



TECHNISCHE UNIVERSITÄT MÜNCHEN
Department of INFORMATICS

MASTER'S THESIS
in AUTOMOTIVE SOFTWARE ENGINEERING

**Driving Profile and Energy Demand
Analysis for Electrical Vehicles based on
GPS Trajectories**

**Fahrprofil- und Energiebedarfsanalyse für
Elektrische Fahrzeuge basierend auf GPS
Trajektorien**

Author: Sascha MOECKER

Supervisor: Prof. Dr. Thomas HAMACHER

Advisor: Dipl.-Phys. Reinhard SELLMAIR

Submission date: 15. November 2014



Declaration of Authorship

I, Sascha MOECKER, confirm that this master's thesis titled '*Driving Profile and Energy Demand Analysis for Electrical Vehicles based on GPS Trajectories*' ('*Fahrprofil- und Energiebedarfsanalyse für Elektrische Fahrzeuge basierend auf GPS Trajektorien*') is my own work and I have documented all sources and material used.

Contact: sascha.moecker@live.de

Signed: _____

Date: _____

*“There are two ways of constructing a software design:
One way is to make it so simple that there are obviously no deficiencies,
and the other is to make it so complicated that there are no obvious deficiencies.
The first method is far more difficult.”*

- C.A.R. Hoare

TECHNISCHE UNIVERSITÄT MÜNCHEN
Department of INFORMATICS

MASTER'S THESIS
in AUTOMOTIVE SOFTWARE ENGINEERING

Abstract

Driving Profile and Energy Demand Analysis for Electrical Vehicles based on GPS Trajectories

by Sascha MOECKER

A subject of great interest for energy providers is the influence that electrical vehicles (EVs) will have on the electric grid of a city, how to cope with the emerging demand, and how to plan a sufficient charging infrastructure. The goal of this work is to evaluate the extra energy demand emerging through the introduction of EVs by analysing their driving patterns in a certain city using Singapore as an example [1]. The obtained results will be incorporated into a comprehensive taxi simulation aiming at optimising charging infrastructure in Singapore.

Two analysis tasks are conducted: driving profile and energy demand analysis. The driving profile unit focuses on raw GPS trajectories obtained from GPS loggers and incorporates various processing steps like filtering, map-matching, division into trips subject to vehicle's status, route reconstruction, and statistical evaluation of driving patterns. The energy demand part aims at estimating energy consumptions of EVs for various sampling rates and various trajectory or route resolutions, suitable for a simulation context. It comprises three approaches: a static, a dynamic and an instant one of which the two first mentioned include a conceptualization of energetic databases established upon historical GPS trajectories.

The driving profile analysis reveals insightful statistical evaluations combining SMRT's logging and booking data with LTA and auxiliary data sources. The performed pre-processing and map-matching steps prepare the trajectories successfully for a satisfying energy estimation. Accordingly, the implemented energy analysis approaches perform well under given conditions employing three concepts: a static energy map of Singapore, a dynamic driving share approach, and an instant driving feature approach. Each is capable of predicting energy at the detail level or road segments.

An evaluation of energetic results against the comprehensive TUM CREATE EV Model reveals considerably compliance. The dynamic driving share approach performs best with an average deviation of $\pm 14\%$ for the best 90% trips due to its most flexible, driving dynamics anticipating, and in-detail design. The static energy map approach yields similar findings with an average deviation of $\pm 18\%$, but lacks in incorporating particular driving and traffic conditions as it merely relies on average energy values per road segment. The instant driving feature approach, leveraging fewest historical data inputs, produces a mean deviation of $\pm 20\%$ but is rather applicable for high resolution trajectories and thus inappropriate for the targeted simulation purpose.

Acknowledgements

I would like to thank the entire TUM CREATE team for their warm welcome in the first days in Singapore, the collaborative and motivating work atmosphere during my stay, and helpful advises while writing this thesis. In addition, I very appreciated the ability to gain insights into various projects conducted by open minded people in a multi-cultural environment like Singapore. Particularly, the huge group of students and research associates with different educational backgrounds encouraged me - on the technical side - to think out of the box, and - on the personal side - made my stay pleasant regarding activities and excursions.

Special thanks go to my supervisor Prof. Dr. Thomas Hamacher from the TUM energy economy chair, who was willing to supervise this thesis although based at the department for informatics, as well as to my advisers Dipl.-Phys. Reinhard Sellmair and Dr. Tobias Massier for their great commitment and support during the creation of this thesis. The weekly discussions providing continuous feedback, bi-weekly Skype conferences with Munich, and bi-weekly RP 8 meetings supported me to keep track of my work and to focus on the most important tasks. Moreover, their profound knowledge in their fields and their gratitude to share their research insights was particular helpful while writing this thesis.

In addition, my appreciation goes towards the external partner SMRT for their openness to provide valuable taxi data for this present research purpose in form of logging and booking data and for the allowance to install GPS logger devices into their vehicles. The feedback in monthly update meetings supported the quality of the data analysis, and worthwhile feedback has been incorporated into this work. The data has great potential - not just as utilized in this work but also for further related projects at TUM CREATE.

Last but not least, a great opportunity like writing the master's thesis abroad and being absent from home for more than half a year, demands understanding by family and friends, which I hereby want to acknowledge - saying thank you for your support and for sticking to my ambitions.

Contents

Declaration of Authorship	i
Abstract	iii
Acknowledgements	iv
Contents	v
List of Figures	viii
List of Tables	xi
Abbreviations	xii
Physical Constants	xiii
Symbols	xiv
1 Introduction	1
1.1 Thesis Goal and Integration into Current Research	1
1.2 Potential and Impact of Electrical Vehicles	2
1.3 Particularities of Environment and Taxi System	3
1.4 Thesis Structure	3
2 Related Work	5
2.1 Driving Profile Analysis	5
2.2 Energy Demand Analysis	6
2.2.1 Generic Approaches	6
2.2.2 Specific Applications	6
3 Data Sources	8
3.1 Taxi Data	8
3.1.1 Taxi Logging Data (GPS Trajectories)	9
3.1.2 Taxi Booking Data (Status Data)	11
3.2 Auxiliary Data	13
3.2.1 Singapore Road Network Source and Preparation	13
3.2.2 Infrastructure and Taxi Data	18
3.2.3 Spatial and Geographical Data	20

4	Driving Profile Analysis - Methodology	23
4.1	Data Import and Handling	24
4.2	Trajectory Pre-Processing	26
4.2.1	Combination of Logging and Booking Data	26
4.2.2	GPS Trajectory Filtering	28
4.2.3	GPS Trajectory Interpolation	32
4.3	Trip and Shift Extraction	35
4.3.1	Trip Extraction	35
4.3.2	Shift Extraction	40
4.4	Map-Matching	42
4.4.1	Map-Matching ST Algorithm	43
4.4.2	Application of Map-Matching on recorded GPS Trajectories	46
4.5	Micro Trips and Driving Features	48
4.5.1	Micro Trip Extraction	48
4.5.2	Driving Feature Extraction and Clustering	49
4.5.3	Driving Feature Classification	50
4.6	Statistical Analysis	52
4.6.1	Logging Data based Statistics	52
4.6.2	Booking Data based Statistics	53
5	Driving Profile Analysis - Results and Evaluation	55
5.1	Pre-Processing Evaluation	55
5.1.1	Dataset Combination Evaluation	55
5.1.2	Filtering and Interpolation Evaluation	56
5.2	Map-Matching Evaluation	61
5.3	Dataset Statistics	62
5.3.1	Speed Profile	63
5.3.2	Booking Status Distribution	65
5.3.3	Mileage and Distance Analysis	66
5.3.4	Region related Analysis	68
5.3.5	Cluster Analysis	70
5.4	Comparison and Validation with LTA Data	71
6	Energy Demand Analysis - Methodology	72
6.1	Electric Vehicle Models	73
6.1.1	TUM CREATE EV Model	75
6.1.2	Backward Model	75
6.1.3	Model Tailoring and Enhancements	76
6.2	Static Energy Map Approach	76
6.2.1	Energy Map Construction	77
6.2.2	Energy Map Application	81
6.3	Dynamic Driving Share Approach	83
6.3.1	Driving Share Approach Basics	83
6.3.2	Driving Shares for Energy Estimation	86
6.3.3	Driving Share Application	89
6.4	Instant Approaches	91
6.4.1	Instant Driving Share Approach	91

6.4.2	Instant Driving Feature Approach	92
7	Energy Demand Analysis - Results and Evaluation	93
7.1	Comparison of Energy Models	94
7.2	Environment and Scope of Evaluation	95
7.2.1	Evaluation Environment	96
7.2.2	Application Set-Up	98
7.3	Evaluation Results	99
7.3.1	Results for Energy Map Approach	100
7.3.2	Results for Driving Share Approach	102
7.3.3	Results for Instant Approaches	105
8	Conclusion and Outlook	106
A	Logging Data Example	108
B	Booking Data Example	109
C	Taxi Database Structure	110
D	Shift Set Structure	111
E	Additional Statistics	113
F	Shift Set Statistics	116
G	GPS Logger Manual	118
H	Program Parameters	120
I	Matlab Data Folder	123
J	Matlab Source Folder	124
K	Regions, Planning Areas, and Subzones in Singapore	127
L	Taxi Stands in Singapore	128
M	Matlab Code Excerpts	133
	Bibliography	157

List of Figures

1.1	Thesis integration into RP 8 research topics	1
3.1	Columbus V-990 GPS logger	10
3.2	Example of a recorded trip and a detailed excerpt	11
3.3	Comparison of speed values from CAN Bus and GPS logger	11
3.4	Booking data example showing positions and statuses	13
3.5	OSM definitions for <i>Node</i> , <i>Way</i> , <i>Road Segment</i> , <i>Link</i> and <i>Intersection</i>	14
3.6	Unfiltered and filtered OSM excerpt	15
3.7	Class diagram for processing OSM data	16
3.8	OSM Vertex Quantity based connectivity matrix	18
3.9	LTA speed band values excerpt for 1000 queried sensors	20
3.10	Elevation and contour map of Singapore	22
4.1	Class diagram for <i>ShiftSet</i> structure	24
4.2	Folder structure for booking and logging data	25
4.3	Internal trip and booking instance structure in Matlab	26
4.4	Outline of the combination algorithm for logging and booking data	27
4.5	Active periods of logging and booking data	28
4.6	Possible GPS immanent inaccuracies, deviations, and erroneous points	29
4.7	Spatial outlier exceeding Singapore’s borders	29
4.8	Dynamics outliers exceeding acceleration and speed thresholds	30
4.9	Dynamic filtering algorithm flowchart	31
4.10	Distance and speed relations for the dynamic filtering algorithm	32
4.11	Interpolation of gaps and disarranged sample points	33
4.12	Comparison of altitude values for GPS logger and elevation map	34
4.13	Anchor points for altitude assignment	35
4.14	Status change extraction challenges	36
4.15	Status change trip extraction methods	38
4.16	Most distant point analysis	38
4.17	Long distant trip extraction method	39
4.18	Shift change pattern analysis	41
4.19	Example trip on the underlying road network	43
4.20	Projected candidate points for given sample point	44
4.21	Example for required transmission analysis	45
4.22	Example for required temporal component	45
4.23	Candidate graph construction	46
4.24	Map matching algorithm for candidate graph construction flowchart	47
4.25	Map matching post-processing algorithm flowchart	48

4.26	Micro trip extraction from a real-world trip	49
4.27	Cluster centroids for various two-dimensional excerpts	51
5.1	Evaluation of the combination of logging and booking data	56
5.2	Comparison of raw and filtered speed profiles	57
5.3	Results for acceleration and speed filtering	58
5.4	Comparison of speed values from GPS logger and computed speed	58
5.5	Interpolation of a tunnel transit	60
5.6	Map-matching evaluation for different sampling rates	62
5.7	Routing for different connectivity matrices	63
5.8	Average speed distribution subject to time over one day	64
5.9	Status distribution over one day	65
5.10	Status distribution per hirer scheme	66
5.11	Average daily mileage and hired share	67
5.12	Daily mileage distribution	67
5.13	Average hired trip distance	67
5.14	Hired trip distance distribution	68
5.15	Occurrence frequency heat map	69
5.16	Origin/Destination map for hired trips	69
5.17	Origin/Destination difference map	70
5.18	Cluster statistics displaying mileage and productivity	70
6.1	EV model application scheme	72
6.2	Share of consumed vs. recovered energy for different ARTEMIS cycles	74
6.3	Share of energy consumers for different ARTEMIS cycles	74
6.4	First layer of the TUM CREATE EV Model	75
6.5	Backward Model submodules	76
6.6	Principal scheme of the energy map conceptualization	77
6.7	Boundary effects resulting in length mismatch	78
6.8	Road type distribution	81
6.9	Road type micro trip length vs. road segment micro trip length	81
6.10	Schematic overview of energy map application	82
6.11	Comparison between map-matched trip route and origin/destination route	83
6.12	Impact of driving power submodules subject to speed	85
6.13	Curve fitting for <i>driving shares</i> functions of four driving phases	88
6.14	Curve fitting for <i>dynamics</i> functions of acceleration and deceleration	89
6.15	Curve fitting contemplating all values	90
6.16	Driving share application scheme	90
6.17	Extract of established 18 clusters of driving feature approach	92
7.1	Extract of representative trips for validation	93
7.2	Comparison of energy models for all representative trips	95
7.3	Overview of different sampling rates	96
7.4	Box plot basics	99
7.5	Energy demand and distance estimation for different sampling rates	100
7.6	Energy estimation results for the energy map approach	102
7.7	Evaluation of road segment based energy map	102
7.8	Energy estimation results for driving share approach	103

7.9	Evaluation of road segment based driving share approach	103
7.10	Driving share and dynamics per road type	104
7.11	Instant driving share approach evaluation	105
7.12	Instant driving feature approach evaluation	105
D.1	Shift set structure overview	111
D.2	Trip set structure overview	112
D.3	Trip structure overview	112
E.1	Average status distribution for all taxis	113
E.2	Average status share per taxis type	113
E.3	One- and two-hirer scheme status distribution	114
E.4	Daily booking status distribution per cluster	115
K.1	Singapore Regions, Planning Areas, and Subzones	127

List of Tables

3.1	Dataset statistics from 9 th June to 8 th September	9
3.2	GPS logger fields of Columbus V-990	10
3.3	Booking data fields as recorded from a MDT	12
3.4	Overview of OSM road network entity types	14
3.5	Boundaries of downloaded Singapore’s OSM extract	15
4.1	Fields of a feature vector for the driving feature approach	51
5.1	Average dynamics for an example trip with a predominant Motorway share	59
5.2	Average dynamics for various ARTEMIS cycles	59
5.3	Altitude and gradient evaluation	59
5.4	Comparison of various distance computation methods	61
5.5	Map-matching evaluation	61
5.6	Trip set statistics for September	64
5.7	Comparison of evaluated statistics with LTA data	71
6.1	Electric vehicle characteristics for EV models	74
6.2	Time zone characteristics	79
6.3	Road type overview, clusters and average speeds	80
7.1	Comparison of energy models for road type clusters	94
7.2	Characteristics of representative trips	97
7.3	Validation categories segmented into static, dynamic, and instant approaches	97
7.4	Comparison of energy approach results	99
7.5	Energy consumption per road type and time zone	101
A.1	Logging data example	108
B.1	Booking data example	109
F.1	Trip set statistics June	116
F.2	Trip set statistics July	117
F.3	Trip set statistics August	117
L.1	Taxi stands in Singapore, taken from [2]	128

Abbreviations

EV	E lectric V ehicle
ESS	E nergy S torage S ystem
PHEV	P lugin H ybrid E lectric V ehicle
SOC	S tate O f C harge
ICE	I nternal C ombustion E ngine
GPS	G lobal P ositioning S ystem
ABS	A nti-lock B raking S ystem
ESP	E lectronic S tability P rogram
LTA	L and T ransport A uthority
URA	U rban P lanning A rea
CBS	C entral B usines D istrict
MRT	M ass R apid T ransit
TDVL	T axi D river's V ocational L icence
MDT	M obile D ata T erminal
SD	S ecure D igital
CREATE	C ampus for R esearch E xcellence A nd T echnological E nterprise
RP	R esearch P roject
CPU	C entral P rocessing U nit
CAN	C ontral A area N etwork
MATLAB	M atrix L aboratory
OOP	O bject O riented P rogramming
PCHIP	P iecewise C ubic H ermite I nterpolating P olynomial
API	A pplication P rogramming I nterface
DB	D ata B ase
DBMS	D atabase M anagement S ystem
SQL	S tructured Q uery L anguage
VB	V isual B asic
LOC	L ines of C ode
JSON	J ava S cript O bject N otation
XML	E xtensible M arkup L anguage
CSV	C omma S eparated V alue

Physical Constants

Earth radius	R	=	6,371 km
Gravity	g	=	9.81 m/s ²
Speed of light	c	=	299,792,458 m/s
Air density (at 28°C)	ρ_{air}	=	1.1687 kg/m ³
Singapore North-South extent per degree	ϕ_{NS}	=	111.194 km/°
Singapore East-West extent per degree	λ_{EW}	=	111.158 km/°

Symbols

v	Speed	[m/s]
\bar{v}	Vehicle's average speed	[m/s]
$\overline{v^3}$	Cubed mean velocity	[m ³ /s ³]
v_{max}	Top speed	[km/h]
a	Acceleration	[m/s ²]
\bar{a}	Vehicle's average acceleration	[m/s ²]
d	Distance	[m]
h	Height	[m]
r	Radius	[m]
r_{wheel}	Wheel radius	[m]
A	Area	[m ²]
A_{front}	Frontal area	[m ²]
V	Volume	[m ³]
V_{cabin}	Cabin volume	[m ³]
m	Mass	[kg]
$m_{passenger}$	Passenger weight	[kg]
m_{car}	Car weight	[kg]
T	Temperature	[°C]
T_{cabin}	Desired cabin temperature	[°C]
$T_{ambient}$	Ambient temperature	[°C]
E	Energy	[kWh]
E_{total}	Total energy demand	[kWh]
E_{idling}	Energy demand for idling	[kWh]
$E_{cruising}$	Energy demand for cruising	[kWh]
$E_{accelerating}$	Energy demand for accelerating	[kWh]
$E_{decelerating}$	Energy demand for decelerating	[kWh]
P	Power	[W]
$P_{auxiliary}$	On-board power supply	[W]
P_{aircon}	Air-conditioning power	[W]
P_{max}	Maximum power	[kW]
P_{roll}	Needed power to overcome rolling resistance	[kW]
P_{air}	Needed power to overcome air resistance	[kW]
$P_{accelerating}$	Needed power for acceleration	[kW]
$P_{decelerating}$	Gained power from deceleration	[kW]
P_{grade}	Needed power to overcome road gradient	[kW]
$P_{auxiliary}$	Needed power for auxiliary power demand	[kW]

F	Force	[N]
M_{max}	Maximum torque	[Nm]
η	Efficiency	-
$\eta_{transmission}$	Transmission efficiency	-
η_{engine}	Engine efficiency	-
$\eta_{inverter}$	Inverter efficiency	-
η_{charge}	Charging efficiency	-
$\eta_{discharge}$	Discharging efficiency	-
$\alpha_{gradient}$	Road gradient	[%]
α_{head}	Vehicle's heading	[°]
ϕ	Latitude	[°]
λ	Longitude	[°]
ϕ	Humidity	[%]
$\phi_{ambient}$	Ambient humidity	[%]
ϕ_{cabin}	Desired cabin humidity	[%]
ρ	Density	[kg/m ³]
ρ_{air}	Air density	[kg/m ³]
τ	Share of driving phase	[%]
σ	Standard deviation	-
μ	Mean deviation	-
$\mu_{battery}$	Battery energy density	[kWh/m ³]
t	Time	[s]
YYYY	Year	-
MM	Month	-
DD	Day	-
hh	Hour	-
mm	Minute	-
ss	Second	-
O	Algorithm complexity	-
f_{sample}	Sampling rate	[1/s]
T_{sample}	Sampling period	[s]
p_i	Sample point i	-
c_i	Candidate point i	-
c_i^t	Candidate point i at time t	-
e_i	Road Segment i	-
e_i^k	Road segment i for candidate k	-
x_i^s	Distance between sample point and candidate	[m]
$d_{i-1 \rightarrow i}$	Distance between two sample points	[m]
$w_{(i-1,t) \rightarrow (i,s)}$	Shortest path between two candidates	[m]
$v_{(i-1,t) \rightarrow (i,s)}$	Actual speed of taken path	[km/h]
$\bar{v}_{(i-1,t) \rightarrow (i,s)}$	Average speed of proposed path	[km/h]
$N(c_i^s)$	Observation probability	-
$V(c_{i-1}^t \rightarrow c_i^s)$	Transmission probability	-
$F_t(c_{i-1}^t \rightarrow c_i^s)$	Spatial probability	-
$F_s(c_{i-1}^t \rightarrow c_i^s)$	Temporal probability	-
$F(c_{i-1}^t \rightarrow c_i^s)$	Taken path probability	-

F_r	Rolling resistance coefficient	-
c_w	Drag coefficient	-
$N_{passenger}$	Number of passengers	-
$C_{battery}$	Battery capacity	[kWh]
SOC_{init}	Initial SOC	[%]
SOC_{lower}	Lower SOC limit	[%]
SOC_{usable}	Usable SOC range	[%]
f_{idle}	Idle share function	-
f_{cruise}	Cruising share function	-
$f_{acc/dec}$	Acceleration share function	-
n_{trips}	Number of representative trips	-
n_{taxi}	Number of taxis	-
n_{time}	Number of time zones	-
$n_{cluster}$	Number of road clusters	-
n_{trip}	Number of targeted trips per category	-

Chapter 1

Introduction

Great attention has been paid in recent time to which extent a large-scale launch of electric vehicles will influence the electric grid and how to optimize the planning of a sufficient charging infrastructure. This thesis tackles the bottommost layer as outlined in Figure 1.1 where it comes to estimating energy demand of EVs as one of the atomic entities in the entire electrical grid and as a foundation towards stepping the layers upwards.

It focuses on two major parts: the *Driving Profile Analysis* and the *Energy Demand Analysis*. The driving profile analysis investigates a huge number of GPS trajectories recorded from taxis equipped with GPS loggers. As a driving profile, the speed and altitude trend subject to time is denoted. Obtained results serve as inputs for a subsequent carried out energy demand analysis where taxis' energy consumptions are examined by employing various estimation approaches. Herein, a comprehensive energy database is conceptualized, which can be queried for computing energy consumptions for arbitrary trips in the context of a simulation.

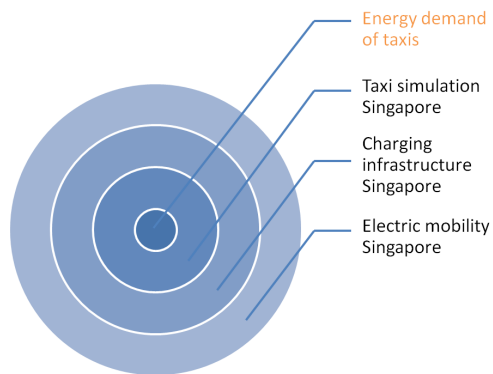


FIGURE 1.1: Thesis integration into RP 8 research topics

1.1 Thesis Goal and Integration into Current Research

The overall thesis' goal is the estimation of energy consumption for trips of different sampling periods, ranging from one second, over three minutes, up to the sole information about trip's origin and destination. Developed programs are designed to fit the needs of a taxi simulation in Singapore requiring to compute energy consumptions for simulated trips.

This simulation aims at establishing a suitable charging station infrastructure at *optimal* locations - optimal in the sense of projecting the actual driving patterns sufficiently, including

locations of taxi stands, breaks, and shift changes and virtually placing charging stations where the energy is needed. Such a simulation is vital for an economical operation of charging stations which fit the vehicle owner's (in this case the taxi company) and taxi driver's needs. A widely disseminated charging infrastructure is regarded as one of the main moderating issues for EV's market penetration.

The way charging stations are distributed can influence the overall electric grid, particularly pervasive for fast charging systems which demand a huge energy amount. Approaches towards smart charging [3] of EVs, which take network load and electricity prices into account, and vehicle-to-grid [4], which allow bi-directional energy flows treating EVs not just as a consumer but also as a temporary storage system, are evidences that this field attracts a lot of research. TUM CREATE also developed and prototyped an electric taxi named EVA, incorporating one of the first super fast charging systems prevalent [5].

During the editing of this thesis, the focus shifted slightly from a proposed 70–30 preference on the energy demand analysis part, to an eventually allocation of 50–50. Moreover, the development of the Matlab programs demanded a great portion of time resulting in a total number of 9090 effective Lines of Code (LOC), rationale to the focus realignment.

1.2 Potential and Impact of Electrical Vehicles

There is an ongoing discussion whether EVs are already economically feasible, what its actual CO² footprint is, and when will it pervasively penetrate into the market. In fact, benefits of EVs are an improvement in air quality - particularly in big cities, reduction of noise level, and the independence of gas and oil, which will definitely run out in near future. EVs have the potential for entirely emission free driving depending on which source is used for providing the electrical energy and eventually determining the degree of eco-friendly driving. To date, hybrid solutions are favoured comprising light hybrids like the Toyota Prius and Plug-in Hybrid Electrical Vehicles (PHEV) like the Chevrolet Volt.

