Speed Control for Low-Energy On-Time Buses

Richard Teck Ken Wong
Somerville College

Department of Engineering Science
University of Oxford

Thesis submitted for the degree of
Doctor of Philosophy
Hilary 2017

Declaration

The work presented in this dissertation was carried out between November 2012 and March 2017 under the supervision of Prof. Malcolm D. McCulloch.

The author wishes to declare that, except for commonly understood and accepted ideas or where reference is made to the work of others, the work in this thesis is his own, and includes nothing which is the outcome of work done in collaboration. It has not been submitted in part, or in whole, to any other university for a degree, diploma or other qualification.

Richard T. K. Wong
Somerville College
Hilary 2017

Acknowledgements

First of all, I would like to take this opportunity to thank my supervisor, Prof. Malcolm McCulloch for his guidance and support throughout the years.

I would like to show my gratitude to the current and previous members in Energy and Power Group for their friendship and support.

I want to express my appreciation to the Oxford Bus Company for providing the data of the bus speed to make some sections of this thesis possible.

I am deeply grateful to the Public Service Department of Malaysia for the financial support during the four years of my DPhil study.

I would also like to grab this opportunity to thank all my friends who have made my life in Oxford enjoyable. I would like to show my appreciation especially to a few who have been supporting me emotionally during my time of difficulties throughout the years. They are Keng Wai Chan, Joshua Bell, Timothy and Danica Kuiper, Andi Wang, and Abraham Ng. Also to Andi and Abraham for their help with proofreading. My appreciation extends to other friends whose names are not listed here.

I am indebted to my family, my father, Chi Sie Wong, my mother, Mei Ho Lee, and my sisters, Doris Wong, Linda Wong, and Tracy Wong, for being patient, considerate, and supportive of me throughout these years.

Last but foremost, I would like to thank God for His grace for the opportunity to start this DPhil project and sustaining me through all the ups and downs. May the praise and glory be to God our Lord and Jesus Christ our saviour.

For the Lord gives wisdom;
from his mouth come knowledge and understanding;

Proverbs 2:6

Richard T. K. Wong
April 2017

Abstract

In 2015, 40% of the total energy used by final users in the United Kingdom was used in the transport sector. Hence it is important to minimise the energy consumption in the transport sector. This research focuses on the energy consumption minimisation of a bus using the approach of optimal control.

The four objectives defined for this research are (i) the setting up of an optimal control problem for a bus journey, (ii) the formulation of a real-world bus route for open loop optimisation, (iii) the estimation of on-road bus speed for open loop optimisation, and (iv) the setting up of a real-time system to provide and update the optimal speed.

An optimal control problem is set up to minimise the energy consumption and the discrepancy between the arrival time and the scheduled time at bus stops. The formulation of on-demand bus stops, compulsory bus stops, and slope changes is discussed. On-demand bus stops are formulated in the bost function, compulsory bus stops are formulated using multiphase approach, and the slope changes are interpolated using a smooth linear interpolation method. PSOPT, an open source optimal control program which implements the direct collocation method is used to solve the optimal control problem. The optimal profile of a basic route is compared with the profile resulted from an aggressive driving style. The optimal profile is found to consume 60% less energy than the aggressive driving profile.

Two aspects are studied to incorporate a real-world bus route into the open loop optimisation process. The first aspect is the location-related aspect addressed by objective (ii). Methods to translate geodesic coordinates of a bus route and the bus stops are explored. The procedure to identify the stopping probabilities at bus stops is also discussed. The second aspect is the traffic-related aspect addressed by objective (iii). On-road traffic affects the ability of the bus to be optimally driven, therefore neural networks are designed to estimate the bus speed on both the mixed traffic lanes and the bus lanes.

A real-time on-board system is set up to provide and update the optimal speed on the bus. The system acquires real-time information of the bus and adapts the optimal profile using a proposed adjustment algorithm when there are disruptions to the optimal driving.

The adjustment algorithm found to be able to provide a near-optimal result with only 0.23% extra energy consumption compared to the optimal result with shorter execution time.

This thesis uses a real-world bus route from Oxford City Centre to Pear Tree Park and Ride for case studies in different sections.

Nomenclature

α	Angular acceleration (rad/s^2)
α_A	Equivalent constant value for $A(t)$
α_v	Intermediate variable in Vincenty formula
β	Location of bus stop or location of bus to be fitted on a bus route
β_i'	Alternative position of bus
Δ	Stopping period at bus stop (s)
δ	Time tolerance of bus arrival (s)
δ_ι	Angular distance between two points
Δ_c	Stopping time period at compulsory bus stop (s)
δ_c	Arrival time tolerance at compulsory bus stop (s)
δ_d	Arrival time tolerance at on-demand bus stop (s)
η_{wa}	Wheel-axle efficiency
$\int_0^t i\,dt$	Actual battery charge removed (Ah)
λ	longitude of GPS point in radians
λ°	longitude in degrees
ω	Angular speed
ω_{gb}	Gearbox angular speed (rad/s)
ω_{mt}	Motor angular speed (rad/s)
ω_{wa}	Wheel-axle angular speed (rad/s)

ϕ	latitude of GPS point in radians
ϕ°	latitude in degrees
ϕ_{mt}	Field flux in armature winding (Wb)
ρ	Air density (kg/m^3)
σ	Intermediate variable in Vincenty formula
σ_m	Intermediate variable in Vincenty formula
σ_u	Unbiased standard deviation
τ	Torque
τ_D	Aerodynamic drag (Nm)
τ_g	Gravitational torque (Nm)
τ_r	Rolling friction (Nm)
τ_v	Vehicle torque (Nm)
τ_{GB}	Gearbox torque (Nm)
τ_{iner}	Rolling inertia (Nm)
τ_{maxgb}	Gearbox maximum torque (Nm)
τ_{mt}	Motor torque (Nm)
τ_{wa}	Wheel-axle torque (Nm)
θ	Road slope ($^\circ$)
θ_i	Instantaneous road slope ($^\circ$)
A	Exponential zone amplitude (V)
a	Constant in smooth heaviside function
$A(t)$	Coefficient to form adjusted profile
$a(t)$	Acceleration profile (m/s^2)
a_ι	Intermediate variable in Slerp-based interpolation
a_e	Major semiaxis of ellipsoid

A_f Frontal area (m^2)

a_g Intermediate variable in great circle distance calculation

a_L Lower acceleration bound (m/s^2)

a_U Upper acceleration bound (m/s^2)

A_v Intermediate variable in Vincenty formula

a_v Vehicle acceleration (m/s^2)

B Exponential zone time constant inverse (Ah^{-1})

b_0 Parameter in line fitting

b_1 Parameter in line fitting

b_ι Intermediate variable in Slerp-based interpolation

b_e Minor semiaxis of ellipsoid

B_v Intermediate variable in Vincenty formula

c Category of a bus stop

C_d Drag coefficient

c_e Horizontal coordinate of south-east vertex of square GPS bound

c_g Intermediate variable in great circle distance calculation

c_h Horizontal coordinate of the centre of GPS bound

c_n Vertical coordinate of north-west vertex of square GPS bound

C_r Rolling resistance coefficient

c_s Vertical coordinate of south-east vertex of square GPS bound

c_v Vertical coordinate of the centre of GPS bound

c_w Horizontal coordinate of north-west vertex of square GPS bound

D Squared distance difference between on-demand bus stop and bus location (m^2)

d Distance (m)

d_0 Starting distance (m)

d_c Distance of compulsory bus stop (m)

d_d Distance of on-demand bus stop (m)

d_f Final distance (m)

d_g Great circle distance

d_i Instantaneous distance (m)

d_i Instantaneous distance (m)

d_p Distance from centre to border of GPS bound

d_s Location of the slope change (m)

d_T Total distance of a route (m)

d_v Distance calculated with Vincenty formula

$d_{p_o p_i}$ Distance between points p_o and p_i

E Battery no-load voltage (V)

$e(\cdot)$ Boundary conditions

E_0 Battery constant voltage (V)

e_L Lower bound of boundary conditions

e_U Upper bound of boundary conditions

e_{mt} Motor voltage (V)

$f(\cdot)$ Generic function

f_0 Basic coefficient for rolling resistance coefficient

f_ι Fraction along great circle route for interpolation

$f_a(t)$ Transformed smooth heaviside function

$f_b(t)$ Transformed smooth heaviside function

f_e Flattening of ellipsoid

f_s Speed effect coefficient for rolling resistance coefficient

g Gravitational acceleration (m/s^2)

h	Elevation of road (m)
$H(\cdot)$	Smooth heaviside function
$h(\cdot)$	Path constraints
h_L	Lower bound of path constraints
h_U	Upper bound of path constraints
I	Electric current
I_{ess}	ESS current (A)
I_{pe}	Motor current (A)
J	Performance index
J_w	Wheel-axle moment of inertia (kgm^2)
K	Polarisation voltage (V)
k_b	Bearing coefficient
k_e	Voltage constant of motor (Vs/Wb/rad)
k_m	Meshing coefficient
k_s	Seal coefficient
k_t	Torque constant of motor (Nm/A/Wb)
k_{cd}	Coefficient of conduction losses (V/A)
k_{mt}	Numerical value for torque/voltage constant of motor
k_{sw}	Coefficient of switching losses
l	Convergence variable in Vincenty formula
$L(\cdot)$	Path cost
$M(\cdot)$	Terminal cost
m_v	Vehicle mass (kg)
N	Number of sample data
n_0	Initial point approximation algorithm

n_c	Number of compulsory bus stops
N_h	Number of hidden neurons
N_i	Total counts of bus passing
n_I	Number of input elements
n_p	Number of phases
n_t	Total number of points
N_{gb}	Gear ratio
n_{si}	Total counts of bus stopping at a bus stop
P_c	Consumed power of ESS (Nm/s)
p_i	Profiles in function of time
p_o	Centre of GPS bound
P_s	Stop probability
p_w	Centre point of the west side of a square GPS bound
P_{cd}	Power electronic conduction loss (Nm/s)
P_{ess}	Power demand passed from power electronics block to ESS block (Nm/s)
P_{sw}	Power electronic switching loss (Nm/s)
Q	Battery capacity (Ah)
r	Pearson correlation coefficient
R_E	the Earth's radius = 6271 km
r_w	Vehicle wheel radius (m)
R_{int}	Battery internal resistance (Ω)
R_{mt}	Internal resistance in motor block (Ω)
s_c	Current location
s_n	Location at next time-constrained location
t	Time (s)

T_0	Starting time (s)
t_0	Initial time in optimal control problem
T_c	Time of arrival at compulsory bus stop (s)
t_c	Current time
T_d	Time of arrival at on-demand bus stop (s)
T_f	Maximum time (s)
t_f	Terminal time in optimal control problem
t_i	Instantaneous time (s)
t_n	Time at next time-constrained location
t_o	Optimal time for current location
t_T	Total time to finish a route (s)
$u(t)$	Control in an optimal control problem
U_i	Reduced latitude of latitude ϕ_i
u_v	Intermediate variable in Vincenty formula
V	Electric potential
$V(\cdot)$	Instantaneous vehicle power consumption
$v(t)$	Speed profile (m/s)
v_0	Starting speed (m/s)
v_c	Current speed
v_f	Final speed (m/s)
v_L	Lower speed bound (m/s)
$v_n(t)$	Adjusted speed profile
$v_o(t)$	Extracted optimal speed profile
v_U	Upper speed bound (m/s)
v_v	Vehicle speed (m/s)

$v_{o'}(t)$ Scaled optimal speed profile

V_{pe} Voltage demand passed from motor block to power electronics block(V)

w Multiplication factor of standard deviation

w_d Weight for on-demand bus stop formulation

X One column in a generic tabular data

$x(t)$ State in an optimal control problem

x_ι Intermediate variable in Slerp-based interpolation

$x_i(t)$ ith state of optimal control problem

Y One column in a generic tabular data

y_ι Intermediate variable in Slerp-based interpolation

z_ι Intermediate variable in Slerp-based interpolation

$\mathbb{A}$ GPS point on a bus route

$\mathbb{B}$ GPS point on a bus route

$\mathbb{C}$ GPS point on a bus route

$\mathbb{D}$ GPS point on a bus route

Acronyms

2D two dimensions.

3D three dimensions.

ANN artificial neural network.

API application programming interface.

APMonitor optimal control program [80].

APOPT optimisation problem solver [87].

ARIMA auto-regressive integrated moving average.

Bocop optimal control program [81].

BPOPT optimisation problem solver.

BPOpt (building information modelling)-based performance optimisation [88].

CAN controlled area network.

CDCS charge-depletion and charge-sustenance.

COIN-OR Computational Infrastructure for Operations Research.

CPU central processing unit.

DB9 D-Subminiature types connector with 9 pins/holes.

DP dynamic programming.

EMMV energy management strategy based on mean values.

ESS energy storage system.

EV electric vehicle.

FARIMA fractional ARIMA.

FGPMMOPA6H GPS module from GlobalTop Technology Inc..

GPGGA NMEA sentence for GPS fix data.

GPGSA NMEA sentence for satellites status.

GPGSV NMEA sentence for satellites in view information.

GPOPS-II optimal control program [78].

GPRMC NMEA sentence for recommended minimum GPS data.

GPS global positioning system.

GPVTG NMEA sentence for the information of the course over ground and the ground speed.

GUI graphic user interface.

H-ARIMA historical ARIMA.

HAM historical average model.

HAT hardware attached on top.

IOXFORD66 a weather station in Oxford.

IPOPT Interior Point Optimizer [82].

JModelica optimal control program [79].

JSON Javascript Object Notation.

KNITRO optimisation problem solver [85, 86].

MATLAB mathematical computing software using matrix-based language.

MCP2515 CAN controller from Microchip.

NLP non-linear programming.

NMEA National Marine Electronics Association.

OS operating system.

OVEM Oxford Vehicle Model.

PiCAN2 CAN bus module.

PMP Pontryagin's Minimum (or Maximum) Principle.

PPS precise positioning service.

PSOPT optimal control program [77].

Pybrain Python library for neural network development [145].

RTK Real Time Kinematic.

SC104 format for transmitting real-time differential GPS corrections.

Slerp spherical linear interpolation.

SNOPT optimisation problem solver [84].

SNR signal-to-noise ratio.

SPI serial peripheral interface.

SPS standard positioning service.

SV space vehicles (GPS satellites).

UART universal asynchronous receiver/transmitter.

UK United Kingdom.

UTC coordinated universal time.

WGS84 World Geodetic System standard established in 1984.

Contents

Declaration i

Acknowledgements ii

Abstract iii

Nomenclature v

Acronyms xiii

Contents xvi

List of Figures xix

List of Tables xxiii

1 Introduction 1
- 1.1 Understanding the challenges . . . 1
- 1.2 Research question, objectives, and criteria . . . 4
- 1.3 Approach to the problem . . . 6

2 Literature Review 13
- 2.1 Bus journey optimisation for energy minimisation . . . 13
- 2.2 Estimation of bus journey . . . 20
- 2.3 Gap analysis . . . 28

3 Optimal Control of a Bus 30
- 3.1 Optimal control algorithm and program . . . 33
  - 3.1.1 Optimal control algorithm . . . 34
  - 3.1.2 Optimal control program . . . 35
- 3.2 Models . . . 40
  - 3.2.1 The model of vehicle . . . 40
  - 3.2.2 The model of road . . . 50

3.2.3 The model of bus route . . . 51
3.3 Optimal control problem formulation . . . 53
3.3.1 Basic route . . . 54
3.3.2 Route with an on-demand bus stop . . . 57
3.3.3 Route with a compulsory bus stop . . . 59
3.3.4 Route with different slopes . . . 64
3.3.5 Route with an on-demand bus stop, a compulsory bus stop and a slope change . . . 69
3.4 Case study . . . 73
3.4.1 Case I: Base case . . . 76
3.4.2 Case II: Bus route with bus stops . . . 80
3.4.3 Case III: Bus route with bus stops and slope changes . . . 82
3.5 Summary . . . 84

4 Parameter Identification for Real-World Implementation of Optimal Bus Driving 86
4.1 Basic formulae . . . 88
4.1.1 GPS distance calculation . . . 89
4.1.2 GPS location interpolation . . . 98
4.1.3 GPS bound identification . . . 106
4.2 Bus route processing . . . 108
4.3 Bus stop fitting . . . 116
4.4 Determination of optimal speed at on-demand bus stop . . . 118
4.5 Identification of stop probability of bus at bus stop . . . 119
4.6 Summary . . . 124

5 Bus Speed Estimation 130
5.1 Data description . . . 131
5.1.1 Time data . . . 131
5.1.2 Spatial data . . . 131
5.1.3 Weather data . . . 132
5.1.4 Social data . . . 133
5.2 Artificial neural network . . . 133
5.2.1 Data correlation . . . 135
5.2.2 Data formatting . . . 136
5.2.3 Data normalisation . . . 137
5.2.4 Network architecture . . . 138
5.2.5 Data division for training, testing and validation . . . 143
5.2.6 Network training . . . 143
5.3 Case study . . . 144
5.3.1 Average speed on road estimation . . . 145
5.3.2 Bus lane speed estimation . . . 154

5.4 Summary 164

6 Realisation of Optimal Bus Driving with Low-Cost System 167
6.1 Hardware setup 168
6.1.1 Computational unit 169
6.1.2 Data acquisition unit 171
6.1.3 Communication unit 173
6.2 Software setup 175
6.2.1 GPS NMEA sentences processing 176
6.2.2 Vehicle position identification 180
6.2.3 Adjustment algorithm 184
6.2.4 Software structure 191
6.3 Scenarios for profile adjustment 194
6.4 Summary 198

7 Conclusion 200
7.1 The key objectives 200
7.2 Contributions 206
7.3 Future work 207

References 209

List of Figures

1.1 Relations between the optimal control algorithm, the bus, and the adjustment algorithm . . . 12
2.1 Summary of operational optimisation of vehicle to minimise energy consumption . . . 21
3.1 Parameter flow in backwards simulation . . . 41
3.2 Equivalent circuit of a direct-current motor . . . 45
3.3 Battery model . . . 47
3.4 Basic bus route . . . 54
3.5 Bus route with an on-demand bus stop . . . 57
3.6 Rectangular function activated within $T_d \pm \delta_d$ . . . 58
3.7 Compound of two smooth heaviside functions to form a smoothed rectangular function . . . 58
3.8 Bus route with a compulsory bus stop . . . 60
3.9 Expected speed profile between $T_c - \delta_c$ and $T_c + \Delta_c$ . . . 60
3.10 Bus route with slope change . . . 64
3.11 Conditioned slope profile for smooth linear interpolation . . . 68
3.12 Solution time comparison of two approaches for slope change formulation 69
3.13 Bus route with an on-demand bus stop, a compulsory bus stop and a slope change . . . 70
3.14 Bus route from Oxford City Centre to Pear Tree Park and Ride . . . 74
3.15 Slope and elevation profile for bus route from Oxford City Centre to Pear Tree Park and Ride . . . 75
3.16 Bus route from Oxford City Centre to Pear Tree Park and Ride . . . 76
3.17 Optimal profiles with single stop at the end and zero slope . . . 77
3.18 Profiles of an aggressive driving style with single stop at the end and zero slope . . . 79
3.19 Optimal profiles with bus stops and zero slope . . . 81
3.20 Optimal profiles with bus stops and slope changes . . . 83
4.1 Typical (blue) and testing (red) graphical area . . . 87
4.2 Illustration of major a_e and minor b_e semiaxes of ellipsoid . . . 90
4.3 Points for cosine-haversine and Vincenty formulae comparison in testing area . . . 93

4.4 Graphical representations for comparison of consine-haversine and Vincenty formulae in testing area . . . 93
4.5 Points for cosine-haversine and Vincenty formulae comparison in typical area with point 0 as the centre of the typical area . . . 94
4.6 Graphical representations for comparison of consine-haversine and Vincenty formulae in typical area . . . 96
4.7 Points for cosine-haversine and Vincenty formulae comparison in typical area (different bearing) with point 0 as the centre of the typical area . . . 97
4.8 Graphical representations for comparison of consine-haversine and Vincenty formulae in typical area (different bearing) . . . 97
4.9 Comparison of interpolated latitude using Slerp-based interpolation and linear interpolation . . . 101
4.10 Comparison of $\frac{(1-f_\iota)\cos\phi_1\sin\lambda_1+f_\iota\cos\phi_2\sin\lambda_2}{(1-f_\iota)\cos\phi_1\cos\lambda_1+f_\iota\cos\phi_2\cos\lambda_2}$ with $\tan((1-f_\iota)\lambda_1+f_\iota\lambda_2)$. 102
4.11 Comparison of interpolated longitudes using Slerp-based interpolation and linear interpolation . . . 103
4.12 Comparison of approximation results with the typical geographical area . 104
4.13 Errors between Slerp-based interpolation and linear interpolation from (30.1,30.1), denoted by the white dot with red border, to all possible GPS coordinates on Earth (rounded to 0.5°) with f_ι as 0.5. Top graph shows the errors for latitude interpolation and bottom graph shows the errors for longitude interpolation. . . . 105
4.14 Algorithm for repetitive interpolation . . . 106
4.15 Bound illustration . . . 106
4.16 Illustration of relevant points in coordinates identification algorithm . . . 107
4.17 Workflow to add road segment to bus route . . . 109
4.18 Relationship of slope and elevation . . . 110
4.19 Example of fitting two consecutive lines . . . 111
4.20 Bus route from Oxford City Centre (A) to Pear Tree Park and Ride (B) for elevation extraction . . . 113
4.21 Mean square errors of adjusted elevation profile in parameter sweep . . . 114
4.22 Mean square errors of adjusted slope profile in parameter sweep . . . 114
4.23 Sum of the normalised mean square errors of adjusted elevation and slope profile and the normalised number of slope change . . . 115
4.24 Original and adjusted elevation and slope profiles . . . 116
4.25 Example of bus stop along a bus route . . . 117
4.26 Modified example of bus stop along a bus route . . . 117
4.27 Modified example of bus stop along a bus route . . . 117
4.28 Daily profiles in each cluster . . . 122
4.29 Centroids of daily profile clusters . . . 123
4.30 Analysis of clustered profiles . . . 123
4.31 Collected data to identify speed threshold for vehicle halt identification . 125

4.32 Percentage of false positives and false negative for using different speed threshold as halt . . . 126
4.33 Stop probability on Sunday . . . 127
4.34 Stop probability on Monday and Tuesday . . . 127
4.35 Stop probability on Wednesday, Thursday and Friday . . . 128
4.36 Stop probability on Saturday . . . 128
5.1 Basic network without bias . . . 141
5.2 Basic network with bias . . . 141
5.3 Output of basic network without bias at various weights . . . 142
5.4 Output of basic network with bias at various weights . . . 142
5.5 Road segment (First Turn to South Parade) and weather station (IOXFORD66) used in case study . . . 145
5.6 Speed data for road segment First Turn to South Parade . . . 146
5.7 Weather data (temperature) for weather station IOXFORD66 . . . 146
5.8 Weather data (pressure) for weather station IOXFORD66 . . . 146
5.9 Weather data (rainfall) for weather station IOXFORD66 . . . 147
5.10 Weather data (humidity) for weather station IOXFORD66 . . . 147
5.11 Weather data (dew point) for weather station IOXFORD66 . . . 147
5.12 Weather data (precipitation) for weather station IOXFORD66 . . . 148
5.13 Weather data (wind speed) for weather station IOXFORD66 . . . 148
5.14 Speed data for road segment against weather data . . . 149
5.15 Comparison of target and predicted output for validation data . . . 153
5.16 Comparison of target and predicted output for validation data (extracted) . . . 153
5.17 GPS bounds for road segment First Turn to South Parade . . . 155
5.18 Bus speed data plotted against the road segment speed data . . . 156
5.19 Bus speed data plotted against the weather data and road segment speed data 157
5.20 Comparison of target and predicted output for validation data . . . 160
5.21 Comparison of target and predicted output for validation data with input data of speed data only . . . 161
5.22 Comparison of target and predicted output for validation data with input data of speed data, weather data and social data . . . 161
5.23 Comparison of target and predicted output for validation data with input data of speed data and weather data . . . 162
5.24 Comparison of target and predicted output for validation data with input data of speed data and social data . . . 162
5.25 Scatter plots of target values against predicted values of neural network trained with input data of (i) speed data only, (ii) speed data, weather data, and social data, (iii) speed data and weather data, and (iv) speed data and social data . . . 163
6.1 Block diagram for hardware setup . . . 169
6.2 Image of the hardware setup . . . 169
6.3 Image of Raspberry Pi 1 model B . . . 170

6.4 Image of Adafruit Ultimate GPS HAT 171
6.5 Image of external antenna for Adafruit Ultimate GPS HAT 172
6.6 Image of PiCAN2 174
6.7 GPGGA sentence definition 177
6.8 GPGSA sentence definition 178
6.9 GPGSV sentence definition 179
6.10 GPRMC sentence definition 181
6.11 GPVTG sentence definition 182
6.12 Vehicle position relative to a route 182
6.13 Vehicle position on a route if the nearest point is the first or the last point . 183
6.14 Example of optimal profiles for demonstration of adjustment algorithm (Top: speed profile; Bottom: distance profile) 185
6.15 Function $A(t)$ to be derived 188
6.16 Base functions for $A_1(t)$ (left) and $A_2(t)$ (right) respectively 189
6.17 Functions $A_1(t)$ (left) and $A_2(t)$ (right) 189
6.18 Interaction between scripts and data files in the software system 194
6.19 Example journey 195
6.20 When stopping at on-demand bus stop 196
6.21 When start slow from a bus stop 197
6.22 When delayed due to traffic 198

List of Tables

2.1 Summary of literature on bus journey estimation 27
3.1 Solvers used by the optimal control programs 38
3.2 Model of the road between Oxford City Centre to Pear Tree Park and Ride 74
3.3 Model of the route between Oxford City Centre to Pear Tree Park and Ride 75
3.4 Bounding variable values for bus route from Oxford City Centre to Pear Tree Park and Ride 75
4.1 Comparison of cosine-haversine and Vincenty formulae in testing area . . 92
4.2 Comparison of cosine-haversine and Vincenty formulae in typical area . . 95
4.3 Comparison of cosine-haversine and Vincenty formulae in typical area (different bearing) 95
5.1 Commonly used activation functions 140
5.2 Minimum and maximum values to which the data are normalised 150
5.3 Comparisons of different sets of input data 151
5.4 Comparisons of different training/testing data split 152
5.5 Comparisons of criteria for hidden neuron number calculation 152
5.6 Optimal parameters for average speed on road estimation neural network . 152
5.7 Minimum and maximum values for normalisation of data set containing only the bus speed data and speed on road data 158
5.8 Minimum and maximum values for normalisation of data set containing the bus speed data, speed on road data, and weather data 158
5.9 Comparisons of different sets of input data 159
5.10 Comparisons of different training/testing data split 159
5.11 Comparisons of criteria for hidden neuron number calculation 160
5.12 Optimal parameters for average speed on road estimation neural network . 161
6.1 Energy consumption of profiles in different scenarios 198

Chapter 1

Introduction

This thesis addresses the research question of minimising the energy consumption of an electric bus along a bus route while improving the arrival time of the bus at bus stops along the route.

1.1 Understanding the challenges

Global warming is causing climate change across the world and increment of sea level which can potentially be disastrous especially to coastal cities. One major cause for global warming is the emission of greenhouse gases such as carbon dioxide, methane, nitrous oxide, and fluorinated gases. From the report of the Intergovernmental Panel on Climate Change 2014 [1], transport sector is one of the main contributors to the emission of these gases. According to the report published by the United Kingdom (UK) Department of Transport [2], 51.6 out of 70.2 million tonnes or 73% of total petroleum used in the UK in 2015 is consumed in the transport sector. In the same year, 40% of the total energy used by the final users in the UK is used in the transport sector [3, 4]. This is followed by the

domestic sector (29%), industry sector (17%), and service sector (14%) [4].

Air pollution is another issue experienced at different levels throughout the world. The depletion of air quality puts the health of the people at risk. According to the Air Quality Guidelines published by the World Health Organization [5] and the report published by the UK Department for Environment, Food and Rural Affairs [6], road transport is the main source of multiple air pollutants including nitrogen oxides (NO_x), particulate matter of 2.5 micrometers or less in diameter ($PM_{2.5}$), benzene (C_6H_6), and carbon monoxide (CO). Road transport also contributes partly to the emission of particular matter of 10 micrometers or less in diameter (PM_{10}). The emission of NO_x also leads to the production of Ozone (O_3) due to the reaction catalysed by the sunlight.

The reduction of energy consumption in the transport sector therefore reduces the emission of both greenhouse gases and air pollutants, which in the long run, contributes to the reduction of global warming and air pollution.

Scarbrough et al. [7] suggest the usage of electric vehicle (EV) as one of the measures to reduce air pollution. The promotion of EV reduces the usage of petroleum fuel vehicles that rely on the combustion of fossil fuel and thus reduces the emission of both greenhouse gases and air pollutants. On top of that, as EV is powered by electricity generated from fossil fuel, reducing the energy consumption of an EV also reduces the carbon dioxide emissions.

A report by Carslaw and Priestman [8] states that buses of the same bus model (Euro V SCR hybrid vehicle) operated by two different companies (Oxford Bus Company and

Stagecoach) at the same location (High Street, Oxford) have consistently different NO_x emission. It implies that it is not just the design of the power train that determines the energy consumption of the vehicle, but also the operational aspect of the vehicle. Therefore, energy consumption of a vehicle can be reduced by optimising the driving of the vehicle.

A personal vehicle journey is normally not properly bound, i.e. the time to complete the journey is not defined. However, the journey of a bus is pre-defined with scheduled time at various fixed locations along the journey and destination. Therefore, it is possible to optimise the driving of a bus. The focus on the electric bus is supported by the report published by Cenex [9] as the report forecasts more and more electric buses will be operated on the streets due to the maturing of EV technology.

The use of buses helps reduce the problem of traffic congestion. However, the efforts to promote the use of buses have not produced encouraging outcomes. The consistent complaint from bus users is that the bus does not follow the scheduled time. This results in the difficulty to pre-plan their commutes.

The main challenge to implement the optimal speed in a real-world context is the limitation of bus speed due to traffic conditions on the road. The traffic speed on the road provides the top allowable speed for the bus. Therefore either the traffic speed should be considered in the optimisation problem or the pre-computed optimal speed should be updated when the speed of bus is limited by the traffic.

1.2 Research question, objectives, and criteria

The main research question of minimising the energy consumption of an electric bus along a bus route while improving the arrival time of the bus at bus stops along the route is broken down to four objectives.

The first objective of this research is to set up a procedure to define the key parameters of the optimisation problem that aims to minimise the energy consumption of an electric bus along a route and improve the arrival time of the bus at bus stops along the route. The difference between the optimisation problem of a personal vehicle and a bus is the presence of bus stops along the route and their respective scheduled time of arrival. There are two different types of bus stops, namely compulsory and on-demand bus stops. Both bus stops are defined with a scheduled time of arrival. A bus is always required to stop at a compulsory bus stop for a defined time period before departing whereas a bus only stops at an on-demand bus stop if there is a request to alight or board the bus. The criterion of this objective is to obtain an optimal profile of a bus along a bus route with different types of bus stops. The optimality of the profile is determined by comparing the optimal profile with a typical driving profile of a bus driver.

The second objective is to consider real-world bus routes in the optimisation process to obtain the optimal profile. A bus route and bus stops are identified as geodesic locations from a map. A geodesic location is a geographical location defined as a pair of coordinates consist of its latitude and longitude. This can be obtained either from a global positioning

system (GPS) device or from online resources such as the Google Map. The parameters of the bus route such as the length of the route and the distance of each location from the beginning of the route need to be extracted. The relative positions of the bus stops on the route also need to be identified. The parameters extraction of the bus route and the identification of the bus stop positions enable the open loop optimisation to be performed on a real-world bus route. The criterion of this objective is to process a real-world bus route and obtain the appropriate and necessary parameters of the bus route.

The third objective is to predict the real-world traffic speed. The predicted traffic speed can be used to improve the open loop optimisation process. The traffic conditions along the bus route provide limits to the bus speed in addition to the legal speed limits. Therefore the ability to predict traffic speed along the bus route allows the definition of a realistic optimal profile that conforms to the real-world traffic conditions at the time of implementation. The effect of the traffic speed on the bus also depends on whether the bus travels on a mixed traffic lane or a bus lane. A bus travels on a mixed traffic lane as part of the traffic flow formed by other vehicles like personal cars, trucks, and so on. When a bus travels on a bus lane, it is separated from the traffic flow of other vehicles that runs on parallel alongside the bus lane. The criterion for this objective is to provide the methodology for the prediction of on-road speed that acts as a speed limitation for the bus. The methodology will be demonstrated to predict the traffic speed of a section of a road.

The fourth objective is to set up a real-time system that provides and updates the optimal speed when necessary. The real-time system is vital for the implementation of the

optimal profile in real world. The system acquires the real-time physical location of the bus and provides the appropriate optimal speed in order for the bus to arrive at bus stops on time with minimum energy. As uncertainties happen in real world, the system is required to update the optimal profile when the bus is at suboptimal location. The criterion of this objective is to provide a low-cost embedded system that delivers the optimal speed and updates the optimal profile in real time.

1.3 Approach to the problem

As the main focus of this thesis is to minimise the energy consumption of a bus along the bus route and improve the arrival time of the bus at bus stops along the route, the controlling parameter of this optimisation problem is the speed of the bus. The speed of the bus along the bus route is not a constant value but a time-varying variable. Therefore, an optimal control problem is set up to achieve the goals. An optimal control algorithm solves a problem in which the control variable is solved as a time series. Other approaches to reduce energy consumption of vehicle driving include power flow optimisation and eco-driving training. However, it is difficult to include the arrival time requirements in these approaches. In the optimal control of the bus speed, the energy consumption is calculated with a vehicle model which takes the speed and acceleration of the bus and the slope of the road as inputs. The improvement of bus arrival time at bus stops is measured by how close the actual arrival time and the scheduled time. Therefore in the optimal control problem, to improve the bus arrival time at bus stops is to minimise the discrepancy between the actual

arrival time of the bus and the scheduled time at the bus stop.

The main challenge in the first objective is to consider compulsory and on-demand bus stops in the optimal control problem. Both bus stops are defined with specific arrival time with a time tolerance. As a compulsory bus stop requires the bus to stop and wait until the departure time, a waiting time is defined as the period of time for the bus to wait at the bus stop, within which the speed of the bus is kept at zero. A compulsory bus stop is formulated using the multiphase approach in which the bus journeys before, at, and after the bus stop are defined as three different phases. For an on-demand bus stop, the arriving speed of the bus at the bus stop is not defined. The only requirement of an on-demand bus stop is the requirement of arrival time within a certain tolerance. Therefore the arrival time of an on-demand bus stop is formulated in the objective function of the optimal control problem to be minimised.

Chapter 3 discusses the work on the setting up of the optimal control problem. The selection of the algorithm and the program to perform optimal control is explained. The models that define different parameters of the problem are developed. These models include the model of the vehicle for energy consumption calculation, the model of the road for the slope of the road that affects the energy consumption, and the model of the bus route for the bus stop locations and arrival time requirements. EV is selected as the model of the vehicle used for the calculation of the energy consumption since the model of an EV is simpler, and the energy storage is a more critical issue in an EV compared to diesel and petrol cars or hybrid vehicles. From the usage point of view, EV is also becoming a

more dominant vehicle type on the road. The methodology to formulate the on-demand bus stops, the compulsory bus stops, and the slope changes in the optimal control problem is discussed.

The second objective focuses on the real-world consideration of the locations in the open loop optimisation process. There are two challenges regarding this objective. The first challenge is to identify the positions of bus stops on a bus route. The location of a bus stop is provided in the form of geodesic coordinates and a bus route is provided as a series of points with geodesic coordinates. It is not straightforward to identify the relative position of the bus stop on the bus route. Therefore an algorithm that utilises the distances and the relative bearings between the geodesic points is designed to identify the points on the bus route that encapsulate the bus stop.

The second challenge in the second objective is to identify a realistic optimal speed at an on-demand bus stop. As the decision to stop at an on-demand bus stop depends on the requests of the passengers to alight or board the bus, the speed of the bus passing an on-demand bus stop is not bound. However, it is possible to identify the probability of the bus to stop at an on-demand bus stop based on historical data. The probability can then be used to adjust the speed of the bus passing an on-demand bus stop. A procedure is developed to identify the stop probability of a bus at an on-demand bus stop.

Chapter 4 first discusses the basic formulae used in the processing of geodesic coordinates. These include distance calculation between two points, interpolation of multiple points between two points, and the identification of a geodesic bound using a centre point

and its distance to the boundary. These formulae are then used for the setting up of a real-world bus route and the identification of the relative position of a bus stop on a bus route. The basic formulae are also used in the location-based extraction of historical bus data to identify the stop probability of a bus at an on-demand bus stop.

The third objective focuses on the consideration of the real-world traffic speed in the open loop optimisation process. The main challenge in the third objective is to identify the traffic condition that limits the speed of bus on the road which in turn limits the ability of a bus to follow the optimal profile. Therefore, the prediction of the traffic speed along the bus route can potentially be included in the optimal control problem to provide an optimal profile that is closer to the real-world situation. As the speed parameter that limits the bus speed is different depending on whether the bus travels on a mixed traffic lane or a bus lane, the predictions of speed on a mixed traffic lane and speed on a bus lane are explored separately. The parameters used to predict the speed on a mixed traffic lane include the time data, weather data, and social events. Social events are activities that changes the commuting behaviours of the road users such as school holidays and weekends. As the real-world observations show a dependency between the speed on the bus lane and the speed on the corresponding mixed traffic lane, the prediction of the speed on a bus lane using the speed of the corresponding mixed traffic lane is attempted. These predictions are performed using artificial neural networks (ANNs).

Chapter 5 discusses the procedure of developing ANNs. The chapter first describes the data sources of the time data, spatial data, weather data, and social data. The methods to

clean the data are also discussed where necessary. The procedure in developing an ANN includes data formatting, data normalisation, network architecture design, proportion split of data for training, testing, and validation, and network training. The process to design and develop the two specific ANNs for the prediction of speed on the mixed traffic lane and the prediction of speed on the bus lane are discussed.

The fourth objective focuses on the real-world implementation of an optimal control system. Two challenges are faced in this objective. The first challenge is to deliver the optimal speed in real-time. Real-world implementation involves the identification of location of the bus on the bus route and the delivery of the optimal speed corresponds to the location. An on-board system is designed to acquire the bus location and extract from the optimal profile the optimal speed corresponds to the location.

The second challenge is the adaptation of optimal profile when disruptions occur. This is important as real-world uncertainties cause the bus to be unable to follow the optimal profile. In order to perform the update of the optimal profile in reasonable time, one potential option is to design a system that communicates with a server that performs re-optimisation in a short time. However, there is a possibility that there is no robust communication link along the route. Alternatively, a standalone system with supercomputer can be designed and placed on the bus to perform the re-optimisation. The problem of this option is the high cost of a supercomputer. Therefore, an adjustment algorithm is designed to update the optimal profile to ensure the fulfilment of the requirements of bus stops and destinations. The adjustment algorithm also allows the usage of a low-cost on-board system that

can be deployed on the bus.

Chapter 6 explains the hardware and software of the designed system. The on-board system consists of a computational unit, a data acquisition unit, and a communication unit is designed to acquire the real-time location of the bus, identify the current optimal speed, and convey the optimal speed to external systems. The software of the system is designed to process the acquired data to identify the relative location of the bus on the route and provide the optimal speed. An adjustment algorithm is developed to modify the optimal profile when the bus does not follow the optimal profile. The scenarios in which the bus does not follow the optimal profile are discussed together with the adaptation to each scenario.

The optimal control algorithm in Chapter 3 provides an optimal speed profile that acts as the reference for the bus. The optimal profile is established in an open loop manner. The aim is to drive the bus following the reference profile. When the bus is unable to follow the reference profile, the adjustment algorithm in Chapter 6 modifies the reference profile in order to suit the actual location and speed of the bus and provide a new reference profile to the bus. The adjustment algorithm provides a closed loop mechanism to update the reference profile. The relations between the optimal control algorithm, the bus, and the adjustment algorithm are shown in Figure 1.1.

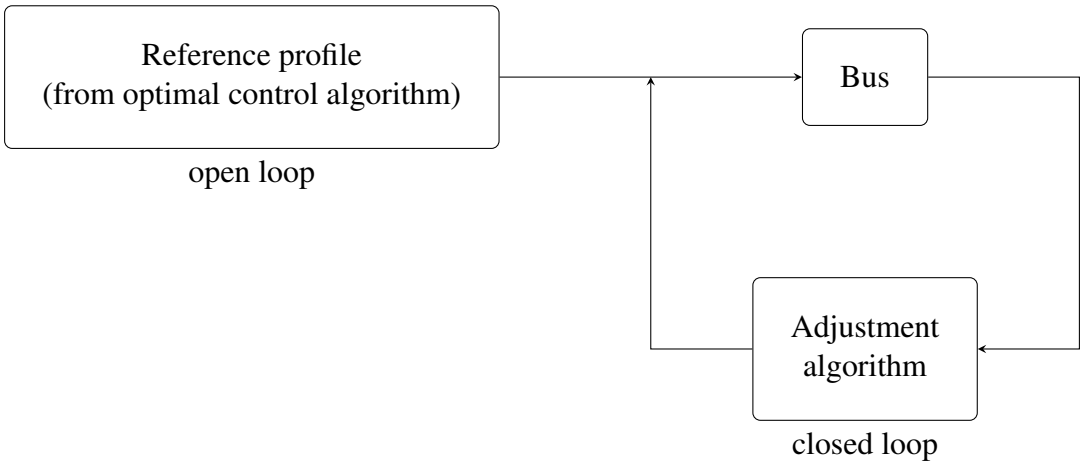


Figure 1.1: Relations between the optimal control algorithm, the bus, and the adjustment algorithm

Chapter 2

Literature Review

This chapter focuses on the review of the literature in two main areas: (i) the minimisation of energy consumption of a vehicle and (ii) the estimation of a bus journey's time and speed. It mainly focuses on the objectives of formulating a bus journey for optimal control purposes and considering real-world traffic speed in the optimisation process. As this thesis focuses on the operational optimisation of buses to minimise energy consumption along the journey, different approaches to achieve this are discussed and compared. The challenge to implement the result of the operational optimisation is the effect of traffic on the speed of the bus. Therefore this chapter also discusses several methods to estimate a bus journey.

2.1 Bus journey optimisation for energy minimisation

The operational optimisation of a vehicle has been explored by different researchers. The main interest of optimising the operation of a vehicle is to minimise the energy consumption. The reduction of energy consumption not only means cost reduction, but also more efficient use of the resources. Operational optimisation can be performed by optimising

either the power management of a powertrain or the driving styles of a driver.

The optimisation of power management is only applicable to hybrid vehicles. As there is more than one energy source in a hybrid vehicle, the optimisation algorithm identifies the optimal power split between different energy sources along the journey. The methods of power management optimisation fall generally into rule-based or optimisation-based algorithms.

A rule-based algorithm, as suggested by the name, controls the power split between sources based on a set of rules that should allow the energy sources to operate at their optimal states. Some examples of rule-based algorithms include charge-depletion and charge-sustenance (CDCS), blended, fuzzy logic controller, and energy management strategy based on mean values (EMMV).

CDCS [10, 11] algorithm uses the state of charge of the battery as the reference to trigger the usage of battery or other energy sources. The basic principle of CDCS is to solely utilise the battery when its state of charge is more than the lower limit (charge-depletion phase) and utilise the engine to run the vehicle and charge the battery when the state of charge is less than the lower limit (charge-sustenance phase).

The blended algorithm [12–14] is similar to CDCS. In the charge-depletion phase, the blended algorithm uses a blending ratio to blend the electric energy and fuel energy to provide the power requirement of the vehicle. The blending ratio is normally defined as the ratio of the electric energy to the sum of electric and fuel energy. CDCS is, in fact, an extreme case of the blended algorithm where the blending ratio is one during charge-

depletion phase. By introducing the blending ratio, the blended algorithm extends the charge-depletion phase and therefore delays the entry time to the charge-sustenance phase. Due to the observation that, the later the system enters the charge-sustenance phase, the lesser the energy it consumes [11, 14], the delay on the entry time allows the blended algorithm to have a better performance than CDCS. The optimal situation in using a blended algorithm is such that the charge-depletion phase terminates right at the end of the journey. The system will therefore not enter into the charge-sustenance phase and all the usable energy from the battery is depleted. However, this situation highly dependent on the knowledge of the route the vehicle travels.

Fuzzy logic controller, initiated by the work of Mamdani and Assilian [15], is based on the fuzzy set theory introduced by Zadeh [16]. Fuzzy logic controller defines the membership functions of each variable involved in the problem, and a set of rules is constructed to make decisions based on the membership functions. Kim et al. [17], Li et al. [18], and Hemi et al. [19] apply fuzzy logic controller in the controlling of power management of a hybrid vehicle with fuel cell and battery. The fuzzy logic controller uses the inputs of the power demand and the state of charge of the battery to identify the output of the fuel cell power. As the rules are highly dependent on the membership functions, the definition of the membership functions is a critical aspect of the fuzzy logic controller. Therefore, the optimality of the solution depends on the quality of the definition of the membership functions.

EMMV, proposed by Xu et al. [20], first identifies the parameters that determine the

optimal operation of the fuel cell. The average values of these parameters are obtained in real time to determine the instantaneous operation of the fuel cell. The real-time average values are calculated using low frequency filters or a Kalman filter [21]. Since the real-time calculations involved are simple and quick, this method is suitable for real-time implementation. The results obtained by Xu et al. [20] show that EMMV achieves longer driving mileage than CDCS and the blended algorithm.

An optimisation-based algorithm calculates the instantaneous numerical values of the optimising parameter along the journey. Optimisation-based algorithms require a priori knowledge of the route in order to calculate the numerical values of the power split ratio between different energy sources along the route. The examples of optimisation-based algorithms include dynamic programming (DP) and Pontryagin's Minimum (or Maximum) Principle (PMP).

DP, originally proposed by Bellman [22] based on the principle of optimality, solves the problem of bringing a system from one state to another state through a path by decomposing the path into a sequence of simpler minimisation problems. The application of DP in developing power management strategy provides a globally optimised strategy given the prior knowledge of the associated driving cycle. However, the time-consuming process of DP renders it unsuitable as a real-time optimisation algorithm. Chen et al. [23] use the result provided by DP to develop a rule-based algorithm for real-time application. As DP provides a globally optimal solution, it is also used as the reference to compare the optimality of the solution of other methods [14].

PMP is proposed by L.S. Pontryagin [24] to solve an operational optimisation problem by solving the Hamiltonian derived from the problem [25, 26]. The Hamiltonian is derived using the cost function and the state equations of the optimisation problem. The solution of PMP provides necessary conditions which may not be the sufficient condition [26]. Therefore PMP does not guarantee a globally optimal solution. However, studies [25, 27] have shown that PMP provides results similar to that of DP. The shorter execution time compare to DP and the near-optimal results allow PMP to be used as a real-time solution to the operational optimisation problem.

Rule-based algorithms generally have shorter execution time and simpler computation compare to optimisation-based algorithms. However, rule-based algorithms normally provide less optimal solutions. Xu et al. [11] show that the solution of CDCS has a much higher operation cost than that of DP or PMP. Therefore, it is common to develop rule-based algorithms and use the solution from an optimisation-based algorithm as evaluation reference.

As mentioned, the application of operational optimisation of power management is limited to hybrid vehicles. On the other hand, the optimisation of the driving styles of a driver is applicable in all vehicles as it only concerns the higher level of control of the vehicle, i.e. the speed and acceleration of the vehicle.

The optimisation of driving styles can be achieved by either providing eco-driving training to the drivers, or applying optimal control algorithms to compute an optimal driving profile.

Eco-driving training [28–31] is discussed by multiple researchers. Drivers are trained to make decisions that minimise the fuel consumption when they are on the road [28, 29]. Real-time and off-line feedback systems are developed to help the drivers to improve their decision makings. Real-time feedback systems can be either a notification system [29] that notifies the drivers of undesirable behaviours or a speed suggestion [30]. Off-line feedback systems such as driver coaching systems [31] are designed to collect and analyse information of the previous trip to provide personalised advices to improve fuel-efficient driving to the drivers.

Optimal control [32–39] is discussed as one of the options to optimise driving styles. Optimal control can be applied to a pre-defined route to compute the optimal driving profile which is then provided to the drivers.

The main parameters in defining an optimal control problem are the objective, the control parameter, and the simulated environment. The objective of an optimal control problem is used to formulate the performance index or objective function to be minimised. The control parameter is the variable that is controlled and modified by the optimal control algorithm to achieve the objective. The simulated environment determines the models and the constraints in an optimal control problem.

The common objective included in the optimal control of the driving of a vehicle is to minimise the energy consumption. The energy consumption is either calculated directly as the energy consumption [32–35], or the fuel consumption [36–39] of the vehicle. [32] and [33] consider the objective of comfortable driving on top of the minimisation of the energy

consumption. Comfortable driving is achieved by minimising the acceleration duration and total jerk of the vehicle along the journey. The calculation using directly the energy instead of fuel allows easy implementation of different vehicle types.

In controlling the driving of a vehicle, it is natural for one to select either the speed [35–37, 39] or acceleration [32–34, 38] of the vehicle as the control parameter. Since the gradient of the speed profile provides the acceleration profile, the acceleration profile of the vehicle is not required to be smooth whereas the speed profile has to be smooth to avoid infinite acceleration.

There are two features of a simulated environment in the optimal control of vehicle. First is the terrain of the road, and second is the traffic situation. Level road is often simulated as a base case [34–36,39]. The terrain of a simple hill [36–38], a hill cycle [36] or uphill with a certain inclination [39] is modelled to investigate the effect of road inclination on the optimal speed profile. The inclinations of the road that has been simulated are ±3% (1.72°) [36, 38], ±6% (3.43°) [36], 10% (5.71°) [37] and 15% (8.53°) [39]. The results have shown that the optimal speed of the vehicle reduces on a positive inclination and increases at a negative inclination. The effect is similar for different inclinations but the speed variation increases as the inclination increases.

Five different traffic congestions are simulated by Saboohi and Farzaneh [39]. They are (i) jam convey traffic flow in a crowded urban highway, (ii) fluid traffic flow in a free urban highway, (iii) bunched convey traffic flow in a high crowded urban highway, (iv) bunched convey traffic flow in an urban street with 15% inclination, and (v) jam traffic flow in an

urban street. The traffic conditions quantified as the traffic density and the net gap within a fluid convoy has an inverse relationship with the maximum speed and the maximum engine load of the optimal driving strategy. Similar to the simulation of traffic congestion, Pudney and Howlett [34] have simulated different speed limits at certain parts of a train journey.

Apart from Saboohi and Farzaneh [39], none of the optimal control related literatures has included the formulation of stops in the middle of the trip. Even in the work of Saboohi and Farzaneh [39], the idling of the vehicle is due to the traffic and not designated stops as in the case of a typical bus trip.

The different approaches used in the operational optimisation of vehicle to minimise the energy consumption is summarised in Figure 2.1.

This work uses acceleration as the control, to minimise two objectives (i) minimisation of energy consumption and (ii) improving of the bus punctuality to arrive at bus stops, and the environment is a varying terrain road based on a real-world road.

2.2 Estimation of bus journey

The estimated time of arrival for public transport has always been a public interest. An accurate estimation of arrival time increases the consumers' confident in the service and hence boosts the number of users and the profit of the service providers. Therefore different approaches of estimating time of arrival are proposed by multiple literature. The main features of an estimation algorithm include the output to be estimated, inputs to be used, and methods to estimate. The input parameters are used to estimate the output parameters

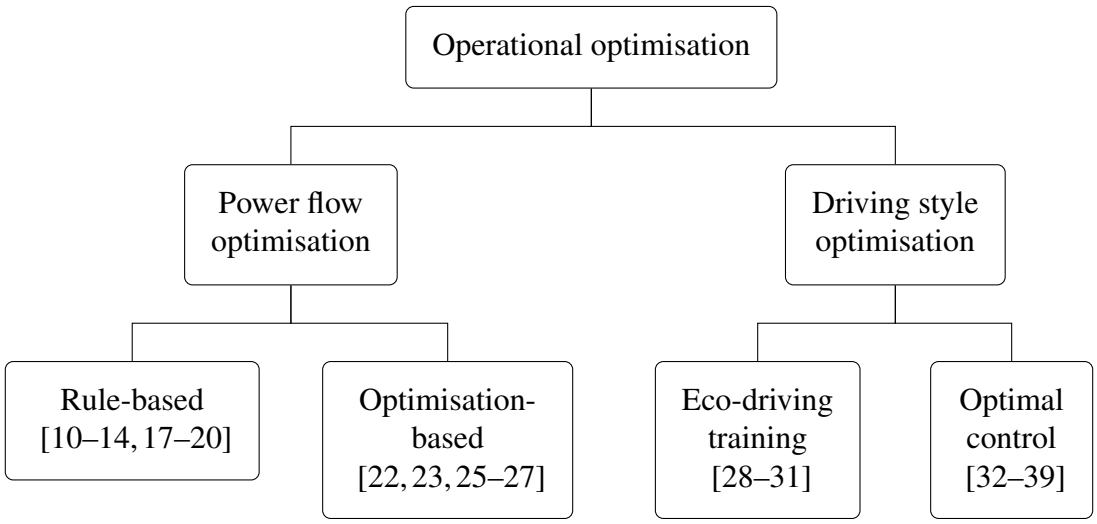


Figure 2.1: Summary of operational optimisation of vehicle to minimise energy consumption

using the estimation methods.

The output parameters used in the literature are the arrival time, travel time, dwell time, running speed, and traffic volume.

The arrival time of a bus at a bus stop [41, 42] is used directly as the output of the estimation as it removes the need to develop another model to compute the arrival time from other parameters. The arrival time to be estimated can either be the arrival time of the same bus at the next bus stop [41] or the arrival time of the next bus at the same bus stop [42].

Travel time from a bus stop to another [43–49] is most commonly used as the output parameter for estimation algorithms. A numerical function or model is often used to identify the estimated time of arrival of the bus at a specific bus stop from the estimated travel time.

Dwell time at bus stops is an important factor to determine the arrival time and is

estimated by Meng and Qu [50], Zhang and Teng [51], and Fan et. al. [52]. The bus dwell time is affected by the number of alighting and boarding passengers, the time of opening and closing the bus doors, and the number of vehicles at the bus stop. The bus dwell time is an unavoidable disturbance to a bus journey. The dwelling time varies a lot from time to time.

