $U = U_N$, $EA_N(U_{N-1})$ is the total energy absorbed so far by Zone N when the impactor has displaced an amount $U = U_{N-1}$. U is the distance the impactor has traveled (or in practical terms how far a reference point has moved) after impact (see Figure 7.2.2 and Section 7.3 for more details).

We assume here that Zone NZ should not crush at all, so the role of the final term in Equation (7.1) is to penalize PEA for energy absorption that occurs in Zone NZ. In system level crash models, often there is a zone such as NZ that must maintain its integrity during the crash. Further, in component level models such as considered here, it is common to constrain the rear of the rail at a location beyond which crush is expected or desired. So crush in Zone NZ is undesirable in both scenarios.

The progressive energy as described above does not consider any energy absorbed in Zone N, after the impactor has reached a displacement U_N. By definition, any energy absorbed in Zone N before the impactor has reached a displacement U_{N-1} is also not considered. Therefore, the impactor displacement range for which the energy absorbed by Zone N contributes to the progressive energy absorbed is:

$$U_{N-1} \leq U \leq U_N \tag{7.3}$$

A rail design that absorbs significant energy in each zone N (for N = 1 to NZ-1) while the impactor displacement corresponds to that of Equation (7.3) has a much higher PEA than a rail design that absorbs the same amount of total energy that is not in conjunction with the impactor displacement as described in Equation (7.3). The former design would, by definition, have a crush mode that originates at the front of the rail and propogates rearward. The latter design could allow energy absorption to occur in a non-sequential manner, and therefore could have a non-progressive crush mode.

The optimization problem statement for progressive crush optimization can now be stated as:

Maximize: *PEA*

Subject to: Constraints on mass, force, etc.

This is contrary to typical crash optimization statements that do not encourage progressive crush:

Maximize: *EA*

Subject to: Constraints on mass, force, etc.

where EA is the total energy absorbed by the rail during crush. In the latter case, the optimal solution may find a design that crushes progressively or non-progressively.

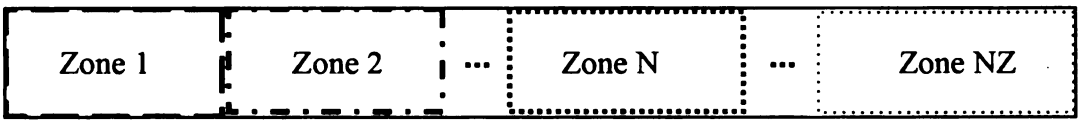


Figure 7.2.1. Specified crush zones in a rail. The tip of the rail corresponds to Zone 1, and the rear of the rail corresponds to Zone NZ.

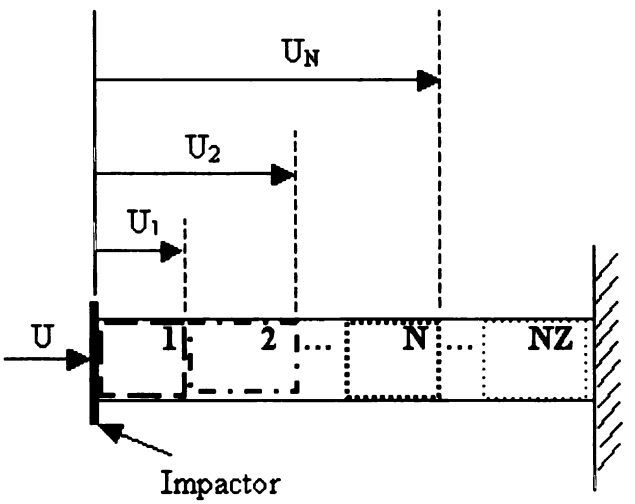


Figure 7.2.2. Definition of impactor displacements, as used in the calculation of PEA.

7.3 Implementation of Strategy

As was mentioned briefly in the previous section, the impactor displacements are measured utilizing a reference point (or node) located on the impactor. As the impactor moves, the relative displacement of the reference point can be measured. Since the analyses performed in this thesis were performed with LS-DYNA [50], the displacement of the node or reference point was measured from the *nodout* file.

The zones were achieved in LS-DYNA by making each zone a separate part. With different parts making up the rail, the internal energy of each part was written out independently to the *matsum* file by LS-DYNA at different times.

As defined by Equation (7.2) the primary interest for the progressive energy definition is the energy absorption of a specific zone (or part) while the reference point is at a specific location. However, the displacements of the reference point are reported in the *nodout* file and the energy absorptions of the parts in the *matsum* file. Both are linked however by the time at which they occur (the *nodout* file reports the displacements for a given time and the *matsum* file reports the energy for a given time). A simple executable therefore was created that would report the energy absorption for a given part (zone) given a requested displacement of the reference node. The executable takes the requested displacement, finds in the *nodout* file the time at which it occurs, and then obtains from the *matsum* file the energy absorption for the part at that time (see Figure 7.3.1). Utilizing this simple executable, the energy absorbed in zone N at an impactor displacement U_{N-1} could be obtained along with the energy in zone N at an impactor displacement U_N, such that Equation (7.2) could be used to calculate the *PEA* of zone N.

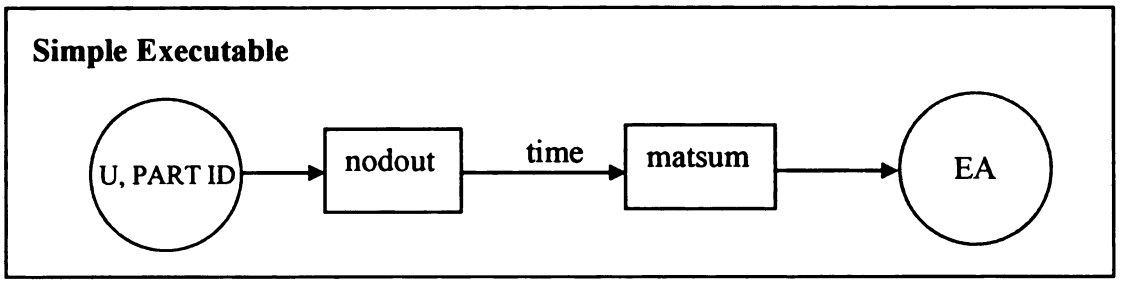


Figure 7.3.1. Simple executable used with LS-DYNA to obtain EA(U).

CHAPTER 8: AXIAL RAIL CRUSH AND ANALYSIS MODEL

In this thesis, the following crush scenario and corresponding model were used to assess the proposed strategy for progressive crush optimization. A tube (rail) of length 500 mm was fixed at its right (rear) end. The tube was made of steel with a density of 7.85e-6 kg/mm^3, a Young's Modulus of 207 GPa, Poisson's ratio of 0.28, yield strength of 500Mpa, and the plasticity data of Table 8.1.1. The finite element model for the tube consisted of 7479 nodes and 7391 shell elements (Belytschko-Tsay with 2 through shell thickness integration points [50]). A flat rigid impactor with 500 kg mass travelling initially at 50 kph was used to crush the tube for 25 ms, as shown in Figure 8.1.1.

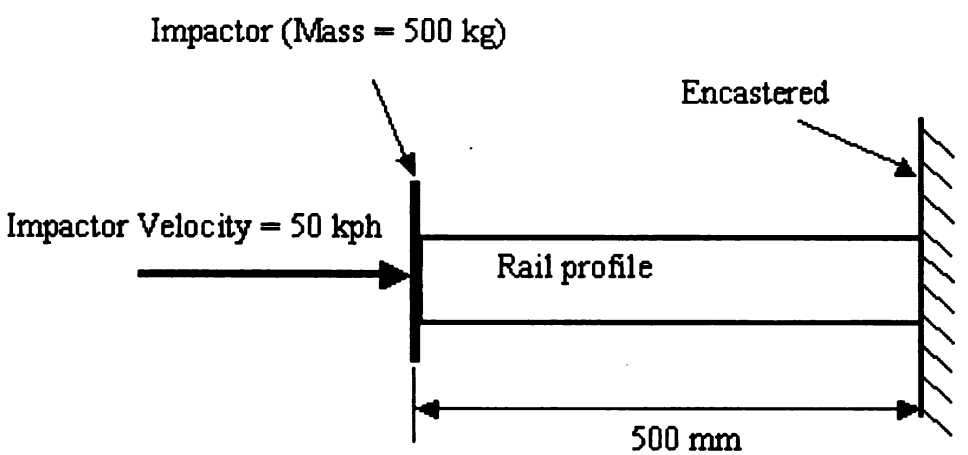


Figure 8.1.1. Analysis model used for the progressive crush strategy.

Table 8.1.1. Plasticity data for the rail analyses. For different strain rates, the stress vs. effective plastic strain is defined.

Strain Rate (mm/ms)									
0.005		0.1		10		100		500	
Effective Plastic Strain	**Stress (MPa)**	**Effective Plastic Strain**	**Stress (MPa)**	**Effective Plastic Strain**	**Stress (MPa)**	**Effective Plastic Strain**	**Stress (MPa)**	**Effective Plastic Strain**	**Stress (MPa)**
0	500	0	500	0	500	0	500	0	500
0.0004	541.65	0.0004	545	0.0005	570.16	0.001	672.62	0.0016	823.46
0.0018	591.96	0.0013	586.24	0.0016	600.13	0.0021	712.03	0.0027	853.28
0.0022	598.91	0.0035	630	0.003	630.85	0.0093	778.56	0.0047	878.03
0.0044	622.32	0.0065	665	0.0056	668.82	0.0122	797.12	0.01	910.59
0.0079	651	0.0086	690	0.0075	689.51	0.0152	817.83	0.0232	934.62
0.0115	678.29	0.011	715	0.0096	712.58	0.0236	853.36	0.0305	945
0.0147	700.93	0.0134	730	0.0118	738.19	0.0284	871	0.0496	967.48
0.0194	730.46	0.0162	750	0.0143	758.35	0.0379	902.44	0.0567	979.02
0.0242	757.27	0.0214	790	0.019	792.49	0.045	920.14	0.066	990.73
0.0297	784.09	0.0286	815	0.0285	838.85	0.0568	948	0.0753	1003.73
0.0349	806.15	0.0334	833.71	0.0357	867.81	0.0661	967.12	0.0845	1015.92
0.0422	831.51	0.0429	859.62	0.0451	899.89	0.0777	987.85	0.0936	1028.2
0.0475	846.92	0.0477	871.39	0.0569	933	0.0846	998.21	0.1004	1039.41
0.054	862.44	0.0547	887.7	0.0616	941.76	0.1027	1020.37	0.1093	1051.58
0.0601	874.27	0.0617	902.55	0.0685	955.43	0.1072	1026.19	0.1182	1063.55
0.0659	883.51	0.0687	916.09	0.0778	970.7	0.1139	1030.59	0.127	1075.19
0.0722	891.57	0.0871	946.66	0.0892	986.21	0.125	1041.74	0.1357	1084.39
0.0788	898.41	0.0962	959.29	0.096	994.1	0.1293	1044.67	0.1422	1092.42
0.0849	903.66	0.1007	965	0.1073	1005.54	0.1359	1050.66	0.1508	1102
0.0941	910.37	0.1052	970.32	0.1118	1009.62	0.1489	1058.86	0.1594	1110.85
0.0976	912.71	0.1186	983.88	0.1162	1013.46	0.1553	1064.84	0.1678	1116.24
0.1046	917.3	0.1252	989.25	0.1294	1023.43	0.1596	1066.64	0.1763	1119.57
0.111	921.62	0.1318	993.6	0.1338	1026.15	0.1638	1068.96	0.1825	1124.27
0.1177	926.49	0.1383	996.82	0.1404	1029.53	0.1681	1070.74	0.1908	1129.38
0.1234	930.99	0.1427	998.28	0.1447	1031.2	0.1723	1075.92	0.199	1133.03
0.1303	937.06	0.1492	999.32	0.1512	1032.62	0.2072	1090.8	0.2072	1136.7
0.2072	969	0.2072	1008.6	0.2072	1044.9				

