4-2022

Autonomous Eco-Driving With Traffic Light And Lead Vehicle Constraints: An Application Of Best Constrained Interpolation

Yara Hazem Mohamed Mahmoud
Western Michigan University

Follow this and additional works at: https://scholarworks.wmich.edu/masters_theses

Part of the Mathematics Commons

Recommended Citation
https://scholarworks.wmich.edu/masters_theses/5332

This Masters Thesis-Open Access is brought to you for free and open access by the Graduate College at ScholarWorks at WMU. It has been accepted for inclusion in Masters Theses by an authorized administrator of ScholarWorks at WMU. For more information, please contact wmu-scholarworks@wmich.edu.
Eco-Driving is a critical technology for improving automotive transportation efficiency. It is achieved by modifying the driving trajectory over a particular route to minimize required propulsion energy. Eco-Driving can be approached as an optimal control problem subject to driving constraints such as traffic lights and positions of other vehicles. Best interpolation in a strip is a problem in approximation theory and optimal control. The solution to this problem is a cubic spline. In this research we demonstrate the connection between Eco-Driving and best interpolation in the strip. By exploiting this connection, we are able to generate optimal Eco-Driving trajectories that could be driven with an autonomous system. Using the simulation tool FASTSim, we can evaluate the trajectories using conventional, hybrid electric, and fully electric vehicle models. Our results quantify the energy efficiency improvements that can be achieved with the proposed approach.
AUTONOMOUS ECO-DRIVING WITH TRAFFIC LIGHT AND LEAD VEHICLE CONSTRAINTS: AN APPLICATION OF BEST CONSTRAINED INTERPOLATION

by

Yara Hazem Mohamed Mahmoud

A thesis submitted to the Graduate College in partial fulfillment of the requirements for the degree of Master of Science in Mathematics Western Michigan University April 2022

Thesis Committee:

Melinda Koelling, Ph.D., Chair
Zachary D. Asher, Ph.D.
Jim Zhu, Ph.D.
ACKNOWLEDGMENTS

I would like to thank my thesis Advisor, Dr Melinda Koelling for helping me to get to do a thesis-based masters and start a program from scratch, Dr Asher for believing in me and taking me in his lab when no one else did, The EEA V lab graduate students for being with me in my journey and helping me out whenever they can.

I would like to thank especially Dr Ziebarth (Chair of the Math Department) for providing me the funds needed to complete my education. Without this fund I would not have been able to focus on my work and I would have struggled financially.

My parents and friends who bore my crying nights and anxiety. My mom’s and grandma’s prayers, Noha who helped me throughout my first year of college, Mariam, who was my partner in crime from day one and college and graduate school sucks without her, and Matthew who is still helping me with Linear Algebra class.

Last but not least, I would like to thank my self for getting me to this point of education and being determined to achieve my dreams. It has not been all pink and white, I had some grey and black days, however I was determined to prove myself and I did.

Yara Hazem Mohamed Mahmoud
TABLE OF CONTENTS

ACKNOWLEDGMENTS ........................................................................................................ ii

LIST OF TABLES ........................................................................................................ iv

LIST OF FIGURES .......................................................................................................... v

LIST OF ABBREVIATIONS .............................................................................................. vi

CHAPTER

1 INTRODUCTION ........................................................................................................... 1
  1.1 Literature Review .................................................................................................... 1
  1.2 Importance of Eco-Driving .................................................................................... 2
  1.3 Research Goals and Contribution ........................................................................... 3

2 METHODOLOGY ......................................................................................................... 5
  2.1 Problem Statement and Assumptions ...................................................................... 5
  2.2 Optimal Control Problem Formulation .................................................................... 6
  2.3 Optimal Control Solution ....................................................................................... 11
  2.4 FE Evaluation ......................................................................................................... 16

3 RESULTS ...................................................................................................................... 17
  3.1 Optimal Control Solution ....................................................................................... 17
  3.2 FE Results ............................................................................................................... 19

4 CONCLUSION ............................................................................................................. 21

BIBLIOGRAPHY ............................................................................................................. 23

APPENDIX ....................................................................................................................... 27
LIST OF TABLES

3.1 Energy efficiency (i.e. FE) of all drive cycles for an HEV, EV, and CV . . . . . . . . . 20
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Visualization of SPaT data of 3 traffic lights where red dashes indicate when each light is red. Traffic lights are located approximately 1.9, 3.9, and 5.7 tenth of a mile.</td>
</tr>
<tr>
<td>2.2</td>
<td>In (blue) trajectory of forecasted lead vehicle travelling at speed limit and stops at any red light it encounters. In (pink) the trajectory window just before the traffic light turn red.</td>
</tr>
<tr>
<td>2.3</td>
<td>Visualization of the gates (double-ended arrows) that constrain the location of the ego vehicle.</td>
</tr>
<tr>
<td>2.4</td>
<td>Visualization of the location of waypoints (black circles) and velocity of the ego vehicle (slope of strike through black lines).</td>
</tr>
<tr>
<td>2.5</td>
<td>General problem geometry showing in the green dashed circle a zoomed in strip between ( t_i = 3.1 ) and ( t_{i+1} = 4.8 ) mins, the lead vehicle ( d(t) ), red light phase constraint ( e(t) ).</td>
</tr>
<tr>
<td>2.6</td>
<td>Details of the Kalamazoo Arterial route which is used to collect lead vehicle and SPaT data constraints.</td>
</tr>
<tr>
<td>2.7</td>
<td>Details of the Kalamazoo Downtown route which is used to collect lead vehicle and SPaT data constraints.</td>
</tr>
<tr>
<td>2.8</td>
<td>No active constraints definition, where ( a = t_i ) and ( b = t_{i+1} ).</td>
</tr>
<tr>
<td>2.9</td>
<td>Single touching point definition, where ( a = t_i ) and ( b = t_{i+1} ).</td>
</tr>
<tr>
<td>2.10</td>
<td>Two touching points, subarc, definition, where ( a = t_i ) and ( b = t_{i+1} ).</td>
</tr>
<tr>
<td>2.11</td>
<td>Touching pair definition, where ( a = t_i ) and ( b = t_{i+1} ).</td>
</tr>
<tr>
<td>3.1</td>
<td>Kalamazoo Arterial route results for the Eco-Driving solution with an initial velocity chosen by the optimizer, drive cycle KA1 (a), and an initial velocity assigned as half of the speed limit, drive cycle KA2 (b).</td>
</tr>
<tr>
<td>3.2</td>
<td>Kalamazoo Downtown route results for the Eco-Driving solution with an initial velocity chosen by the optimizer, drive cycle KD1 (a), and an initial velocity assigned as half of the speed limit, drive cycle KD2 (b).</td>
</tr>
</tbody>
</table>
## LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAFE</td>
<td>Corporate Average Fuel Economy</td>
</tr>
<tr>
<td>CV</td>
<td>Conventional Vehicle</td>
</tr>
<tr>
<td>EEAV lab</td>
<td>Energy Efficiency and Autonomous Vehicles lab</td>
</tr>
<tr>
<td>Ego Vehicle</td>
<td>Testing Vehicle</td>
</tr>
<tr>
<td>EPA</td>
<td>Environmental protection agency</td>
</tr>
<tr>
<td>EV</td>
<td>Electric Vehicle</td>
</tr>
<tr>
<td>FASTSim</td>
<td>Future Automotive Systems Technology Simulator</td>
</tr>
<tr>
<td>FE</td>
<td>Fuel Economy</td>
</tr>
<tr>
<td>HEV</td>
<td>Hybrid Electric Vehicle</td>
</tr>
<tr>
<td>KA</td>
<td>Kalamazoo Arterial</td>
</tr>
<tr>
<td>KD</td>
<td>Kalamazoo Downtown</td>
</tr>
<tr>
<td>Lead Vehicle</td>
<td>Upper limit (speed) constraint</td>
</tr>
<tr>
<td>MPGge</td>
<td>Mileage per gallon gasoline equivalent</td>
</tr>
<tr>
<td>MPG</td>
<td>Mileage per gallon</td>
</tr>
<tr>
<td>SPaT</td>
<td>Signal Phasing and Timing</td>
</tr>
<tr>
<td>V2I</td>
<td>Vehicle-to-Infrastructure</td>
</tr>
<tr>
<td>V2V</td>
<td>Vehicle-to-Vehicle</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