Running speed of a bus [52–56] at different segments of a bus journey can be estimated in order to calculate the arrival time at a specific bus stop. The estimation of the running speed is also used together with the estimation of the dwelling time to identify the actual bus arrival time [52].

Traffic volume affects the top speed at which a vehicle runs on the road. The estimation of traffic volume is therefore proposed by multiple researchers [55, 57–59].

The input parameters used in the literature are the number of passengers, weather, temporal data, bus dwell time, running speed, travel time, time delays, vehicle locations, and traffic volume.

The passenger number has a high correlation with bus dwell time at bus stop [50]. Therefore the number of passengers [41, 50, 51] is used as an input parameter mostly in estimating the dwelling time of a bus at a bus stop. The number of passengers is often obtained with an automatic passenger counter.

As weather condition changes the commuting habits of the people, it is included as an input parameter for estimation of bus arrival time in some cases [41,49]. The parameters to define the weather condition include wind, precipitation, visibility, and hourly temperature.

Precipitation is found to have the most significant impact [41, 49].

Temporal data [41,47,49] is often included as one of the input parameters of estimation algorithms as it is related to social activities that change people's commuting behaviours. The temporal data used are the time of day [41, 49], day of week [41, 49], and date [47].

As mentioned before, the bus dwell time affects the bus arrival time. Thus it is used as an input parameter in some literature [41, 52]. [52] utilises both the average bus dwelling time based on history, and also the bus dwelling time of the preceding bus on the same route.

The running speed of the bus [45,46,53–56] is most commonly used as input parameter for the estimation algorithms. Feng and Shu [54] and Clark [55] utilise the historical running speed while Zhang and Rice [45], van Lint et. al. [46], and Song et. al. [53] use the real-time running speed to predict the future speed. Real-time running speed is often used for short-term estimation [45, 46, 53]. Pan et. al. [56] takes both historical and real-time speed data to improve the algorithm to perform both short-term and long-term estimation.

Estimation algorithms using travel time as input are proposed by Yu et. al. [42], Padmanaban et. al. [44], and Lee et. al. [48]. Historical and real-time travel time of a bus travelling on a bus route are used to perform estimation of future travel time of the same bus on the same route [44, 48]. As Yu et. al. [42] aims to estimate the arrival time of the next bus at a bus stop, the travel time of buses of different routes that arrive at the same bus stop is used as the input parameter.

Time delays [43], together with the non-stopping travel time, provide the actual travel time. The causes for the time delays include the roundabouts, intersections, traffic lights, and bus stops.

Vehicle locations [41, 47, 51] are sometimes used to estimate the arrival time, dwell time, and travel time of a bus. The vehicle locations often come from a GPS device in the form of latitude and longitude.

Traffic volume [55,57–59] has also been used to estimate the on-road situation. Historical and real-time traffic volume are used to estimate the future traffic volume [55,57–59].

The estimation methods used in the literature are ANN, regression models, auto-regressive integrated moving average (ARIMA)-based methods, wavelet-based models, and different numerical models.

ANN [41,42,46,49,54,57–59] is a machine learning algorithm inspired by the biological neural system. At least two layers are available in a neural network and the weights of the connections between nodes in different layers are trained with supervised data. More explanations of ANN can be found [60]. ANN is able to learn complex relationships between the inputs and the outputs without the need to explicitly specify their relationships. However, there are multiple network design parameters such as the activation functions, number of layers, and number of hidden neurons that affect the accuracy of the network. The training data for the network also require pre-conditioning before the training process.

The regression models used in literature include a linear regression model and a non-parametric regression model, namely k-NN. Linear regression model [42, 43] models the

relationships of different variables with linear relationships. Yu et. al. [42] proposed a model which performs logarithmic transform of the variables before passing them to the regression function. The linear regression model is compared with support vector machine, ANN, and k-NN [42]. Linear regression is found to have the worst prediction performance while having the simplest structure. k-NN [42, 55, 57] method performs predictions based on the closest observations. The predicted value is obtained by taking the weighted average of the closest observations. In different comparisons [42, 57], k-NN has shown to perform similarly with ANN. The difficulty of using a k-NN model is to define a meaningful distance function. Even though k-NN does not require training before prediction, the computational time for the prediction is long as the model need to identify the neighbourhood of the point to be predicted [57] from all available observations.

ARIMA [54] and ARIMA-based [54,56] methods are used in the estimation of running speed. The ARIMA method is useful in estimating the future value of a time series. Therefore the input of the ARIMA method is the historical time series of the parameter to be estimated. The method thus cannot be used to perform prediction based on other parameters. Variations of ARIMA method are also discussed. Fractional ARIMA (FARIMA) [54] modifies the ARIMA method from a short-range dependent process to a long-range dependent model. Historical ARIMA (H-ARIMA) [56] is a hybrid method that combines ARIMA and historical average model (HAM). ARIMA has a better short-term prediction accuracy while HAM is more accurate for long-term prediction. H-ARIMA makes decision to choose from ARIMA and HAM whenever a prediction request is present.

Wavelet-based [54] methods decompose the original signal into high and low frequency components. The high frequency components are used for short-term predictions whereas low frequency components for long-term predictions. The wavelet-based methods have lower computational complexity compare to ANN and ARIMA-based methods [54], but it is harder to identify the appropriate boundary conditions that provide sufficient prediction accuracy.

The numerical methods [44, 45, 47, 48, 50–53] are derived by respective authors and not based on other general prediction algorithms. These models required in-depth understandings of the relationship between parameters. Therefore, to perform prediction on a different region requires the study of the local data and re-modelling.

Table 2.1 shows the summary of the literature that has been discussed regarding the estimation of bus journey.

The choice of output parameters is dependent on the availability of the data and the purpose of the predictions. The running speed is chosen as the output parameter in this work because the predicted output will be used for the control of the driving speed of the bus journey. The arrival time, travel time, dwell time, and traffic volume may potentially provide suitable information but further computations will be required.

The input parameters in this work are the weather data, temporal data, running speeds, vehicle locations, and social events. Examples of social events include holidays and weekends. These events affect the commuting behaviours of the road users, and therefore included as inputs.

Table 2.1: Summary of literature on bus journey estimation

	Output					**Input**									**Method**				
	Arrival time	Travel time	Dwell time	Running speed	Traffic volume	Passenger number	Weather	Temporal data	Dwell time	Running speed	Travel time	Time delays	Vehicle locations	Traffic volume	ANN	Regression models	ARIMA-based methods	Wavelet-based models	Numerical models
[41]	✓					✓	✓	✓	✓				✓		✓				
[49]		✓					✓	✓							✓				
[42]	✓										✓				✓	✓			
[43]		✓										✓				✓			
[44]		✓									✓								✓
[45]		✓								✓									✓
[46]		✓								✓					✓				
[47]		✓						✓					✓						✓
[48]		✓									✓								✓
[50]			✓			✓													✓
[51]			✓			✓							✓						✓
[52]			✓	✓					✓										✓
[53]				✓						✓									✓
[54]				✓						✓					✓		✓	✓	
[55]				✓	✓					✓				✓		✓			
[56]				✓						✓							✓		
[57]					✓									✓	✓	✓			
[58]					✓									✓	✓				
[59]					✓									✓	✓				

From the discussion of the estimation methods, ARIMA-based methods are unsuitable as they predict the parameter using only the historical values of the parameter itself. ARIMA-based methods also require a complete set of time series data to perform accurate prediction. The issue with the wavelet-base methods is the difficulty of finding appropriate boundary conditions for good prediction accuracy. The method of deriving a numerical model is unsuitable as the relationship between input and output parameters may sometimes be too complicated to be modelled. Re-modelling of the numerical models when applying the method on different regions is also undesirable. ANN and k-NN are most suitable for this work as they have shown sufficiently good performance comparing to other methods [42, 57]. Both methods do not require in-depth understandings of the re-

lationship between input and output parameters from the users. ANN is implemented as opposed to k-NN since the prediction time for k-NN is longer.

The discussed literature does not address if bus lanes exist along the route. A bus lane, sometimes referred to as the dedicated lane for buses, is a lane parallel to the mixed traffic that is only allowed to be used by buses. Bus lanes are found to reduce bus travel time [61] and improve traffic efficiency of mixed traffic lane [62–65]. Intermittent bus lanes, which are available for all traffic at certain periods of a day and only for buses at other time periods, are also proposed [66–70]. No discussion of the effect of mixed traffic lanes on bus lanes is found in the literature. Therefore, we assume that the bus lanes are considered to be always independent of the mixed traffic lanes in the literature. However, this is not the case from real-world observations.

Thus two neural networks are developed. The first neural network is used to estimate the speed on the mixed traffic lane using the spatiotemporal data, weather data, and social events. The second neural network is for estimating the bus lane speed using the mixed traffic lane speed.

2.3 Gap analysis

From the reviews of literature, we have identified the following gaps.

(i) The formulation of a bus journey which consists of multiple stops of different types and a fixed schedule for optimal control is not found. Therefore this contributes to the focus of Chapter 3 which is the formulation of a bus route. The formulated bus

route includes multiple bus stops of different types, fixed schedule, and the road slope along the route.

(ii) The relationship between the speed on the mixed traffic lane and that on the dedicated bus lane has been mentioned in literature but no supportive literature is mentioned. This contributes to the focus of Chapter 5 which is the design of a neural network to predict the bus lane speed with the mixed traffic lane speed.

Chapter 3

Optimal Control of a Bus

This chapter focuses on different components that are required to achieve the optimal control of a bus. This optimal control aims to (i) reduce the energy consumption of the bus, and (ii) improve the punctuality of the bus to arrive at each bus stop along a bus route by identifying the optimal speed profile. Aim (i) requires the acceleration and speed to be as low as possible whereas aim (ii) fixes an average speed between bus stops. Neither of these provides a mean to determine the instantaneous speed of the bus along the route. Therefore an optimal control problem which considers the energy consumption of a bus, the physical parameters of a route, and the bus schedule is employed to identify the optimal speed profile. This chapter describes the components to solve the optimal control problem: the optimal control tool, the dynamic model, and the specific problem formulation.

The main contribution in this chapter is the formulation of a bus journey with multiple bus stops of different types, fixed schedule, and slopes along the route. The formulation of a bus journey has not been found in previous literature. This chapter also introduces and defines two types of bus stops, namely the compulsory and on-demand bus stops, which will be further explained in Section 3.2.3.

The general formulation of an optimal control problem is often expressed in the Bolza form [71, 72]. The general form of a Bolza problem as described in Equation 3.1 consists of two parts, the terminal cost, $M(\cdot)$, and the path cost (also known as the running cost), $L(\cdot)$ [72]. The aim of an optimal control problem is to determine the state $x(t)$, the control $u(t)$, the initial time t_0, and the terminal time t_f that minimise the performance index J

$$J = M(x(t_0), t_0, x(t_f), t_f) + \int_{t_0}^{t_f} L(x(t), u(t), t)\, dt, \tag{3.1}$$

where $M(\cdot)$ is the terminal cost that accounts for the starting and ending of the path, and $L(\cdot)$ is the path cost that accounts for each point along the path, subject to the dynamic model

$$\dot{x}(t) = f(x(t), u(t), t), \qquad t \in [t_0, t_f] \tag{3.2}$$

where $\dot{x}(t)$ is the differential of the state variable $x(t)$, the path constraints h

$$h_L \leq h(x(t), u(t), t) \leq h_U, \qquad t \in [t_0, t_f] \tag{3.3}$$

and the boundary conditions e

$$e_L \leq e(x(t_0), u(t_0), x(t_f), u(t_f), t_0, t_f) \leq e_U. \tag{3.4}$$

The specific optimal control problem this research aims to solve is the problem of identifying the distance, speed, and acceleration profiles of a bus that minimise the energy consumption and improve bus punctuality at bus stops. The state variables, $x(t)$, are the distance and the speed of the bus whereas the control variable, $u(t)$, is the acceleration of the bus. The acceleration of a vehicle has a stronger effect on its power consumption compared to its speed. Therefore by defining the acceleration instead of the speed as the control variable it is more straightforward in identifying the optimal profile for the purpose of minimising energy consumption. Moreover, as the change in acceleration does not depend on the constraint of another variable, the decision on the current value of acceleration is not dependent on the previous value of acceleration. In practice, instantaneous acceleration can be controlled by limiting the input from the accelerator pedal of the vehicle. The performance index, also known as the objective function, J, considers the energy consumption of the bus and the arrival time at bus stops. A model of vehicle is developed to provide the computation of energy consumption. As the slope of the bus route contributes to the calculation of energy consumption, a model of road is required to provide this information. The power consumption at each point of the route is used as the path cost, $L(\cdot)$. As time, distance, and speed at terminal (beginning and ending) points are not free, no terminal cost is defined, therefore $M(\cdot) = 0$. Different constraints such as the time and distance of the starting and ending points, and the speed and acceleration limits are also defined based on the information in the model of road. A model of bus route provides information of the

distances and timetable of each of the bus stop. The time and distance of the starting and ending points are formulated in the boundary conditions, $e(\cdot)$. The speed and acceleration limits are formulated in the path constraints, $h(\cdot)$.

Section 3.1 explains the choice of the optimal control solver used in this work. The optimal control algorithm used in the solver is also described in this section.

Section 3.2 defines the different models that are required for calculation of the objective function. These include the model of the vehicle, the road and the bus route.

Section 3.3 provides the formulation of the optimal control problem. The formulation of a generic optimal control problem has been discussed at the beginning of this chapter. Specific problem formulations are discussed in this section. The relationship of the models in Section 3.2 with the objective function and constraints is also explained.

Section 3.4 provides a case study for the bus route from Oxford City Centre to Pear Tree Park and Ride. The parameters of the specific models for this case are defined. The formulation of the optimal control problem is described and the solution to the problem is discussed.

3.1 Optimal control algorithm and program

In this section we will identify the suitable algorithm and program to solve the optimal control problem for a bus journey. The different optimal control algorithms are described and the algorithm to be used in this work is selected based on a comparison done by Rao [73]. Three criteria are set up to determine the suitable optimal control program from the

five programs that implement the selected optimal control algorithm.

3.1.1 Optimal control algorithm

The methods to solve an optimal control problem fall into two major categories, the indirect and direct method [73, 74]. The indirect method solves the optimal control problem by converting it into a multiple point boundary value problem. The direct method, on the other hand, approximates the control, sometimes the states as well, to transcribe the optimal control problem to a non-linear optimisation or non-linear programming (NLP) problem and find the minimum of the objective function. The indirect method solves the optimal control problem as a continuous-time problem whereas the direct method solves it as a discrete-time problem.

In each of the two categories, there are three common methods: the shooting method, the multiple-shooting method, and the collocation method [73]. An initial guess is provided to the shooting method to calculate the trajectory and the performance index. The guess is then updated to reduce the performance index. This process is repeated until the cost is at a minimum while satisfying the constraints. The multiple-shooting method divides the time interval into subintervals and performs the shooting method over each subinterval. The collocation method also divides the time interval into subintervals. However, within each subintervals the states (and the control, in direct method) are approximated with different functions and the coefficients in the functions are determined to find the solution of the problem. More detailed explanations for different methods to solve optimal control

problem can be found in the work by Rao [73].

Rao [73] concludes that even though an indirect shooting method is able to produce highly accurate solutions when it converges, the fact that the method requires the derivation of the first-order optimality conditions causes the difficulty to implement the method in a general-purpose software program. Büskens and Maurer [75] and Wang et. al. [76] also comment on the difficulty of indirect shooting method to converge without a precise initial guess. The direct shooting method is good at solving a relatively simple problem but the direct collocation method is the preferred method as the complexity of the problem increases. This is because the direct collocation method formulates the problem to be solved with NLP solvers. NLP solvers are designed to converge with poor initial guesses and are computationally efficient, hence they are able to solve highly complex problem.

Therefore the direct collocation method is selected as the optimal control algorithm. Therefore the optimal control problem will be formulated as a discrete-time problem. There are multiple optimal control programs that implement the direct collocation method, namely PSOPT, GPOPS-II, JModelica, APMonitor, and Bocop. These programs are discussed and compared in the next section to identify the program that suits for this work.

3.1.2 Optimal control program

A number of optimal control programs have been developed to solve large scale optimal control problems. In order to make the choice of the program that suits this work, three criteria are set up to evaluate the suitability of the programs. These criteria are (i) the use

of a quality solver, (ii) the ability to interface with Python, and (iii) the ease to define the optimal control problem. The use of a quality optimal control solver allows the confidence in the results since this work does not focus on developing the solver algorithm. As the rest of this work is done in Python, it is important for the program to be able to interface with Python to allow the exchange of data. The ease to define the optimal control problem allows the problem to be set up without much hassle such as learning a new modelling language.

As mentioned in the previous section, the direct collocation method is chosen and the programs that implement the direct collocation method are PSOPT, GPOPS-II, JModelica, APMonitor, and Bocop.

PSOPT [77], developed by V. M. Becerra, is an open source optimal control program. The program is intended to solve complex optimal control problem using direct collocation method and an NLP solver. GPOPS-II [78], developed by M. Patterson and A. Rao, is designed as a general purpose program to solve continuous optimal control problem in MATLAB. JModelica [79] is an open source program for optimisation, simulation, and analysis of dynamic system. The program is based on the Modelica modelling language and it makes use of the Optimica extension of Modelica to perform dynamic optimisation. APMonitor [80] is a server-based optimisation program designed to be accessible through MATLAB, Python or web browsers. Bocop [81] is another open source program to solve optimal control problems. Bocop is a graphic user interface (GUI) oriented program. Even though the program can also be executed using command line interface, its developers

recommend the use of GUI.

The first criterion to choose the optimal control program is that the program uses a quality solver. Each of these programs uses one or more NLP solvers to perform the optimisation after the optimal problem is transcribed to an NLP problem. The solvers used by each program is listed in Table 3.1. IPOPT [82] is an open source interior point optimiser designed for large-scale non-linear optimisation under COIN-OR Foundation [83]. SNOPT [84], created by P. Gill, W. Murray and M. Saunders, is designed to solve large-scale constrained optimisation problems. SNOPT is available as a commercial product with commercial or academic licences. KNITRO [85, 86] is designed by the company Ziena Optimization for the purpose of solving non-linear optimisation problems. KNITRO is available as a commercial product with three versions: student, academic, or university version. Not much information can be found on APOPT and BPOPT. A web page [87] is dedicated for APOPT but no detail of the underlying algorithm used in the solver or whatsoever is available on the web page. A reference to a paper in a national meeting is available in the publication for APMonitor [80]. However, it cannot be found on the internet despite the availability of web link in the reference. There is even less information on BPOPT solver. The search of BPOPT on the internet leads to another optimisation algorithm named BPOpt [88]. Since no information on APOPT and BPOPT can be found, the options of the solver are reduced to IPOPT, SNOPT, and KNITRO. IPOPT is chosen based on two reasons. First, SNOPT and KNITRO both require the purchase of licence and therefore the uses of these solvers might be limited for future purposes. Second, IPOPT is

Table 3.1: Solvers used by the optimal control programs

Program	Solver				
	IPOPT	SNOPT	KNITRO	APOPT	BPOPT
PSOPT	✓	✓			
GPOPS-II	✓	✓	✓		
JModelica	✓				
APMonitor	✓			✓	✓
Bocop	✓				

used in all the programs discussed here. Therefore it is safe to say that IPOPT is widely accepted as a good NLP solver.

The decision on selecting IPOPT as the NLP solver for this work has not eliminated any of the listed programs to be used as the optimal control program.

The second criterion evaluates the ability of the program to interface with Python. The main purpose to interface with Python is so that the parameters for an optimal control problem can be defined in Python and passed to the program. PSOPT requires the problem to be defined as C++ functions. There are two ways to allow Python to interact with the problem definition. The first is to generate the C++ functions using a Python script with predefined C++ function templates. The second is to define and compile a generic optimal control problem in C++ such that the compiled C++ program accepts the parameters definitions as input arguments. GPOPS-II is designed as a MATLAB toolbox. Therefore it is tightly integrated in MATLAB software and difficult to interface with Python. JModelica and APMonitor are both designed with the integration of Python in the plan. Both programs provide Python libraries that can be used from within Python to define the problem and start the execution of the programs. As the design of Bocop is GUI oriented, it is designed

to be a standalone program and thus difficult to interface with Python. Hence, the list of feasible programs is reduced to PSOPT, JModelica, and APMonitor.

The third criterion to be considered is the ease to define an optimal control problem using the program. The problem definition in PSOPT is in the format of C++ functions. In APMonitor, the model and the problem is defined using the APMonitor modelling language. JModelica requires the model and the problem to be defined in Modelica [89] and Optimica [90]. APMonitor modelling language, Modelica, and Optimica are languages designed with a more specific purpose than C++. APMonitor modelling language is designed for APMonitor and therefore one has to learn the language when first using the program. Modelica is an object-oriented language designed for the purpose of modelling complex physical system. Optimica is an extension of the Modelica language developed for the purpose to optimise the parameters of a model modelled in Modelica. Therefore, unless one has experience with Modelica previously, the use of JModelica involves the learning of a new language. As C++ is a general purpose language and we have more experiences in C++ compare to the other three languages, it is easiest to formulate an optimal control problem in PSOPT compared to APMonitor and JModelica.

Hence, PSOPT is chosen as the optimal control program to be used in this work since (i) it uses a quality NLP solver: IPOPT, (ii) it is able to interface with Python, and (iii) it is relatively easier to define an optimal control problem. As mentioned, there are two ways to interface PSOPT with Python. The second approach in which a generic problem is defined in C++ and the definitions of the parameters are passed to the compiled C++ program as

input arguments is used. This avoids the re-compilation of a C++ program when there are changes in the parameters.

3.2 Models

Models are developed for the purpose of introducing parameters from different aspects to the optimal control problem. To solve an optimal control problem of a bus, there are three basic models that are required: the model of vehicle, road, and bus route.

3.2.1 The model of vehicle

The model of vehicle used in this work is based on the Oxford Vehicle Model (OVEM) [91], a MATLAB based modelling tool. OVEM takes the input of a speed profile and calculates the acceleration implicitly. However, for the purpose of controlling the acceleration of the vehicle, the model is modified to accept instantaneous acceleration and speed as the inputs.

Various simulation methods like backwards, forwards and backwards-forwards simulations have been discussed in OVEM. Each of the methods have its advantages and drawbacks. Backwards simulation is used in this model as it imposes no performance limit. This is essential as this work only concerns the energy consumption of the vehicle given the acceleration and speed profiles instead of the capability of the vehicle to achieve the requirements. To overcome this concern, all the solutions developed are checked to ensure this specific bus is capable of delivering the profile, therefore negating the requirement of

having the performance limit in the model.

Backwards simulation passes the power requirement either in mechanical form, $\tau \cdot \omega$ where τ is the torque and ω is the angular speed, or electrical form, $I \cdot V$ where I is the current and V is the voltage, through the powertrain of a vehicle in the reverse direction of the actual power flow. Figure 3.1 shows the parameter flow of backwards simulation in the powertrain. For simplicity, the powertrain of a pure EV is used. The vehicle body accepts the vehicle speed and acceleration as inputs. The vehicle body considers the aerodynamic torque, the rolling friction torque, the gravitational torque, and the rolling inertia to calculate the torque and angular speed requirement of wheel and axle. The wheel-axle block uses a constant efficiency to calculate the torque and angular speed requirement of the gearbox. The gearbox calculates the torque and speed requirement of the motor by considering the losses due to bearing, meshing, and seal. The motor block then calculates the voltage and current requirement of the power electronics using the model of a general equivalent brushless direct-current motor. The power electronics block calculates the power demand from the ESS by considering the conduction and switching losses. The ESS block is adapted from the battery model discussed by Tremblay et. al. [94].

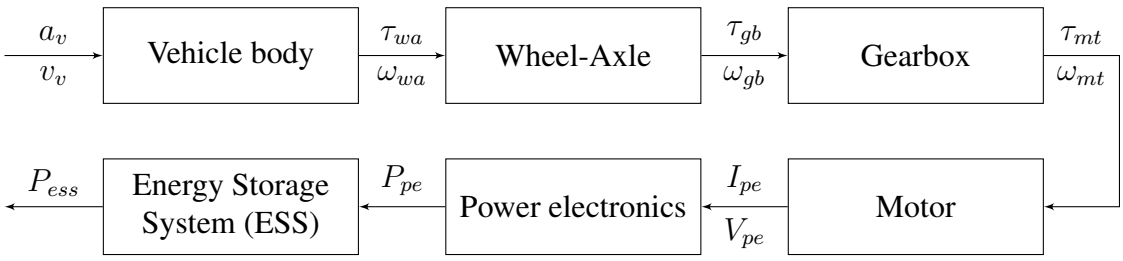


Figure 3.1: Parameter flow in backwards simulation

Vehicle Body

The vehicle body block accepts inputs of instantaneous acceleration and velocity of the vehicle and models the vehicle dynamics. The aerodynamic torque τ_D, the rolling friction torque τ_r, the gravitational torque τ_g, and the rolling inertia τ_{iner} are calculated using the acceleration and velocity together with the slope of the road θ in Equations 3.5 to 3.9. τ_v is the vehicle torque, m_v is the vehicle mass, r_w is the vehicle wheel radius, a_v is the vehicle acceleration, ρ is the density of air, C_d is the drag coefficient, A_f is the frontal area of the vehicle, v_v is the vehicle speed, C_r is the rolling resistance coefficient, g is the gravitational acceleration, $theta$ is the road slope in degree, J_w is the wheel-axle moment of inertia, α is the angular acceleration of the vehicle, f_0 is the basic coefficient for rolling resistance coefficient, f_s is the speed effect coefficient for rolling resistance coefficient.

$$\tau_v = m_v r_w a_v \tag{3.5}$$

$$\tau_D = \frac{1}{2} r_w \rho C_d A_f v_v^2 \tag{3.6}$$

$$\tau_r = r_w C_r m_v g \cos\theta \tag{3.7}$$

$$\tau_g = r_w m_v g \sin\theta \tag{3.8}$$

$$\tau_{iner} = J_w \alpha \tag{3.9}$$

$$\text{where} \quad C_r = f_0 + 3.24 f_s (0.0223 v_v)^{2.5} \tag{3.10}$$

The formula for the rolling resistance coefficient C_r (Equation 3.10) is based on Gillespie's work [92]. The power demand in the form of wheel-axle torque, τ_{wa} and wheel-axle angular speed, ω_{wa} to be passed to the wheel-axle block is calculated with Equations 3.11 and 3.12.

$$\tau_{wa} = \tau_v + \tau_D + \tau_r + \tau_g + \tau_{iner} \tag{3.11}$$

$$\omega_{wa} = \frac{v_v}{r_w} \tag{3.12}$$

Wheel-Axle

The wheel-axle block identifies the torque and speed demands of the gearbox block τ_{gb} and ω_{gb} based on the requests from the vehicle body block τ_{wa} and ω_{wa}. The wheel-axle block models the axle as a constant efficiency component which transmits power from gearbox to the wheel. At acceleration, torque demand of the gearbox block τ_{gb} is the division of τ_{wa} by the wheel-axle efficiency η_{wa}. At deceleration with regenerative brake, τ_{gb} is the multiplication of τ_{wa} by η_{wa}. When regenerative brake is not activated, τ_{gb} is zero at deceleration. The speed demand does not change when passed through the wheel-axle block.

$$\tau_{gb} = \begin{cases} \frac{\tau_{wa}}{\eta_{wa}}, & \text{for } \tau_{wa} \geq 0 \\ \begin{cases} \eta_{wa}\tau_{wa}, & \text{if regenerative} \\ 0, & \text{otherwise} \end{cases}, & \text{for } \tau_{wa} < 0 \end{cases} \tag{3.13}$$

$$\omega_{gb} = \omega_{wa} \tag{3.14}$$

Gearbox

The gearbox block calculates the torque demand of the motor τ_{mt} by considering the torque demand of the gearbox τ_{gb} and the losses due to the bearing, meshing and seal. These losses are quantified by the coefficients bearing coefficient k_b, meshing coefficient k_m, and seal coefficient k_s. Then the torque and speed demands are calculated based on the gearbox ratio N_{gb}. Equations 3.15 and 3.16 show the formulae to calculate the torque and speed demand of the motor. The speed demand is denoted by ω_{mt}

$$\tau_{mt} = \frac{\tau_{gb} + k_s\tau_{maxgb} + (k_m + k_b)\tau_{gb}}{N_{gb}} \tag{3.15}$$

$$\omega_{mt} = \omega_{gb}N_{gb} \tag{3.16}$$

Motor

A general equivalent brushless direct-current motor is modelled from the equivalent circuit [93] which consists of a resistor, inductor and motor as shown in Figure 3.2. As the inputs of the model are instantaneous acceleration and speed, the transient current is neglected, and therefore the inductor is omitted in Equations 3.17 to 3.19.

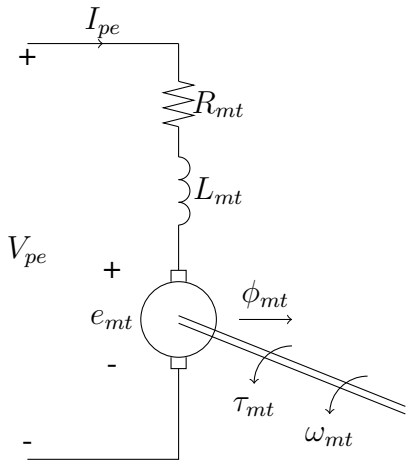


Figure 3.2: Equivalent circuit of a direct-current motor

Torque and speed demands of the motor block τ_{mt} and ω_{mt} are translated into the motor current I_{pe} and the motor voltage e_{mt} using Equations 3.17 and 3.18.

$$I_{pe} = \frac{\tau_{mt}}{k_t \phi_{mt}} \tag{3.17}$$

$$e_{mt} = k_e \phi_{mt} \omega_{mt} \tag{3.18}$$

Torque constant of the motor k_t and voltage constant of the motor k_e are numerically

equal and denoted as k_{mt}. The voltage demand of the power electronics V_{pe} can then be formulated as in Equation 3.19 by applying Kirchhoff's Voltage Law and Ohm's Law. The power demands in electrical form passed to the power electronics block are I_{pe} and V_{pe}. Only resistive loss is considered in Equation 3.19. Constant losses are neglected as it does not have an effect on the identification of the optimal profile.

$$V_{pe} = k_{mt}\phi_{mt}\omega_{mt} + I_{pe}R_{mt} \tag{3.19}$$

where R_{mt} is the internal resistance in the motor block.

Power Electronics

The power electronics block models the conduction loss and the switching loss in the power electronics. The conduction loss is given by Equation 3.20 and the switching loss in Equation 3.21. The power demand for energy storage system (ESS) P_{ess} is given by Equation 3.22 as the summation of the losses in power electronics and the power demand for the power electronics.

$$P_{cd} = k_{cd}I_{pe}^2 \tag{3.20}$$

$$P_{sw} = k_{sw}I_{pe}V_{pe} \tag{3.21}$$

$$P_{ess} = I_{pe}V_{pe} + P_{cd} + P_{sw} \tag{3.22}$$

where P_{cd} is the power electronic conduction loss, k_{cd} is the coefficient of conduction losses, P_{sw} is the power electronic switching loss, and k_{sw} is the coefficient of switching losses.

Energy Storage System

As the power train of a pure EV is to be modelled, the ESS discussed here is a battery. The battery model in Figure 3.3 is adapted from the work by Tremblay et. al. [94]. The battery's no-load voltage E can be calculated from the battery's constant voltage E_0 and its state of charge using Equation 3.23. The current flow through the battery I_{ess} can be calculated with Equation 3.24. The power consumed from ESS P_c is given by Equation 3.25.

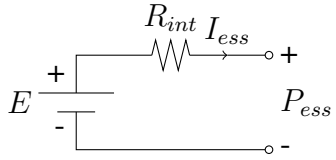


Figure 3.3: Battery model

$$E = E_0 - K\frac{Q}{Q - \int_0^t i\,dt} + Ae^{-B\int_0^t i\,dt} \tag{3.23}$$

$$I_{ess} = \begin{cases} \frac{E-\sqrt{E^2-4R_{int}P_{ess}}}{2R_{int}}, & \text{discharging} \\ \frac{-E+\sqrt{E^2+4R_{int}P_{ess}}}{2R_{int}}, & \text{charging} \end{cases} \tag{3.24}$$

$$P_c = I_{ess}E \tag{3.25}$$

where E is the no-load voltage of the battery (V)

E_0 is the constant voltage of the battery (V)

K is the polarisation voltage (V)

Q is the capacity of the battery (Ah)

$\int_0^t i\,dt$ is the actual battery charge removed (Ah) from time = 0 to current time t

A is the exponential zone amplitude (V)

B is the exponential zone time constant inverse (Ah^{-1})

R_{int} is the internal resistance of the battery (Ω)

I_{ess} is the ESS current (A)

P_{ess} is the power demand for ESS (Nm/s)

P_c is the consumed power of ESS (Nm/s)

As mentioned at the beginning of Section 3.2.1, there should be no performance limit in the model as the model should only provide power consumption based on the acceleration and speed of the vehicle. There are two ways to achieve this with the current ESS model. One is to provide a huge battery that the battery's limit will never be exceeded. From Equation 3.23 this implies that E^2 must be greater than the largest value of $4R_{int}P_{ess}$ in the simulation. Two is to use a dynamic E such that E^2 always equals to $4R_{int}P_{ess}$, i.e. the battery always operates at maximum capacity.

The implication of case one, i.e. using a very huge value of E, is that the effect of internal resistance in Figure 3.3 becomes negligible due to small value of current I_{ess} and the power consumption is close to P_{ess}. This effect can be solved by simulating using the acceleration and speed profiles without the ESS block to identify the maximum value of P_{ess}. This P_{ess} value can then be used to determine the value of E which is sufficient for the profiles. However, as this model is used to solve an optimal control problem, each iteration will provide new acceleration and speed profiles. This approach will need to compute a new value of E at each iteration. That means the power consumptions at different iteration are not comparable. This will then introduce an additional undesired variable into the optimal control problem.

The second approach that uses a dynamic value of E, according to Equations 3.26 to 3.32, the effect of the internal resistance will be maximised. The power consumption calculated by the ESS block will always be double of the power consumption calculated by the power electronics block. The results of solving the optimal control problem with and without the ESS block will thus be identical.

$$I_{ess} = \frac{E - \sqrt{E^2 - 4R_{int}P_{ess}}}{2R_{int}} \qquad \text{for discharging} \tag{3.26}$$

$$= \frac{2\sqrt{R_{int}P_{ess}}}{2R_{int}} \qquad \text{as } E^2 = 4R_{int}P_{ess} \tag{3.27}$$

$$= \sqrt{\frac{P_{ess}}{R_{int}}} \tag{3.28}$$

$$P_c = I_{ess}E \tag{3.29}$$

$$= \sqrt{\frac{P_{ess}}{R_{int}}} E \tag{3.30}$$

$$= \sqrt{\frac{P_{ess}}{R_{int}} 4 R_{int} P_{ess}} \tag{3.31}$$

$$= 2P_{ess} \tag{3.32}$$

Therefore the ESS block is not considered in the simulation model in the later part of this chapter. The total power consumption is taken to be the power demand calculated in the power electronics block.

3.2.2 The model of road

The model of road is responsible to convey the features of a route to the optimal control problem. The model of road includes the total distance and time of the route, and the slope profile along the route. Junctions and turns are not included in the model as turnings are not modelled in the model of vehicle described in Section 3.2.1.

The total distance d_T and time t_T of the route are first defined. The definition of the slope profile must have a distance less than the total distance d_T.

Slope profile instead of elevation profile is defined since the model of vehicle is able to use the slope directly without conversion. The slope is expressed as the angle of inclination to the horizontal surface θ_i at the distance at which the slope starts d_i. The two dimensional array that defines the slope profile of a slope with N slope changes is shown as S.

$$\mathsf{S} = [[d_0, \theta_0], [d_1, \theta_1], [d_2, \theta_2], \ldots, [d_{N-1}, \theta_{N-1}]] \quad \text{, with} \quad d_{N-1} < d_T \tag{3.33}$$

3.2.3 The model of bus route

The model of bus route describes the location of bus stops, the category of bus stops and the time components related to each bus stop. The three time components for each bus stop are the time of bus arrival, time tolerance of bus arrival and the stopping period at the bus stop.

The bus stop locations and bus arrival time are limited by the total distance and time defined in the model of road. The locations of the bus stops must lie within the full length of the route, and the bus arrival time must be less than or equal to the total time.

The model of bus route is defined as a two dimensional array. For a bus trip with N_b bus stops, the array is defined as BS.

$$\begin{aligned} \mathtt{BS} = [[t_0, \quad & d_0, \quad c_0, \quad \delta_0, \quad \Delta_0 \quad], \\ [t_1, \quad & d_1, \quad c_1, \quad \delta_1, \quad \Delta_1 \quad], \\ [t_2, \quad & d_2, \quad c_2, \quad \delta_2, \quad \Delta_2 \quad], \\ & \ldots \\ [t_{N-1}, & d_{N-1}, c_{N-1}, \delta_{N-1}, \Delta_{N-1}]] \end{aligned} \tag{3.34}$$

where t denotes the time to arrive at the bus stop,

d denotes the location of bus stops,

c denotes the bus stop categories,

δ denotes the time tolerance for the time of arrival,

Δ denotes the stopping time period at the bus stop.

The location of a bus stop along the route d is defined as the distance of the bus stop measured from the beginning of the route, $d = 0$. The list of bus stop should be arranged in the order of increasing distance so the order of the list corresponds to the order of the bus reaching the bus stops.

The category of a bus stop c is either a compulsory stop or an on-demand stop. These categories are defined based on the compulsoriness of a bus to stop at the bus stop. A compulsory bus stop implies that the bus should always stop and wait at the bus stop. An on-demand stop, on the other hand, requires the bus to stop at the bus stop only if there is a demand to alight or board the bus. The category of a bus stop also affects the way the stopping time period Δ is processed.

As the bus stop locations are defined in the order of increasing distance, the order of bus arrival time is in chronological order. The highest value for the arrival time t_{N_b-1} must be less than or equal to the total time t_T defined in the route model. The following condition of t must be satisfied.

$$t_0 < t_1 < t_2 < \ldots < t_{N-1} \leq t_T$$

The time tolerance δ is the tolerance of arrival time. This provides a buffer to the arrival time of the bus which relaxes the solution to the optimal control problem. With the bus arrival time as t_i and time tolerance as δ_i, the bus should arrive at the bus stop between $t_i - \delta_i$ and $t_i + \delta_i$.

The stopping period Δ defines the length of time period a bus should stay at the bus stop. This parameter is handled differently for different categories of bus stop. For an on-demand bus stop this has no effect as the bus is not required to stop, but for a compulsory bus stop this defines the departure time of the bus from the bus stop. With time of arrival as t_i and the stopping period as Δ_i, the departure time from the compulsory bus stop is $t_i + \Delta_i$.

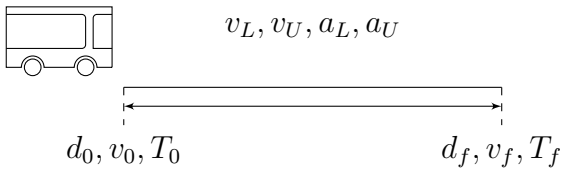
3.3 Optimal control problem formulation

The general formulation of an optimal control problem has been discussed in the beginning of this chapter. As mentioned earlier, the specific optimal control problem to be solved is to identify the distance, speed and acceleration profiles to minimise the energy consumption and improve punctuality at bus stops. The distance and speed of the bus are chosen as the state variables and the acceleration as the control variable due to the dependency of distance and speed profiles on the acceleration profile. This section starts in discussing

the problem formulation for a basic route with no bus stop (Section 3.3.1), and then introducing different types of bus stops separately (Sections 3.3.2 and 3.3.3) before introducing the changes in slope along the route (Section 3.3.4). Section 3.3.5 presents a case where different types of bus stops and slope changes exist on the same bus route.

3.3.1 Basic route

For a bus route with a single stop, i.e. at the end, as shown in Figure 3.4, the variables are defined in Equations 3.35 to 3.38 and the objective function is defined as Equation 3.39. As the minimisation of journey time is not necessary, the endpoint cost (first term in the Equation 3.1) is not required. As the destination of a bus route is a bus stop in most cases, the finishing time is set to be the maximum time instead of a flexible time to be determined.



where d_0 is the starting distance, d_f is the final distance,
v_0 is the starting speed, v_f is the final speed,
T_0 is the starting time, T_f is the maximum time,
v_L is the lower speed bound, v_U is the upper speed bound,
a_L is the lower acceleration bound, a_U is the upper acceleration bound

Figure 3.4: Basic bus route

The control and state variables are defined as

$$u(t) = a(t), \tag{3.35}$$

$$x_1(t) = v(t), \tag{3.36}$$

$$x_2(t) = d(t), \text{ and} \tag{3.37}$$

$$x(t) = \begin{bmatrix} x_1(t) \\ \\ x_2(t) \end{bmatrix} \tag{3.38}$$

with objective function

$$J = \int_{t_0}^{t_f} V(x(t), u(t), t)\, dt \tag{3.39}$$

where $V(\cdot)$ is the instantaneous vehicle power consumption provided by the model of the vehicle defined in Figure 3.1 ($V(\cdot) = P_{ess}$ in Equation 3.22), subject to the dynamic constraints

$$\dot{x}_1(t) = u(t) \text{ and} \tag{3.40}$$

$$\dot{x}_2(t) = x_1(t)\ , \tag{3.41}$$

the path constraints

$$v_L \quad \leq \quad x_1(t) \quad \leq \quad v_U \quad , \tag{3.42}$$

$$d_0 \quad \leq \quad x_2(t) \quad \leq \quad d_f \quad \text{, and} \tag{3.43}$$

$$a_L \quad \leq \quad u(t) \quad \leq \quad a_U \quad , \tag{3.44}$$

and the boundary conditions

$$x_1(t_0) = v_0, \tag{3.45}$$

$$x_1(t_f) = v_f, \tag{3.46}$$

$$x_2(t_0) = d_0, \tag{3.47}$$

$$x_2(t_f) = d_f, \tag{3.48}$$

$$t_0 = T_0, \text{ and} \tag{3.49}$$

$$t_f = T_f. \tag{3.50}$$

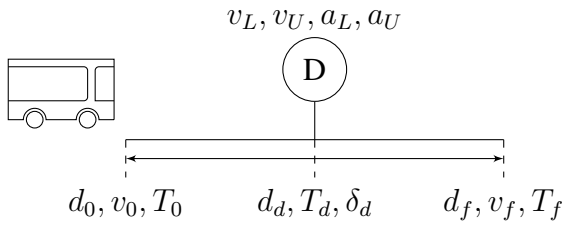
The equality boundary conditions (Equations 3.45 to 3.50) can be specified as double-sided inequalities as required in PSOPT.

The path cost $V(\cdot)$ in the objective function shown in Equation 3.39 can be simplified as the function shown in Equation 3.51. Therefore it is likely that $V(\cdot)$ is not convex. However, in the latter part of this chapter, identical solutions can be found using different initial guesses, showing that there exists optimality for this problem.

$$V(x(t), u(t), t) = x_1^5 + x_1^{4.5} + \ldots \tag{3.51}$$

3.3.2 Route with an on-demand bus stop

When an on-demand bus stop with estimated arrival time T_d and time tolerance δ_d is introduced into the same bus route at distance d_d as in Figure 3.5, the objective function is modified to accommodate the bus stop. As mentioned in Section 3.2.3, a bus is not required to stop at an on-demand bus stop unless requested. Therefore the problem is defined such that the bus arrives at the bus stop at $T_d \pm \delta_d$. This is achieved by introducing an extra term in the objective function such that the problem minimises the difference between the distance travelled by bus and the distance of the bus stop within $T_d \pm \delta_d$. The magnitude of the difference between the two distances D can be found by using Equation 3.52.



where $d_0, d_f, v_0, v_f, T_0, T_f, v_L, v_U, a_L, a_U$ inherited from Figure 3.4
d_d is the location of the bus stop,
T_d is the arrival time of bus at the bus stop,
δ_d is the arrival time tolerance

Figure 3.5: Bus route with an on-demand bus stop

$$D = (x_2(t) - d_d)^2 \tag{3.52}$$

The squared of the difference instead of its absolute value is used to preserve the smoothness of the objective function. As only the difference within time period of $T_d \pm \delta_d$ is concerned, Equation 3.52 is multiplied with a rectangular function shown in Figure 3.6. This activates Equation 3.52 only within the range of $T_d - \delta_d$ to $T_d + \delta_d$.

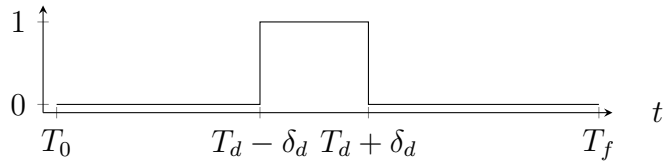


Figure 3.6: Rectangular function activated within $T_d \pm \delta_d$

As the rectangular function is a discontinuous function, it is represented with the compound of two smooth heaviside functions as shown in Figure 3.7.

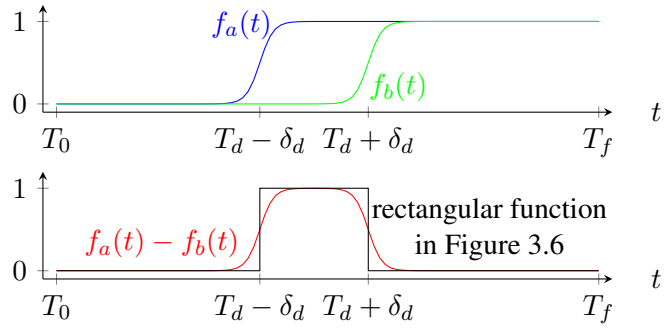


Figure 3.7: Compound of two smooth heaviside functions to form a smoothed rectangular function

A smooth heaviside function is given as

$$H(x) = 0.5(1 + \tanh(\frac{x}{a})) \tag{3.53}$$

where a is a small positive real number. $f_a(t)$ and $f_b(t)$ are obtained by applying function transformation to move the transition of a smooth heaviside function from zero to $T_d - \delta_d$ and $T_d + \delta_d$ respectively.

$$f_a(t) = H(t - (T_d - \delta_d)) \tag{3.54}$$

$$f_b(t) = H(t - (T_d + \delta_d)) \tag{3.55}$$

The modified objective function now takes the form in Equation 3.56.

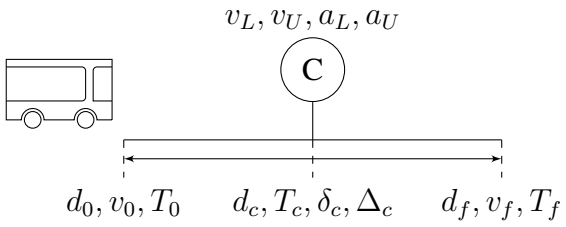
$$J = \int_{t_0}^{t_f} V(x(t), u(t), t) + w_d(x_2(t) - d_d)^2[H(t - (T_d - \delta_d)) - H(t - (T_d + \delta_d))]\, dt \tag{3.56}$$

where w_d is the weight of the extra term.

This approach scales up easily with more on-demand bus stops. Each additional bus stop contributes to an extra term in the objective function.

3.3.3 Route with a compulsory bus stop

The bus route formed by introducing a compulsory bus stop into the basic route in Section 3.4 is shown in Figure 3.8. The compulsory bus stop is located at distance d_c with the



where $d_0, d_f, v_0, v_f, T_0, T_f, v_L, v_U, a_L, a_U$ inherited from Figure 3.4

d_c is the location of the bus stop,

T_c is the arrival time of bus at the bus stop,

δ_c is the arrival time tolerance,

Δ_c is the stopping time period at the bus stop

Figure 3.8: Bus route with a compulsory bus stop

estimated arrival time $T_c \pm \delta_c$ and the stopping period Δ_c. The resultant speed profile between $T_c - \delta_c$ to $T_c + \Delta_c$ is expected to be in the form as Figure 3.9.

The joint points at $T_c \pm \delta_c$ and $T_c + \Delta_c$ are clearly not smooth. Therefore a multiphase problem is formulated. The formulation of a multiphase problem is similar to that of a single phase problem as shown at the beginning of Section 3.3. The problem is separated into different phases of time in which each phase is a single phase problem with additional phase linkage constraints. The phase linkage constraints define the constraints to be obeyed at the connecting points of two phases. In this case, the phase linkage constraints preserve the continuity of the optimal profile.

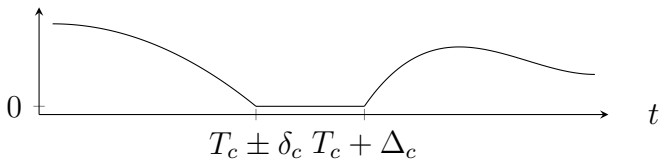


Figure 3.9: Expected speed profile between $T_c - \delta_c$ and $T_c + \Delta_c$

For bus route in Figure 3.8 with expected speed profile in Figure 3.9, it is obvious that three phases will be formulated in this situation, one before $T_c \pm \delta_c$, one between $T_c \pm \delta_c$ and $T_c + \Delta_c$, and one after $T_c + \Delta_c$.

The objective function, the dynamic constraints and the path constraints are identical to that of the basic route whereas the boundary conditions are modified according to the parameters in each phase.

In phase one, the bus is supposed to reach and stop at distance d_c from d_0 at time $T_c \pm \delta_c$. The boundary conditions to be fulfilled are

$$x_1^{(1)}(t_0^{(1)}) = v_0, \tag{3.57}$$

$$x_1^{(1)}(t_f^{(1)}) = 0, \tag{3.58}$$

$$x_2^{(1)}(t_0^{(1)}) = d_0, \tag{3.59}$$

$$x_2^{(1)}(t_f^{(1)}) = d_c, \tag{3.60}$$

$$t_0^{(1)} = T_0, \text{ and} \tag{3.61}$$

$$T_c - \delta_c \leq t_f^{(1)} \leq T_c + \delta_c. \tag{3.62}$$

The superscript in the bracket denotes the phase number of the variables. During phase two the bus will stay at halt from $t_f^{(1)}$ to $T_c + \Delta_c$. The boundary conditions at phase two are

$$x_1^{(2)}(t_0^{(2)}) = 0, \tag{3.63}$$

$$x_1^{(2)}(t_f^{(2)}) = 0, \tag{3.64}$$

$$x_2^{(2)}(t_0^{(2)}) = d_c, \tag{3.65}$$

$$x_2^{(2)}(t_f^{(2)}) = d_c, \tag{3.66}$$

$$T_c - \delta_c \leq t_0^{(2)} \leq T_c + \delta_c, \text{ and} \tag{3.67}$$

$$t_f^{(2)} = T_c + \Delta_c. \tag{3.68}$$

For the purpose of reducing computation, the path constraint for the control (acceleration) is set to zero in phase two ($0 \leq u^{(2)}(t^{(2)}) \leq 0$). This prevents the solver to look for different combinations of control that fulfil the boundary conditions of states.

The bus is expected to start moving in phase three at time $T_c + \Delta_c$ and reach the destination d_f at time T_f. The boundary conditions for this phase are

$$x_1^{(3)}(t_0^{(3)}) = 0, \tag{3.69}$$

$$x_1^{(3)}(t_f^{(3)}) = v_f, \tag{3.70}$$

$$x_2^{(3)}(t_0^{(3)}) = d_c, \tag{3.71}$$

$$x_2^{(3)}(t_f^{(3)}) = d_f, \tag{3.72}$$

$$t_0^{(3)} = T_c + \Delta_c, \text{ and} \tag{3.73}$$

$$T_f \leq t_f^{(3)} \leq T_f. \tag{3.74}$$

The phase linkage constraints for this problem (Equations 3.75 to 3.82) are formed such that the starting values for acceleration, speed, distance and time in each phase are

identical to the endpoint values of the previous phase.

$$x_1^{(1)}(t_f^{(1)}) - x_1^{(2)}(t_0^{(2)}) = 0 \tag{3.75}$$

$$x_1^{(2)}(t_f^{(2)}) - x_1^{(3)}(t_0^{(3)}) = 0 \tag{3.76}$$

$$x_2^{(1)}(t_f^{(1)}) - x_2^{(2)}(t_0^{(2)}) = 0 \tag{3.77}$$

$$x_2^{(2)}(t_f^{(2)}) - x_2^{(3)}(t_0^{(3)}) = 0 \tag{3.78}$$

$$u^{(1)}(t_f^{(1)}) - u^{(2)}(t_0^{(2)}) = 0 \tag{3.79}$$

$$u^{(2)}(t_f^{(2)}) - u^{(3)}(t_0^{(3)}) = 0 \tag{3.80}$$

$$t_f^{(1)} - t_0^{(2)} = 0 \tag{3.81}$$

$$t_f^{(2)} - t_0^{(3)} = 0 \tag{3.82}$$

With the implementation of phase linkage constraints the starting time in phase two will align with the ending time in phase one. Phase two is an over-constrained problem as the starting and the ending values of x_2 (distance) are identical. This situation makes sure the values of x_1 (speed) throughout the phase to be zero.

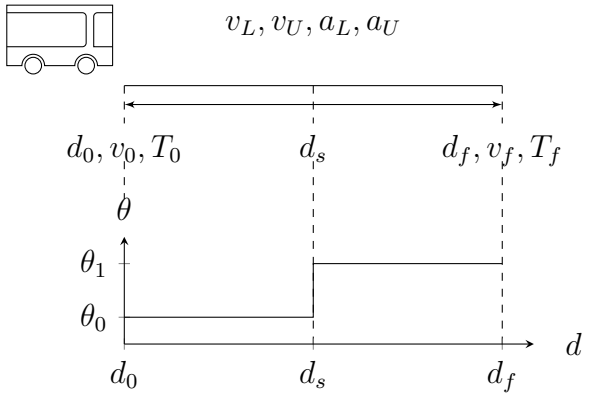
An alternative to formulate this multiphase approach is to use a hybrid automaton [95]. A hybrid automaton can be used to model a dynamical system with discrete and continuous components. A hybrid automaton is essentially a state machine. The discrete components are modelled by the switching between states. The variables within a state models the continuous components. Specifically to the multiphase approach to model the compulsory bus stops, each phase can be viewed as individual state with the distance travelled by bus

and time spent at the compulsory bus stop as the jump conditions (the conditions that trigger the transition of states). Examples of optimal control of hybrid automata are shown in multiple literature [96, 97].