At each iteration during the optimization study, the shape of the rail was varied according to the following scheme. The cross-sectional shapes at several stations along the length of the profile (see Figure 8.1.2) were designed independently, thereby allowing the shape of the profile to vary along the length. Nine independent stations were used along the length of the rail. Each cross-sectional shape was governed by 12 control points (only 4 of which were independent due to symmetry). Control points that could move in the horizontal direction (in Figure 8.1.3) were allowed to vary within 30% of the baseline

coordinate. Control points that could move in the vertical direction were allowed to vary within 60% of the baseline coordinate. For manufacturability, the total perimeter of all cross-sections was constrained to vary only within 2% of the baseline perimeter. This was controlled by scaling of the shape produced for a given cross-section until the perimeter satisfied the perimeter constraint. Symmetry about vertical and horizontal planes was maintained in each cross-section.

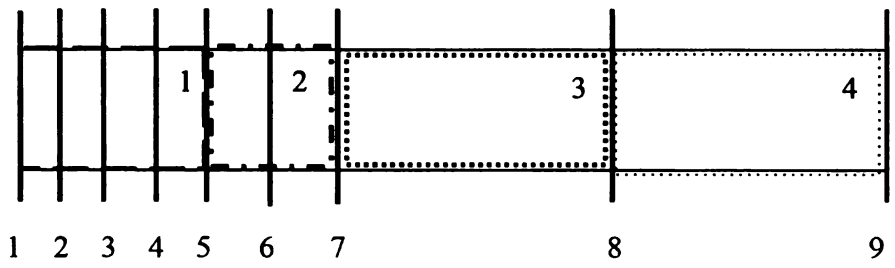


Figure 8.1.2. Crush zones and cross-sectional design stations used in the rail analysis and design model.

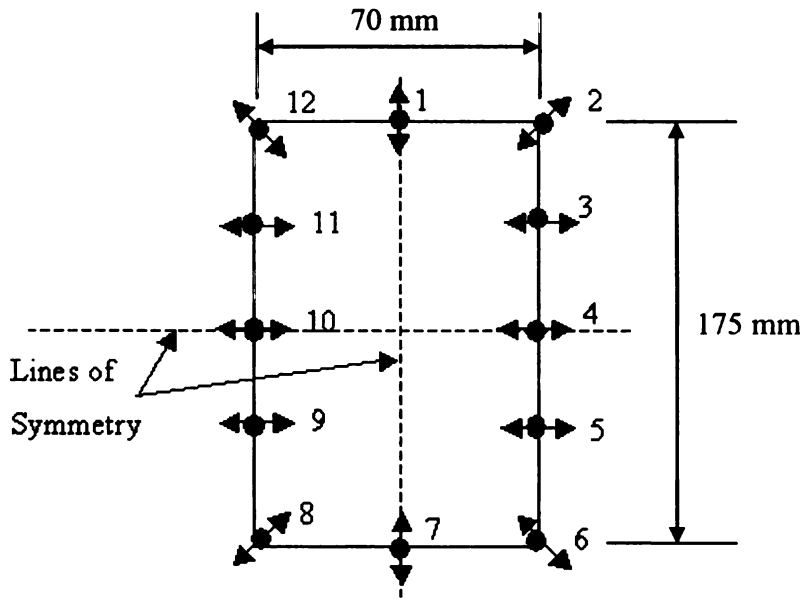


Figure 8.1.3. Section profiles were controlled by 4 independent control points (1 through 4). The other control points were positioned by satisfying vertical and horizontal lines of symmetry. The cross-section perimeter was scaled to ensure that it was within 2% of the baseline perimeter.

The number of stations along the length of the rail was chosen largely based upon intuition and experience. Since the goal is to design a rail that progressively crushes, it is desirable to create more shape variation at the front of the rail, and less towards the rear of the rail. This helps with crush initiation, as well as creating a crush mode that propagates rearward. At the same time, it has been observed that too much shape variation overcomplicates the design and causes complexity in manufacturing as well as inferior performing designs.

With nine cross sections and five degrees of freedom per cross-section, there were 45 shape design variables for this problem. In addition, the gage thickness was allowed to vary, so there was a total of 46 design variables.

Figure 8.1.4 displays the zones and impactor displacements used for the majority of the studies in this thesis (unless noted otherwise). The selection of the location of the zones was chosen largely based on intuition. In order to emphasize the initial energy absorption that occurs in the front of the rail, the first two zones were made smaller than the last two zones which are toward the back of the rail. Attachment locations as well as stipulations on the size of Zone NZ also should guide the selection of zone locations and lengths. Note in Figure 8.1.4 that the definition of U_3 used in the study does not correspond with the end of Zone 3. This modification was necessary to account for stack up in the crushed rail (as will be proven in Section 4 of Chapter 9).

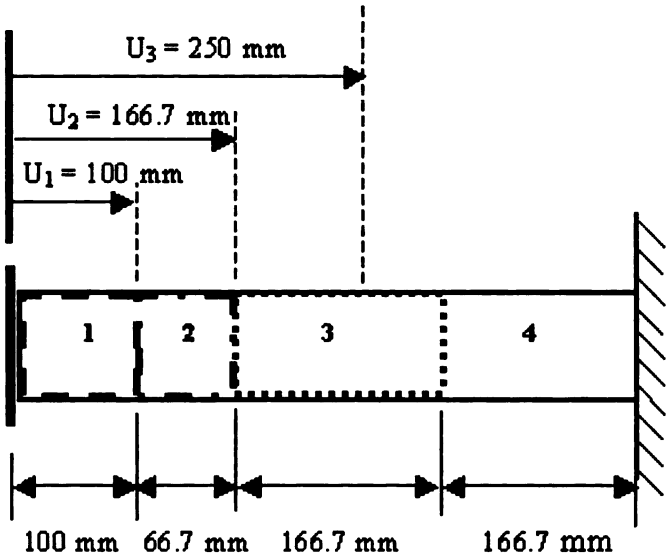


Figure 8.1.4. Zone dimensions and impactor displacement specifications in the current model.

In rail design, it is commonly required to impose a constraint on the maximum wall reaction force. A typical crush event is shown in Figure 8.1.5. In the current study, the first reaction force peak is ignored when calculating the maximum force, as this force is often artificially high due to numerical inaccuracies. Instead, the highest force after F_1 occurs, referred to as F_2, was used as a constraint for optimization.

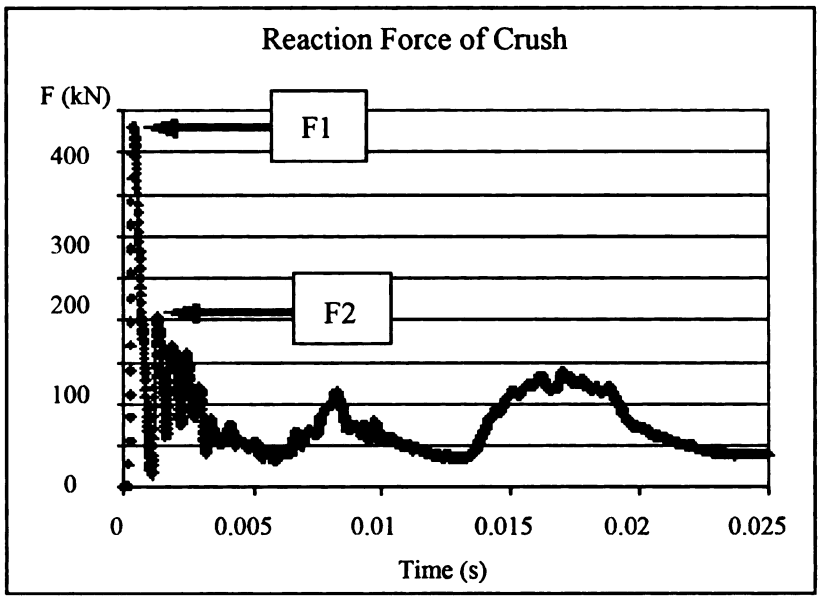


Figure 8.1.5. Reaction force versus time during a typical crush event.

CHAPTER 9: OPTIMIZATION USING THE PROGRESSIVE CRUSH STRATEGY

9.1. Relation Between Progressive Energy and Total Energy

The advantages of progressive crush would be diluted if the associated total energy absorbed was less than that for a non-progressively crushing rail. To investigate this possible trade-off, a multi-objective optimization study was performed in which both PEA and EA were maximized for the rail model described in Chapter 8. Multi-objective optimization studies generate multiple optimal solutions representative of the tradeoffs among the objectives. This is commonly called Pareto optimization. In this study, the multi-objective crush optimization problem statement used was:

Maximize: *PEA* and *EA*

Subject to: $F_2 \leq 150\ kN$

By Varying: *Control Points*

Rail Thickness

Where *PEA* is the progressive energy absorbed, *EA* is the total energy absorbed, F_2 is the maximum reaction force after initial crush initiation, *Control Points* refers to the position of the 45 independent shape control points, and *Rail Thickness* is the rail gage thickness, which varies between 1-3 mm.