In the United States, the National Highway Traffic Safety Administration’s (NHTSA) Corporate Average fuel economy (CAFE) requirement is gradually raised on an annual basis to reduce the adverse consequences of automotive transportation (1). These adverse consequences include economic costs for petroleum importation and vulnerability to geopolitical stability (2), accelerated global warming due to large amounts of greenhouse gas emissions (3), and adverse effects on human health due to air pollution (4; 5). To date, the CAFE policy has been shown to be effective in promoting energy efficiency technology in modern commercially-available vehicles (6).

1.1 Literature Review

In 1977, Schwarzkopf first demonstrated the fuel economy in road vehicles does not only depend on the drive train but also on the driving pattern (7). Schwarzkopf algorithm was derived from the Pontryagin Maximum Principle to provide a mathematically optimal driving performance subject to the driver’s trip time limit. In 1993, Dontchev introduced the problem of the best interpolation in a strip. Dontchev proved that the solution to his problem is a cubic spline which depends continuously on the data. In 1995, Dontchev and Kolmanovsky presented a method for the reduction of best interpolation in a strip to an unconstrained minimization problem that can be used in many optimization applications (8).

In 2001, Ericsson (9) showed that different driving patterns have different effects on vehicle emissions and fuel consumption. In 2012, the National Renewable Energy Laboratory researchers showed that implementing Eco-Driving behavior results in fuel economy improvements (10). In 2014, researchers at the University of Minnesota applied a traffic light model to predict future vehicle velocity with vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) technologies as
inputs (11). Their research employed Pontryagin’s Minimization Principle which resulted in an improvement in the fuel economy (FE) with the prediction-with-error scenario. Starting in 2018, researchers at Colorado State University utilized a speed prediction method utilizing simulated V2V communication with real-world driving data and a drive cycle database to improve Vehicles FE using a predictive energy management system (12).

Additionally, research that was conducted at the University of Michigan and then transitioned to Western Michigan University continued to investigate and exploit further methods to improve FE using different control methods. In 2020, Western Michigan University Energy efficient and Autonomous Vehicles (EEAV) lab and the Department of Mathematics collaborated with the University of Michigan researchers to apply the “Best interpolation in a strip” problem to a real-life application that resulted in this research.

1.2 Importance of Eco-Driving

Eco-Driving is an effective method of achieving significant energy efficiency improvements in vehicles. Eco-Driving is the realization of a more energy-efficient drive cycle for a given route. It is typically achieved by eliminating full stops, maintaining a constant speed, limiting acceleration, and smoothing the velocity profile (10) (13) (14). Implementing energy-efficient driver behavior reduces the vehicle propulsion energy required to drive the route.

Energy-efficient driver behavior typically results in longer travel time and it has historically lower user acceptance by human drivers (10). However, human driver resistance may be reduced when combined with driving automation technologies. Automation technologies, such as advanced driving assistance systems (ADAS) and autonomous vehicles (AV) have high user acceptance (15), and it is likely that users can look past the increased travel time from Eco-Driving in exchange for not having to do all the driving.

Recent research has shown that when implementing a heuristic set of goals such as removing stops, traveling at an energy-efficient speed (in general, this could be a higher or lower overall speed), and limiting acceleration and deceleration magnitudes, it is possible to achieve Fuel Economy (FE) improvements of approximately 10% for modern vehicles and 30% for fully autonomous vehicles (13). Also, when Eco-driving is applied to electrified vehicles the result is longer battery-
life and slower battery degradation (16). Additionally, vehicles without Eco-Driving technology can still improve their energy efficiency if they follow an Eco-driving equipped lead vehicle (17).

The problem of maximizing the energy efficiency improvements from Eco-Driving can be formulated as an optimal control problem if a prediction of driving conditions along the route, such as traffic lights and traffic positions, are available. These predictions are informed by vehicle sensors and vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) communication technologies. As an example of a typical Eco-Driving study, predictions of traffic light signal phase and timing (a V2I technology) were used to change driving behavior and demonstrated a fuel economy improvement of 12-14% (18).

Another study showed that using V2I-enabled traffic lights decreases the energy consumed in Eco-Driving (17). Other approaches include the optimization of speed and power distribution between the motor and engine in a connected HEV subject to A/C thermal load (19), imposing constraints on vehicles that are queued for a green light (20; 21), and using information from the vehicle in front and traffic light data to avoid red lights (22). Additionally, applications of Model Predictive Control (MPC) and Dynamic Programming (DP) have been considered in a lead vehicle following scenario (23). Other connections between control theory and spline interpolation are illustrated in (24).