3.3.4 Route with different slopes

The bus routes prior to this section are considered to be on a flat surface, i.e. having zero slope along the entire route. This section will thus discuss the effect of introducing slope changes in a basic route on the optimal control problem formulation.

Consider a basic route with a single stop at the end that has one change of slope as shown in Figure 3.10.



where $d_0, d_f, v_0, v_f, T_0, T_f, v_L, v_U, a_L, a_U$ inherited from Figure 3.4
d_s is the location of the slope change,
θ_0 is the slope from d_0 to d_s,
θ_1 is the slope from d_s to d_f

Figure 3.10: Bus route with slope change

Slope profile is a function of distance in contrast with other variables which are time

dependence. The change of slope does not have the constraint of time or speed attached to it.

The slope profile is neither smooth nor continuous at the change of slope. There are two approaches to formulate the slope changes. The first approach is to formulate the problem as a multiphase problem where a slope change marks a phase boundary. The problem shown in Figure 3.10 is formulated as a two-phase problem with phase one between d_0 to d_s, and phase two from d_s to d_f.

As similar to the case in Section 3.3.3, the objective function of each phase in this problem is identical to that in Section 3.3.1. To introduce slope into the problem such that the model of the vehicle is a function of control, states and time, slope is formulated as the third state $x_3(t)$ apart from speed and distance. As the slope profile is not a function of speed or distance in a single phase, no dynamic constraint is added. The value of slope in each phase is fixed by using the path constraints. The path constraints bound the third state $x_3(t)$ to the values of the slope. The additional path constraints for phase one and phase two are shown in Equations 3.83 and 3.84 respectively.

$$\theta_0 \leq x_3^{(1)}(t^{(1)}) \leq \theta_0 \tag{3.83}$$

$$\theta_1 \leq x_3^{(2)}(t^{(2)}) \leq \theta_1 \tag{3.84}$$

The boundary conditions for phase one, with distance from d_0 to d_s, are

$$x_2^{(1)}(t_0^{(1)}) = d_0, \tag{3.85}$$

$$x_2^{(1)}(t_f^{(1)}) = d_s, \tag{3.86}$$

$$T_0 \leq t_0^{(1)} \leq T_f, \text{ and} \tag{3.87}$$

$$T_0 \leq t_f^{(1)} \leq T_f. \tag{3.88}$$

The boundary conditions for phase two are

$$x_2^{(2)}(t_0^{(2)}) = d_s, \tag{3.89}$$

$$x_2^{(2)}(t_f^{(2)}) = d_f, \tag{3.90}$$

$$T_0 \leq t_0^{(2)} \leq T_f, \text{ and} \tag{3.91}$$

$$T_f \leq t_f^{(2)} \leq T_f. \tag{3.92}$$

The phase linkage constraints between the two phases are

$$x_1^{(1)}(t_f^{(1)}) - x_1^{(2)}(t_0^{(2)}) = 0, \tag{3.93}$$

$$x_2^{(1)}(t_f^{(1)}) - x_2^{(2)}(t_0^{(2)}) = 0, \tag{3.94}$$

$$u^{(1)}(t_f^{(1)}) - u^{(2)}(t_0^{(2)}) = 0, \text{ and} \tag{3.95}$$

$$t_f^{(1)} - t_0^{(2)} = 0. \tag{3.96}$$

The second approach to formulate the slope change is to use an approximation method

to translate the slope profile from tabular data to a function of distance. Popular approximation methods include the use of Fourier series and Chebyshev polynomials. However, both methods introduce the Gibbs phenomenon [98] into the approximated profile. Gibbs phenomenon is the oscillating effect that occurs when approximating a profile with jump discontinuity. The PSOPT tool provides a smooth linear interpolation function that translates a tabular data to a differentiable function. Quoting from the user manual of PSOPT [99]:

> *"...The method used avoids joining sharp corners between adjacent linear segments. Instead smoothed pulse functions are used to join the segments. The function is suitable for automatic differentiation. ..."*

Equation 3.97 shows the smooth linear interpolation method to determine the value y at x from the tabular data X and Y of size N where X is a monotonically increasing series and Y_i is the value at X_i.

$$\begin{aligned} y = \sum_{i=0}^{n-2} \big[H(x - X_i) - H(x - X_{i+1})\big] \big[Y_i + \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i}(x - X_i)\big] \\ + \big[H(x - [X_0 - 100a_{N-2}]) - H(x - X_i))\big] Y_0 \\ + \big[H(x - X_{N-1}) - H(x - [X_{N-1} + 100a_{N-2}])\big] \end{aligned} \tag{3.97}$$

where $H(\cdot)$ is a smooth heaviside function (Equation 3.53)

a is the parameter in smooth heaviside function (refer to Equation 3.53)

a is defined using Equation 3.98.

$$a_i = \begin{cases} 0.1 \times |X_i - X_{i+1}| & \text{for } i = 1 \\ 0.1 \times \min(|X_{i-1} - X_i|, |X_i - X_{i+1}|) & \text{otherwise} \end{cases} \tag{3.98}$$

The original slope profile is conditioned before passing to the smooth linear interpolation function to avoid the definition of two slope values at the same distance, i.e. at the point of slope change. The conditioning is done by defining the previous slope value at a distance slightly smaller than the distance at slope change, and the next slope value at a distance slightly larger than the distance at slope change.

The original slope profile in Figure 3.10 is conditioned to be the slope profile in Figure 3.11. d_s^- and d_s^+ are defined in Equations 3.99 and 3.100 respectively.

$$d_s^- = d_s - 0.1 \times (d_s - d_0) \tag{3.99}$$

$$d_s^+ = d_s + 0.1 \times (d_f - d_s) \tag{3.100}$$

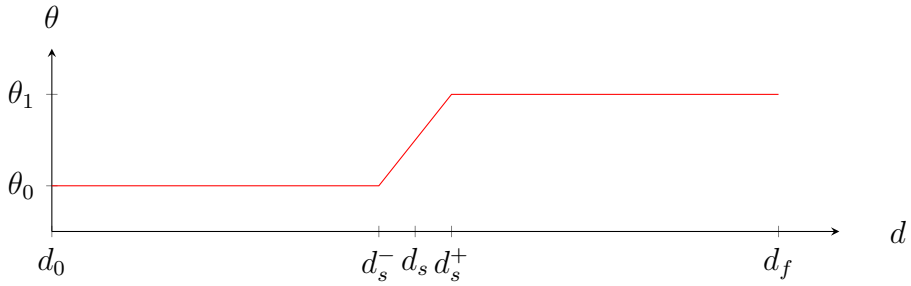


Figure 3.11: Conditioned slope profile for smooth linear interpolation

By using this approach, no additional state is introduced. The smooth linear interpolation function is implemented in the model of the vehicle to calculate the slope based on

the distance. A quick test has shown that the second approach provides a quicker solution with comparable results to the first approach. Figure 3.12 shows the comparison of the solution time with different numbers of slope change for the two approaches. The solution time for the first approach increases at a faster rate compare to that of the second approach. Therefore the second approach is used.

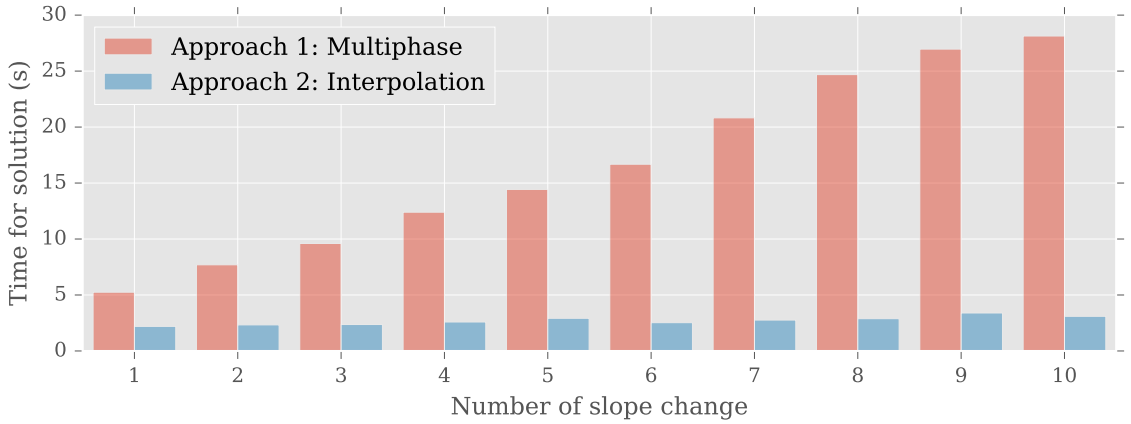


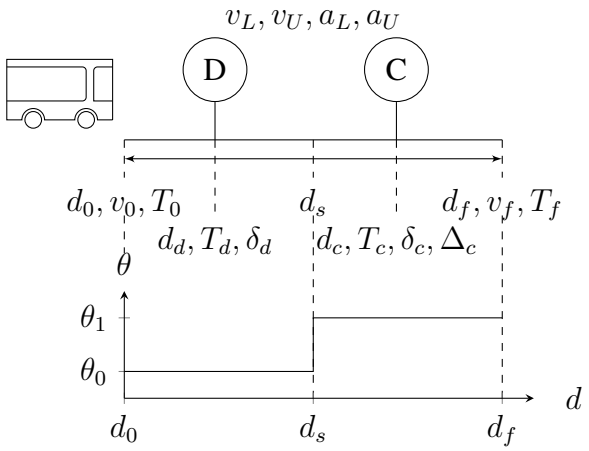
Figure 3.12: Solution time comparison of two approaches for slope change formulation

3.3.5 Route with an on-demand bus stop, a compulsory bus stop and a slope change

Consider a route with an on-demand bus stop at distance d_d with estimated arrival time $T_d \pm \delta_d$, a compulsory bus stop at distance d_c with estimated arrival time $T_c \pm \delta_c$ and stop period Δ_c, and a slope change at d_s from θ_0 to θ_1. This route is visualised in Figure 3.13.

The number of phases can be calculated with

$$n_p = 1 + (2n_c) \tag{3.101}$$



where $d_0, d_f, v_0, v_f, T_0, T_f, v_L, v_U, a_L, a_U$ inherited from Figure 3.4
d_d, T_d, δ_d inherited from Figure 3.5
$d_c, T_c, \delta_c, \Delta_c$ inherited from Figure 3.8
d_s, θ_0, θ_1 inherited from Figure 3.10

Figure 3.13: Bus route with an on-demand bus stop, a compulsory bus stop and a slope change

where n_p is the number of phases and n_c is the number of compulsory bus stops.

This problem is thus split into $(1+2(1)) = 3$ phases with phase one from the beginning of the route d_0 to the location of the compulsory bus stop d_c with terminal time constraint of $T_c \pm \delta_c$, phase two at the compulsory bus stop d_c, to be at halt from $T_c \pm \delta_c$ to $T_c + \Delta_c$, and phase three from $T_c + \Delta_c$ at d_c to the destination d_f at time T_f.

The objective functions for all phases are computed with

$$J^{(i)} = \int_{t_0}^{t_f} V(x^{(i)}(t^{(i)}), u^{(i)}(t), t^{(i)}) + w_d(x_2^{(i)}(t) - d_d)^2[H(t^{(i)} - T_d + \delta_d) - H(t^{(i)} - T_d - \delta_d)]\, dt \tag{3.102}$$

where i is the phase number, i.e. 1, 2 or 3, subject to the dynamic constraints

$$\dot{x}_1^{(i)}(t^{(i)}) = u^{(i)}(t^{(i)}) \text{ and} \tag{3.103}$$

$$\dot{x}_2^{(i)}(t^{(i)}) = x_1^{(i)}(t^{(i)}), \tag{3.104}$$

and the path constraints

$$v_L \leq \quad x_1^{(i)}(t^{(i)}) \quad \leq v_U, \tag{3.105}$$

$$d_0 \leq \quad x_2^{(i)}(t^{(i)}) \quad \leq d_f, \text{ and} \tag{3.106}$$

$$\left\{ \begin{array}{ll} 0 & i = 2 \\ a_L & otherwise \end{array} \right\} \leq \quad u^{(i)}(t^{(i)}) \quad \leq \left\{ \begin{array}{ll} 0 & \text{for } i = 2 \\ a_U & \text{otherwise} \end{array} \right\}. \tag{3.107}$$

The second term in the integrand part of the objective function will only be activated in phase one within $T_d \pm \delta_d$. In other phases this term will always be zero.

The boundary conditions for phase 1:

$$x_1^{(1)}(t_0^{(1)}) = v_0 \tag{3.108}$$

$$x_1^{(1)}(t_f^{(1)}) = 0 \tag{3.109}$$

$$x_2^{(1)}(t_0^{(1)}) = d_0 \tag{3.110}$$

$$x_2^{(1)}(t_f^{(1)}) = d_c \tag{3.111}$$

$$t_0^{(1)} = T_0 \tag{3.112}$$

$$T_c - \delta_c \leq t_f^{(1)} \leq T_c + \delta_c \tag{3.113}$$

phase 2:

$$x_1^{(2)}(t_0^{(2)}) = 0 \tag{3.114}$$

$$x_1^{(2)}(t_f^{(2)}) = 0 \tag{3.115}$$

$$x_2^{(2)}(t_0^{(2)}) = d_c \tag{3.116}$$

$$x_2^{(2)}(t_f^{(2)}) = d_c \tag{3.117}$$

$$T_c - \delta_c \leq t_0^{(2)} \leq T_c + \delta_c \tag{3.118}$$

$$t_f^{(2)} = T_c + \Delta_c \tag{3.119}$$

and phase 3:

$$x_1^{(3)}(t_0^{(3)}) = 0 \tag{3.120}$$

$$x_1^{(3)}(t_f^{(3)}) = v_f \tag{3.121}$$

$$x_2^{(3)}(t_0^{(3)}) = d_c \tag{3.122}$$

$$x_2^{(3)}(t_f^{(3)}) = d_f \tag{3.123}$$

$$t_0^{(3)} = T_c + \Delta_c \tag{3.124}$$

$$T_0 \leq t_f^{(3)} \leq T_f \tag{3.125}$$

The phase linkage constraints are

$$x_1^{(i)}(t_f^{(i)}) - x_1^{(i+1)}(t_0^{(i+1)}) = 0, \tag{3.126}$$

$$x_2^{(i)}(t_f^{(i)}) - x_2^{(i+1))}(t_0^{(i+1)}) = 0, \tag{3.127}$$

$$u^{(i)}(t_f^{(i)}) - u^{(i+1)}(t_0^{(i+1)}) = 0, \text{ and} \tag{3.128}$$

$$t_f^{(i)} - t_0^{(i+1)} = 0 \tag{3.129}$$

with i = 1 or 2.

The conditioned slope profile will have the profile as shown in Figure 3.11.

3.4 Case study

This case study is based on an actual bus route of the bus service 300 operated by Oxford Bus Company. The bus route from Oxford City Centre to Pear Tree Park and Ride is modelled and the optimal speed profile is identified for one bus schedule. The actual bus route on the map is shown in Figure 3.14. The model of the road is specified in Table 3.2 and the model of the route in Table 3.3. The models are visualised in Figures 3.15 and 3.16. The values of the bounding variables are listed in Table 3.4. The speed limit along path is chosen to be 22.0 m/s, i.e. 79.2 km/h since the national speed limit in the UK for a bus on single carriageways is 80 km/h [100]. The deceleration and acceleration limits are set as -2.5 and 2.5 m/s^2 as the maximum typical deceleration and acceleration of a bus are -1.86$\pm$0.50 and 1.47$\pm$0.20 m/s^2 respectively [101]. The relaxed limits for acceleration

and deceleration are chosen to allow more freedom to the optimal control algorithm to generate the optimal profile. High accelerations will be penalised in the algorithm as high accelerations contribute to high energy consumption.

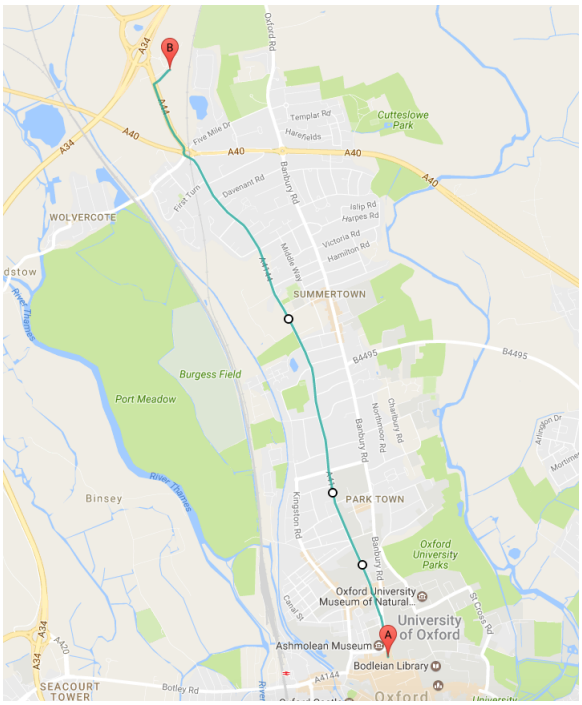


Figure 3.14: Bus route from Oxford City Centre to Pear Tree Park and Ride

Table 3.2: Model of the road between Oxford City Centre to Pear Tree Park and Ride

d (m)	θ (°)	d (m)	θ (°)	d (m)	θ (°)
0.0	-0.61	340.0	0.23	1359.0	0.07
1699.0	0.02	2378.0	-0.01	2718.0	0.50
3057.0	0.66	3397.0	-0.43	3736.0	0.92
4076.0	-1.01	4415.0	-0.90	4739.0	1.54

Table 3.3: Model of the route between Oxford City Centre to Pear Tree Park and Ride

t (s)	d (m)	c	δ (s)	Δ (s)
60.0	713.0	compulsory	5.0	30.0
180.0	1292.0	on-demand	5.0	0.0
360.0	2626.0	on-demand	5.0	0.0

Table 3.4: Bounding variable values for bus route from Oxford City Centre to Pear Tree Park and Ride

Variables	**Values**	**Variables**	**Values**
initial speed (m/s)	0.0	deceleration limit (m/s^2)	-2.5
speed at destination (m/s)	0.0	acceleration limit (m/s^2)	2.5
speed limit along path (m/s)	22.0	starting time (s)	0.0
starting distance (m)	0.0	maximum time at destination (s)	600.0
distance at destination (m)	4812.0		

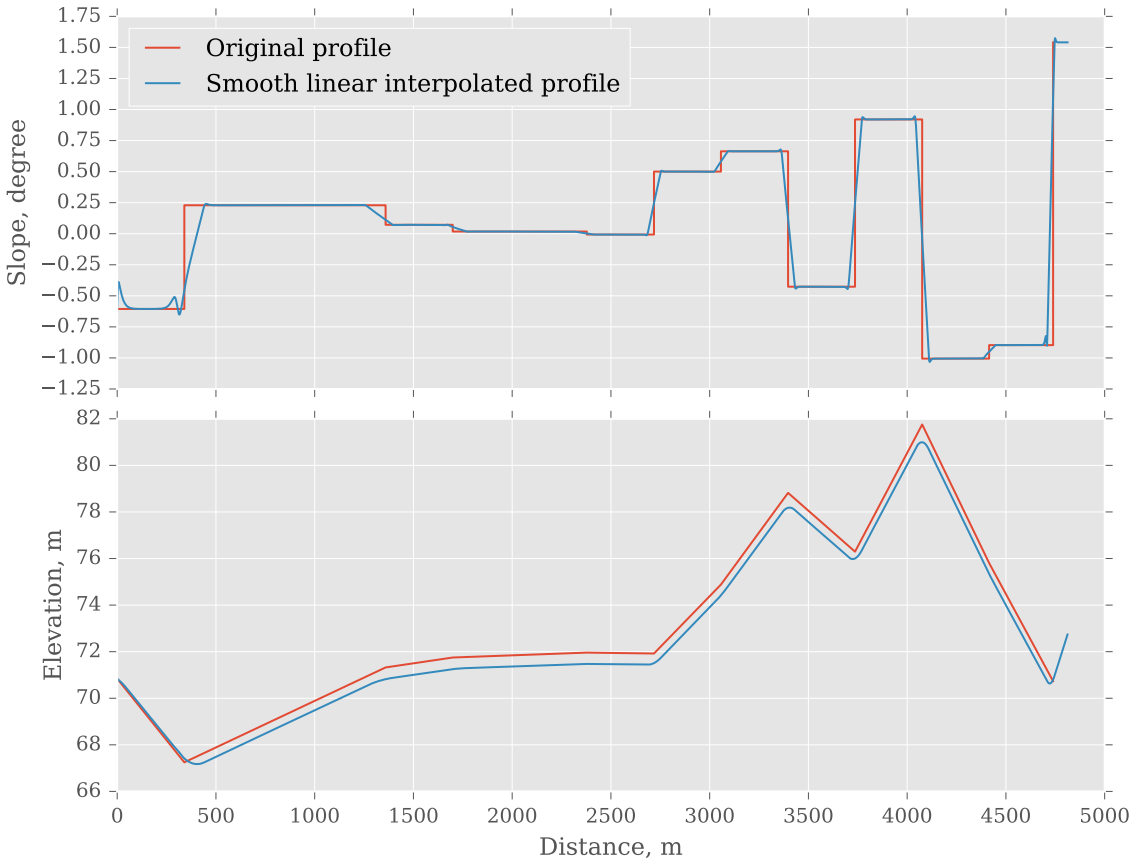


Figure 3.15: Slope and elevation profile for bus route from Oxford City Centre to Pear Tree Park and Ride

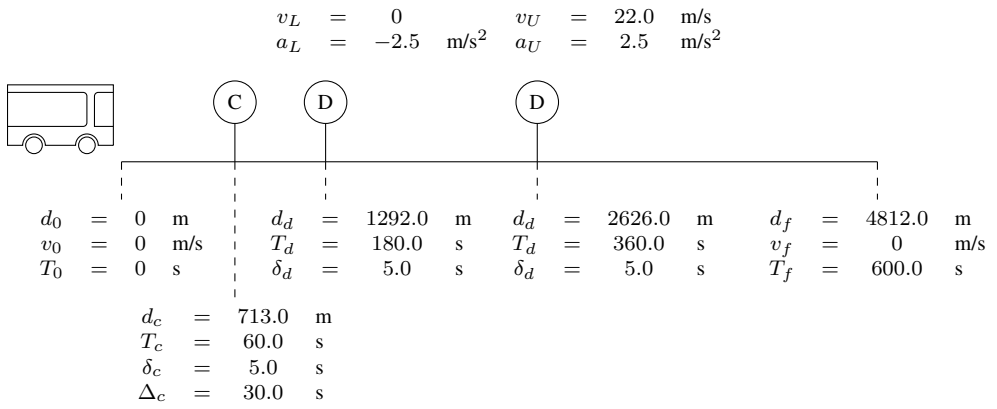


Figure 3.16: Bus route from Oxford City Centre to Pear Tree Park and Ride

Three different cases are simulated to identify the effect of bus stops and slope changes on the optimal profile. Case I solves for the optimal control problem with a single stop (the destination) and zero slope throughout the route. The result is compared to the profile of an aggressive driving style. Case II includes the requirements of the bus stops on top of Case I. Case III then introduces the slope changes on top of Case II.

3.4.1 Case I: Base case

The optimal control problem to be solved in this case has a single stop at the destination and zero slope throughout the route. The bus stops along the route are not included. The resultant profiles are shown in Figure 3.17.

The result shows a constant speed along most of the route apart from the beginning and the end of the route due to the constraints to start and end at zero speed. The constant speed minimises the acceleration along the route which in turns minimises the power consumption of the vehicle. As there is no bus stop along the route, the objective function consists

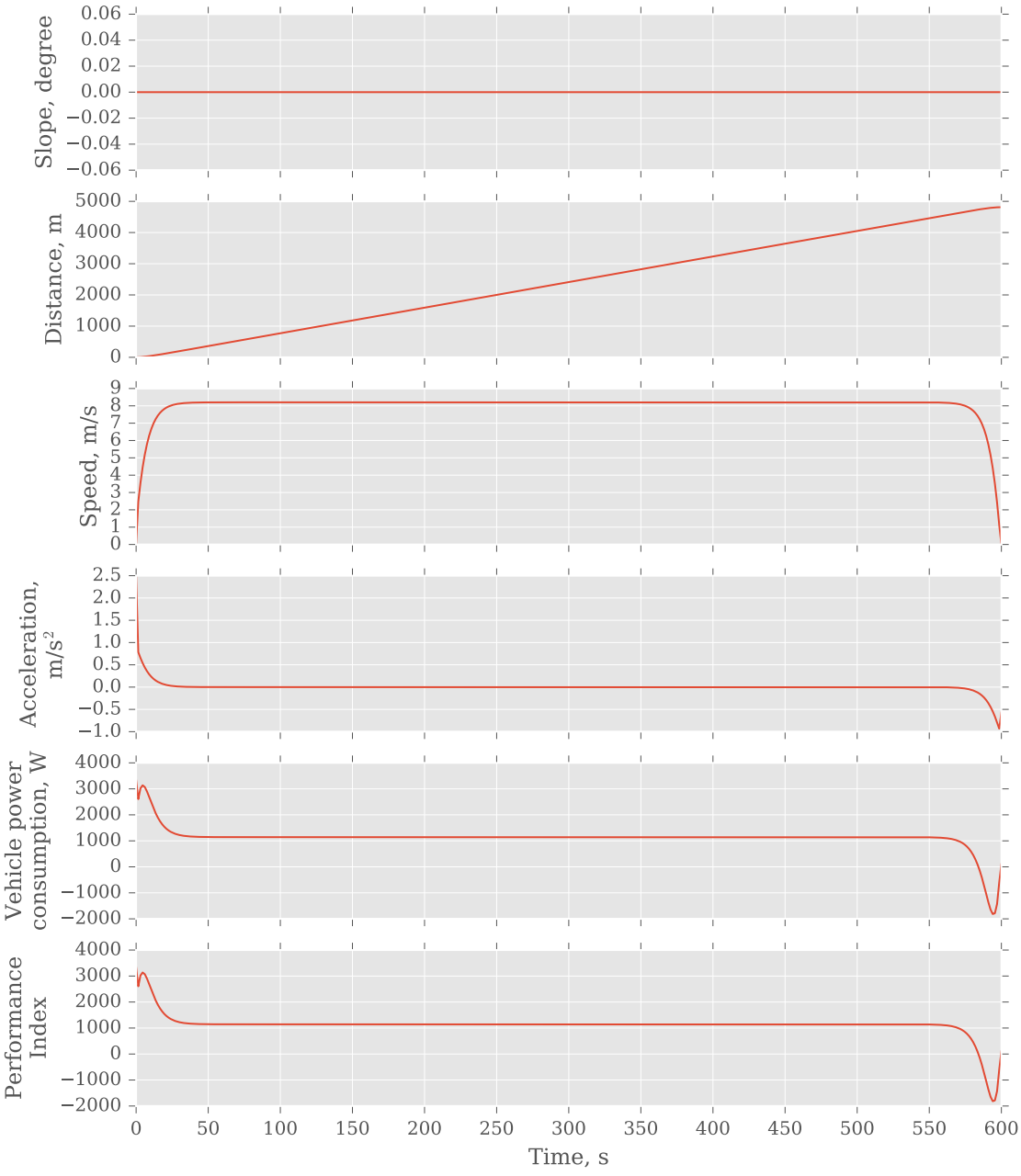


Figure 3.17: Optimal profiles with single stop at the end and zero slope

of only the power consumption of the vehicle. The total energy consumption of this profile is 671.44 kJ. It can be observed that the initial point of the acceleration profile is at 2.5 m/s^2, i.e. the upper limit of acceleration. However, since it occurs only at the initial point and drops steeply right after, in practical the acceleration will be much lower and closer to the second point, i.e. below 1.0 m/s^2.

An aggressive driving style is set up to compare the energy saving of the optimal profile. The driver drives at maximum acceleration to reach maximum speed and decelerate at maximum deceleration upon reaching the destination. Therefore the bus arrives at the destination sooner than it need to be and waits until the departure time from the destination. The resultant profile is shown in Figure 3.18. The bus reaches the destination in less than 230 seconds and therefore will be waiting at the destination for 370 seconds (slightly more than 6 minutes). The total energy consumption is 1677.48 kJ. The optimal profile reduces the energy consumption of the aggressive driving style by 60%.

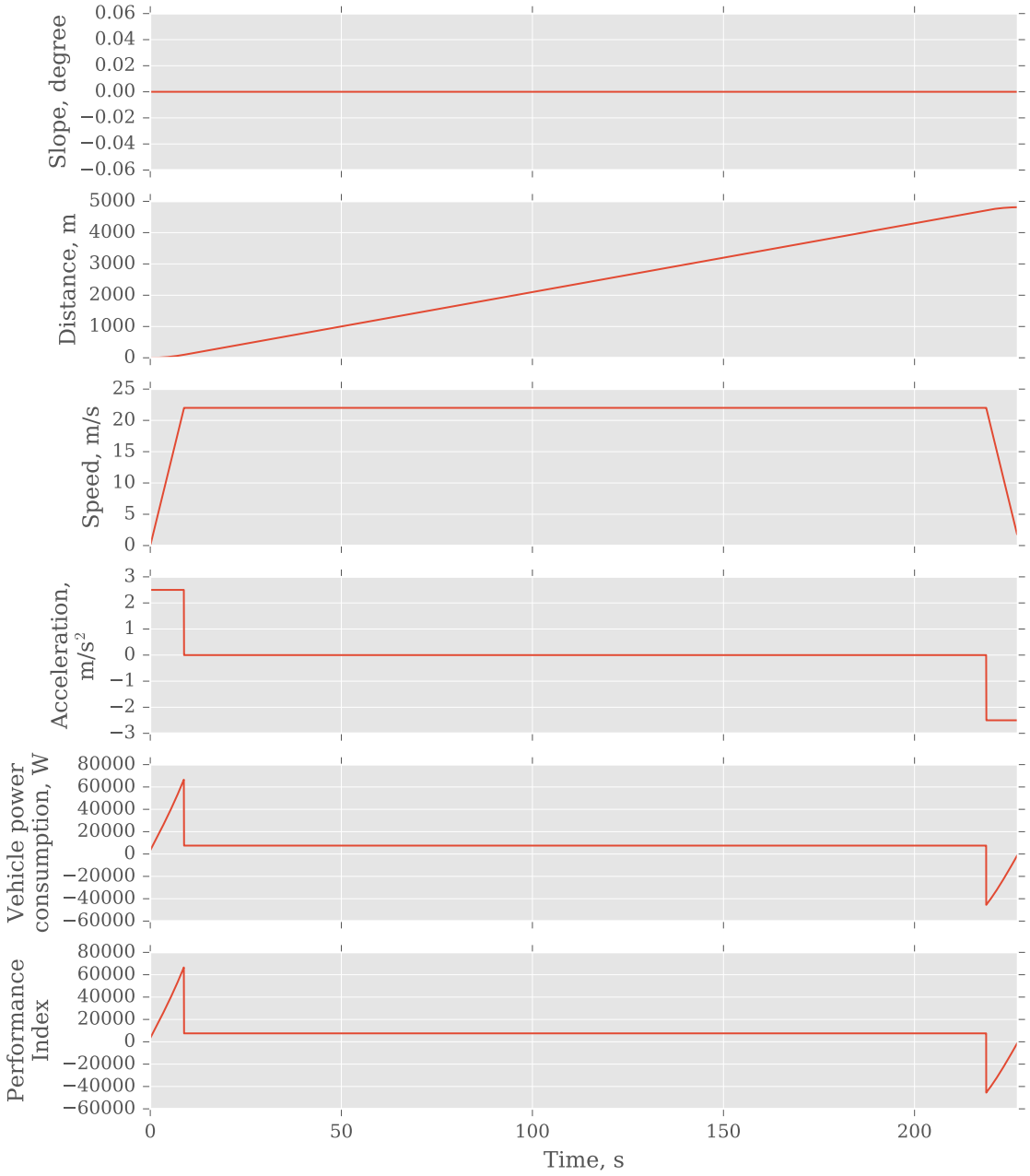


Figure 3.18: Profiles of an aggressive driving style with single stop at the end and zero slope

3.4.2 Case II: Bus route with bus stops

Case II includes the bus stops requirements on top of Case I. The results of Case II are shown in Figure 3.19. The filled circles on the graphs show the locations of the bus stops and the shaded area shows the time period when the bus is at the bus stops. The results show that the location of bus coincides with the locations of bus stops at the desired time of arrival as defined in Table 3.3. The resultant speed profile provides a smooth speed profile which avoids the increase in energy consumption when the bus is approaching the on-demand bus stops.

The main differences between the profiles in Case II (Figure 3.19) and Case I (Figure 3.17) are first, the power consumption when approaching and leaving the compulsory bus stop due to the deceleration and acceleration of the bus; second, the speed is maintained at a lower value before the first on-demand bus stop in order to arrive at the bus stop at scheduled time; third, the hike of the speed just before deceleration to arrive at the destination in time.

The total energy consumption of the profile in Case II is 779.79 kJ. The energy consumption is 16% more than that in Case I. This is mainly due to the requirements to decelerate to stop and accelerate from rest at the compulsory bus stop.

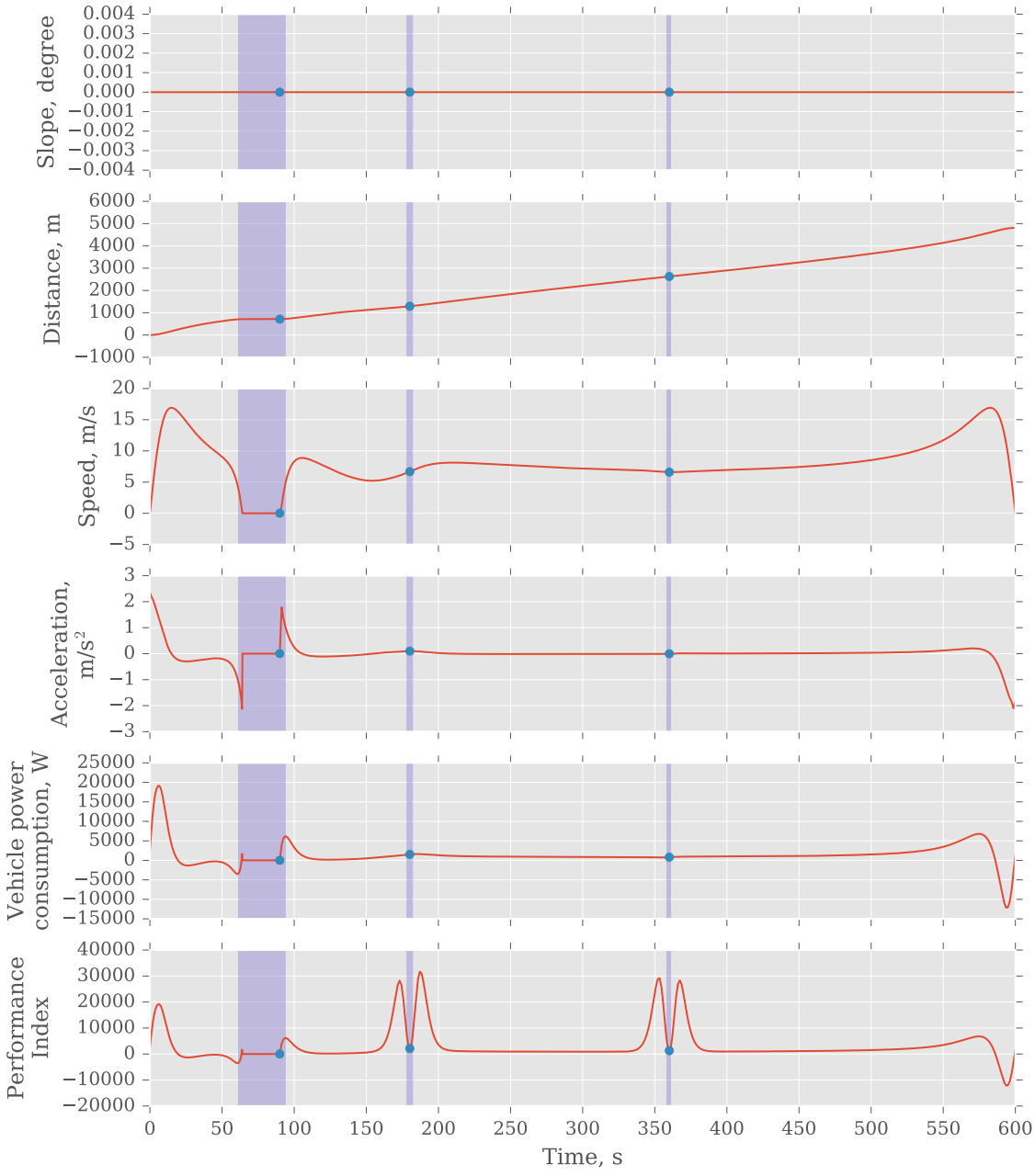


Figure 3.19: Optimal profiles with bus stops and zero slope

3.4.3 Case III: Bus route with bus stops and slope changes

Case III introduces the slope changes on top of Case II. The results are shown in Figure 3.20. The profiles of the distance, speed, and acceleration in Figure 3.20 are identical to that in Case II (Figure 3.19). However differences can be observed in the profile of the power consumption. There are two sections where abrupt changes are present in the slope and thus affect the power consumption of the vehicle. The first section occurs just before the first bus stop and the second section, which is more significant, is just after the last bus stop. At a positive slope, the vehicle consumes more energy to maintain its speed while at a negative slope, less energy is required for the same speed. The effect of slope is due to the gravitational drag that is formulated in the model of vehicle in Equation 3.8.

The total energy consumption of the profile in this case is 795.93 kJ. The energy consumption is 2.1% more than the energy consumed in Case II. By comparing their respective power consumption profile, this is due to the presence of non-zero slopes at later part of the route.

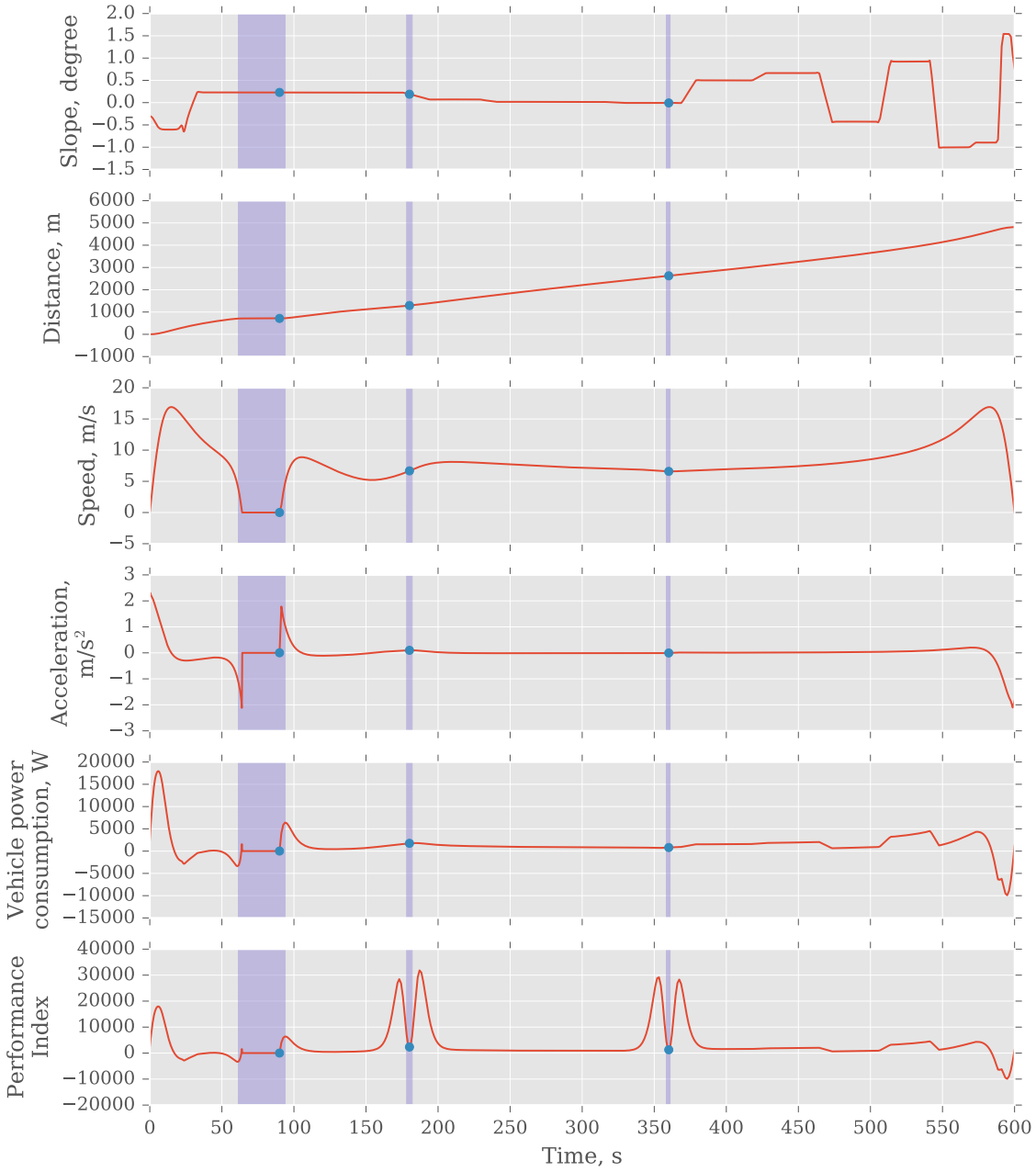


Figure 3.20: Optimal profiles with bus stops and slope changes

3.5 Summary

The formulation of an optimal driving problem for bus has been discussed. The purpose of the optimal driving is to reduce the energy consumption and improve the punctuality fo the bus to arrive at each bus stop along a bus route. The direct collocation method is used to solve the optimal control problem and PSOPT is selected as the optimal control program. A model of vehicle based on OVEM is used for the calculation of energy consumption. The format of the model of a road and a bus route has been proposed to allow the specification of the features of a route and information of bus stops on the route. Bus stops are categorised into two types, on-demand and compulsory bus stop. The problem formulation for (i) a basic route without bus stop at zero slope, (ii) a route with an on-demand bus stop at zero slope, (iii) a route with a compulsory bus stop at zero slope, (iv) a route with slope changes but no bus stop, and (v) a route with an on-demand bus stop, a compulsory bus stop and a slope change are presented. The requirements of an on-demand bus stop are formulated as part of the objective function. A compulsory bus stop is formulated using the approach of multiphase problem. A smooth linear interpolation method is used to model the slope changes along the route. The slope changes are used in the calculation of energy consumption that is part of the objective function. A case study of the bus route from Oxford City Centre to Pear Tree Park and Ride is presented. Three different cases on the same route are investigated. Case I provides the problem formulation and the optimal profiles with single stop at the end of the route at zero slope. This case is compared to an

aggressive driving style that is common among bus drivers and the comparison shows 60% reduction in energy consumption. Case II considers the bus stops along the route at zero slope and Case III includes both bus stops and slope changes along the route. The optimal profiles for all cases have shown a consistent feature of minimising the acceleration along the route.

Chapter 4

Parameter Identification for Real-World Implementation of Optimal Bus Driving

This chapter focuses on the identification of the parameters required to implement the optimal bus driving in real-world scenarios. These parameters include the slopes and length of the bus route, the locations and the types of bus stops on the route, and the stopping probability at each bus stop.

The route of a specific bus service is obtained from the official website of the bus company. The actual GPS coordinates of the bus route is obtained by joining a series of road segments based on the information from the Oxfordshire County Council traffic portal [107]. The GPS coordinates of the road segments are obtained using the Google Maps API. The elevations at each GPS coordinates, which are later translated to slopes, are also obtained through the Google Maps API. The locations of the bus stops are identified from the Google Maps manually. The historical bus speed data used to identify the stopping probability at a bus stop is provided by the Oxford Bus Company. The term 'GPS coordinates' in this chapter always refers to the pair of latitude and longitude values that represent

a geographical location. Altitude is not discussed in this chapter.

Section 4.1 first presents the formulae that are commonly used in identifying distance between two GPS points and interpolating points between two points. These formulae form the basis of the algorithms used in the rest of the chapter. These formulae are validated in particular typical area in which the case study occurs, as well as wider testing area in which discrepancy between algorithms is larger. The two areas are shown in Figure 4.1 with the blue box as the typical area and the red box as the testing area. The typical area is within the bound of (51.82, -1.35) and (51.70, -1.15), and the testing area is within the bound of (52.00, -1.60) and (51.50, -0.90).

Figure 4.1: Typical (blue) and testing (red) graphical area

Section 4.2 discusses the formation of a bus route. A bus route is essentially joint road segments. Road segments are sections of a road defined for the ease of information transfer. More information is discussed in the section. This section explains the methodology to

connect different road segments by splicing two consecutive but not necessarily connected road segments. As a gap might exist between two consecutive road segments, the filling of this gap is also explained. The slope along the route is then adjusted such that it minimises the computational demands while preserving an accurate representation of the slope.

The locations of the bus stops are identified separately from the formation of a bus route. Section 4.3 explains the algorithm used to fit the bus stops into an existing bus route. The algorithm identifies the two points on the bus route between which the bus stop lies and inserts the bus stop at the appropriate distance.

As defined in Chapter 3, bus stops are categorised as compulsory or on-demand bus stops. The compulsory bus stop is deemed to have a speed limit of zero at a specified time and location, and the on-demand bus stop is deemed to only have a location constraint at specified time with no speed constraint. Section 4.4 discusses the methodology to identify the optimal speed profile such that the approaching speed of the bus to an on-demand bus stop reflects the situation at the bus stop using the stop probability at the bus stop. Section 4.5 then discusses the methodology to determine the stop probability of a bus at an on-demand bus stop using historical bus data.

4.1 Basic formulae

This section discusses the formulae to calculate the distance between two GPS coordinates, interpolate points between two GPS coordinates, and identify the GPS bound given the centre of the bound and the distance between the centre and the border of the bound.

The cosine-haversine formula and Vincenty formula are compared for the calculation of distance between GPS coordinates. A method based on the spherical linear interpolation (Slerp) [102] is compared with a simple linear interpolation for the interpolation of points between two GPS coordinates. A method to identify the GPS bound is developed based on the interpolation of GPS points.

4.1.1 GPS distance calculation

The cosine-haversine formula [103] is a common formula used for the calculation of great circle distance between two GPS points, i.e. the shortest distance on the Earth. The formula considers the Earth as a sphere with a radius of 6371 km. The formulae to calculate the great circle distance d between point 1 (ϕ_1, λ_1) and point 2 (ϕ_2, λ_2) are presented in Equations 4.1 to 4.3 [104].

$$a_g = \sin^2 \tfrac{\Delta\phi}{2} + \cos\phi_1 \cos\phi_2 \sin^2 \tfrac{\Delta\lambda}{2} \tag{4.1}$$

$$c_g = 2\,\mathrm{atan2}(\sqrt{a_g}, \sqrt{1-a_g}) \tag{4.2}$$

$$d_g = R_E c_g \tag{4.3}$$

where $\Delta\phi = \phi_2 - \phi_1$

$\Delta\lambda = \lambda_2 - \lambda_1$

Since the Earth is not a perfect sphere but rather, an ellipsoid, T. Vincenty [105] proposed an iterative method to compute the length of a geodesic formed by two geographical

points assuming the Earth as an ellipsoid. The solution is commonly known as the inverse method of the Vincenty formula.

The Vincenty formula defines the Earth as an ellipsoid with a_e and b_e as the major and minor semiaxes of the ellipsoid (Figure 4.2). The flattening, f_e of the ellipsoid is given by Equation 4.4.

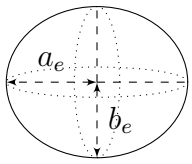


Figure 4.2: Illustration of major a_e and minor b_e semiaxes of ellipsoid

$$f_e = \frac{a_e - b_e}{a_e} \tag{4.4}$$

There are different standards that define the values of a_e, b_e and $\frac{1}{f_e}$ for the Earth. The latest standard defined in 1984 is the WGS84 which defines a_e, b_e and $\frac{1}{f_e}$ as 6378.137 km, 6356.7523142 km and 298.257223563 respectively [106].

The Vincenty formula computes the distance between (ϕ_1, λ_1) and (ϕ_2, λ_2) by iterating a set of equations until the value of the variable l converges. The reduced latitudes U_1 and U_2 are first calculated from the latitudes ϕ_1 and ϕ_2 using Equation 4.5.

$$\tan U_i = (1 - f_e) \tan \phi_i \qquad \text{for } i = 1,\ 2 \tag{4.5}$$

At first approximation, l is taken as the difference in the longitudes of the two points, i.e.

$$l = \lambda_2 - \lambda_1 \tag{4.6}$$

Equations 4.7 to 4.14 are then iterated until the change in l is negligible.

$$\sin\sigma = \sqrt{(\cos U_2 \sin l)^2 + (\cos U_1 \sin U_2 - \sin U_1 \cos U_2 \cos l)^2} \tag{4.7}$$

$$\cos\sigma = \sin U_1 \sin U_2 + \cos U_1 \cos U_2 \cos l \tag{4.8}$$

$$\sigma = \operatorname{atan2}(\sin\sigma, \cos\sigma) \tag{4.9}$$

$$\sin\alpha_v = \frac{\cos U_1 \cos U_2 \sin l}{\sin\sigma} \tag{4.10}$$

$$\cos^2\alpha_v = 1 - \sin^2\alpha_v \tag{4.11}$$

$$\cos 2\sigma_m = \cos\sigma - 2\frac{\sin U_1 \sin U_2}{\cos^2\alpha_v} \tag{4.12}$$

$$C = \frac{f_e}{16}[4 + f(4 - 3\cos^2\alpha_v)]\cos^2\alpha_v \tag{4.13}$$

$$l = (\lambda_2 - \lambda_1) + (1 - C) f_e \sin\alpha_v \big(\sigma + C\sin\sigma[\cos 2\sigma_m + C\cos\sigma(-1 + 2\cos^2 2\sigma_m)]\big) \tag{4.14}$$

The result of the iterations is then used to calculate the distance between the two points, d_v using Equation 4.15.

$$d_v = b_e A_v(\sigma - \Delta\sigma) \tag{4.15}$$

where $u_v^2 = \cos^2\alpha_v \, \frac{a_e^2 - b_e^2}{b_e^2}$

$$A_v = 1 + \frac{u_v^2}{16384}\{4096 + u_v^2[-768 + u_v^2(320 - 175u_v^2)]\}$$

$$B_v = \frac{u_v^2}{1024}\{256 + u_v^2[-128 + u_v^2(74 - 47u_v^2)]\}$$

$$\Delta\sigma = B_v \sin\sigma\{\cos 2\sigma_m + \tfrac{1}{4}B_v[\cos\sigma(-1 + 2\cos^2 2\sigma_m) - \tfrac{1}{6}B_v \cos 2\sigma_m(-3 + 4\sin^2\sigma)(-3 + 4\cos^2 2\sigma_m)]\}$$

The cosine-haversine formula and the Vincenty formula are compared using pairs of points within the testing geographical area separated at different distances. The points are displayed in Figure 4.3. Table 4.1 shows the coordinates of the points and the results of the two formulae. Figure 4.4 compares the distance calculation results of cosine-haversine and Vincenty formulae.

Table 4.1: Comparison of cosine-haversine and Vincenty formulae in testing area

Label	p_0	p_1	**Cosine-haversine**		**Vincenty**		**Difference (m)**	**%-age difference**[†]
			Distance (km)	Exec. time[*] (s)	Distance (km)	Exec. time[*] (s)		
A	(51.78, -1.28)	(51.74, -1.22)	6.0709	2.0259e-05	6.0802	3.6485e-05	9.2580	0.1523
B	(51.72, -1.32)	(51.80, -1.18)	13.1176	1.9982e-05	13.1402	4.1205e-05	22.6100	0.1721
C	(51.82, -1.35)	(51.70, -1.15)	19.1762	2.0065e-05	19.2081	3.6162e-05	31.8487	0.1658
D	(51.88, -1.43)	(51.63, -1.07)	37.2502	1.9959e-05	37.3050	3.9811e-05	54.8416	0.1470
E	(51.94, -0.98)	(51.57, -1.52)	55.4612	2.0376e-05	55.5438	4.0043e-05	82.6025	0.1487
F	(52.00, -1.60)	(51.50, -0.90)	73.5943	2.0030e-05	73.6999	3.9757e-05	105.6776	0.1434

[*]Execution time
[†]Percentage Difference

From the comparisons, the absolute difference between the two methods decreases when the distance between the points decreases. Further comparisons are performed to identify the discrepancy between the two methods at smaller distances. A series of points

Figure 4.3: Points for cosine-haversine and Vincenty formulae comparison in testing area

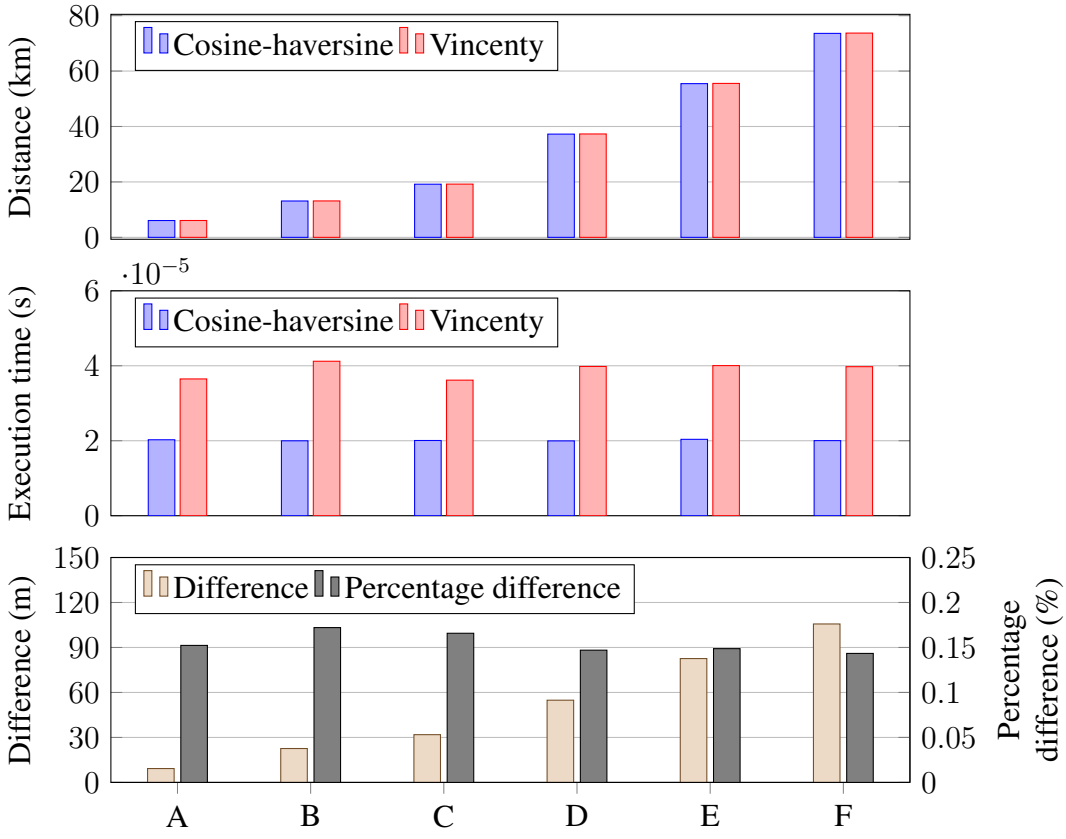


Figure 4.4: Graphical representations for comparison of consine-haversine and Vincenty formulae in testing area

between the centre and the north-west vertex of the typical geographical area are identified as shown in Figure 4.5. The distances from the centre to each of the points are calculated and the comparisons are presented in Table 4.2.