The simulations were performed using the LS-DYNA explicit finite element code [50] as the analysis solver, and the hybrid and adaptive algorithm SHERPA within the HEEDS optimization software [29] as the optimizer. The study was performed for 15,000 evaluations, at which stage the results were considered nearly converged. The study was then repeated to obtain very nearly the same results. Figures 9.1.1 and 9.1.2 illustrate the resulting feasible designs from this study along with the rank 1 (non-dominated) designs on the Pareto front. In Figure 9.1.1, it is apparent that many designs with very high total energy absorption have very poor progressive energy absorption. In some instances, the progressive energy absorption measure is negative. These designs absorb a significant amount of total energy, but do so with a very non-progressive crush mode. Contrarily, there are no designs with high progressive energy and low total energy. Figure 9.1.2 focuses on the local range of best designs found, and better shows the rank 1 designs of the Pareto front. Within the Pareto front identified here, the EA varies by only 3%, and this variation appears to be decreasing (note the apparent convergence toward a single point for maximum PEA and EA in Figure 9.1.1) as the search progresses. These results indicate that the tradeoff between PEA and EA is very small, if not negligible, for the rail considered herein.

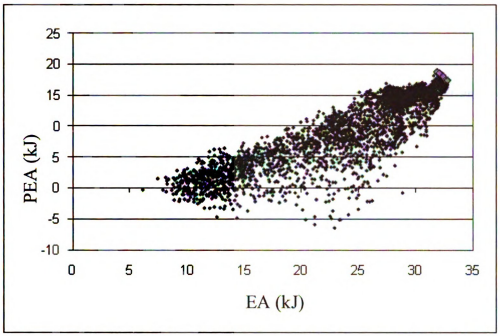


Figure 9.1.1. Scatter plot of all evaluations performed during the multi-objective optimization. There are numerous designs with total energy above 15 kJ but with negative progressive energies. Furthermore, there are designs at 25 kJ and above that have negative or near-zero progressive energies.

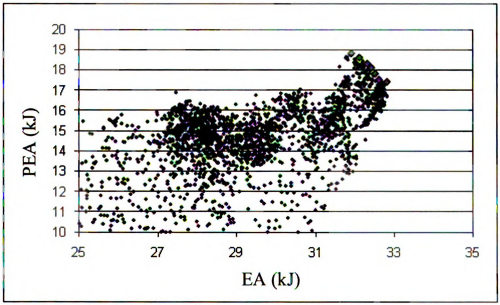


Figure 9.1.2. Scatter plot of good designs in terms of both total energy absorbed and progressive energy absorbed. Highlighted is the Pareto front of the rank-1 designs from the multi-objective optimization.

9.2 Progressive Crush Optimization - without considering mass

Two optimization studies were performed – one with a conventional strategy which maximized total energy absorbed, and one with the proposed new strategy which maximized progressive energy absorbed. In both strategies, the impact of mass on the design was ignored. The results of both studies are presented here to assess the ability of each to produce designs that absorb energy progressively.

For each of the two studies, four optimization runs were performed. Each optimization run started with the same baseline design with the rail having a constant rectangular cross-sectional shape along its length, as shown in Figure 8.1.3. Each of the four optimizations however, started with a different random seed that controlled the initial set of designs for the optimization. With different random seeds, the four optimizations started their search in different locations in the design landscape of the problem. This was done to account for the effects of stochasticity and initial starting conditions on the search of both strategies. The same four random seeds were used for both strategies.

In each run, 2,000 crash evaluations were performed using the LS-DYNA explicit finite element code. The hybrid and adaptive algorithm SHERPA within the HEEDS optimization software was used for the optimization runs. At 2,000 evaluations, the optimization runs were not fully converged, but they had made significant progress toward identifying high performing designs, and their rate of improvement had begun to decrease significantly. Since in most industrial applications of optimization it is not feasible to achieve complete convergence due to time constraints, it was deemed more

valuable here to compare the results of four nearly converged designs than one fully converged design.

9.2.1 Conventional Crush Optimization to Maximize Total Energy Absorbed

The following conventional crush optimization problem definition was used:

Maximize: EA

Subject to: $F_2 \leq 150\ kN$

By Varying: *Control Points*

Rail Thickness

Where EA is the total energy absorbed by the rail during crush, F_2 is the maximum reaction force after initial crush initiation, *Control Points* are the independent control points varied in Figure 8.1.3, and *Rail Thickness* is the rail gage thickness, which varies between 1 mm and 3mm.

Figure 9.2.1 (a) shows the shapes of the optimized designs found using this design statement, while Figure 9.2.2 (a) shows their force-displacement curves. The performance of these designs is summarized in Table 9.2.1.

9.2.2 Progressive Crush Optimization to Maximize Progressive Energy Absorbed

The progressive crush optimization problem statement used was:

Maximize: *PEA*

Subject to: $F_2 \leq 150\ kN$

By Varying: *Control Points*

Rail Thickness

Where *PEA* is the progressive energy absorbed, as defined in Equations (7.1,7.2), F_2 is the maximum reaction force after initial crush initiation, *Control Points* are the independent control points varied, and *Rail Thickness* is the rail gage thickness, which varies between 1-3 mm.

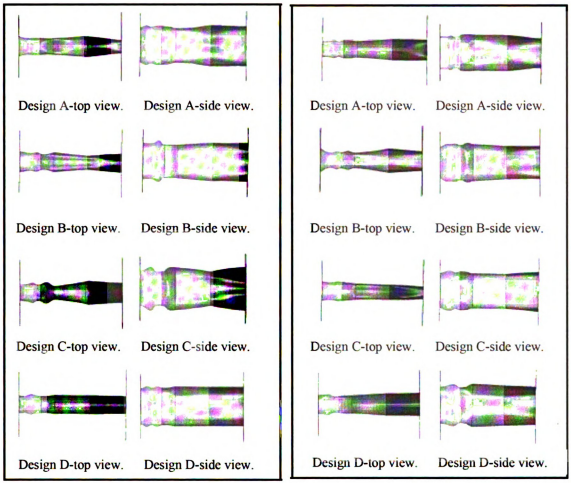
Figure 9.2.1 (b) shows the shapes of the optimized designs found using this design statement, while Figure 9.2.2 (b) shows their force-displacement curves. The performance of these designs is summarized in Table 9.2.1.

9.2.3 Optimization Results

Based on the optimization results, the following conclusions can be reached. Three out of four runs using the conventional crush design optimization statement yielded optimized solutions that did not exhibit progressive crush behavior, while one of the designs (design D) exhibited very good progressive crush. All of the optimized designs resulting from the progressive crush optimization statement performed very well in terms

of progressive crush. Figures 9.2.3-9.2.6 depict the crushing modes of these designs and support these conclusions.

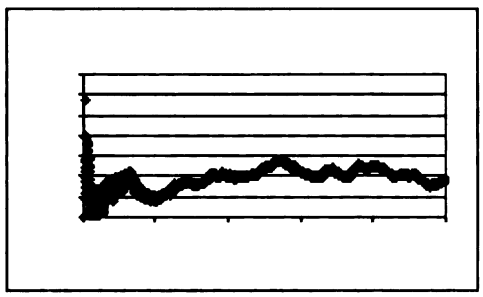
On average, the progressive crush strategy yielded designs with over 70% more PEA and 3.6% less total energy than did the conventional crush strategy. Perhaps more importantly, the standard deviation of EA and PEA for designs based on the progressive crush strategy was significantly lower than that for designs based on the conventional strategy. This indicates that the progressive crush strategy was much more robust in terms of producing designs that behaved as intended.



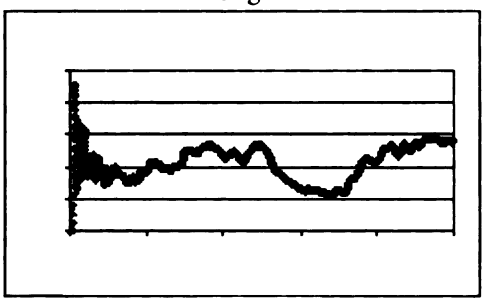
(a) Optimized design shapes from the conventional progressive crush strategy (maximizing EA).

(b) Optimized design shapes from the crush strategy (maximizing PEA).

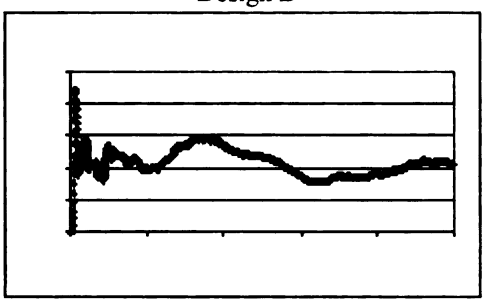
Figure 9.2.1. Comparison of the optimized design shapes from the conventional crush optimization and progressive crush optimization. After 2000 evaluations the optimization runs were not fully converged but had made significant progress.



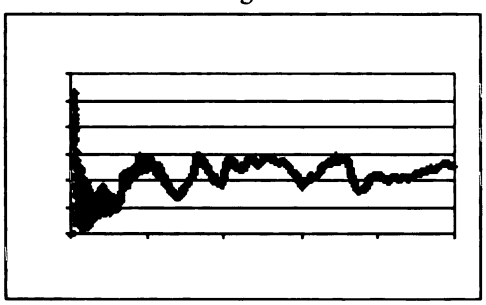
Design A



Design B

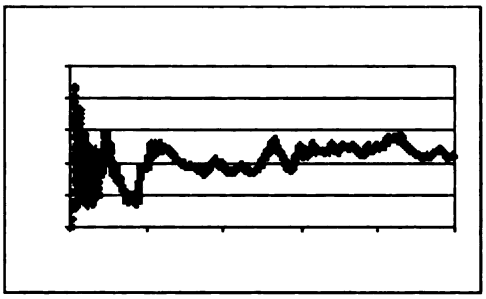


Design C

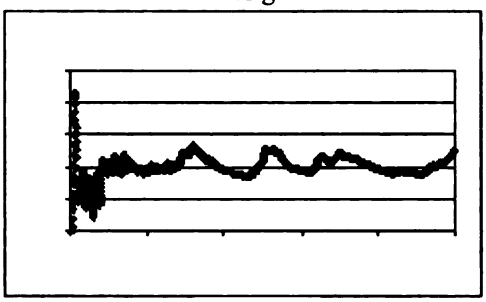


Design D

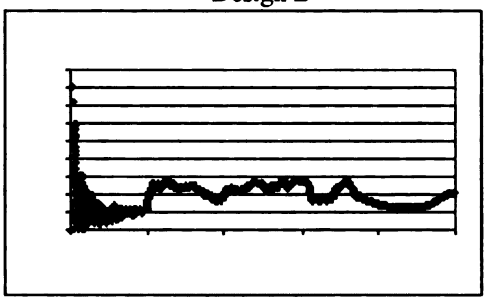
(a) Force-displacement curves of the optimized designs from the conventional crush strategy (maximizing EA).