1.3 Research Goals and Contribution

Our approach here is unique compared to the past efforts because we demonstrate a connection between Eco-Driving and a problem in approximation theory and optimal control (25; 26). This problem is called “best interpolation in a strip”. By exploiting this connection, we show that it is possible to quickly generate optimal Eco-Driving trajectories for real-world driving scenarios with traffic lights and lead vehicle constraints. As expected, these derived optimal trajectories show significant energy efficiency improvements when used in real-world validated models and compared to EPA reported FE.

After this chapter, we proceed as follows.

In Chapter 2, we first state the methodology where assumptions are explained, the problem is formulated graphically, and the optimal control solution derivation and possible solutions to
the problem explained. Furthermore, we go over the implementation of the problem algorithm on different routes and using different vehicle architecture. In Chapter 3, we study the Optimal control solution and the resulting FE and compare these results to the EPA FE for a city route. Finally, we conclude our research in Chapter 4 and discuss further research.
CHAPTER 2

METHODOLOGY

We consider a problem of generating an Eco-Driving trajectory subject to real-world driving constraints of traffic lights and avoiding collisions with a lead vehicle. Since precise vehicle control is required, this has primary applications in autonomous or assisted control rather than human driver control.

2.1 Problem Statement and Assumptions

Several assumptions are made:

1. **Vehicle trajectory is to be specified as a function of time**
   Specifically, we assume that we are able to maintain precise control of the subject vehicle (ego vehicle) velocity on a second-by-second basis, which would be enabled by autonomous or assisted vehicle control.

2. **Minimizing the integral of the square of the acceleration over time increases energy efficiency**
   Previous research into Eco-Driving suggests that the Eco-Driving can be achieved without high-fidelity models of vehicle fuel consumption; instead, it is sufficient to minimize acceleration (23; 27; 28) and, whenever possible, pass through the traffic lights when they are green to avoid braking, stopping and idling. This approach is adopted in this paper as well.

3. **Piece-wise linear lead vehicle constraint**
   We further assume that the ego vehicle is following a lead vehicle which is represented here with a simple piecewise linear distance vs. time function.
4. Asynchronous SPaT along a straight drive cycle

We consider asynchronous traffic light signal phase and timing (SPaT) along a straight route during which the ego vehicle will stop for red lights.

Noting that the optimal solution to this problem would be driven towards extremely low speeds which is impractical for real world applications. To combat this tendency, we will impose an additional constraint on travel time to ensure that the vehicle is not a traffic hazard. These assumptions lead to an optimal control formulation which is also linked to a problem in approximation theory: the best interpolation in a strip.

2.2 Optimal Control Problem Formulation

![Figure 2.1: Visualization of SPaT data of 3 traffic lights where red dashes indicate when each light is red. Traffic lights are located approximately 1.9, 3.9, and 5.7 tenth of a mile.](image)

Figure 2.1: Visualization of SPaT data of 3 traffic lights where red dashes indicate when each light is red. Traffic lights are located approximately 1.9, 3.9, and 5.7 tenth of a mile.

To visualize how the optimal control problem is formulated, time is represented by the x-axis in minutes and distance by the y-axis in a tenth of mile increments. Three traffic lights are used to represent the red-light phase (Fig.2.1), where the first traffic light occurs at 0.19 miles and lasts for almost a minute. The lead vehicle trajectory is forecasted as the upper constraints, represented by the blue line (Fig.2.2). To assure that the ego vehicle always arrives at the traffic intersection before the light turn red, a lower constraint, represented by the pink line, is forecasted; it bounds the trajectory window to include time and location just before the traffic light turns red.
Figure 2.2: In (blue) trajectory of forecasted lead vehicle traveling at speed limit and stops at any red light it encounters. In (pink) the trajectory window just before the traffic light turns red. But not include red lights in the interior of the trajectory window (Fig. 2.2). It can be seen that the lead vehicle stopped at the first red light phase of the first traffic light, then stopped for almost the entire duration of the second red light phase of the second traffic light, and, finally, encountered the third traffic light in a green phase and thus drove through the intersection.

Figure 2.3: Visualization of the gates (double-ended arrows) that constrain the location of the ego vehicle.

Consider now the ego vehicle’s position vs time, $f(t)$. The location of the ego vehicle is represented by waypoints. These waypoints lie somewhere within the corresponding gates delin-
eled by orange double ended arrows (Fig. 2.3). These gate locations are set based on changes in boundary conditions (e.g. the lead vehicle transitions from zero velocity to some constant velocity). Where \( f(t) \) has to connect the waypoints which are designated by black circles in Fig. 2.4, the ego vehicle velocities at each waypoint are indicated by black lines. The waypoint locations, the vehicle velocities at the waypoints, and the trajectory “pieces” between the waypoints are determined by the theory of best interpolation in a strip.

Fig. 2.5 shows a visualization of the distance vs time problem geometry with SPaT data of three traffic lights, lead vehicle trajectory \( d(t) \), blue line and the trajectory window just before the traffic light turns red \( e(t) \), pink line). Note that the lower constraint, \( e(t) \), is determined based on trailing red light SPaT phases (in other words, the ego vehicle must pass through each intersection before the light turns red).

By assigning the orange markers as “gates”, we can now designate the interval between gates as the “strip”. Shown in the inset dashed green circle in Fig. 2.5 is a strip where the left boundary (the start time of the strip) is designated as \( t_i \) and the right boundary (the end time of the strip) is designated as \( t_{i+1} \). The ego vehicle position at certain time instant \( f(t_i) \), where \( i \) is the discrete time step index, is bounded above by \( d(t_i) \) and below by \( e(t_i) \). Overall, the ego vehicle trajectory has to adhere to the following constraints:
1. $f(t) \leq d(t)$, where $d(t)$ is informed by the forecasted lead vehicle position offset by safety margin;

2. $f(t) \geq e(t)$, where $e(t)$ is informed by the transition of the traffic signal phase to red.

3. Left: Start of the time interval $\rightarrow t_i$

4. Right: End of the time interval $\rightarrow t_{i+1}$

Note that constraints $e(t) \leq f(t) \leq d(t)$ form a strip that has two gates in distance vs. time that the ego vehicle must pass through similar to the way a slalom skier passes through gates during a competition. However, unlike a slalom skier, who pursues a minimum time trajectory, we seek to minimize the $L_2$ norm of the ego vehicle acceleration trajectory which in turn improves energy efficiency.