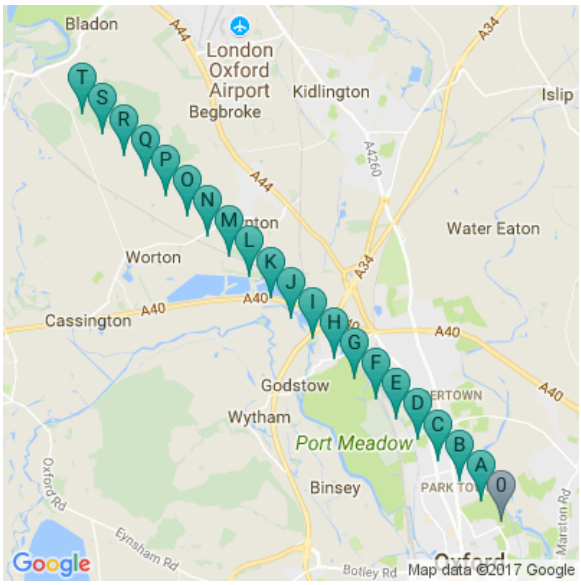


Figure 4.5: Points for cosine-haversine and Vincenty formulae comparison in typical area with point 0 as the centre of the typical area

As the percentage of difference between the two methods found to be consistent in Table 4.2 and Figure 4.6, another examination is carried out to investigate the effect of bearing between the two points on the percentage of difference. The centre is again chosen as the first point. The points used as the second point are chosen such that bearing from the centre to each point is different. This examination is also carried out within the typical area. The locations of the points are displayed in Figure 4.7. The result is shown in Table 4.3 and displayed in Figure 4.8. It shows that the percentage of difference between two methods is influenced by the difference of bearing.

Table 4.2: Comparison of cosine-haversine and Vincenty formulae in typical area

Label	p_0	p_1	Cosine-haversine		Vincenty		Difference (m)	%-age difference[†]
			Distance (km)	Exec. time[*] (s)	Distance (km)	Exec. time[*] (s)		
A	(51.76, -1.25)	(51.763, -1.255)	0.4794	2.0229e-05	0.4802	3.5896e-05	0.7963	0.1658
B	(51.76, -1.25)	(51.766, -1.260)	0.9588	2.0054e-05	0.9604	3.6559e-05	1.5926	0.1658
C	(51.76, -1.25)	(51.769, -1.265)	1.4381	2.0058e-05	1.4405	3.6049e-05	2.3891	0.1658
D	(51.76, -1.25)	(51.772, -1.270)	1.9175	1.9988e-05	1.9207	3.6196e-05	3.1856	0.1659
E	(51.76, -1.25)	(51.775, -1.275)	2.3968	1.9936e-05	2.4008	3.5855e-05	3.9823	0.1659
F	(51.76, -1.25)	(51.778, -1.280)	2.8761	2.0038e-05	2.8809	3.6020e-05	4.7790	0.1659
G	(51.76, -1.25)	(51.781, -1.285)	3.3554	1.9907e-05	3.3610	3.6170e-05	5.5758	0.1659
H	(51.76, -1.25)	(51.784, -1.290)	3.8347	1.9955e-05	3.8411	3.5857e-05	6.3727	0.1659
I	(51.76, -1.25)	(51.787, -1.295)	4.3140	1.9996e-05	4.3212	4.2735e-05	7.1698	0.1659
J	(51.76, -1.25)	(51.790, -1.300)	4.7932	2.0440e-05	4.8012	3.6726e-05	7.9669	0.1659
K	(51.76, -1.25)	(51.793, -1.305)	5.2725	2.0000e-05	5.2812	3.6147e-05	8.7641	0.1659
L	(51.76, -1.25)	(51.796, -1.310)	5.7517	1.9936e-05	5.7612	3.6088e-05	9.5613	0.1660
M	(51.76, -1.25)	(51.799, -1.315)	6.2309	1.9821e-05	6.2412	3.6319e-05	10.3587	0.1660
N	(51.76, -1.25)	(51.802, -1.320)	6.7101	2.0801e-05	6.7212	3.6291e-05	11.1562	0.1660
O	(51.76, -1.25)	(51.805, -1.325)	7.1892	1.9911e-05	7.2012	3.6162e-05	11.9538	0.1660
P	(51.76, -1.25)	(51.808, -1.330)	7.6684	1.9815e-05	7.6811	3.5991e-05	12.7514	0.1660
Q	(51.76, -1.25)	(51.811, -1.335)	8.1475	2.0057e-05	8.1611	3.6238e-05	13.5492	0.1660
R	(51.76, -1.25)	(51.814, -1.340)	8.6266	2.0215e-05	8.6410	3.6465e-05	14.3470	0.1660
S	(51.76, -1.25)	(51.817, -1.345)	9.1057	2.0451e-05	9.1209	3.6715e-05	15.1450	0.1660
T	(51.76, -1.25)	(51.820, -1.350)	9.5848	2.0205e-05	9.6008	3.6202e-05	15.9430	0.1661

[*]Execution time
[†]Percentage Difference

Table 4.3: Comparison of cosine-haversine and Vincenty formulae in typical area (different bearing)

Label	p_0	p_1	Cosine-haversine		Vincenty		Difference (m)	%-age difference[†]
			Distance (km)	Exec. time[*] (s)	Distance (km)	Exec. time[*] (s)		
A	(51.76,-1.25)	(51.7648,-1.2485)	0.5395	2.0682e-05	0.5397	3.7077e-05	0.2316	0.0429
B	(51.76,-1.25)	(51.7640,-1.2471)	0.4933	2.0863e-05	0.4937	3.7661e-05	0.3767	0.0763
C	(51.76,-1.25)	(51.7629,-1.2460)	0.4294	2.0724e-05	0.4300	3.6661e-05	0.6079	0.1414
D	(51.76,-1.25)	(51.7615,-1.2452)	0.3697	2.0288e-05	0.3706	3.6482e-05	0.8703	0.2348
E	(51.76,-1.25)	(51.7600,-1.2450)	0.3442	2.0299e-05	0.3452	3.8561e-05	1.0021	0.2903
F	(51.76,-1.25)	(51.7585,-1.2452)	0.3697	2.0446e-05	0.3706	3.6576e-05	0.8703	0.2348
G	(51.76,-1.25)	(51.7571,-1.2460)	0.4294	2.0547e-05	0.4300	3.6554e-05	0.6078	0.1413
H	(51.76,-1.25)	(51.7560,-1.2471)	0.4933	2.0460e-05	0.4937	3.6725e-05	0.3765	0.0763
I	(51.76,-1.25)	(51.7552,-1.2485)	0.5395	2.0566e-05	0.5397	3.6616e-05	0.2312	0.0428
J	(51.76,-1.25)	(51.7550,-1.2500)	0.5561	2.0425e-05	0.5563	2.9824e-05	0.1824	0.0328
K	(51.76,-1.25)	(51.7552,-1.2515)	0.5395	2.0416e-05	0.5397	3.6658e-05	0.2312	0.0428
L	(51.76,-1.25)	(51.7560,-1.2529)	0.4933	2.0440e-05	0.4937	3.6689e-05	0.3765	0.0763
M	(51.76,-1.25)	(51.7571,-1.2540)	0.4294	2.0491e-05	0.4300	3.6563e-05	0.6078	0.1413
N	(51.76,-1.25)	(51.7585,-1.2548)	0.3697	2.0416e-05	0.3706	3.6606e-05	0.8703	0.2348
O	(51.76,-1.25)	(51.7600,-1.2550)	0.3442	2.0396e-05	0.3452	3.6647e-05	1.0021	0.2903
P	(51.76,-1.25)	(51.7615,-1.2548)	0.3697	2.0529e-05	0.3706	3.6641e-05	0.8703	0.2348
Q	(51.76,-1.25)	(51.7629,-1.2540)	0.4294	2.0509e-05	0.4300	3.6746e-05	0.6079	0.1414
R	(51.76,-1.25)	(51.7640,-1.2529)	0.4933	2.0450e-05	0.4937	3.6648e-05	0.3767	0.0763
S	(51.76,-1.25)	(51.7648,-1.2515)	0.5395	2.0525e-05	0.5397	3.6749e-05	0.2316	0.0429
T	(51.76,-1.25)	(51.7650,-1.2500)	0.5561	2.0288e-05	0.5563	2.9720e-05	0.1829	0.0329

[*]Execution time
[†]Percentage Difference

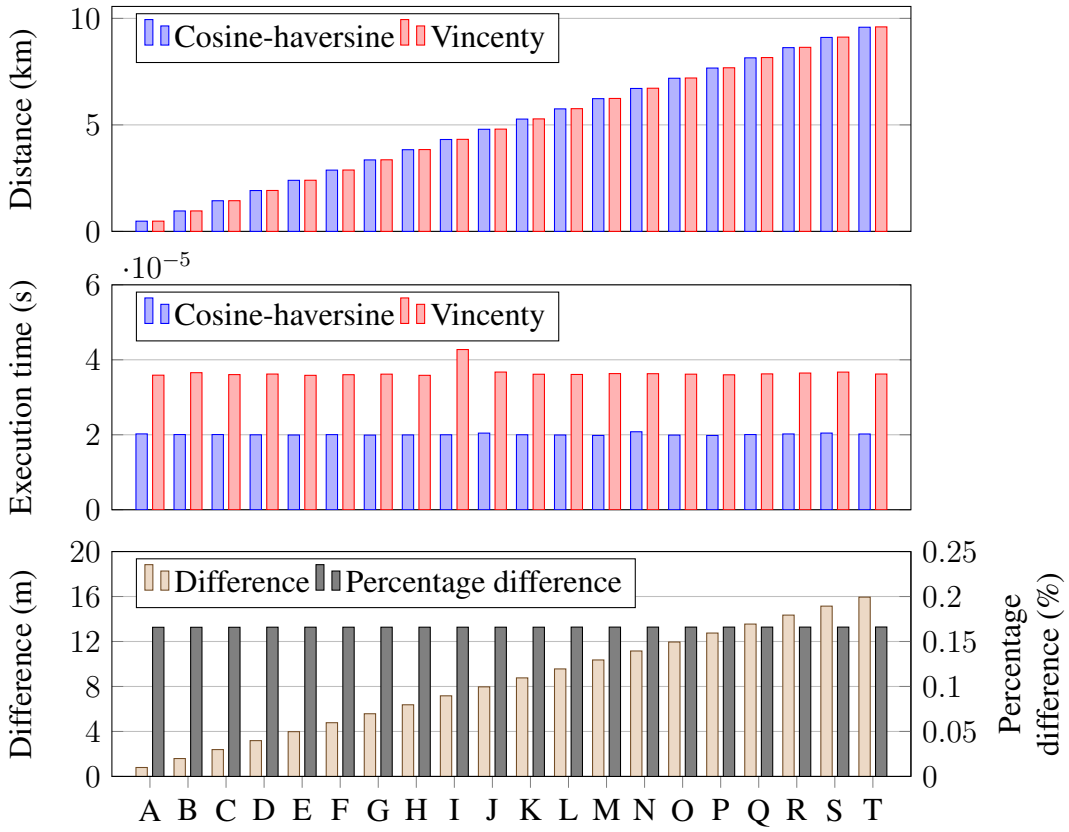


Figure 4.6: Graphical representations for comparison of consine-haversine and Vincenty formulae in typical area

Figure 4.7: Points for cosine-haversine and Vincenty formulae comparison in typical area (different bearing) with point 0 as the centre of the typical area

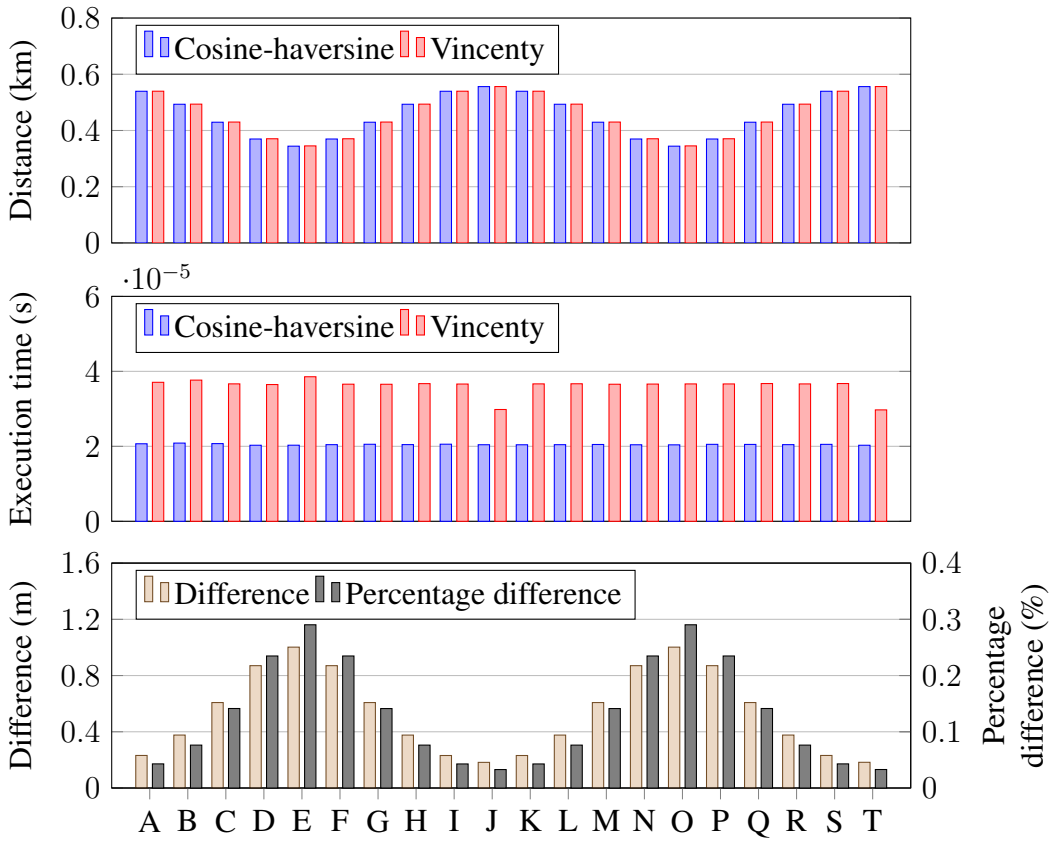


Figure 4.8: Graphical representations for comparison of consine-haversine and Vincenty formulae in typical area (different bearing)

The results show that the percentage difference between the calculated distance using the two methods are small, for the purpose of reducing the execution time and computational complexity, cosine-haversine formula is used to calculate the distance between GPS points.

4.1.2 GPS location interpolation

Given the GPS location of two geographical points, points that are separated by a specific distance can be interpolated between them. The interpolation formula is based on Slerp to calculate the intermediate point between two GPS points (ϕ_1, λ_1) and (ϕ_2, λ_2). Equations 4.16 to 4.22 show the formulae to calculate an intermediate point (ϕ_i, λ_i) between points (ϕ_1, λ_1) and (ϕ_2, λ_2) at a fraction f_ι of the distance between the two points measured from point (ϕ_1, λ_1) [104].

$$a_\iota = \frac{\sin((1 - f_\iota)\delta_\iota)}{\sin \delta_\iota} \tag{4.16}$$

$$b_\iota = \frac{\sin(f_\iota \delta_\iota)}{\sin \delta_\iota} \tag{4.17}$$

$$x_\iota = a_\iota \cos \phi_1 \cos \lambda_1 + b_\iota \cos \phi_2 \cos \lambda_2 \tag{4.18}$$

$$y_\iota = a_\iota \cos \phi_1 \sin \lambda_1 + b_\iota \cos \phi_2 \sin \lambda_2 \tag{4.19}$$

$$z_\iota = a_\iota \sin \phi_1 + b_\iota \sin \phi_2 \tag{4.20}$$

$$\phi_i = \text{atan2}(z_\iota, \sqrt{x_\iota^2 + y_\iota^2}) \tag{4.21}$$

$$\lambda_i = \text{atan2}(y_\iota, x_\iota) \tag{4.22}$$

where f_ι is the fraction along great circle route

($f_\iota = 0$ is point (ϕ_1, λ_1), $f_\iota = 1$ is point (ϕ_2, λ_2)),

δ_ι is the angular distance, $\frac{d}{R_E}$ between the two points

As the area of interest in this work only spans over a small area relative to the Earth's radius, the angular distance is close to zero ($\delta_\iota \rightarrow 0$). After applying the small angle approximation of sine ($\sin\theta \approx \theta$),

$$a_\iota \approx 1 - f_\iota \tag{4.23}$$

$$b_\iota \approx f_\iota \tag{4.24}$$

and therefore Equation 4.21 can be reduced to

$$\tan\phi_i = \frac{(1-f_\iota)\sin\phi_1 + f_\iota \sin\phi_2}{\sqrt{(1-f_\iota)^2\cos^2\phi_1 + f_\iota^2\cos^2\phi_2 + 2f_\iota(1-f_\iota)\cos\phi_1\cos\phi_2\cos(\lambda_2-\lambda_1)}} \tag{4.25}$$

Since the difference between the longitude of two points is small ($\lambda_2 - \lambda_1 \rightarrow 0$), $\cos(\lambda_2 - \lambda_1)$ is approximated to $1 - \frac{(\lambda_2-\lambda_1)^2}{2}$. As the longitude difference is relatively much smaller than one ($\lambda_2 - \lambda_1 \ll 1$), the approximation can be reduced to just one ($1 - \frac{(\lambda_2-\lambda_1)^2}{2} \rightarrow 1$). Equation 4.25 is further reduced to Equation 4.26.

$$\tan \phi_i = \frac{(1 - f_\iota) \sin \phi_1 + f_\iota \sin \phi_2}{(1 - f_\iota) \cos \phi_1 + f_\iota \cos \phi_2} \tag{4.26}$$

Using the two opposite vertices of the testing area ((ϕ_1°, λ_1°) as (52.0°, -1.6°) and (ϕ_2°, λ_2°) as (51.5°, -0.9°)), the values of $(1 - f_\iota) \sin \phi_1 + f_\iota \sin \phi_2$ is compared with $\sin((1 - f_\iota)\phi_1 + f_\iota \phi_2)$ and $(1 - f_\iota) \cos \phi_1 + f_\iota \cos \phi_2$ with $\cos((1 - f_\iota)\phi_1 + f_\iota \phi_2)$ for the inclusive range of f_ι between zero and one. The comparison between the two functions is quantified using the maximum error. If we let $F_1(f_\iota)$ be $(1 - f_\iota) \sin \phi_1 + f_\iota \sin \phi_2$ and $F_2(f_\iota)$ be $\sin((1 - f_\iota)\phi_1 + f_\iota \phi_2)$, the maximum error is given by Equation 4.27.

$$\text{maximum error} = \max\left(|F_1(f_\iota) - F_2(f_\iota)|\right) \quad \text{for } f_\iota \in [0, 1] \tag{4.27}$$

The comparison between $(1 - f_\iota) \sin \phi_1 + f_\iota \sin \phi_2$ and $\sin((1 - f_\iota)\phi_1 + f_\iota \phi_2)$ gives a maximum error of $7.48 \times 10^{-6^\circ}$. The comparison between $(1 - f_\iota) \cos \phi_1 + f_\iota \cos \phi_2$ with $\cos((1 - f_\iota)\phi_1 + f_\iota \phi_2)$ also gives a maximum error of $5.89 \times 10^{-6^\circ}$. Therefore the approximations in Equations 4.28 and 4.29 are true. Equation 4.26 can then approximated as Equation 4.32, which is a linear interpolation. This is effectively a small angle approximation.

$$(1 - f_\iota) \sin \phi_1 + f_\iota \sin \phi_2 \approx \sin((1 - f_\iota)\phi_1 + f_\iota \phi_2) \tag{4.28}$$

$$(1 - f_\iota) \cos \phi_1 + f_\iota \cos \phi_2 \approx \cos((1 - f_\iota)\phi_1 + f_\iota \phi_2) \tag{4.29}$$

$$\tan\phi_i \approx \frac{\sin((1-f_\iota)\phi_1 + f_\iota\phi_2)}{\cos((1-f_\iota)\phi_1 + f_\iota\phi_2)} \tag{4.30}$$

$$= \tan((1-f_\iota)\phi_1 + f_\iota\phi_2) \tag{4.31}$$

$$\phi_i = (1-f_\iota)\phi_1 + f_\iota\phi_2 \tag{4.32}$$

The values of interpolated latitude ϕ_i between the two opposite vertices of the testing geographical area using Equation 4.26 and 4.32 are compared in Figure 4.9. The maximum approximation error is found to be $6.09 \times 10^{-7^\circ}$.

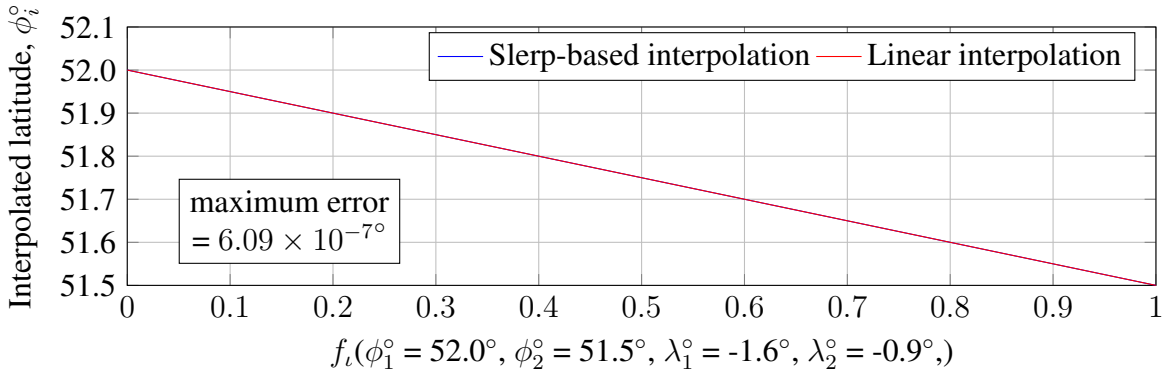


Figure 4.9: Comparison of interpolated latitude using Slerp-based interpolation and linear interpolation

Similarly for longitude, Equation 4.22 is reduced to Equation 4.33 using small angle approximation.

$$\tan\lambda_i = \frac{(1-f_\iota)\cos\phi_1 \sin\lambda_1 + f_\iota \cos\phi_2 \sin\lambda_2}{(1-f_\iota)\cos\phi_1 \cos\lambda_1 + f_\iota \cos\phi_2 \cos\lambda_2} \tag{4.33}$$

The values of Equation 4.33 for f_ι within the inclusive range of zero and one are compared with the values of $\tan((1-f_\iota)\lambda_1 + f_\iota\lambda_2)$. The values for ϕ_1, ϕ_2, λ_1, and λ_2 are the

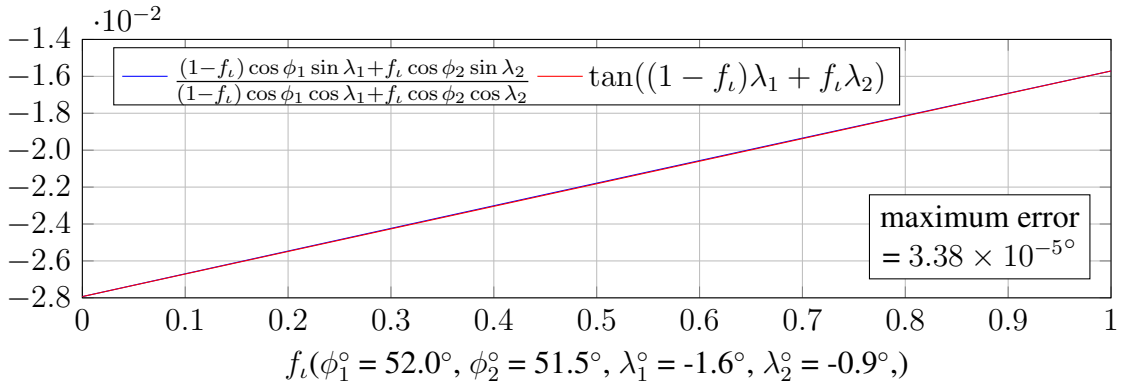


Figure 4.10: Comparison of $\frac{(1-f_\iota)\cos\phi_1\sin\lambda_1+f_\iota\cos\phi_2\sin\lambda_2}{(1-f_\iota)\cos\phi_1\cos\lambda_1+f_\iota\cos\phi_2\cos\lambda_2}$ with $\tan((1-f_\iota)\lambda_1+f_\iota\lambda_2)$

coordinates for the two opposite vertices of the testing area. The maximum error between the two is $3.38 \times 10^{-5^\circ}$. This shows that the approximation in Equation 4.34 is true and therefore Equation 4.33 can be approximated as linear interpolation in Equation 4.36.

$$\frac{(1-f_\iota)\cos\phi_1\sin\lambda_1+f_\iota\cos\phi_2\sin\lambda_2}{(1-f_\iota)\cos\phi_1\cos\lambda_1+f_\iota\cos\phi_2\cos\lambda_2} \approx \tan((1-f_\iota)\lambda_1+f_\iota\lambda_2) \tag{4.34}$$

$$\tan\lambda_i \approx \tan((1-f_\iota)\lambda_1+f_\iota\lambda_2) \tag{4.35}$$

$$\lambda_i = (1-f_\iota)\lambda_1+f_\iota\lambda_2 \tag{4.36}$$

The interpolated longitudes λ_i between the two opposite vertices of the testing geographical area computed using the Slerp-based formulae (Equation 4.33) and linear interpolation (Equation 4.36) are compared in Figure 4.11. The maximum approximation error is identified to be 0.00194°.

The approximations of Equations 4.26 and 4.33 to linear interpolation are also examined with the typical geographical area. The results in Figure 4.12 show that the maximum

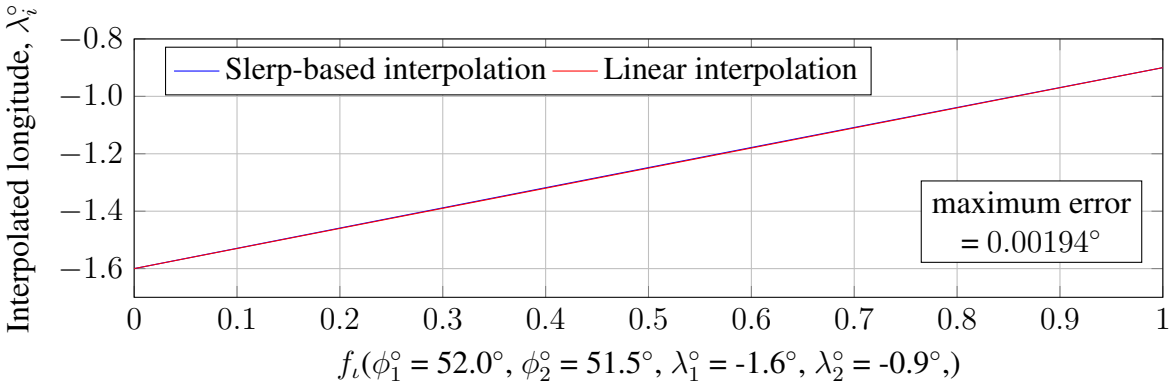


Figure 4.11: Comparison of interpolated longitudes using Slerp-based interpolation and linear interpolation

percentage errors for interpolated latitude ϕ_i and longitude λ_i are small.

Therefore, for the purpose of reducing computational demands, this work, unless specified, always uses linear interpolation to interpolate between geographical points.

For the purpose of investigating the effect of the earth's curvature to this approximation, a GPS coordinates (30.1,30.1) is chosen as the starting point. Interpolations with the fraction, f_ι as 0.5 are performed between the starting point to all possible GPS coordinates on Earth (rounded to 0.5°). The errors of the two interpolation methods are shown in Figure 4.13. Equations 4.16 to 4.22 are used for the Slerp-based interpolation while Equations 4.32 and 4.36 are used for the linear interpolation. Only errors below 10° are shown in the figures to allow more accurate representation.

The approximation with linear interpolation for latitude interpolation holds true within the error of 0.5° when the interpolation is performed between the starting point and the points with longitude of $30.1 \pm 15°$. The approximation with linear interpolation for longitude interpolation is more limited. It only holds true within the error of 0.5° when the inter-

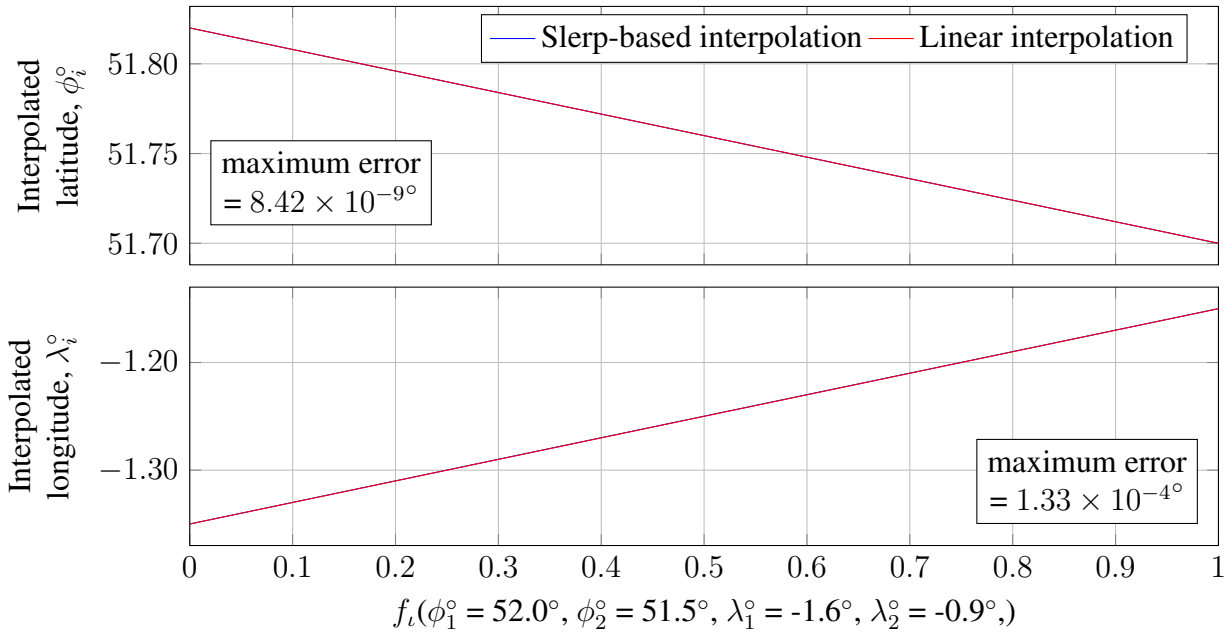


Figure 4.12: Comparison of approximation results with the typical geographical area

polation is performed between the starting point and the points with longitude of $30.1 \pm 10^\circ$ and latitude of $|30.1| \pm 10^\circ$.

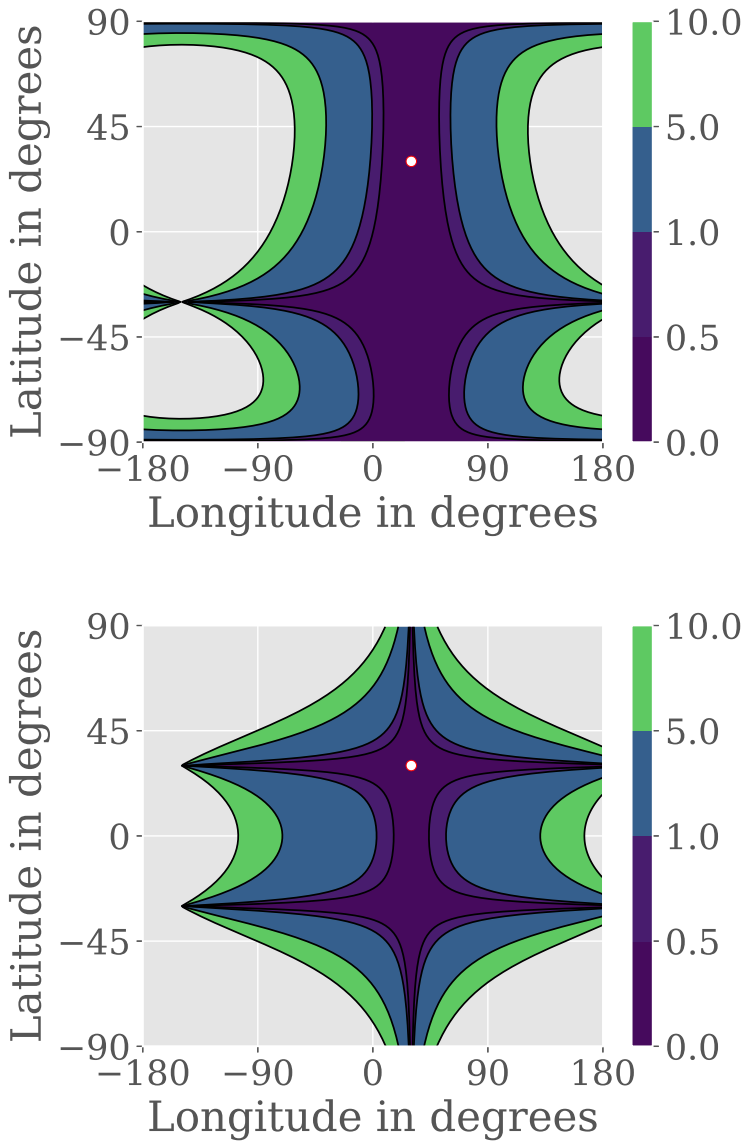


Figure 4.13: Errors between Slerp-based interpolation and linear interpolation from (30.1,30.1), denoted by the white dot with red border, to all possible GPS coordinates on Earth (rounded to 0.5°) with f_ι as 0.5. Top graph shows the errors for latitude interpolation and bottom graph shows the errors for longitude interpolation.

One common usage of GPS points interpolation is to interpolate repeatedly between two points at a fixed separation. Figure 4.14 shows the algorithm to perform repetitive

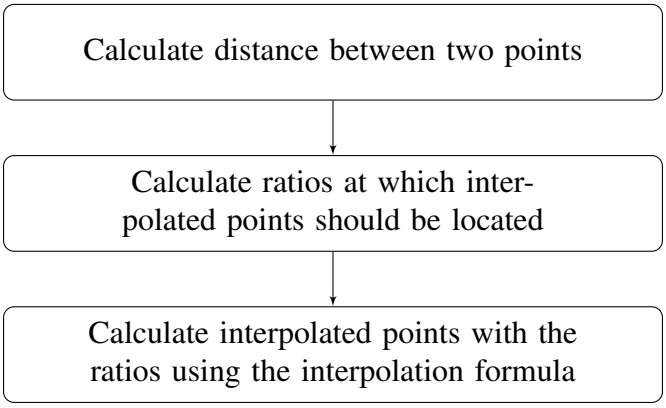


Figure 4.14: Algorithm for repetitive interpolation

interpolation.

4.1.3 GPS bound identification

In some cases, a small area instead of a single point is of interest. This area is constructed from a centre point p_o and distance d_p of the boundary from the point. Ideally this will be a circle with centre p_o and radius d_p. However, for computational simplicity to determine if a point lies within the bound, a square with centre p_o and sides of length $2d_p$ as shown in Figure 4.15 is used.

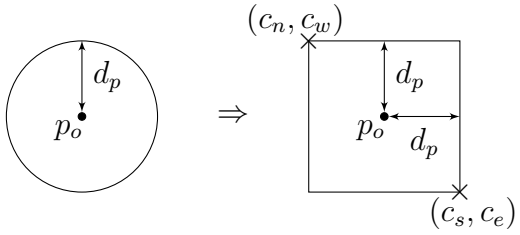


Figure 4.15: Bound illustration

Given the GPS coordinates of p_o as $(c_h,\ c_v)$, the task of identifying the bound is to identify the GPS coordinates of two opposite vertices of the square, namely $(c_n,\ c_w)$ for

north-west vertex and (c_s, c_e) for south-east vertex. These coordinates are determined one at a time. The coordinates identification algorithm is explained using the determination of c_w as example.

To determine the value of c_w, the longitude of the north-west vertex, the latitude is kept at the same value as the latitude of the centre. This is essentially the centre point of the west side of the square, p_w. An intermediate point p_i with coordinates of (c_{hi}, c_{vi}) is first identified such that the distance between p_o and p_i, $d_{p_o p_i}$ is further than that between p_o and p_w, and the point p_w lies on the line joining p_o and p_i. As p_w and p_o has the same latitude, the latitude of p_i has the same value too, i.e. c_{vi} = c_v. The position of p_i is illustrated in Figure 4.16.

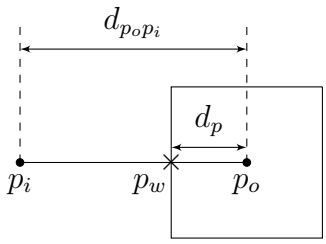


Figure 4.16: Illustration of relevant points in coordinates identification algorithm

This thus reduces the problem of identifying the coordinates of p_w to an interpolation problem. As previously stated in Section 4.1.2, the interpolation between two points within the testing geographical area can be approximated using linear interpolation. Therefore the longitude of p_w, c_w can be identified with Equation 4.37.

$$c_w = \left(1 - \frac{d_p}{d_{p_o p_i}}\right) c_h + \frac{d_p}{d_{p_o p_i}} c_{hi} \tag{4.37}$$

The coordinates c_n, c_s and c_e can be determined with the same method with different direction from p_o.

The inclusion of a point in the square bound can be easily identified by performing a range comparison of the point's coordinates with the the coordinates of the bounds.

4.2 Bus route processing

A bus route is a series of connecting road segments associated with a bus service. The list of the road segments are obtained from the traffic information portal of Oxfordshire County Council available on the internet [107]. This is due to the fact that some of the real-world data is collected from the site and the formation of bus route from the road segments allows easy integration of data from the road segment to the bus route if necessary.

A Javascript tool is developed to select and combine road segments from a list. The workflow to specify road segments and form a bus route is presented in Figure 4.17.

The road segments that form the current bus route are selected in sequence from a list of available road segments. Whenever a road segment is added to the list, the distance between the first point of the new road segment and the last point of the last road segment is calculated. If this distance is greater than a defined gap size, interpolated points are calculated using the formulae in Section 4.1.2. The interpolated points are then processed with the Google Maps Roads API Snap to Roads service to ensure the points lie on the road. The interpolated points are then saved to the route between the new and the previous road segments.

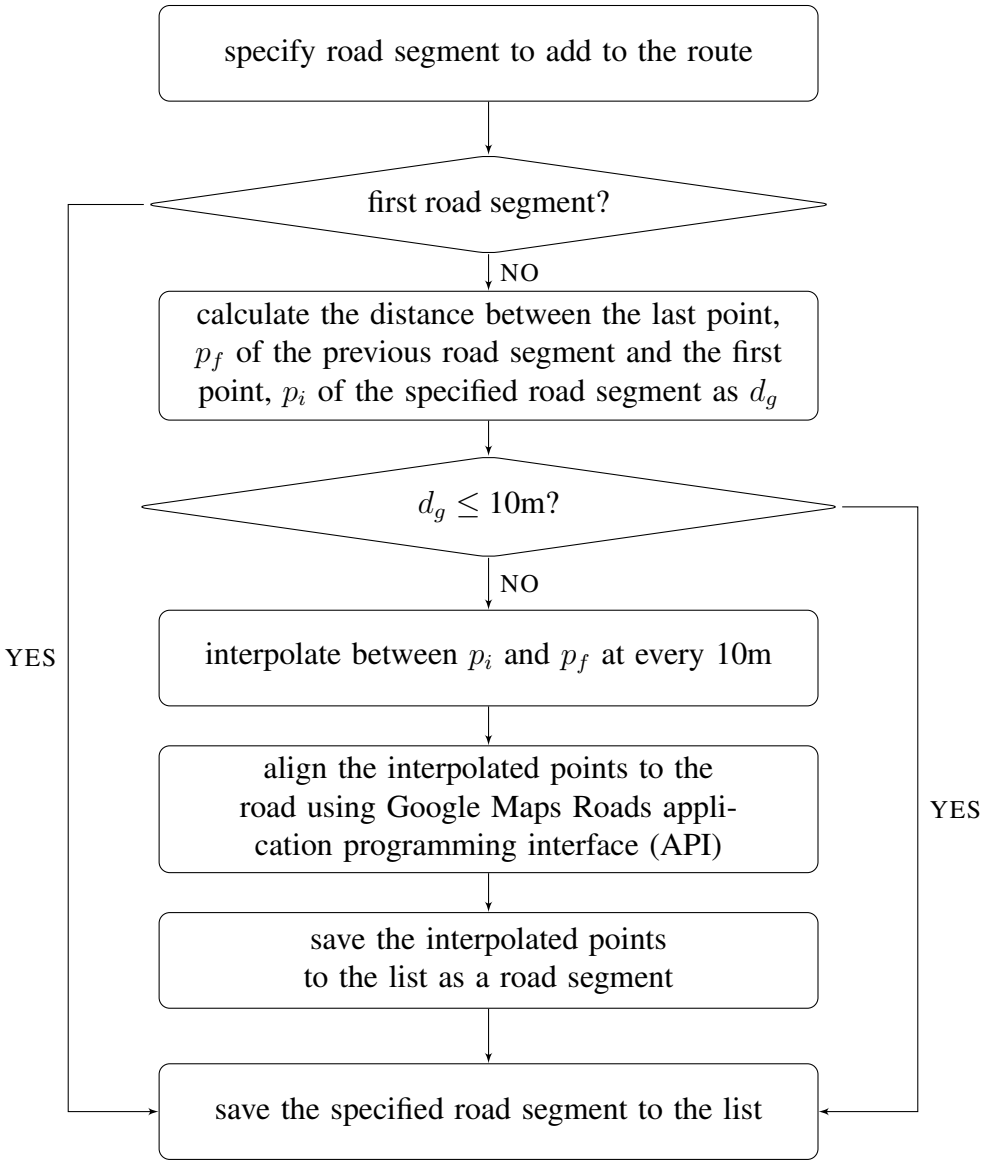


Figure 4.17: Workflow to add road segment to bus route

The elevations of the road segments are obtained through the Google Maps Elevation API. The illustration of a basic slope is shown in Figure 4.18. The slopes are calculated from the elevation using the basic trigonometric formula in Equation 4.38.

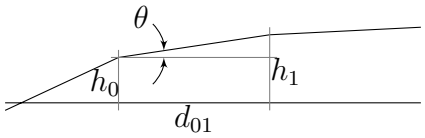


Figure 4.18: Relationship of slope and elevation

$$\theta = \tan^{-1} \frac{h_1 - h_0}{d_{01}} \tag{4.38}$$

As the number of slope changes affect the computational complexity, an approximation algorithm based on a least square fit algorithm in Clements's work [108] is used to maximise the accuracy of representation of elevation and minimise the number of changes in the slope. The approximation algorithm is applied on the elevation profile to identify the minimum number of connected straight lines sufficient to represent the elevation profile which is later converted to the slope profile.

The algorithm starts by performing a least square fit of a straight line on the first n_0 points of the elevation profile. The straight line models the elevation h as a function of horizontal distance d from the first data point as in Equation 4.39. b_0 and b_1 are two parameters to be identified by the least square fitting.

$$h = b_0 d + b_1 \tag{4.39}$$

A slight modification is done to ensure the continuity of the fitted lines by fixing the value of the first point of the line using the value of the last point of the previous line. Using the example in Figure 4.19, the linear function in Equation 4.39 is modified by shifting the horizontal distance d by the value of the horizontal distance of the first point d_0 in line l_0 so that the elevation of the first point h_0 is used as the last term of the straight line function. Therefore the function to be fitted for l_0 starts as Equation 4.40. The least square fitting is only responsible in finding the value of b_0.

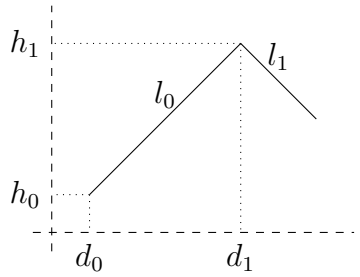


Figure 4.19: Example of fitting two consecutive lines

$$h = b_0(d - d_0) + h_0 \tag{4.40}$$

The unbiased standard deviation σ_u of the fit is then determined using Equation 4.41. The elevation of the next point is calculated with the fitted function using its horizontal distance. If the actual elevation is within an integral multiple w of the standard deviation of the calculated elevation, the line is re-fitted with the new point included. Otherwise, a new line is started with the calculated elevation of the previous point and the next n_0 points. The approximated elevation is converted into the slope profile using basic trigonometric

formula in Equation 4.38.

$$\sigma_u = \frac{\text{mean square error}}{n_t - 2} \tag{4.41}$$

The two variables that determine the accuracy of the approximated profiles are w, the multiplication factor of standard deviation and n_0, the number of initial points when a new line is created. The multiplication factor w determines how far a new point is from the current fitted line before a new line should be initiated. The minimum value of w is 1 as w is defined to be an integer. A value of zero for w means that the next point must fit on the current fitted line for a new line not be created.

The number of initial points n_0 affects the fitted lines at initial state. The higher the value of n_0, the larger the initial mean square error of the line, and therefore the higher the possibility for a new point to be incorporated to the current line. The minimum value of n_0 is 3 as a value of 2 causes the fitted line to have zero mean square error and in most situations, the next point will never be incorporated. The approximated result will then be a series of linear functions that each of them only connects to 2 points.

Parameter sweeps are performed for w from 1 to 10 and n_0 from 3 to 10. The bus route used in this comparison is the bus route from Oxford City Centre to Pear Tree Park and Ride as shown in Figure 4.20. The mean square errors of the approximated profiles for elevations and slopes are presented in Figures 4.21 and 4.22. Figures 4.21 and 4.22 both show lower mean squared errors at lower values of n_0. A desirable approximation is such

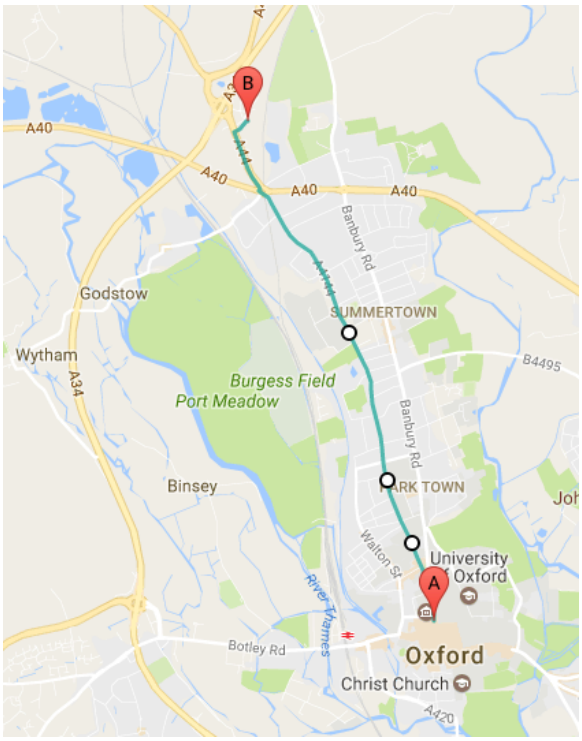


Figure 4.20: Bus route from Oxford City Centre (A) to Pear Tree Park and Ride (B) for elevation extraction

that the approximated profile is accurate, i.e. has a low mean squared error compares to the original profile, and the number of slope change is at minimum. Therefore the normalised mean squared errors from the two figures and the normalised number of slope change at each combination of w and n_0 are summed to produced Figure 4.23. The decision of the values for w and n_0 is made by comparing the sum of the normalised mean square errors and the normalised number of slope change as shown in Figure 4.23.

The original and approximated profiles for the bus route is visualised in Figure 4.24 using the values of w as 1 and n_0 as 7. The mean square errors of the approximated elevation and slope profiles are 0.056 and 0.189 respectively. The observation on Figure 4.24 shows that the adjusted elevation profile between 750 to 1000 meters does not cap-

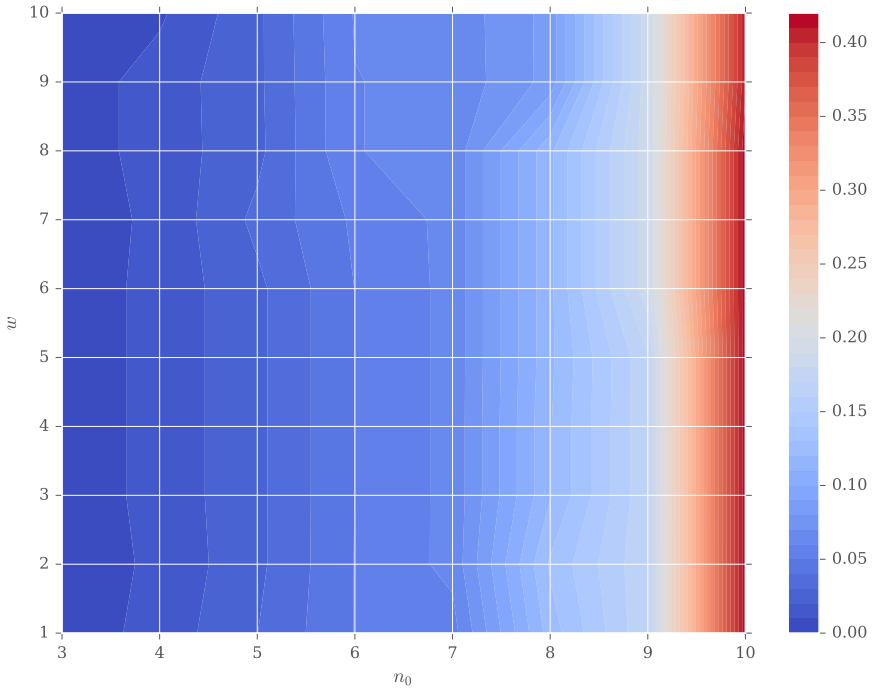


Figure 4.21: Mean square errors of adjusted elevation profile in parameter sweep

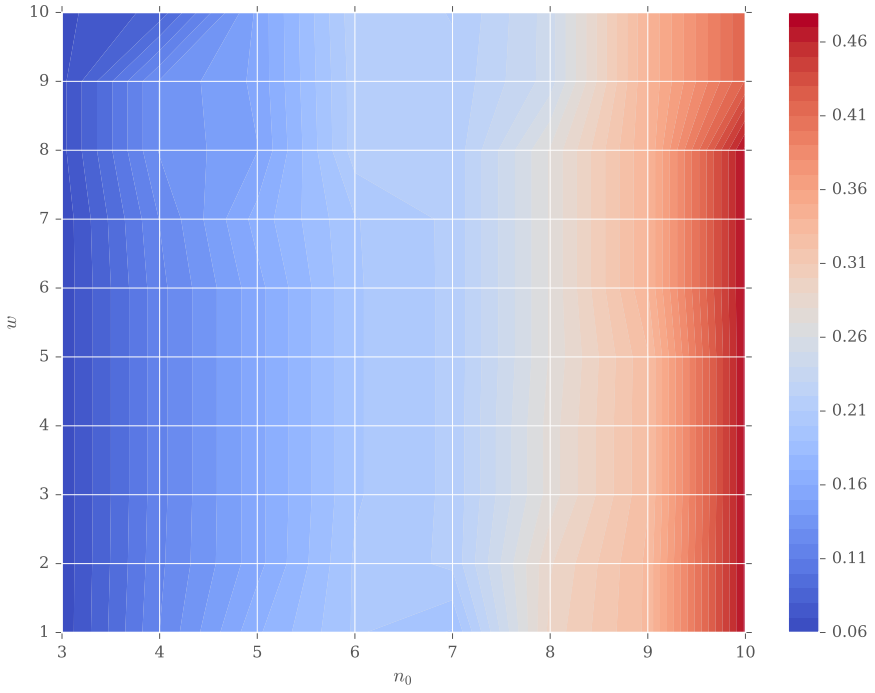


Figure 4.22: Mean square errors of adjusted slope profile in parameter sweep

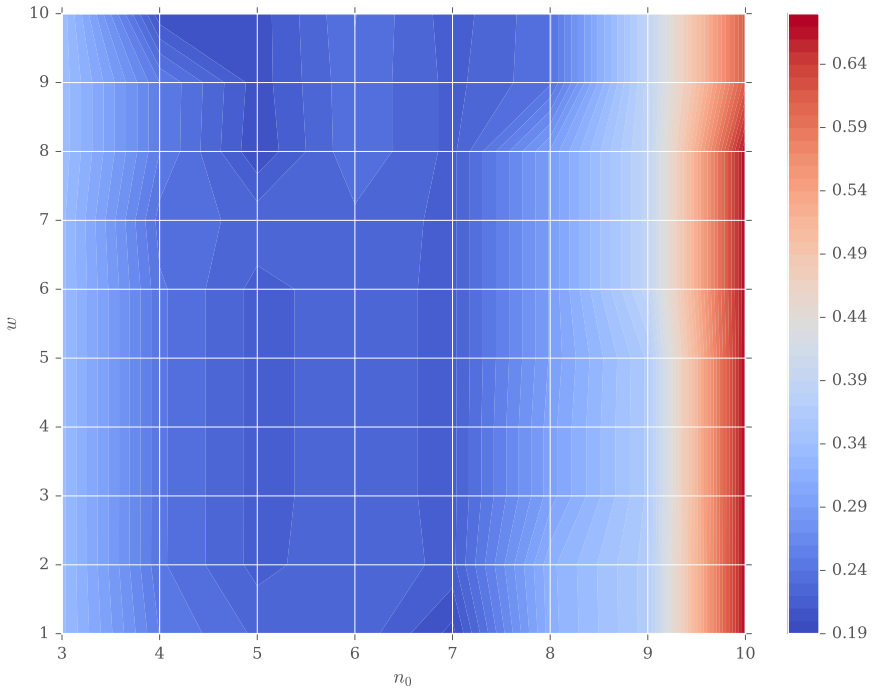


Figure 4.23: Sum of the normalised mean square errors of adjusted elevation and slope profile and the normalised number of slope change

ture the high rate of change in the original elevation profile. However, as the discrepancy between the two profiles spans over a small distance, i.e. around 200 meters, the effect of the discrepancy will not be significant when used in later part of the work.