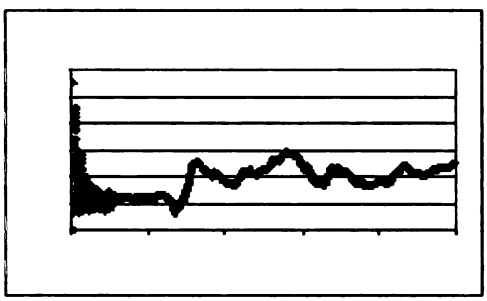
Design A



Design B



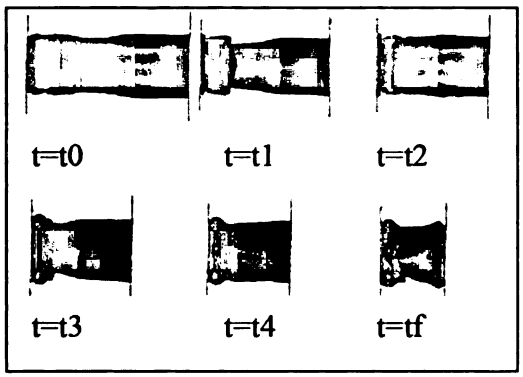
Design C



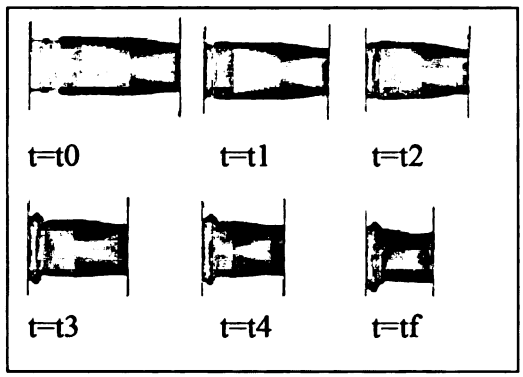
Design D

(b) Force-displacement curves of the optimized designs from the progressive crush strategy (maximizing PEA).

Figure 9.2.2. Comparison of the force-displacement curves for the optimized shapes from the conventional crush optimization (maximize EA) and the progressive crush optimization (maximize PEA).

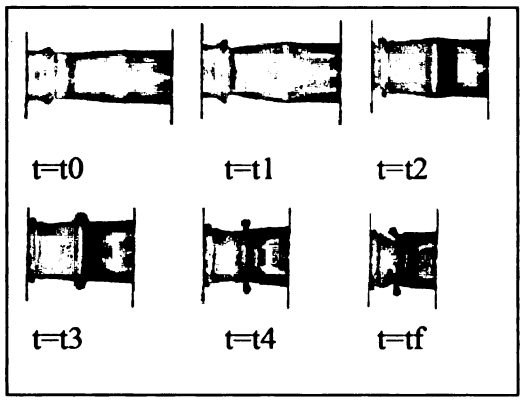


(a) Conventional strategy crush mode.

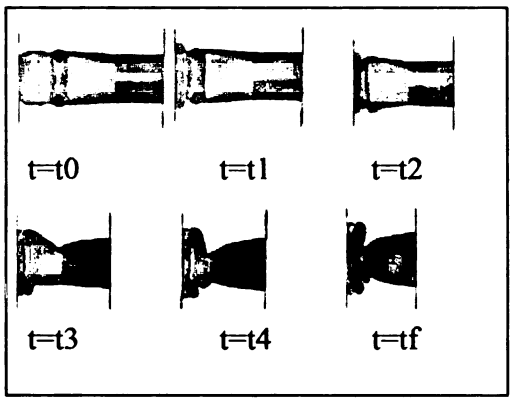


(b) Progressive strategy crush mode.

Figure 9.2.3. Crush modes of the Design A optimized designs. At t = t2, the conventional strategy's optimized design begins to non-progressively crush.

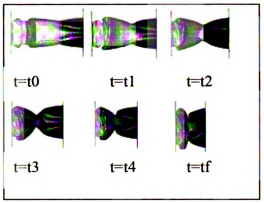


(a) Conventional strategy crush mode.

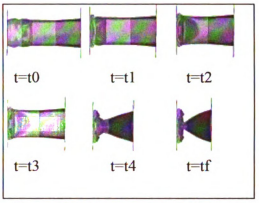


(b) Progressive strategy crush mode.

Figure 9.2.4. Crush modes of the Design B optimized designs. At t = t2, the conventional strategy's optimized design begins to non-progressively crush.

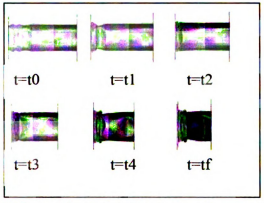


(a) Conventional strategy crush mode.

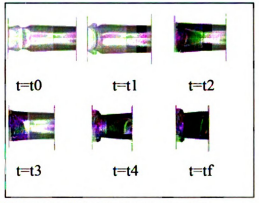


(b) Progressive strategy crush mode.

Figure 9.2.5. Crush modes of the Design C optimized designs. At t = t2, the conventional strategy's optimized design begins to non-progressively crush.



(a) Conventional strategy crush mode.



(b) Progressive strategy crush mode.

Figure 9.2.6. Crush modes of the Design D optimized designs. Both the conventional strategy and progressive strategy yielded optimized designs that progressively crushed.

Table 9.2.1. Comparison of progressive crush and conventional crush optimized results.

	Progressive Crush Strategy		Conventional Crush Strategy		Improvement of Progressive Crush Over Conventional	
Optimized Design	EA (kJ)	PEA (kJ)	EA (kJ)	PEA (kJ)	EA	PEA
A	29.21	17.73	26.19	10.28	11.53 %	72.97 %
B	28.12	16.25	28.92	5.96	-2.76 %	172.65 %
C	26.79	15.79	29.60	5.66	-9.5 %	178.98 %
D	26.33	15.25	29.87	16.29	-11.85 %	-6.38 %
Average	27.61	16.25	28.64	9.55	-3.60%	70.39 %
St Dev	1.31	1.06	1.68	4.96	NA	NA

9.3 Progressive Crush Optimization- accounting for mass

Many times in industry, it is desirable to have a design that absorbs as much energy as possible while not being too heavy. Design statements such as this create high performing designs that are cheaply made, thus maximizing performance and profits. This concept can be extended to use with the progressive crush strategy implemented in the previous section so that it is desired to have a design that absorbs as much progressive energy as possible while not being too heavy.

Two optimization studies were performed – one with a conventional strategy which maximized total energy absorbed per unit mass (specific energy absorption: SEA), and one with the new strategy which maximized progressive energy absorbed per unit mass (specific progressive energy absorption: SPEA). The results of both studies are presented here to assess the ability of each to produce designs that absorb energy progressively, while accounting for mass.

For each of the two studies, four optimization runs were performed. In each of these four runs, a different set of starting conditions was used to account for the effects of stochasticity and starting conditions on the search. The same four starting conditions were used for the two studies. In each run, 2,000 crash evaluations were performed using the LS-DYNA explicit finite element code. The hybrid and adaptive algorithm SHERPA within the HEEDS optimization software was again used for the optimization runs. At 2,000 evaluations, the optimization runs were not fully converged, but they had made significant progress toward identifying high performing designs, and their rate of improvement had begun to decrease significantly. Since in most industrial applications of optimization it is not feasible to achieve complete convergence due to time constraints, it

was once more deemed more valuable here to compare the results of four nearly converged designs than one fully converged design.

9.3.1 Conventional Crush Optimization to Maximize Total Energy Absorbed per Unit Mass

The following conventional crush optimization problem definition was used:

Maximize: $SEA = EA/m$

Subject to: $F_2 \leq 150\ kN$

$EA \geq 20\ kJ$

By Varying: *Control Points*

Rail Thickness

Where *SEA* is the total energy absorbed per unit mass by the rail during crush (specific energy absorbed), F_2 is the maximum reaction force after initial crush initiation, *EA* is the total energy absorbed, *Control Points* are the independent control points varied in Figure 8.1.3, and *Rail Thickness* is the rail gage thickness, which varies between 1 mm and 3mm. It should be noted that in addition to the force constraint imposed, there is also an energy constraint. This constraint ensures that low energy absorbing designs with very low masses are deemed inferior designs. This is necessary to ensure high energy absorbing designs are still sought over very low mass designs.

Figure 9.3.1 (a) shows the shapes of the optimized designs found using this design statement, while Figure 9.3.2 (a) shows their force-displacement curves. The performance of these designs is summarized in Table 9.3.1.

9.3.2 Progressive Crush Optimization to Maximize Progressive Energy Absorbed per Unit Mass

The progressive crush optimization problem statement used was:

Maximize: $SPEA = PEA/m$

Subject to: $F_2 \leq 150\ kN$

$PEA \geq 14\ kJ$

By Varying: *Control Points*

Rail Thickness

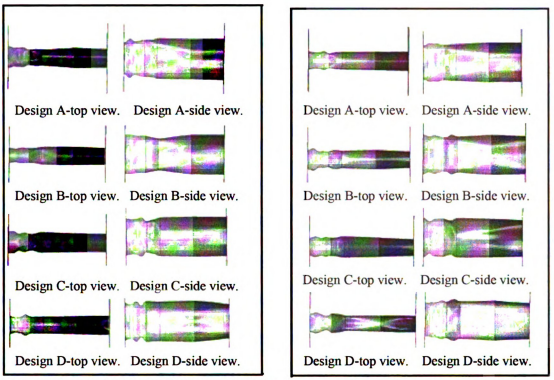
Where *SPEA* is the progressive energy absorbed per unit mass (specific progressive energy absorbed), F_2 is the maximum reaction force after initial crush initiation, *PEA* is the progressive energy absorbed as defined in Equations (7.1,7.2), *Control Points* are the independent control points varied, and *Rail Thickness* is the rail gage thickness, which varies between 1-3 mm. The constraint on *PEA* is necessary to ensure that high energy absorbing designs is sought over low energy absorbing designs with a very low mass.

Figure 9.3.1 (b) shows the shapes of the optimized designs found using this design statement, while Figure 9.3.2 (b) shows their force-displacement curves. The performance of these designs is summarized in Table 9.3.1.