For established companies, however, it can be a high barrier to get involved into solely electrical powered cars since they have a rooted reputation and customers tend to expect a certain vehicle style. For instance, BMW stands for sporty cars whereas Daimler emphasises on safety concerns, both requiring (and currently proceeding) a sensible integration of EVs into their portfolio. It is easier for small startups like Tesla to enter the market and develop a tailored EV from scratch.

As said, the quality and quantity of established charging infrastructure is highly influential to the pervasiveness of EVs. As long as there is only a low number of charging stations available, customer may refuse to buy an EV at their first choice. For investors, on the other hand, investing into charging infrastructure for only a few EVs will not be worthwhile and hence facilitates rejection from investments.

Not less important is the economical production of battery packs and the possible driving range. Up to date, EVs are more costly and demand frequent recharging, largely driven by the high price of battery packs and its limited energy density compared to gas. Furthermore, subsidies by government can ease introduction to overcome the vast price differences between internal combustion engine powered and electrical powered cars.

1.3 Particularities of Environment and Taxi System

When working on taxi data, the underlying taxi system and prevalent environmental conditions influence data analysis to a certain extent. Compared to the German taxi system, where drivers are usually permanently appointed with a fix monthly salary, Singaporean taxi drivers are waged by a comparably small fixed amount added by the passenger's revenue on a fare basis. The more trips the driver conducts, the more income is gained. The driver is hence self responsible for the amount of driving as long as he/she reaches the targeted mileage of the prevailing Taxi Available Standard of, up to date, 250 km per day [6].

Another peculiarity is the high amount of issued taxi licences of 99,711 in 2014 [7] and the fact that only Singaporean permanent residents are entitled to receive such a licence. With a total number of 27,865 taxis in the island state of Singapore in June 2014 [7], which is 2.9% of all 969,910 registered vehicles as of 2012 [8], there is a comparably high share of taxis. The taxi vehicles are hereby owned by the contracted taxi company and rented on a daily basis. This taxi share is even higher when examining the average daily mileages of private cars vs. taxis, former not exceeding 50 km [8] while latter reaches around 300 km, as taken from own investigations.

Including this, taxis are responsible for about 15% of all mileage and therefore emissions in Singapore. This enables a higher leverage effect when endeavouring EV efforts in the context of taxis and encourages concerned research. Changing ordinary taxis with internal combustion engines to battery powered drivetrains can hence have a great positive influence on environmental issues, particularly in Singapore with its high taxi density.

The Singapore based public transport company SMRT was chosen as a provider for taxi trajectories because it also runs Mass Rapid Transit (MRT) trains, which require a huge amount of electricity; and because it may also be a strategic partner in terms of providing electricity supplies for possible fast charging stations in MRT station's vicinity.

Environmental particularities are Singapore's tropical climate with diurnal temperature variation, high temperatures, and high humidity throughout the year. In terms of taxis this requires a constantly active air conditioning and expedites mechanical wear. Additionally impactful is the bounded spatial extent of only roughly 40 km to 60 km and the maximum speed limit of 90 km/h in the entire country. Both are taken into account when analysing recorded trips.

1.4 Thesis Structure

The present thesis is structured as follows.

Chapter 1 gives basic background information about electrical vehicles, the integration into current research with respect to charging infrastructure, as well as particularities of Singapore's taxi system.

In Chapter 2, applications and studies of previous conducted and related work for analysing driving profiles and estimating energy demands of EVs are reviewed. Those are summarized and distinctions, commonalities, and reusable approaches for this thesis are carried out.

Most important data sources utilized in this work are described in Chapter 3. Here, the data sources' origin, its structure, and necessary pre-processing and conversion steps for using those appropriately in the developed programs are explained.

The driving profile analysis is discussed in Chapter 4. It includes an in-detail examination of the processing chain to convert raw data into usable trips, comprising import, pre-processing, trip

extraction, map-matching, and eventually the generation of uploadable shift sets for an established SQL database.

Concluding results are stated in Chapter 5, where priorly described methods and approaches are evaluated and validated. Furthermore, a number of evaluated statistics about taxi driving patterns are plotted, which have been compiled in collaboration with SMRT and which describe the dataset in an insightful graphical manner.

The subsequent energy demand analysis is conducted in Chapter 6. Utilized energy estimation methods differentiated by static, dynamic, and instant approaches are introduced in detail. In addition, energy models for database set-up and validation purposes are described.

Followed by a result unit in Chapter 7, an evaluation of proposed approaches is presented and validation results are shown. The validation process is stated and results are compared against a reference vehicle model, and possible deviation sources are stated.

Eventually in Chapter 8, the work is concluded, results discussed, and outcomes assessed based on evaluation criteria established in previous chapters. Suggestions for application opportunities, particularly in terms of the planned taxi simulation in Singapore, are given, and the thesis finishes with recommending further areas of work.

Chapter 2

Related Work

The literature review undertaken in this thesis strives to outline pervasive methods for analysing GPS trajectories in form of driving profiles with its required pre-processing steps to mitigate GPS specific shortcomings. It furthermore presents a non-exhaustive abstract about latest energy estimation applications fed by prepared GPS trips and states which ideas and methods flow into the development and implementations.

2.1 Driving Profile Analysis

The opening of the Global Position System (GPS) for private purposes in 2000, discontinuing the artificial signal deterioration of *Selective Availability* and allowing users to receive an undistorted signal globally, escalated research in that field [9]. Initially fully available in 1995, GPS underwent a continuous modernisation process in which precision and reliability increased dramatically. Nowadays, a GPS receiver is shipped with every modern Smartphone leading to an escalating development of navigation applications. In a professional context, like car navigation and approaches towards autonomous driving, applications rely on precise positions gathered from GPS satellites.

In [10], the theory, algorithms, and typical applications are discussed in a detailed manner. This work employs therein proposed pre-processing steps in Section 4.2, contemplating GPS characteristics with its deviations, influence factors, and limitations. Discussed pre-processing filters are Kalman, Mean, and Median filters, of whom the two last mentioned are incorporated.

Extracting trips from a pool of numerous GPS points is topic of [11]. The paper deals with route predictions from trip observations, exploiting that trips are most likely to follow a reoccurring pattern rather than being chosen randomly. For this, equally to nominated needs, the paper dives into extracting, cleaning, and filtering trips. Described methods are adjusted to the dominant environment but in principle are adopted as proposed.

In Section 4.5, a micro trip extraction according to conditions approached in [12] is employed. Micro trips are short, roughly three minute long excerpts of real-world trips and are a convenient way to analyse distinct driving patterns on small independent portions.

A major part of the driving profile analysis is spend on map-matching, a technique to project real-world GPS trajectories to an underlying road network. Since the established approaches seek to cover a broad range of sampling rates, conducted literature review focused on relatively low sampling rate proposals and made a good match with [13] and in particular [14]. The algorithm introduced in the latter paper is implemented in this work as it appeared to suit nominated

requirements best. Its applicability for low sampling rates is highly determined by the global map-matching background which takes not only the whole trip from origin to destination into account, but which also incorporates the road network and traffic constraints.

2.2 Energy Demand Analysis

For estimating energy consumptions of EVs, there are basically four approaches prevalent, using either a vehicle model, a machine learning method, a statistical method, or a traffic simulation method. A combination of those is also perceivable. The following sections firstly present generic approaches towards estimating energy and secondly follow up with a number of specific, state-of-the-art applications.

2.2.1 Generic Approaches

A vehicle model simulation is applicable if a high resolution driving profile with detailed dynamics (speed, acceleration, altitude) is available. This is needed because a model mimes an actual driven trip solely on basis of its dynamic information and its trend. This study uses an electrical version of a vehicle model for constructing a road segment based energy map and for validating results among all proposed energy demand estimation approaches.

Employing a machine learning method targets at finding typical and reoccurring patterns in driving profiles or traffic conditions. Often, these driving profiles are grasped from standard driving cycles [15] and applied on arbitrary real-world trips. Those are processed utilizing machine learning methods for clustering (portioning of similar pattern into groups) and classification (assignment of unclassified trips to established clusters), as for instance done in [16]. In the present work, similar methods are implemented in the driving feature approach where distinct driving patterns are linked to energy consumptions.

Statistical methods make use of a number of recorded trips for establishing a database for trip related statistics, such as dynamic information, road network correspondence, or energy consumptions. An initial attempt to construct an energy map of Singapore is proposed in [12], in which each cell of a meshed grid denotes an average value of energy demands of a number of recorded real-world trajectories. This method is employed in this thesis for the road segment based energy map and road segment based driving share approach, where all recorded trips so far are included.

The use of traffic simulation tools such as SUMO [17] aims at emphasizing relations between energy consumption and traffic conditions as proposed in [18]. Although derived results show that the energy consumption is indeed heavily dependent on traffic conditions, basically following the state of congestion and average speed, the present work focuses on recorded real-world trips in the context of taxis and therefore does not include a traffic simulation into the analysis.

2.2.2 Specific Applications

There are numerous vehicle models available for modelling classic internal combustion engines (ICE) as well as electrical drivetrains. A basic idea for the challenges constructing an electric vehicle model is given in [19], where all sub-modules are described in detail. One of its implementations is obtainable from [20] as a Matlab integration. The VEHLIB [21] and PHEM [22] models are widespread for estimating vehicles' emissions and are designed for internal combustion

engines. Although in principle similar to electrical vehicle's models regarding vehicle dynamics, they fall short for energy recuperation, charging, and discharging processes.

In [23], an eco-routing navigation system is elaborated using the CMEM [24] emission model employed on real-world driving profiles. Appended with multiple historical data sources, like roadway characteristics and results from traffic simulations, it provides a comprehensive energy estimation concept. However, its focus lays on real-time energy estimation for current traffic conditions and therefore does not fit to this study's objective of constructing a database of historical trips for subsequent trip simulations.

A modular, crowdsourced approach is introduced in [25] using a combination of collected real-world speed profiles, in-vehicle sensor data, and traffic information. It targets at a real-time energy estimation based on a comprehensive back-end server infrastructure communicating with the vehicle. Alike herein proposed approaches, real-world trajectories, road network, and traffic constraints are taken into account; however, it differs in targeted application scenarios and infrastructural context. Nevertheless, portions of herein established methods are adopted.

In [26], a method operating on driving shares is proposed. A driving share is a distribution of four distinct driving phases: idling, cruising, accelerating, and decelerating. An extraction of functional parameters of overarching function shapes describes driving phase distributions in accordance to speed, reveals the cars dynamic behaviour, and alludes to consumed energy. These shares are used to compute driving phase related energy consumption for various road types and gradients, corresponding to the set of parameters and average speed. An adopted version of this method is profoundly implemented in this work for the driving share approach.

Chapter 3

Data Sources

This chapter introduces various data sources and resources which contributed to the analysis, program development, and statistical evaluation. The main data source this work establishes upon are real-world GPS trajectories recorded by GPS loggers built into a number of 20 taxis and extended with booking data revealing the taxi's status. The participating vehicles are part of the SMRT taxi fleet, a Singaporean based public transport company running MRTs, buses, and taxis. In terms of taxis, it is the fourth biggest taxi carrier in Singapore with 3227 taxis registered in June 2014 [7]. In addition, auxiliary data was obtained by querying a number of open source or crowd sourced data resources which provide valuable additional input material for the data mining process.

The opportunity to access a number of different data sources requires two steps: independent pre-processing of each source and combination of all sources into a utilizable format. Merging multiple sources improves quality and insightfulness of data mining, allows for more detailed statistics, and enables to reuse data in a wider range for related projects.

The cooperation with SMRT was chosen for several reasons. One is the opportunity to access taxi data in a tropical city like Singapore at which also the EVA project aims [27]. Additionally, SMRT has an extensive MRT station infrastructure, each demanding high amount of energy and thus unlocking opportunities for EV charging stations with possible energy supply points at those stations. Moreover, it is granted to also obtain taxi's status information through a booking dataset, essentially enhancing the trip extraction process.

3.1 Taxi Data

The developed programs primarily operate on received taxi data from SMRT, which includes logging and booking data. The logging data, with a sampling period of one second, is captured from installed GPS loggers whereas the booking data, with a three minute sampling period, is gathered from the taxi's Mobile Data Terminal (MDT), which is responsible for managing bookings and fare deduction. A set of 20 loggers was installed into a representative selection of taxis, chosen according to various features like hirer scheme, drivers' home location, and vehicle type.

The logger devices are expected to record trajectories for a duration of six month, given an option of extending recording time for a subsequent six months period to add up to one year. During the creation of this thesis, due to successful results, the number of installed loggers was

increased to 50; albeit the here proposed analysis and discussed results only cover data from the first phase with 20 loggers.

Recorded data is stored on Micro SD memory cards equipped to the logger devices and exchanged with a new set of memory cards each month to guarantee uninterrupted data recording. Hence, there are total 40 Micro SD cards in stock. A change of memory cards is required due to GPS logger recording constraints of maximum 512 files per card, which is expected to be exceeded after roughly one and a half month, thus feasible in the planned set-up. The exchange also allows to check data quality and to conduct data analysis on a monthly basis.

The initial set-up, testing, and installation was carried out by the participating research group, while also introducing the system’s basic functionality and usage guidelines to the drivers. For proper operation, deployed GPS loggers have to be prepared prior to deployment by formatting the memory cards and adjusting recording properties. The card exchange, on the other hand, is done by SMRT.

During monthly update meetings, recorded files of all memory cards are collected while providing empty cards for the next exchange process. Furthermore, during those update meetings, insights into preliminary data analysis results are presented, occurred problems discussed, and valuable feedback is retrieved - aiding further analysis actions. To allow for fast and straightforward memory card exchange, all memory cards are numbered with a unique id and maintained in an Excel sheet, which records the link between taxi licence plate number and currently plugged memory card id.

An overview of recorded data is depicted in Table 3.1, divided into subsets for each recorded month. The following sections firstly, delineate the GPS trajectories obtained by GPS loggers and secondly, explain the status information as recorded from MDTs.

TABLE 3.1: Dataset statistics over a period from 9th June to 8th September, divided into monthly subsets

Statistic	June	July	August	September	Total
Taxis	19	20	20	20	20
Recorded days	22	30	31	8	91
Extracted shifts	766	1,045	1,006	224	3,041
Extracted trips	19,802	26,047	26,023	5,434	77,306
Distance	130,440 km	170,700 km	166,610 km	35,336 km	503,086 km
Sample points	15,787,061	20,962,228	20,325,569	4,278,416	61,353,274

3.1.1 Taxi Logging Data (GPS Trajectories)

This section begins with a brief introduction into the Global Position System’s (GPS) functionality in general and hands over to the specific GPS logger type as utilized in this work.

GPS Basics

The GPS is a complex system for determining a position on Earth’s surface with an accuracy of nowadays ± 20 m under normal operating conditions. Launched in its fully operational mode in 1990, it comprises a set of 24 satellites in six orbital planes, each equipped with four satellites [10, p. 2].

The essential functional principal bases on measuring time differences between satellites in the orbit (sender) and GPS device on Earth (receiver). Those time differences are translated into

metric distances between sender and receiver as well as distances among the satellites, and they eventually result into a fairly accurate three-dimensional position. With inclusion of satellites' exact locations, the position is expressed in commonly used spheric coordinates of latitude and longitude values, both relative to the equator and the Greenwich meridian, respectively.

Since transmission signals are running at the speed of light, which is extraordinary high, the systems demands a very high clock precision and clock synchronisation among all satellites and between sender and receiver. Thus, the receiver can only determine a valid position if it is concurrently connected to at least four different GPS satellites, three for providing the three-dimensional position and one for synchronizing the receivers internal clock.

GPS Logger

To record GPS trajectories, a set of Columbus V-990 Multifunctional GPS Data Loggers (Figure 3.1) is used, incorporating an Enhanced Positioning System (EPS) [28] for accuracy improvement. Important criteria in this project's scenario are a fast cold and warm start-up phase as well as a high accuracy and high sensitivity - even when operated in a moving car. Furthermore, requirements for devices comprise the capability to overcome power interruptions, robustness to taxi driving conditions, and long durability. Because the Columbus logger was already proved valuable in previously conducted, similar studies like in [29], the decision was made to acquire a set of it. The logger stores one sample point per second (sampling rate 1 s^{-1}), each attached with fields provided in Table 3.2.



FIGURE 3.1: Columbus V-990 GPS logger

TABLE 3.2: GPS logger fields of Columbus V-990

Logger Field	Typical Value
Id	2015
Tag	T
Date	140901
Time	25014
Latitude	01.284309N
Longitude	103.848928E
Height	58
Speed	41
Heading	115

The device is automatically turned on when the taxi's on board power supply is activated because power is directly obtained through the cigar lighter socket. This normally happens when the engine starts and all devices are supplied with electricity. However, in some vehicle types, the power supply does not align to the engine start and stop pattern but remains active throughout the time. Here, a manual on and off switching is required by the driver, each time a shift is interrupted. For each stint between a logger on and off mode, a unique Comma Separated Value (CSV) file is generated and stored on the plugged memory card.

Under the given operational scenario faced in the taxi tracking purpose, GPS positions underlie a number of uncertainties, inaccuracies, and challenging traffic situations such as tunnels transits, loss of signals, multipath effects, and position drifts as it is later discussed in Section 4.2. A basic insight into the nature of a recorded trip is illustrated in two detail levels in Figure 3.2, of which an excerpt of its raw CSV source is provided in Appendix A.

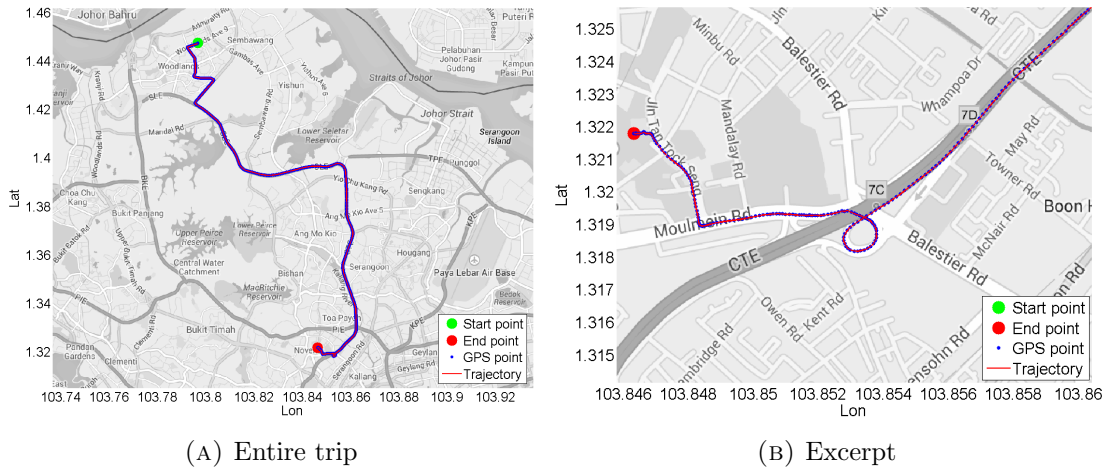


FIGURE 3.2: Example of a recorded trip and a detailed excerpt, background images taken from Google Maps API [30]

An essential information for this work's imposed intention are the speed values, which either can be calculated programmatically using latitude, longitude, and time values of two adjacent sample points, or which are directly obtained from the logger's *Speed* field. Investigations revealed that the provided speed values from the logger are more reliable and are in high accordance to those obtainable from the CAN Bus. CAN Bus values are eminent as a good reference source since they are directly taken from wheel sensors, which instantaneously calculate the speed in a very exact manner. Another evidence is the fact that CAN BUS speed values determine security relevant systems like ABS and ESP and therefore must be of high accuracy. A comparison of both speed value sources in a similar set-up employing the same logger type is presented in Figure 3.3.

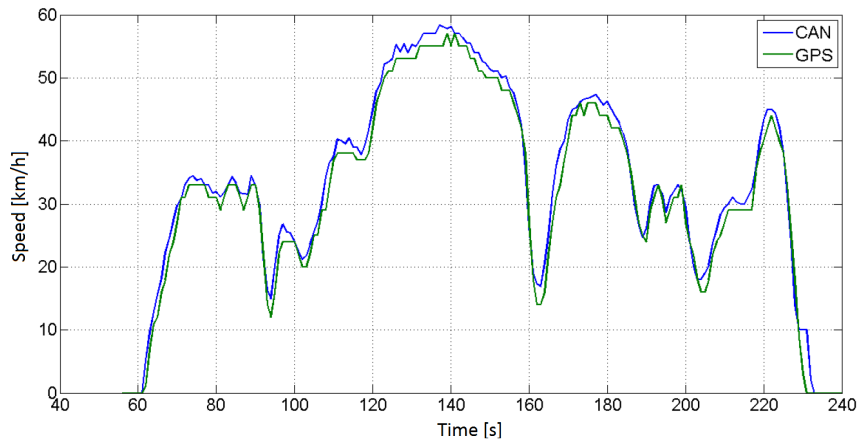


FIGURE 3.3: Comparison of speed values from CAN Bus and GPS logger, adopted from [31]

3.1.2 Taxi Booking Data (Status Data)

In line with the logging data proceedings, booking data was obtained during update meetings with SMRT on a monthly basis. Each taxi is equipped with a MDT, which continuously captures the taxi's GPS position, timestamp, and status information in an approximate sampling period of three minutes and acts as a booking manager and device for deducing taxi fares. It requires

minor interaction with the drivers despite the input about the current trip's purpose (the actual taxi's status).

Since the fare collection starts when the driver hits the *Hired* button and stops when the driver hits the *Payment* button, it is confident that provided status information is valid. However, a number of statuses like *For Hire* or *Busy* are dependent on driver's choice and taste and should be treated with more courtesy. All captured data is automatically transmitted to an SMRT database which steadily receives statuses from all active taxis. Of particular interest is the time and status datum which is forwarded in an Excel sheet format as an excerpt is listed in Table 3.3.

TABLE 3.3: Booking data fields as recorded from a MDT

Booking Field	Typical Value
Taxi Licence	SHF323S
Latitude	01.362683
Longitude	103.7447
Status	HIRED
Date and Time	9/8/2014 12:08

The overall quality of booking data in terms of accuracy and completeness is considerably good. Challenging is, at times, the unpredictable behaviour of the booking data collection system (the MDT and connected database). Examinations yield that the system in general records data whenever the taxi is active, assuming the MDT online and taxi's engine running. Sometimes however, gaps in booking data are encountered in comparison to the corresponding logging data. If latter are available and former not, it is evidence for the incompleteness of booking data.

Another issue, which nevertheless does not have a strong deteriorating impact, occurs vice versa where the booking systems continues capturing *Log Off* statuses while the taxi is not active at all. Further investigations touching that issue are conducted in Section 4.2.1 in which the data combination is approached. To some degree impactful is the unsteady sampling rate which exacerbates the combination process and requires interpolation.

Figure 3.4 presents a booking data excerpt for one taxi and one day, denoting sample points' positions and corresponding statuses by means of different colours. The colour-code is used unified throughout this work and explained as follows.

■ **Unknown** The status could not be gathered due to a mismatch between booking and logging data or due to missing booking values for the observed logging period.

■ **Log Off** The taxi is not active, or the hirer has logged off the MDT, respectively.

■ **Hired** A passenger has hired the taxi, and the taximeter is activated.

■ **For Hire** No passenger is on board, and the taxi is searching for a next customers.

■ **On Call** The taxi received a confirmed booking and drives towards the agreed booking's pick-up location.

■ **STC** When a passenger is on board, the driver switches to this status to inform that he/she is close to the destination and about to clear the job soon. Hence, this status will only occur after a preceding Hired status.

■ **Arrived** The taxi arrives at the destination as defined in a booking for a passenger pick-up. This status is most likely to happen after a preceding On Call status.

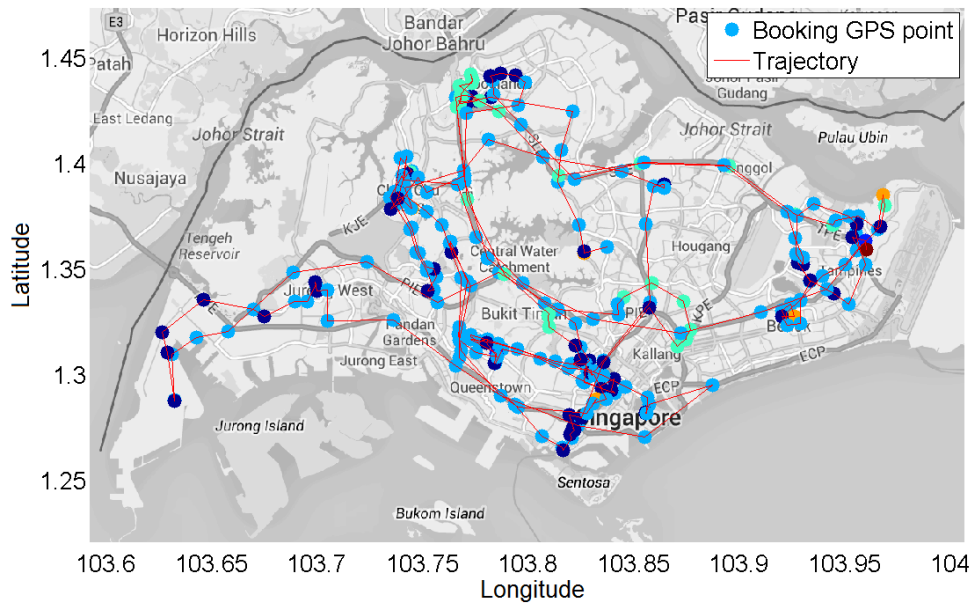


FIGURE 3.4: Booking data example showing recorded sample point's positions and corresponding statuses for one taxi and one day, background image taken from [30]

■ **Payment** The trip has finished and the passenger is about to pay why the taximeter is stopped for payment. Alike STC, a previous trip with status Hired is most likely.