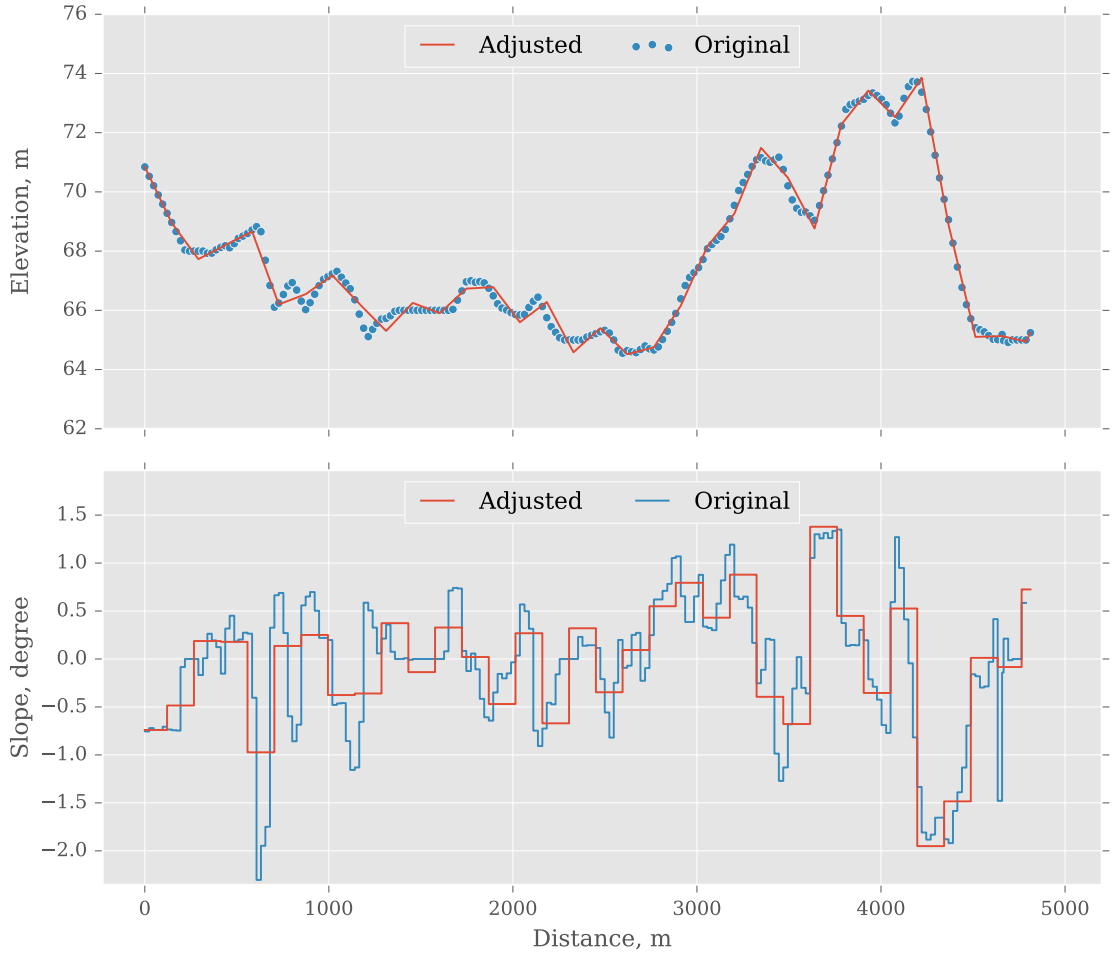


Figure 4.24: Original and adjusted elevation and slope profiles

4.3 Bus stop fitting

Bus stops along a bus route are specified in the form of GPS coordinates separately after the bus route is formed from the road segments. The relative position of the bus stops in the route is required to include the bus stops into the computation of optimal control. The process of identifying the relative position of a bus stop is termed as bus stop fitting.

Figure 4.25 shows an example of a bus stop along a bus route. β is the bus stop and $\mathbb{A}$,

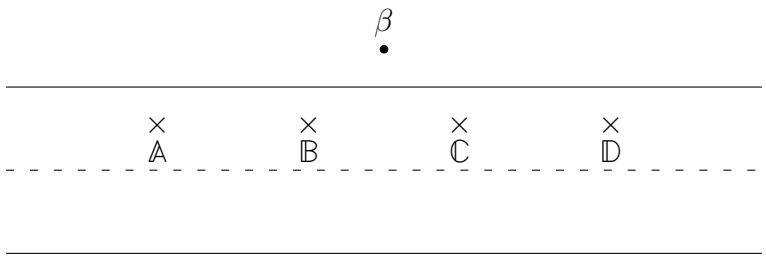


Figure 4.25: Example of bus stop along a bus route

$\mathbb{B}$, $\mathbb{C}$ and $\mathbb{D}$ are the points of the bus route. In this case, it is clear that the bus stop β lies between point $\mathbb{B}$ and $\mathbb{C}$, which are of shortest distance from the bus stop. However, only considering the closest points will give an incorrect result for the case in Figure 4.26.

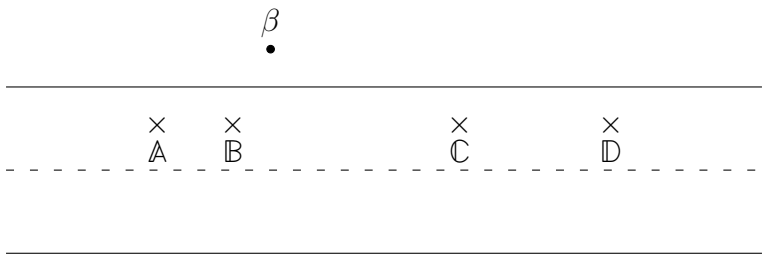


Figure 4.26: Modified example of bus stop along a bus route

The algorithm that identifies the bus stop to be between the two closest points will identify that the bus stop β lies between $\mathbb{A}$ and $\mathbb{B}$ instead of $\mathbb{B}$ and $\mathbb{C}$. Comparison of bearings from the bus stop to each points helps solving this issue.

The bearings from bus stop β to points $\mathbb{A}$, $\mathbb{B}$ and $\mathbb{C}$ are denoted as bearing 1, 2 and 3

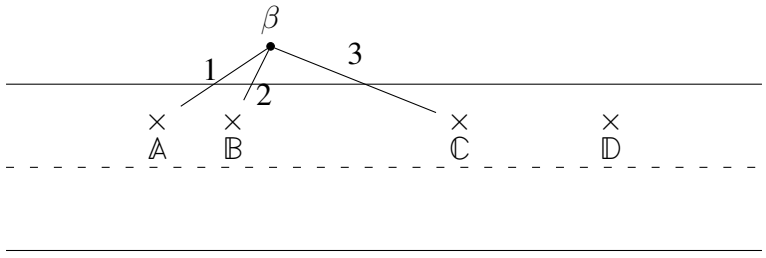


Figure 4.27: Modified example of bus stop along a bus route

respectively. The angular difference between bearing 2 and 3 is larger than that between 1 and 2. Therefore the bus stop should lie between points $\mathbb{B}$ and $\mathbb{C}$. This holds true for all positions of β between points $\mathbb{B}$ and $\mathbb{C}$.

The procedure can be summarised as:

(i) identify three points from the series closes to β, $\mathbb{B}$, $\mathbb{A}$ and $\mathbb{C}$, in the order of increasing distance from β,

(ii) find the angles of $\angle\mathbb{B}\beta\mathbb{A}$ and $\angle\mathbb{B}\beta\mathbb{C}$, and

(iii) the larger angle corresponds to the two points between which β lies.

4.4 Determination of optimal speed at on-demand bus stop

Bus stops are categorised as compulsory or on-demand bus stops. A compulsory bus stop is defined as having the speed constraint of zero at its respective location whereas an on-demand bus stop has no speed constraint. This means that the optimal control algorithm will assign an arbitrary speed such that the bus will arrive at the location of an on-demand bus stop at a specified time of arrival.

For the purpose of obtaining a speed at the on-demand bus stop that reflects the actual situation of the bus stop, two optimal speed profiles are identified. The first profile is such that the bus does not stop at the on-demand bus stop and the second profile provides the profile when the bus stops at the on-demand bus stop. The first profile is produced by performing optimal control with no speed constraint at the on-demand bus stop. The

second profile considers the speed constraint at the on-demand bus stop to be zero.

The resultant profile is formed by taking the weighted average of the two profiles, using the stop probability of a bus at the bus stop as the weighing parameter. This resultant profile is not necessarily optimal. The mathematical representation to produce the new profile p_{12} from the two profiles p_1 and p_2 with the stopping probability of P_s is shown in Equation 4.42.

$$p_{12} = (1 - P_s) \cdot p_1 + P_s \cdot p_2,\ P_s \in [0, 1] \tag{4.42}$$

4.5 Identification of stop probability of bus at bus stop

The stop probability at each on-demand bus stop is identified from the historical bus data between May 2015 to May 2016 obtained from Oxford Bus Company. In order to determine the stop probability at an on-demand bus stop, the significant temporal groups need to be identified such that the daily speed profiles of the buses passing by the bus stop are similar in the group but distinct to other groups. The grouping of the daily profiles are done with a clustering algorithm. The stop probability at the bus stop is then calculated for each temporal group.

Before the grouping of the daily profiles are carried out, the bus data related to a specific bus stop needs to be extracted from the raw data provided by Oxford Bus Company. As time discrepancy exists between two successively recorded points in the raw data, there are

cases where the last data before the bus stop and the first data after the bus stop are far from the bus stop. To solve this problem, the raw data is interpolated using algorithm described in Section 4.1.2 such that the separation between any two consecutive points is at most 10 m.

To extract the data related to a specific bus stop from the bus data, the GPS and bearing bounds of the bus stop need to be defined. GPS bound of a bus stop is calculated from the GPS coordinates of the bus stop and the distance of the boundary from the bus stop using the algorithm in Section 4.1.3. The bearing bound of a bus stop is defined as the bearing of the traffic direction with a tolerance value. The bearing bound is used to identify if the bus is moving on the direction that concerns the bus stop. The bus data that are within the GPS bound and having the bearing within the bearing bound are extracted as the data for the bus stop.

The highest temporal resolution of the bus data is 1 minute, therefore there might be points on the same minute. These duplicated points are aggregated by taking the minimum of the speeds. The minimum is used as it defines the mobility status of the bus, i.e. moving or stop.

As there is only sparse data (i.e. 3 stops in an hour), the extracted bus data is re-sampled to every x minutes. x is selected to be the lowest value at which the re-sampled data provide daily profiles with no missing data points. It has been found that to satisfy the criterion, x is between 30 minutes to 1 hour. The average speed of the original data is taken as the speed of the re-sampled data.

Clustering is then performed on the daily profiles of a bus stop data using k-means algorithm [109]. Each point in a profile is treated as a feature. If we let n_f be the number of features in a profile, X as a profile to be clustered, and Y as the centroid profile of a cluster, X and Y are column matrices with n_f elements. The distance between the two profiles is given by the Euclidean distance as shown in Equation 4.43.

$$\text{Euclidean distance} = \sum_{i=0}^{n_f-1} (X_i - Y_i)^2 \tag{4.43}$$

Figures 4.28 to 4.30 show the results of clustering daily profiles of the bus stop into 5 clusters. Figure 4.28 shows all the daily profiles in each cluster, and Figure 4.29 shows the centroid profiles of each cluster. The centroid profiles are calculated by taking the average of the daily profiles in the cluster. Figure 4.30 groups the clusters according to day of the week.

There are generally four temporal groups can be observed from Figure 4.30: (i) Sunday, (ii) Monday and Tuesday, (iii) Wednesday, Thursday and Friday, and (iv) Saturday .

The historical bus data are grouped according to the temporal groups to calculate stop probability. The stop probability at a bus stop is calculated by dividing the total counts of bus stopping by the total counts of bus passing as shown in Equation 4.44.

$$P_s = \frac{\sum n_{si}}{\sum N_i} \times 100\% \tag{4.44}$$

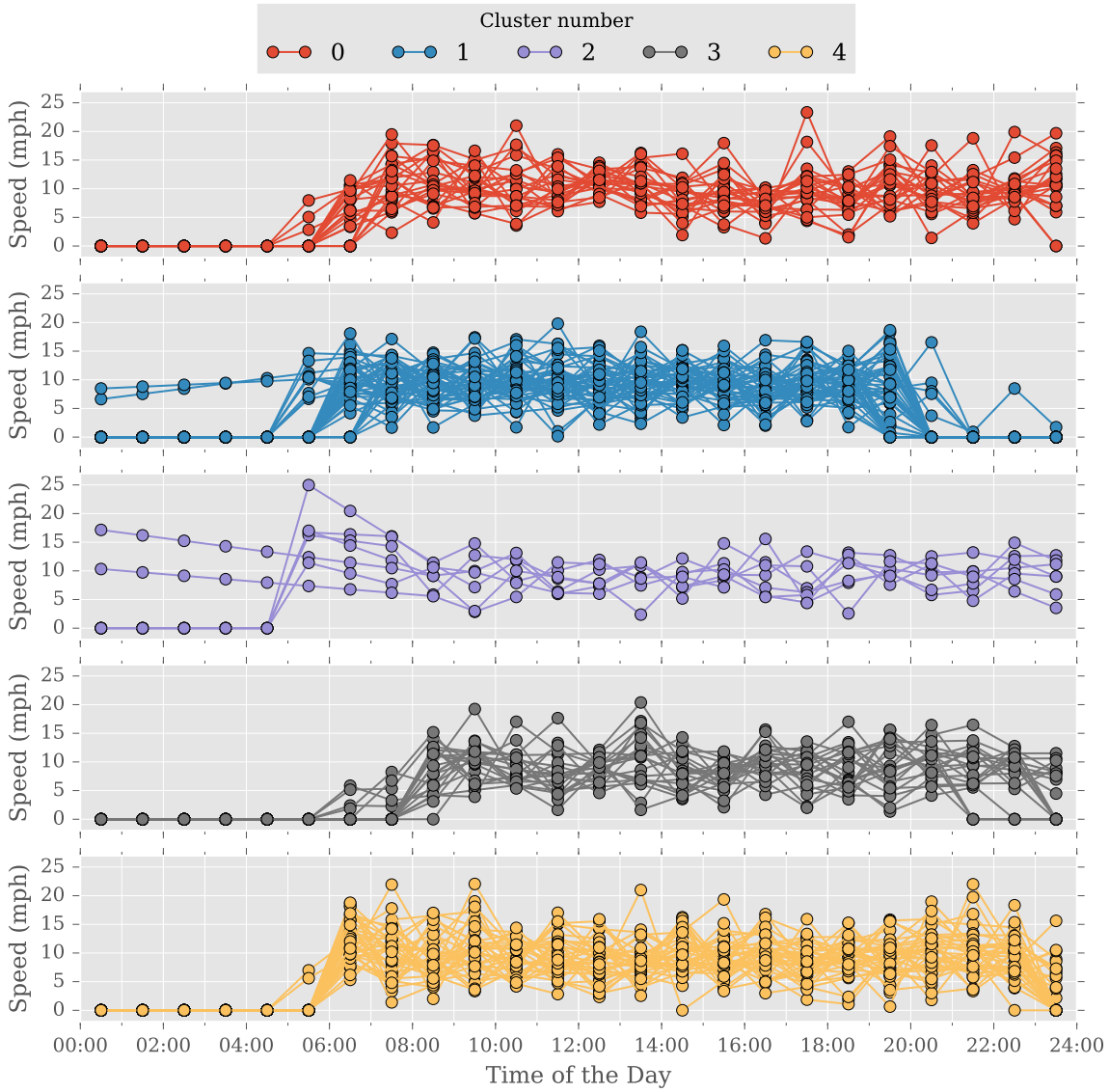


Figure 4.28: Daily profiles in each cluster

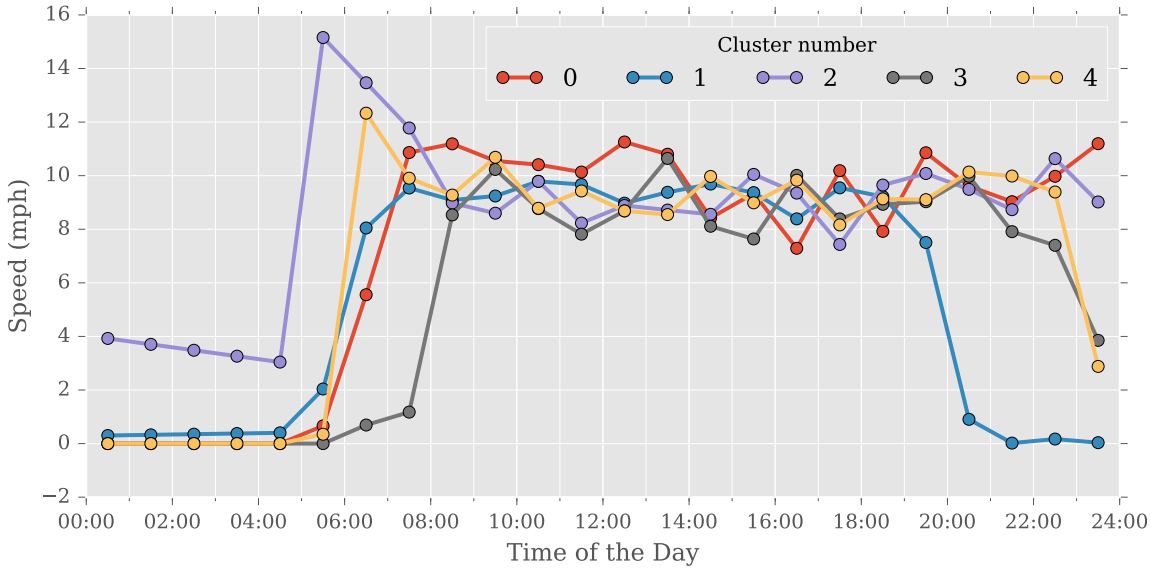


Figure 4.29: Centroids of daily profile clusters

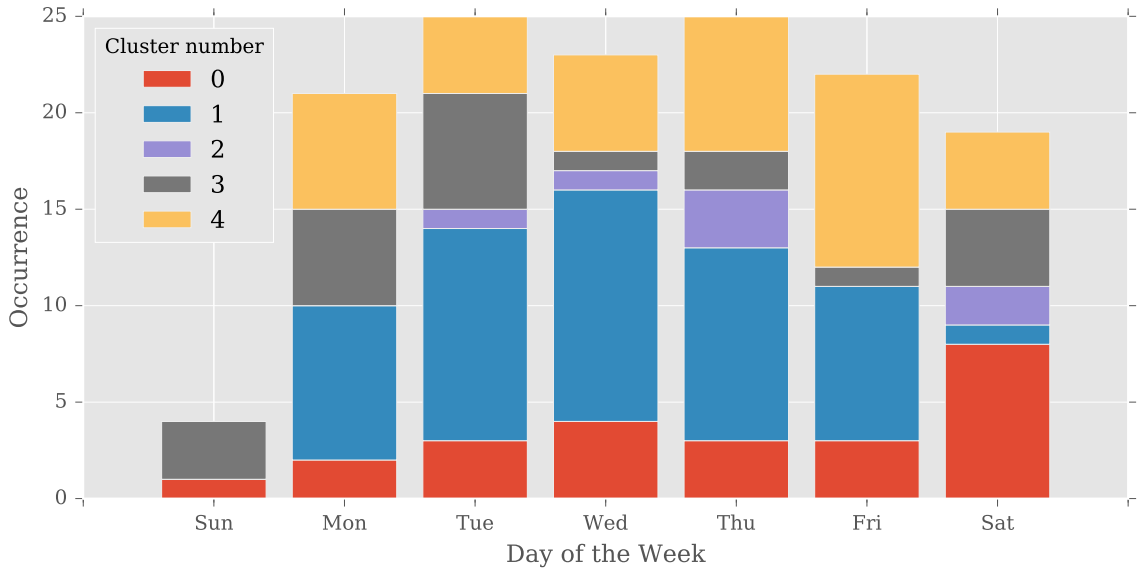


Figure 4.30: Analysis of clustered profiles

As the GPS devices do not always get an immediate fix at a location, the GPS logger might not have recorded the data of zero speed when the bus stops at the bus stop. However, the velocity of the bus before and after the stopping will be low. A speed threshold is identified such that below which the bus is considered stopped and above which the bus is moving.

The speed threshold is identified through a simple experiment. A mobile phone is set to log GPS data automatically and vehicle mobility status manually. The collected data are displayed in Figure 4.31. A sweep of speed threshold from zero to the maximum logged speed is done on the collected data to identify the amount of false positives and false negatives at different speed threshold. The result is shown in Figure 4.32. The optimal speed thresholds based on the result is around 7 to 8 mph.

The stop probability profile at the bus stop for each temporal group is then identified using the speed threshold of 7.5 mph and shown in Figures 4.33 to 4.36.

4.6 Summary

The real world implementation of optimal bus driving can only be achieved by translating real world information into formats that are understandable by the optimal control algorithm. Several methods have been presented. Two common formulae for GPS distance calculation, the cosine-haversine formula and the Vincenty formula, are compared. The cosine-haversine formula is chosen since the difference between the two formulae is small

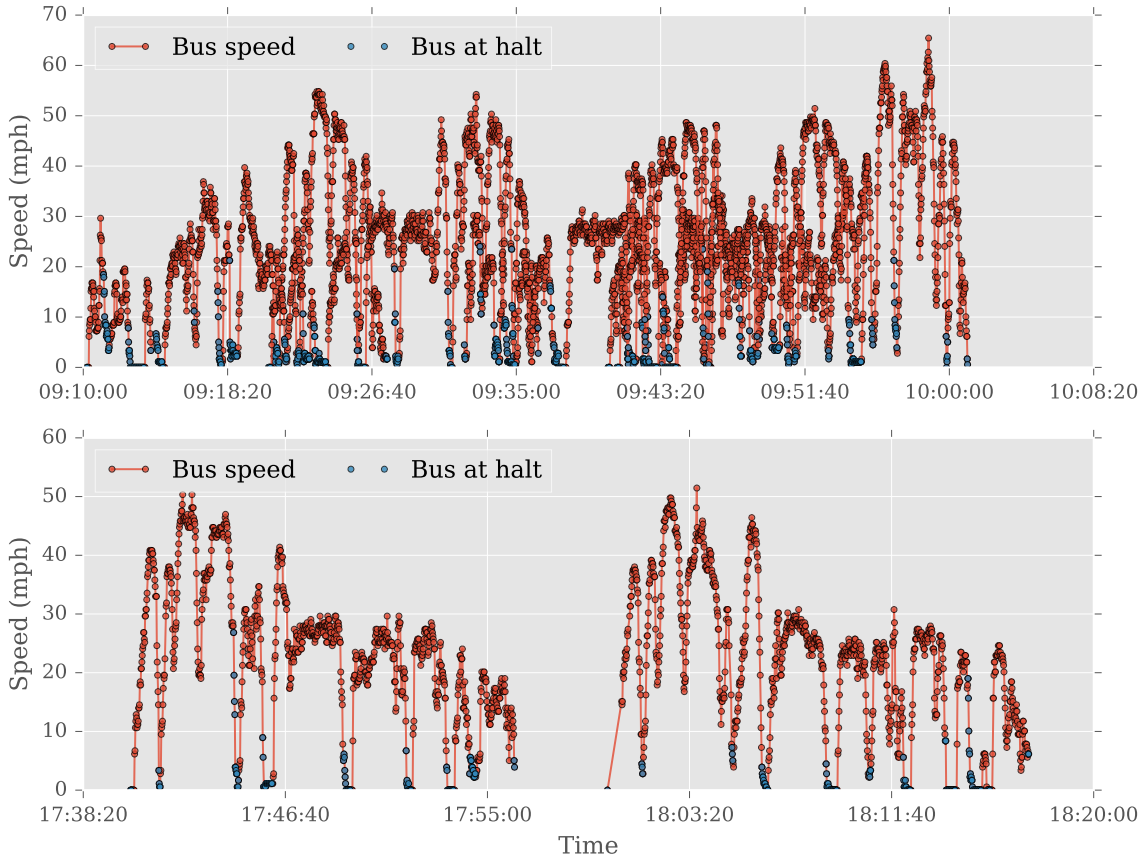


Figure 4.31: Collected data to identify speed threshold for vehicle halt identification

Figure 4.32: Percentage of false positives and false negative for using different speed threshold as halt

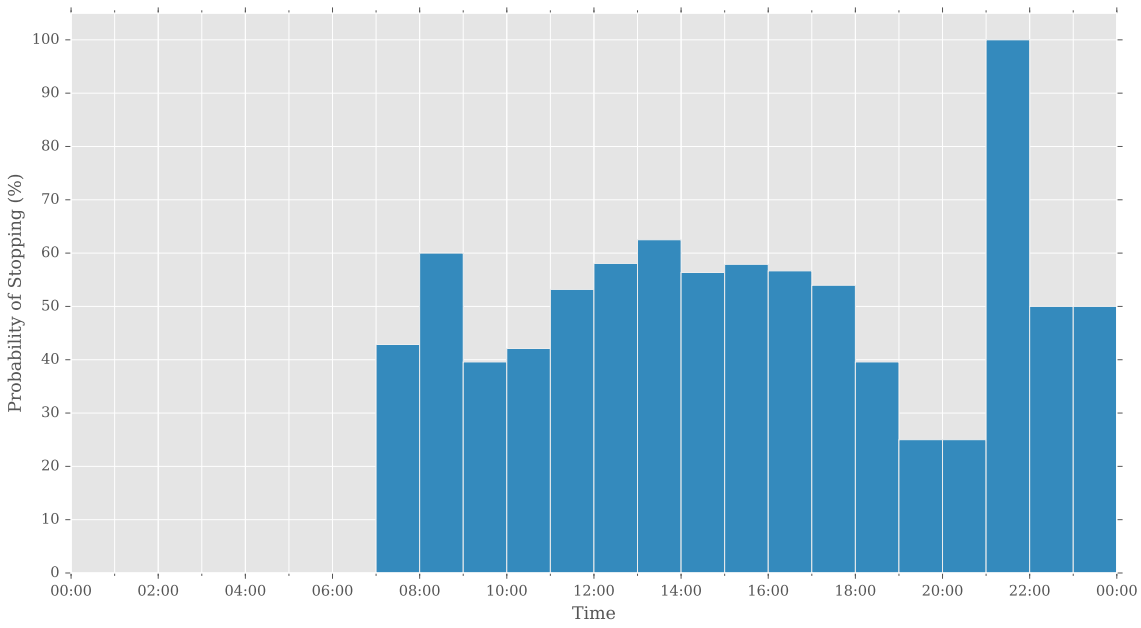


Figure 4.33: Stop probability on Sunday

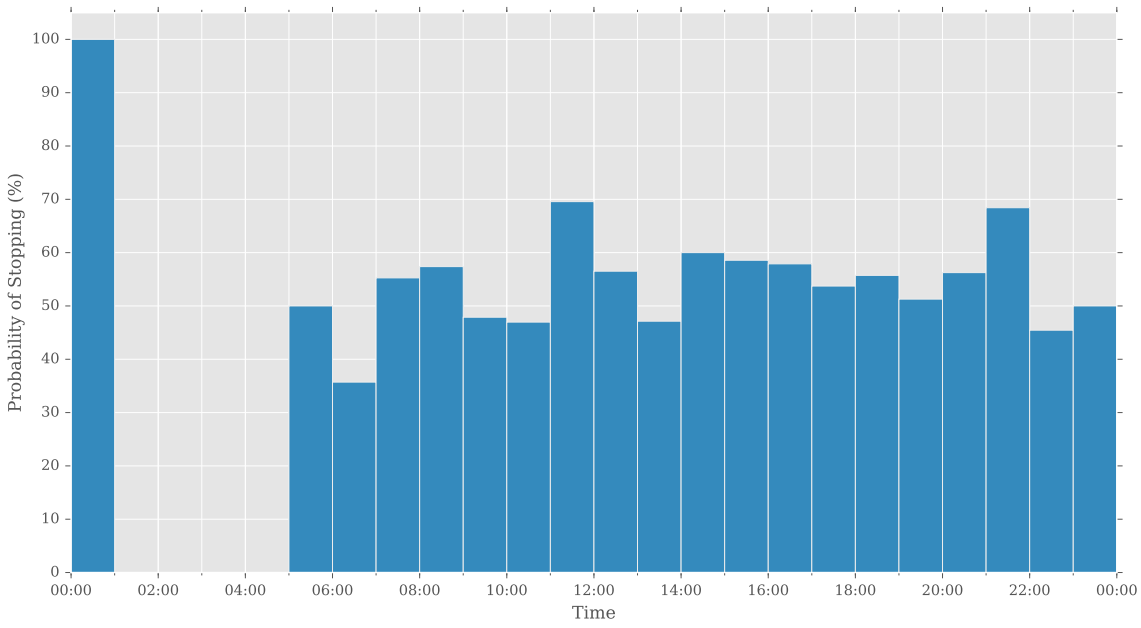


Figure 4.34: Stop probability on Monday and Tuesday

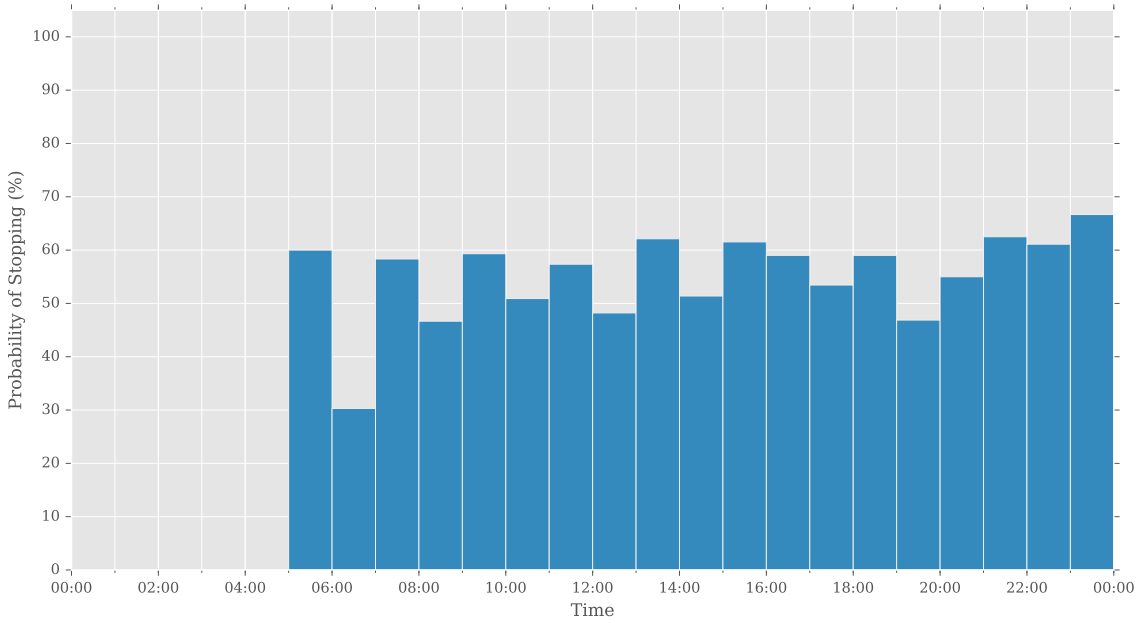


Figure 4.35: Stop probability on Wednesday, Thursday and Friday

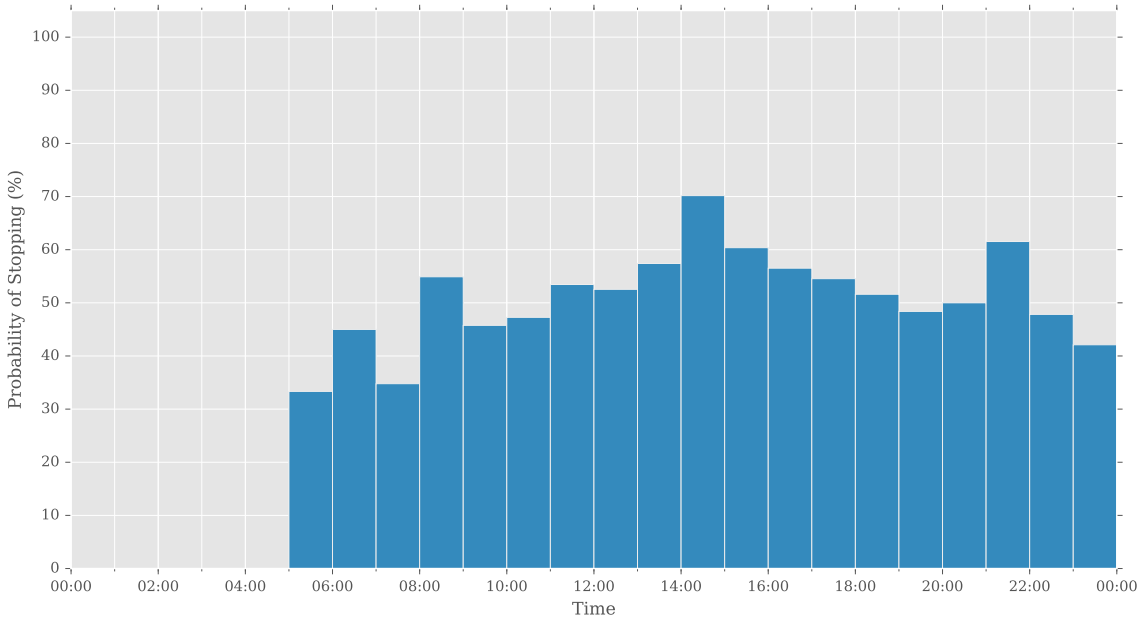


Figure 4.36: Stop probability on Saturday

at small distance and the cosine-haversine formula requires shorter execution time and less computational resources. Linear interpolation and Slerp-based interpolation are compared for GPS location interpolation. The comparison shows that a linear interpolation provides similar result as the Slerp-based interpolation when the distance between two GPS locations is near. Since the GPS points relevant in this work are at relatively close proximity to one another, the linear interpolation is used for GPS location interpolation to reduce computational complexity. An algorithm to identify GPS bound from the GPS coordinates of the centre point and the bounding distance is discussed. The GPS bound is later used to identify if a GPS point is located within a certain proximity of the centre point. A Javascript tool is developed to ease the process of specifying a bus route. A bus route is formed by combining a number of selected road segments. Google Maps Elevation API is used to obtain the road elevation along the bus route. An approximation algorithm based on a least square fit algorithm is proposed. The algorithm aims to minimise the number of slope changes while maintaining sufficient accuracy in the representation of elevations. After the bus route is specified, the relative position of each bus stop along the route is identified with a fitting algorithm. The fitting algorithm utilises the bearing from a bus stop to each GPS point on the route to locate the bus stop relative position. The methodology to identify the stop probability of a bus at a bus stop using historical bus data is described. The daily speed profiles of the buses passing by a bus stop are clustered to identify the significant temporal groups. The stop probability at the bus stop is then identified for each temporal group.

Chapter 5

Bus Speed Estimation

When the optimal profile is applied in a real-world setting, the main challenge for the bus to follow the optimal profile is the limitation of the traffic flow. The effect of the traffic flow can be reduced if the optimal control algorithm has the knowledge of the future traffic flow. Therefore, this chapter focuses on the estimation of on-road bus speed.

The estimation of bus speed is carried out using neural networks. Section 5.1 describes the available data that will be used later as the inputs for the neural network training. The data includes the time data, spatial data, weather data, and social data. Section 5.2 discusses the design and the training of the neural networks. A case study is presented in Section 5.3 to estimate the on-road speed of the bus on a segment of the road along the bus route from Oxford City Centre to Pear Tree Park and Ride. The on-road speed of the bus depends on the lane the bus travels on, namely the mixed traffic lane and the bus lane. The speed of the bus on a mixed traffic lane is estimated using the weather data, time data, and social data. As from real-world observations the speed of the bus on the bus lane is affected by the speed of its corresponding mixed traffic lane, the speed of the bus on a bus lane is estimated with the speed of the corresponding mixed traffic lane together with weather and

social data.

5.1 Data description

The data used in this chapter include the time data, spatial data, weather data, and social data. This section explains the information provided by each type of data, the data sources from which they are collected, and the methodology to clean the data.

5.1.1 Time data

Time data consists of the date and time of other collected data. This mainly corresponds to the daily routine such as the time to go to and leave work, sending children to and fetching them from school, etc. As this data is always part of the information of other data, no data cleaning is required.

5.1.2 Spatial data

Spatial data is the most important data in this chapter as this chapter focuses on the estimation of speed along a road. The spatial data involved is the GPS locations (latitude and longitude) of the data and the on-road speed or bus speed at the location.

The on-road speed of a road is obtained from the Oxforshire County Council transport website. The data is recorded with a computer script every 15 minutes. On the website, the roads are divided into individual road segments and each road segment is provided with a value for its average speed or average time of travel. The average time of travel can

be translated to the average speed on road by identifying the length of the road segment. Examination on the data reveals that at certain cases, the values provided by the source of the same road segment remain constant for a period of time, for example the average speed of a certain road segment is 4 m/s for 10 hours. As there is no access to the data acquisition, this situation is assumed to be an error. This data segment of constant value is thus removed from the dataset.

The speed of bus is obtained from Oxford Bus Company. The data consists of the GPS locations of the bus and its speed at the location. For the purpose of aligning the bus speed data with the average on-road speed data, the bus speed is grouped to the nearest 15 minutes of the day. For the same purpose, a GPS rectangular bound is defined for each road segment. The bus speed data located within the bound will be aligned to the speed data along that road segment. If multiple values exist on the same road segment in the same 15-minute group, the average of the values is taken as the value at that time group.

5.1.3 Weather data

Weather data provides the quantified information of the ambient conditions that affect the visibility of the driver and the conditions on the road surface. The data can be obtained from weather stations and it includes wind speed, precipitation, rainfall, humidity, temperature, pressure, and dew point.

This data is collected from the web page of OpenWeatherMap [110] every 15 minutes using a computer script and stored as Javascript Object Notation (JSON) file. As the data

source is reliable and the sampling frequency is controlled by the script, the data output does not required cleaning. In rare occasions where failure of the script or network connection occurs, the missing data points are interpolated.

5.1.4 Social data

Social data provides information of the social events that affect the mobility behaviours. It includes the bank holidays, weekends, university terms, and school holidays. These occasions change the daily routine of the users and hence affect the traffic on the road. This data is static data that does not require constant polling of another source, therefore they are stored as a static table and no data cleaning is required.

5.2 Artificial neural network

ANN and logistic regression can produce very similar result from the same problem. Logistic regression predicts the output by identifying the probability of each potential predictor. Different researchers [111–118] have carried out comparisons of performance between neural network and logistic regression. In general, neural network is able to perform approximation of any nonlinear mathematical function automatically without the prior knowledge of the relationship between variables [114]. This is useful as the relationship between variables can sometimes be complicated. As the interactions between variables can be identified through the training process, neural network can be easily scaled up.

Kalman filter [21, 119] is also often used in prediction. Kalman filter provides a recur-

sive solution to predict the state in a discrete-time process. A predictor is used to estimate the state of the current time step from the state in the previous time step. Then a corrector uses the measured value of the state to update the parameters in the predictor functions by identifying the error between the predicted and the measured values. Therefore the Kalman filter requires data of a discrete-time series. However, as the prediction focussed in this chapter is to use other states to estimate the state of interest and the continuity in time of the data is not certain, ANN serves as a better option.

ANN [60, 120, 121] is inspired by the understanding of human nervous system. The simplest neural network consists of a layer of input nodes and a layer of output nodes, with all input nodes connected to all output nodes with weighted links. The weight of each link is identified through training with datasets. Each node is also defined with an activation function that is normally nonlinear. Activation function computes the output of the node using the weighted sum of the inputs of the node. Layers of hidden neurons can be introduced between the input nodes and output nodes. Neural networks with hidden layers are also known as multilayer neural networks. Multilayer neural networks have proven to be able to solve more complicated problems than neural networks without hidden layer [122, 123].

This section discusses neural network from three different aspects. Section 5.2.1 first discusses the method to examine correlation between the input data with the output data to identify necessary input data. Sections 5.2.2 and 5.2.3 discuss the data conditioning of the input and output datasets. Section 5.2.4 explains the architecture of the neural network.

Section 5.2.5 describes the data division for training, testing and validation datasets. The training algorithm for the neural network is discussed in Section 5.2.6.

5.2.1 Data correlation

Before the data is used for the training of a neural network to perform prediction, the correlation of data for each input with the data for the output can first be studied. This process helps to identify if any of the input data has a significant correlation with the output data. If an input data is highly correlated to the output data, there will be no doubt that the data should be an input of the neural network.

Pearson correlation coefficient [124] is used to identify the correlation between two datasets. Pearson correlation coefficient, also known as Pearson r of two datasets X_1 and X_2 is calculated with

$$r = \frac{\sum(X_1 - \overline{X}_1)(X_2 - \overline{X}_2)}{\sqrt{\sum(X_1 - \overline{X}_1)^2}\sqrt{\sum(X_2 - \overline{X}_2)^2}} \tag{5.1}$$

The Pearson correlation coefficient, r lies between -1 to 1. The closer the absolute value of the coefficient is to one, the more accurate the relationship of the two datasets can be modelled as a linear function. If the Pearson correlation coefficient is close to zero, it implies that the relationship between the two datasets cannot be modelled with a linear function but it does not necessarily reject the possibility of the existence of relationship

between the two datasets.

If the Pearson correlation coefficient of the input data and the output data is close to one, the input data is a necessary input of the neural network. If it is close to zero, the relationship between the input data and the output data is undetermined. As the neural networks do not require the prior knowledge of the nature of the relationship, the input data will be included to identify its significance to predict the output data.

5.2.2 Data formatting

The available data needs to be formatted to the correct format to allow the significance of the data to show when it is used as the input of a neural network. Weather data is continuous data, therefore no additional formatting is required. Social data are categorical data. For example, a data entry can be said to be on a bank holiday, a weekend and not a school holiday. It is meaningless to say a data entry is between a bank holiday and a school holiday. Therefore the social data should not be grouped under an input with different numerical values to represent the events. Instead, the social data is formatted as boolean data. Each social event is used as one input. The value of the input is either 0 or 1, which corresponds to the state of false or true.

The time data is separated into two parts. First is the hour and the minute. As this can be considered to be continuous data, the hour and the minute of the data are converted to the minute of the day, i.e. in the range of 0 (00:00) to 1439 (23:59). The second part is the month and the day of the month. As these are more discrete than the time of the day,

they are considered as categorical data. Twelve inputs are used for month data, with each input corresponds to one month. For days of the month, thirty one inputs are introduced to represent each day in a month. The values of these inputs are 0 or 1 that corresponds to the state of false or true. The days of the week (Monday to Sunday) are considered to be categorical data using seven boolean inputs. This approach has been similarly used by Yasdi [59].

5.2.3 Data normalisation

The input and output datasets are normally of very different range. Therefore data normalisation is required to ensure that every element in input and output datasets has similar range. The data is normalised using a min-max technique shown in Equation 5.2 which scales every dataset to the range of zero (X'_{min}) and one (X'_{max}).

$$\text{Normalised input, } X'_i = \frac{X_i - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}(X'_{\text{max}} - X'_{\text{min}}) + X'_{\text{min}} \tag{5.2}$$

where X_i is actual input data,

$X_{\text{min}},\ X_{\text{max}}$ are minimum and maximum input data,

$X'_{\text{min}},\ X'_{\text{max}}$ are minimum and maximum target value

As the categorical data are formatted in boolean form, which is either 0 or 1, the normalisation is only applied on continuous data, i.e. the weather data, time of day and speed

data.

5.2.4 Network architecture

Feedforward and recurrent networks are two common architectures of neural network. Feedforward network passes the signal in one direction, i.e. from the input nodes to hidden nodes if available and then to the output nodes. A recurrent network incorporates internal feedbacks in the architecture [125, 126]. The signal fed back into the network is the previous values of the output nodes. Therefore a recurrent network is normally used for time series prediction. The feedback signal is the output value of the previous time step.

A recurrent network is reported to have higher accuracy than feedforward network for forecasting a time series even with noisy signals as the historical noise is fed back to the network and can be used to evaluate current noise [127]. However, this requires the datasets to be in continuous time steps. If the training data does not conform to this, i.e. missing data points exist in the time series, the feedback signals are not the desired signal and this introduces wrong training datasets to the network. As the data acquired for this work relies largely on the availability of third party sources, this issue is observed to happen in the datasets. Therefore a recurrent network is not ideal and a feedforward network is employed.

As the relationships between the predictor datasets and the speed data are not straightforward, multilayer feedforward network is used for the purpose of speed estimation. The network consists of one input layer, one hidden layer, and one output layer. The numbers

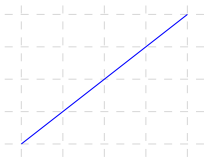
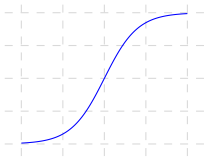
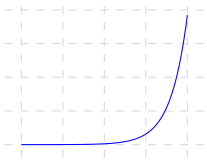
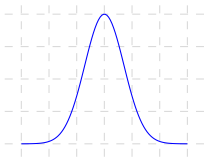
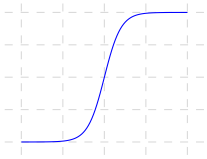
of neurons in input and output layers depend on the amount of elements in input and output datasets. Multiple approaches [128–132] have been suggested to determine the number of hidden neurons. The lack of hidden neurons leads to the failure to solve the problem whereas the redundancy of hidden neurons may cause the overfitting of the trained network. One common approach to solve the problem is to test different numbers of hidden neurons and identify the optimal number through network pruning or growing [129, 132, 133]. Network pruning starts with large number of hidden neurons and reduces the number while network growing starts with a small number of hidden neurons and increases the number. This requires high computational resources as repeated trainings with different number of hidden neurons need to be performed.

Another approach uses different formulae to calculate the number of hidden neurons based on the number of input and output neurons [131, 134–139]. Sheela and Deepa [131] compare the performance of existing approaches by evaluating the mean squared error between the actual and predicted results. This approach is less time consuming compare to the previous approach. Based on the result in the work by Sheela and Deepa [131], Equation 5.3 provides the number of hidden neurons, N_h of a network with n_I input elements that gives the prediction with lowest mean squared error.

$$N_h = \frac{4n_I^2 + 3}{n_I^2 - 8} \tag{5.3}$$

Each neuron has an activation function. The input of the neuron is passed to the activation function to obtain the output of the neuron. Table 5.1 lists some commonly used activation functions. Different activation functions for different layers are tested in the case study in Section 5.3.

Table 5.1: Commonly used activation functions

Name	Graph	Equation
Linear		$f(x) = x$
Sigmoid		$f(x) = \frac{1}{1+e^{-x}}$
Softmax		$f(x) = \frac{e^x}{\sum e^x}$
Gaussian		$f(x) = e^{-x^2}$
Tanh		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$

Bias can be introduced to the network to provide more flexibility to the network. A bias is basically a node of value 1.0 that is connected to non-input neurons. To show the effect of a bias, the results of two simple neural networks shown in Figure 5.1 and 5.2 are

compared. Figure 5.1 shows the simplest network with no bias and Figure 5.2 shows the same network but with a bias. The activation function used in the example is a sigmoid function.

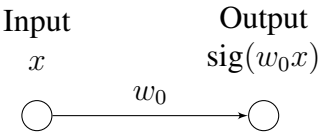


Figure 5.1: Basic network without bias

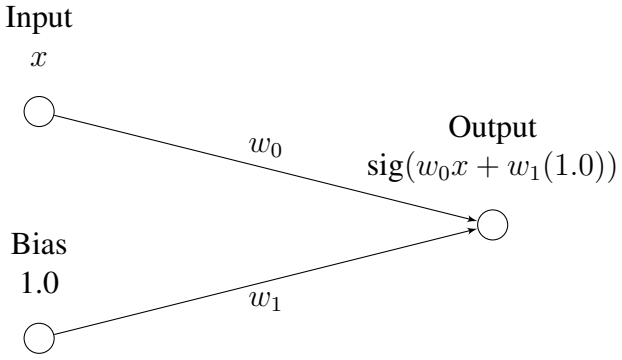


Figure 5.2: Basic network with bias

Figure 5.3 shows the values of the output neuron in Figure 5.1 with different weights and Figure 5.4 shows that of Figure 5.2. The comparison of Figures 5.3 and 5.4 shows that the inclusion of bias increases the ability of the network to learn a more complex relation. Therefore bias is implemented in the prediction network in Section 5.3.

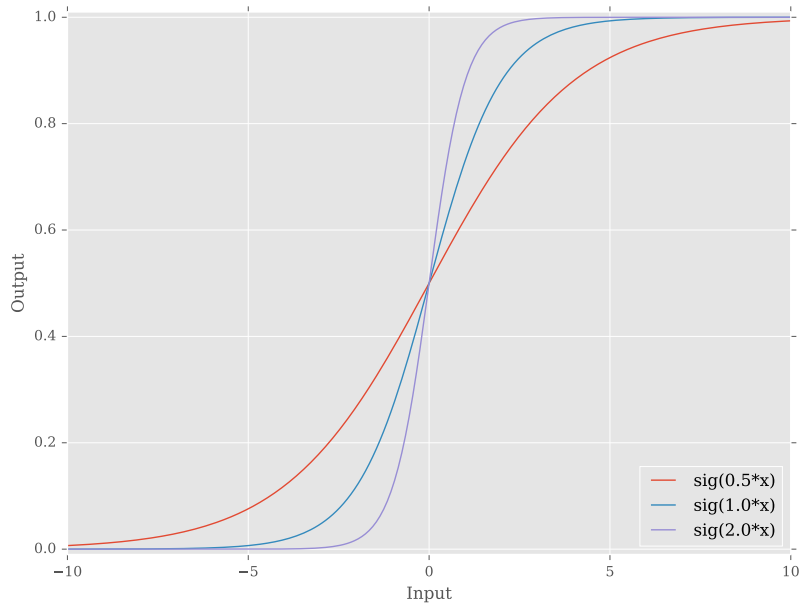


Figure 5.3: Output of basic network without bias at various weights

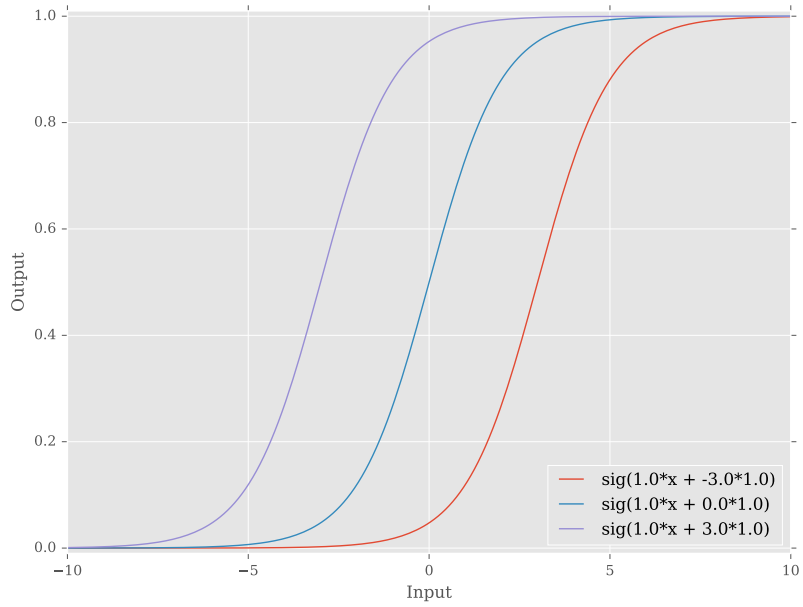


Figure 5.4: Output of basic network with bias at various weights

5.2.5 Data division for training, testing and validation

In the learning process of neural network, the available datasets are used for three purposes: training, testing and validation. Training data, as suggested by the name, are used for the actual training of the neural network. Testing data prevent the overfitting of the neural network. Validation data are used to evaluate the performance of the resultant neural network. Different researchers have suggested different methods to determine the proportions to which the data is divided for each purpose. Haykin [133] suggests 80/20 split of train+test/validation datasets based on [140]. Crowther and Cox [141] mention the traditional split of 33/33/33 for train/test/validation and also suggest the usage of one third of the data for validation with repeated testing between 50/50 to 70/30 for train/test split. As the importance of data division depends on the amount of data available, Crowther and Cox's method is implemented. The method is summarised as follows:

(i) Remove one third of the data for validation.

(ii) Run a series of tests that split the remaining data between train and test in the ratio 50/50 through to 70/30 in step of 1.

(iii) Choose the ratio with the highest accuracy and use it to train the neural network.

5.2.6 Network training

Backpropagation training algorithm [142] is used as the training algorithm for the neural network in this work. The backpropagation algorithm first feeds an input into the network.

Then the errors, i.e. the difference between target and actual value of every output and hidden neuron are calculated in backward direction from the output to input neurons [143, 144].

In neural network, the concept of epoch is used in the training of the network. One training epoch refers to one complete pass of all the training data. Therefore, one epoch normally has multiple iterations. After every epoch, the errors of training and testing data are computed. The training is terminated when the error converges. Then the performance of the resultant neural network is evaluated with validation data.