The constraints $e(t)$ and $d(t)$ are assumed to be piecewise linear; they are linear functions of time in each time interval $[t_i, t_{i+1}]$, where the time intervals are fixed. There are a few caveats
regarding this. Firstly, for real-world lead vehicles, this corresponds to an approximated representation. The theory of best interpolation in a strip does allow for piecewise cubic constraints which could be incorporated in future research. Secondly, the continuity of $e(t)$ is enforced by not allowing infinite slope in the $e(t)$ function as can be seen in Fig. 2.5. Lastly, it should also be mentioned that realistically, the $d(t)$ constraint should include a distance margin from a lead vehicle.

To demonstrate the concept of this research, we will use the routes shown in Figs. 2.6 and 2.7 with their associated real-world red light locations and speed limits. We will designate the Fig. 2.6 route as Kalamazoo Arterial (KA) and the Fig. 2.7 route as Kalamazoo Downtown (KD).

For these routes, real world vehicle velocity was recorded along two different roads in Kalamazoo, MI: one along Stadium drive to serve as the arterial route and one along Kalamazoo Mall to serve as the downtown route. A consistent SPaT cycle of 120 seconds was used for this demonstration since it represents a broad use case (29; 30). This SPaT data then informs specific piecewise linear upper and lower constraints for the optimal control problem. We note that the lead vehicle constraints in this paper are defined by speed limits and converted to piecewise linear functions. Based on results in (25; 8; 26), piecewise cubic constraints can also be handled, but this is left to future work.

![Figure 2.6: Details of the Kalamazoo Arterial route which is used to collect lead vehicle and SPaT data constraints.](image)

In the above and subsequent developments we assumed that there is either no vehicle trailing the ego vehicle, or such a trailing vehicle does not influence the ego vehicle trajectory. In principle, this assumption could be relaxed, e.g., the lower constraint on the ego vehicle trajectory
can be additionally informed by a trailing vehicle trajectory in order to minimize disruption to its motion.

2.3 Optimal Control Solution

The eco-driving problem setup exploits the connection with the problem of best interpolation in a strip (25; 8; 26). In particular, the planning of vehicle distance trajectory versus time, $f(t)$, $t_0 \leq t \leq t_N$, can be posed as the following problem:

Given time instants $t_0 < t_1 \cdots < t_N$, initial and final slopes $s_0$ and $s_N$, lower and upper bounding functions $e$ and $d$ satisfying $e(t) < d(t)$ for all $t \in [t_0,t_N]$ which are continuous on $[t_0,t_N]$, and linear on $[t_i,t_{i+1}]$, determine the function $f$ that satisfies the following optimization problem:

\[
\text{Minimize} \quad \|f''\|_{L^2_{[t_0,t_N]}} \\
\text{subject to} \\
e(t) \leq f(t) \leq d(t), \quad t_0 \leq t \leq t_N, \\
\dot{f}(0) = s_0, \quad \dot{f}(N) = s_N.
\] (2.1)

The computational approach to finding such a function $f$ then decomposes into an outer
and an inner optimization problems. The outer loop problem is to minimize the function

$$\phi(y, s) = \sum_{i=0}^{N-1} \phi_i(y_i, s_i, y_{i+1}, s_{i+1}),$$  \hspace{1cm} (2.2)

where \( y = (y_0, \cdots, y_N) \) and \( s = (s_0, \cdots, s_N) \) are vectors of function values and slopes at \( t_i \) where \( y_0, y_N, s_0 \) are predetermined constants, \( s_N = 0 \) indicating the ego vehicle comes to a full stop at the end of the trip, \( i = 0, \cdots, N \), and \( \phi_i(y_i, s_i, y_{i+1}, s_{i+1}), i = 0, \cdots, N-1 \) are computed by solving \( N \) independent inner optimization problems:

\[
\begin{align*}
\text{Minimize} & \quad \|f''\|_{L^2_{[t_i,t_{i+1}]}} \\
\text{subject to} & \quad f(t_i) = y_i, \quad f(t_{i+1}) = y_{i+1} \\
 & \quad \dot{f}(t_i) = s_i, \quad \dot{f}(t_{i+1}) = s_{i+1} \\
 & \quad e(t) \leq f(t) \leq d(t), \quad t_i \leq t \leq t_{i+1}.
\end{align*}
\]  \hspace{1cm} (2.3)

The function \( \phi \) can be shown to be a convex and coercive function of the arguments \( y \) and \( s \), and it has a unique minimum (25; 8; 26). The solution to the inner loop optimization problem (2.3) is a \( C^2 \) cubic spline that is one of the following types determined by its interactions with the lower and upper constraining functions \( e(t) \) and \( d(t) \), \( t_i \leq t \leq t_{i+1} \):

**Case 1:** Constraints are inactive;

**Case 2:** Single touching point on the lower constraint \( e \);

**Case 3:** Single touching point on the upper constraint \( d \);

**Case 4:** Single subarc on the lower constraint \( e \);

**Case 5:** Single subarc on the upper constraint \( d \);

**Case 6:** Touching pair with the first touching point on the lower constraint \( e \) and second touching point on the upper constraint \( d \);
Case 7: Touching pair with the first touching point on the upper constraint \(d\) and second touching point on the lower constraint \(e\).

Note that when \(y_i, y_{i+1}, s_i\) and \(s_{i+1}\) are given, the optimal trajectory of (2.3) is unique and corresponds to one of the cases 1-7. In order to find this optimal trajectory we evaluate the cost for each of the cases for which a feasible solution exists and select the case corresponding to the minimal value. The construction of these solutions involves solving a system of linear algebraic equations for the coefficients of a piecewise cubic polynomial. In that sense problem (2.3) is solved as a combinatorial optimization problem.

The candidate solutions of the 7 cases are in the form of one of the below 4 different systems of equations. In some cases knots are added, which are determined by the points of intersection of \(f(t)\) and the upper or lower bounds. Mathematically, a knot represents the transition between cubics.

Case 1: the constraints are not active (no knots added);

\[
f_1(t) = k_1(t - t_i)^3 + k_2(t - t_i)^2 + k_3(t - t_i) + k_4
\]  

(2.4)

In case 1, Fig. 2.8, there is no active constraints where there are no additional knots other than the fixed knots \([t_i, t_{i+1}]\). The optimal solution is \(f(t)\) which is the vehicles trajectory between the lower, \(e(t)\), and upper, \(d(t)\), constraints.