■ **Busy** The taxi driver uses the taxi for his/her own purpose and the Busy status is logged in the MDT.

3.2 Auxiliary Data

Auxiliary data supports the data mining process in various ways and enhances recorded GPS trajectories with road network correspondence, elevation values, and zone matching (division of Singapore). Those data is fetched from multiple sources and undergoes a comprehensive pre-processing as also described in this section. First, Singapore's road network as the foundational input for map-matching is regarded. Afterwards, general infrastructure and the gathering of taxi data is outlined, followed by spatial and geographical data sources as used to re-assign flawed GPS fields.

3.2.1 Singapore Road Network Source and Preparation

The coherence between trips and road network is of principle interest for the energy estimation approaches, especially for those requiring a point-by-point matching to road segments. For this purpose, a road network is required which is easily accessible, of high quality, and straightforwardly importable into the Matlab development environment. The road network utilized in this work is gathered from Open Street Map (OSM), a crowdsourced and licence-free accessible database for road networks and infrastructural entities around the world [32]. The map quality and level of completeness is conditional to the location but is generally available in greater detail and more complete for urban areas. In case of Singapore, a very satisfying quality in terms of completeness and detail level is perceived with only negligible limitations.

In principal, OSM stores road network information as of *Nodes*, which are points with precise GPS coordinates (latitude and longitude fields), connected by *Ways*, which link a set of nodes to form an entity. Note that the object Way is not exclusively used for roads but more generic also designates all other entities like buildings, pathways, and even country borders. Both types can optionally be attached with additional *Tags*, indicating meta information such as traffic lights or pedestrian crossings - in case of nodes - and street names or road types in case of ways. In addition, OSM introduces *Relations* which are compounds of nodes and ways to indicate abstract resources such as parks but disregarded herein.

Table 3.4 lists the quantity and average length of nodes and ways for the downloaded version of a bounded Singapore OSM extract as of May 2014. It also includes the characteristics of *Road Segments* and *Links*, which are introduced afterwards.

TABLE 3.4: Overview of OSM road network entity types with its quantity and average length

Type	Value	Average Length
Nodes	107,388	-
Ways	20,878	216.56 m
Road Segments	40,425	134.70 m
Links	95,245	24.71 m

Road segments are derived from ways and oblige the restriction to have no intermediate intersections. They are introduced for the purpose to consider a road segment as a separate and closed unit, avoiding cars to leave the segment. OSM reveals no standard recommendations when to split a way into multiple parts; hence, a transformation from raw OSM into a form of non intermediate intersections is targeted to enable uniformness. Links, on the other hand, are defined as atomic connections between two adjacent nodes.

Applicable definitions can be observed from Figure 3.5. Herein, the road segment is bounded by two intersections, which are considered to be intermediate in relation to the way, and thus it requires splitting.

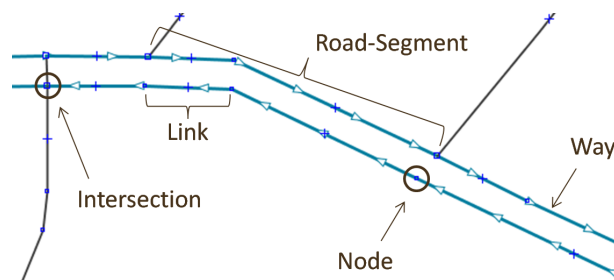


FIGURE 3.5: OSM definitions for *Node*, *Way*, *Road Segment*, *Link* and *Intersection*

Downsides of crowdsourced data is its to a certain extent naturally arising unreliability and inconsistency. Since everyone of the more than 1,700,000 registered users (as of August 2014 [33]) is entitled to edit the content without major restrictions, different mapping styles and interpretations of OSM best practice advises are unavoidable. Challenging are, for instance, unconnected ways, ambiguous assignment of road types to ways, and the general problem that node and way tags are not mandatory. Those issues makes only a subset of the entire OSM dataset feasibly utilizable.

Import of OSM and Filtering

To be independent of internet connection and to accelerate processing speed, a copy of an OSM extract of Singapore was acquired, defined by boundaries of longitudinal and latitudinal extent as depicted in Table 3.5.

TABLE 3.5: Boundaries of downloaded Singapore’s OSM extract and its spatial extent

Direction	Boundary	Position	Extent
Longitude	North	1.4728372°	31.81 km
	South	1.1867929°	
Latitude	West	103.6106791°	52.97 km
	East	104.0871962°	

The OSM website provides the opportunity to download an XML formatted file, which solely consists of the three previous introduced markup types (Node, Way, and Relation) and attaches optional tags in a common key-value pair structure. The raw file contains a number of unimportant entities like buildings, forests, and pedestrian pathways, which are unnecessary for this thesis’ scope and can be dismissed. Regardless of its nature, each entity is mapped in the same manner using nodes and ways and would therefore inevitably impede efforts to extract only road network related information. Thus, the map was filtered using the tool `osmfilter`, capable of dismissing predefined entities.

Since the tool expects a rectangular boundary for the OSM export, which obviously does not project the real shape of Singapore, an additional cropping step has to be applied, partly automatically executed by the tool `osmconvert` and - for fine-tuning - partly done manually. Cropping aims to dismiss entities which do not lay in the desired boundaries. Results comparing an unfiltered with a filtered and cropped OSM excerpt are illustrated in Figure 3.6, where only the interesting entities (roads and corresponding nodes) are left. As a final step for importing the road network, the library package `osmfunctions` is used, which translates the XML file structure into a suitable Matlab format and is obtainable from Matlab’s File Exchange website under [34].

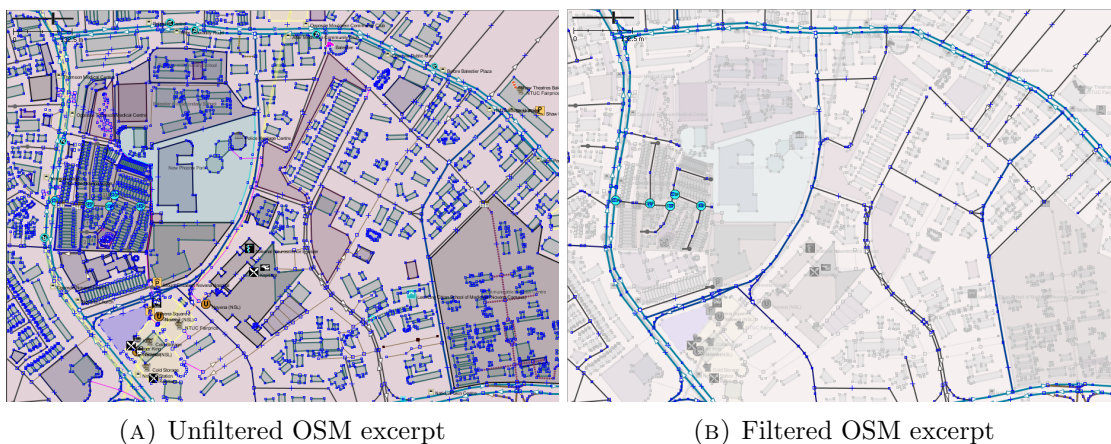


FIGURE 3.6: Unfiltered and filtered OSM excerpt dismissing irrelevant entities

Preparation of OSM in Matlab

As described in the class structure overview in Figure 3.7, the imported XML file undergoes a number of processing steps until sufficiently prepared for the planned usage scenario. Hereby, Matlab's built-in Object Oriented Programming (OOP) module was frequently leveraged to handle complexity more manageable.

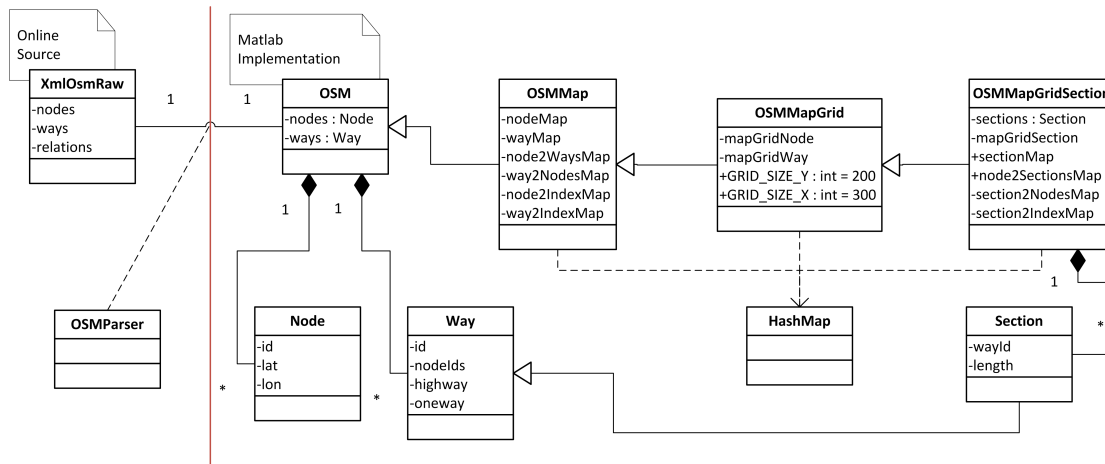


FIGURE 3.7: Class diagram for processing OSM data, including *OsmXmlRaw*, *OSM*, *OSMMap*, *OSMMapGrid* and *OSMMapGridSection*

In a first step, the imported OSM road network is converted into an instance of an appropriate class based format *OSM*, using an instance of *OSMParser* to extract XML tags of nodes and ways and to create, for each, instances of classes *Node* and *Way*. The node instances inherit latitude, longitude, and the node's id field. All ways store a list of included node ids, and the additional key-values pairs *highway* and *oneway* tags are extracted. It is noted that the term 'road type' and 'highway' are used synonymously to avoid confusion with the road type Motorway.

A crucial step towards performant utilization of the road network representation in Matlab is to enhance it with *hash maps* (or lookup tables) of most occurring queries, comparable to the technique of indexing large database tables. One reason is that ids of nodes in OSM live in a global namespace among which they are unique. Due to the high id range (up to Billions), they cannot be used efficiently as indices for Matlab arrays. This requires to build a fast lookup opportunity to match global node ids to the internal representation of local ids.

Another reason is given due to the fact that only ways store a set of node ids, but not vice versa. Without sophisticated hashing, thus, no direct information have be extractable about a node-to-way link. For instance, in order to retrieve all ways which contain a demanded node, without hashing, each way has to be traversed by searching for the asked node id. With hashing, however, a previously establish node-to-way hash map is utilized, which allows for comfortably fetching of all ways for the given node id. The hashing enhancement results in an instance of *OSMMap*, which inherits from the *OSM* class and contains six different maps for frequently demanded queries.

Particularly vital for the map-matching is a spatial relation between a certain areal and ways contained. It is of major interest to find neighbourly way candidates for a given sample point (see Section 4.4.2). This is achieved by dividing the area of Singapore into a 200×300 meshed grid, resulting in 60,000 rectangular cells. As expressed in the class *OSMMapGrid*, two additional hash maps for ways and nodes are set up.

A last step, which came up during first challenges in implementations, is the extraction of road segments denoted as *sections* in the class diagram. Road segments are a subset of ways, restricted to have no intersection with any adjacent road segment other than at its very first or very last edge. Particularly for the road segment based energy estimation approaches, this is an important requirement because it enables to consider a road segment as a closed and undividable unit, which has either been traversed by the car or has not. To extract those, ways are cut into two chunks at intermediate intersections. It is recursively proceeded in this manner with the remainder until no such intersections are left. The processing result in an instance of *OSMMapGridSection*, attached with a field *sections*, fields of all previous introduced refinements, and enhancing steps in the context of road segments.

Extraction of OSM Connectivity Matrix

The map-matching process implements the shortest path problem which employs a directed and weighted graph connecting all nodes of the imported OSM road network. This directed graph is in this work's context called *connectivity matrix*. According to the definition of a directed graph, each OSM node acts as a *vertex*, and each link connecting two nodes acts as a *path*. The graph is chosen to be directed because it contains one-way streets (one directional = directed) and the projected road network itself already behaves like a directed graph. A *weight* value denotes the 'cost' of traversing a path from node to node.

The weights for each path can be calculated subject to various targeted intentions as summarized in the following delineations.

Vertex Quantity The weight attributes solely base on the amount of intermediary vertices from one node to another. Since the directed graph contains a weight value for each link between all possible nodes and not considering a path traversing more than two nodes, the weights are all set to unity by default (unity = one point per link). A special advantage of this graph version is its high computation performance since all weights are equal. This allows for using the high performance Breadth First Search algorithm (BFS) with complexity $O(|V| + |E|)$ [35, p. 597], where V is the number of vertices and E the number of edges (or paths). Yet, it omits the actual important distance between nodes and is thus less of value.

Euclidean Distance A weight factor is computed based on the Euclidean distance between two adjacent nodes with its latitude and longitude coordinates. In routing algorithms, this graph type is known as the actual shortest possible distance, but it does not imperatively translate into the fastest journey time. This connectivity matrix requires Dijkstra's algorithm with complexity $O(|V|^2)$, where V is the number of vertices [36].

Time Demand The weights origin from an estimated time for traversing a link. Similarly to the previous version, the Euclidean distance is computed and additionally divided by the average speed for the corresponding road type as obtained from observed trips or LTA speed band sensors. Applying this graph to a navigation system would resemble the fastest route option. The shortest path problem is solved again employing Dijkstra's algorithm.

To constitute such a connectivity matrix, an algorithm basically sifts through all nodes and attaches a weight factor for each connection to adjacent nodes. As described, these weights are obtainable using different variants which have all been implemented. This algorithm already takes advantage of the previous described hashing step, which allows for a higher performance in the lookup of nodes and ways.

To prepare the connectivity matrix for extensive application in a map-matching environment in terms of performance issues, the Matlab concept of *sparse matrices* is exploited, which only store valid connections (where weights are unequal zero), and the majority of unconnected transitions remain untouched. In Figure 3.8, the Vertex Quantity based connectivity matrix with unity weights is illustrated, each point representing a connection between two nodes (the node ids are displayed on both the x- and y-axis). The thicker diagonal line originates from a high number of connected nodes with similar node ids.

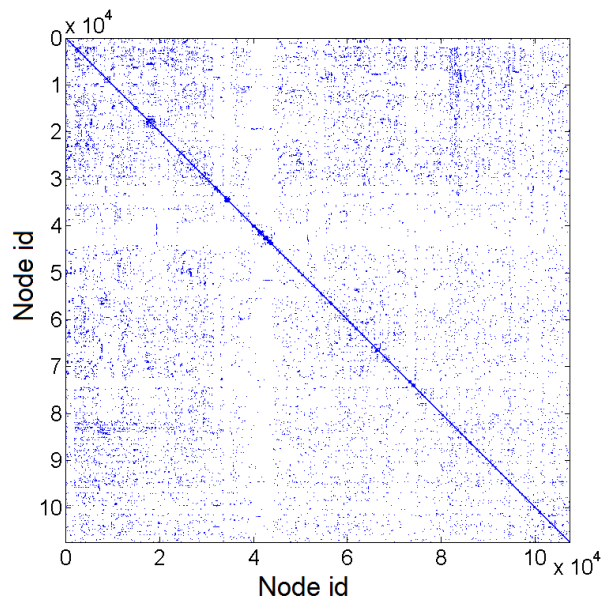


FIGURE 3.8: OSM Vertex Quantity based connectivity matrix for all 107388 nodes of the overall OSM road network

3.2.2 Infrastructure and Taxi Data

Infrastructure and taxi related data is mainly used in trip and shift extractions and as a validation option for speed distribution around Singapore's road network. As all auxiliary data, it also influences and enhances the statistical evaluation. An outline about the acquisition of taxi fleet, taxi stands, and LTA traffic speed band data is given in the following sections.

Taxi Fleet Data

As previously mentioned, the partner for recording GPS trajectories is the Singaporean public transport company SMRT. In a first project stage, a subset of 20 taxis was equipped with GPS loggers, keeping the option to extend the set to a total of 50 taxis. It is of great importance for the degree of validity that the dataset is as representative as possible to derive relevant statements which are extendable or upscalable for the entire taxi fleet. The representativeness is regarded as the distribution of drivers home addresses, vehicle type, and hirer scheme, which all should reflect the actual distribution among the entire taxi fleet.

The driver's home address is significant because taxi drivers tend to serve their neighbourhood preferably. Here, SMRT was asked to deliver a good match to the overall home address distribution and four drivers from the *North*, three from the *East*, one from the *South*, six from the *West* and six from the *North-East* were chosen.

Also the vehicle type certainly has an influence on GPS trajectories on account of its unique characteristics and vehicle specific dynamic behaviour. The SMRT fleet is mainly composed of four types: *Chevrolet Epica*, *Chrysler 300C*, *Ssangyong Rodius*, and *Toyota Prius*, of which also a representative distribution was selected, resulting in 13 Chevrolet, one Chrysler, two Ssangyong and four Toyota.

Likewise, the hirer scheme which constitutes the number of hirers agreed to drive one taxi, was chosen accordingly and eleven two-hirer, and nine one-hirer scheme taxis were picked.

Taxi Stands in Singapore

Although less important than compared to the German taxi system, the geographical positions of taxi stands in Singapore reveal locations with high occurrence probability and therefore play a major role when locating a suitable charging station infrastructure. As already alluded in Section 1.3, the Singaporean taxi system has some peculiarities which also affect search strategies for customers. Other than the German example, in the majority of cases in order to board a taxi, customer just flag down a taxi virtually anywhere on a street, and passengers are not required to wait at taxi stands.

Nevertheless, taxi stands are often contemplated in metropolitan areas where a high taxi demand is faced in certain peak hours. Examples are points of interest like shopping malls, hotels, hospitals, and sights with taxi stands or pickup bays right in front. The data source also includes locations of bus stops which are sometimes misused for passenger pick-ups but not investigated under this thesis' scope. Yet, they may be consulted in future work.

Motivated by the rational to act as possible charging locations, a taxi stand database was queried and imported into Matlab to be incorporated into the analysis. The demanded data is available free-of-charge from Singapore Government Data website [37], which provides a great set of useful and high quality data.

The data was received in the form of *Shape Files*, which initially do not reveal the actual latitude and longitude coordinates of interest. Hence, the files have to be processed using the tool *QGIS Desktop* to convert it into *GeoJSON* format. This GeoJSON files can now be imported into Matlab using a JSON parser as available on Matlab File Exchange [34]. A list of all imported taxi stands divided into ten major sectors is attached in Appendix L.

LTA Traffic Data

Preliminary to validate recorded GPS trajectory speed values and to gather average speeds for Singapore's road network, the *LTA Speed Band* database was queried as available under [38]. This data source holds values of an extensive amount of more than 40,000 speed band sensors distributed all over the island. A speed band is herein defined as a pair of lower and upper speed limits, denoting the average speed of vehicles for the last five to ten minutes in a resolution of 20 km/h steps. Values of each speed band sensor are periodically updated and asynchronously fetched and hence accessible in quasi real-time from an API.

To do so, a one-year account was registered entitling to query the API at all times. Each query is restricted to a subset of 50 sensors per call and produces an output in XML format, which is processed into Matlab using a built-in XML parser `xmlread`. The API works on HTML requests with base address `http://datamall.mytransport.sg/ltaodataservice.svc/`, the web service name `TrafficSpeedBandSet`, plus an additional passed parameter `skip=50`, which determines the start of the requested sensor subset.

An example of a 1000 sensors encompassing excerpt is displayed in Figure 3.9, where the colour depicts the speed band values using green for the fastest (80 – 100 km/h) and red for the slowest speed band (0 – 20 km/h).

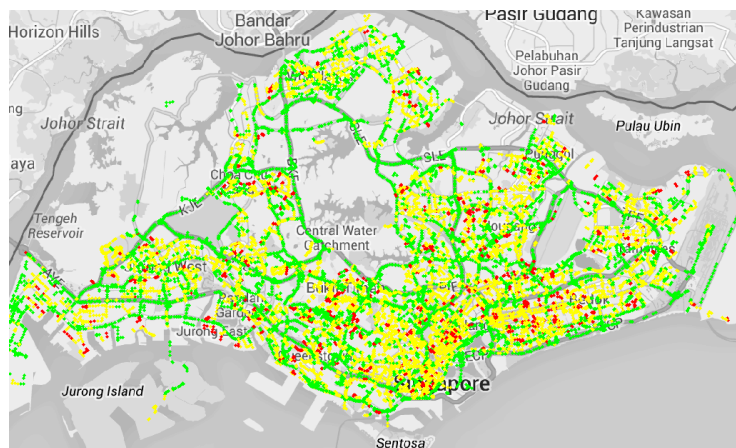


FIGURE 3.9: LTA speed band values excerpt for 1000 queried sensors, where green dots indicate high speeds, yellow dots moderate speeds, and red dots low speeds, retrieved from MyTransport DataMall [38], background image taken from [30]

3.2.3 Spatial and Geographical Data

Spatial data is another category of auxiliary data, aiding to improve statistical evaluation and especially to validate and adjust flawed altitude values of recorded GPS trajectories. This section introduces the retrieval and pre-processing steps of different queried data sources, comprising various Singapore's zones and terrain's altitude profile.

Singapore's Regions, Planning Areas and Subzones

Other than linking a GPS points to a road segment as done in the map-matching process, it is more robust, faster, and still effective to link a GPS point to a bounded area. Here, the provided division into zones by the Department of Statistics is taken. Singapore is officially portioned into three categories - arranged in the level of detail: *Regions*, *Urban Planning Areas (URA)*, and *Subzones*. Herein, a Region is the most coarse-scaled and a Subzone is the most fine-scaled division, and each successor is a subset of the predecessor. In other words, a Region includes a number of Planning Areas which in turn consists of a number of Subzones. In total there are five Regions, 55 URAs and 322 Subzones received according to [39].

Alike the retrieval of taxi stands, the data comes in the format of Shape files requiring a conversion into GeoJSON to finally import those into Matlab. In Appendix K, a map illustrating those categories' boundaries is plotted.

Elevation Map of Singapore

An elevation map is in principal a three-dimensional meshed grid of latitude and longitude coordinates for x- and y-axis, projecting terrain's elevation values on the z-axis. This section introduces the retrieval and pre-processing of such an elevation map for the purpose of re-assignment of recorded sample points from employed GPS loggers, which are very unreliable and often produce

wrong or impossible measurements. It aims to replace these height values with those obtained from a different source: the Google Elevation API [40].

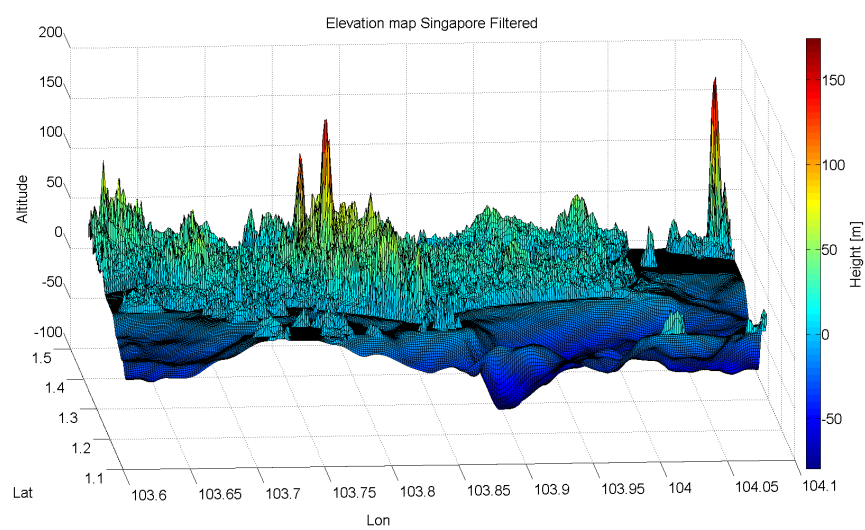
To construct the elevation map, the area of Singapore is initially portioned into a meshed grid of size 200×300 . For each of the resulting 60,000 unique anchor points, the API is queried and values are retrieved and stored into an equally sized Matlab matrix. Google periodically obtains these values from satellite measurements, which are probably simultaneously taken during Google Maps image recordings. The measurements do not include heights of buildings or bridges but mirror the terrain's elevation and hence perfectly suits this study's needs.

In detail, a Google Developer account was created and a project via the Google Developers Console [41] registered, while acquiring an API key to access the web service. Due to a limitation of only 2,500 queries per day in private usage mode, several Google accounts were combined to speed up the entire data retrieval process. With the API key and mandatory latitude and longitude fields, an HTML request is assembled with the base address `https://maps.googleapis.com/`, concatenated with the web service address `maps/api/elevation/json`, and finally passed with required parameters `locations=1.29,103.13` and `key=AIzaSyD4`, for instance.