The backpropagation training algorithm can be summarised in the following list:

		Data used
i.	Forward pass Pass the data through the neural network from input to output.	*training data*
ii.	Backward pass Calculate the difference between the target and actual value of output and hidden neurons.	*training data*
iii.	Update weights of neurons	*training data*
iv.	Evaluate errors	*testing data*
v.	Repeat i to iv until the errors calculated in iv converge	–
vi.	Evaluate the network performance with validation data	*validation data*

5.3 Case study

Case study is carried out on the road segment and the weather station shown on Figure 5.5. The time range of the available data is between 17th of February 2015 to 17th of February 2016. Two neural networks are trained. Section 5.3.1 describes the training of the neural network to estimate the average speed on road whereas Section 5.3.2 the speed on the bus

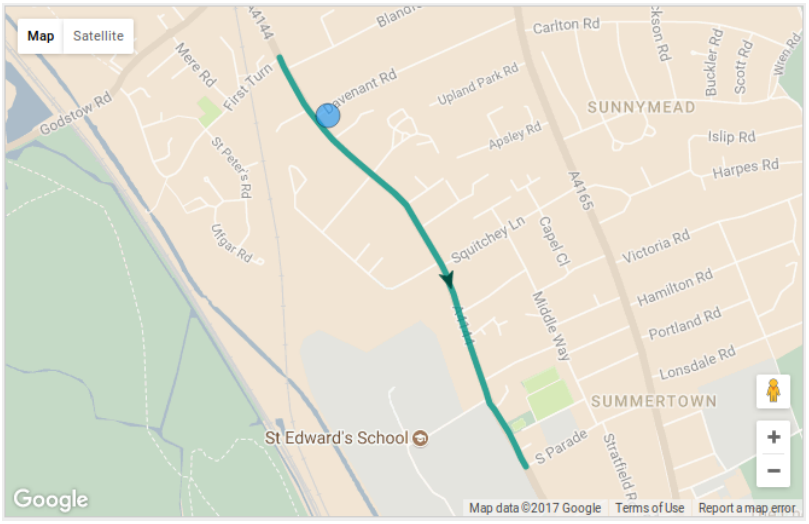


Figure 5.5: Road segment (First Turn to South Parade) and weather station (IOXFORD66) used in case study. Weather station is denoted by the blue dot.

lane. The neural network is built using Pybrain Python Library [145].

5.3.1 Average speed on road estimation

Data conditioning

The speed data of the road segment and the weather data from the weather station are first extracted separately. The data are then aligned using the timestamp of both datasets. The merging of the two datasets are done such that only timestamps available in both datasets are preserved. Timestamps that only exist in one of the two datasets are removed from the new dataset. The new aligned dataset is visualised in Figures 5.6 to 5.13. The precipitation in Figure 5.12 gives zero value for about 90% of the data, and the wind speed in Figure 5.13 does not provide valid value. Therefore the data of the precipitation and wind speed data is removed from the dataset.

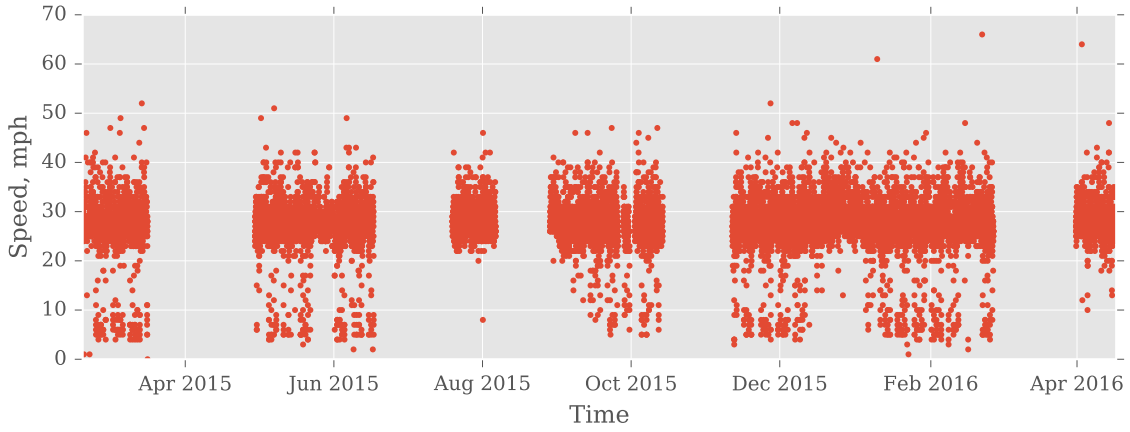


Figure 5.6: Speed data for road segment First Turn to South Parade

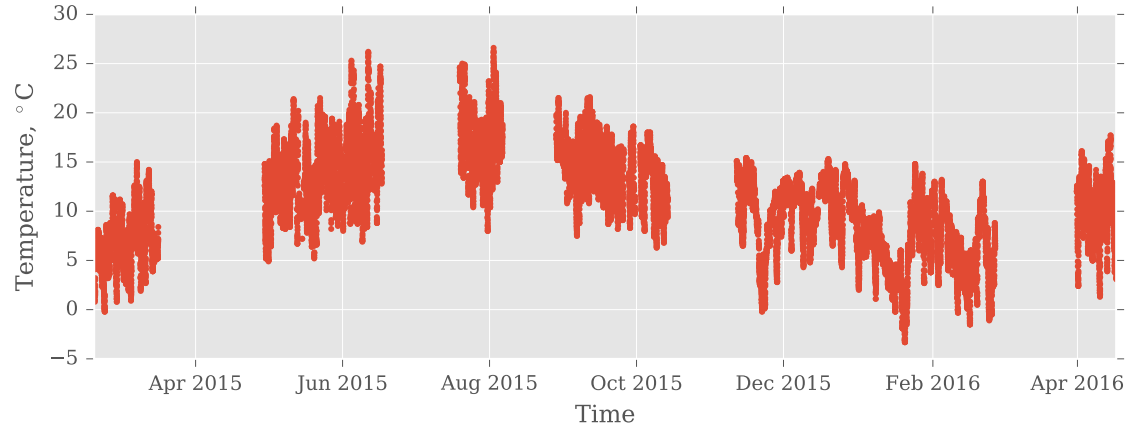


Figure 5.7: Weather data (temperature) for weather station IOXFORD66

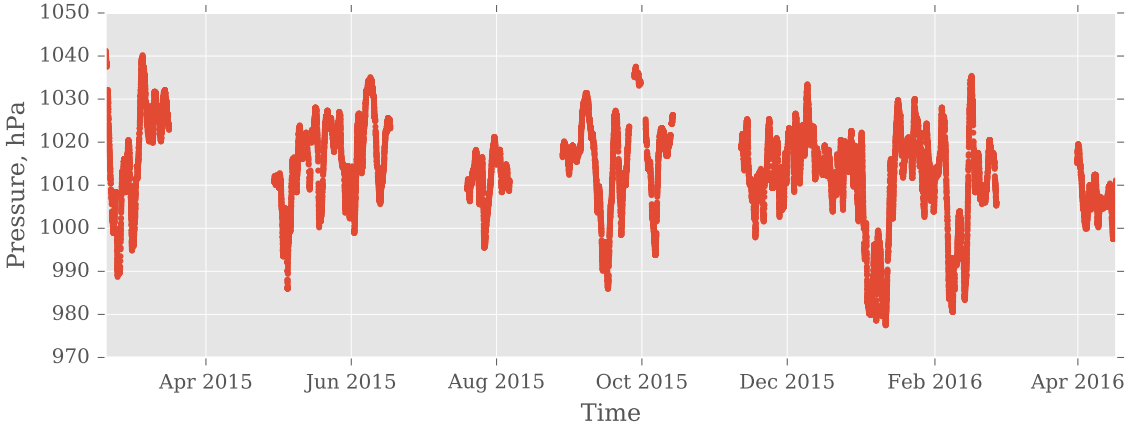


Figure 5.8: Weather data (pressure) for weather station IOXFORD66

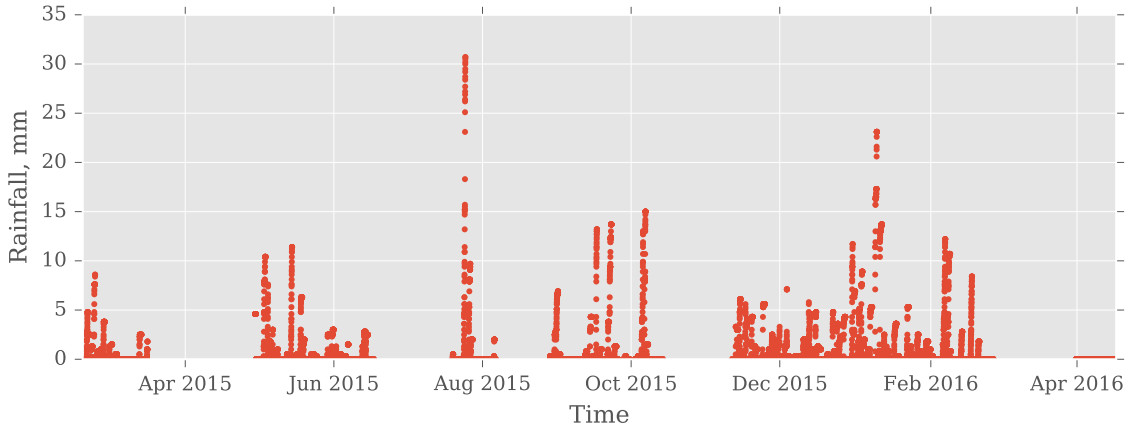


Figure 5.9: Weather data (rainfall) for weather station IOXFORD66

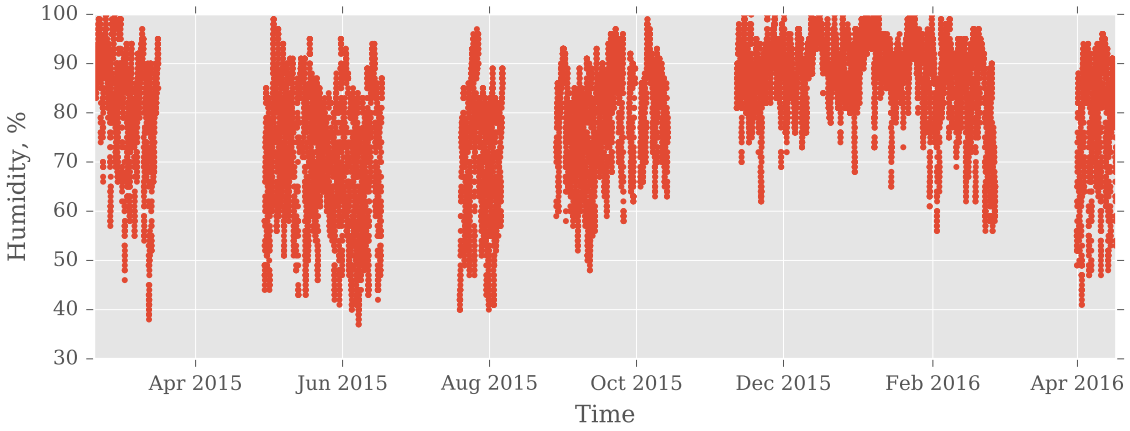


Figure 5.10: Weather data (humidity) for weather station IOXFORD66

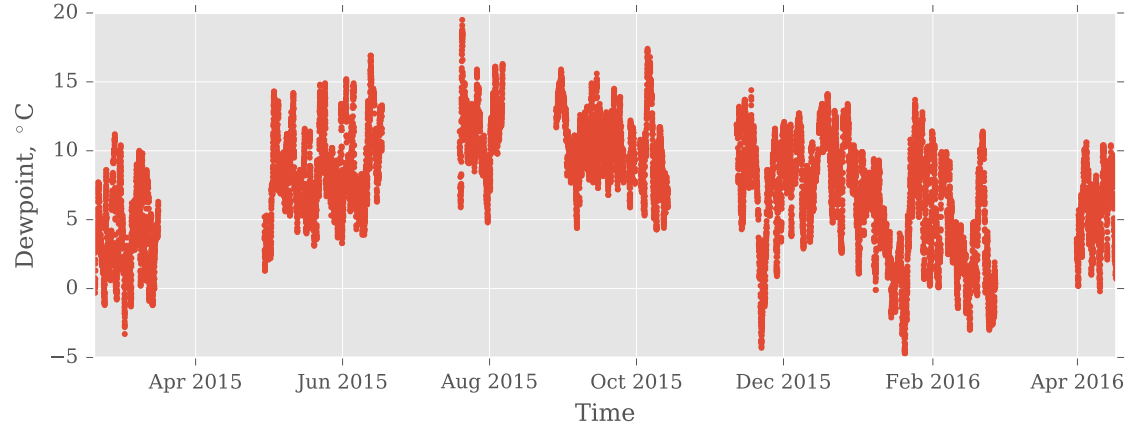


Figure 5.11: Weather data (dew point) for weather station IOXFORD66

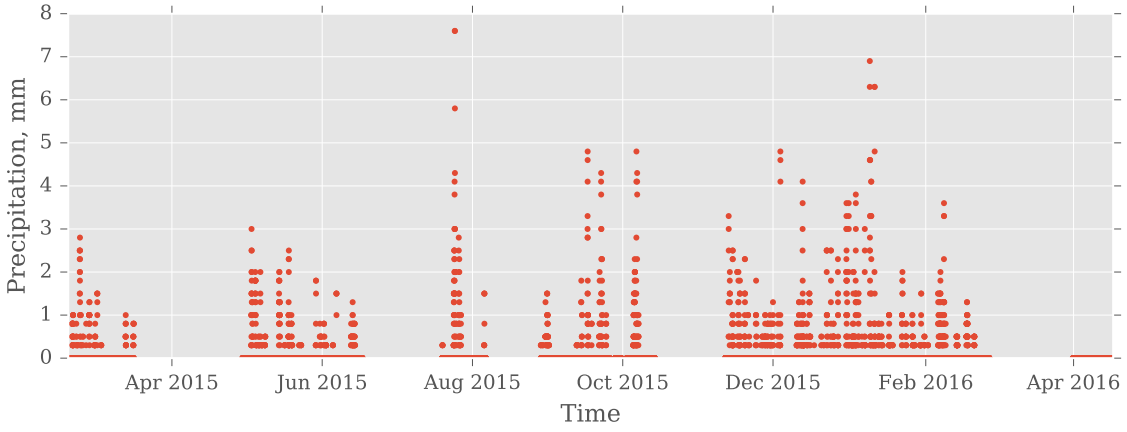


Figure 5.12: Weather data (precipitation) for weather station IOXFORD66

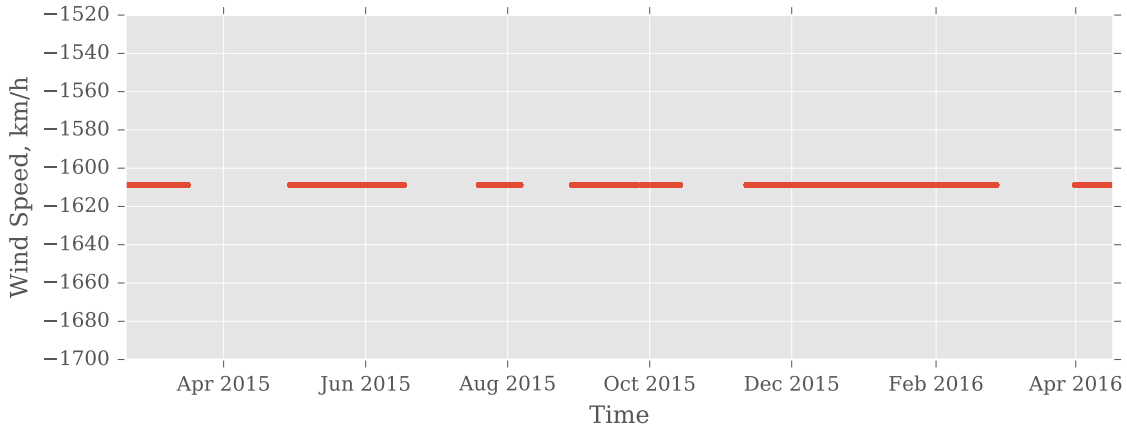


Figure 5.13: Weather data (wind speed) for weather station IOXFORD66

The speed data for the road segment is plotted against different types of weather data in Figure 5.14 and their corresponding Pearson correlation coefficients are shown in the graphs. The Pearson correlation coefficients show that there is no significant correlation between the weather data and the speed data. This implies that there is no clear linear relationship between the weather data and the speed data.

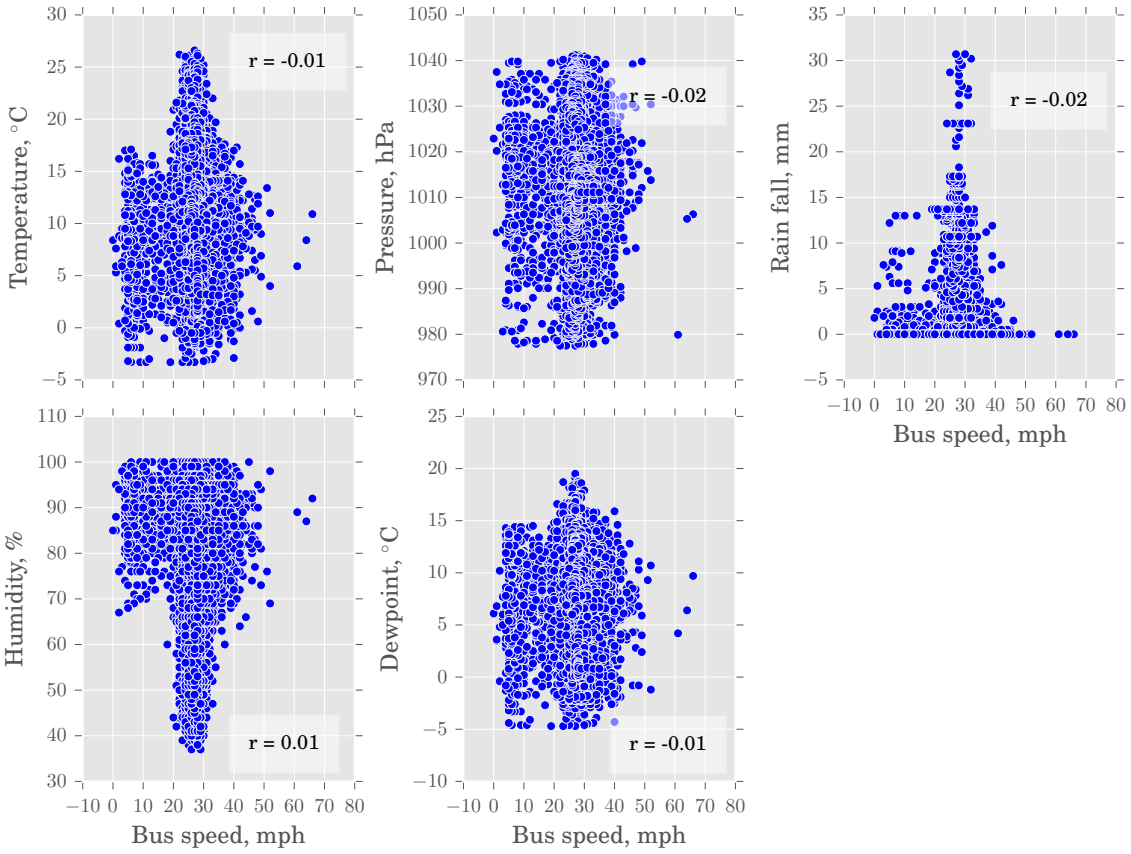


Figure 5.14: Speed data for road segment against weather data

As mentioned in Section 5.2.2, social events and part of time data (month and day) are formatted as categorical data, and the time of day as continuous data. The combined dataset has the columns of timestamp, speed on road, temperature, pressure, rainfall, humidity, dew point, bank holiday, weekend, Monday to Sunday, university term, school day, time

of day, month01 to month12 (for January to December), and day01 to day31.

The columns of speed on road, temperature, pressure, rainfall, humidity, dew point and time of day are normalised to the range of zero and one using Equation 5.2. Table 5.2 shows the X'_{min} and X'_{max} of each set of data.

Table 5.2: Minimum and maximum values to which the data are normalised

Column name	X'_{min}	X'_{max}	**Column name**	X'_{min}	X'_{max}
Dew point (°C)	-4.7	19.5	Pressure (hPa)	977.5	1041.2
Temperature (°C)	-3.3	26.6	Humidity (%)	37.0	100.0
Rainfall (mm)	0.0	30.7	Time of day (min)	0.0	1425.0
Speed on road (mph)	0.0	66.0			

Specific network architecture

A three layers feedforward neural network is designed with one input layer, one hidden layer, and one output layer. A bias is introduced to the network and connected to all non-input neurons.

Repeated testing is set up to identify the optimal parameters. These parameters include the activation function of each layer, the datasets used as the input data, the proportion split of training/testing/validation, and the number of hidden neurons.

First the neural network is built with all columns in the combined dataset except the speed on road as the inputs, training/testing/validation split of (70/30)/30, and number of hidden neurons two times the number of input elements. Different combinations of activation functions for each layer are tested and their validation errors are evaluated to identify the optimal activation functions combination. The list of activation functions in

Table 5.1 are used to form the combinations.

Then, with the optimal activation functions combinations, three different sets of input data are tested. The three different sets of input data are (i) the weather data, (ii) the social data and time data, and (iii) the combination of (i) and (ii).

The network training is then repeated with the training/testing split of 70/30 to 50/50 of non-validation dataset. The validation data proportion is always set to be one third of the complete dataset.

Four criteria for hidden neuron number calculation are tested. The four criteria are $2n_I$, $\frac{4n_I^2+3}{n_I^2-8}$, n_I, and $\frac{n_I}{2}$ with n_I as the number of input elements. As $\frac{4n_I^2+3}{n_I^2-8}$ and $\frac{n_I}{2}$ may give value of less than one, a minimum number of one is also defined for the number of hidden neurons.

Results

Tables 5.3 to 5.5 present the results of the testing of parameters. The performance is evaluated using the validation error of the trained network.

Table 5.3: Comparisons of different sets of input data

Input data sets	**Validation error**
All data	0.00216
Weather data	0.00252
Social and time data	0.00220

The optimal parameters are summarised in Table 5.6. The comparison between the target and predicted output of the trained network for the validation data is visualised in Figure 5.15.

Table 5.4: Comparisons of different training/testing data split

Train/test data split	Validation error	Train/test data split	Validation error
0.70/0.30	0.00219	0.59/0.41	0.00231
0.69/0.31	0.00222	0.58/0.42	0.00204
0.68/0.32	0.00217	0.57/0.43	0.00217
0.67/0.33	0.00220	0.56/0.44	0.00230
0.66/0.34	0.00222	0.55/0.45	0.00234
0.65/0.35	0.00226	0.54/0.46	0.00232
0.64/0.36	0.00227	0.53/0.47	0.00215
0.63/0.37	0.00239	0.52/0.48	0.00225
0.62/0.38	0.00221	0.51/0.49	0.00218
0.61/0.39	0.00229	0.50/0.50	0.00227
0.60/0.40	0.00226		

Table 5.5: Comparisons of criteria for hidden neuron number calculation

Criteria	Number of hidden neurons	Validation error
$2n_I$	120	0.00225
$\frac{4n_I^2+3}{n_I^2-8}$	4	0.00226
n_I	60	0.00234
$\frac{n_I}{2}$	30	0.00244

Table 5.6: Optimal parameters for average speed on road estimation neural network

Parameters	Value
Activation functions	[Sigmoid, Softmax, Linear]
Input data	All data (Weather data, social data and time data)
Train/test split	0.58/0.42
Hidden neuron number	120 ($2n_I$)

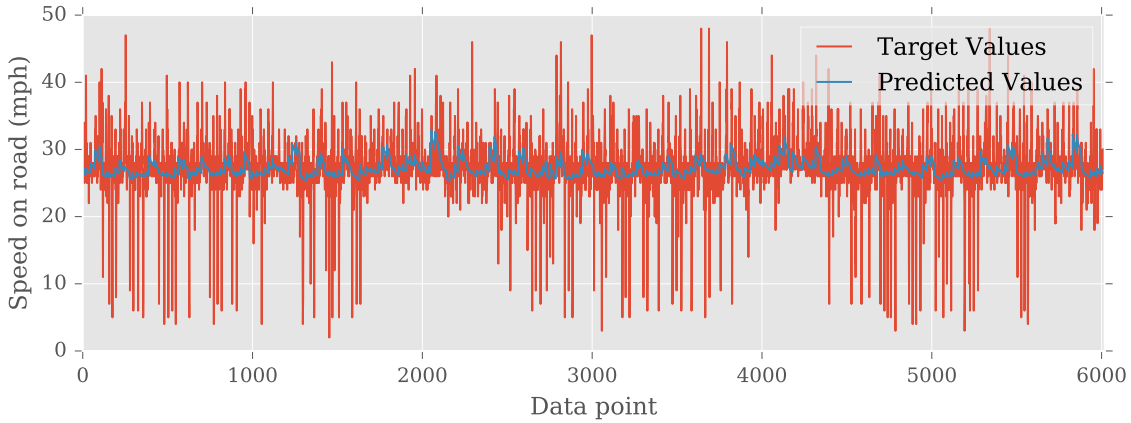


Figure 5.15: Comparison of target and predicted output for validation data

Figure 5.16 shows an extracted segment of Figure 5.15. It shows that the profile of the predicted speed follows the general trend of the target speed. The neural network is able to learn the basic relationship between the input data and the target data up to a certain degree. Future work is needed to improve accuracy of the prediction.

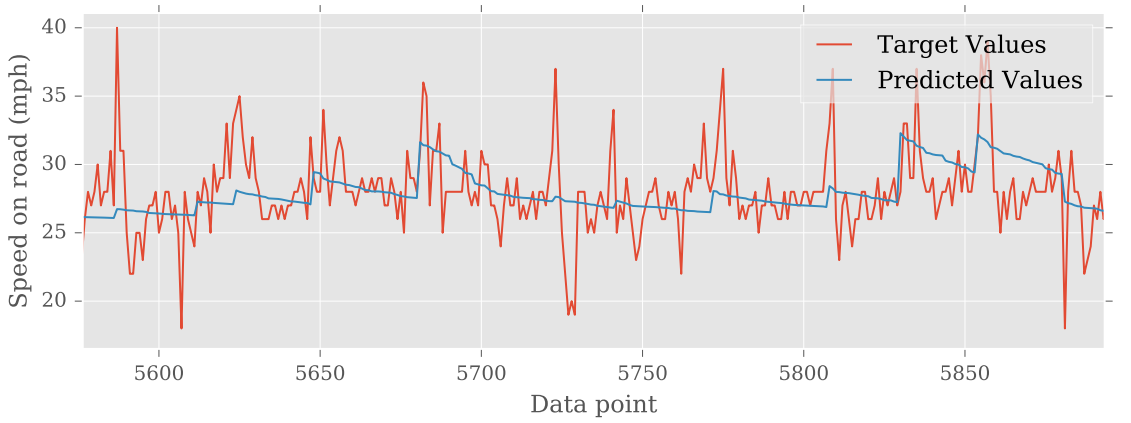


Figure 5.16: Comparison of target and predicted output for validation data (extracted)

5.3.2 Bus lane speed estimation

Data conditioning

As the bus lane is only used by buses, the speed on the bus lane is the bus speed along the road segment with bus lane. The data of the bus speed is obtained from Oxford Bus Company. First, GPS bounds for the road segment need to be identified to extract bus speed data corresponds to the road segment. GPS bounds instead of a series of GPS locations of road are used because the errors in the GPS location data of the bus may cause the bus to appear at the side of the road and not exactly on the road. Since one single bound for the whole road segment causes the bound to include adjacent roads that are not part of the road segment, the road segment is divided into a series of rectangular GPS bounds as shown in Figure 5.17. Connecting GPS bounds are overlapped to ensure the inclusion of the data points near the connecting zones.

As the bus speed data are not recorded in a fixed time interval that is aligned with the speed on road data, the bus speed data are resampled to the nearest 15 minutes. Bus speeds on the duplicated time are aggregated by taking the average of the data.

Oxford Bus Company has provided the data of three buses. As these buses may be used for different bus routes on different days, bus data on a certain bus route may not be available for a number of weeks. After the alignment of the bus speed data to the speed data of the road segment, there are only 223 data points available. This set of data is plotted in Figure 5.18. When the bus speed data is aligned with the speed data together with the

Figure 5.17: GPS bounds for road segment First Turn to South Parade

weather data, there are 147 data points available. Bus speed data is plotted against the speed data and different weather data in Figure 5.19.

The Pearson correlation coefficients between bus speed data and different data are calculated and shown in their respective graph. Judging by the values of correlation coefficients lower than 0.10, the correlation between bus speed data and weather data is very weak. With the correlation coefficients of 0.35 in Figure 5.18 and 0.24 in Figure 5.19, the correlation between bus speed data and the speed on road data is also weak.

However, the correlation coefficient only captures the relationship between two variables. There might be more complicated relationships between the bus speed data and the combined data of the speed on road data and weather data. As neural network has the potential to learn the more complicated relationships between multiple data sets, neural networks with different input data sets are developed to identify if the weather data can

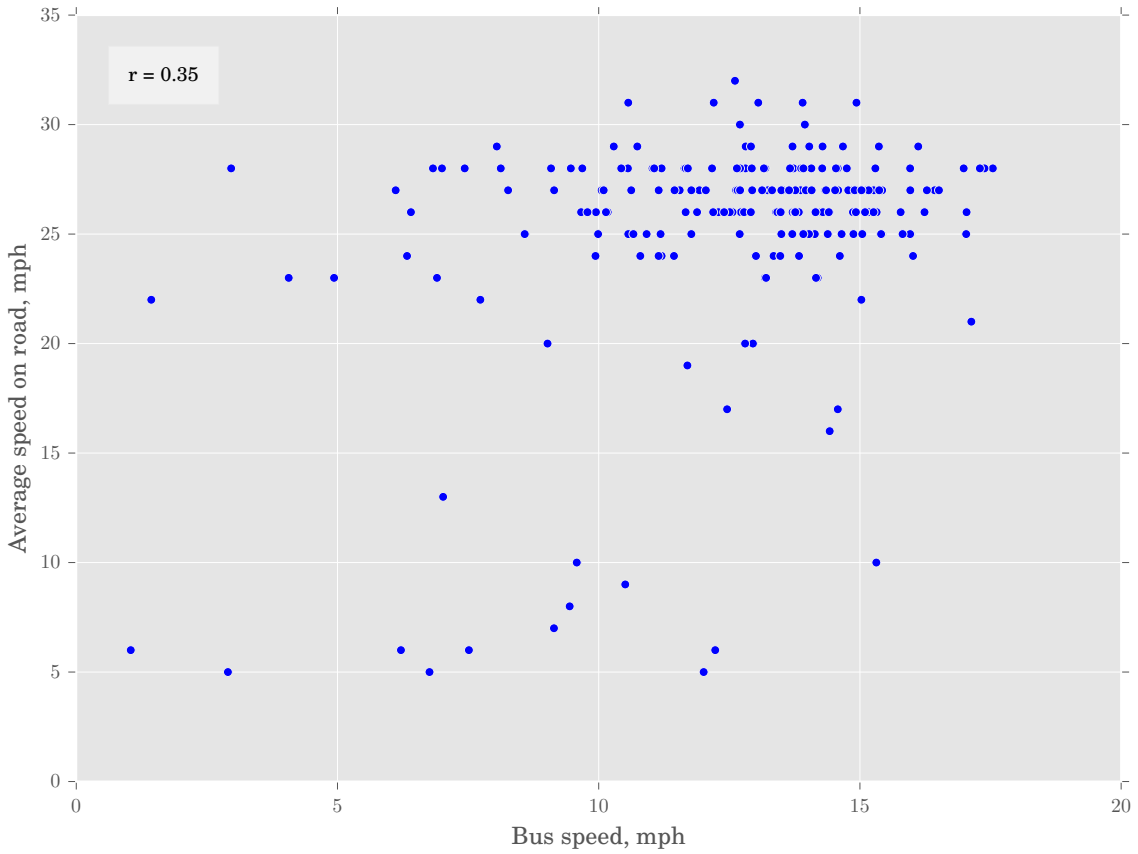


Figure 5.18: Bus speed data plotted against the road segment speed data

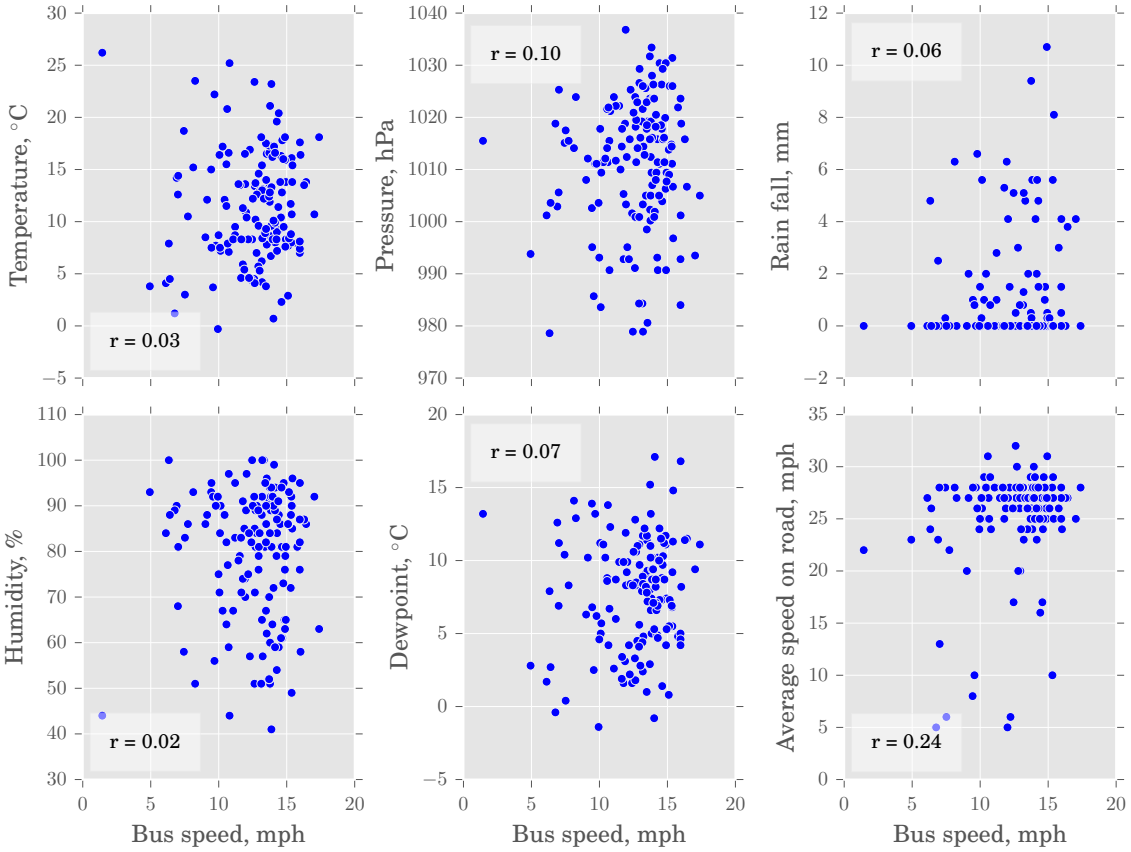


Figure 5.19: Bus speed data plotted against the weather data and road segment speed data

improve the accuracy of the prediction despite the low correlation coefficients.

As there are only 223 data points in Figure 5.18 and 147 in Figure 5.19, the trained neural network may has the potential of being undertrained due to the lack of training data.

The normalisation process for the two sets of data in Figures 5.18 and 5.19 is carried out separately. For the aligned data set containing only the bus speed data and the speed on road data, X'_{min} and X'_{max} for the data are shown in Table 5.7. Table 5.8 shows the values of X'_{min} and X'_{max} for different data in the data set containing the bus speed data, the speed on road data, and the weather data.

Table 5.7: Minimum and maximum values for normalisation of data set containing only the bus speed data and speed on road data

Column name	X'_{min}	X'_{max}	**Column name**	X'_{min}	X'_{max}
Bus speed (mph)	1.04	17.5	Speed on road (mph)	5.0	32.0

Table 5.8: Minimum and maximum values for normalisation of data set containing the bus speed data, speed on road data, and weather data

Column name	X'_{min}	X'_{max}	**Column name**	X'_{min}	X'_{max}
Dew point (°C)	-1.4	17.1	Pressure (hPa)	978.6	1036.8
Temperature (°C)	-0.3	26.2	Humidity (%)	41.0	100.0
Rainfall (mm)	0.0	10.7	Speed on road (mph)	5.0	32.0
Bus speed (mph)	1.44	17.38			

Specific network architecture

Similar testings in Section 5.3.1 are performed on the neural network designed for bus lane speed estimation. Same combinations of activation functions, training/testing split and criteria for hidden neuron number are tested. The sets of input data tested are (i) only speed on road, (ii) speed on road with weather data, (iii) speed on road with social data,

and (iv) speed on road with weather data and social data. The output dataset is the bus speed data.

Results

Table 5.9 to 5.11 present the results of the different parameter testing with the validation error as the performance comparison index.

Table 5.9: Comparisons of different sets of input data

Input data sets	**Validation error**
Speed on road only	0.01492
Speed on road with weather and social data	0.01777
Speed on road with weather data	0.01628
Speed on road with social data	0.02080

Table 5.10: Comparisons of different training/testing data split

Train/test data split	**Validation error**	**Train/test data split**	**Validation error**
0.70/0.30	0.01582	0.59/0.41	0.02111
0.69/0.31	0.02261	0.58/0.42	0.01063
0.68/0.32	0.01590	0.57/0.43	0.01714
0.67/0.33	0.01473	0.56/0.44	0.01351
0.66/0.34	0.01482	0.55/0.45	0.01707
0.65/0.35	0.01879	0.54/0.46	0.01300
0.64/0.36	0.02354	0.53/0.47	0.01163
0.63/0.37	0.01442	0.52/0.48	0.01890
0.62/0.38	0.02121	0.51/0.49	0.01271
0.61/0.39	0.01573	0.50/0.50	0.01823
0.60/0.40	0.01240		

The optimal parameters are summarised in Table 5.12. The target and predicted output of the neural network trained with optimal parameters are plotted in Figure 5.20.

Figure 5.20 shows that the optimal neural network does not provide a reasonable predicted result. Therefore the comparisons of the target and predicted values for validation

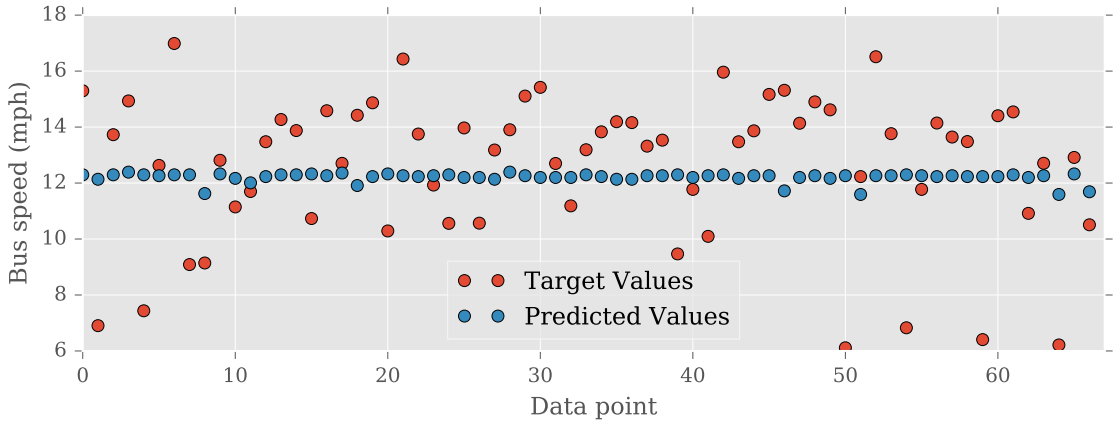


Figure 5.20: Comparison of target and predicted output for validation data

data with different input data sets are shown in Figures 5.21 to 5.24. The validation errors for the data in these figures are shown in Table 5.9. From visual observations the neural networks trained with input data of speed, weather and social data (Figure 5.22), and input data of speed and weather data (Figure 5.23) seem to have a better prediction result than the network trained with only speed data as input data despite the latter has the lowest validation error. Hence, the scatter plot of the target values against predicted values of the four input data sets are shown in Figure 5.25. The line for perfect fit ($x = y$) is plotted on each graph.

From Figure 5.25, a high density area can be observed in Figure 5.25(i) near the target values of 12 to 14. This area is close to the perfect-fit line but the rest of the points lies

Table 5.11: Comparisons of criteria for hidden neuron number calculation

Criteria	**Number of hidden neurons**	**Validation error**
$2n_I$	2	0.01490
$\frac{4n_I^2+3}{n_I^2-8}$	$\max(1,-1)=1$	0.01230
n_I	1	0.01947
$\frac{n_I}{2}$	$\max(1,0.5)=1$	0.01546

Table 5.12: Optimal parameters for average speed on road estimation neural network

Parameters	Value
Activation functions	[Linear, Linear, Linear]
Input data	Speed on road only
Train/test split	0.58/0.42
Hidden neuron number	1

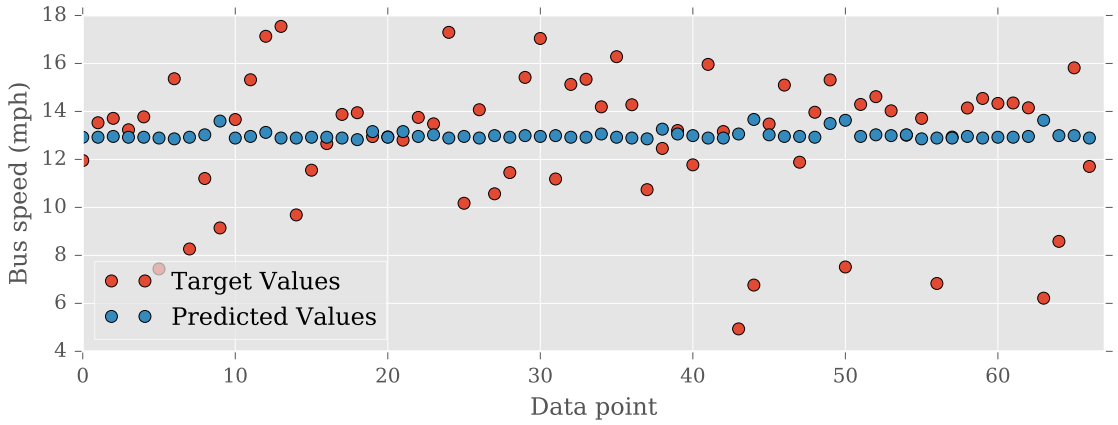


Figure 5.21: Comparison of target and predicted output for validation data with input data of speed data only

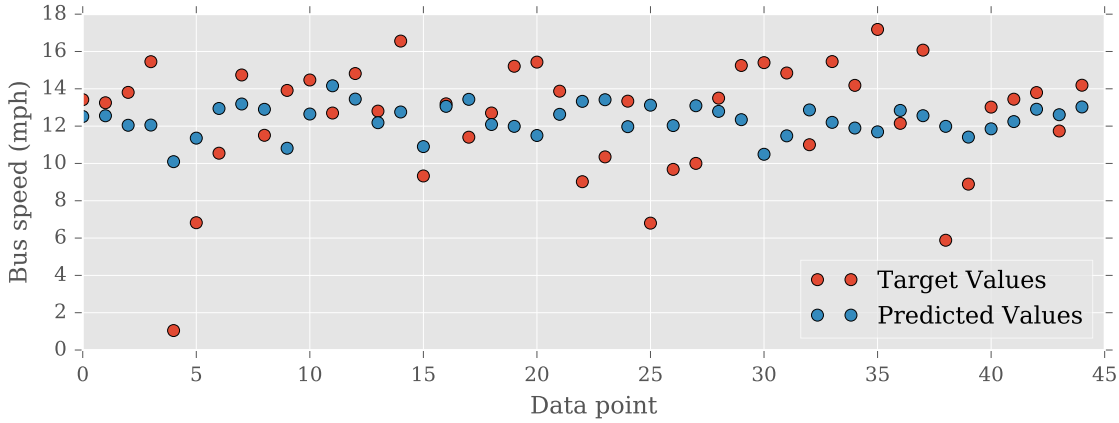


Figure 5.22: Comparison of target and predicted output for validation data with input data of speed data, weather data and social data

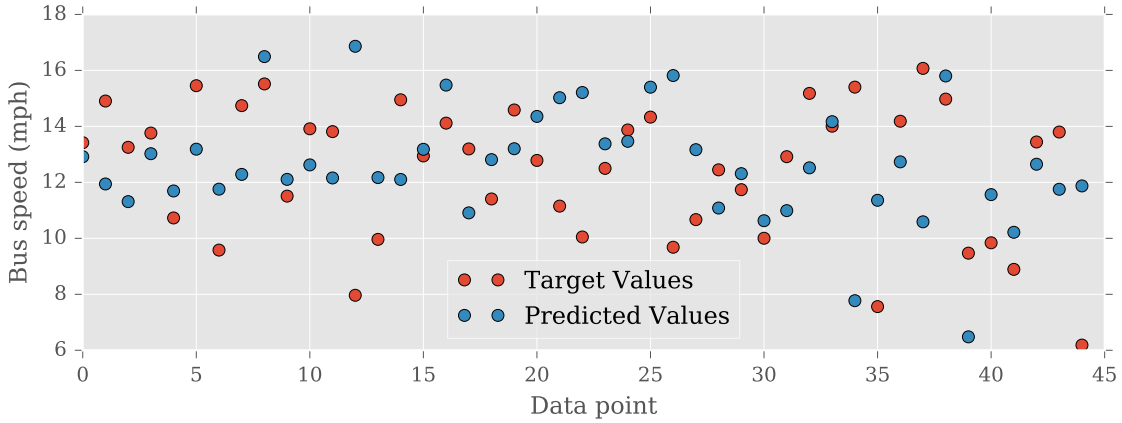


Figure 5.23: Comparison of target and predicted output for validation data with input data of speed data and weather data

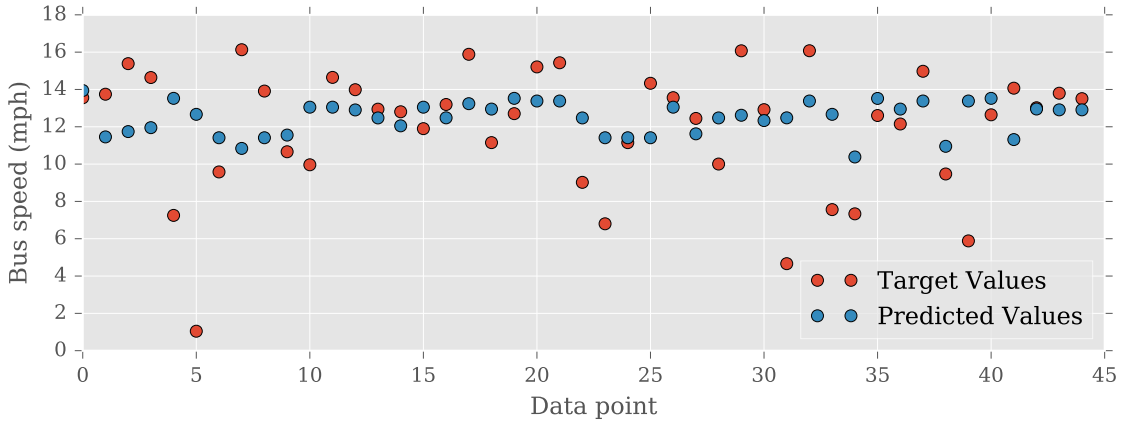


Figure 5.24: Comparison of target and predicted output for validation data with input data of speed data and social data

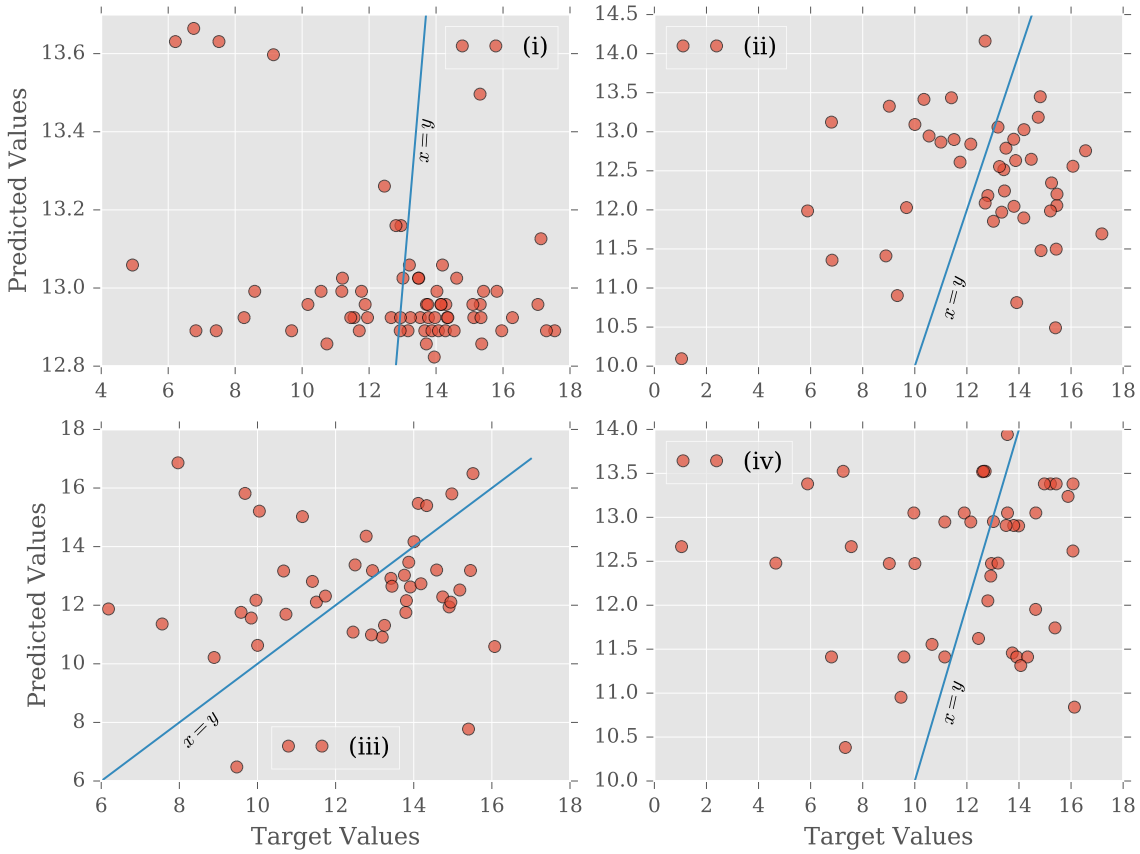


Figure 5.25: Scatter plots of target values against predicted values of neural network trained with input data of (i) speed data only, (ii) speed data, weather data, and social data, (iii) speed data and weather data, and (iv) speed data and social data

far from the line. This provides the low value of the validation error. For Figure 5.25(ii) and Figure 5.25(iii), the points that lie close to the line spread over a wider range. This situation contributes to visually better predictions as observed in Figures 5.22 and 5.23.

However, the target and predicted values do not agree in any of these figures. Two hypotheses are suggested to explain this issue. First is the lacking of training data. As previously mentioned, there are only 147 or 223 data points in total. The low amount of data points might cause the neural network to be under trained. Second is that the bus lane speed is actually independent of the mixed traffic lane speed, which corresponds to assumption of literatures. Future work is needed to confirm the hypotheses.

5.4 Summary

The estimation of bus speed is performed through the estimation of bus speed on the mixed traffic lane and the estimation of bus speed in the bus lane. Two neural networks are designed to perform the two estimations respectively. The first neural network is used to estimate the speed on the mixed traffic lane using input data of the weather data, the social events and the time data. The second neural network estimates the speed on the bus lane using the speed on mixed traffic lane, the weather data and/or the social events. The data is first formatted according to the data type. Continuous data is not changed at this stage whereas categorical data is formatted into different classes where each class holds a boolean value. The continuous data is then normalised to the range of zero to one. As categorical data is now in boolean form, no normalisation is required. Both neural networks

are designed to have three layers: input, hidden, and output layers. Bias is connected to each node in hidden and output layers for higher flexibility of the networks.

Neural networks with different parameters are trained and evaluated to identify the optimal parameters for both neural networks. The parameters to be evaluated include the input dataset, the activation function of each layer, the number of hidden neuron, and the proportion split of data for training and testing purposes. The different input datasets tested for traffic speed estimation are (i) weather data, (ii) social and temporal data, and (iii) the combined data of (i) and (ii). For bus lane speed estimation, the input data sets examined are (i) on-road speed, (ii) on-road speed with weather data, (iii) on-road speed with social data, and (iv) combined data of (i), (ii), and (iii). The activation functions that are tested for every layer are linear, sigmoid, softmax, gaussian, and tanh functions. The different criteria to identify the number of hidden neuron are $2n_I$, $\frac{4n_I^2+3}{n_I^2-8}$, n_I, $\frac{n_I}{2}$ where n_I is the number of input neurons. Minimum number of one is also defined for the number of hidden neurons. The data split for training, testing and validation is designed such that thirty percent of the data is used for validation. The proportion split for training and testing data are evaluated between 0.5/0.5 to 0.7/0.3.

The optimal neural network for on-road speed estimation on the mixed traffic lane has the activation functions of sigmoid, softmax, and linear for input, hidden, and output layers, input dataset of the weather, social, and time data, the training, testing, and validation data split of 0.41, 0.29, and 0.30, and the hidden neuron number of 60. The estimated values of the neural network show that the neural network is able to estimate the trend of the target

values but more work is needed for higher accuracy.

The result of neural network training for the estimation of speed on the bus lane has not provided reasonably accurate predictions. Two hypotheses are suggested: (i) the neural network is undertrained due to the lack of data and (ii) the speed on the bus lane is independent of the speed on mixed traffic lane as suggested by literature.

Chapter 6

Realisation of Optimal Bus Driving with Low-Cost System

This chapter focuses on the development of a low-cost platform for real-time deployment of optimal profile on a bus. Real-time information of the bus is required to provide optimal speed in real time. This real-time information includes the location of the bus, the position of the bus relative to the route, the speed of the bus, and the direction to which the bus heads. Therefore we have designed a system to acquire the real-time information of the bus and provide the optimal speed in real time.