Figure 2.8: No active constraints definition, where \(a = t_i\) and \(b = t_{i+1}\).
**Case 2 & 3:** a single touching point on one of the constraints (one additional knot);

\[
f_2(t) = \begin{cases} 
  k_1(\delta - t)^3 + k_2(\delta - t)^2 + k_3(\delta - t) + k_4 \\
  & t \in [t_i, \delta] \\
  m_1(-\delta + t)^3 + m_2(-\delta + t)^2 + m_3(-\delta + t) + m_4 \\
  & t \in [\delta, t_{i+1}] 
\end{cases}
\tag{2.5}
\]

The case, Fig. 2.9, with one the Eco-Vehicle, \(f(t)\), hitting the upper, \(d(t)\), or lower, \(e(t)\), constraint. Reaching this case would cause the Eco-Vehicle to decelerate and accelerate over a given amount of time to maintain optimal solution.

![Figure 2.9: Single touching point definition, where \(a = t_i\) and \(b = t_{i+1}\).](image)

(a) single touching point on lower constraint  (b) single touching point on upper constraint

**Case 4 & 5:** a sub-arc on one of the constraints (two additional knots);

\[
f_3(t) = \begin{cases} 
  k_1(\delta_1 - t)^3 + k_2(\delta_1 - t)^2 + k_3(\delta_1 - t) + k_4 \\
  & t \in [t_i, \delta_1] \\
  e(t) \\
  & t \in [\delta_1, \delta_2] \\
  m_1(-\delta_2 + t)^3 + m_2(-\delta_2 + t)^2 + m_3(-\delta_2 + t) + m_4 \\
  & t \in [\delta_2, t_{i+1}] 
\end{cases}
\tag{2.6}
\]
These cases, Fig. 2.10, cause a sub-arc to touch the upper, $d(t)$ or lower, $e(t)$, limit. This trajectory can simulate a vehicle following the lead vehicle in traffic or possibly the speed limit while not speeding to maintain optimal FE in order to avoid an upcoming stop light in traffic. This must happen to make sure the $v > 0$ to avoid stop lights. In our test it is unlikely that the solution will contact the lower constraint $e(t)$ to keep momentum in the vehicles to minimize the change in velocity.

**Case 6 & 7**: a touching pair (two additional knots):

\[
\begin{align*}
    f_4(t) &= \begin{cases} 
    k_1(\delta_1 - t)^3 + k_2(\delta_1 - t)^2 \\
    + k_3(\delta_1 - t) + k_4 \\
    &+ m_1(-\delta_1 + t)^3 + m_2(-\delta_1 + t)^2 \\
    &+ m_3(-\delta_1 + t) + m_4x \\
    &+ g_1(-\delta_2 + t)^3 + g_2(-\delta_2 + t)^2 \\
    &+ g_3(-\delta_2 + t) + g_4 \\
    t &\in [\delta_1, \delta_2] \\
    &\in [\delta, t_{i+1}] 
    \end{cases} \tag{2.7}
\end{align*}
\]

This case of two touching points, Fig. 2.11, that adds two additional knots, $\delta_1$ and $\delta_2$, in the interval. This is similar to cases 1 and 2. This can simulate bumper to bumper traffic while maintaining $v > 0$. Having this oscillation would be the least optimal case due to the drastic changes in velocity to maintain within the upper and lower constraints, $d(t)$ and $e(t)$. Practically, the constraint difference could be very small, that the variation in the case will also be small.
As an example, the candidate solution in case 4 (a single subarc on the lower constraint) is in the form of (2.6) where the subarc on the lower constraint occurs for $t \in [\delta_1, \delta_2]$. In this case all coefficients $k_1, k_2, k_3, k_4, m_1, m_2, m_3, m_4$ and parameters $\delta_1$ and $\delta_2$ can be computed by solving linear algebraic equations only. The function (2.2) is computed by adding up the minimum costs from each interval.

2.4 FE Evaluation

This optimal solution technique is implemented in Matlab and is used to compute the optimal Eco-Driving control for the Kalamazoo Arterial and the Kalamazoo Downtown drive cycles. Once the optimal trajectory of the ego vehicle has been determined, the FE is evaluated using established and real-world validated vehicle modeling software. For this study, we chose to use the FASTSim software which is freely available from the National Renewable Energy Laboratory (31).

We are interested in quantifying the results of this technique for several different types of vehicle architecture including Hybrid Electric Vehicles (HEVs), Conventional Vehicles (CVs), and Electric Vehicles (EVs). To meet these criteria, we chose a 2016 Toyota Highlander Hybrid, 2016 Toyota Camry 4 cylinder, and 2016 Tesla Model S60. We have used the Python version of FASTSim for greater flexibility. It uses the drive cycle as an input and provides FE as an output.

The optimal trajectory result will be subsequently shown in distance (mi) versus time (min). In order to apply FASTSim as our validation tool, the second-by-second vehicle velocity time history was inferred from the optimal position trajectory. The road grade was set to zero.
CHAPTER 3

RESULTS

The results of this study are the determined upper and lower constraints, the derived Eco-
Driving drive cycle, and the associated FE for the three vehicle models for the Kalamazoo Arterial
and the Kalamazoo Downtown routes.

3.1 Optimal Control Solution

The constraints and Eco-Driving drive cycles for the two routes are shown in Figs. 3.1 and 3.2. The lower constraint (red) is the time window just before the traffic turns red and the upper constraint (blue) is the lead vehicle. The optimized trajectory is the green spline which in each time interval, can fall into the seven cases described in section 2.3. Note that Figs. 4 and 5 show two different drive cycles for each route. This is because the initial condition of the ego vehicle route is somewhat arbitrary. For both routes, drive cycle 1 (subfigure a) has an initial slope (velocity of the ego vehicle) determined by the optimizer, whereas drive cycle 2 (subfigure b) has an initial slope chosen to be half the slope of the upper constraint (lead vehicle velocity). This choice is meant to provide distinct results and insights into the optimal solution.

Drive cycles KA1 and KA2 cover a total distance of 3.31 miles in 7.5 minutes. Fig. 4a shows KA1 where the initial slope was chosen by the optimizer. The KA1 trajectory only uses two of the seven possible cases for the optimal solution: case 1 and case 3. The single touching point is located towards the end of the drive cycle at \( t = 5.85 \) and 6.81 minutes which is the only additional gate other than the 12 fixed gates located at \( t = 0, 0.96, 1.96, 2.97, 3.4, 3.96, 4.4, 5.4, 5.8, 6.8, 7, \) and 7.44 minutes. The KA2 trajectory also only shows case 1 and case 3. The touching pair is located towards the end of the drive cycle at \( t = 6.84 \) and 7 minutes which is the only 2 additional gates other than the 12 fixed gates. Comparing the drive cycle 1 and 2, both trajectories
Figure 3.1: Kalamazoo Arterial route results for the Eco-Driving solution with an initial velocity chosen by the optimizer, drive cycle KA1 (a), and an initial velocity assigned as half of the speed limit, drive cycle KA2 (b).

were approximately similar where both had case 1 and case 3 for the possible optimal solution.