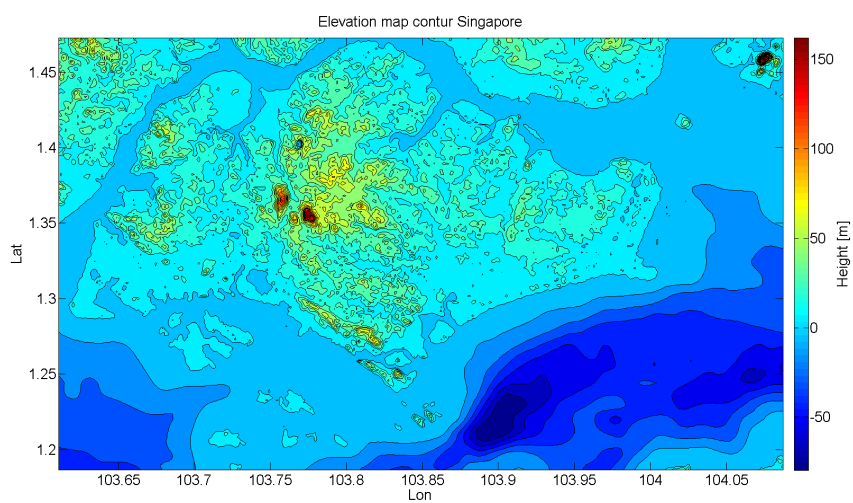
The altitude profile is a substantial factor determining the power to overcome gradient resistance forces, which in turn influence the energy consumption to a high degree - especially for hilly trips. Because Singapore is considered as a rather flat country, it might diminishes this impact but is nevertheless influential and hence included in this work. Its application finds expression in the substitution of altitude values for each sample point according to the constructed elevation map and, in this way, simultaneously prohibits from computing wrong gradient resistances from wrong altitude values.

The raw elevation map translates, according to boundary definition and spatial extent presented in Table 3.5, to a resolution of 159 m in North-South and 176 m in East-West direction. For the subsequent height assignment step, which substitutes flawed altitude values on the layer of sample points, this resolution turned out to be too low and requires an additional interpolation step to gain intermediate points. This results in an increase in latitude and longitude stretch by factor 20, in turn leading to a meshed grid of 4000×6000 anchor points and a resolution of about 8 m in North-South and 9 m in East-West direction. The higher resolution suits mentioned requirements better.

An interpolation runs the risk of overemphasising outliers or unreasonable values, which are obtained in trust of the Google Elevation API's data quality. Consequently, an ordinary mean filter of window size 20 is applied, which smooths the data. The result and eventually utilized elevation map is illustrated - together with a contour plot - in Figure 3.10.



(A) Elevation map



(B) Elevation contour map

FIGURE 3.10: Elevation and contour map of Singapore, retrieved from the Google Elevation API as of May 2014 [40]

Chapter 4

Driving Profile Analysis - Methodology

The driving profile analysis unit of this thesis comprises descriptions of methods, algorithms, and implementations to analyse driving profiles in detail. A driving profile is a recorded sequence of spatial, speed, and altitude values subject to time.

This section begins by emphasizing the data import and data handling, which became more important the more data was recorded, and continues with the pre-processing steps by immersing into GPS immanent inaccuracies, which require mitigation at the layer of sample points. Subsequently, trip and shift extraction methods are introduced and the vital map-matching process for the contemplated context is elaborated.

As an attempt to uniform the used vocabulary, frequently occurring expressions are defined in the following to work on a unambiguous set of terms and prohibiting inconsistencies.

Sample Point A *sample point* consists of a spatial part with latitude, longitude, and altitude fields, a dynamic part with speed and heading fields, and optional fields like status, road segment id, road type, and Subzone id. It is the atomic building block for the logging and booking dataset architecture.

GPS Point When focusing solely on a sample point's position, the term *GPS point* is often used to indicate the mere interest into the spheric coordinates latitude and longitude.

Trip A *trip* is an array of connected sample points which generally follow one distinct taxi status and therefore one driving purpose. Trips' distances typically vary between zero to 40 km and last for one to 60 minutes, heavily dependent on the taxi's status and whether there is a passenger on-board or not. In Section 4.3.1 the trips are elaborated in greater detail.

Shift A *shift* contains a number of trips for one taxi and one driver. It is meant to be a closed stint which normally occurs once per day per driver. A typical duration for a shift is eight to twelve hours, which, however, is subject to high variance and extensible up to 16 hours as it has been recorded. Shifts are further examined in Section 4.3.2.

Driving Profile A *driving profile* essentially defines the same as a trip does, but it rather focuses on the dynamic trend as on the position. It comprises a sequence of time, speed, and altitude values.

Dynamics With *dynamics* the traffic and vehicle regarded dynamic values speed, acceleration, and deceleration are meant.

Road Type A *road type* is a category of roads as defined in OSM and is introduced in greater detail in Section 6.2.

Road Segment A *road segment* is basically a road or street as mapped in OSM which is restricted to have no intermediate intersections with any other road except on its very first or very last edge.

Micro Trip A *micro trip* is considered as a subset of a trip's sample points. It can basically originate from three sources: (a) micro trips as an extract of a normal trip with targeted duration of roughly 210 seconds, defined by boundary conditions below a certain speed and acceleration threshold (see Section 4.5.1); (b) road segment based micro trips include an array of connected sample points on a single road segment; (c) road type micro trips which are essentially the same as previously described but are split at the transitions between two different road types and not at road segment transitions; thus they are usually longer.

The overall goal of the driving profile analysis is to extract a set of shifts comprising a number of driven shifts per taxi, each in turn including a set of trips of an array of sample points. The class diagram presented in Figure 4.1 illustrates the structure and interconnections among each entity.

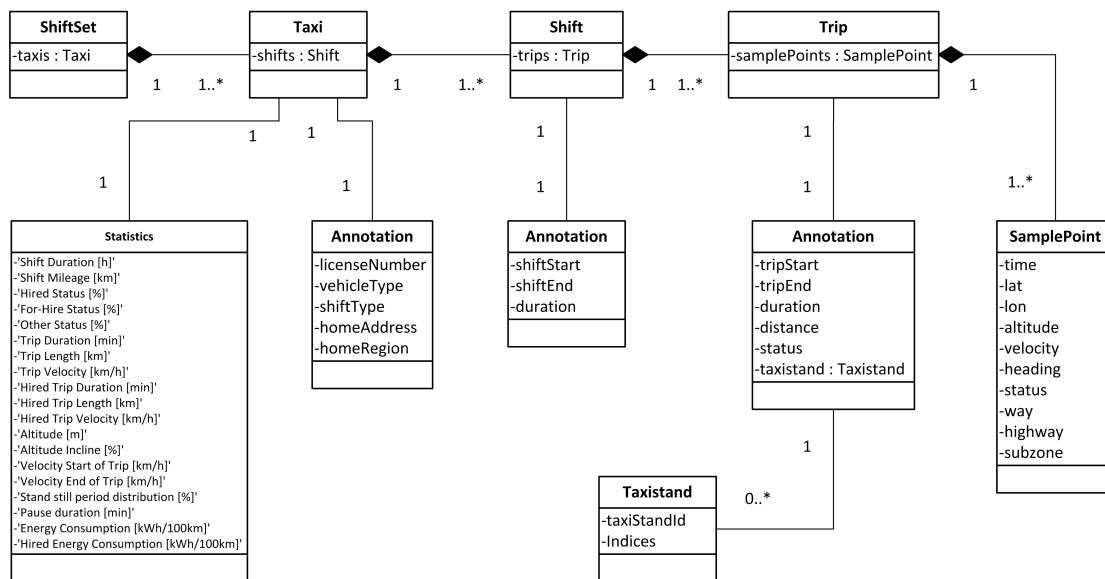


FIGURE 4.1: Class diagram for *ShiftSet* structure, including *Taxi*, *Shift*, *Trip* and *SamplePoint* and its interconnections

4.1 Data Import and Handling

The data import is an essential step for transforming table formatted, raw logging and booking data received from GPS loggers and MDTs, respectively, into Matlab. Since all subsequent processing steps are carried out in Matlab, this interface has to be treated with great care. As introduced in Section 3.1, for each of the 20 participating taxis a single GPS logger is equipped with a single MicroSD memory card. For faster exchange and data retrieval, another 20 spare memory cards are in stock, which are alternated at each card exchange appointment. This exchange takes place on a monthly basis while keeping track of the mapping between taxi licence plate number and current plugged memory card.

On the logging side, all files are initially processed by copying the entire memory card content to a folder named alike the card's id and located on a local server. Next, the files are manually moved

into a folder structure presented in Figure 4.2 according to the tracked mapping of card id \rightarrow licence plate number, as well as according to the recorded month denoted by the files' timestamps. This folder structure eases later analysis, which is tailored to run on monthly chunks. Each taxi folder contains a set of subfolders, named according to the recorded month, which in turn contain a *log* folder, storing the actual raw CSV files and a couple of MAT files, caching processing steps as an interim output of developed programs. An example excerpt of a typical logging file in CSV format is presented in Appendix A.

A suggestion to further work is to automate the transfer step from memory cards to the utilized Matlab structure to prevent errors and to be capable of scaling the system to a higher number of tracked taxis.



FIGURE 4.2: Folder structure as organised in Matlab for booking and logging data

On the booking side, a similar structure is established, omitting the log folder and the cache files and just concentrating on the monthly booking files provided in the XLSM format, an Excel sheet with Macro capabilities. Those Macros originate from a required pre-processing step realised directly in Excel using a Visual Basic (VB) script. The script browses through an initially unordered list of all available booking sample points in a single sheet and extracts sets of points belonging to the same licence plate number. It then merges each set into an own sheet and names it alike the licence plate number. An extract of such a XLSM file before the pre-processing step is attached in Appendix B.

The import of logging and booking files is executed by utilizing the built-in function `textscan` for ordinary CSV files and the more powerful function `xlsread` for XLSM files. Both file formats are table based, which enables the import of a predefined set of columns of interest. The major difference in importing is the fact that the latter requires the desired licence plate number to select the corresponding sheet for the taxi booking set in the XLSM files.

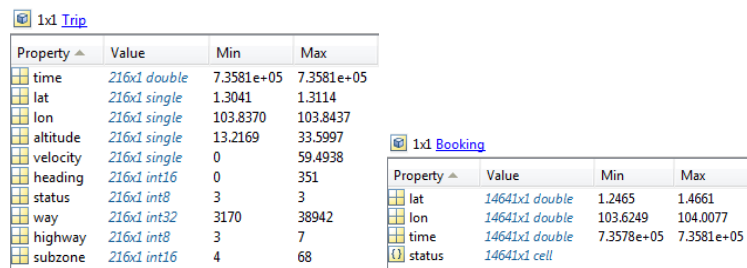
The actual import is straightforwardly performed in three steps: In case of logging data, for each recorded CSV file a new instance of class `Trip` is generated first. Second, each table column denoting one logging field is matched to the corresponding field of the trip instance. Third, each sample point is concatenated with an existing array of previous captured sample points. The booking import works identically, except for the different instantiation of objects of class `Booking` and its respective fields.

Unlike an object-orientated approach may would recommend, Matlab's strength of arrays is exploited, and each sample point's field is split into a single array. This is an essential and impactful software architecture decision and was undertaken after an unsuccessful approach to instantiate one object of a class `SamplePoint` for each actual recorded sample point, leading to an

overuse of memory space in Matlab. A disadvantage of this structure is the necessity to handle all trip's field arrays concurrently.

Figure 4.3 displays the outlined structure and used data types depicted in the second column. Virtually, there are only small differences in demanded memory space among the chosen data types. For instance, the data type `double` requires eight bytes, whereas a value encoded in `int16` only needs two bytes. This might not impact memory demand for small datasets; it is however impactful for huge datasets as it is the case in this work.

During the implementation phase, emerging challenges regarding shortcomings of memory capacities and inconveniences working with large Matlab MAT files were encountered. It was consequentially decided to set up a Microsoft SQL Database Server and upload all imported and processed sample points and its contextual related trips and shifts into this database. The resulting architecture essentially resembles the structure employed in Matlab (see class diagram in Figure 4.1) but was adjusted to suit the needs of the underlying Database Management System (DBMS). Most classes found hereby expression into tables. Further advantages of a DBMS are the opportunity for tailored and more fine grained search queries, eased statistics generation, as well as concurrent accessibility from multiple users while still acting on a unique data repository. The database's table structure is illustrated in Appendix C. Alike hash maps in Matlab, the database tables underwent indexing steps to speed up query times.



(A) Trip instance structure

Property	Value	Min	Max
time	216x1 double	7.3581e+05	7.3581e+05
lat	216x1 single	1.3041	1.3114
lon	216x1 single	103.8370	103.8437
altitude	216x1 single	13.2169	33.5997
velocity	216x1 single	0	59.4938
heading	216x1 int16	0	351
status	216x1 int8	3	3
way	216x1 int32	3170	38942
highway	216x1 int8	3	7
subzone	216x1 int16	4	68

(B) Booking instance structure

Property	Value	Min	Max
lat	14641x1 double	1.2465	1.4661
lon	14641x1 double	103.6249	104.0077
time	14641x1 double	7.3578e+05	7.3581e+05
status	14641x1 cell		

FIGURE 4.3: Internal trip and booking instance structure in Matlab

4.2 Trajectory Pre-Processing

The trajectory pre-processing covers the areas of combination of booking and logging data sources, filtering, and interpolation. All three steps are of high importance for the subsequent energy demand analysis as they bring the raw sources into a cleaned, enhanced, and utilizable form.

The combination process, which merges both sources and attaches status tags to sample points, primarily affects the later trip extraction, whereas the filtering and interpolation steps do also have an impact on the energy estimation results. This means, when immersing into latter mentioned methods in following sections, it is operated at the lowest layer of actual sample points as the atomic parts. The dynamic interactions among those, in turn, are the foundation of the proposed estimation approaches. Hence, these steps are influential and carried out with great care.

4.2.1 Combination of Logging and Booking Data

To exploit the full power of available trajectory related data sources, a combination of logging and booking data is mandatory. Since both sources are equipped with the same timestamp and

position fields, they can easily be merged. This status tag is in particular vital for the extraction of trips, which are usually bounded by the status revealing the taxi's driving intention.

The algorithm principally searches for equal timestamps and attaches the status information gathered from a booking sample point to the according logging sample point. However, it is required to extend this algorithm to a broader moving window technique which compares both timestamps within a tolerance band of ± 30 seconds. This is because both sets could contain gaps as sample points are not unconditionally always available or do not always follow the expected sampling rate throughout the whole trip.

The employed framed window attempts to search for possible corresponding logging sample points in the vicinity of booking sample anchor points. These contain the demanded status information and undergo a lower fluctuation in sampling rate; it is thus more reliable to use those as anchor points. The algorithm behind combining both data sets is outlined in Figure 4.4, where a number of three status tags from booking sample points are matched to the corresponding logging sample points in terms of their timestamps.

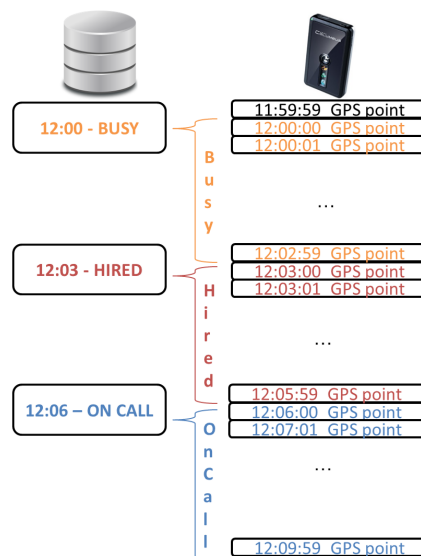


FIGURE 4.4: Outline of the combination algorithm for logging and booking data for an excerpt of three booking sample points, where each status is matched to the corresponding logging sample point

In case of a match, the algorithm attaches the booking sample point's status to the logger's sample point counterpart and additionally fills the range to the next available status tag with the same status. This way, oversampling of the logging set is inevitably, which, however, is tackled in subsequent processing steps particularly in the status based trip extraction.

Before the actual combination algorithm is applied, the general matching of booking and logging data is validated as shown in Figure 4.5. Here, in both cases, all active times are coloured in grey, whereas occurrences of more interesting recording gaps are coloured in green (for the booking set) and red (for the logging set). It gives a basic estimation if both sets belong together and throws an exception if active times deviate to a high degree. This is reasonable because a large part of previous conducted steps included manual processing, which is commonly more susceptible for flaws.

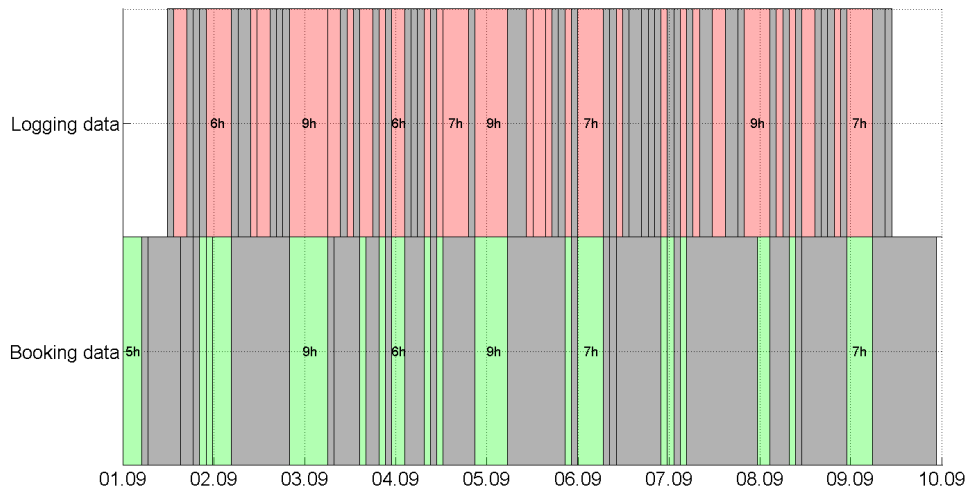


FIGURE 4.5: Active periods of logging and booking data, showing active logging periods and gaps in upper chart and booking periods and gaps in bottom chart

4.2.2 GPS Trajectory Filtering

GPS trajectory filtering comprises steps towards transforming raw and erroneous input trajectories into a sufficient quality for further processing steps. There are a number of typical error sources which are counteracted by means of two categories: *Outlier Filtering* and *Dynamics Filtering*. The former focuses on erasing spatial and dynamical anomalies such as flawed sample points. The latter targets on preparing the trajectory's driving profile for the subsequent energy demand analysis.

Both categories are of high importance since outliers and dynamic inaccuracies have a deteriorating effect on energy estimation quality. For instance, an unreasonably high acceleration or speed, mistakenly recorded by the GPS logger, can lead to an unreasonable high energy consumption. In general, the following problems and deviations are faced requiring specific mitigation efforts.

Outliers Outliers denote sample points with unreasonable values regarding position, speed, altitude, or heading, caused by poor a GPS signal quality. It comprises single sample points way off the Singaporean boundary, sample points exceeding physically possible speed or acceleration values, and altitude values exceeding the highest elevation of Singapore.

Gaps Gaps are missing values within a GPS trajectory, often occurring as a consequence of poor GPS satellite coverage or tunnel transits. Typically, values close to the gap's boundary (e.g., tunnel entry and exit) are also of poor quality.

Drift With drifting, a GPS signal is termed which records a movement while the car actually remains at its position. This is most likely to appear after a stint with high speed followed by a sudden standstill period.

Multipath Multipath means that the GPS device receives multiple signals simultaneously from the same sender resulting in a position mismatch, mostly due to surrounding mirroring areas such as high buildings in the CBD and is thus particularly challenging in Singapore.

An excerpt of an example trip where occurrences of all four deviations are marked is presented in Figure 4.6.

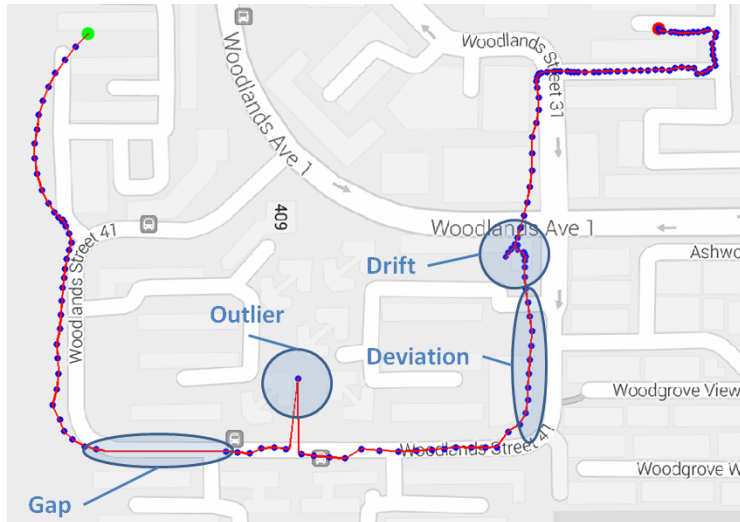


FIGURE 4.6: Possible GPS immanent inaccuracies, deviations, and erroneous points, background image taken from [30]

Besides trip specific filtering methods, which operate at the layer of sample points, filters which treat outliers regarding the entire trip are also applied. Those erase trips which have either less than a threshold of 30 points or have recorded a period less than 30 seconds. In the subset of all trips of July, this filtering step removed 9.6 % of all entailed trips.

Outlier Filtering

Outliers are tackled by introducing thresholds expressing maximum or minimum possible physical constraints. Sample points are tested against these thresholds, and over- and undershoots are mitigated by simple elimination. In the following, for each filter, the share of erased points is given related to a trip subset of July with 26,047 trips. It is differentiated between spatial and dynamic outliers.

Spatial outliers are comparably easy to detect as boundaries of Singapore are predefined and the only task is to check a sample point's position to fit within this area as illustrated in Figure 4.7. This step erased 0.06 % of all sample points of the considered test set, which is a rather small amount compared to other filtering methods but would have a very disturbing influence if left untouched.

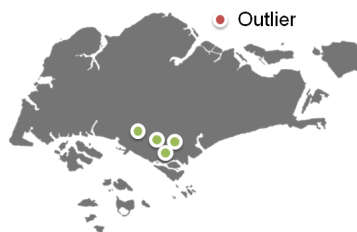


FIGURE 4.7: Spatial outlier exceeding Singapore's borders

The more comprehensive dynamic outlier filtering comprises mitigation approaches to detect an erroneous sample point's position by exploiting its dynamic behaviour. It shall be highlighted that the methods at this stage do not tackle the actual dynamic values as recorded from the GPS logger (the Speed field), but they solely use independently computed dynamics from GPS coordinates. In

that sense, an outlier is defined as a sample point whose position is physically impossible respect to its predecessor or successor, exceeding physically constrained speed or acceleration values.

Those speed and acceleration values are computed first, by calculating the distance between two GPS positions denoted by longitude and latitude coordinates, and second, by computing the first derivative $v = \dot{s} = \frac{ds}{dt}$ for speed and the second derivative $a = \dot{v} = \ddot{s} = \frac{d^2s}{dt^2}$ for acceleration, respectively. The term ds herein denotes the distance and dt the time between two considered points. Figure 4.8 outlines the basic principle behind dynamic outlier filtering and highlights the areas where values exceed established thresholds.

A more detailed view on the performing algorithm is described later in this section when immersing in the dynamics filtering; both outlier and dynamics filtering use the same algorithm. A share of 15.2% erroneous sample points for the considered test set could be detected, which erased a remarkably great deal of sample points.

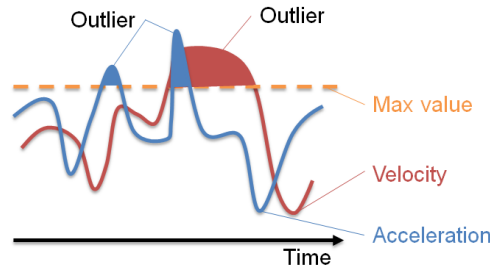


FIGURE 4.8: Dynamics outliers exceeding acceleration and speed thresholds

To work on GPS coordinates in the environment of the more commonly used metric system, the spherical coordinates have to be projected into a utilizable Euclidean coordinate system for employing metric measurements. Since the Earth shapes alike an ellipsoid, a mathematical correct result for computing the distance of two spherical positions delivers the Haversine Formula as presented in Equation 4.1 and taken from [42] .

$$d = R \cdot 2 \cdot \arcsin \left(\sqrt{\sin^2 \left(\frac{\phi_2 - \phi_1}{2} \right) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2 \left(\frac{\lambda_2 - \lambda_1}{2} \right)} \right) \quad (4.1)$$

- d ... Euclidean distance between two sample points in spherical coordinates
- R ... Earth radius
- ϕ_1, ϕ_2 ... Latitude of point 1 and latitude of point 2
- λ_1, λ_2 ... Longitude of point 1 and longitude of point 2

However, a simpler Flat Earth model in form of the Pythagorean Theorem as denoted in Equation 4.2 can be applied for small distances, effectively decreasing computation costs where the calculation is occurring very frequently as it applies in the regarded scenario. As discussed in [43], for distances less than 50 km, which is the maximum expected trip distance in Singapore, the error would be negligibly low with less than 24 m. This is determined by the proximity to the equator and increases with greater remoteness.