9.3.3 Optimization Results

Based on the optimization results, the following conclusions can be reached. Two out of four runs using the conventional crush design optimization statement yielded optimized solutions that did not exhibit progressive crush behavior, while one of the designs (design D) exhibited very good progressive crush and one of the designs (design A) exhibited moderate progressive crush. All of the optimized designs resulting from the progressive crush optimization statement performed very well in terms of progressive crush. Figures 9.3.3-9.3.6 depict the crushing modes of these designs and support these conclusions.

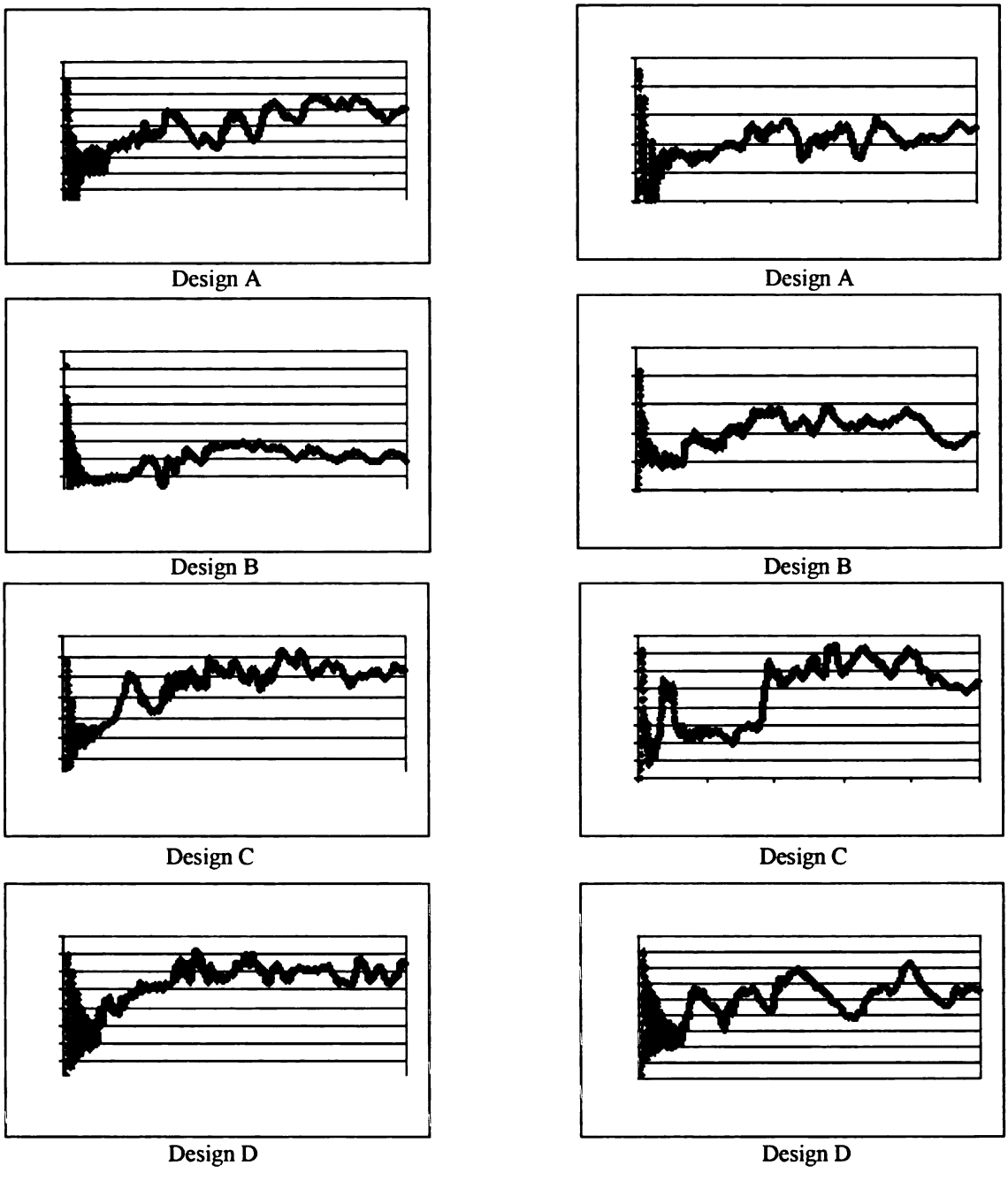
On average, the progressive crush strategy yielded designs with over 27% more specific progressive energy absorbed (SPEA) and 12.6% less total specific energy absorbed (SEA) than did the conventional crush strategy. Perhaps more importantly, the standard deviation of SPEA for designs based on the progressive crush strategy was significantly lower than that for designs based on the conventional strategy. This indicates that the progressive crush strategy was once more, much more robust in terms of producing designs that behaved as intended.



(a) Optimized design shapes from the conventional progressive crush strategy (maximizing SEA = EA/m).

(b) Optimized design shapes from the crush strategy (maximizing SPEA = PEA/m).

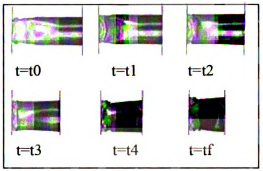
Figure 9.3.1. Comparison of the optimized design shapes from the conventional crush strategy (maximizing SEA=EA/m) and the progressive crush strategy (maximizing SPEA=PEA/m). After 2000 evaluations the optimization runs were not fully converged but had made significant progress.



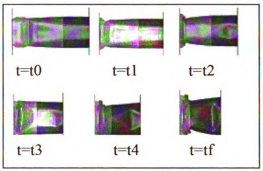
(a) Force-displacement curves of the optimized designs from the conventional crush strategy (maximizing EA/m).

(b) Force-displacement curves of the optimized designs from the progressive crush strategy (maximizing PEA/m).

Figure 9.3.2. Comparison of the force-displacement curves for the optimized shapes from the conventional crush optimization (maximize SEA=EA/m) and the progressive crush optimization (maximize SPEA=PEA/m).

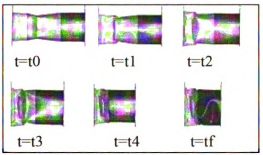


(a) Conventional strategy crush mode.

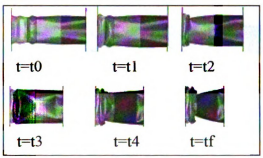


(b) Progressive strategy crush mode.

Figure 9.3.3. Crush modes of the Design A optimized designs. Both the conventional strategy and progressive strategy yielded optimized designs that progressively crushed.

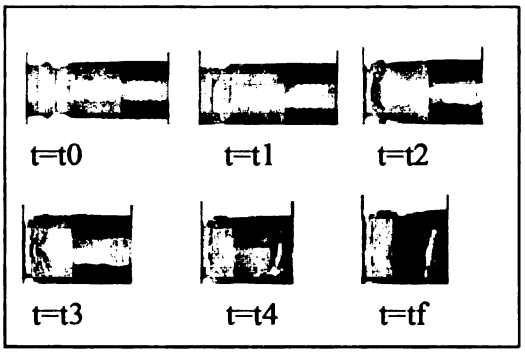


(a) Conventional strategy crush mode.

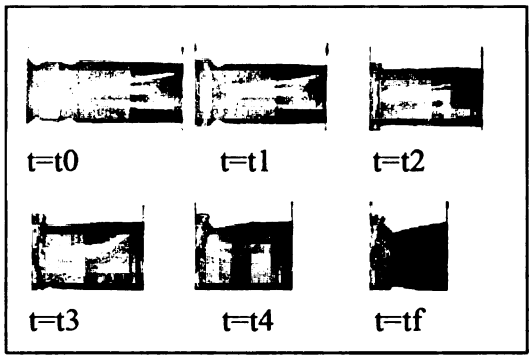


(b) Progressive strategy crush mode.

Figure 9.3.4. Crush modes of the Design B optimized designs. At t = t1, the conventional strategy's optimized design begins to non-progressively crush.

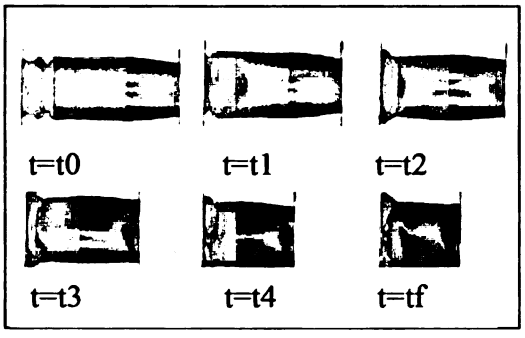


(a) Conventional strategy crush mode.

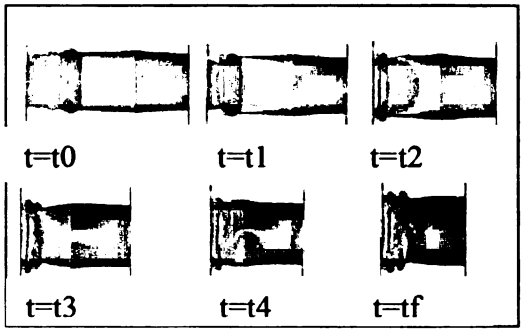


(b) Progressive strategy crush mode.

Figure 9.3.5. Crush modes of the Design C optimized designs. At t = t1, the conventional strategy's optimized design begins to non-progressively crush



(a) Conventional strategy crush mode.



(b) Progressive strategy crush mode.

Figure 9.3.6. Crush modes of the Design D optimized designs. Both the conventional strategy and progressive strategy yielded optimized designs that progressively crushed.

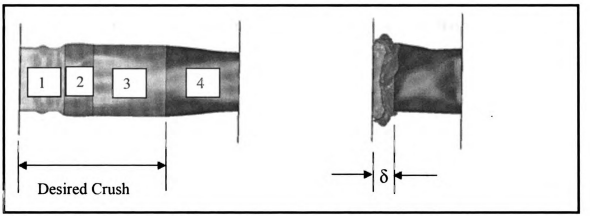
Table 9.3.1. Comparison of progressive crush and conventional crush optimized results when accounting for mass.