Drive cycles KD1 and KD2 cover a total distance of 0.9 miles in 5.9 minutes. Overall this route provides tighter upper and lower constraints for the vehicles trajectory to follow. KD1 again only shows case 1 and case 3. The single touching point is located towards the end of the drive cycle at \( t = 4.4 \) and 4.6 minutes which is the only additional gate other than the 14 fixed gates located at \( t = 0, 0.2, 0.5, 1.3, 1.5, 1.9, 2.6, 2.9, 3, 3.6, 4.1, 4.4, 4.6, \) and 5.9 minutes. The KD2 trajectory slightly exceeds the speed limit to make the stop light at \( t = 1.9 \) minutes. KD2 also has three cases: case 1, case 2, and case 3. The touching pair is located towards the end of the drive cycle at \( t = 1.3 \) and 4.6 minutes which are the only 2 additional gates other than the 14 fixed gates.
3.2 FE Results

MPGe (miles per gallon gasoline equivalent) and MPG are calculated for all derived drive cycles according to the standard (J1711-201006) (32). Each result is then contrasted with EPA FE measurements for city driving (33) in Table 3.1. Note that the results vary widely depending on the vehicle model used.

Table 3.1 shows a significant energy efficiency increase for all vehicles in both routes when compared to EPA city estimates. Additionally, there is a significant difference in the energy efficiency improvements for the arterial roads vs. downtown. This is because the main-arterial roads are known to have a better FE since they have greater distances between stop lights and their speed limit is higher. Downtown routes have closely spaced traffic lights with lots of acceleration and deceleration. Both the Toyota Camry and the Tesla Model S show FE increases in drive cycle 1 over
Table 3.1: Energy efficiency (i.e. FE) of all drive cycles for an HEV, EV, and CV.

<table>
<thead>
<tr>
<th>3*Drive Cycle</th>
<th>2016 Toyota Highlander Hybrid (HEV) MPG</th>
<th>2016 Tesla Model S60 (EV) MPGe</th>
<th>2016 Toyota Camry 4cyl (CV) MPG</th>
</tr>
</thead>
<tbody>
<tr>
<td>2*Main Arterial Drive-cycle 1</td>
<td>65.5</td>
<td>220.5</td>
<td>48.6</td>
</tr>
<tr>
<td>Drive-cycle 2</td>
<td>65.8</td>
<td>218.5</td>
<td>45.5</td>
</tr>
<tr>
<td>2*Downtown Drive-cycle 1</td>
<td>41.4</td>
<td>243.7</td>
<td>30.6</td>
</tr>
<tr>
<td>Drive-cycle 2</td>
<td>45.0</td>
<td>238.6</td>
<td>30.0</td>
</tr>
<tr>
<td>EPA FE (city)</td>
<td>27</td>
<td>98</td>
<td>28</td>
</tr>
</tbody>
</table>

drive cycle 2 in both routes. The Toyota Highlander Hybrid on the other hand shows a significant increase in drive cycle 2 over drive cycle 1 in both routes.
CHAPTER 4
CONCLUSION

This thesis proposed an approach for increasing the energy efficiency of eco-driving vehicles based on the best interpolation in a strip. Two real-world routes were chosen: Kalamazoo Arterial and Kalamazoo Downtown. These routes were converted to distance over time constraints for an optimal control problem to be solvable using the best interpolation in a strip. The optimization problem was programmed into MATLAB software and the MATLAB fmincon tool was used to produce optimized routes for both Kalamazoo Arterial and Kalamazoo Downtown.

Two drive cycles were tested, one drive cycle utilized $s_0$ equal to half the velocity of the lead vehicle, and the other utilizing the optimizer to determine $s_0$. The Eco-Driving solution was used as an input to the real-world validated FASTSim models that were used to simulate the performance of an HEV, CV, and EV. FASTSim results were compared to the EPA reported FE for city driving Results. There was a significant increase in $MPG$ and $MPGe_{electric}$ in all drive cycles showing a significant improvement compared to EPA. This increase can yield to future advancement of automotive and allows for improvements in the field of advanced driving-assistant technologies.

Additionally, the requirement that $e$ and $d$ are piecewise linear can be relaxed to $e$ and $d$ being piecewise cubic but the computations become more involved. While an alternative brute force approach to (2.1) involving time discretization and conversion to a quadratic program is also possible, the disadvantages of that route include higher dimensionality of the optimization problem and the potential for “inter-sample” constraint violations (e.g., collisions with lead vehicle or violation of traffic light rules in extreme cases).

Eco-Driving is an important technology to realize sustainable transportation. The method presented here can be used in all levels of vehicle automation provided surrounding vehicle and SPaT data is available. Moreover, Eco-Driving are not only entitled to the improvement of automotive industry technologies but it can reduces the cost of transportation for many industries that
are heavily based on transportation, and help in the reduction of green house gas emissions.

Future work includes measuring the human driver FE of a vehicle along this route, and using real-world measured SPaT data specific to these routes. Utilizing the real world SPaT data and a human driver will help build a high-fidelity model where we can get more accurate results of our application when applied in day-to-day driving activities.

In real vehicles, there are limits on acceleration and velocity. For example, We know that in real world instantaneous acceleration is tied to the vehicle structure and build and its limited by the RpM; vehicles can speed up or slow down only so much in such a short period of time. In our theoretical solution, the acceleration and velocity seems to comply with the speed limit of the routes and it seems that a vehicle can implement the acceleration. For further research, We would like to thoroughly investigate the values of instantaneous acceleration and velocity to be expected in the solution.

Furthermore, we would like to expand beyond the Kalamazoo routes. To be able to expand we will need to perform testing in other states or use simulation tools like FASTsim or the Chassis Dynamometer located in the EEA V lab on different cycles. This will allow us to analyze the different behaviors a driver might encounter on daily basis in addition that it will expand the scope of our research to be applied in the automotive industry.