The spherical coordinates (unit: degree $[\circ]$) are simply transformed into the Euclidean coordinate system (unit: metre [m]) by multiplying both latitudinal and longitudinal projected distances $\Delta x = x_2 - x_1$ and $\Delta y = y_2 - y_1$ with a constant factor in the observation unit m/\circ - one factor per case (see Appendix H). The factors are assumed to be constant since trips are

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (4.2)$$

- d ... Euclidean distance between two sample points in Euclidean coordinates
 x_1, x_2 ... Points in Cartesian coordinates for longitudinal projection
 y_1, y_2 ... Points in Cartesian coordinates for latitudinal projection

considerably short and Singapore is very close to the equator with a latitude value close to zero; both facts diminishing the effect of the Earth's ellipsoidal shape.

Dynamics Filtering

The prior described outlier filtering aims at eliminating erroneous sample points solely based on spatial information in form of the GPS coordinates. In the dynamics filtering step, the sample point's position is now expected to be correct in terms of physically feasibility, and the dynamic values are analysed as they are directly obtained from the GPS logger.

This additional filtering is vital for the subsequent energy demand analysis, and hence the trips carefully undergo a twofold filtering process. First, likewise the dynamic outlier filtering, speed and acceleration values are filtered which exceed physical constraints. However, this time, those values are not computed manually but are directly taken from the GPS logger's recorded speed profile. Merely the acceleration cannot be obtained directly from the logger and is thus computed as the first derivative of speed subject to time ($a = \dot{v} = \frac{dv}{dt}$).

The algorithm is essentially the same as the one used in the outlier filtering and follows the flowchart in Figure 4.9 according to relations and definitions in Figure 4.10. Applying this filter, a share of 7.4% erroneous sample points for the considered test set could be detected.

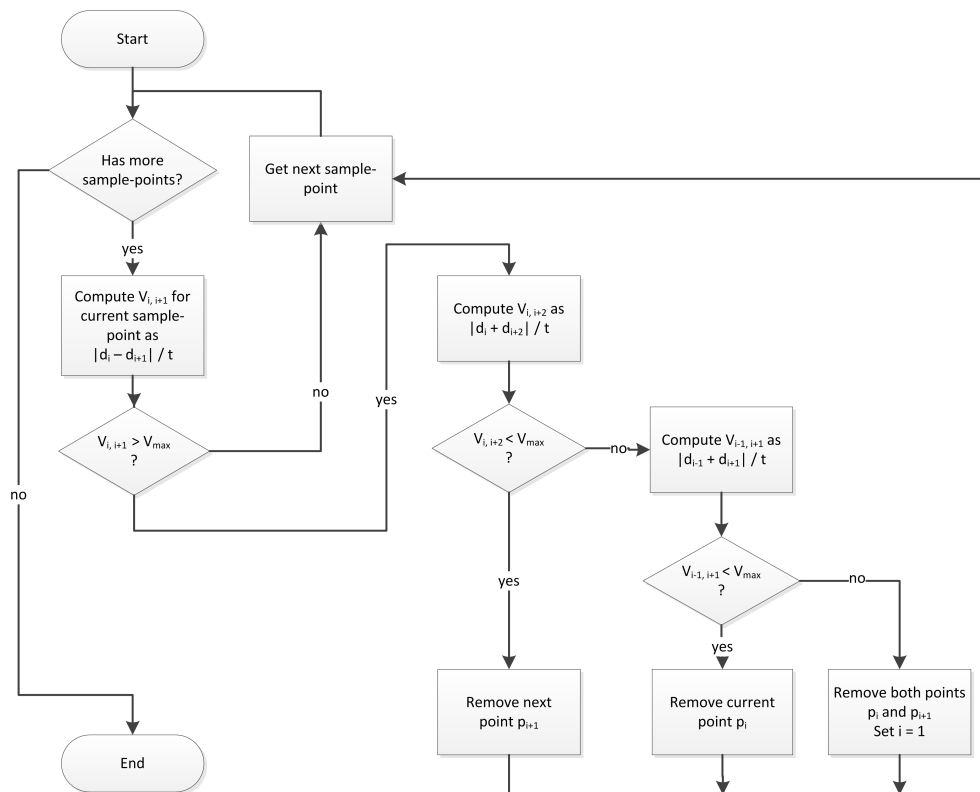


FIGURE 4.9: Dynamics filtering algorithm flowchart to detect and eliminate physically impossible sample points regarding speed and acceleration

Utilized relations are explained in Figure 4.10 with p_i denoting an actual sample point, $v_{i,j}$ the speed from point i to point j , and $|d_i - d_j|$ the Euclidean distance from point i to point j .

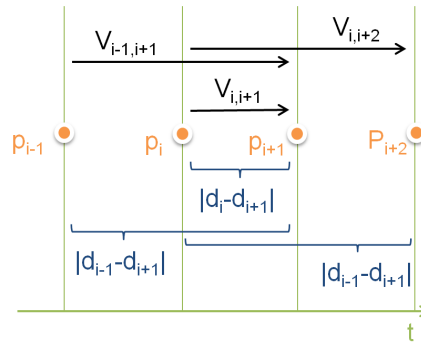


FIGURE 4.10: Distance and speed relations for the dynamic filtering algorithm

After removing a number of erroneous sample points employing both filtering steps, still a number of inconsistencies in trips' speed profiles were encountered. Comparing those to driving cycles, CAN Bus values, and previously recorded trips, a higher fluctuation in speed values is noticed than one would have expected. Therefore, a decision for a third filtering step was made which additionally applies a simple Gaussian filter of window size five to give the speed profile a little more smoothness. In the result part in Chapter 5, this choice is justified by comparing average acceleration and deceleration values as a benchmark for fluctuation.

Regarding further dynamic values like heading and altitude, no action is taken in the filtering step since heading is just of minor importance and is already of decent quality. Altitude values are tackled in the following interpolation step in Section 4.2.3.

4.2.3 GPS Trajectory Interpolation

An interpolation is the process of substituting gaps of missing or disarranged values in terms of disobeying the targeted sampling rate. The necessity for interpolation origins twofold: on the one hand, the preceding filtering steps erased certain sample points and generated gaps; on the other hand, incomplete sample data is likely provided due to GPS logger inconsistencies. In general, gaps are acceptable because each sample point also includes a timestamp which unambiguously links positions and dynamics to time. It allows for simply connecting gap's border points; this resembles the method of a linear interpolation.

However, performing a more comprehensive interpolation including additional inputs can improve the quality of the output trajectory. A further reason is the facilitating effect on subsequent computation steps when a fixed sampling rate can be guaranteed.

There are basically two different interpolation methods available: continuous and discrete interpolation; former is applicable for continuous signals like GPS coordinates, and latter is utilizable for discrete values such as status information.

Interpolation of Gaps and disarranged Sample Points

To interpolate gaps, it is proceeded in two steps. First, a targeted time vector is established. That is in principle the duration from first to last sample point with a sampling period of one second. This rate is reasonable since GPS trajectories are recorded alike. The process of defining a new sampling rate, or in this context's case rather repeatedly applying the same rate once more,

is called re-sampling. The employed algorithm basically immerses into each point and checks whether interpolation is necessary or whether a suitable point in terms of fitting of sampling rate is already present.

Once the time vector is defined, all fields of unsuitable sampling rates or gaps are estimated using various interpolation techniques as described in the following. As a result, each sample point follows the predefined sampling rate and gaps are dismissed. In the actual implementation, Matlab's built-in `interp1` function is utilized. Figure 4.11 schemes this process comparing a raw with an interpolated trajectory.

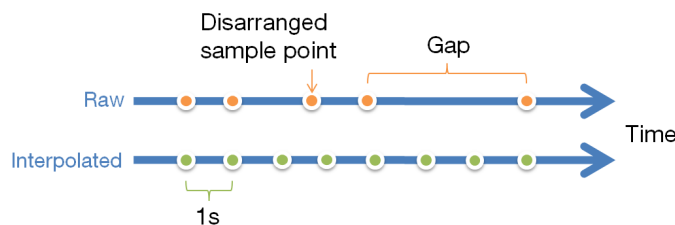


FIGURE 4.11: Interpolation of gaps and disarranged sample points

A special interpolation method is demanded for large gaps like tunnels. Here, it is not reliable to just consider values straight before and after the tunnel, because those tend to have a high uncertainty. Rather, a more comprehensive method is applied: initially, the actual length of the gap is calculated by GPS coordinate based distance computation; afterwards, the average speed for this portion is retrieved; finally, the interpolated intermediate points are tagged with this speed value. In the transition areas, acceleration and deceleration constraints are obeyed by deriving a slope towards and from the targeted mean speed value to gain a smooth transition. In the result part in Chapter 5, a tunnel example which underwent this interpolation method is presented.

Linear Interpolation Exemplified, linear interpolation draws a line to connect the gap's border points. The border points can either be just the two last known adjacent points, risking inaccuracies, or the mean of a number of border points to allow for a smoother transition and more robust linear interpolation.

Spline Interpolation A “spline is a numeric function that is piecewise-defined by polynomial functions, and which possesses a sufficiently high degree of smoothness at the places where the polynomial pieces connect” [44, p. 225]. It is a reasonable choice for floating values where smoothness is an important requirement. It is successfully applicable for interpolating GPS coordinates but has its downsides for interpolating speed and height, where the high emphasis on smooth transitions may lead to negative speed or altitudes.

Pchip Interpolation The method of Pchip (Piecewise Cubic Hermite Interpolating Polynomial) acts similar to the spline interpolation, but the resulting curve restricts each “cubic to be monotone in an interval. [...] The curve produced contains no extraneous ‘bumps’ or ‘wiggles’, which makes it more readily acceptable to scientists and engineers” [45, p. 238].

Nearest Neighbour Interpolation Nearest Neighbour interpolation is the simplest of all introduced methods, assigning the value of the nearest neighbours to an interpolated point. It is in particular applicable and feasible for discrete values like the status and road segment field.

Altitude Assignment

In the altitude assignment step, flawed altitude values gathered from the logger are substituted with more reasonable values obtained from the Google Elevation API. Although Singapore is relatively flat and the elevation impact on energy consumption is fairly small, this step is necessary to avoid wrong conclusions when computing with erroneous altitude values.

A comparison of a sample trip before and after the assignment step is plotted in Figure 4.12. The green line representing the raw values as gained from the GPS logger reveals a high fluctuation and instinctively has an unlikely high number and magnitude of hills and valleys. In contrast, the blue line, visualizing the assigned values, behaves smoother and is less undulating. Both charts, however, meet at a similar average altitude. Although the assigned values cannot be considered as the ground truth without any exception, evaluation results in Section 5.1.2 build confidence that assigned values reflect the reality way better than the raw sample point's altitude values from the GPS logger do.

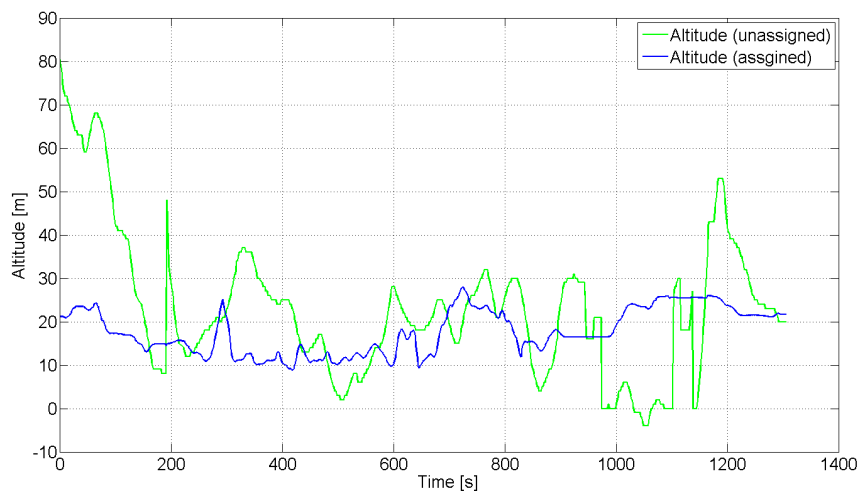


FIGURE 4.12: Comparison of altitude values for GPS logger and elevation map

Other approaches exploit the relative average height value obtained from all trips and do not incorporate further resources as done here. For instance, in [29, p. 12], the same GPS logger was used in a comparable set-up, following an alternative approach to obtain all recorded trips' altitude values for each cell of a meshed grid and extracting the considered ground truth elevation by forming the average.

The implemented altitude assignment algorithm basically goes through each sample point's position and, in a first step, locates the point on a meshed grid of Singapore. As discussed in Section 3.2.3, elevation values have been retrieved via the Google Elevation API for anchor points of an established grid of size 4000×6000 (after interpolation). It then computes the Euclidean distances to the four adjacent anchor points which are used as weights. An example of a sample point (green) with its four anchor grid points (orange) is observable from Figure 4.13. The closer the anchor point to the sample point, the higher its influence is weighted. Those four weight values - normalized to unity - are finally incorporated into the actual calculation of the sample point's altitude, which sums up all anchor points' altitude values multiplied with the weight factor.

It was further elicited whether the original GPS logger altitude values should flow into the assignment step by weighting both inputs according to a ratio. However, it was refused from doing so since Google Elevation API values are already very satisfactory.

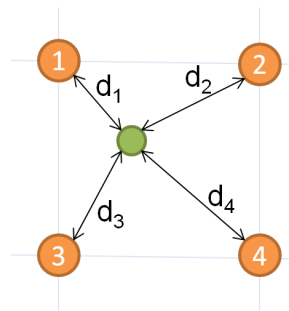


FIGURE 4.13: Anchor points (orange) for altitude assignment of a sample point (green), weighted with distances d_i

4.3 Trip and Shift Extraction

All previous described pre-processing and filtering steps are specific for each trip and operate at the layer of sample points. To be able to consider trips and shifts as contextually reasonable bounded portions with an interrelated array of sample points, an extraction is required and presented herein. Trip extraction not just eases and restricts dynamic filtering, but it also sets map-matching bounds and is of significance for statistical evaluation.

First, several trip extraction methods are examined operating in the transition between sample point layer and trip layer. An initial, implicit trip extraction was already done on data import, where each CSV file translated into exactly one trip. These stints, however, are merely split at transitions from loggers' on to off mode and thus rather cover a too large amount of sample points as would be suitable for one trip.

Afterwards, the shift extraction, which operates between trip and shift layers, is introduced. Here, the objective is to arrange trips into a suitable contextual and time bounded stint, which is denoted as a shift.

4.3.1 Trip Extraction

Various methods for extracting trips from a huge array of sample points are realised. The basic proceeding, which is common among those methods, is depicted by finding valid edge indices for reasonable cutting points according to obliged conditions. Its objective is to split big trips which do not reflect the reality into two or more smaller trips as they are most likely to have occurred in the recorded scenario.

Arranged in the order of importance, implemented extraction methods are fundamentally based on: *Status*, *Long Distant Trip*, *Taxi stand*, *Time Difference*, and *Standstill Period*. Restrictively, extractions are founded on various empirically determined parameters and are subject to change as they are driven by new outcomes and learnings. Analysis done in related studies (e.g., [29]) and feedback from SMRT set a basic parameter framework, which found solidification in the implemented programs as parameters listed in Appendix H.

Status Based Extraction

As the major extraction method, the status based trip extraction fully harvests both the logging and the booking data derived from a previous combination. The status itself precisely offers a means of trip's intention, whether a passenger is on board, a taxi is used on private purposes, or

even whether the taxi is active at all. It is the fundamental criterion for trip cutting and leads to the definition of a trip as a closed unit with defined purpose. Hence, a trip is mainly considered as having exactly one status.

The gathered status information is periodically recorded approximately every three minutes, which does not straightforwardly reveal the exact actual status change times but leaves an uncertainty of ± 3 minutes.

For instance, in the case illustrated in Figure 4.14a, three statuses have been captured at times 3:00, 3:03 and 3:06 - one Hired status enveloped by two For Hire status. The actual times a passenger boarded and alighted the taxi and hence caused the status change is, however, at times 3:01 and 3:05, respectively. The sole information that a passenger trip has occurred in between both status changes does not imply when this exactly (by means of a resolution of seconds) happened. Further examinations are thus essential to find the exact status change times.

Another issues is likely to arise when two status changes have occurred within a short time bound, undercutting the three minute sampling rate as displayed in Figure 4.14b. In the examined case, the actual status changes at times 3:01 and 3:02 are not caught by the booking database since the MDT notifies the back-end system at inappropriate times (3:00 and 3:03), exceeding the time the intermediate status was prevalent. This case frequently occurs, e.g., at favoured taxi stands at peak times where a taxi drops off a passenger and immediately picks up another one. This is why the developed extraction methods also have to handle this issue. It resulted in the introduction of an additional trip extraction step for long distant trips as explained later.

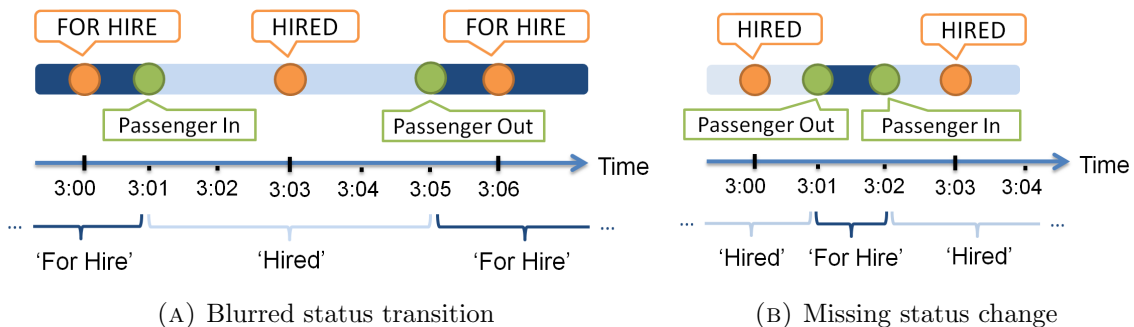


FIGURE 4.14: Status change extraction challenges for over- and undersampling of status changes

To find the most likely actual status change points as delineated in Figure 4.14a, four methods are defined and utilized using a combination of *Stand Still Periods*, *Taxi Stands*, *Most Distant Points*, and *U-Turns*. Those are applied on a small trip's subset between the second last and last known status of roughly three minutes, identical to the booking sampling period between both neighbouring status anchor points.

All four methods, individually contemplated, merely unreliably and inaccurately indicate an estimation candidate. Therefore, the power of a combination of all methods is exploited by assigning weights to each method's resulting candidates; the one with the highest value as a summation of all partial weights is eventually chosen. Applicable for all methods, it is always attempted to take the lowest speed at status change candidates if no real standstill period could be observed, as displayed in the case of Figure 4.15a.

Stand Still Period Most of the status changes occur during a standstill period of the taxi, be it a passenger pick up, drop off, or a the beginning or end of a break. The information about the standstill periods is thus of major interest and observable from Figure 4.15b. However,

not all status changes took place in the vicinity of a stand still period (e.g., a For Hire \rightarrow Busy change), why this is not a necessary requirement.

Taxi Stand Taxi stands are frequently visited by taxis to pick up or drop off passengers and thus play an important role for status changes of type For Hire \rightarrow Hired or vice versa. In combination with a minimum stop time threshold of 10 seconds, it is a very good indicator for a status change candidate. To determine whether a taxi has reached a taxi stand, a circular neighbouring area of 50 m is defined and minor outliers are allowed to catch GPS immanent deviations.

Most Distant Point Another indicator of possible status changes is the consideration of the most distant points in the scope of the trip's extract, which leads to the case displayed in Figure 4.15c. It attempts to find the most distant point: either in terms of the biggest distance to the destination point, depicted in Figure 4.16a, or the biggest projected distance on a direct line (the 'bee-line') connecting origin and destination as displayed in Figure 4.16b. The rationale behind this method is that taxis tend to use the most direct route and avoid detours. If a most distant point occurs, it indicates a passenger pick-up or drop-off in its vicinity.

U-turn A U-turn detection attempts to find a U-turn pattern in a trip excerpt. It primary bases on the vehicle's heading of which a change of roughly 180° alludes to a candidate. Two restrictive issues arose: First, a U-turn is not imperatively defined by an exact 180° turn but was also observed for a slightly lower or higher value, eventually leading to a tolerance band of $\pm 30^\circ$. Second, logger's heading values, which have not been priorly pre-processed, are subject to noise and especially under poor conditions likely to fluctuate inappropriately. This justifies the introduction of another threshold determining a minimum time period in which a U-turn has to be detected. An example states Figure 4.15d.

Peculiarities for status changes are prevalent for the statuses Payment, Arrived, STC, and Unknown. The three first-mentioned indicate a trip of status Hired which is about to end soon or has ended recently. It would not make sense to cut trips along these status transitions because the trip's intention is essentially the same. The status Unknown is the initialization value and most likely persists if the match between booking and logging data was unsuccessful.

Long Distant Trip Extraction

From the example given in Figure 4.14b, the need for further mining approaches in case of undersampling of status changes can be derived. The corresponding trip split points cannot be detected by solely residing on the recorded status values. An extraction mode called the *long distant trip extraction* tackles trips of more than 10 km targeting at finding unrecognised trip splits.

It basically applies the same methods as introduced in the status based extraction in an adopted version, contemplating an entire trip rather than just a three minute excerpt. Particularly the U-turn detection requires a subsequent validation step because, in case of a whole trip, a U-turn is broader defined and rather translates to a 'direction change detection', nevertheless inheriting from the U-turn detection algorithm. A case example is plotted in Figure 4.17.

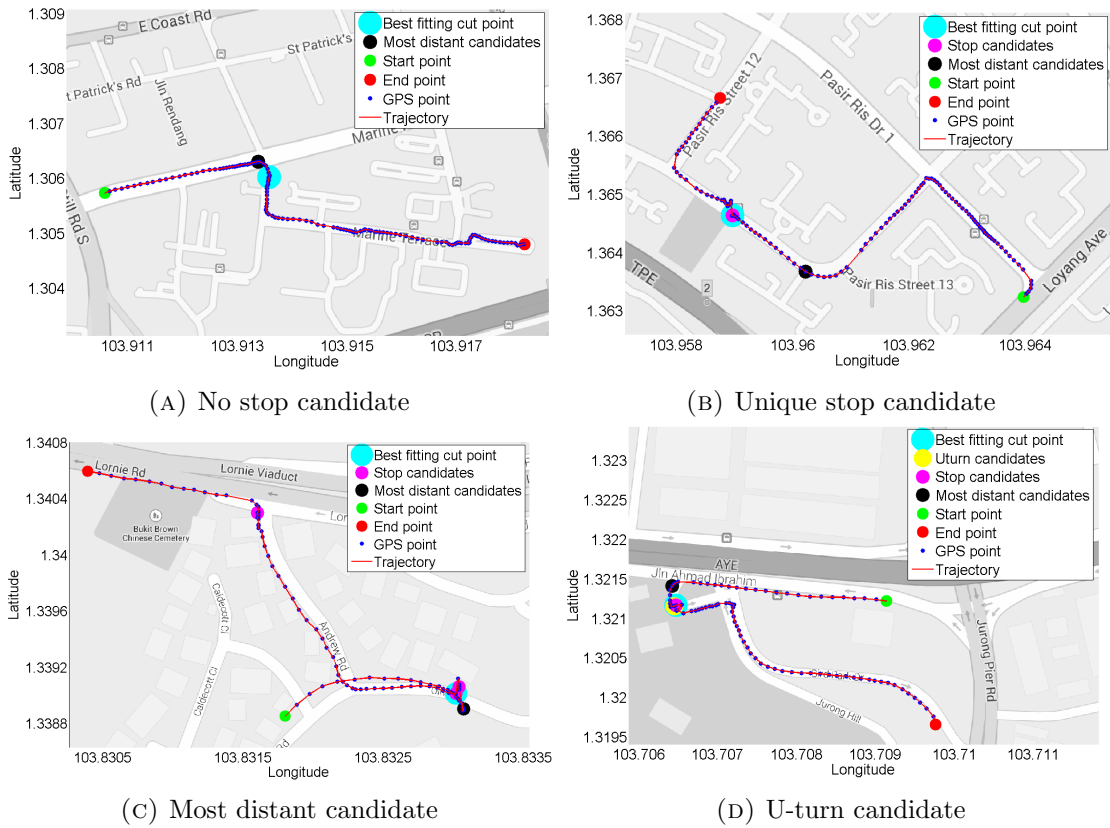


FIGURE 4.15: Status change extraction methods analysing inter-status cut candidates displaying various scenarios of candidate constellations, background images taken from [30]

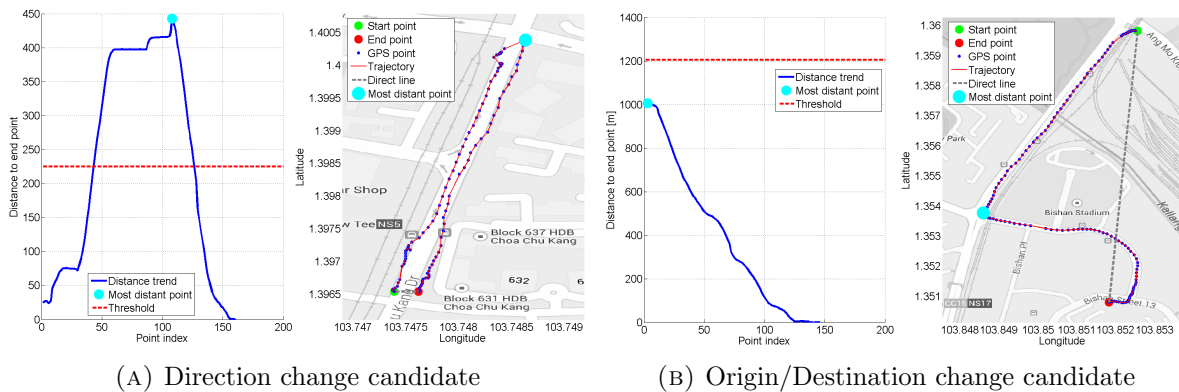


FIGURE 4.16: Most distant point analysis detecting status change candidates for two different scenarios, background images taken from [30]

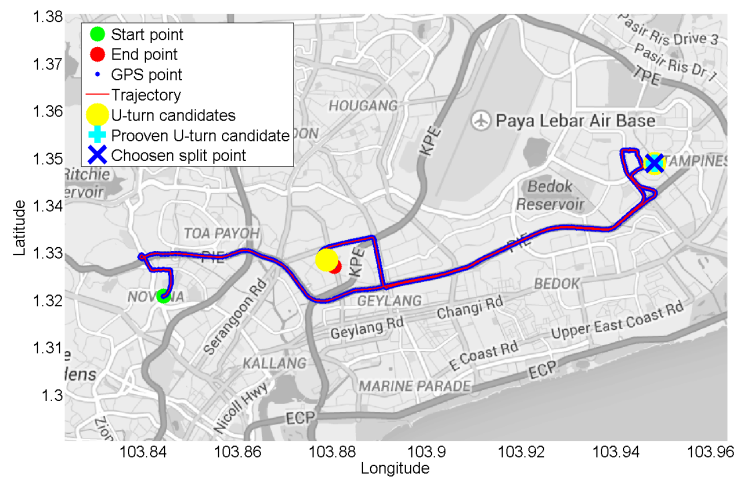


FIGURE 4.17: Extraction of long distant trips which were not covered by previous extraction methods, emphasizing U-turns as possible trip split candidates, background image taken from [30]

Taxi Stand Extraction

Although - in a fine-scaled manner - already part of the status based extraction, this extraction method additionally aids to detect trip splits which have been undersampled. Alike the taxi stand detection as a submodule of the status based extraction, it searches for taxi stands in the vicinity of the GPS trajectory and checks whether a stand still period obeys imposed taxi stand metrics. This is: minimum standstill time, maximum distance to taxi stand location, and outlier and maximum speed tolerances.