	Progressive Crush Strategy					*Conventional Crush Strategy*				
Opt. Design	EA (kJ)	PEA (kJ)	Mass (kg)	SPEA (kJ/kg)	SEA (kJ/kg)	EA (kJ)	PEA (kJ)	Mass (kg)	SPEA (kJ/kg)	SEA (kJ/kg)
A	22.37	18.52	2.85	6.50	7.85	27.35	14.23	2.68	5.34	10.22
B	28.51	17.03	3.06	5.58	9.33	27.31	6.59	2.65	2.49	10.31
C	27.23	14.80	2.86	5.17	9.52	25.82	9.09	2.49	3.65	10.36
D	28.10	16.07	3.06	5.25	9.18	29.30	17.54	2.87	6.11	10.20
Average	26.55	16.60	2.96	5.62	8.97	27.45	11.86	2.67	4.40	10.27
Std Dev	2.84	1.57	0.12	0.61	0.76	1.43	4.94	0.16	1.63	0.07

9.4 Zone 3 Definition Study

9.4.1 Introduction

It was assumed that the displacement measure U_3 had to be offset from being aligned with the end of Zone 3 to account for stackup (recall Figure 8.1.4). This was an assumption justified by the fact that the material crushed during an optimization does not vanish, but rather stacks or "builds up" such that there is a distance δ of stackup (see Figure 9.4.1). Since it is desired that there be no crush occurring in Zone 4, it was assumed that there needed to be a lag in U_3 of distance δ from the end of Zone 3 (the start of Zone 4). This was done with no proof throughout Section 2 of this thesis, with the justification of this concept intuitively making sense.



(a) Uncrushed rail with zones labeled. The desired crush zones are Zones 1-3.

(b) Fully crushed rail.

Figure 9.4.1. Stackup of a typical rail that progressively crushes. Stackup occurs with a material buildup of thickness δ.

Alternatively, the stackup δ is the sum of the stackup occurring in the first three zones (there is a stackup δ_1 in Zone 1, a stackup δ_2 in Zone 2, as well as the stackup

occurring in Zone 3 (δ_3)). For an arbitrary rail of *N* zones, the stackup can be expressed as:

$$\delta = \sum_{i=1}^{N-1} \delta_i \tag{9.1}$$

It has been assumed that accounting for the total stackup (δ) at the end of Zone 3 is a better practice for encouraging progressive crush than not accounting for stackup at all. The value of δ however, has been chosen arbitrarily up until now as 83.4 mm (see Figure 9.4.2). The benefit of accounting for δ, as well as the best value for δ in encouraging progressive crush will be explored in this section. Also, the effect of using δ at the end of Zone 3 as opposed to δ_1 at the end of Zone 1, δ_2 at the end of Zone 2, and δ_3 at the end of Zone 3 (see Figure 9.4.3) will be explored. The decision to use the former in the previous work in this thesis instead of the latter was based upon a quick separate study performed. The validity of this decision will be proved here.

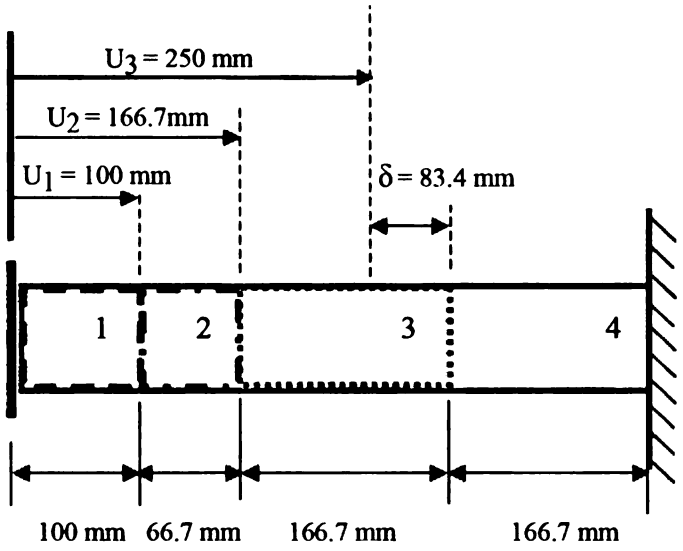


Figure 9.4.2. Zone definitions and impactor displacement specifications used within this study. The material stackup has arbitrarily been set at 83.4 mm.

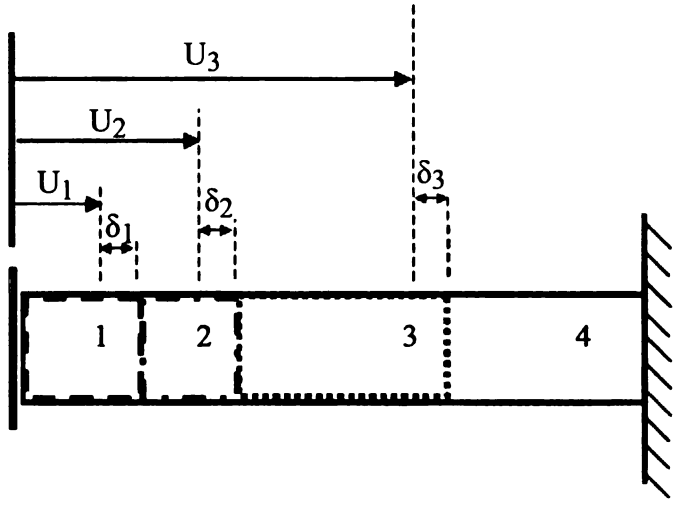


Figure 9.4.3. Alternative definition of accounting for stackup: $\delta_1 + \delta_2 + \delta_3 = \delta$.

9.4.2 Evaluation of the Effect of δ

9.4.2.1 Problem Definition

The size and necessity of the stackup distance δ, along with its effect on the progressive crush optimization strategy, was studied through four optimizations. Each optimization had the same optimization problem statement, and began in the same starting location in the design space with the same random seed for the search. Each optimization however had a different value for δ (see Table 9.4.1). A different value for δ meant that each optimization had a different value of U_3 used in the progressive crush statement. Figure 9.4.4 shows the values of U_i used for each optimization. Aside from this alteration, the analysis model and optimization problem statement remained the same as used throughout this thesis, with the progressive energy optimization statement here being:

Maximize: *PEA*

Subject to: $F_2 \leq 150\ kN$

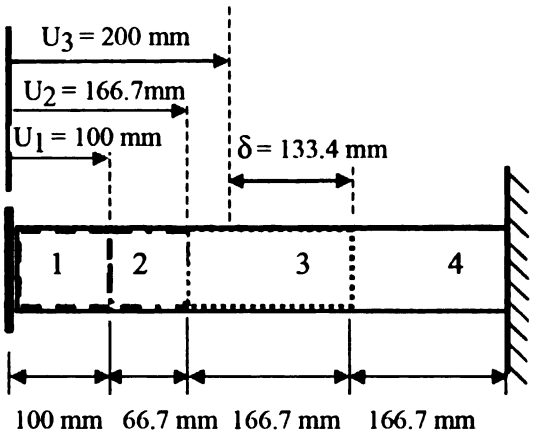
By Varying: *Control Points*

Rail Thickness

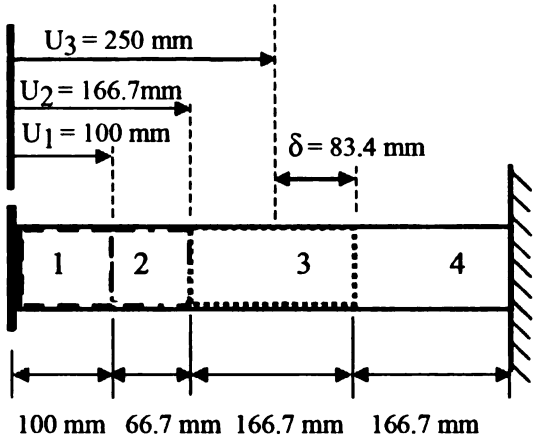
Where *PEA* is the progressive energy absorbed (as defined previously), F_2 is the maximum reaction force after initial crush initiation, *Control Points* are the independent control points varied, and *Rail Thickness* is the rail gage thickness, which varies between 1-3 mm.

Table 9.4.1. Stackup distance δ allotted for each optimization in the study.

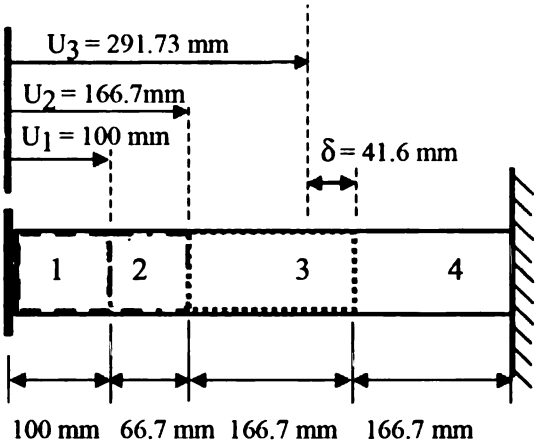
	Optimization A	Optimization B	Optimization C	Optimization D
δ (mm)	133.4	83.4	41.6	0



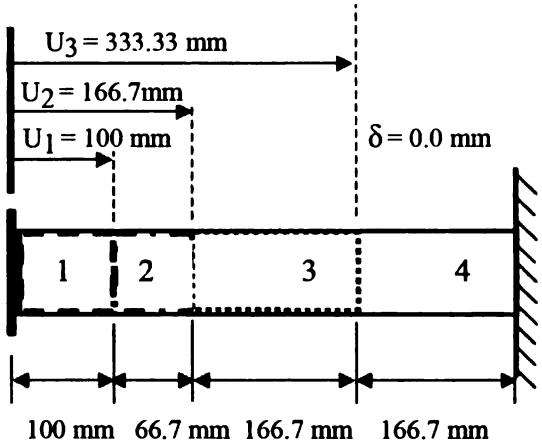
(a) Zone definition for Optim. A (δ = 133.4 mm).



(b) Zone definition for Optim. B (δ = 83.4 mm).



(c) Zone definition for Optim. C (δ = 41.6 mm).



(d) Zone definition for Optim. D (δ = 0.0 mm).

Figure 9.4.4. Zone definitions used for the different optimizations performed in the study to determine the effect of δ.

9.4.2.2 Results

The resulting optimized designs from the four optimizations performed with different values of U_3 are shown in Table 9.4.2. From this table it is clear to see that the second optimization (Optimization B with U_3 = 250 mm) yielded the design that absorbed the most total energy. Comparing the progressive energy for the optimized designs from the four optimizations is not direct however, since the progressive energy problem statement used in each optimization was different (due to U_3 being different). Therefore for comparison, the four optimized designs had PEA calculated for four different values of δ. Table 9.4.3 and Figure 9.4.5 show these results.