The design of the system is divided into two parts, the hardware and the software. The hardware is responsible for the acquisition of the data and the delivery of current optimal speed to the bus. Section 6.1 describes the modules used in the hardware for computation, data acquisition, and communication.

The software is responsible to process the acquired data in order to compute the optimal speed profile according to the location and speed of the bus. The software processes the data acquired through the hardware to identify the real-time information of the bus. The real-time information is then used to adjust the optimal profile if the states of bus are not at

optimal. Section 6.2 explains the processing of GPS data, the identification of the position of the bus relative to the route, and the adjustment of optimal profile.

The adjustment of optimal profile is required when the states of the bus do not match the optimal states of the bus. Three scenarios in which this may happen are discussed in Section 6.3. First scenario is when the bus is required to stop at an on-demand bus stop due to passengers alighting or boarding the bus. The second scenario is when the bus has a delayed start from rest. The bus can be starting from a bus stop or the beginning of the route. The third scenario is when the speed of the bus is slowed by the traffic.

6.1 Hardware setup

The purposes of the hardware of the system are to acquire the position and motion data and deliver the optimal speed to the vehicle. The hardware can be divided into three units, the computational unit, the data acquisition unit, and the communication unit. The computational unit used in the system is a Raspberry Pi. Details for the setup of the Raspberry Pi is discussed in Section 6.1.1. The data acquisition unit is a GPS module. The specifications of the GPS module is described in Section 6.1.2. The communication unit is used to transfer information to the vehicle. A controlled area network (CAN) bus module compatible with Raspberry Pi is used. The details of the CAN bus module are presented in Section 6.1.3. Figure 6.1 shows the block diagram for the overall hardware setup and Figure 6.2 shows the photo of the hardware setup.

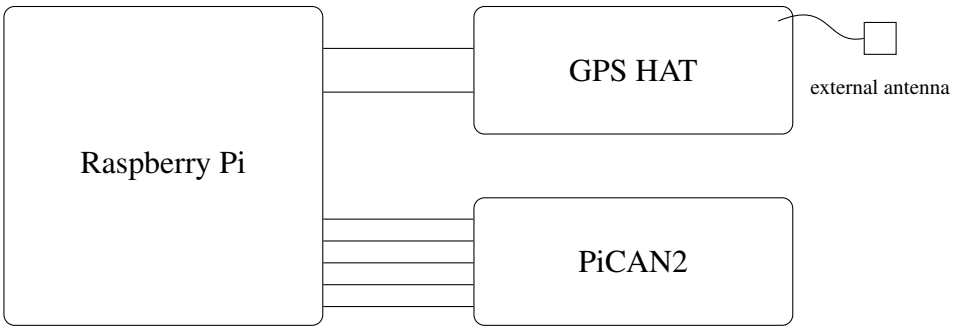


Figure 6.1: Block diagram for hardware setup

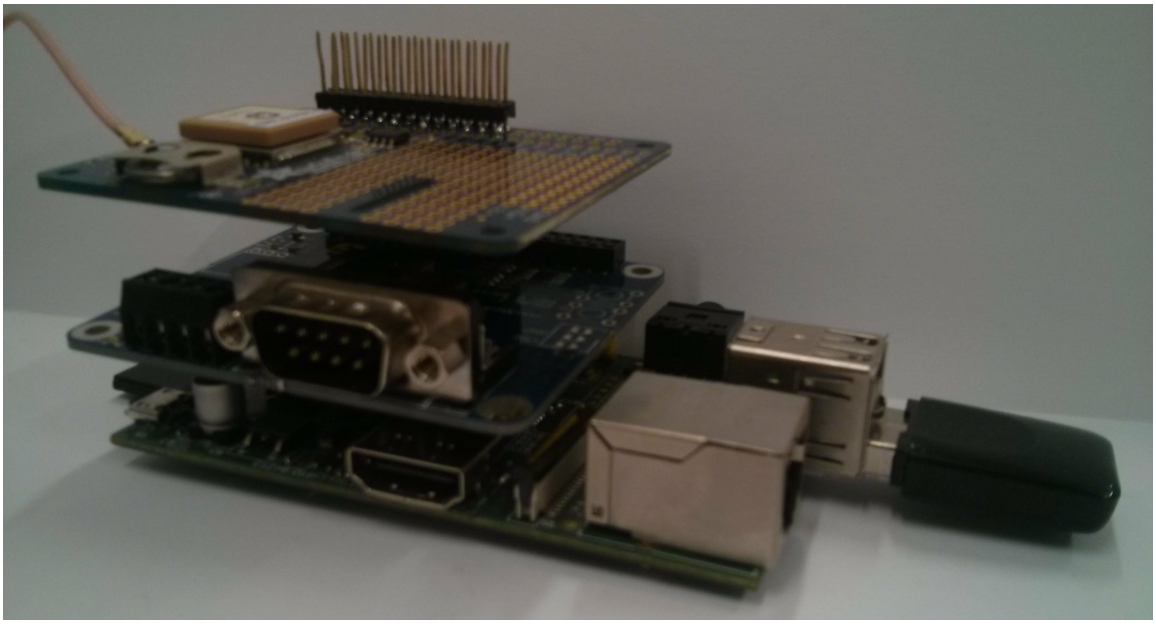


Figure 6.2: Image of the hardware setup

6.1.1 Computational unit

The computational unit in the system processes the data acquired and computes the optimal speed profile based on the acquired data. A Raspberry Pi is selected as the computational unit due to (i) its low cost nature, (ii) the availability of extension modules to extend its functionality, and (iii) the general-purpose operating system (OS) on board. As the Rasp-

berry Pi has a general-purpose OS, almost any programming language can be compiled and executed on board. This also allows the use of external libraries without the need to obtain a specialised library for the unit. A Raspberry Pi, i.e. a microcomputer is selected over a microcontroller such as Arduino because of the potential of executing optimal control algorithm on the go. The Raspberry Pi used in this setup is the Raspberry Pi 1 model B. It has a 700 MHz single-core ARM1176JZF-S central processing unit (CPU) and 512 MB random access memory on board. The OS installed on the Raspberry Pi is Raspbian, a Linux OS based on Debian developed specifically for Raspberry Pi. Figure 6.3 shows a photo of the Raspberry Pi model.

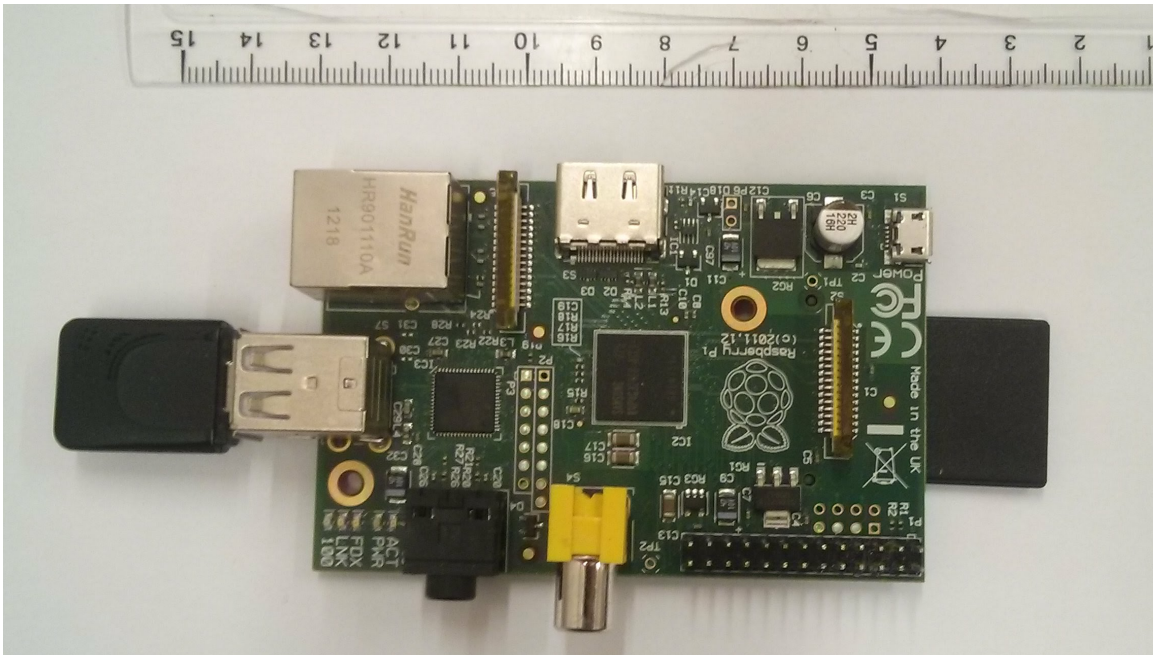


Figure 6.3: Image of Raspberry Pi 1 model B

6.1.2 Data acquisition unit

The purpose of the data acquisition unit in this system is to obtain the data of the location, speed, and heading. As this data can be obtained from GPS data, a GPS module is used. Since the computational unit is a Raspberry Pi, the Adafruit Ultimate GPS HAT for Raspberry Pi is chosen to be the GPS module. HAT is a general term meaning hardware attached on top defined by the Raspberry Pi Foundation to describe an extension board that can be attached on top of a Raspberry Pi. Figure 6.4 shows the photo of the Adafruit Ultimate GPS HAT.

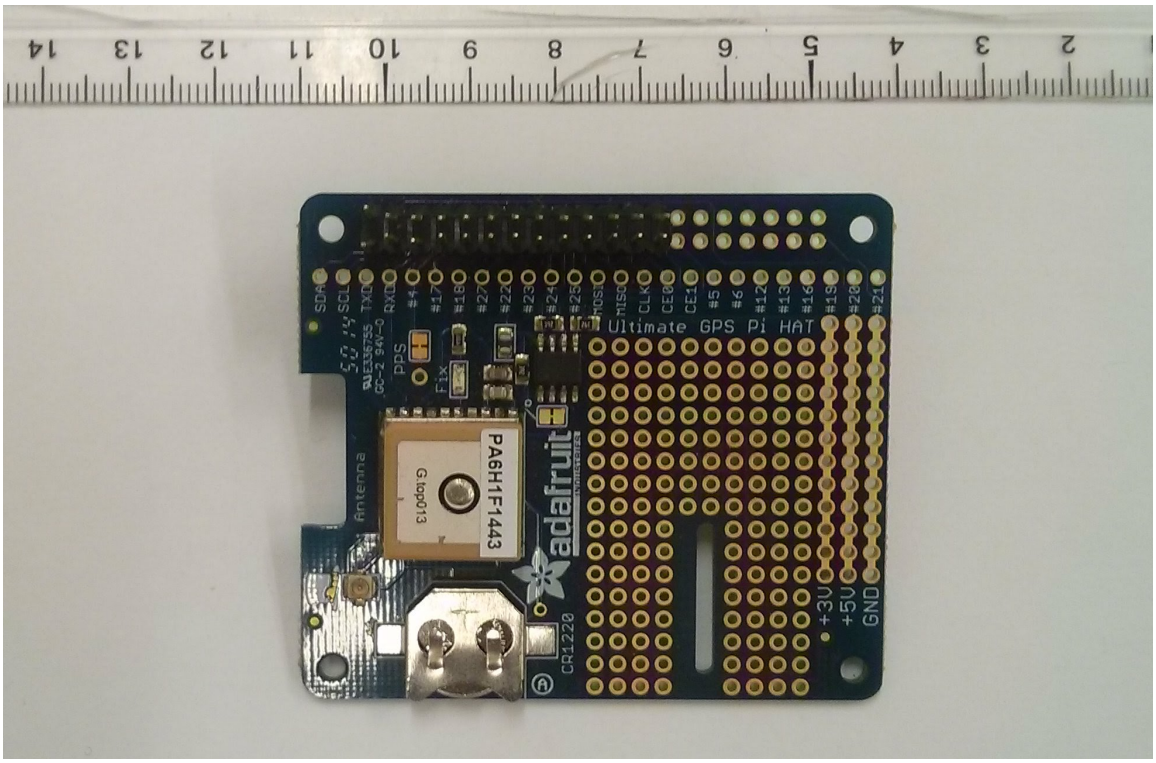


Figure 6.4: Image of Adafruit Ultimate GPS HAT

The GPS HAT communicates with the Raspberry Pi's main board using the univer-

sal asynchronous receiver/transmitter (UART) interface. A built-in patch antenna with -165 dBm is available on the board. As this system is meant to placed in the interior of the bus, an external active antenna is used to increase the accuracy of the acquired data. The external antenna provides an additional gain of 28 dB for the signal. Figure 6.5 shows the photo of the external antenna.

Figure 6.5: Image of external antenna for Adafruit Ultimate GPS HAT

The GPS module used on the HAT is the FGPMMOPA6H module [146] from GlobalTop Technology Inc. The module has a default baud rate of 9600 bits per second. The accuracies for the position and velocity are 3.0 m and 0.1 m/s respectively. The module is able to provide the reading of velocity up to 515 m/s, altitude up to 18000 m and acceleration up to 4G (39.24 m/s^2). The update rate for the module ranges from 1 to 10 Hz. The GPS HAT has, by default, set the update rate to be 10 Hz. The time to first fix of the module is 1 second for hot start, 33 seconds for warm start, and 35 seconds for cold start.

Configurations on the Raspberry Pi need to be edited to enable the communication

between the GPS HAT and the Raspberry Pi. By default, the Raspberry Pi's UART is used for the serial port to communicate with another serial device such as a computer. A computer, connected to the serial port of the Raspberry Pi, is able to log in to the Raspberry Pi system using a serial terminal. In order to enable the UART interface for the GPS HAT, the serial console function of the Raspberry Pi has to be first disabled. The serial console can be disabled using the built-in configuration tool, `raspi-config`. The tool is called by issuing the command '`sudo raspi-config`' through a commandline interface and then select '`Advanced Options > Serial > No`'. The configuration tool simultaneously disables both the serial console and the UART interface. Therefore the UART interface has to be re-activated manually by putting the line '`enable_uart=1`' in the file '`/boot/config.txt`'.

6.1.3 Communication unit

The communication unit is used to transfer the optimal speed to the vehicle in real time. A CAN bus module is chosen since CAN bus protocol is commonly used for communication within vehicle systems. The CAN bus module used is the PiCAN2 [147] CAN bus board that uses the Microchip MCP2515 CAN controller with the MCP2251 CAN transceiver. The CAN bus communication can be connected with a DB9 connector or a 4-way screw terminal. The CAN bus module uses the CAN v2.0B specification with the data transfer rate of 1 Mb/s. The module communicates with the Raspberry Pi using the serial peripheral interface (SPI) bus at 10 MHz. Figure 6.6 shows the photo of the PiCAN2 module.

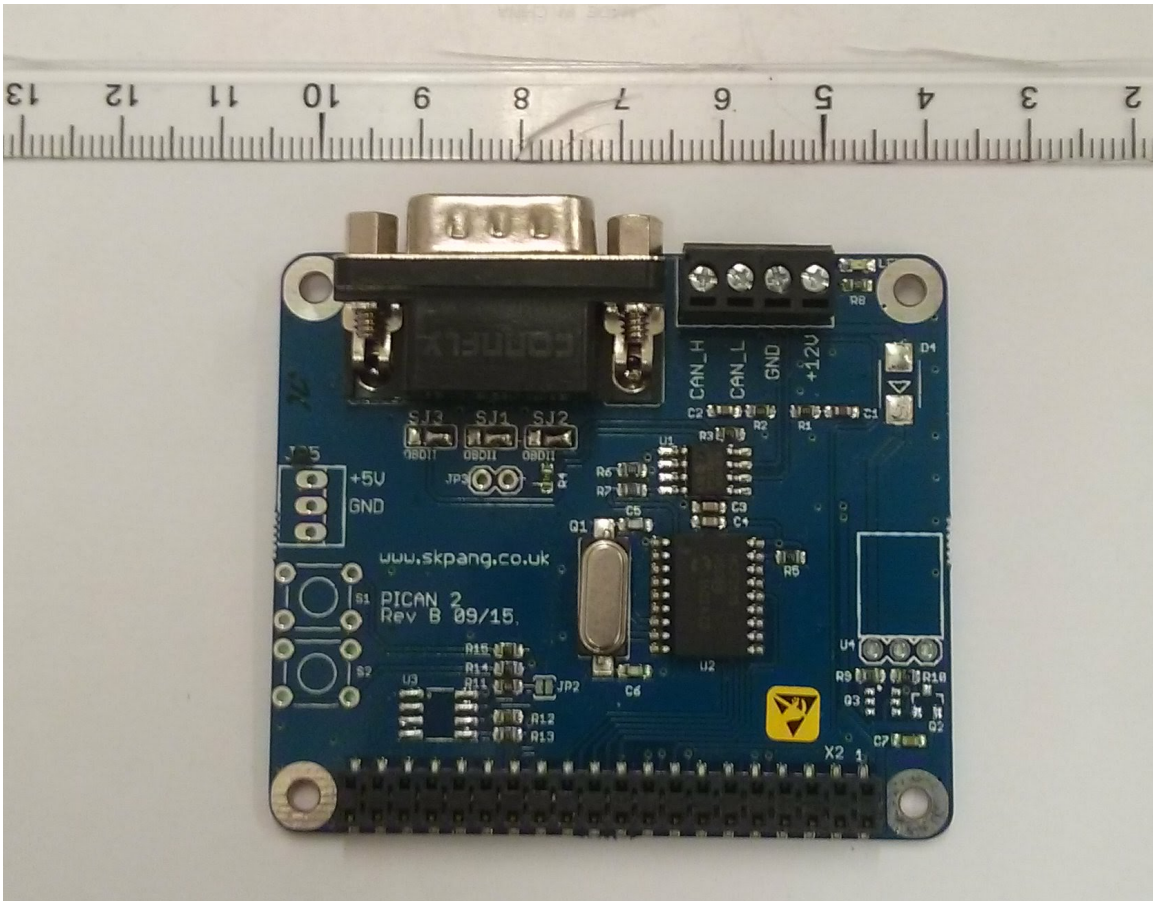


Figure 6.6: Image of PiCAN2

As the PiCAN2 board uses the SPI bus for communication with the Raspberry Pi, the SPI function needs to be enabled on the Raspberry Pi. This is done by including the following lines in the file '`/boot/config.txt`':

```
dtparam=spi=on
dtoverlay=mcp2515-can0,oscillator=16000000,interrupt=25
dtoverlay=spi-bcm2835-overlay.
```

The CAN interface can be activated using the command

```
sudo /sbin/ip link set can0 up type can bitrate 500000.
```

The manufacturer has compiled a set of commandline programs that can be used for sending or receiving CAN messages. A Python library and C header files are also provided to implement the CAN interface in Python or C.

6.2 Software setup

The functions of the software are to first, process the data acquired through the hardware, i.e. the GPS data to identify, of the vehicle, the location, speed, heading and relative position to the route and second, compute the optimal speed profile based on the real-time location and speed of the vehicle.

The GPS data collected using the GPS module is encoded in the format of National Marine Electronics Association (NMEA) sentences. Each sentence provides different set of information. Section 6.2.1 explains the format of NMEA sentences received by the GPS module and identifies the minimal set of NMEA sentences to process to obtain the data of interest.

The location, speed, and heading of the bus can be extracted from the NMEA sentences. For the purpose of identifying the optimal speed at the current location, one has to identify the position of the bus relative to the route needs. Section 6.2.2 discusses the method to find the relative position of the bus on a pre-defined route using its location and heading.

As the bus is moving on the route, multiple reasons can cause the bus not be able to follow the optimal profile. When this happens, an adjustment algorithm is used to modify the optimal profile to achieve the arrival time requirements of the subsequent bus stops.

Section 6.2.3 discusses the adjustment algorithm to modify the optimal profile based on the current position and speed of the bus so that the bus will be able to arrive at subsequent bus stops and the destination on the scheduled time.

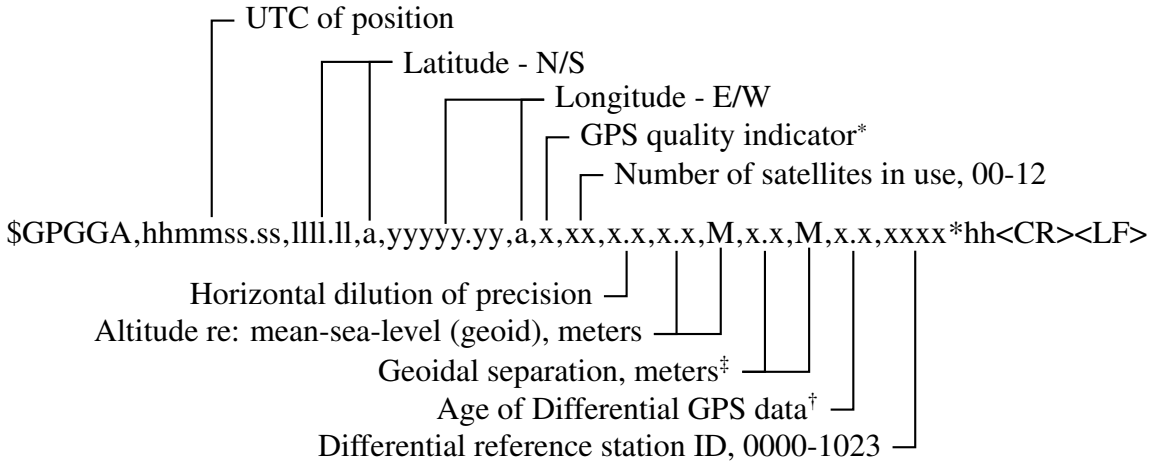
Section 6.2.4 presents the overall software structure of the system. This section explains the relationship between different scripts to provide a real time optimal driving profile.

6.2.1 GPS NMEA sentences processing

GPS NMEA sentences are GPS data formatted in NMEA standard, a standard developed by the National Marine Electronics Association for the purpose of interfacing marine electronic devices [148]. Different devices provide different sets of NMEA sentences and each sentence corresponds to a different collection of information.

The NMEA sentences provided by the GPS HAT used in this setup are GPGGA, GPGSA, GPGSV, GPRMC and GPVTG. The first two characters of the sentence name constitute the talker identification. The talker device is the source of the data transmission. In the listed sentences, the talker identification is GP, which implies that the talker device is a GPS device. The definitions of these sentences are based on the three characters following the talker identification.

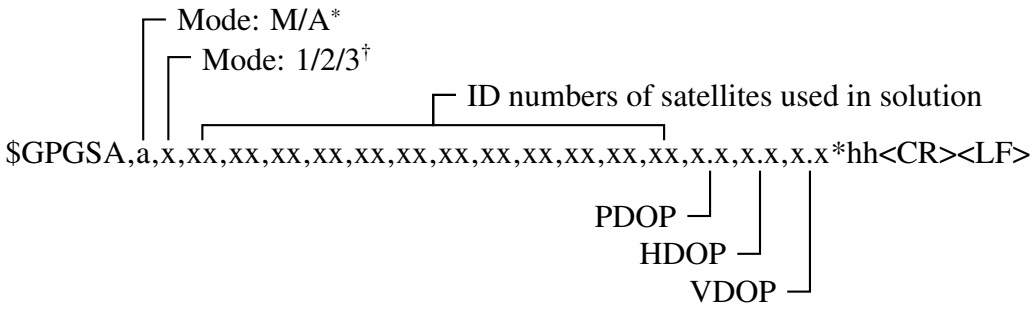
The GPGGA sentence provides the GPS fix data. The sentence provides the time of the data, position and parameters that define the quality of the fix. Figure 6.7 shows the general format and definition of each word within the GPGGA sentence.



Notes:

1. *GPS Quality Indicator: (The GPS Quality Indicator field shall not be a null field)
   - 0 = Fix not available or invalid
   - 1 = GPS SPS Mode, fix valid
   - 2 = Differential GPS, SPS Mode, fix valid
   - 3 = GPS PPS Mode, fix valid
   - 4 = Real Time Kinematic. System used in RTK mode with fixed integers
   - 5 = Float RTK. Satellite system used in RTK mode, floating integers
   - 6 = Estimated (dead reckoning) Mode
   - 7 = Manual Input Mode
   - 8 = Simulator Mode
2. †Time in seconds since last SC104 Type 1 or 9 update, null field when differential GPS is not used
3. ‡Geoidal Separation: the difference between the WGS-84 earth ellipsoid surface and mean-sea-level (geoid) surface, "-" = mean-sea-level surface below WGS-84 ellipsoid surface.

Figure 6.7: GPGGA sentence definition



Notes:

1. *Mode:
   M = Manual, forced to operate in 2D or 3D mode
   A = Automatic, allowed to automatically switch 2D/3D

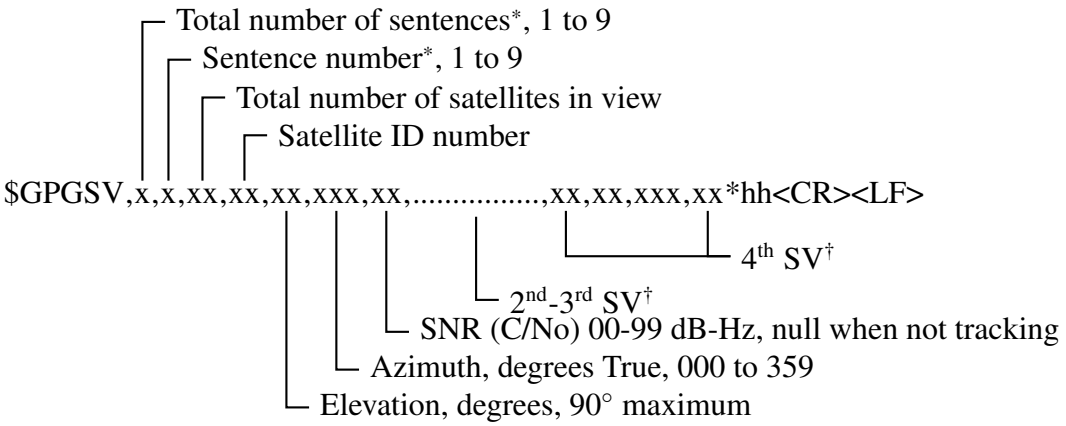
2. †Mode:
   1 = Fix not available
   2 = 2D
   3 = 3D

Figure 6.8: GPGSA sentence definition

The GPGSA sentence provides the status of the satellites by encoding the number of satellites and the dilution of precision. The format and definitions for the GPGSA sentence are presented in Figure 6.8.

The GPGSV sentence provides the information of satellites in view. This information of the number of satellites and the individual details of each satellite. If the total number of satellites in view is more than four, the satellite details will be separated into multiple sentences with each sentence housing the information of four satellites at most where only one sentence may have less than four. For example, the data for ten satellites would be split as four, four, two. The format and definitions for GPGSV sentence are shown in Figure 6.9.

The GPRMC sentence provides the recommended minimum GPS data, which includes



Notes:

1. *Satellite information may require the transmission of multiple sentences all containing identical field formats when sending a complete message. The first field specifies the total number of sentences, minimum value 1. The second field identifies the order of this sentence (sentence number), minimum value 1. For efficiency it is recommended that null fields be used in the additional sentences when the data is unchanged from the first sentence.

2. †A variable number of "Satellite ID-Elevation-Azimuth-SNR" sets are allowed up to a maximum of four sets per sentence. Null fields are not required for unused sets when less than four sets are transmitted.

Figure 6.9: GPGSV sentence definition

the position, speed, direction and time of fix of the GPS receiver. Figure 6.10 presents the format and definitions of the GPRMC sentence.

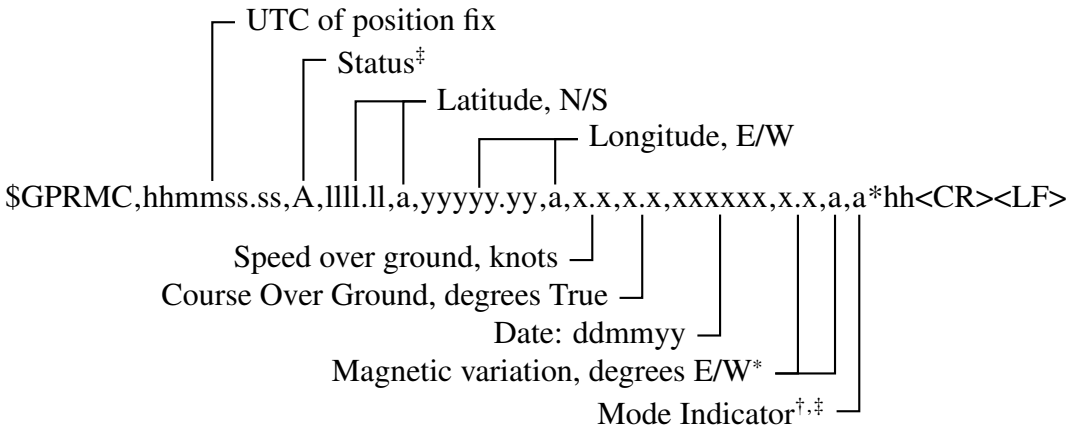
The GPVTG sentence provides the information of the course over ground and the ground speed, i.e. the course and the speed relative to the ground. Figure 6.11 shows the format and definitions for the GPVTG sentence.

The only data needed for the purpose of this system is the location in the form of latitude and longitude, the speed, the direction to which the vehicle/receiver is heading, and the recorded time of the data. The data can be obtained solely from the GPRMC sentence. The lack of satellite fix will also be reflected in the GPRMC sentence. Therefore, by only processing the GPRMC sentence, the availability of satellite and the required information can be obtained.

6.2.2 Vehicle position identification

The data obtained from the GPRMC sentence gives the location, speed and heading of the vehicle. However, this information is not sufficient to identify the optimal speed or update the optimal speed profile. This absolute location of the vehicle has to be translated into the location of the vehicle relative to the route.

Given a single route and the prior knowledge that the vehicle is on the route, the relative position of the vehicle on the route can be found by using an algorithm modifying the one in Section 4.3. The modification is made by replacing the bus stop location with the vehicle GPS location.



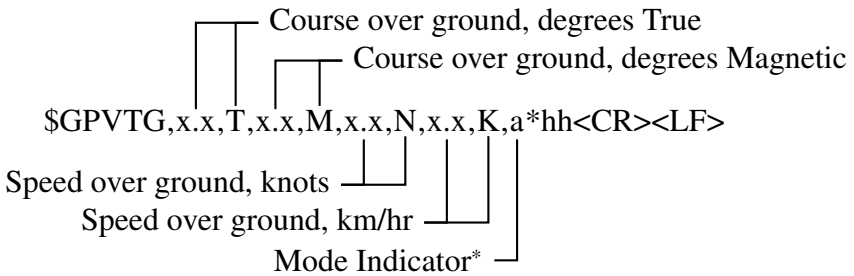
Notes:

1. *Easterly variation (E) substracts from True course; Westerly variation (W) adds to True course

2. †Positioning system Mode Indicator:
   A = Autonomous mode
   D = Differential mode
   E = Estimated (dead reckoning) mode
   M = Manual input mode
   S = Simulator mode
   N = Data not valid

3. ‡The positioning system Mode Indicator field supplements the positioning system Status field, the Status field shall be set to V = Invalid for all values of Indicator mode except for A= Autonomous and D = Differential. The positioning system Mode Indicator and Status fields shall not be null fields.

Figure 6.10: GPRMC sentence definition



Notes:

1. *Refers to GPRMC definition (Figure 6.10) note 2.

Figure 6.11: GPVTG sentence definition

Using the example in Figure 6.12, with points $\mathbb{A}$, $\mathbb{B}$, $\mathbb{C}$, and $\mathbb{D}$ as the points on the route and point β as the vehicle position, the pair of points between which the vehicle is located can be found by first identifying the point in the route nearest to the vehicle. In this case the nearest point is $\mathbb{B}$. Then the smallest angle formed by the nearest point, the vehicle, and the point prior to the nearest point in the listed order, namely $\angle\mathbb{B}\beta\mathbb{A}$, is compared with the smallest angle formed by the nearest point, the vehicle, and the point after the nearest point, namely $\angle\mathbb{B}\beta\mathbb{C}$. The larger of the two angles provides the point that encapsulates the vehicle together with the nearest point. In this case, point $\mathbb{C}$. This approach suffice even if the vehicle does not follow the route accurately, such as at positions β'_1, and β'_2.

An exception occurs when the nearest point to the vehicle is the first or last point on

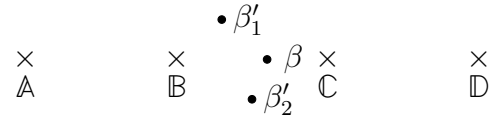


Figure 6.12: Vehicle position relative to a route

the route. When the nearest point is an extremal point on the route, there will only be one angle as opposed to the aforementioned two angles. If this angle is sufficiently large, the vehicle lies between the nearest point and the other point. Otherwise the vehicle lies before the first point or after the last point. This is illustrated in Figure 6.13. β_1, β_2, β'_1, and β'_2 are the possible positions of the vehicle if the nearest point is the first or last point on the route. β_1 and β_2 are the vehicle positions that do not lie between any two points whereas β'_1 and β'_2 lie between two points of the route.

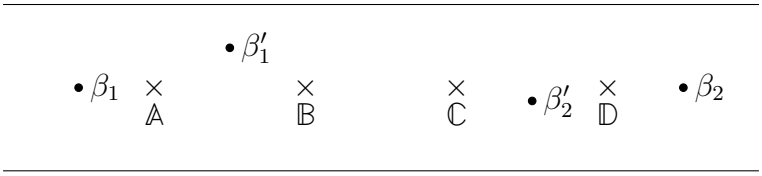


Figure 6.13: Vehicle position on a route if the nearest point is the first or the last point

The vehicle location relative to the route in terms of its distance from the start of the route can be calculated by summing the distance between the vehicle position and the last point prior to the position with the distance of that point from the first point of route. The distance of each point on the route from the first point is pre-calculated and available in the route database.

Most bus routes are designed such that the bus goes from one end to another and go back to the original location through the same path in the opposite direction. As the route in the opposite direction has different bus schedules, they are considered as two routes. An algorithm is developed to identify the changing of the route. This algorithm identifies if the bus has entered the starting zone of either one of the routes and resets the current route name and the starting time of the current schedule. The current schedule is identified when

the bus leaves the starting zone of the route. The schedule with the closest starting time to the time of the bus leaving the starting zone is registered as the current schedule.

6.2.3 Adjustment algorithm

When the bus does not follow the optimal profile, the profile needs to be updated according to the latest known states (speed and location) of the bus. One way to update the optimal profile is to execute the optimal control algorithm on the go. This is possible as the Raspberry Pi used as the computational unit has a general-purpose OS. However, due to the low computing power of the CPU of Raspberry Pi compared to a personal computer's CPU, the time to execute the optimal control algorithm is too long for real-time implementation. The time taken to solve an optimal control problem with one-kilometre one-minute route with no bus stop on the Raspberry Pi is 17.23 seconds. Therefore, when the updated optimal profile is obtained, the bus will be at a different location from the states used for updating the optimal profile. The updated profile is not the optimal profile for the bus at the latest states.

In order to reduce the computational time, an adjustment algorithm is developed to modify the optimal profile when the bus does not follow the optimal profile. The algorithm identifies the segment of the optimal profile to be scaled and updates the profile such that the time-location constraints are retained.

When the bus is unable to follow the optimal profile, the actual location of the bus is different from its optimal location. In order to arrive at the next time-constrained location

on the scheduled time, the speed profile needs to be updated. A time-constrained location is a location at which the arrival time is scheduled. This is either a bus stop or the destination.

When the discrepancy between the actual and the optimal locations exceeds an appropriately defined tolerance, the adjustment algorithm is initiated. Let the location, speed, and time at this moment be s_c, v_c, and t_c respectively. First, the location and the scheduled time of the next time-constrained location are identified as s_n and t_n respectively. The optimal time to arrive at location s_c is identified as t_o from the optimal distance profile.

Let $v_o(t)$ and $s_o(t)$ be the optimal speed and distance profiles respectively. The profiles are shown in Figures 6.14 together with variables s_c, s_n, v_c, v_o, t_o, t_c, and t_n.

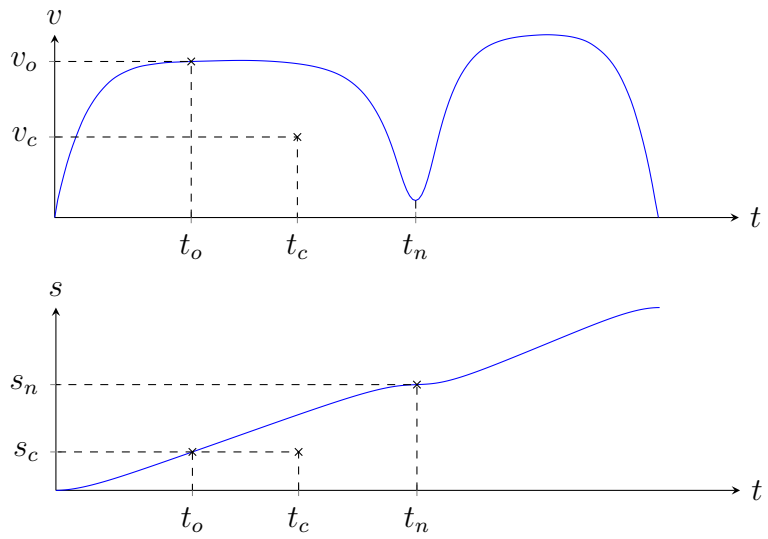


Figure 6.14: Example of optimal profiles for demonstration of adjustment algorithm (Top: speed profile; Bottom: distance profile)

The speed profile from time t_o to t_n is extracted as $v_o(t)$. The time variable is then scaled from the range of $[t_o, t_n]$ to $[t_c, t_n]$. The new speed profile with the scaled time is defined as $v_{o'}(t)$ in Equation 6.1.

$$v_{o'}(t) = v_o\left(\frac{t - t_c}{t_n - t_c}\left(t_n - t_o\right) + t_o\right) \quad \text{for } t \in [t_c, t_n] \tag{6.1}$$

The total distance travelled with the speed profile $v_{o'}(t)$ is different from the total distance travelled with the speed profile $v_o(t)$. Therefore $v_{o'}(t)$ needs to be adjusted using a coefficient $A(t)$ to form the adjusted speed profile $v_n(t)$ as shown in Equation 6.2. The adjusted speed profile $v_n(t)$ should have the same distance travelled as the speed profile $v_o(t)$. This condition is shown in Equation 6.3. As the total distance travelled with the speed profile $v_o(t)$ from t_o to t_n equals the distance between current location s_c and the next time-constrained location s_n, using Equation 6.2, Equation 6.3 can be rewritten as Equation 6.4.

$$v_n(t) = A(t)v_{o'}(t) \quad \text{for } t \in [t_c, t_n] \tag{6.2}$$

$$\int_{t_c}^{t_n} v_n(t)\, dt = \int_{t_o}^{t_n} v_o(t)\, dt \tag{6.3}$$

$$\int_{t_c}^{t_n} A(t)v_{o'}\, dt = s_n - s_c \tag{6.4}$$

The adjusted profile can then be identified by solving for the function $A(t)$. The simplest form of $A(t)$ is a constant. If we let $A(t)$ be the constant α_A, it can be calculated using Equation 6.5.

$$\alpha_A = \frac{s_n - s_c}{\int_{t_c}^{t_n} v_{o'}\, dt} \tag{6.5}$$

The adjusted profile replaces the segment of the optimal profile from t_c to t_n. If the next time-constrained location is a compulsory bus stop or the destination, $v_{o'}(t_n)$ is zero and its multiplication with α does not change the value. The transition from the adjusted profile to the optimal profile from t_n to the finishing time t_f is smooth. However, if the next time-constrained location is an on-demand bus stop, $v_{o'}(t_n)$ may not be zero. The multiplication of $v_{o'}(t_n)$ with α changes its value. This introduces a sudden change at t_n of the joint profile, and this leads to a high acceleration at the location of the on-demand bus stop.

Another issue with the definition of $A(t)$ as a constant is the difference between the speed at the beginning of the adjusted profile $v_n(t_c)$ and the actual speed of the bus v_c when $v_{o'}(t_c)$ is a non-zero value. This has the effect of a high acceleration when the bus tries to follow the adjusted profile immediately.

Therefore, the function $A(t)$ has to fulfil the criteria of (i) the ending value is one, (ii) the starting value is $\frac{v_c}{v_{o'}(t_c)}$, (iii) the distances travelled from t_o to t_n by $v_n(t)$ and $v_o(t)$ are equal, and (iv) the first derivative of the function is small. Figure 6.15 shows the function that fulfil criteria (i), (ii), and (iv). Criterion (iii) is fulfilled by solving Equation 6.4 to identify the value of a_u.

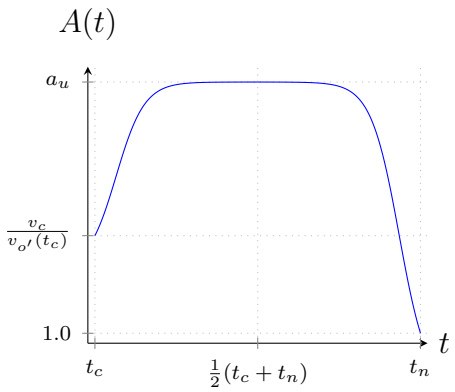


Figure 6.15: Function $A(t)$ to be derived

$A(t)$ is constructed in two parts with the first part $A_1(t)$ from t_c to $\frac{1}{2}(t_c + t_n)$, and the second part $A_2(t)$ from $\frac{1}{2}(t_c + t_n)$ to t_n. The base function for $A_1(t)$ is a sigmoid function $f_1(x)$ with the domain of -15 to 100 as in Equation 6.7. The base function for $A_2(t)$ is a reverse sigmoid function $f_2(x)$ with the domain of -100 to 15 as in Equation 6.8. The functions $f_1(x)$ and $f_2(x)$ are visualised in Figure 6.16.

$$A(t) = \begin{cases} A_1(t) & \text{for} \quad t_c \leq t < \frac{1}{2}(t_c + t_n) \\ A_2(t) & \text{for} \quad \frac{1}{2}(t_c + t_n) \leq t \leq t_n \end{cases} \tag{6.6}$$

$$f_1(x) = \frac{1}{1 + e^{-0.1x}} \qquad \text{for} \quad -15 \leq x < 100 \tag{6.7}$$

$$f_2(x) = \frac{1}{1 + e^{0.1x}} \qquad \text{for} \quad -100 \leq x \leq 15 \tag{6.8}$$

The transformation from the base functions $f_1(x)$ and $f_2(x)$ to functions $A_1(t)$ and $A_2(t)$ is achieved by scaling the base functions in horizontal and vertical axes with appropriate parameters. $A_1(t)$ and $A_2(t)$ are visualised in Figure 6.17 and their formulae are

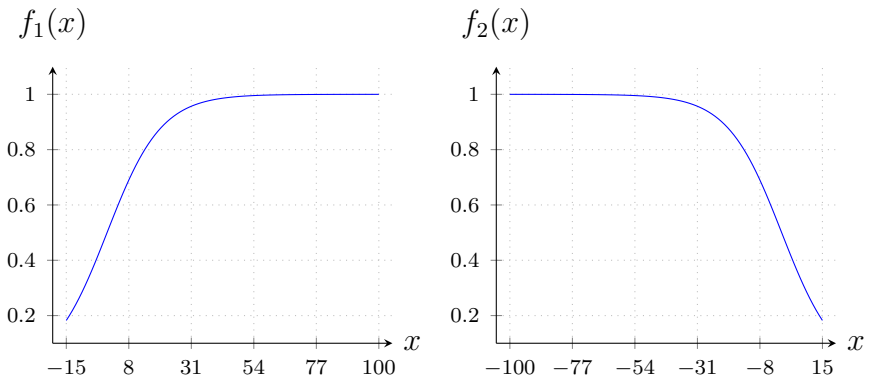


Figure 6.16: Base functions for $A_1(t)$ (left) and $A_2(t)$ (right) respectively

shown in Equations 6.9 and 6.10. $x_{1\text{min}}$ and $x_{1\text{max}}$ are the minimum and maximum values of the range of x in $f_1(x)$. In this case $x_{1\text{min}}$ and $x_{1\text{max}}$ are -15 and 100 respectively. Similarly, $x_{2\text{min}}$ and $x_{2\text{max}}$ are the minimum and maximum values of the range of x in $f_2(x)$ and they are -100 and 15 respectively.

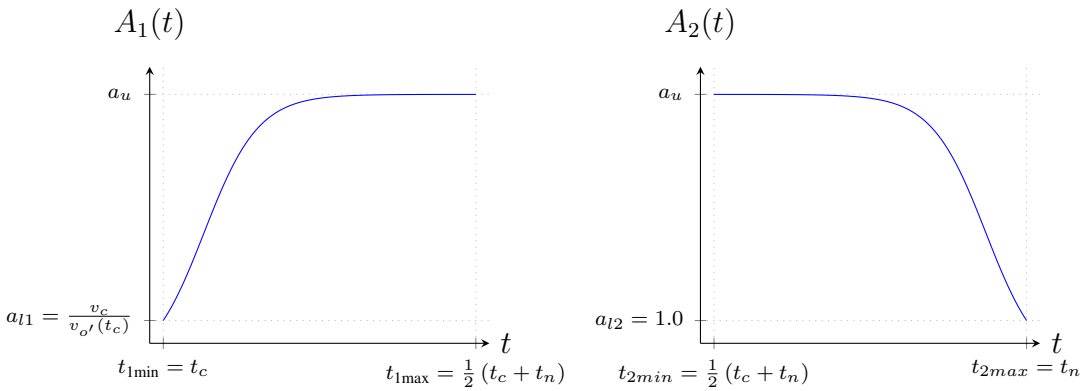


Figure 6.17: Functions $A_1(t)$ (left) and $A_2(t)$ (right)

$$A_1(t) = \frac{g_1(t) - \min(g_1(t))}{\max(g_1(t)) - \min(g_1(t))}(a_u - a_{l1}) + a_{l1} \tag{6.9}$$

$$\text{for} \qquad t_c \leq t < \frac{1}{2}(t_c + t_n)$$

$$\text{where} \quad g_1(t) = f_1(T_1)$$

$$\text{and} \quad T_1 = \frac{t - t_{1\text{min}}}{t_{1\text{max}} - t_{1\text{min}}} \left(x_{1\text{max}} - x_{1\text{min}}\right) + x_{1\text{min}}$$

$$A_2(t) = \frac{g_2(t) - \min(g_2(t))}{\max(g_2(t)) - \min(g_2(t))} \left(a_u - a_{l2}\right) + a_{l2} \tag{6.10}$$

$$\text{for} \quad \frac{1}{2}\left(t_c + t_n\right) \leq t \leq t_n$$

$$\text{where} \quad g_2(t) = f_2(T_2)$$

$$\text{and} \quad T_2 = \frac{t - t_{2\text{min}}}{t_{2\text{max}} - t_{2\text{min}}} \left(x_{2\text{max}} - x_{2\text{min}}\right) + x_{2\text{min}}$$

By solving Equation 6.4 with the formulation of $A(t)$ in Equations 6.6, 6.9, and 6.10, the value of a_u can be identified using Equation 6.11.

$$a_u = \frac{\int_{t_c}^{t_n} v_{o'}(t) \left[1 - a_l \left(1 - G(t)\right)\right] dt}{\int_{t_c}^{t_n} G(t) v_{o'}(t)} \tag{6.11}$$

$$\text{where} \quad a_l = \begin{cases} a_{l1} & \text{for} \quad t_c \leq t < \frac{1}{2}\left(t_c + t_n\right) \\ a_{l2} & \text{for} \quad \frac{1}{2}\left(t_c + t_n\right) \leq t \leq t_n \end{cases}$$

$$\text{and} \quad G(t) = \begin{cases} \frac{g_1(t) - \min(g_1(t))}{\max(g_1(t)) - \min(g_1(t))} & \text{for} \quad t_c \leq t < \frac{1}{2}\left(t_c + t_n\right) \\ \frac{g_2(t) - \min(g_2(t))}{\max(g_2(t)) - \min(g_2(t))} & \text{for} \quad \frac{1}{2}\left(t_c + t_n\right) \leq t \leq t_n \end{cases}$$

The identified function of $A(t)$ is then multiplied with $v_{o'}(t)$ and the resultant speed profile replaces the segment of the optimal profile from t_c to t_n. The corresponding distance

profile is calculated by integration of the new speed profile.

The adjustment algorithm can be summarised as:

1. extract $v_o(t)$ for $t \in [t_o, t_n]$
2. scale $v_o(t)\Big|_{t_o}^{t_n}$ to be $v_{o'}(t)\Big|_{t_c}^{t_n}$ using Equation 6.1
3. new profile, $v_n(t)$ calculated with Equation 6.2

6.2.4 Software structure

The software layer of the system is responsible to (i) obtain the location and speed data of the system, (ii) identify the position of the system relative to the bus route, (iii) identify the need to adjust the optimal profile and execute the adjustment algorithm, and (iv) deliver the optimal speed to the external device.

This software layer is implemented with multiple scripts interacting through a number of dynamic files that store different data. This approach, compared to a single script approach, reduces the complexity of the code by splitting a potentially large script into multiple small scripts. Each script is only responsible for a single task. This also reduces the response time of the system as the scripts run in parallel instead of in sequence. The response time for one cycle (each script runs at least once) is the maximum of all the script execution times. If the scripts are implemented in a sequential manner, the response time for one cycle will be the sum of the individual script execution times.

There are in total six individual scripts that each performs a different task alongside

with four base database files that store the basic data. The four database files are `route`, `startlocationbound`, `schstart`, and `optprof`. `route` contains the route data of the bus including the distance between each sampled set of GPS coordinates along the route and the cumulated distance of each set from the beginning of the route. `startlocationbound` consists of the GPS bound that is considered as the starting location for each route. `schstart` stores the lists of times at which the bus should leave the starting locations. `optprof` provides the pre-calculated optimal speed profile for each route.

The six scripts are `gpsreader`, `gpsparser`, `identifystartofsch`, `mapsgpsroute`, `correctoptim`, and `sendoptimaltocan`. `gpsreader` continuously polls the data from the GPS device and calls the `gpsparser` function as another process when a GPRMC sentence is received. The interaction between `gpsreader` and `gpsparser` is the only exception that does not use a dynamic file but a function call. `gpsparser` interprets the GPRMC sentence and records the time, the GPS coordinates, the bearing, and the speed in a dynamic file named `lastdata`. `identifystartofsch` function polls the data from the `lastdata` file, and based on the `startlocationbound` database, identifies if the system is within the bound of any start location. If the system is in a start location, the script will reset the content of the `lastdistance` file and `schstart` file to `0`. The `lastdistance` file stores the distance of the current location from the beginning of the route and the `schstart` file stores the supposed starting time of the current run. At the same time, the `lastroute` file is updated with the route name corresponding to the start location. If the system is not in a start location, the script checks the content of the `schstart` file. If the

schstart file contains the value of 0, the script will identify the closest starting time of the current route from the lists in the schstart database and update the schstart file with it. The optprofcsv file is then updated with the pre-calculated optimal profile from the optprof database. optprofcsv contains the current optimal profile. If the schstart file contains a valid time string when the system is not in a start location, the script will not perform any action and just continue to poll data from the lastdata file until any mentioned condition is fulfilled.

The mapgpstoroute script first checks the content of the schstart file. If the schstart file contains a valid time string, the script polls the lastdata file for the GPS data of the system and identifies the position of the system relative to the current route using the algorithm explained in Section 6.2.2. The current route is obtained based on the data in the lastroute file. This script then updates the lastdistance file with the current distance of the system from the beginning of the current route.

The correctoptim script is responsible for identifying and performing any necessary adjustment of the optimal profile. The script first checks the schstart file for a valid time string. The current time relative to the content of the schstart file, i.e. the beginning of the route, is identified. This relative time is used to interpolate the supposed current optimal distance from the optimalprofcsv file. This distance is compared to the actual distance in lastdistance file. If these values do not agree with each other within an appropriately defined tolerance, the adjustment algorithm described in Section 6.2.3 is triggered and the optimal profile in optprofcsv is then updated with the adjusted profile.

sendoptimaltocan script constantly polls from the optprofcsv file and identifies the optimal speed at current time. The script then sends the optimal speed through the CAN bus module to the external device. Figure 6.18 presents the interactions between the scripts and data files.

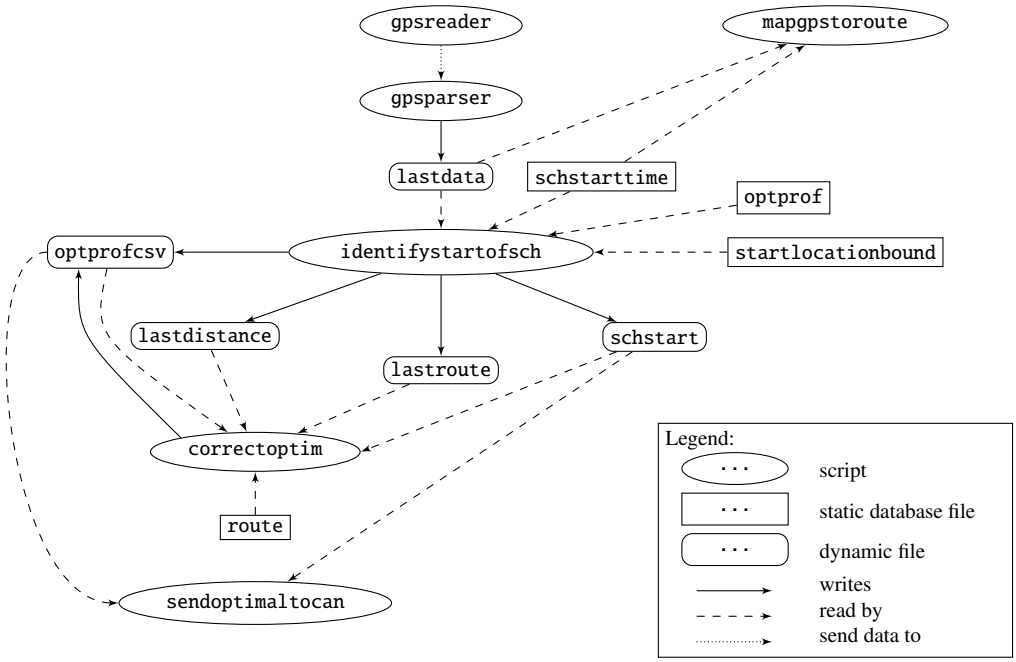


Figure 6.18: Interaction between scripts and data files in the software system

6.3 Scenarios for profile adjustment

The necessity to update the optimal profile occurs when the bus does not follow the optimal profile. This situation is identified when the discrepancy between the actual location of the bus and the optimal location of the bus exceeds an appropriately defined tolerance.