If a match is detected, the actual cutting takes place after the taxi has left the taxi stand. Given the example of a long queue in front of a favoured taxi stand, this decision is reasonable because the actual change in status (in this case For Hire \rightarrow Hired) has occurred after the taxi left the stand.

Time Difference Extraction

This method employs time differences in GPS trajectories indicating possible trip splits - especially for the case of undersampled status changes. Usually, sample points received from the GPS logger come in a sampling period of one second with minor deviations of ± 3 seconds. However, in some cases, despite an active recording mode, major gaps in GPS trajectories are encountered and accounted for by GPS satellites signal losses.

Typical cases are tunnel transits and pick-up or drop-off locations which are covered by a spanned roof, likely to occur at the airport and shopping malls or more seldom at hospitals and hotels. Tunnel transits, however, do not reveal any enrichment in localising trip splits since it is definitely part of the concerned trip. Here, two thresholds are introduced to catch the latter cases for possible trip splits and disregard the former. Since the *time difference extraction* method is admittedly quite unreliable, a rather defensive set of those parameters was chosen.

The thresholds comprise (a), a minimum required time for the GPS signal loss and (b), a maximum allowed travelled distance during signal loss. The first mentioned is set to 30 seconds. It was encountered that this value - solely applied - did not reveal the desired detection quality.

Hence an additional distance measurement was introduced, which starts measuring when the signal is lost and ends when it is regained. If this distance exceed the threshold of 500 m, it is considerably certain that a tunnel transit is present, rather than a pick-up or drop-off point.

Standstill Period Extraction

A trip extraction with regard to standstill periods targets at splitting trips after a comparably long time without vehicle motion. It is a less common, but yet occurring case and applicable either on driver's breaks with active GPS logger and ongoing recording mode or after waiting for a passenger. A standstill period is defined as the time a taxi stood still for at least five minutes. A distinguishing between commonly occurring waiting times (e.g., at a traffic signal), passenger waiting times (e.g., at taxi stands), and breaks or shift changes is crucial. A speed threshold of 1 km/h determines the maximum speed regarded as standstill.

4.3.2 Shift Extraction

Shift extraction aims at arranging trips into shifts according to time and context bounds, which is in principle alignment with one stint for one driver for one day. Its duration varies among the drivers and from day to day but normally lays in the limits of eight to 16 hours. This high variability is accounted for by different hirer schemes of one or two drivers, eligible to drive the same taxi, but also for the different interpretation and execution of these schemes.

As shown in concerned parts of Chapter 5, a two-hirer scheme does not unconditionally lead to a daily alternation in drivers but can also mean that the taxi is shared on a day to day basis; or even a combination of both is conceivable. This exacerbates the shift extraction analysis vastly because it is non-trivial to predefine a targeted duration per shift.

This section examines the shift extraction process in two distinct steps: first, analysis of underlying patterns pointing towards reoccurring shift change times and locations, and second, application of gained patterns on daily trips.

Shift Change Pattern Analysis

To extract shifts from a number of unrelated trips, reoccurring patterns in shift change times and locations are exploited. Shift changes are not recorded by any means, neither in the logging nor in the booking set. It hence is the task to estimate possible shift changes employing a set of four criteria: time of the day, number of hirers, location, and status distribution.

In a first step, the time of the day and the status distribution is considered by merging all monthly recorded sample points for each minute of the day subject to taxi states. The state is either active (all statuses except Log Off) or inactive (Log Off and gaps in logging data). This relation is plotted in Figure 4.18 and reveals the average active times for a taxi with 60 sample points as the possible maximum value as derivable from the one second sampling period. The black dots indicate possible shift change times, and both red dots illustrate the best fitting shift time combination.

The purpose is to gain possible and reoccurring shift change times, assuming that a shift change is defined where a taxi has status Log Off or at least is inactive for a certain duration. Those time thresholds are once more empirically examined. It is assumed that after two hours of inactivity a shift change must have been occurred, whereas a stint of more than 16 hours requires

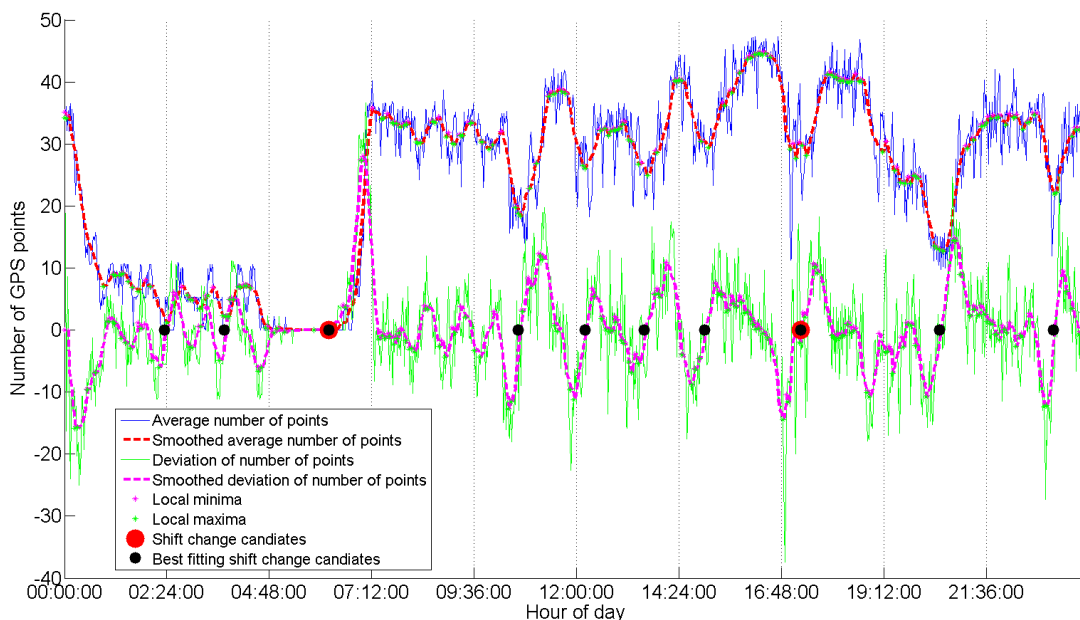


FIGURE 4.18: Analysis of shift change patterns by examining active daily recording times of a typical two-shift taxi, the blue line denoting the absolute active times and the green line denoting its derivative

a split into two or more shifts. These thresholds define basic boundaries in which further detailed analysis are necessary.

The algorithm, investigating shift changes on time dependency, basically aims at finding minima in active times or maxima in its derivative. It initially smooths the active times distribution to avoid a too neat detection of extrema. Secondly, it finds a limited number of minima, which are the basic shift change candidates.

Meanwhile, the derivative is computed, which supports the data mining. The resulting trend denotes the magnitude of transitions between active and inactive times and is thus a good indicator for relatively fixed shift change times. Alike the previous step, a limited number of maxima is now computed. Those maxima are input for a subsequent search for the best fitting pair with respect to average shift time as well as respect to maximum and minimum shift duration constraints.

The actual extraction algorithm starts with the highest derivative, computes time difference to the second highest value, and matches the result with the anticipated shift length. This anticipated value equals half the duration of all normally occurring active times for this specific taxi. As ‘normal occurring’ the average active times above 20% are considered to avoid impacts of noise. If the time difference matches to the assumed shift duration, the algorithm is finished and the possible shift change candidates are found. If not, it is continued with the distance between the first and the third highest deviation and so on. Should this again does not produce a match in any of these constellations, the highest derivative is dismissed and the process is repeated with the second highest value as a reference.

Eventually, the two absolute occurrence likelihoods which are closest to the retrieved derivative candidates are selected. Herewith, the findings are again projected to the actual scenario where a minima in active times denotes the best candidate. Nevertheless, initially extracting candidates on the basis of derivatives in the first step produced more reliable results as if solely the absolute distribution would have been taken.

By incorporating locations, it is anticipated that shifts are not just likely to occur at same times but also at same locations. This assumption is in line with empirical experiences, and they can even be extended to a preferred shift change at driver's home locations which are available from SMRT.

Extraction of Shifts and Post-Processing

The elaborated shift change candidates and examined patterns are now applied on a daily basis. Because shifts are likely to span over more than one day, always two days are considered - the previous and the current one. First, alike the candidate extraction, a break or gap of more than two hours, where a shift change could have taken place, is attempted to be found. In case of one-hirer schemes, it is easier to assign trips to shifts as there commonly happens only one shift per day. Drivers of two-hirer scheme taxis, on the other hand, tend to change shift twice a day which leads to usually two shifts daily.

However, as presented in the statistical evaluation part in Section 4.6, it cannot be assumed that each two-hirer scheme taxi always comprises two shifts per day. Furthermore, it seems that the underlying patterns are subject to change from day to day, exacerbating the data analysis. For instance, a two-hirer scheme taxi was observed which drove ten hours over three days a week and almost twice as long for the rest of the week. To aggravate things even further, one-hirer taxis do also reveal patterns indicating shift changes in a number of cases.

One mitigation approach to handle this outlined uncertainties and inaccuracies is to improve the previous pattern recognition with locations in the vicinity of already selected shift change candidates. Those are additionally applied to the actual shift extraction on a day to day basis. Moreover, recommendations for further investigations also include the driver's home addresses and more detailed statuses of last assumed trips. This, however, would have stepped beyond this work's scope and is thus not examined further.

The final step in the extraction of shifts is the actual assignment of trips to an instance of class Shift (see Figure 4.1), enveloping a number of trips into a contained array. A final post-processing step adjusts extracted shifts which have less than a minimum number of trips included or which exceed the predefined maximum duration of one shift.

4.4 Map-Matching

The process of map-matching has the purpose of matching sample points to an underlying road network. This is necessary since received GPS coordinates typically come with a Gaussian distributed inaccuracy of about ± 20 metres [46]. Hence, a good match cannot be guaranteed for trivial approaches which attempt to find the closest road segment to a sample point solely using its raw positional data. A more sophisticated matching approach is unavoidable.

There are basically two map-matching categories available: a global and an incremental algorithm. In principal, the global one uses the overall trip taking each GPS point from origin to destination into account. The incremental method, by contrast, merely uses a limited range of previous sample points, which are incrementally fed to the algorithm.

Typically, global algorithms are more comprehensive and more difficult to implement but yield better results. They take all interconnections of road segments - the road *network* - into account and do not just rely on sample point's position as the incremental ones do. To limit computation costs, which are especially expensive for the following introduced transmission probability, only a

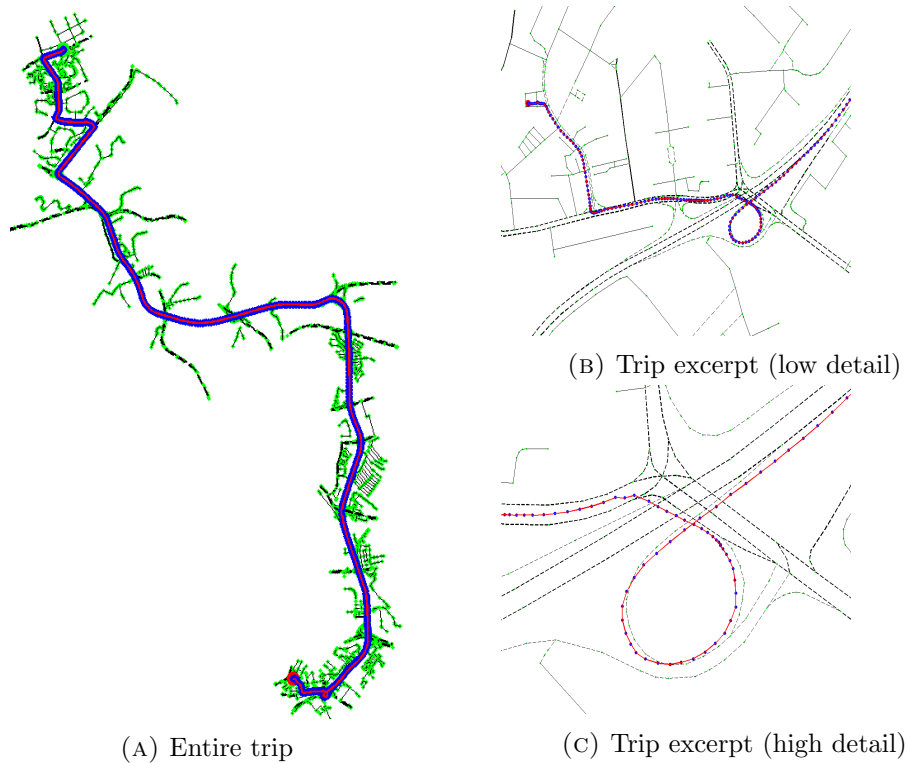


FIGURE 4.19: Example trip compared to the underlying road network for various detail levels

subset of each 60th sample point is used. In this way, the sampling period is artificially increased to 60 seconds.

Map-matching implementations either tag the road segment id to the sample point or adjust the actual sample point's GPS coordinates to the matched road segment and, as a consequence, change the trajectory directly. In this implementation, the former method is used since information about traversed road segments is already sufficient. The speed profile information is vital, whereas detailed position information is less important with respect to energy consideration.

An example trip projected on the road network is illustrated in Figure 4.19. The left-hand chart shows the trip's trajectory in blue, connecting sample points in red, on the background of a subset of neighbouring black ways and green nodes as imported from OSM. The more detailed excerpts in the right-hand chart reveal a considerably good match between trajectory and roads but also highlights the narrow and fine meshed network, requiring a global map-matching approach.

4.4.1 Map-Matching ST Algorithm

The choice fell on the ST (Spatial-Temporal) map-matching algorithm as proposed in [14], which addresses a wide range of feasible sampling rates, particularly supporting low sampling periods of three to five minutes. With respect to this thesis' goal of estimating energy demand of trips with a high variety of sampling rates (periods of one second up to the sole information about origin and destination), it suits requirements very well. The ST algorithm pertains to the global map-matching approaches, contemplating the entire trip's trajectory. It consists of a spatial and a temporal part, which jointly reveal its strength and are presented as follows.

Spatial (S) Component

The spatial component is the major influencing and most powerful component, consisting of an *Observation* and a *Transmission* part. In the foundational observation part, the sample point's position is analysed and distances to neighbouring - and thus possibly traversed road segments - computed. In order to compute these distances, a perpendicular line from road segment to the sample point in question is projected, and the Euclidean distance in metre is retrieved, following the method already utilized in the trajectory filtering step in Section 4.2.2.

In Figure 4.20, a sample point p_i has three road segments e_i^k , $1 \leq k \leq 3$ in its vicinity, with each projection resulting in a corresponding candidate c_i . The distance is converted into a normalized weight factor for all considered nearby road segment, which is higher for closer segments and lower for distant ones.

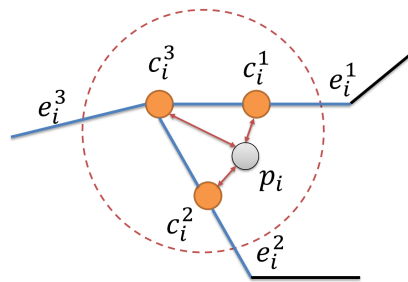


FIGURE 4.20: Candidate points c_i^1 , c_i^2 and c_i^3 on road segments e_i^1 , e_i^2 and e_i^3 for a sample point p_i , adopted from [14, p. 4]

Equation 4.3 expresses the observation probability N mathematically, where x_i^s is the distance between p_i and c_i^s , μ the mean deviation (in the GPS context equal to zero), and σ the standard deviation (in the GPS context equal to ± 20 m) [14, p. 4].

$$N(c_i^s) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{(x_i^s - \mu)^2}{2\sigma^2}} \quad (4.3)$$

The transmission part of the spatial component computes transmission probability, expressing the likelihood that a car moved from point A to B. It computes the Euclidean distance between two recorded sample points and additionally the distance between corresponding projected points on the candidate's road segment. In this cases, however, the latter distance is not imperatively equal to the Euclidean distance, but it rather reflects the actual distance if a car would have driven on that specific road. It hence incorporates the projection of sample points on the road network and receives the distance through a *graph shortest path* computation (details are to follow).

After that, the algorithm forms the ratio between both distances, technically comparing the smallest possible 'bee-line' with the actual driven distance. The higher the ratio, the higher the likelihood that the road segment was traversed, assuming that cars usually take the shortest path between two adjacent sample points. An example where transmission analysis is required illustrates Figure 4.21, in which the closest candidate projection c_i^1 of point p_i would have been classified wrongly.

The addressed shortest path computation aims at finding the shortest possible path through a network of vertices and paths in terms of weights for each connecting link. In case of map-matching and a real road network, the vertices translate to nodes, and the paths translate to roads. The weights rely on the established connectivity matrices of the imported OSM network, denoting the

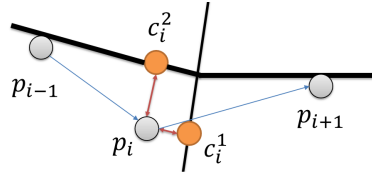


FIGURE 4.21: Example of points and candidates which require transmission probability as p_i has two candidates c_i^1 and c_i^2 , and c_i^2 has higher observation probability but would be the wrong match according to transmission likelihood, adopted from [14, p. 5]

distance between two nodes. Matlab's built-in `graphshortestpath` function utilizing Dijkstra's algorithm [36] can handle this task.

Mathematically, the transmission probability is expressed in Equation 4.4, where $d_{i-1 \rightarrow i}$ is the Euclidean distance between p_i and p_{i-1} and $w_{(i-1,t) \rightarrow (i,s)}$ equals the length of the shortest path from c_{i-1}^t to c_i^s .

$$V(c_{i-1}^t \rightarrow c_i^s) = \frac{d_{i-1 \rightarrow i}}{w_{(i-1,t) \rightarrow (i,s)}} \quad (4.4)$$

Both spatial components, the observation and transmission probability, result in the final spatial probability factor F_s as expressed in Equation 4.5.

$$F_s(c_{i-1}^t \rightarrow c_i^s) = N(c_i^s) \cdot V(c_{i-1}^t \rightarrow c_i^s) \quad (4.5)$$

Temporal (T) Component

The temporal component addresses road specific speed constraints. It computes a temporal correlation value, expressing the matching between average speed of the actual trip extract of two adjacent sample points and the historical average road segment speed. The latter, historical value has either been gathered from actual recorded trajectories or from LTA speed band sensors.

The correlation is fundamentally computed by the Cosine distance, which expresses the similarity between both speeds. An exemplary situation where temporal analysis is needed illustrates Figure 4.22. Here, the spatial analysis, including observation and transmission probability, would reveal a draw; but the fundamentally different temporal constraints for a Service Road and a Highway would indicate the more likely road segment (which is not derivable from this example as no speed values are given).

Its mathematical expression is found in Equation 4.6, where $v_{(i-1,t) \rightarrow (i,s)}$ is the speed from candidate point c_{i-1}^t to c_i^s , and $\bar{v}_{(i-1,t) \rightarrow (i,s)}$ is its reference average value.

$$F_t(c_{i-1}^t \rightarrow c_i^s) = \frac{v_{(i-1,t) \rightarrow (i,s)} \cdot \bar{v}_{(i-1,t) \rightarrow (i,s)}}{\sqrt{v_{(i-1,t) \rightarrow (i,s)}^2 \cdot \bar{v}_{(i-1,t) \rightarrow (i,s)}^2}} \quad (4.6)$$

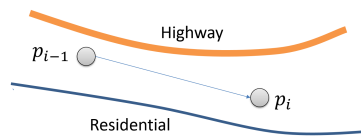


FIGURE 4.22: Example for required temporal component for a GPS trajectory described by points P_i and P_i with ambiguous matchable road segments, adopted from [14, p. 5]

4.4.2 Application of Map-Matching on recorded GPS Trajectories

For each recorded trip, map-matching is conducted by performing both a spatial and a temporal analysis, and results are combined according to Equation 4.7. Eventually, F reveals the likelihood that a candidate c_i^s at time s follows a preceding candidate c_{i-1}^t at time t .

$$F(c_{i-1}^t \rightarrow c_i^s) = F_s(c_{i-1}^t \rightarrow c_i^s) \cdot F_t(c_{i-1}^t \rightarrow c_i^s) \quad (4.7)$$

As mentioned in this section's introduction, in order to limit computation time only a subset of sample points is used - typically each 60th. To still exploit the strength of a high sampling rate as available from the GPS logger, intermediate sample points are not simply skipped but are rather estimated by using an interpolation technique in form of a fitting line through a number of vicinity points and thus still indirectly take all sample points into consideration. The map-matching process takes a comparably long time why computation is largely outsourced using Matlab's parallel computing pools on a remote computer.

Construction of Candidate Graph

A candidate graph displays the weights of connected candidates for adjacent sample points. Candidates are possible points on vicinal road segments projected from an actual recorded sample point. The weights are the resulting values of F including both algorithm components Spatial S and Temporal T.

An example for a variable number of candidates per sample point is presented in Figure 4.23. Given the fact that the sample point p_1 has the candidate c_1^1 , and the subsequent sample point p_2 has the candidate c_2^1 , then the weight is $c_1^1 \rightarrow c_2^1$ expressing the probability that the car has taken this path.

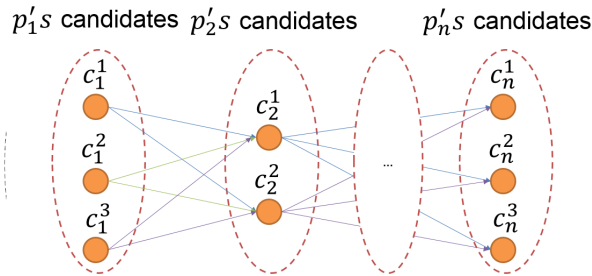


FIGURE 4.23: Candidate graph construction for points P_i with candidates c_i^j and weights $c_i^j \rightarrow c_{i+1}^j$ for $1 \leq i \leq n$ with n denoting the number of points, and $1 \leq j \leq m$ with m denoting the number of candidates, adopted from [14, p. 5]

The number of candidates is either confined by a circular area around the sample points coordinates or is fixed to a constant number. In the contemplated case, the number is fixed to three, allowing to speed up the candidate gathering and candidate graph construction. The resulting sizes of the spatial components are: for observation probability 3×1 , since points position is not related to each other, and for transmission probability 3×3 , because every candidate has a distinct weight to reach each other candidate. This results in a 3×3 matrix. Also for temporal probability a 3×3 matrix is received since each connection traverses a set of road segments with distinct temporal constraints.