The design from the second optimization (Design B) had the highest PEA regardless of the value of U_3 used in the definition of PEA. The designs from the first (Design A) and fourth (Design D) optimizations had the lowest PEA regardless of the value of U_3 used. Recall that the optimization statement for which Design A was found used a value of U_3 equal to 200 mm (δ = 133.4 mm), while the optimization for which Design D was found used a value of U_3 equal to 333.33 mm (δ = 0.0 mm). Therefore, optimizing for too large of stackup (as shown for δ = 133.4 mm) and no stackup at all (δ = 0.0 mm) is not as efficient of an optimization statement as having values of δ close to the actual stackup that can be expected. Optimizations seeking progressive crush therefore should use a definition of U_3 that accounts for a realistic stackup δ.

In addition, the fact that in Figure 9.4.5 the PEA doesn't change for any of the four designs when calculating PEA with a statement of $\delta = 41.6$ mm and a statement of $\delta = 0.0$ mm, shows that stackup indeed does occur. If this were not the case, from U = 291.73 mm to U = 333.33 mm there would be an increase in energy absorbed in the rail in Zone 3. This however is not the case demonstrating that the material has stacked up as much as possible for the design in the first 3 zones and any additional energy absorbed likely comes from the fourth zone (which is undesired). Figure 9.4.6 shows the crush modes of the four optimized designs found with the different optimization statements via the definition of U_3.

Table 9.4.2. Results for each optimization.

	Optimization A	Optimization B	Optimization C	Optimization D
PEA (kJ)	12.187	18.429	19.480	17.606
EA (kJ)	26.105	29.195	26.600	25.018

Table 9.4.3. Optimized design from each optimization re-evaluated in terms of PEA for four different values of δ.

	Design A		Design B		Design C		Design D	
δ (mm)	PEA (kJ)	EA (kJ)	PEA (kJ)	EA (kJ)	PEA (kJ)	EA (kJ)	PEA (kJ)	EA (kJ)
133.4	12.187	26.105	12.320	29.195	11.043	26.600	10.718	25.018
83.4	14.945	26.105	18.429	29.195	16.666	26.600	15.208	25.018
41.6	16.736	26.105	20.651	29.195	19.480	26.600	17.606	25.018
0.0	16.736	26.105	20.651	29.195	19.480	26.600	17.606	25.018

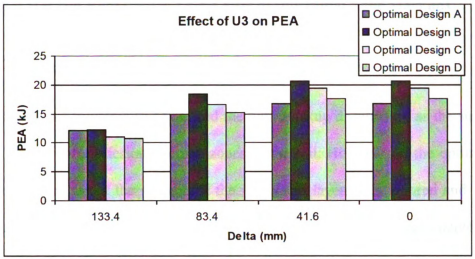


Figure 9.4.5. Optimized designs from the four optimizations with PEA calculated for different values of δ. The optimized design from optimization B had the highest PEA regardless of the statement used. This shows that realistic values for the stackup δ occurring in crush should be used in the optimization problem statement to encourage the best progressive crush.

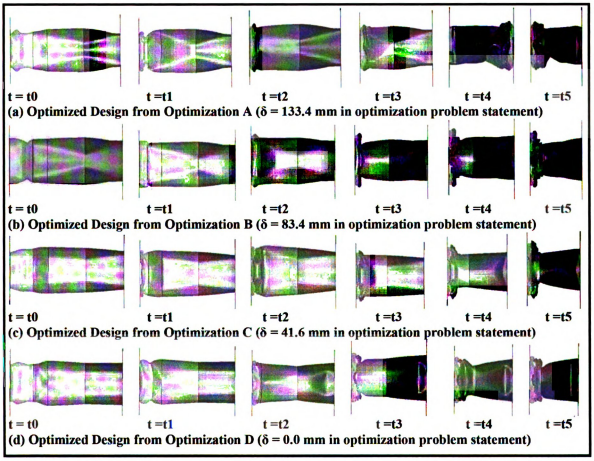


Figure 9.4.6. Crush modes of the optimized designs from the optimization study of the impact of the value of U_3 on the optimization.

9.4.3 Total Stackup Represented by the Sum of Individual Zone Stackups

9.4.3.1 Problem Statement

Recall that Equation (9.1) defined the total stackup distance (δ) as the sum of the material stackup of all desirable crush zones in the rail. However, in the optimizations performed in this report the stackup of the individual layers was ignored and only the total stackup was accounted for; the optimization problem statements considered δ at the end of Zone 3, as opposed to δ_1 at the end of Zone 1, δ_2 at the end of Zone 2, and δ_3 at the end of Zone 3. The validity of considering only δ at the end of Zone 3 will be explored through a series of three optimizations.

Using the observed best value for δ from the optimization study just performed (δ = 83.4 mm; U_3 = 250 mm), two additional optimizations were performed to study the impact the values of δ_1, δ_2, δ_3 have on the optimization for a fixed δ. The optimizations used the same problem statement as before, started in the same location in the design space, and had the same random seed.

Table 9.4.4 displays the differing characteristics among the optimizations and compares them to that of Optimization B from before (where δ = 83.4 mm; U_3 = 250 mm). Optimization B recall, had a value of δ = 83.4 mm applied as δ_3 only. Therefore, U_3 was the only U_i offset from the end of the respective zone. Optimizations B-i and B-ii however, have values of δ_1 and δ_2 in addition to δ_3, meaning U_1 and U_2 are offset from the end of their zones as well. Optimization B-i had $\delta_1 = \delta_2 = \delta_3$ = 27.8 mm (δ = 83.4

mm). This optimization statement assumed equal stackup between zones. This likely isn't the case however, since the zones are in different locations and have different lengths. Optimization B-ii therefore, scaled the lengths of δ_i according to the length of the zone. Since Zone 3 was larger than Zone 2, and Zone 1 larger than Zone 2: $\delta_3 > \delta_1 > \delta_2$ (δ = 83.4 mm).

Table 9.4.4. Stackup definitions used to study the relationship between total stackup (δ) and the individual zone stackups (δ_i).

Optimization	δ (mm)	δ_1 (mm)	δ_2 (mm)	δ_3 (mm)	U_1 (mm)	U_2 (mm)	U_3 (mm)
B	83.4	0	0	83.4	100	166.7	250
B-i	83.4	27.8	27.8	27.8	72.2	138.9	305.6
B-ii	83.4	25	16.7	41.7	75	150	291.7

9.4.3.2 Results

The optimized design from all optimizations had very similar total energy absorbed (29-30 kJ). Again, due to the definitions of their objective function being different (PEA defined differently due to different U_1, U_2, and U_3 for all three optimizations being compared), a direct comparison of progressive energy absorbed cannot be done without calculating PEA for all definitions of δ used. Table 9.4.5 and Figures 9.4.7 and 9.4.8 display these results.

Design B outperforms the other two designs in terms of PEA regardless of the definition used. At the same time, the designs are *very* similar in performance (as is evident by the small standard deviations seen in Figure 9.4.8). This leads to the conclusion that accounting for total stackup (δ) is the same or better than trying to account for the stackup of each individual zone. This is more practical anyhow since

estimating total stackup after the entire crush is performed is likely easier than predicting stackup at various stages of the crush.

Table 9.4.5. Optimized design from each optimization with energy absorbed calculated for three different stackup definitions

			Design B		Design B-i		Design B-ii	
U_1 (mm)	U_2 (mm)	U_3 (mm)	PEA (kJ)	EA (kJ)	PEA (kJ)	EA (kJ)	PEA (kJ)	EA (kJ)
100	166.7	250	18.43	29.19	17.37	29.96	17.34	29.20
72.2	138.9	305.6	23.16	29.19	22.72	29.96	22.82	29.20
75	150	291.7	22.89	29.19	22.55	29.96	21.99	29.20

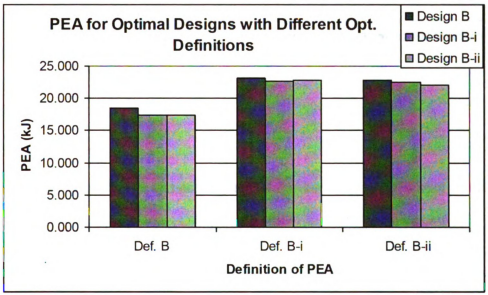


Figure 9.4.7. Progressive energy absorbed by the different optimized designs with the different optimization definitions.

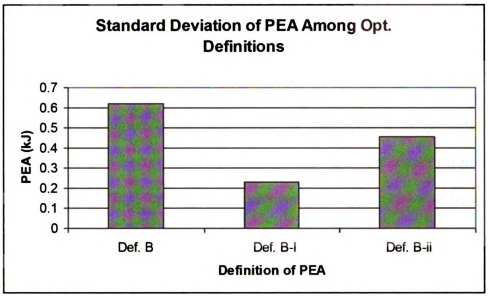


Figure 9.4.8. Standard deviation of the progressive energy absorbed among the three designs for the different optimization definitions.

9.4.4 Conclusions

It has been shown that accounting for stackup in your definition of PEA (through the definition of U_1, U_2, U_3) produces better progressively crushing designs than not accounting for stackup. In addition, it has been shown that giving reasonable stackup definitions gives better performing designs than over-predictions or under-predictions of stackup. Also, it has been demonstrated that approximating the total stackup (δ) and using that to define U_3 (or U_{N-1} for N zones), produces as good or better designs than accounting for the stackup in individual zones (in defining U_1, U_2, and U_3). This section therefore justifies and explains the use of U_3 = 83.4 mm for all work performed in Section II of this thesis.

CHAPTER 10: SUMMARY AND CONCLUSIONS

There are common optimization problem statements used for optimizing the performance of hybrid electric vehicles as well as the crashworthiness of front rails in an automobile vehicle. These optimization statements have been shown to be suitable for achieving certain desired performance characteristics, but not others. In the current study, alternative optimization problem statements have been shown to be as effective as traditional statements for the problems posed, with additional benefits.

Hybrid electric vehicle (HEV) optimizations have primarily been done utilizing a state of charge (SOC) correction to account for the effects of the initial SOC on performance. In this study, this common approach (referred to herein as Strategy 2) was compared with two alternative approaches in optimizing a series hybrid bus. The first alternative strategy (Strategy 1) utilized multiple drive cycles and multiple initial SOC load cases as a means to account for initial SOC, instead of an SOC correction. This optimization strategy considered only the fuel consumption from the engine while neglecting energy usage by the other components. Like Strategy 2, another alternative strategy (Strategy 3) accounted for total energy usage as opposed to engine energy only. This was achieved using the conversion for the electrical energy that can be converted to gasoline through the efficiency of an average power plant in the United States. Strategy 3 like Strategy 1, did not utilize an SOC correction but rather multiple drive cycles and multiple initial SOC load cases.