There are three scenarios that cause the discrepancy between the actual and the optimal location of the bus. The example journey used to demonstrate these scenarios is an

extracted segment of the bus journey from Oxford City Centre to Pear Tree Park and Ride visualised in Figures 3.14 and 3.16 in Chapter 3. The journey segment starts from the compulsory bus stop at 713 m at 90 s and finishes at 1713 m at 240 s with the on-demand bus stop at 1292 m at 180 s. By redefining the starting location and time of the journey segment to be 713 m and 90 s, the example journey is shown in Figure 6.19.

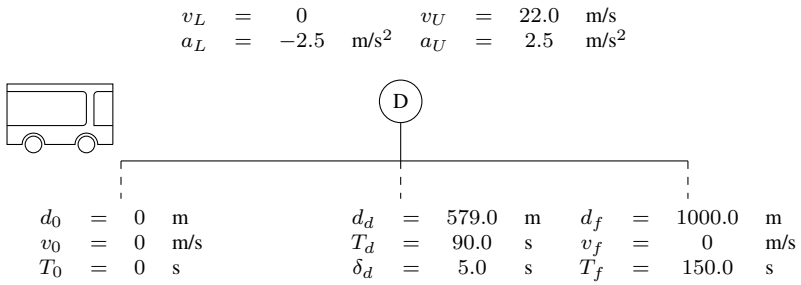


Figure 6.19: Example journey

Scenario I happens when there is a demand to stop at an on-demand bus stop due to passenger alighting or boarding the bus. In order to handle this scenario, an optimal profile in which the on-demand bus stop is defined as compulsory bus stop is pre-computed. When the bus is identified to have stopped at the on-demand bus stop, the second optimal profile is used for the rest of the journey. The two sets of the optimal profiles are shown in Figure 6.20.

Scenario II happens when the bus starts off later than the original scheduled time. The location where the bus starts off can either be the beginning of the journey or a compulsory bus stop. The adjustment algorithm as discussed in Section 6.2.3 is initiated when this happens. For the demonstration of this scenario with the example journey, the bus has a delayed start of 10 s from the beginning of the journey. The adjustment algorithm is

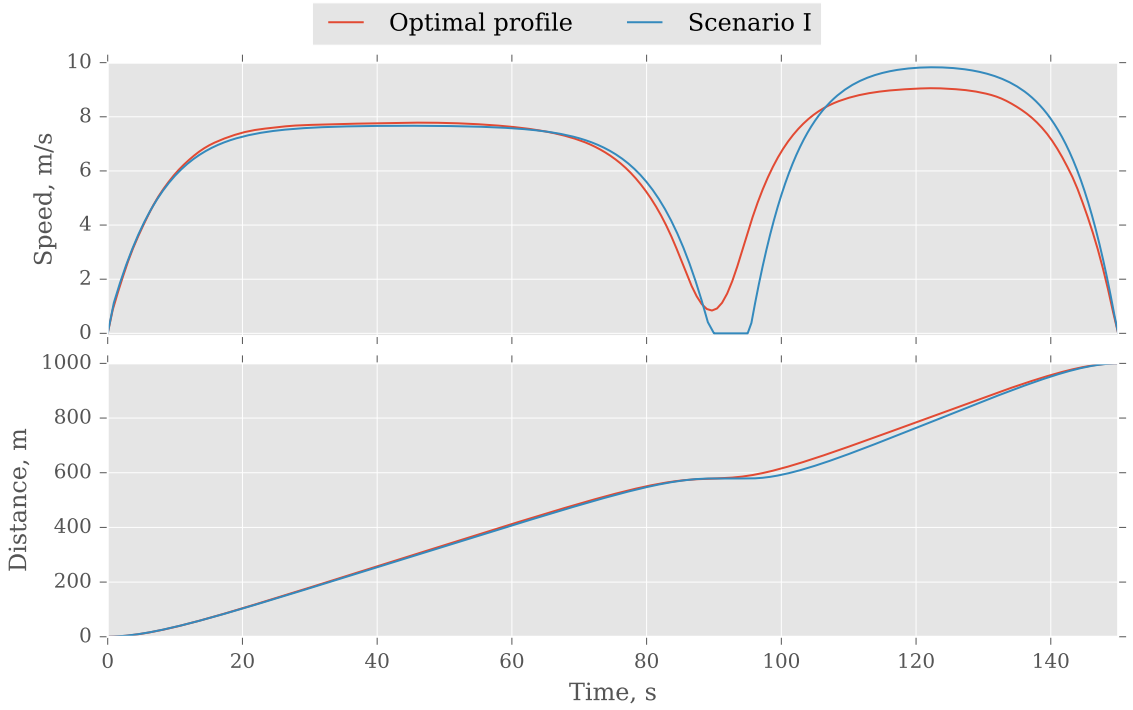


Figure 6.20: When stopping at on-demand bus stop

initiated at $t = 10$ s. Figure 6.21 shows the adjusted profile. Optimal control is also executed with the initial time of 10 s, initial location of 0 m, and initial speed of 0 m/s to obtain the actual optimal profile for the delayed scenario. The re-optimised profiles are also shown in Figure 6.21.

Scenario III happens when the speed of the bus is slowed by the traffic for a period of time, causing the distance travelled by the bus less than the optimal distance to be travelled. For the demonstration of this scenario, the bus is allowed to travel according to the optimal profile at the beginning of the journey. Then the speed of the bus is kept constant due to the limitation of traffic for 20 seconds. The adjustment algorithm is initiated when the limitation due to the traffic is removed. The adjusted profile is shown in Figure 6.22.

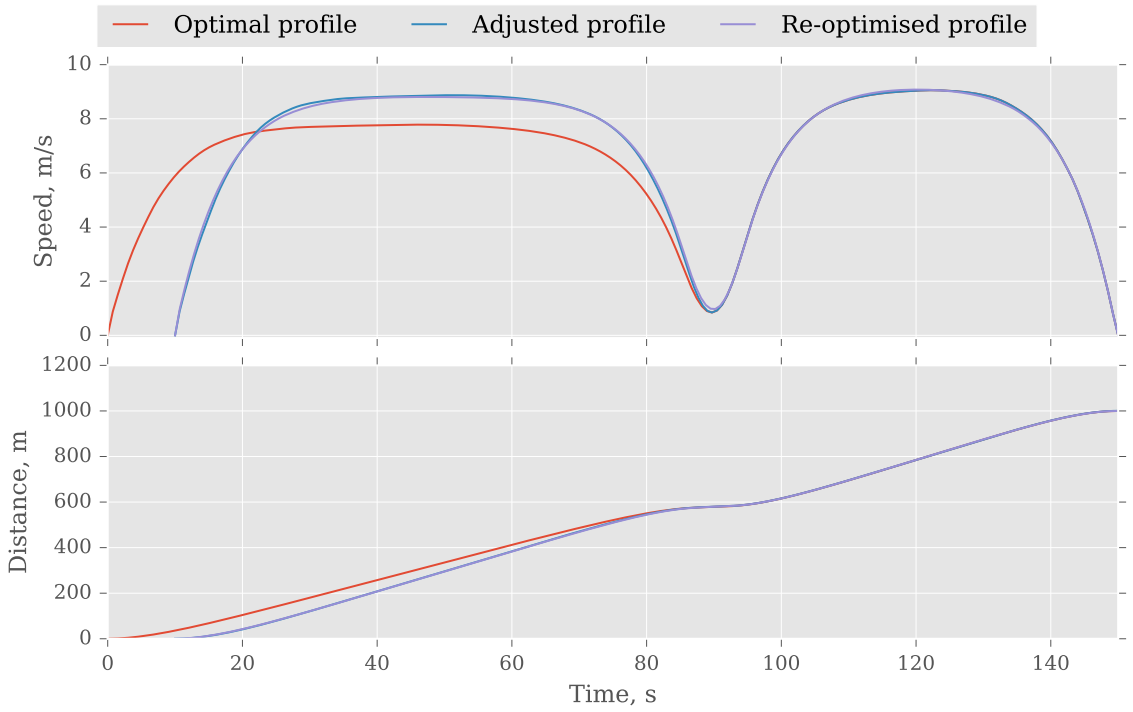


Figure 6.21: When start slow from a bus stop

Optimal profiles obtained from the re-optimisation of the journey with the initial conditions of the delayed scenario are shown in Figure 6.22.

Table 6.1 compares the energy consumption of the original optimal profiles, adjusted profiles and re-optimised profiles for all scenarios. The re-optimised profiles have a lower energy consumption compare to the corresponding adjusted profiles. However, the benefit of re-optimising over adjusting is not significant. The reduction in energy by re-optimising over adjusting is only around 0.2%. The adjustment algorithm is able to provide a similar outcome compares to re-optimisation without the drawback of the time-consuming process.

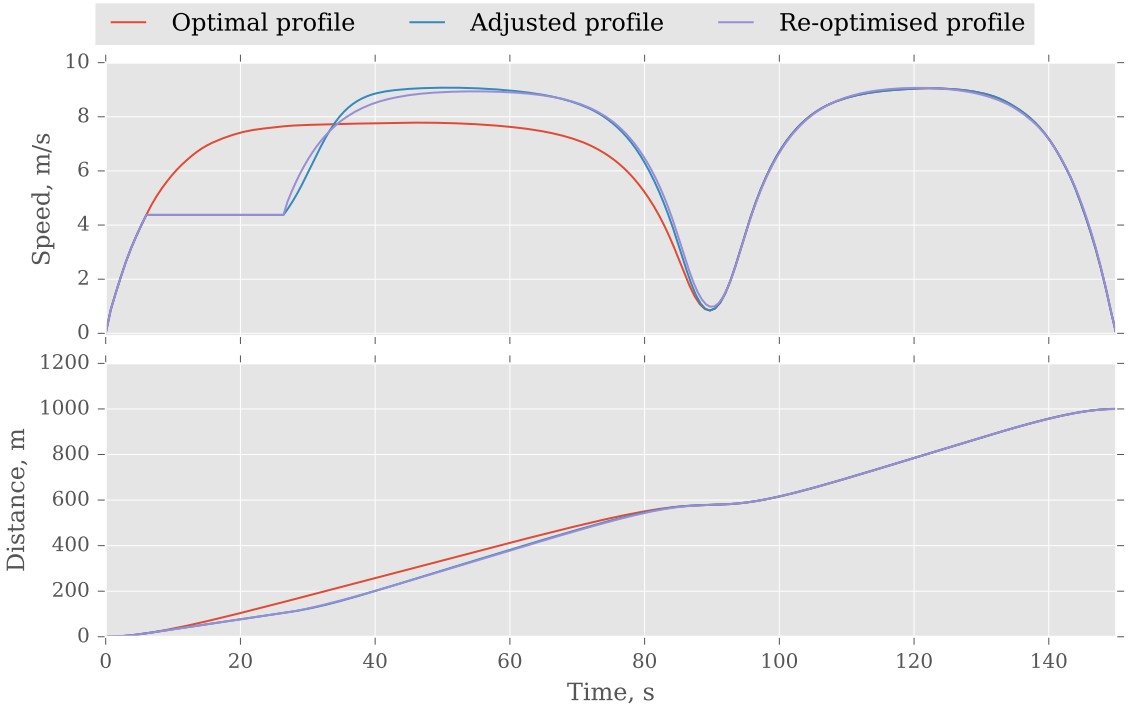


Figure 6.22: When delayed due to traffic

Table 6.1: Energy consumption of profiles in different scenarios

Scenario	Method	Energy consumption (kJ)
optimal		136.15
I (stop on demand)		139.52
II (delayed start)	Adjustment	141.04
	Re-optimisation	140.72
III (slowed by traffic)	Adjustment	138.37
	Re-optimisation	138.05

6.4 Summary

A system is designed to provide optimal speed to the bus at real time based on the real-time information of the bus. The system, in terms of hardware, consists of a computational unit

(Raspberry Pi), a data acquisition unit (GPS HAT) and a communication unit (PiCAN2). The software part of the system is responsible to (i) process the GPS data received through the data acquisition unit, (ii) identify the relative position of the vehicle in the route, and (iii) adjust the optimal speed profile to adapt to real-time situation. Three scenarios in which the bus is unable to follow the optimal profiles are described alongside with their handling strategies. The three scenarios are (i) the stopping of the bus at an on-demand bus stop, (ii) the delayed start of the bus from rest, and (iii) the delayed of the bus due to traffic. Scenario (i) is solved by providing a secondary optimal profile where the on-demand bus stop is defined as a compulsory bus stop. Scenarios (ii) and (iii) are solved by adjusting the optimal profile with the adjustment algorithm. The adjustment algorithm is compared to the re-execution of optimal control algorithm. The adjustment algorithm shown comparable result without the drawback of time-consuming execution of optimal control algorithm.

Chapter 7

Conclusion

This thesis starts with the key research question of "*minimising the energy consumption of an electric bus along a bus route while improving the arrival time of the bus at bus stops along the route*". The main conclusion of this thesis is that buses can indeed be controlled to minimise the energy while meeting a required timetable even when there is a certain level of external disturbances such as traffic flow.

The overarching research question is broken into four objectives. This chapter discusses the work done in meeting these objectives, the contributions of this thesis, and the future work that can be carried out for further research.

7.1 The key objectives

The four objectives set out for this thesis are (i) the setting up of an optimal control problem for a bus journey, (ii) the formulation of real-world bus route in the optimisation process, (iii) the estimation of on-road bus speed for open loop optimisation process, and (iv) the setting up of a real-time system to provide and update the optimal speed.

The first objective is the setting up of an optimal control problem for a bus journey. This is addressed in Chapter 3. The focus of the chapter is to incorporate the scheduled time of a bus at different bus stops into the optimal control problem. The aims of the optimal control problem are to (i) minimise the energy consumption of the bus and (ii) minimise the discrepancy between the actual arrival time and the scheduled time. By minimising the energy consumption of a bus journey, the carbon dioxide emissions from energy generation can be reduced. For the purpose of calculating the energy consumption of the bus, a basic electric vehicle model based on OVEM is used. To formulate the scheduled time in the optimal control problem, we need to first understand the desired behaviour of the bus when it arrives at the bus stops. Bus stops are categorised into on-demand bus stops and compulsory bus stops. The fundamental difference between the two types of bus stop is the constraint on the speed of the bus arriving at the bus stop in the computation of the optimal control profile. A bus is required to fully stop for a defined period of time when it arrives at a compulsory bus stop whereas it can be of any speed within the lower and upper boundaries when it arrives at an on-demand bus stop. The stopping of a bus at an on-demand bus stop is governed by the presence of the demand of passengers to alight or board the bus. Compulsory bus stops are formulated into the optimal control problem using multiphase approach. The bus journey is split into different phases at the location of each compulsory bus stop. At the location of each compulsory bus stop, a phase with the defined time period of stopping and zero speed is inserted to sustain the continuity of the resultant profile. On-demand bus stops are formulated as part of the objective function. By

formulating the on-demand bus stops in the objective function, the speed of the bus arriving at the on-demand bus stop is flexible and is identified by the optimal control algorithm.

Another issue addressed in Chapter 3 regards the formulation of the slope of the bus route. The slope of the route is formulated in the optimal control problem as it is one of the inputs for the calculation of the energy consumption. Two approaches to formulate the slope are proposed. The first approach formulates the problem with slope changes as multiphase problem. This allows discontinuities where the slope changes take place. The second approach uses a smooth linear interpolation to approximate the slope profile. A comparison is done to identify the difference of solution time when the two approaches are used in the optimal control problem. The second approach is found to be better than the first approach as (i) the solution time when using the second approach is shorter than using the first approach regardless of the number of slope changes and (ii) the solution time when using the second approach remains almost constant with increasing number of the slope changes. The second reason provides the scalability of the second approach in the number of slope changes.

Three case studies are performed. The first case identifies the optimal profile of a basic route with no bus stops or slope changes. The optimal profile has an energy consumption of 1677.48 kJ. The energy consumption is compared with the profile produced based on an aggressive driving style. The optimal profile provides a reduction of 60% in energy consumption. The second case identifies the optimal profile of a bus route with a compulsory bus stop and two on-demand bus stops. The energy consumption of the resultant

profile is 779.79 kJ, which is 16% more than the energy consumption of the optimal profile in the first case. This is expected since the deceleration and acceleration at the bus stops contribute to more energy consumption. The third case introduces slope changes to the bus route in the second case. The optimal profile in the third case is 795.93 kJ, which is 2.1% more than the energy consumption in the second case. The extra energy consumption is used to overcome the gravitational pull due to the varying elevation along the route.

The second objective is the formulation of real-world bus route in the optimisation process. This is addressed in Chapter 4 which focuses on the methods to translate real-world static information into the format that can be used in an optimal control problem. The real-world static information required in the optimal control problem includes the bus route and the locations of the bus stops on the route. The information of a physical bus route including the geodesic coordinates and the elevation along the route is obtained through the Google Maps API. The distance between successive geodesic coordinates of the bus route is calculated using the cosine-haversine formula and the slope of the road is calculated from the elevation of the road using basic trigonometric formula. As the computational complexity to solve an optimal control problem increases with the number of slope changes along the route, the slope profile is approximated using the proposed approximation algorithm to minimise the number of slope changes while maximising the accuracy of the representation of the elevation profile. The locations of the bus stops are obtained from the Google Maps API as geodesic coordinates. A bus stop fitting algorithm is proposed to identify their positions in relation to the bus route. As the optimal control algorithm always

assumes the bus does not need to stop at an on-demand bus stop. Therefore the method to calculate the stopping probability of a bus at an on-demand bus stop is explored. The stopping probability reflects the actual condition of the demand for the bus to stop at the bus stop and can be used to adjust the optimal profile.

For the purpose of temporal grouping for the calculation of stopping probability, the days of the week are clustered based on the daily profile of the bus speed passing a specific bus stop. The clustering algorithm has grouped the days of the week into four clusters: (i) Sunday, (ii) Monday and Tuesday, (iii) Wednesday to Friday, and (iv) Saturday. The stopping probability of these temporal groups are then calculated by identifying the frequency of the bus stopping at the bus stop.

The third objective is the estimation of on-road bus speed. This is addressed in Chapter 5. The chapter focuses on the estimation of the bus speed on the mixed traffic lane and on the bus lane. The estimation of the bus speed is achieved using two ANNs, one to estimate the speed on the mixed traffic lane and the other to estimate the speed on the bus lane. Different combinations of weather data, social events, and temporal data are used as the inputs of the ANN for the estimation of the speed on the mixed traffic lane to identify the appropriate input set. The ANN with the lowest validation error is found to have the inputs of all three types of data: weather data, social events, and temporal data. The search for the appropriate input set for the ANN to estimate the speed on the bus lane is performed with different combinations of speed data on the corresponding mixed traffic lane, weather data, and social events. The ANN using the input data of only speed data is found to

have the lowest validation error. However, further investigation found a large disparity between the predicted and the actual values of the trained ANN. The examination on the predicted and the actual values of ANNs with different sets of inputs reveals that none of the input sets provide sufficiently accurate prediction. Two hypotheses are made based on this observation. First is the lacking of training data that leads to an under-trained ANN. Second is that the speed on the bus lane is independent of the speed on the mixed traffic lane. More work is required to confirm the hypotheses.

The fourth objective is the setting up of a real-time system to provide and update the optimal speed. This is addressed in Chapter 6. This chapter focuses on the design of an on-board system to provide and adapt the optimal profile based on real-time location of the bus. A Raspberry Pi is chosen as an indicative computational unit of the system for its low cost and the use of general-purpose OS. The Adafruit Ultimate GPS HAT is used for the acquisition of the real-time location of the bus. An algorithm is proposed to identify the position of the bus in relation to the bus route using the acquired real-time location. The optimality of the bus location is identified by comparing the position of the position with the optimal position. The scenarios that lead to the suboptimality of the bus location are (i) the bus is required to stop at an on-demand bus stop, (ii) the bus starts from halt later than scheduled time, and (iii) the speed of the bus is limited by the traffic. To deal with the first scenario, optimal profiles with on-demand bus stops defined as compulsory bus stops are maintained alongside with the original optimal profile. When the first scenario happens, the optimal profile with on-demand bus stops defined as compulsory bus stops

is used in place of the original optimal profile. In second and third scenarios the optimal profile is modified using an adjustment algorithm to ensure the arrival time requirements of the subsequent bus stops hold.

The adjustment algorithm provides near-optimal profiles. In both scenarios that implement the adjustment algorithm, i.e. the second and third scenarios, the energy consumption of the profiles produced by the adjustment algorithm are only 0.23% more than the energy consumption of the profiles produced by the re-execution of the optimal control algorithm. As the adjustment algorithm involves less computation than the optimal control algorithm, the execution time of the adjustment algorithm is much shorter than the optimal control algorithm which allows a real-time update of the profile.

7.2 Contributions

In the opinion of the author, three contributions are made by this thesis:

(i) The problem formulation of a bus journey including two different types of bus stops. As discussed in the review of literature, the problem formulation of an optimal control of a bus with bus stops have not been provided. This thesis has discussed the formulation of the scheduled time of the compulsory and on-demand bus stops into the optimal control problem. The scheduled time of compulsory bus stops is formulated using the multiphase approach whereas the scheduled time of on-demand bus stops is formulated as part of the objective function to be minimised.

(ii) The estimation of the speed on bus lane using speed on the corresponding mixed traffic lane. Speed on bus lane is assumed to be independent from the mixed traffic lane in literature but no supporting proof is provided. With more work, the disparity between the predicted and the actual values of the estimation neural network may serve as the proof of the independence of bus lane speed on the speed on the mixed traffic lane.

(iii) The adjustment algorithm to modify the optimal profile to enable real-time deployment on appropriate engine control units. Based on the real-time location and speed of the bus, the algorithm identifies the next time-constrained location on the route and modifies the appropriate segment of the optimal profile to ensure the requirements of the bus stops are reached. The adjustment algorithm provides a near-optimal result as compared to the re-execution of optimal control algorithm with a shorter execution time.

7.3 Future work

Four areas are identified for potential further work:

(i) The improvement of the model of vehicle. The model of vehicle used in the optimal control of buses is a simplified model of an electric vehicle based on OVEM. This model is useful to show the proof of concept but not able to provide an accurate computation of the energy consumption of the actual bus. Therefore the improvement

of the model of vehicle is necessary to provide a more accurate representation of the bus that can be used in optimal control.

(ii) The inclusion of on-road speed data in the optimal control problem. The optimal control of buses considers bus stops and slope changes but not the on-road speed data. Further work can be done to include this information into the optimal control problem to provide an optimal profile that accounts for the actual traffic situation.

(iii) The investigation for the cause of the disparity between the actual values and the values predicted by the neural network that estimates the bus speed on bus lane using the speed on mixed traffic lane. Two hypotheses are made regarding the disparity. Further work is needed to test and confirm the hypotheses.

(iv) The strategy of handling cases in which the requirements of subsequent bus stops are unachievable. This thesis has only considered the cases with achievable requirements in the formulation of optimal control problem and the adjustment of optimal profiles. Further work can be done to consider cases in which, due to constraints and disruptions, the bus is unable to arrive at the bus stops on the scheduled time.

References

[1] P. Eickemeier, S. Schlömer, E. Farahani, S. Kadner, S. Brunner, I. Baum, and B. Kriemann, "Climate Change 2014: Mitigation of Climate Change (Working Group III: Contribution to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change)," Tech. Rep., 2014. [Online]. Available: https://www.ipcc.ch/report/ar5/wg3/

[2] Department of Transport UK, "Petroleum consumption by transport mode and fuel type, United Kingdom: 2005 to 2015," Department for Transport, UK, Tech. Rep., 2016. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/567895/env0101.ods

[3] ——, "Energy consumption by transport mode and source of energy: United Kingdom, 2000-2015," Department of Transport, UK, Tech. Rep., 2016. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/567896/env0102.ods

[4] Department for Bussiness Energy & Industrial Stratergy, "Energy Consumption in the UK," Department for Bussiness, Energy & Industrial Stratergy, Tech. Rep. July, 2016. [Online]. Available: www.gov.uk/beis

[5] World Health Organization Europe, *Air Quality Guidelines Global Update 2005*, 2005.

[6] Department for Environment Food and Rural Affairs, "Air Pollution in the UK 2015," Defra (Department for Environment, Food & Rural Affairs, UK), Tech. Rep. September, 2016.

[7] T. Scarbrough, M. Vedrenne, A. Fraser, and B. Grebot, "Exploring and appraising proposed measures to tackle air quality," Department for Environment, Food & Rural Affairs, Tech. Rep. May, 2016. [Online]. Available: https://uk-air.defra.gov.uk/assets/documents/reports/cat05/1605120947_AQ0959_summary_report-exploring_and_appraising_measures_to_improve_air_quality.pdf

[8] D. Carslaw and M. Priestman, "Analysis of the vehicle emission remote sensing campaigns data," King's College London, Tech. Rep., 2015. [Online].

Available: https://uk-air.defra.gov.uk/assets/documents/reports/cat15/1511251131_Analysis_of_the_2013_vehicle_emission_remote_sensing_campaigns_data.pdf

[9] S. Carroll, "Green Fleet Technology Study for Public Transport," Cenex, Tech. Rep. February, 2015. [Online]. Available: http://www.cenex.co.uk/wp-content/uploads/2015/02/670_013-2-Technology-Foresighting-Report-_-Final.pdf

[10] H. Banvait, S. Anwar, and C. Yaobin, "A rule-based energy management strategy for Plug-in Hybrid Electric Vehicle (PHEV)," *American Control Conference, 2009. ACC '09.*, pp. 3938–3943, 2009.

[11] L. Xu, M. Ouyang, J. Li, F. Yang, L. Lu, and J. Hua, "Application of Pontryagin's Minimal Principle to the energy management strategy of plugin fuel cell electric vehicles," *International Journal of Hydrogen Energy*, vol. 38, no. 24, pp. 10 104–10 115, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.ijhydene.2013.05.125

[12] S. Moura, H. Fathy, D. Callaway, and J. Stein, "A Stochastic Optimal Control Approach for Power Management in Plug-In Hybrid Electric Vehicles," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 545–555, 2010. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5439900

[13] S. J. Moura, D. S. Callaway, H. K. Fathy, and J. L. Stein, "Tradeoffs between battery energy capacity and stochastic optimal power management in plug-in hybrid electric vehicles," *Journal of Power Sources*, vol. 195, no. 9, pp. 2979–2988, 2010.

[14] M. P. O'Keefe and T. Markel, "Dynamic Programming Applied to Investigate Energy Management Strategies for a Plug-in HEV," *the 22nd International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium and Exhibition (EVS-22)*, no. November, p. 15, 2006.

[15] E. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, 1975.

[16] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[17] M. Kim, Y. J. Sohn, W. Y. Lee, and C. S. Kim, "Fuzzy control based engine sizing optimization for a fuel cell/battery hybrid mini-bus," *Journal of Power Sources*, vol. 178, no. 2, pp. 706–710, 2008.

[18] Q. Li, W. Chen, Y. Li, S. Liu, and J. Huang, "Energy management strategy for fuel cell/battery/ultracapacitor hybrid vehicle based on fuzzy logic," *International Journal of Electrical Power and Energy Systems*, vol. 43, no. 1, pp. 514–525, 2012.

[19] H. Hemi, J. Ghouili, and A. Cheriti, "A real time fuzzy logic power management strategy for a fuel cell vehicle," *Energy Conversion and Management*, vol. 80, pp. 63–70, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.enconman.2013.12.040

[20] L. Xu, J. Li, and M. Ouyang, "Energy flow modeling and real-time control design basing on mean values for maximizing driving mileage of a fuel cell bus," *International Journal of Hydrogen Energy*, pp. 1–15, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S036031991502251X

[21] G. Welch and G. Bishop, "An introduction to the Kalman filter," 1995.

[22] R. E. Bellman, "The Theory of Dynamic Programming," pp. 503–515, 1954.

[23] B. C. Chen, Y. Y. Wu, and H. C. Tsai, "Design and analysis of power management strategy for range extended electric vehicle using dynamic programming," *Applied Energy*, vol. 113, pp. 1764–1774, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.apenergy.2013.08.018

[24] L. S. Pontryagin, *Mathematical Theory of Optimal Processes*, ser. Classics of Soviet Mathematics. Taylor & Francis, 1987. [Online]. Available: https://books.google.co.uk/books?id=kwzq0F4cBVAC

[25] N. Kim and a. Rousseau, "Sufficient conditions of optimal control based on Pontryagin's minimum principle for use in hybrid electric vehicles," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 226, no. 9, pp. 1160–1170, 2012.

[26] S. Zhang, R. Xiong, and C. Zhang, "Pontryagin's Minimum Principle-based power management of a dual-motor-driven electric bus," *Applied Energy*, vol. 159, pp. 370–380, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S030626191501065X

[27] N. Kim, S. Cha, and H. Peng, "Optimal control of hybrid electric vehicles based on Pontryagin's minimum principle," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 5, pp. 1279–1287, 2011.

[28] M. J. Sullman, L. Dorn, and P. Niemi, "Eco-driving training of professional bus drivers - Does it work?" *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 749–759, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X15001515

[29] C. Rolim, P. Baptista, G. Duarte, T. Farias, and Y. Shiftan, "Quantification of the Impacts of Eco-driving Training and Real-time Feedback on Urban Buses Driver's Behaviour," *Transportation Research Procedia*, vol. 3, no. July,

pp. 70–79, 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S2352146514002555

[30] M. Barth and K. Boriboonsomsin, "Energy and emissions impacts of a freeway-based dynamic eco-driving system," *Transportation Research Part D: Transport and Environment*, vol. 14, no. 6, pp. 400–410, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.trd.2009.01.004

[31] E. Gilman, A. Keskinarkaus, S. Tamminen, S. Pirttikangas, J. Röning, and J. Riekki, "Personalised assistance for fuel-efficient driving," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 681–705, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X15000480

[32] D. Chakraborty, W. Vaz, and A. K. Nandi, "Optimal driving during electric vehicle acceleration using evolutionary algorithms," *Applied Soft Computing*, vol. 34, pp. 217–235, 2015. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84930619697&partnerID=tZOtx3y1

[33] A. K. Nandi, D. Chakraborty, and W. Vaz, "Design of a comfortable optimal driving strategy for electric vehicles using multi-objective optimization," *Journal of Power Sources*, vol. 283, pp. 1–18, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378775315003547

[34] P. Pudney and P. Howlett, "Optimal Driving Strategies for a Train Journey With Speed Limits," *Journal of the Australian Mathematical Society Series B-Applied Mathematics*, vol. 36, no. 1, pp. 38–49, 1994.

[35] V. De Martinis and M. Gallo, "Models and Methods to Optimise Train Speed Profiles with and without Energy Recovery Systems: A Suburban Test Case," *Procedia - Social and Behavioral Sciences*, vol. 87, pp. 222–233, 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877042813040512

[36] J. Hooker, "Optimal driving for single-vehicle fuel economy," *Transportation Research Part A: General*, vol. 22, no. 3, pp. 183–201, 1988. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0191260788900362

[37] A. Schwarzkopf and R. Leipnik, "Control of highway vehicles for minimum fuel consumption over varying terrain," *Transportation Research*, vol. 11, no. 4, pp. 279–286, 1977. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0041164777900934

[38] V. V. Monastyrsky and I. M. Golownykh, "Rapid computation of optimal control for vehicles," *Transportation Research Part B: Methodological*, vol. 27, no. 3, pp. 219–227, 1993. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0191261593900315

[39] Y. Saboohi and H. Farzaneh, "Model for developing an eco-driving strategy of a passenger vehicle based on the least fuel consumption," *Applied Energy*, vol. 86, no. 10, pp. 1925–1932, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.apenergy.2008.12.017

[40] Y. Zhou, S. Ou, J. Lian, and L. Li, "Optimization of Hybrid Electric Bus Driving System's Control Strategy," *Procedia Engineering*, vol. 15, pp. 240–245, 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877705811015499

[41] M. Chen, J. Yaw, S. I. Chien, and X. Liu, "Using Automatic Passenger Counter Data in Bus Arrival Time Prediction," *Journal of Advanced Transportation*, vol. 41, no. 3, pp. 267–283, jun 2007. [Online]. Available: http://doi.wiley.com/10.1002/atr.5670410304

[42] B. Yu, W. H. Lam, and M. L. Tam, "Bus arrival time prediction at bus stop with multiple routes," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 6, pp. 1157–1170, 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X11000155

[43] A. Tirachini, "Estimation of travel time and the benefits of upgrading the fare payment technology in urban bus services," *Transportation Research Part C: Emerging Technologies*, vol. 30, pp. 239–256, 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X11001598

[44] R. Padmanaban, K. Divakar, L. Vanajakshi, and S. Subramanian, "Development of a real-time bus arrival prediction system for Indian traffic conditions," *IET Intelligent Transport Systems*, vol. 4, no. 3, p. 189, 2010. [Online]. Available: http://digital-library.theiet.org/content/journals/10.1049/iet-its.2009.0079

[45] X. Zhang and J. a. Rice, "Short-term travel time prediction," *Transportation Research Part C: Emerging Technologies*, vol. 11, no. 3-4, pp. 187–210, 2003. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X03000263

[46] J. van Lint, S. Hoogendoorn, and H. van Zuylen, "Accurate freeway travel time prediction with state-space neural networks under missing data," *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 5-6, pp. 347–369, 2005. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X0500029X

[47] a. Jiménez-Meza, J. Arámburo-Lizárraga, and E. de la Fuente, "Framework for Estimating Travel Time, Distance, Speed, and Street Segment Level of Service (LOS), based on GPS Data," *Procedia Technology*, vol. 7, pp. 61–70, 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S2212017313000091

[48] W.-H. Lee, S.-S. Tseng, and S.-H. Tsai, "A knowledge based real-time travel time prediction system for urban network," *Expert Systems with Applications*, vol. 36, no. 3, pp. 4239–4247, 2009. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0957417408001875

[49] M. Chen, X. Liu, J. Xia, and S. I. Chien, "A Dynamic Bus-Arrival Time Prediction Model Based on APC Data," *Computer-Aided Civil and Infrastructure Engineering*, vol. 19, no. 5, pp. 364–376, 2004. [Online]. Available: http://doi.wiley.com/10.1111/j.1467-8667.2004.00363.x

[50] Q. Meng and X. Qu, "Bus dwell time estimation at bus bays: A probabilistic approach," *Transportation Research Part C: Emerging Technologies*, vol. 36, pp. 61–71, 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X1300171X

[51] C. Zhang and J. Teng, "Bus Dwell Time Estimation and Prediction: A Study Case in Shanghai-China," *Procedia - Social and Behavioral Sciences*, vol. 96, no. Cictp, pp. 1329–1340, 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877042813022775

[52] Yidan Fan, Kun Niu, and Nanjie Deng, "A real-time bus arrival prediction method based on energy-efficient cell-tower positioning," in *2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems*. IEEE, nov 2014, pp. 717–721. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7175827

[53] X. Song, J. Teng, G. Chen, and Q. Shu, "Predicting Bus Real-time Travel Time Basing on both GPS and RFID Data," *Procedia - Social and Behavioral Sciences*, vol. 96, no. Cictp, pp. 2287–2299, nov 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877042813023847

[54] H. Feng and Y. Shu, "Study on network traffic prediction techniques," in *Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.*, vol. 2. IEEE, sep 2005, pp. 995–998. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1544219

[55] S. Clark, "Traffic Prediction Using Multivariate Nonparametric Regression," *Journal of Transportation Engineering*, vol. 129, no. 2, pp. 161–168, mar 2003. [Online]. Available: http://ascelibrary.org/doi/abs/10.1061/(ASCE)0733-947X(2003)129:2(161)

[56] B. Pan, U. Demiryurek, and C. Shahabi, "Utilizing Real-World Transportation Data for Accurate Traffic Prediction," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, dec 2012, pp. 595–604.

[57] B. L. Smith and M. J. Demetsky, "Traffic Flow Forecasting: Comparison of Modeling Approaches," *Journal of Transportation Engineering*, vol. 123, no. 4, pp. 261–266, jul 1997. [Online]. Available: http://ascelibrary.org/doi/10.1061/%28ASCE%290733-947X%281997%29123%3A4%28261%29

[58] C. Quek, M. Pasquier, B. Boon, and S. Lim, "POP-TRAFFIC : A Novel Fuzzy Neural Approach to Road Traffic Analysis and Prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 2, pp. 133–146, 2006. [Online]. Available: https://www.researchgate.net/profile/Chai_Quek/publication/3427965_POP-TRAFFIC_A_novel_fuzzy_neural_approach_to_road_traffic_analysis_and_prediction/links/09e4150b6bf7929ad2000000.pdf

[59] R. Yasdi, "Prediction of Road Traffic using a Neural Network Approach," *Neural Computing & Applications*, vol. 8, no. 2, pp. 135–142, 2014. [Online]. Available: http://link.springer.com/10.1007/s005210050015

[60] J. Hopfield, "Artificial neural networks," *IEEE Circuits and Devices Magazine*, vol. 4, no. 5, pp. 3–10, sep 1988. [Online]. Available: http://ieeexplore.ieee.org/ielx1/101/428/00008118.pdf

[61] J. Gibson, M. A. Munizaga, C. Schneider, and A. Tirachini, "Estimating the bus user time benefits of implementing a median busway: Methodology and case study," *Transportation Research Part A: Policy and Practice*, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0965856415001998

[62] B. Yu and X. Kun, "Discussion about Evaluation Method of Traffic Efficiency Adaptability of Bus Lane," *2009 Second International Conference on Intelligent Computation Technology and Automation*, pp. 566–570, 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5288040

[63] L. J. Basso, C. A. Guevara, A. Gschwender, and M. Fuster, "Congestion pricing, transit subsidies and dedicated bus lanes: Efficient and practical solutions to congestion," *Transport Policy*, vol. 18, no. 5, pp. 676–684, sep 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0967070X1100014X

[64] Y. Chen, G. Chen, and K. Wu, "Evaluation of Performance of Bus Lanes on Urban Expressway Using Paramics Micro-simulation Model," *Procedia Engineering*, vol. 137, pp. 523–530, 2016. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1877705816003155

[65] H. J. Kim, "Performance of Bus Lanes in Seoul," *IATSS Research*, vol. 27, no. 2, pp. 36–45, 2003. [Online]. Available: http://dx.doi.org/10.1016/S0386-1112(14)60142-4http://linkinghub.elsevier.com/retrieve/pii/S0386111214601424

[66] J. Viegas and B. Lu, "Widening the scope for bus priority with intermittent bus lanes," *Transportation Planning and Technology*, vol. 24, no. 2, pp. 87–110, 2001. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-0034443327&partnerID=tZOtx3y1

[67] H. Zhu, "Numerical study of urban traffic flow with dedicated bus lane and intermittent bus lane," *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 16, pp. 3134–3139, 2010. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0378437110002852

[68] M. Eichler and C. F. Daganzo, "Bus lanes with intermittent priority: Strategy formulae and an evaluation," *Transportation Research Part B: Methodological*, vol. 40, no. 9, pp. 731–744, 2006.

[69] J. Viegas and B. Lu, "The Intermittent Bus Lane signals setting within an area," *Transportation Research Part C: Emerging Technologies*, vol. 12, no. 6, pp. 453–469, 2004.

[70] V. Zyryanov and A. Mironchuk, "Simulation Study of Intermittent Bus Lane and Bus Signal Priority Strategy," *Procedia - Social and Behavioral Sciences*, vol. 48, pp. 1464–1471, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877042812028583

[71] O. Bolza, *Lectures On The Calculus Of Variations*, 1904.

[72] D. Liberzon, "Calculus of Variations and Optimal Control Theory: A Concise Introduction," pp. 1—-190, 2011.

[73] A. V. Rao, "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, vol. 135, no. 1, pp. 497–528, 2009. [Online]. Available: http://vdol.mae.ufl.edu/ConferencePublications/trajectorySurveyAAS.pdf

[74] O. Stryk and R. Bulirsch, "Direct and indirect methods for trajectory optimization," *Annals of Operations Research*, vol. 37, no. 1, pp. 357–373, dec 1992. [Online]. Available: http://dx.doi.org/10.1007/BF02071065http://link.springer.com/10.1007/BF02071065

[75] C. Büskens and H. Maurer, "SQP-methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real-time control," *Journal of Computational and Applied Mathematics*, vol. 120, no. 1-2, pp. 85–108, aug 2000. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0377042700003058

[76] Y. Wang, Y. Zhao, and S. A. Bortoff, "A numerical algorithm to solve a class of multi-point boundary value problems," vol. 1, no. 2, pp. 2262–2268, 2013.

[77] V. M. Becerra, "Solving complex optimal control problems at no cost with PSOPT," *Proceedings of the IEEE International Symposium on Computer-Aided Control System Design*, pp. 1391–1396, 2010.

[78] A. V. Rao, D. a. Benson, C. Darby, M. a. Patterson, C. Francolin, I. Sanders, and G. T. Huntington, "GPOPS-II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," *ACM Transactions on Mathematical Software*, vol. 37, no. 2, pp. 1–39, 2010. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1731022.1731032

[79] J. Akesson, K.-E. Årzén, M. Gäfvert, T. Bergdahl, and H. Tummescheit, "Modeling and Optimization with Optimica and JModelica.org- languages and tools for solving large-scale dynamic optimization problems," *Comput. Chem. Eng.*, vol. 34, pp. 1737–1749, 2010.

[80] J. D. Hedengren, R. A. Shishavan, K. M. Powell, and T. F. Edgar, "Nonlinear modeling, estimation and predictive control in APMonitor," *Computers & Chemical Engineering*, vol. 70, pp. 133–148, nov 2014. [Online]. Available: http://dx.doi.org/10.1016/j.compchemeng.2014.04.013http://linkinghub.elsevier.com/retrieve/pii/S0098135414001306

[81] F. Bonnans, V. Grelard, and P. Martinon, "Bocop, the optimal control solver, open source toolbox for optimal control problems," *URL http://bocop. org*, 2011.

[82] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, mar 2006. [Online]. Available: http://link.springer.com/10.1007/s10107-004-0559-y

[83] R. Lougee-Heimer, "The Common Optimization INterface for Operations Research: Promoting open-source software in the operations research community," *IBM Journal of Research and Development*, vol. 47, no. 1, pp. 57–66, 2003.

[84] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Journal on Optimization*, vol. 12, no. 4, pp. 979–1006, 2002. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/S1052623499350013

[85] R. H. Byrd, J. Nocedal, and R. A. Waltz, "Knitro: An Integrated Package for Nonlinear Optimization," in *Large-Scale Nonlinear Optimization*, G. Di Pillo and M. Roma, Eds. Boston, MA: Springer US, 2006, pp. 35–59. [Online]. Available: http://dx.doi.org/10.1007/0-387-30065-1_4

[86] Artelys, "Artelys Knitro - Nonlinear optimization solver." [Online]. Available: https://www.artelys.com/en/optimization-tools/knitro

[87] Advanced Process Solutions, "APOPT." [Online]. Available: http://apopt.com/

[88] M. Rahmani Asl, S. Zarrinmehr, M. Bergin, and W. Yan, "BPOpt: A framework for BIM-based performance optimization," *Energy and Buildings*, vol. 108, pp. 401–412, dec 2015. [Online]. Available: http://dx.doi.org/10.1016/j.enbuild.2015.09.011http://linkinghub.elsevier.com/retrieve/pii/S0378778815302528

[89] Hilding Elmqvist, "Modelica - A unified object-oriented language for physical systems modeling," *Simulation Practice and Theory*, vol. 5, no. 6, p. p32, 1997. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0928486997842577

[90] J. Åkesson, "Optimica: an extension of modelica supporting dynamic optimization," *6th International Modelica Conference 2008*, pp. 57–66, 2008.

[91] R. Doucette, "The Oxford Vehicle Model : A tool for modeling and simulating the powertrains of electric and hybrid electric vehicles," Ph.D. dissertation, University of Oxford, 2012.

[92] T. D. Gillespie, *Fundamentals of Vehicle Dynamics*. Society of Automotive Engineers, 1992. [Online]. Available: http://books.google.co.uk/books?id=L6xd0nx5KbwC

[93] N. Mohan, T. M. Undeland, and W. P. Robbins, *Power electronics: converters, applications, and design*. Wiley, 1995. [Online]. Available: http://books.google.co.uk/books?id=KbkQAQAAMAAJ

[94] O. Tremblay, L.-A. Dessaint, and A.-I. Dekkiche, "A Generic Battery Model for the Dynamic Simulation of Hybrid Electric Vehicles," in *2007 IEEE Vehicle Power and Propulsion Conference*, no. V. IEEE, sep 2007, pp. 284–289. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4544139

[95] T. Henzinger, "The theory of hybrid automata," in *Proceedings 11th Annual IEEE Symposium on Logic in Computer Science*, vol. 2, no. Lics 96. IEEE Comput. Soc. Press, 2000, pp. 278–292. [Online]. Available: http://ieeexplore.ieee.org/document/561342/

[96] D. Corona, A. Giua, and C. Seatzu, "Optimal control of hybrid automata : design of a semiactive suspension," vol. 12, pp. 1305–1318, 2004.

[97] M. Branicky, V. Borkar, and S. Mitter, "A unified framework for hybrid control: model and optimal control theory," *IEEE Transactions on Automatic Control*, vol. 43, no. 1, pp. 31–45, 1998. [Online]. Available: http://ieeexplore.ieee.org/document/654885/

[98] E. Hewitt and R. E. Hewitt, "The Gibbs-Wilbraham phenomenon: An episode in fourier analysis," *Archive for History of Exact Sciences*, vol. 21, no. 2, pp. 129–160, 1979.

[99] V. M. Becerra, "User Manual for PSOPT," Tech. Rep., 2015. [Online]. Available: https://github.com/PSOPT/psopt/blob/master/PSOPT/doc/PSOPT_Manual_R4.pdf

[100] UK Government Digital Service, "Speed limits." [Online]. Available: https://www.gov.uk/speed-limits

[101] M. Kirchner, P. Schubert, and C. T. Haas, "Characterisation of Real-World Bus Acceleration and Deceleration Signals," *Journal of Signal and Information Processing*, vol. 2014, no. February, pp. 8–13, 2014.

[102] K. Shoemake, "Animating rotation with quaternion curves," *ACM SIGGRAPH Computer Graphics*, vol. 19, no. 3, pp. 245–254, 1985. [Online]. Available: http://portal.acm.org/citation.cfm?doid=325165.325242

[103] C. C. Robusto, "The Cosine-Haversine Formula," *The American Mathematical Monthly*, vol. 64, no. 1, p. 38, jan 1957. [Online]. Available: http://www.jstor.org/stable/2309088?origin=crossref

[104] C. Veness, "Calculate distance, bearing and more between Latitude/Longitude points." [Online]. Available: http://www.movable-type.co.uk/scripts/latlong.html

[105] T. Vincenty, "Direct and Inverse Solutions of Geodesics on the Ellipsoid with application of nested equations," *Survey Review*, vol. 33, no. 176, pp. 88–93, 1975.

[106] NIMA, "Department of Defense World Geodetic System 1984," Tech. Rep., 2000. [Online]. Available: http://earth-info.nga.mil/GandG/publications/tr8350.2/wgs84fin.pdf

[107] Oxfordshire County Council, "Oxfordshire County Council." [Online]. Available: http://voyager.oxfordshire.gov.uk/map.aspx

[108] A. F. Clements, M. D. McCulloch, and K. J. Nixon, "Low-loss, high-compression of energy profiles," *2015 International Conference on Renewable Energy Research and Applications, ICRERA 2015*, vol. 5, pp. 949–953, 2016.

[109] J. Macqueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 233, pp. 281–297, 1967.

[110] "OpenWeatherMap." [Online]. Available: https://openweathermap.org/

[111] S. Aljahdali and A. Sheta, "Prediction of Software Reliability: A Comparison Between Regression and Neural Network Non-Parametric Models," *Computer Systems and Applications*, no. i, pp. 470–473, 2001. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=934046

[112] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: A methodology review," *Journal of Biomedical Informatics*, vol. 35, no. 5-6, pp. 352–359, 2002.

[113] K. J. Ottenbacher, R. T. Linn, P. M. Smith, S. B. Illig, M. Mancuso, and C. V. Granger, "Comparison of logistic regression and neural network analysis applied to predicting living setting after hip fracture," *Annals of Epidemiology*, vol. 14, no. 8, pp. 551–559, 2004.

[114] M. Paliwal and U. A. Kumar, "Neural networks and statistical techniques: A review of applications," *Expert Systems with Applications*, vol. 36, no. 1, pp. 2–17, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2007.10.005

[115] M. A. Razi and K. Athappilly, "A comparative predictive analysis of neural networks (NNs), nonlinear regression and classification and regression tree (CART) models," *Expert Systems with Applications*, vol. 29, no. 1, pp. 65–74, 2005.

[116] W. S. Sarle, "Neural Networks and Statistical Models," *Proceedings of the Nineteenth Annual SAS Users Group International Conference, April, 1994*, pp. 1–13, 1994.

[117] A. E. SMITH and A. K. MASON, "Cost Estimation Predictive Modeling: Regression Versus Neural Network," *The Engineering Economist*, vol. 42, no. 2, pp. 137–161, 1997. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/00137919708903174

[118] J. V. Tu, "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes," *Journal of Clinical Epidemiology*, vol. 49, no. 11, pp. 1225–1231, 1996.

[119] R. E. Kalman and Others, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[120] R. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Magazine*, vol. 4, no. 2, pp. 4–22, 1987. [Online]. Available: http://ieeexplore.ieee.org/document/1165576/

[121] A. Jain, Jianchang Mao, and K. Mohiuddin, "Artificial neural networks: a tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, mar 1996. [Online]. Available: http://ieeexplore.ieee.org/document/485891/

[122] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, jan 1989. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/0893608089900208

[123] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991.

[124] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson Correlation Coefficient," in *Noise Reduction in Speech Processing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1–4. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-00296-0_5

[125] M. I. Jordan, "Serial order: A parallel distributed processing approach," pp. 471–495, 1986. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0166411597801112%5Cnhttp://www.sciencedirect.com/science/article/pii/S0166411597801112

[126] J. L. Elman, "Finding Structure in Time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, mar 1990. [Online]. Available: http://doi.wiley.com/10.1207/s15516709cog1402_1

[127] T.-L. Ramazan-Gencay, "Nonlinear modelling and prediction with feedforward and recurrent networks," *Physica D*, vol. 108, pp. 119–134, 1997.

[128] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A Generalized Growing and Pruning RBF (GGAP-RBF) Neural Network for Function Approximation," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 57–67, jan 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1388458/

[129] J. Zhang and A. J. Morris, "A Sequential Learning Approach for Single Hidden Layer Neural Networks." *Neural networks : the official journal of the International Neural Network Society*, vol. 11, no. 1, pp. 65–80, 1998. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/12662849

[130] H. C. Yuan, F. L. Xiong, and X. Y. Huai, "A method for estimating the number of hidden neurons in feed-forward neural networks based on information entropy," *Computers and Electronics in Agriculture*, vol. 40, no. 1-3, pp. 57–64, 2003.

[131] K. G. Sheela and S. N. Deepa, "Review on Methods to Fix Number of Hidden Neurons in Neural Networks," *Mathematical Problems in Engineering*, vol. 2013, pp. 1–11, 2013. [Online]. Available: http://www.hindawi.com/journals/mpe/2013/425740/

[132] J.-Y. Li, T. Chow, and Y.-L. Yu, "The estimation theory and optimization algorithm for the number of hidden units in the higher-order feedforward neural network,"

Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 3, 1995.

[133] S. S. Haykin, *Neural Networks: A Comprehensive Foundation*. Pearson Education, Inc., feb 1999.

[134] S. Xu and L. Chen, "A novel approach for determining the optimal number of hidden layer neurons for FNN's and its application in data mining," *5th International Conference on Information Technology and Applications (ICITA)*, no. Icita, pp. 683–686, 2008. [Online]. Available: http://eprints.utas.edu.au/6995/

[135] Z. Zhang, X. Ma, and Y. Yang, "Bounds on the number of hidden neurons in three-layer binary neural networks," *Neural Networks*, vol. 16, no. 7, pp. 995–1002, sep 2003. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0893608003000066

[136] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: four layers versus three," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 251–255, mar 1997. [Online]. Available: http://ieeexplore.ieee.org/document/557662/

[137] K. Jinchuan and L. Xinzhe, "Empirical analysis of optimal hidden neurons in neural network modeling for stock prediction," *Proceedings - 2008 Pacific-Asia Workshop on Computational Intelligence and Industrial Application, PACIIA 2008*, vol. 2, pp. 828–832, 2008.

[138] D. Hunter, H. Yu, M. S. Pukish, J. Kolbusz, and B. M. Wilamowski, "Selection of Proper Neural Network Sizes and Architectures - A Comparative Study," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 2, pp. 228–240, 2012.

[139] O. Fujita, "Statistical estimation of the number of hidden units for feedforward neural networks." *Neural networks : the official journal of the International Neural Network Society*, vol. 11, pp. 851–859, 1998. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/12662787

[140] M. J. Kearns, "A Bound on the Error of Cross Validation Using the Approximation and Estimation Rates, with Consequences for the Training-Test Split," *Neural Computation*, vol. 9, no. 5, pp. 1143–1161, 1997.

[141] P. S. Crowther and R. J. Cox, "A Method for Optimal Division of Data Sets for Use in Neural Networks," in *Knowledge-Based Intelligent Information and Engineering Systems*, 2005, vol. 20, pp. 1–7. [Online]. Available: http://link.springer.com/10.1007/11554028_1

[142] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[143] Hecht-Nielsen, "Theory of the backpropagation neural network," in *International Joint Conference on Neural Networks*, vol. 1. IEEE, 1989, pp. 593–605 vol.1. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?&arnumber=118638http://ieeexplore.ieee.org/document/118638/

[144] R. Rojas, *Neural Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, vol. 7, no. 1. [Online]. Available: http://page.mi.fu-berlin.de/rojas/neural/index.html.htmlhttp://link.springer.com/10.1007/978-3-642-61068-4

[145] T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber, "PyBrain," *Journal of Machine Learning Research*, vol. 11, pp. 743–746, 2010.

[146] GlobalTop Technology Inc., "FGPMMOPA6H GPS Standalone Module Data Sheet," no. 16, pp. 1–37, 2011.

[147] SK Pang ELectronics Ltd, "PiCAN 2 SMPS User Guide," pp. 1–9.

[148] NMEA, *NMEA 0183–Standard for interfacing marine electronic devices*. NMEA, 2002. [Online]. Available: http://www.nmea.org