Last but not least we would like to expand the optimization problem formulation to allow for cubic constraints. Cubic constraints will simulate a better behavior of the (lead vehicle) upper constraint and lower constraint. In real life, we do not anticipate that drivers (in this case, upper and lower constraints) maintain constant speed all the time (linear constraints).


APPENDIX

Matlab Code
strip optimization main

Input: csv file with 6 columns of a,b, lower m, lower b, upper m, upper b

Must set Y and S boundary values at endpoints of interval when not an optimization parameter

% housekeeping
close all;
clear;

% setup
csvfname = 'constraintsdatacleanedformatlab.csv';
Ybv_t0 = 0; % y value at the start time
Ybv_tf = 0.9; % y value at the final time
Sbv_tf = 0; % dy/dt (slope) at the final time

% read and process data
[knots,id,lb,n,u,ub] = ecodriving_initialize(csvfname);
plot_constraints

% modify lb and ub for slope as optimization value at t_a, first time point
lb= [lb,l.cf(1,3)];
ub= [ub,u.cf(1,3)];

% boundary values
Ybv= [Ybv_t0,Ybv_tf]; % position at initial and final point
Sbv= Sbv_tf; % slope at final point
% slope at final time set to zero for zero velocity

% initial guess
S0= zeros(1,n-2);
Y0= (lb(end)+ub(end))/2;
s0= (lb(id.SY2Y)+ub(id.SY2Y))/2;
YSs0 = [S0,Y0,s0];

% additional parameters
toll= 1e-12;

% optimization
opt_fmincon= optimset;
opt_fmincon.Algorithm= 'sqp';
opt_fmincon.Display= 'iter';
opt_fmincon.MaxFunEvals= 5000;

% optimization
[YSs,fval,flg_exit,info] = fmincon( ...
 @(YSs) cost_splines_s0(YSs,id,knots,l,n,u,Sbv,toll,Ybv), ...
 YSs0,[],[],[],[],lb,ub,[],opt_fmincon);

% get data
[cost, cons, splines] = cost_splines_s0(YSs,id,knots,l,n,u,Sbv,toll,Ybv);

% plot
plot_splines;

function [knots,id,lb,n,u,ub] = ecodriving_initialize(csvfname)
%
extract information for strip optimization

id: indexing to unwrap S and Y during optimization
lb: lower bound on Y, prevents solver from violating constraints at knots
ub: upper bound on Y, prevents solver from violating constraints at knots

% check file name

27
if (~endsWith(csvfname,'.csv','IgnoreCase',true))
    csvfname= [csvfname,'.csv'];
end

% load data
csvdata = csvread(csvfname);
% plot_csvdata(csvdata)

% number of intervals
n_int= numel(csvdata(:,1));
n= n_int+1;

% knots
knots= [csvdata(:,1);csvdata(end,2)];

% initialize
l= struct;
u= struct;
l.cf= zeros(n_int,4);
u.cf= zeros(n_int,4);

% strip bound equations are defined relative to the last point
% shift data
l.cf(1,3:4)= csvdata(1,3:4);
u.cf(1,3:4)= csvdata(1,5:6);
for ii=2:n_int
    b_l= polyval(csvdata(ii-1,3:4),knots(ii));
    m_l= csvdata(ii,3);
    b_u= polyval(csvdata(ii-1,5:6),knots(ii));
    m_u= csvdata(ii,5);
    l.cf(ii,3:4)= [m_l, b_l];
    u.cf(ii,3:4)= [m_u, b_u];
end

% index and numbers
id= struct;
id.SY2S= 1:(n-2);
id.SY2Y= (n-2+1):(2*(n-2));

% bounds
lb= Inf(1,2*(n-2));
ub= Inf(1,2*(n-2));

% set Y bounds
for ii=2:(n-1)
    % lower
    tmp1= polyval(l.cf(ii-1,:),knots(ii)-knots(ii-1));
    tmp2= polyval(l.cf(ii,:),0);
    lb(id.SY2Y(ii-1))= max(tmp1,tmp2);
    % upper
    tmp1= polyval(u.cf(ii-1,:),knots(ii)-knots(ii-1));
    tmp2= polyval(u.cf(ii,:),0);
    ub(id.SY2Y(ii-1))= min(tmp1,tmp2);
end

end

function [cost, cons, splines]=cost_splines_s0(SYs,id,knots,l,n,u,Sbv,toll,Ybv)
% Sbv, Ybv: S and Y bounday values at a and b
% l,u: lower and upper constraints (use l.cf(i,:), u.cf(i,:) to get coefficients for the ith interval)
% Sbv only includes final slope
% unwrap S and Y
S= SYs(id.SY2S);
Y= SYs(id.SY2Y);
% unwrap s0
s0= SYs(end);

% total position and slope values
S= [s0,S,Sbv];
Y= [Ybv(1),Y,Ybv(end)];

% spline calculation
[cost, cons, splines]= calc_splines(knots,l,n,u,S,toll,Y);
end

function [cost, cons, splines]=calc_splines(knots,l,n,u,S,toll,Y)
% all spline data given knots and slopes

% initial values
cost = 0;
cons= -Inf;

% storage
splines= struct;
splines.X= cell(1,n-1);
splines.cf= cell(1,n-1);
splines.case= cell(1,n-1);

% step through intervals
for ii=1:n-1

% there is a total of n-1 intervals
a= knots(ii);
b= knots(ii+1);
c= Y(ii);
d= Y(ii+1);
s1= S(ii);
s2= S(ii+1);
li= l.cf(ii,:);
ui= u.cf(ii,:);

[inc_cost1, inc_cons1, cf1] = case1(a,b,c,d,li,ui,s1,s2,toll);
if (inc_cons1<=toll)
  % unconstrained cubic spline is the solution
  splines.X{ii}= [a,b];
splines.cf{ii}= cf1(:)';
splines.case{ii}= 1;
cost= cost+inc_cost1;
  cons= max(cons,inc_cons1);
else
  cands=[];
  [inc_cost2l, inc_cons2l, del2l, k2l, m2l]=case2l(a,b,c,d,li,ui,s1,s2,toll);
  if (inc_cons2l<=toll)
    cands=[cands;21,inc_cost2l];
  end

  [inc_cost2u, inc_cons2u, del2u, k2u, m2u]=case2u(a,b,c,d,li,ui,s1,s2,toll);
  if (inc_cons2u<=toll)
    cands=[cands;22,inc_cost2u];
  end

  [inc_cost3l, inc_cons3l, del3l, k3l, m3l, llr3]=case3l(a,b,c,d,li,ui,s1,s2,toll);
  if (inc_cons3l<=toll)
    cands=[cands;31,inc_cost3l];
  end