Strategy 1 was shown to be effective only if energy usage from the engine is the primary concern, and a fixed driving route can be guaranteed frequently. Strategies 2 and 3 however, were shown to be effective if total energy usage is the primary interest. Strategy 3 was shown to have a similar performance as Strategy 2 while being more run-time efficient for the problem posed.

As with HEV optimizations, there is a typical optimization strategy used for optimizing the crashworthiness of front rails in automotive vehicles. This common approach focuses on maximizing total energy absorption by the rail during impact. It does not however, guarantee the rail will crush progressively, as is desired for safety and repair cost reasons. An alternative strategy for optimizing front rails therefore was proposed that accounts for a quantity defined herein as progressive energy absorption (PEA). PEA considers the time in which the energy absorption occurs in a given region of the rail. High values of PEA correspond with designs that crush in the front of the rail and propagate rearward with time.

The alternative approach for optimizing front rails was shown to yield optimized designs with similar total energy absorption as conventional strategies, but with significantly improved progressive crush modes. Through multi-objective optimizations, the relationship between PEA and total energy absorption was explored; it was found that as PEA increases, so does total energy absorption.

REFERENCES

[1] L. Guzzella, A. Sciarretta, *Vehicle Propulsion Systems: Introduction to Modeling and Optimization*, Springer, 1 Edition, September 13, 2005.

[2] J. Miller, *Propulsion Systems for Hybrid Vehicles,* IET, 2004.

[3] M. Ehsani, Y. Gao, J. Miller, *Hybrid Electric Vehicles: Architecture and Motor Drives*, Proceedings of the IEEE, Volume 95, Number 4, April 2007, pp. 719-728.

[4] C. Chan, *The State of the Art of Electric Hybrid, and Fuel Cell Vehicles,* Proceedings of the IEEE, Volume 95, Number 4, April 2007, pp. 704-718.

[5] W. Gao, C. Mi, A. Emadi, *Modeling and Simulation of Electric and Hybrid Vehicles,* Proceedings of the IEEE, Volume 95, Number 4, April 2007, pp. 729-745.

[6] W. Gao, S. Musunuri, *Hybrid Electric Vehicle Modeling and Analysis in Generic Modeling Environment,* Proceedings of the IEEE Vehicle Power and Propulsion Conference, 2006.

[7] W. Gao, S. Neema, et al, *Hybrid Powertrain Design Using a Domain-Specific Modeling Environment,* Proceedings of the IEEE Vehicle Power and Propulsion Conference, 2005, pp. 530-535.

[8] K. Wipke, M. Cuddy, S. Burch, *ADVISOR 2.1: A User-Friendly Advanced Powertrain Simulation Using a Combined Backward/Forward Approach,* National Renewable Energy Laboratory, August 1999.

[9] ADVISOR User's Manual, AVL.

[10] PSAT Documentation. [Online]. Available: http://www.transportation.anl.gov/software/PSAT

[11] PSIM Website. [Online]. Available: http://www.pwersimtech.com/

[12] VTB Website. [Online]. Available: http://www.vtb.ee.sc.edu/

[13] L. Guzzella, A. Amstutz, *CAE Tools for Quasi-Static Modeling and Optimization of Hybrid Powertrains*, IEEE Transactions on Vehicular Technology, Volume 48, Number 6, November 1999, pp. 1762-1769.

[14] K. Oh, J. Kim, et al, *Optimal Power Distribution Control for Parallel Hybrid Electric Vehicles*, IEEE International Conference on Vehicular Electronics and Safety, 2005, pp. 79-85.

[15] M. Montazeri-Gh, A. Poursamad, B. Ghalichi, *Application of genetic algorithm for optimization of control strategy in parallel hybrid electric verhicles*, Journal of the Franklin Institute, 2006, pp. 420-435.

[16] A. Piccolo, L. Ippolito, et al, *Optimisation of energy Flow Management in Hybrid Electric Vehicles via Genetic Algorithms,* IEEE/ASME International Conference on Advanced Intelligent Mechatronics Proceedings, 2001, pp. 434-439.

[17] W. Gao, S. Porandla, *Design Optimization of a Parallel Hybrid Electric Powertrain*, Proceedings of the IEEE Vehicle Power and Propulsion Conference, 2005, Chicago.

[18] T. Markel, K. Wipke, *Optimization Tools for Hybrid Vehicle Systems Analysis,* FY Annual Progress Report on Vehicle Propulsion and Ancillary Subsystems Program, 2001, pp. 5-8.

[19] T. Markel, K. Wipke, *ADVISOR Improvement, Validation, and Application,* FY Annual Progress Report on Vehicle Propulsion and Ancillary Subsystems Program, 2001, pp. 8-12.

[20] D. Assanis, G. Delagrammatikas, et al, *An Optimization Approach to Hybrid Electric Propulsion System Design*, Mechanics of Structures and Machines, Volume 27, No. 4, 1999., pp. 393 – 421.

[21] R. Fellini, et. al, *Optimal Design of Automotive Hybrid Powertrain Systems,* IEEE Proceedings, EcoDesign '99, First International Symposium On Environmentally Conscious Design and Inverse Manufacturing, 1999, pp. 400-405.

[22] T. Moore, *HEV Control Strategy: Implications of Performance Criteria, System Configuration and Design, and Component Selection*, Proceedings of the American Control Conference, June 1997, pp. 679-683.

[23] W. Wayne, et. al, *A Comparison of Emissions and Fuel Economy from Hybrid-Electric and Conventional-Drive Transit Buses*, Energy and Fuels, 2004, 18, pp. 257-270.

[24] SAE J2711, *Recommended Practice for Measuring Fuel Economy and Emissions of Hybrid-Electric and Conventional Heavy-Duty Vehicle,* SAE International Surface Vehicle Recommended Practice, 2002.

[25] Pagerit, S., Rousseau, A., Sharer, P., *Global Optimization to Real Time Control of HEV Power Flow: Example of a Fuel Cell Hybrid Vehicle,* International Electric Vehicle Symposium (EVS20), Monaco, (April 2005).

[26] M. Duoba, *The 1996 FutureCar Challenge Final Rules and Regulations,* 1995, Section E-11.3.3, pp. 55.

[27] R. Senger, *Validation of ADVISOR as a Simulation Tool for a Series Hybrid Electric Vehicle Using the Virginia Tech FutureCar Lumina*, M.S. Thesis, Virginia Polytechnic Institute and State University, 1998.

[28] N. Chase, M.Rademacher, E.Goodman, R.Averill, R.Sidhu, *A Benchmark Study of Optimization Search Algorithms,* Red Cedar Technology, BMK-3022, 2008.

[29] HEEDS Professional User's Manual, Version 5.2, Red Cedar Technology, 2008.

[30] TransTeq. [Online]. Available: http://www.transteq.com/

[31] DiPaolo, B.P., Monteiro, P.J.M., and Gronsky, R. , *Quasi-static axial crush response of a thin-wall, stainless steel box component*, International Journal of Solids and Structures, 41:3707-3733, 2004.

[32] Jones, N., *Structural Impact*, Cambridge University Press, Cambridge,1989.

[33] Karagizova, D., and Jones, N., *Counterintuitive Response of Long Circular Tubes to Axial Impact*, XXI International Congress of Theoretical and Applied Mechanics, Warsaw Poland, August 2004.

[34] Guillow; S.R., Lu, G., and Grzebieta R.H., *Quasi-static axial compression of thin-walled circular aluminum tubes*, International Journal of Mechanical Sciences, 43:2103-2123, 2001.

[35] Zarei, H.R., and Kroger, M., *Multiobjective crashworthiness optimization of circular aluminum tubes,* Thin-Walled Structures, 44:301-308, 2006.

[36] Ku, J.H., Seo, K.S., Park, S.W., and Kim, Y.J, *Axial crushing behavior of the intermittent tack welded cylindrical tubes,* International Journal of Mechanical Sciences, 43:521-542, 2001.

[37] Gumruk, R., and Karadeniz, S., *A numerical study of the influence of bump type triggers on the axial crushing of top hat thin-walled sections*, Thin-Walled Structures, 46:1094-1106, 2008.

[38] Hou, S., Li, Q., Long, S., Yang, X., and Li, W., *Design optimization of regular hexagonal thin-walled columns with crashworthiness criteria*, Finite Elements in Analysis and Design, 43:555-565, 2007.

[39] Kim, H.-S., *New extruded multi-cell aluminum profile for maximum crash energy absorption and weight efficiency,* Thin-Walled Structures, 40:311-327, 2002.

[40] Abramowicz, W., and Jones, N., *Transition from initial global bending to progressive buckling of tubes loaded statically and dynamically*, International Journal of Impact Engineering, 19(5-6):415-437, 1997.

[41] Yamazake, K., and Han, J., *Maximization of the crushing energy absorption of tubes,* Structural Optimization, Springer-Verlag,16:37-46, 1998.

[42] Sidhu, R., Burgueno, J., Averill, R., and Goodman, E., *Improved Vehicle Crashworthiness via Shape Optimization*, Proceedings of the ASME International Mechanical Engineering Congress & Exposition, Washington D.C, 2003.

[43] Sidhu, R., Burgueno, J., Averill, R., and Goodman, E., *Shape Optimization for Improved Vehicle Safety and Reliability*, 2003 ABAQUS User's Conference, 2003.

[44] Cho, Y., Bae, C., Suh, M., and Sin, H., *A vehicle front frame crash design optimization using hole-type and dent-type crush initiator*, Thin-Walled Structures, 44:415-428, 2006.

[45] Kim, H.-S., Chen, W., and Wierzbicki, T., *Weight and crash optimization of foam-filled three-dimensional "S" frame*, Computational Mechanics, 28:417-424, 2002.

[46] Kurtaran, H., Eskandarian, A., Marzougui, D., and Bedewi, N.E., *Crashworthiness design optimization using successive response surface approximations*, Computational Mechanics, 29:409-421, 2002.

[47] Avalle, M., Chiandussi, G., and Belingardi, G, *Design optimization by response surface methodology: application to crashworthiness design of vehicle structures,* Structural Multidisciplinary Optimization, Springer-Verlag, 24:325-332, 2002.

[48] Karbhari, V.M., and Haller, J.E., *Rate and architecture effects on progressive crush of braided tubes,* Composite Structures, 43:93-108, 1998.

[49] Hanssen, A.G., Langseth, M., and Hopperstad, O.S., *Optimum design for energy absorption of square aluminum columns with aluminum foam filler*, International Journal of Mechanical Sciences, 43:153-176, 2001.

[50] *LS-DYNA Users Manual*, 2008.