Retrieval of Best Fitting Sequence

In this step, the task is to compute the highest weight value as the best fitting sequence of connected candidates through the candidate graph. To do so, the candidate graph is flipped, and all weights from end to start are added together. The highest value points to the best fitting sequence, while the algorithm keeps track of each winning candidate's predecessor in each step to later backtrack all best suiting candidate points. The implemented algorithm for the whole map-matching process, from sample points to the best fitting sequence of projected candidate points on road segments, is outlined in Figure 4.24. It consists of the major modules *Sample Point Retrieval*, *Candidate Retrieval*, *Observation Probability Computation*, *Transmission Probability Computation*, *Temporal Analysis*, and finally *Best Fitting Sequence Analysis* with N denoting the share of taken sample points and C denoting the number of candidates.

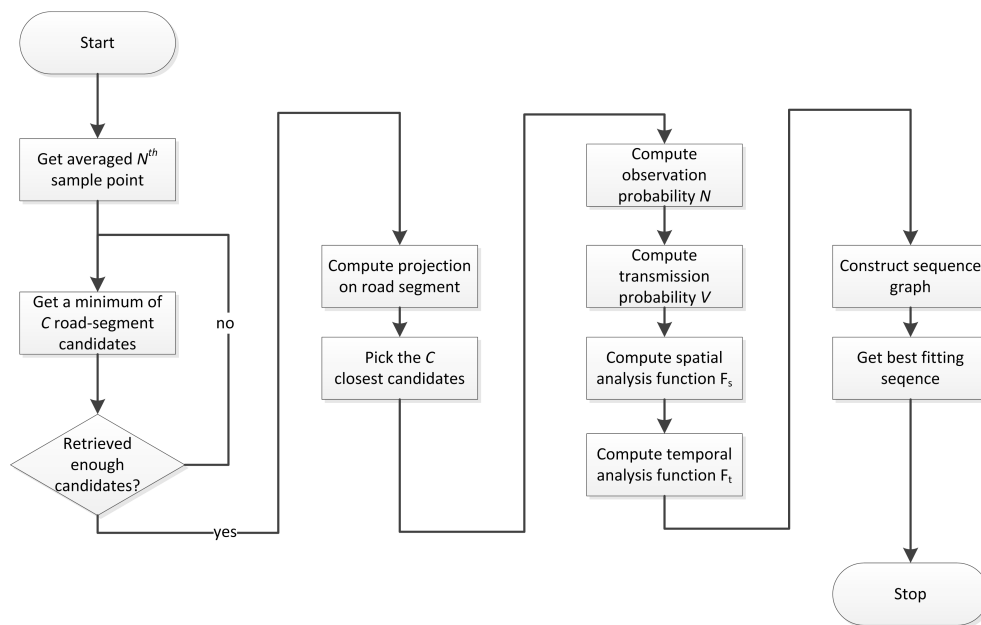


FIGURE 4.24: Map matching algorithm for candidate graph construction from sample points to road segments ids flowchart

Result Matching and Post-Processing

The so far described steps result in a sequence of candidate points or road segments ids, respectively, for the considered trip. Those ids can easily be matched to the corresponding sample points which have been used for map-matching. However, taking into account that only a subset of sample points was used, it is required to fill arisen gaps. This time, a simple interpolation would not work out because the algorithm operates in the context of a discrete road network. The gaps are consequently filled applying the shortest path repeatedly for the segment between each already map-matched anchor point and its successor.

The shortest path analysis reveals intermediate traversed road segments, which are matched according to the sample points Euclidean distance to road segment transitions. Again, it is assumed that a direct way is preferred by the taxi driver expressed by utilizing the Euclidean distance connectivity matrix. The result matching process is feasible since just a relatively short segment of 60 seconds is considered. Evaluation results in Section 5.2 are evident for herein made

choices. The algorithm is presented in the flowchart in Figure 4.25, taking a partial map-matched trip as in input and producing a fully map-matched trip as an output.

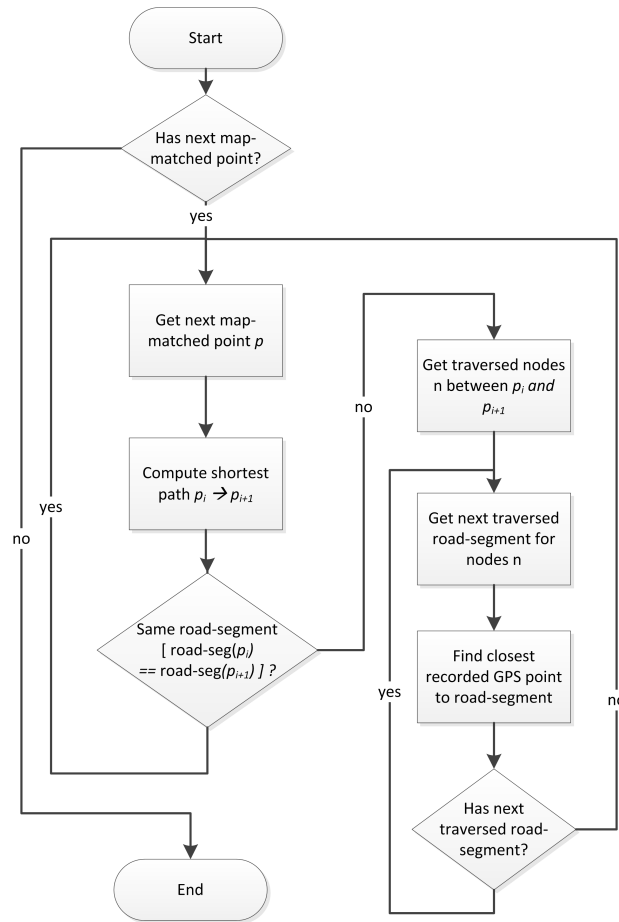


FIGURE 4.25: Map matching post-processing algorithm flowchart

4.5 Micro Trips and Driving Features

This section discusses the extraction of micro trips, which are small partial trips, as well as driving features, which are a set of driving pattern characteristics, both in preparation of the instant driving feature approach for estimating energy consumption. Outlining this approach, extracted micro trip features are firstly parsed into distinct clusters, the energy demand of each cluster is secondly computed, and unclassified micro trips are finally categorized according to these cluster in an application stage. Details for this process are to follow in Section 4.5.2.

4.5.1 Micro Trip Extraction

As a micro trip, a partial trip comprising a set of real-world sample points is defined which is targeted to have a duration of about 210 seconds. The purpose is to have the capability to analyse a micro trip independently from its parent by using mean values to describe basic characteristics. Mean values can only be sufficiently applied if the micro trip is considerably short and its characteristics are as distinct as possible.

A set of features should reflect reoccurring patterns and hence a characteristic energy consumption. The idea is that each micro trip represents a certain driving phase. An application and detailed method's description can be found in [12]. Furthermore, it is possible to assemble multiple micro trips into a driving cycle or into an artificial constructed 'real-world' trip, but this is disregarded in this work.

To obtain reasonable micro trips, two boundary conditions are defined which enable a smooth transition among two micro trips at its edges where they are intentioned to be split. Primarily, speed values are searched obeying a threshold of 1 km/h, which can be considered standstill. This way, a micro trip would automatically result in a closed stint between two stops. However, for long periods on motorways, standstill time will barely occur. This leads to a second boundary condition of 'zero' acceleration. For this, tolerance threshold are introduced, too: 0.15 m/s^2 for acceleration and -0.15 m/s^2 for deceleration to allow for slight deviations. Also the length of targeted 210 seconds is expanded by a threshold band of ± 30 seconds as suggested in [47] with purpose to extract more possible micro trips from a real-world trip.

The term micro trip occurs in different contexts throughout this work. Its is distinguished between a road segment, road type, and floating micro trip. The first two terms are used in the context of map-matching where a micro trip is considered a small partial trip on one road segment or road type, respectively. The term floating micro trip is synonymously used for the here described extraction of micro trips from real-world trips.

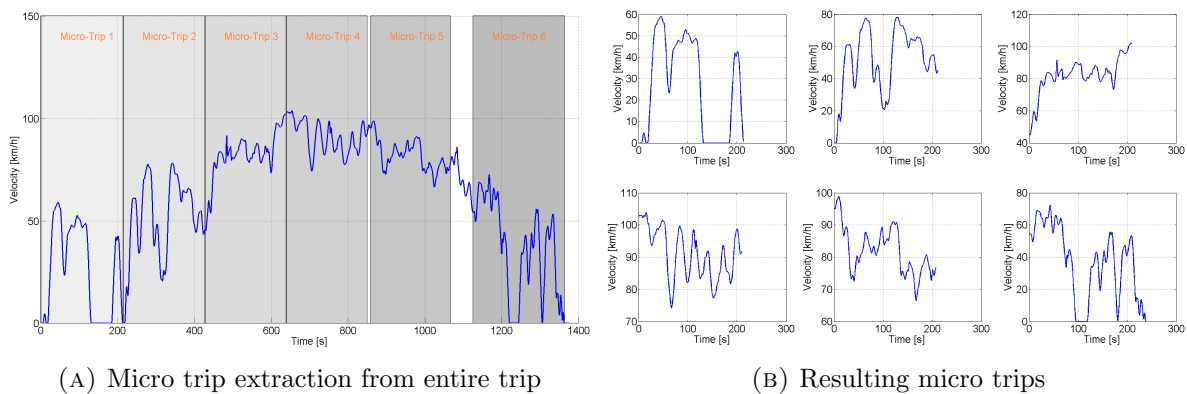


FIGURE 4.26: Micro trip extraction from a real-world trip

4.5.2 Driving Feature Extraction and Clustering

This section explains the proceeding of feature exaction and a thereon based clustering of yielded multidimensional feature vectors. Features are characteristic properties of micro trips like average speed, distance, duration, or major road type. A complete list of appointed features lists Table 4.1, adopted from recommendations given in [12].

The extraction of those is straightforward as it is worked on a limited number of roughly 200 sample points per micro trip. Initially, interim values are computed on a point-to-point basis, such as all accelerations between two subsequent sample points, and afterwards averaged complying with mentioned transition thresholds. For example, thresholds restricting accelerations with a minimum value prohibit from including bias by negligibly low values. A fully examined set of features is called *feature vector* and has the same dimension as the number of incorporated features.

On basis of extracted feature vectors - each per micro trip - it is attempted to find underlying patterns and characteristic driving behaviours derivable from a large number of vectors. This step is called *clustering* as it attempts to assign each vector to a certain category for which most of the entailed features fit best. There are a number of cluster algorithms and tools available. An elaboration of various approaches would go beyond this work's scope, and so only a short introduction in the eventually utilized `kmeans` algorithm is given, as it is conveniently shipped with Matlab's Statistics Toolbox.

Kmeans' principal functionality is determined by minimizing the distance between cluster centroids and akin feature vectors. A centroid is the centre of all entailed vectors of a cluster and, so to speak, indicates the cluster's 'position'. Methods for calculating distances among vectors and centroids are herein variably defined. Common measurement metrics are *squeclidean* and *cityblock*, whose former is the "squared Euclidean distance [...] (where) each centroid is the mean of the points in that cluster" and latter the "sum of absolute differences [...] (where) each centroid is the component-wise median of the points in that cluster" [48]. In this work's performed examinations, the Euclidean distance method is predominantly chosen. The smaller the distance, the more likely a feature vector belongs to the centroid. Ultimately, one centroid is gained for each cluster, which is as dimensional as the feature vector.

Clustering requires the multidimensional feature arrays to be normalized to avoid different scales and ranges among feature's arrays. Without normalizing, the weight and therefore the influence of traditionally higher values like speed or time compared to acceleration or the number of stops would be equally higher. A normalization limits values among each feature to an upper and lower bound and compensates unequal distributions.

A peculiarity of `kmeans` is accounted for by the pre-definition of the amount of clusters, feature vectors are portioned to. This is not an easy task, since the level of distinctiveness among the clusters is not trivially estimable. The number of clusters was eventually determined empirically by varying its quantity for several runs and incorporating the best fit.

Another option is to extract a certain feature from the vector (like road type or time zone), disregard it from the actual clustering process, and place it one level above - thus establishing predefined clusters. For instance, three predefined road type clusters could be used on whose subordinated feature vectors a clustering is executed. However, this would require map-matching to enable road type classification which was aimed to be avoided.

4.5.3 Driving Feature Classification

The previous erected clusters are now utilized for matching feature vectors to clusters. This step is called *classification*. Classification attempts to find the best matching cluster for a given feature vector of an unclassified micro trip aiming at recognizing reoccurring patterns by employing the same distance computation method as used for centroid extraction. In the context of energy estimation, the classification harvests previously computed energy consumptions for each cluster and takes this value for granted for the classified micro trip.

An insight into cluster arrangements and cluster centroids is illustrated in Figure 4.27. Here, four different feature combinations are displayed in a two-dimensional space. Although the actual clustering and classification steps take all 16 features into consideration, it is not feasible to visualize those graphically. Yet, the plots reveal interesting coherences and are evidence for distinct delimitations among established clusters.

TABLE 4.1: Fields of a feature vector for the driving feature approach

Index	Feature	Unit	Typical Value
1	Distance	km	1.5
2	Mean speed total	km/h	30
3	Mean speed excluding stops	km/h	35
4	Maximum speed	km/h	60
5	Minimum speed	km/h	0
6	Share idling	%	20
7	Share cruising	%	40
8	Share accelerating	%	25
9	Share decelerating	%	15
10	Mean acceleration	m/s ²	0.4
11	Mean deceleration	m/s ²	0.6
12	Stop rate	1/km	2
13	Mean stop time	1/km	15
14	Highway cluster type	-	1
15	Time period	-	1
16	Energy consumption	kWh/100km	16

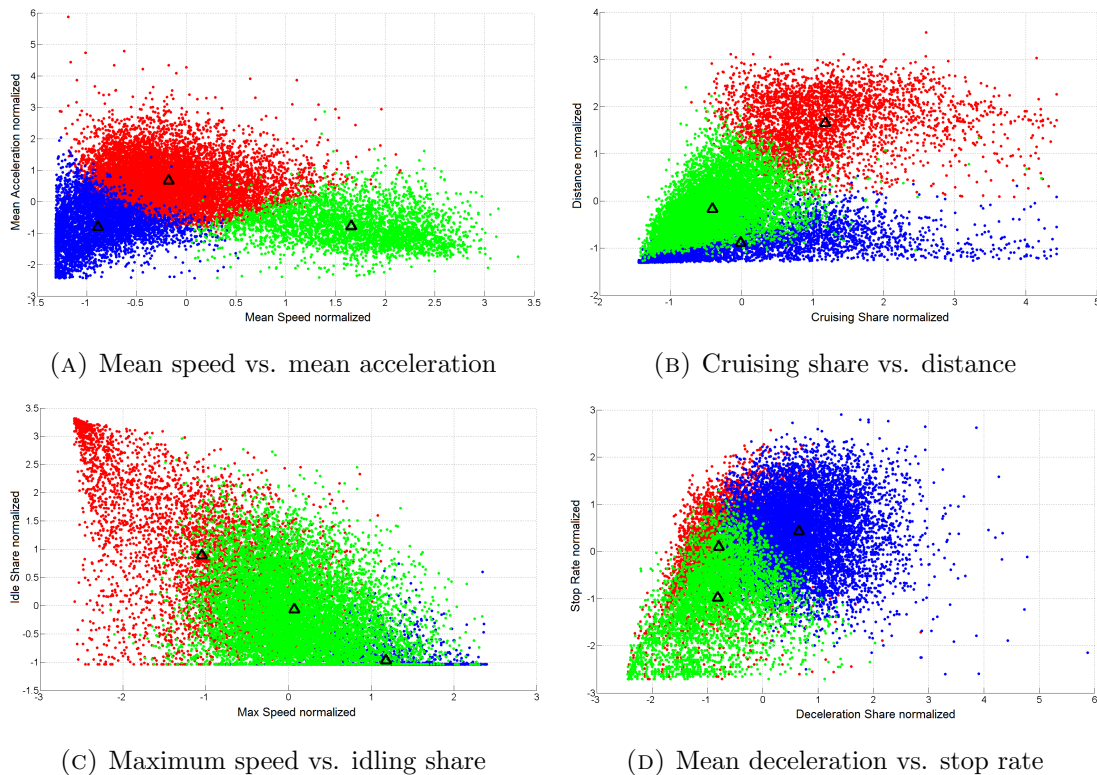


FIGURE 4.27: Cluster centroids for various two-dimensional excerpts of the 16-dimensional feature array

For instance, the graph in Figure 4.27a illustrates micro trips' mean acceleration on the y-axis subject to its mean speed on the x-axis. It is apparent that three clusters with marked boundaries are established, following the general distribution of an Acceleration-Velocity AV matrix [49].

4.6 Statistical Analysis

A statistical evaluation is carried out for several reasons. One is to gain insights into the dataset influencing the latter energy demand analysis. For instance, the analysis of the average booking statuses throughout the day (see Figure 5.10 in the following result unit in Chapter 5) influences the trip and shift extraction. Another reason addresses the validation of the driving profile analysis and its implementations in Matlab. The trip distance distribution in comparison to LTA provided data, for example, gives approximate feedback if the implemented algorithm fulfils the overarching demands (see Figure 5.14). A third reason is the request from SMRT to provide detailed feedback about driving patterns of appointed taxis during monthly update meetings and is part of the mutual agreement.

In a first attempt of compiling the statistics, it was distinguished between luxury and normal taxis for certain statistics like daily mileage and trip distance, expecting this to have a bigger impact than the distinguishing between one- and two-hirer taxis. However, the focus changed as it eventually turned out that the distinguishing of hirer schemes reveals a higher distinction, allowing to gain more meaningful results.

4.6.1 Logging Data based Statistics

Logging data based statistics focus on recorded GPS trajectories, its dynamic behaviour, as well as trip and shift extractions. This section immerses into evaluation methods highlighting the construction of heat maps and the analysis of mileage for trips and shifts.

Heat Map Construction

A heat map is an effective method for displaying a high amount of comprehensive information in a compressed, graphical, and insightful manner which is easily perceivable. In this work, heat maps are mainly used for visualizing position occurrence frequency of recorded taxis on Singapore's road network as well as for displaying the origin and destination frequencies and its differences subject to Planning Areas.

The basic method behind constructing a heat map is to count the number of occurrences of sample points obeying a certain condition in a certain area. The position heat map, for instance, counts the presences of GPS positions, denoted by latitude and longitude, in a rectangular area. A necessary step is therefore to priorly erect a rectangular grid for Singapore, preferably of size 200×300 as already used for the meshed grid for the OSM road network (see Section 3.2.3).

In the algorithm, each sample point's position is unambiguously matched to exactly one cell. Heat maps for origin and destination occurrences subject to Planning Areas are constructed in a similar way. Instead of using each sample point, only the first and last sample points of a trip are taken. The matching is furthermore not based on the assignment to a rectangular cell of the meshed grid, but the points are fitted to a polygon depicting the Planning Area's boundaries (see Appendix K).

In a last step, the occurrence values are normalized to unity and matched to a colour map. The strongest or brightest colour is linked to the highest value, which is unity after normalizing,

whereas the lowest value (zero) matches with the weakest or lightest colour. Intermediate values are matched to the colour map's intermediate range by applying a logarithmic scale. A logarithmic scale is incorporated for presentation reasons to avail for displaying even single occurrences and to smooth the heat map to a certain extent.

Based on those heat maps, small video clips are created displaying a set of heat maps subject to time, particularly revealing the time dependent variations and transitions which would have been hidden when only considering the average values as in an ordinary heat map. For its construction, the described steps are repeated but applied on a time dependent subset. Hence, previously interested data is extracted within a moving window frame of 60 minutes and with a moving speed of ten seconds per frame, which equals the resulting video's frame rate. The ratio between window size and frame rate is deliberately chosen to this value to allow for smooth transitions. A too small window size would only show a very small data extract per frame and thus would hide the meaningful changes subject to time.

Mileage Analysis

Most of the compiled diagrams include a histogram or bar chart to display data in an easy perceivable manner. Examples are given in the result Section 5.3.3. Histograms hereby make use of the built-in Matlab `histc` function, which counts the occurrences of items in an array. Resulting distributions are displayed using the `bar` function, which is often preferred to the ordinary `plot` function preferentially used for displaying trend data.

The mileage is computed using the speed based approach. Also the trip distance is included, which, however, is highly dependent on the prior trip extraction. The shift lengths behave accordingly, but the extraction is even more challenging since there is no real additional input indicating suitable shift changes. Moreover, it can not be taken for granted that there are in fact distinct shifts for two-hirer scheme taxis.

4.6.2 Booking Data based Statistics

As mentioned earlier, the taxi's booking status is retrieved through the in-car MDT, which periodically sends status information to an SMRT server. These status chunks are, however, only send if the MDT is online and inevitably leads to gaps in booking data recordings. Additionally, the definition of 'online' is in a sense blurred because in some cases booking information of status Log Off was encountered over a period of multiple hours. Therefore, the MDT's online mode and hence the update of status information does not reliably link to the taxi being ready for driving or not.

Furthermore, the sampling period of three minutes is not unconditionally valid throughout the set, which leads to fluctuations in recorded timestamps. Latter effect is most likely accounted for by a loss mobile data signal or GPS satellite connection, former used to send the status information and latter for retrieving the position.

Yet, to consistently treat booking statuses over time, an interpolation is necessary. Commonly occurring gaps or unsuitable sampling rates are interpolated using the Nearest Neighbour method, which was already applied for interpolating status information in the logging data set (see Section 4.2.3). To fill prolonged gaps of more than 30 minutes, those are interpolated with the status Log Off, assuming that such a duration of missing status signals translates to an according offline mode of MDT and hence taxi's inactivity.

Prerequisite steps for constructing booking based statistics include the merging of several monthly booking sets with possible overlapping times. To reach a daily status distribution as displayed in Section 5.3.2, a day is divided into portions of 15 minutes, each denoting the excerpt's booking status distribution.

Chapter 5

Driving Profile Analysis - Results and Evaluation

As outlined in the previous chapter, the driving profile analysis plays a major role in preparing the data set into a utilizable form for subsequent processing steps. It is particularly vital for statistical evaluation and the targeted computation of energy consumptions.

This chapter evaluates conducted pre-processing and map-matching steps, and presents results of statistical evaluation by eventually comparing those to LTA data.

5.1 Pre-Processing Evaluation

The pre-processing comprises the combination of logging and booking data, filtering, and interpolation of spatial and dynamic data. It is the primary step in transforming raw and erroneous GPS trajectories and booking data into filtered and processed data for subsequent analysis steps.

5.1.1 Dataset Combination Evaluation

The dataset combination merges the logging set, which is received from installed GPS loggers, with the booking set, which is received from taxi's MDTs. They have different sampling periods of one seconds for the logging and three minutes for the booking set, which requires a comprehensive combination process. As a short reminder, the algorithm basically tries to find same timestamps and tags the received statuses from booking sample points to the corresponding logging sample points.

Since gaps and inaccuracies in sampling rates are encountered, the algorithm is designed robustly to allow for these deviations. Yet, to check for compliance, the deviants in time and position are computed and assessed whether its deviation is below a certain maximum threshold.

Figure 5.1 shows a good match (top plot) and bad match (bottom plot) for exactly this procedure. In each, the chart's upper part shows the time deviation within a window of ± 30 s and the actual match as a red dot, and the chart's lower part reveals the position deviation and the actual matching, accordingly as a red dot. Here, the threshold denoting the maximum distance between correspondence sample point and deciding whether the match is valid or not is set to 500 m.

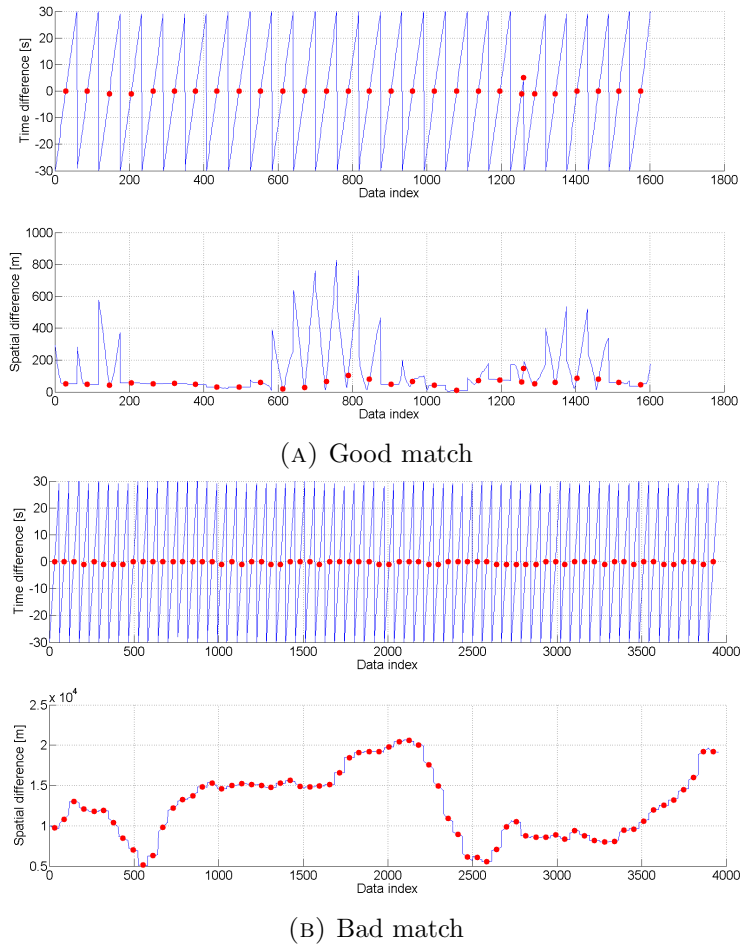


FIGURE 5.1: Evaluation of the combination of logging and booking data

5.1.2 Filtering and Interpolation Evaluation

Filtering, in the sense of GPS trajectories, is the process of analysing a set of contextual connected sample points according to restrictions, limitations, and thresholds with the goal of adjusting those accordingly.