  [inc_cost3u, inc_cons3u, del3u, k3u, m3u, uur3]=case3u(a,b,c,d,li,ui,s1,s2,toll);
  if (inc_cons3u<=toll)
    cands=[cands;32,inc_cost3u];
end
[inc_cost4lu, inc_cons4lu, del4lu, k4lu, m4lu, f4lu] = case4lu(a, b, c, d, li, ui, s1, s2, toll);
if (inc_cons4lu <= toll)
    cands = [cands; 41, inc_cost4lu];
end

[inc_cost4ul, inc_cons4ul, del4ul, k4ul, m4ul, f4ul] = case4ul(a, b, c, d, li, ui, s1, s2, toll);
if (inc_cons4ul <= toll)
    cands = [cands; 42, inc_cost4ul];
end
if isempty(cands)
    % no feasible solution exists (abnormal case)
    % use unconstrained cubic spline solution
    splines.X(ii) = [a, b];
    splines.cf(ii) = cf1(:, ');
    splines.case(ii) = -1;
    cost = cost + inc_cost1;
    cons = max(cons, inc_cons1);
else
    [vmin, imin] = min(cands(:, 2)); % select minimum cost solution from available options
    if (cands(imin, 1) == 21)
        splines.X(ii) = [a, del2l, b];
        splines.cf(ii) = [k2l(:, ''); m2l(:, ')];
        splines.case(ii) = 21;
        cost = cost + inc_cost2l;
        cons = max(cons, inc_cons2l);
    elseif (cands(imin, 1) == 22)
        splines.X(ii) = [a, del2u, b];
        splines.cf(ii) = [k2u(:, ''); m2u(:, ')];
        splines.case(ii) = 22;
        cost = cost + inc_cost2u;
        cons = max(cons, inc_cons2u);
    elseif (cands(imin, 1) == 31)
        splines.X(ii) = [a, del3l, b];
        splines.cf(ii) = [k3l(:, ''); l1r3(:, ''); m3l(:, ')];
        splines.case(ii) = 31;
        cost = cost + inc_cost3l;
        cons = max(cons, inc_cons3l);
    elseif (cands(imin, 1) == 32)
        splines.X(ii) = [a, del3u, b];
        splines.cf(ii) = [k3u(:, ''); uur3(:, ''); m3u(:, ')];
        splines.case(ii) = 32;
        cost = cost + inc_cost3u;
        cons = max(cons, inc_cons3u);
    elseif (cands(imin, 1) == 41)
        splines.X(ii) = [a, del4lu, b];
        splines.cf(ii) = [k4lu(:, ''); m4lu(:, ''); f4lu(:, ')];
        splines.case(ii) = 41;
        cost = cost + inc_cost4lu;
        cons = max(cons, inc_cons4lu);
    elseif (cands(imin, 1) == 42)
        splines.X(ii) = [a, del4ul, b];
        splines.cf(ii) = [k4ul(:, ''); m4ul(:, ''); f4ul(:, ')];
        splines.case(ii) = 42;
        cost = cost + inc_cost4ul;
        cons = max(cons, inc_cons4ul);
    end
end
function [cost, cons, splines]=cost_splines(SY,id,knots,l,n,u,Sbv,toll,Ybv)
% Sbv, Ybv: S and Y boundary values at a and b
% l,u: lower and upper constraints (use l.cf(i,:), u.cf(i,:) to get coefficients for the ith
% interval)
% unwrap S and Y
S= SY(id, SY2S);
Y= SY(id, SY2Y);
% total position and slope values
S= [Sbv(1),S,Sbv(end)];
Y= [Ybv(1),Y,Ybv(end)];
% spline calculation
[cost, cons, splines]= calc_splines(knots,l,n,u,S,toll,Y);
end
%
% plot the lower and upper bounds and the 3rd order splines
% execute in optimization results workspace
%
% setup
figure;
hold on;
% lower and upper constraints
for ii =1:(n -1)
% lower
plot(knots(ii:ii+1),polyval(l.cf(ii,:),[0,knots(ii+1)-knots(ii)]),'r--');
% upper
plot(knots(ii:ii+1),polyval(u.cf(ii,:),[0,knots(ii+1)-knots(ii)]),'b--');
end
slope = struct;
% spline segments
for ii=1:(n-1)
% number of segments in the interval
n_seg= numel(splines.cf{ii}(:,1));
for jj=1:n_seg
tmp1= linspace(splines.X{ii}(jj),splines.X{ii}(jj+1),20);
plot(tmp1,polyval(splines.cf{ii}(jj,:),tmp1-splines.X{ii}(jj)),'g-');
if jj < 20
% slope = (polyval(splines.cf{ii}(jj+1,:),tmp1-splines.X{ii}(jj+1))-(polyval(splines.cf{ii}(jj,:),tmp1-splines.X{ii}(jj)))/(tmp1-splines.X{ii}(jj+1)-tmp1-splines.X{ii}(jj));
fprintf();
end
end
end
% extracting the xy data from the plot
h = findobj(gca,'Type','line');
x= get(h,'Xdata');
y= get(h,'Ydata');
% Storing the data in xyarr, the for loop doesn't add the last point data

iii = 1;

% x{13,1}(20)

for i = 13:-1:1

    for ii = 1:1:19

        xyarr(iii,1) = x{i,1}(ii);
        xyarr(iii,2) = y{i,1}(ii);
        iii = iii +1;

    end

end

xyarr(iii,1) = x{1,1}(20);
xyarr(iii,2) = y{1,1}(20);

% Finding the slope (velocity) and storing it in the 3rd col of xyarr

for i = 1:1:iii

    if i < iii

        xyarr(i,3) = (xyarr(i+1,2)-xyarr(i,2))/(xyarr(i+1,1)-xyarr(i,1));
        xyarr(i,4) = xyarr(i,3)*60;

    end

end

% Finding the acceleration and storing it in the 5th col of xyarr

for i = 1:1:iii

    if i < iii

        xyarr(i,5) = (xyarr(i+1,3)-xyarr(i,3))/(xyarr(i+1,1)-xyarr(i,1));
        xyarr(i,6) = xyarr(i,5)*3600;

    end

end

xlswrite('downtownresults.xls',xyarr);
writematrix(xyarr,'resultsdowntown.csv');

% Complete

hold off;
grid on;
box on;
ylabel('Distance');
xlabel('Time');