Interpolation, on the other hand, substitutes missing or disarranged values with estimations of best fitting intermediate point to complete a trajectory. In this work, the scope of interpolation is extended as the term is also used for the altitude assignment process, where virtually all values are replaced by a different data source.

Filtering Evaluation

In Figure 5.2, a speed profile for a recorded example trip is presented. The green graph depicts the original values as captured from the GPS logger, and the blue graph illustrates the speed profile after filtering. Two major insights are perceivable: One is the existence of apparent outliers at approximately times 50 s and 400 s; the former capturing a too high speed value in the vicinity of slow speeds, whereas the latter records a single unreasonable zero speed value during a relatively constant driving period. During the filtering process, though, both outliers are correctly detected and erased as desired.

The second major insight is the smoothing effect on the whole speed profile. The unfiltered profile shapes with high fluctuations and entails almost no constant driving periods. It is very unlikely that the actual vehicle behaved as the logger has recorded, omitting any periods of constant speed. It is more reasonable that the recorded values are to some extent erroneous. The filtered speed profile, coloured in blue, is smoother than its green counterpart as it underwent a Gaussian filter of a small window size of five sample points, which is applied in addition to the outlier detection. The window size is restricted to this rather small value because the impression of ‘correctness’ solely bases on empirical inspections and no scientifically justification for a distinct value can be given. To do so, a comparison with CAN Bus values as outlined in Figure 3.3 would have been necessary for all tracked vehicle types but was not available at this time.

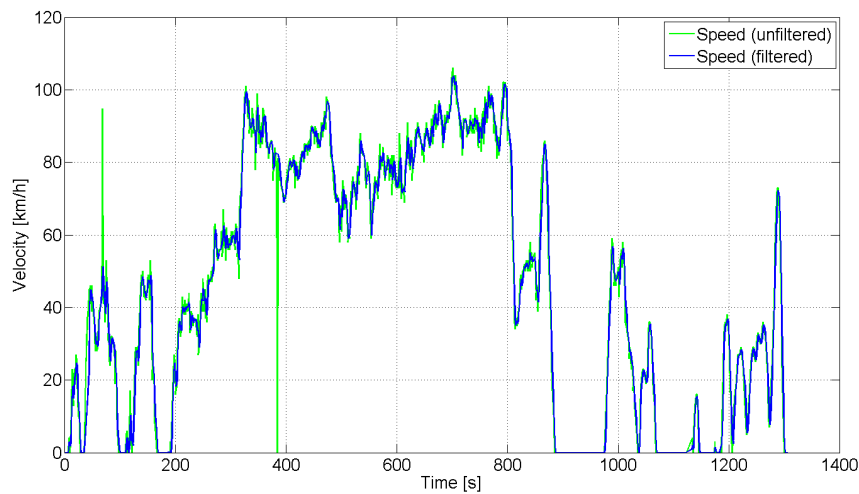


FIGURE 5.2: Comparison of raw and filtered speed profiles

By immersing into the in-detail filtering procedure as results presented in Figure 5.3, the impact of filtering for speed and acceleration values can be observed. Both right-hand charts show the speed subject to time trend, divided into an unfiltered version on the top and a filtered version on the bottom, highlighting the filter threshold of 200 km/h at which sample points are erased from the trip. Both left-hand charts show the according acceleration trend, equally divided into an unfiltered chart in the top and a filtered version in the bottom graph, applying threshold of $+4 \text{ m/s}^2$ and -6 m/s^2 for acceleration and deceleration, respectively.

In terms of filtering, methods are distinguished between spatial and dynamic input types. As already frequently noted, speed values are preferably obtained directly from the GPS logger rather than computing the speed manually by means of latitude and longitude coordinates. From Figure 5.4 is evidence that the position based speed would fluctuate to a high extent, inevitably caused by GPS immanent deviations. The spatial uncertainty, which usually does not impact the energy consideration to a high extend, would in this case be incorporated into the speed values and hence accordingly exacerbatingly fluctuate.

As another prove that computed speed values project the driving behaviour improperly, the average acceleration and deceleration, as well as the former’s standard deviation are computed as depicted in Table 5.1 and compared to literature values for various ARTEMIS cycles in Table 5.2, taken from [50]. In the regarded case, values are presented for a real-world trip with a high share of OSM road type Motorway, comparable to a ARTEMIS Motorway cycle. For the filtered,

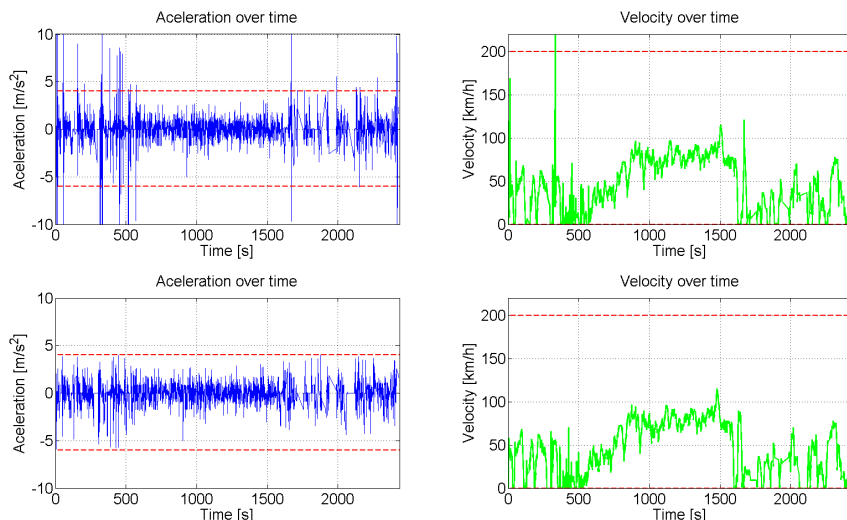


FIGURE 5.3: Results for acceleration and speed filtering using $+4 \text{ m/s}^2$ and -6 m/s^2 thresholds for maximum acceleration and deceleration, respectively, and 200 km/h as the maximum speed threshold



FIGURE 5.4: Comparison of speed values from GPS logger (blue) and speed computed on basis of sample point's GPS coordinates (green)

logger based speed computation method both sets of characteristics align very well, whereas all other methods underperform and do not match satisfyingly.

Interpolation and Height Assignment Evaluation

In the interpolation step, gaps in trips are filled which either originate from loss of connection to GPS satellites or from eliminations in the preceding filtering process. As described, filtering simply erases unreasonable or erroneous sample points and expects the substitution of missing values in the interpolation step. In both cases, gaps' boundary values are used as anchor points to estimate suitable intermediate sample points. In Section 4.2.3, various interpolation strategies for different variable types are introduced, mainly distinguished in continuous and discrete methods. Here, the focus lays on evaluating spatial and speed interpolation, as well as assessing the altitude assignment on basis of an adopted Google elevation map.

TABLE 5.1: Average acceleration, deceleration, and standard deviation for an example trip with a predominant Motorway share

Source	Average Acceleration	Average Deceleration	Standard Deviation of Acceleration
Filtered logger speed	0.54 m/s ²	-0.61 m/s ²	0.42 m/s ²
Unfiltered logger speed	1.18 m/s ²	-1.16 m/s ²	1.10 m/s ²
Filtered GPS position speed	0.73 m/s ²	-0.77 m/s ²	0.79 m/s ²
Unfiltered GPS position speed	0.81 m/s ²	-0.84 m/s ²	0.85 m/s ²

TABLE 5.2: Average acceleration, deceleration, and standard deviation for three ARTEMIS cycles, taken from [50]

ARTEMIS Cycle	Average Acceleration	Average Deceleration	Standard Deviation of Acceleration
ARTEMIS Motorway cycle	0.52 m/s ²	-0.68 m/s ²	0.49 m/s ²
ARTEMIS Road cycle	0.58 m/s ²	-0.65 m/s ²	0.56 m/s ²
ARTEMIS Urban cycle	0.75 m/s ²	-0.75 m/s ²	0.68 m/s ²

The speed profile illustrated in Figure 5.5 is a good example particularly for the interpolation performance in the vicinity of tunnels, where a more comprehensive approach is employed. The left-hand chart shows the state before filtering and interpolation with an unreasonable steep slope during tunnel transit and sudden rising speed at the end of the tunnel. The right-hand, interpolated version mitigates this apparently flawed values with an assumption of constant speed. The bottommost chart displays the actual GPS trajectory of a tunnel transit after filtering and interpolation and highlights important parts.

Singapore's terrain is a lowland with a slightly hilly central plateau. Its highest elevation is Bukit Timah with 163.63 m above sea level, and its lowest point is 0 m as the island has direct access to the sea [51]. Although considered relatively flat with an admittedly small, yet existing impact on vehicle's energy demand, a height assignment is performed following procedures described in Section 4.2.3.

The necessity of this height assignment step can be obtained from Table 5.3 where characteristic domains for altitude values and gradients for an unassigned and an assigned version of a subset of recorded trips are listed. Regarding the altitude, the unassigned version exceeds the maximum possible elevation of Singapore for more than 70 m - indicating the erroneous raw values. Moreover, both the mean and 95 % percentile appear to be way too high compared to the assigned version. In addition, a 5 % percentile of exactly 0.00 m points to a high amount of sample points with exactly this zero value and is likely to be caused by the logger characteristic behaviour to record a zero value for uncaught altitudes or in case of a bad GPS coverage.

TABLE 5.3: Altitude and gradient evaluation before and after assigning with the Google elevation map

Feature	Method	Mean	Min	Max	Percentile 5 %	Percentile 95 %
Altitude	Unassigned	29.86 m	-5.00 m	241.00 m	0.00 m	88.75 m
	Assigned	17.82 m	-3.27 m	70.22 m	7.86 m	34.99 m
Gradient	Unassigned	0.23 %	-132.50 %	149.00 %	-5.12 %	5.19 %
	Assigned	0.00 %	-14.21 %	14.71 %	-1.52 %	1.55 %

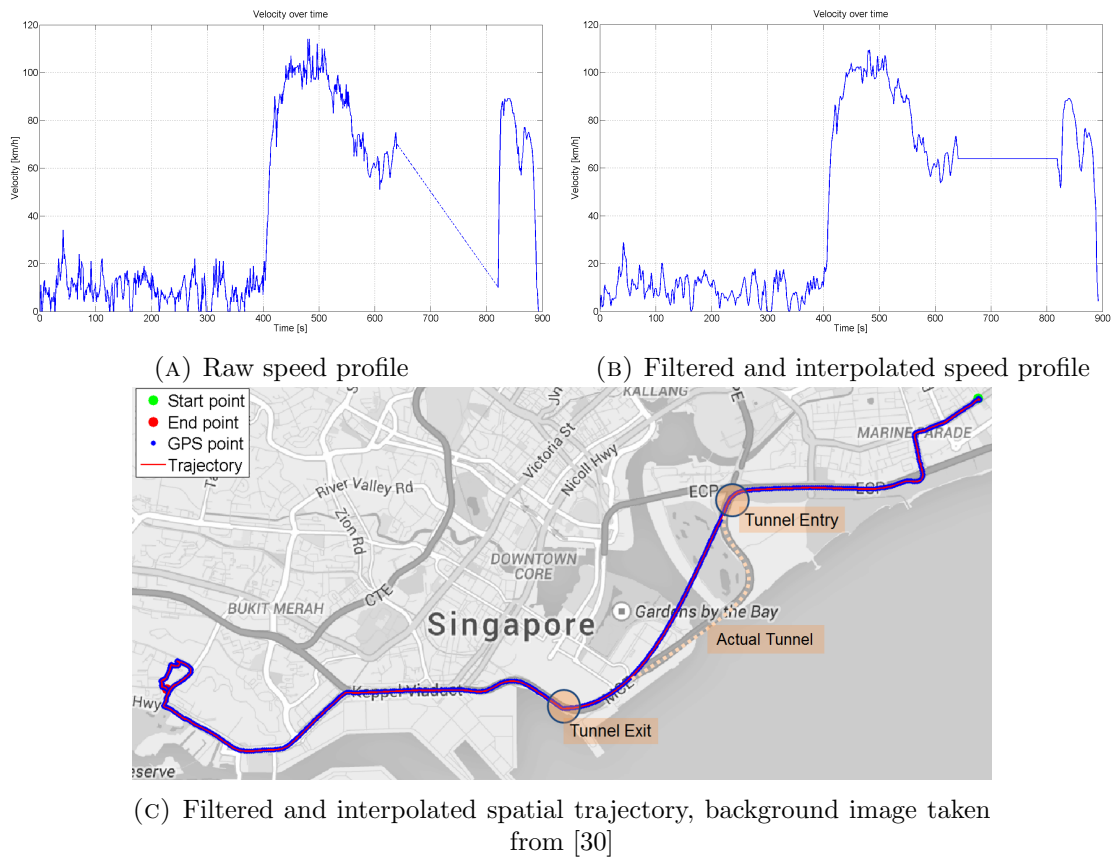


FIGURE 5.5: Interpolation of a tunnel transit for speed profile and spatial trajectory

Distance Computation

During the implementation of the programs, the need for computing trip's distance in a detailed manner was often encountered. Three different distance computation methods are implemented: position based, speed based, and road segment based; and results are compared against a Google Maps API's suggested trip lengths.

It turned out that there is no one-fits-its-all solution, but each approach has its advantages and disadvantages for different scenarios. Eventually, the decision favoured the speed based method as it produces the most exact values in most cases. The method calculates the distance on basis of pre-processed speed values from the GPS logger and the time between two adjacent sample points (which is fixed to one second after interpolation). This approach is already implicitly incorporated into the speed profiles for the energy demand estimation.

Table 5.4 lists comparative values for three scenarios: a good quality trip, a trip in the vicinity of the CBD, and a trip including a tunnel transit. The good quality trip has great GPS satellite signal coverage throughout the time, whereas the CBD trip suffers from multi-path effects caused by prevalent high buildings, mirroring the satellite's signal and resulting in inaccuracies. The tunnel transit trip losses GPS signal during transit resulting in gaps with missing sample points.

Observable higher values for the position based method are likely due to drift effects, where a vehicle stood still but the GPS logger continued to detected slight motions and recorded positions changes. The tendency for longer distances in case of the road segment based method, probably has its origin in map-matching errors. Map-matching often results in over-assignment of road

TABLE 5.4: Comparison of various distance computation methods

Method	Good Quality	CBD	Tunnel Transit
Reference Google Maps value	8.7 km	17.5 km	24.6 km
Position based	9.3 km	19.4 km	24.1 km
Speed based	8.5 km	18.1 km	22.6 km
Road Segment based	8.9 km	18.7 km	29.7 km

segments or - in other words - considers more segments as ‘traversed’ than it would have been the truth.

5.2 Map-Matching Evaluation

The map-matching process has a big influence on proposed approaches and affects results, as it is incorporated into the static energy map and dynamic driving share approach. The evaluation takes place by comparing the correctly matched sample points to the total number of a trip’s sample points. To assess whether a point is considered as correct, a labelling tool for manual matching of trips was developed, which serves as the so called *ground truth* - a reference assignment, results are compared to.

In Table 5.5, a total number of nine trips, three for each road cluster, were randomly selected for manual labelling and evaluation process. Since the labelling process demands comparably much time, the size of the validation set was limited to save time, approving to be only capable of giving an outline of implemented map-matching’s performance. A thorough evaluation of the original ST algorithm can be obtained from the regarded paper in [14].

TABLE 5.5: Map-matching evaluation for various road type clusters

Road Cluster	Total Points	Correctly Classified	Share	Average Share
Motorway	1465	1273	86.9 %	84.1 %
	1308	1103	84.3 %	
	1197	971	81.1 %	
Road	1294	1097	84.8 %	80.7 %
	925	731	79.0 %	
	1212	948	78.2 %	
Urban	1485	1219	82.1 %	78.0 %
	1076	830	77.1 %	
	932	694	74.5 %	

The map-matching implementation is designed to perform well on relatively low sampling rates or periods. The normal sampling period, trips are map-matched throughout the analysis, is 60 seconds. Note that the original sampling period of logging data, which is one second, is indeed shorter and was artificially increased to save computation time. This leads to the sole consideration of a subset of each 60th sample point. For comparison reasons, results in Table 5.5 are also derived from map-matched trips with a 60 second sampling period.

For evaluation, three clusters (Motorway, Road, Urban, as already used beforehand and derived from ARTEMIS) were set up because different results for each were expected; evaluation outcomes confirmed this. On a trip with a high Motorway share, good results were anticipated due to

a wide satellite coverage and less disturbing factors despite tunnel transits at MCE (Marina Coastal Expressway) or Chin Swee Tunnel as part of the CTE (Central Expressway). GPS losses during tunnel transits, however, are mitigated considerably good by the global ST map-matching algorithm as it takes the whole trip into account and can estimate missing values. An Urban trip, on the other hand, is likely being adversely affected by GPS deviations which often occurs under Singapore’s conditions with a huge number of tall buildings.

The effect of different sampling rates on map-matching is likewise evaluated and presented in Figure 5.6 for the average shares of Table 5.5. As apparent from the trend, in general, an increase of sampling rate leads to a decrease in the share of correctly classified sample points. However, the effect is rather slight, correlating to the original paper’s results. Remarkable is the fact that this trend does not apply for the worse performing ten second sampling period which may can be accounted for the tailored design for lower sampling rates.

Regarding the different road type clusters, the trips of cluster Urban are expected to deteriorate with lower sampling rate, whereas the map-matching quality of cluster Motorway shall remain considerably good. Those anticipations are in line with the presented results.

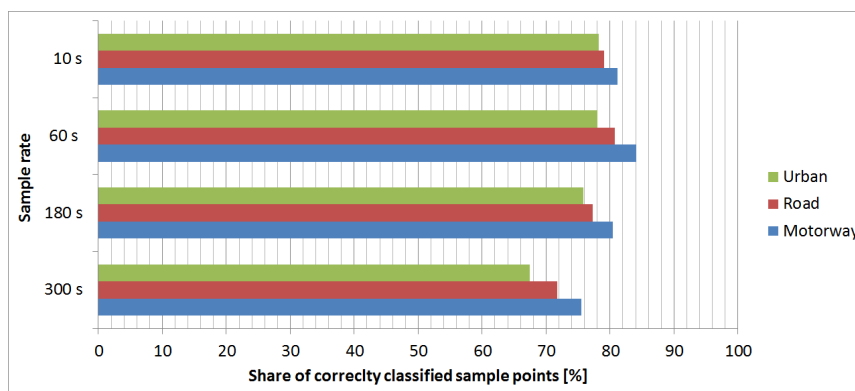


FIGURE 5.6: Map-matching evaluation for different sampling rates on various road categories

The impact of different connectivity matrices, as introduced in Section 3.2.1, is illustrated in Figure 5.7 in which three different matrices are employed for routing a trip from origin to destination. The point count based connectivity matrix lacks of reliable distance information because it solely uses the number of points between two nodes and is heavily dependent on the mapper’s style and the road’s shape. Hence, the routing chooses an unreasonable path omitting speed constraints or expectable traffic conditions a real driver would probably have taken into account. The connectivity matrices incorporating the Euclidean distance and time demand perform better and align with the actual taken route. Particularly the time demand connectivity matrix resembles the real trip almost impeccably.

5.3 Dataset Statistics

The cooperation with SMRT is of mutual benefit: on the one hand, the received data is used for the in this thesis presented investigations and for future related projects, and on the other hand, during monthly update meetings, so far conducted analysis were presented and supported SMRT in understanding driving patterns or may even affecting strategic decisions. A subset of most insightful statistics is presented in this section, which has also aided to understand the data and the underlying patterns. All diagrams are based upon data in the range from 8th June

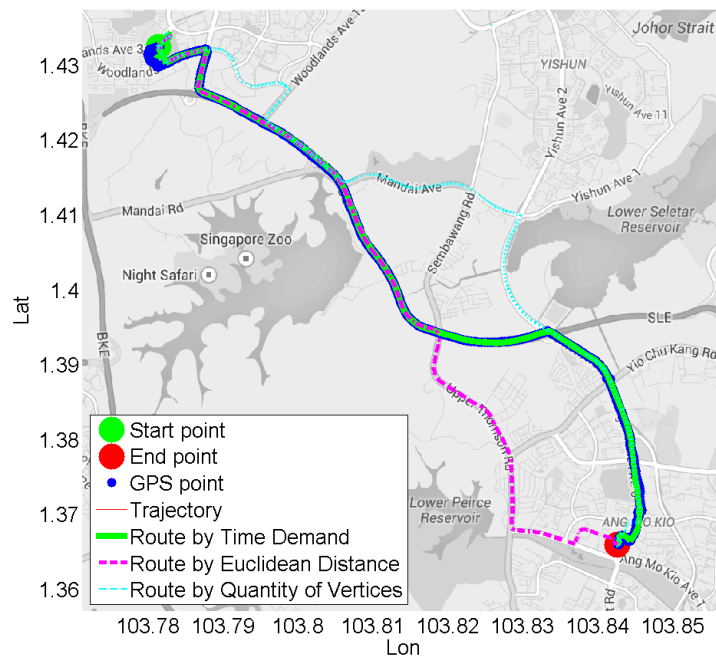


FIGURE 5.7: Routing for different connectivity matrices based on *Point Count*, *Euclidean Distance* and *Time Demand* as introduced in Section 3.2.1, background image taken from [30]

to 9th September. The presented results are depended of chosen parameters as introduced for several implemented programs. A complete list of those parameters is attached in Appendix H. In addition, parameters are subject to change as they are obtained empirically, and new outcomes may require adjustments. The list therefore does not claim to be the perfect set of choice, but rather reflects the optimal values to the best of believes.

Beforehand, basic statistics about monthly datasets are derived to assess conducted pre-processing, filtering, interpolation, and extraction steps. The batch for September is presented in Table 5.6. It already gives a good insight into dataset quality by computing the mean, extrema, and 5% and 95% percentiles denoting the boundary values in which 90% of all values settle. A complete list of those statistics for each month is attached in Appendix F.

5.3.1 Speed Profile

The diagram in Figure 5.8 illustrates the average daily speed distribution on Singapore's roads, distinguished for hired trips (blue graph) and all recorded trips (green graph). As anticipated, the highest speed values occur during night when traffic flow is low and roads are free.

There are two significant drops in average speed values during morning and evening peak hours due to upcoming commuting traffic and resulting congestion. During the day, traffic eases and speed values slightly increase; but they interestingly remain on a comparably low level. The evening drop is more significant than the morning's, which is may due to the fact that traffic congestion has not recuperated entirely. This speed profile analysis also contributed in finding suitable time periods for a 'blowing-out' of engine combustion remains, which requires driving periods greater 50 km/h for at least 15 minutes.

TABLE 5.6: Trip set statistics for September, denoting values for mean, minimum, maximum, 5 % percentile and 95 % percentile

Feature	Mean	Min	Max	5%	95%
Shift Duration [h]	7.24	0.01	21.05	0.79	11.91
Shift Mileage [km]	169.70	0.00	500.83	12.17	342.04
Hired Status [%]	54.00	0.00	100.00	0.00	82.50
For-Hire Status [%]	29.72	0.00	100.00	0.00	56.17
Other Statuses [%]	16.28	0.00	100.00	0.00	63.77
Trip Duration [min]	12.53	0.50	632.38	1.77	31.76
Trip Length [km]	6.25	0.00	178.45	0.11	21.89
Trip Velocity [km/h]	29.82	0.00	150.56	0.00	85.46
Hired Trip Duration [min]	17.00	0.50	585.40	3.62	36.35
Hired Trip Length [km]	9.97	0.00	178.45	0.94	25.64
Hired Trip Velocity [km/h]	35.11	0.00	132.90	0.00	88.25
Altitude [m]	19.32	-18.84	71.09	8.41	34.64
Altitude Incline [%]	0.00	-15.32	17.11	-1.62	1.62
Velocity Start of Trip [km/h]	1.23	0.00	27.94	0.00	7.45
Velocity End of Trip [km/h]	1.70	0.00	100.88	0.00	7.11
Stand still period distribution [%]	29.14	0.00	100.00	4.87	75.69
Pause duration [min]	28.32	5.00	179.57	5.40	96.22
Energy Consumption [kWh/100 km]	17.99	7.46	79.96	12.92	28.95
Hired Energy Consumption [kWh/100 km]	16.43	9.80	78.64	13.10	20.71

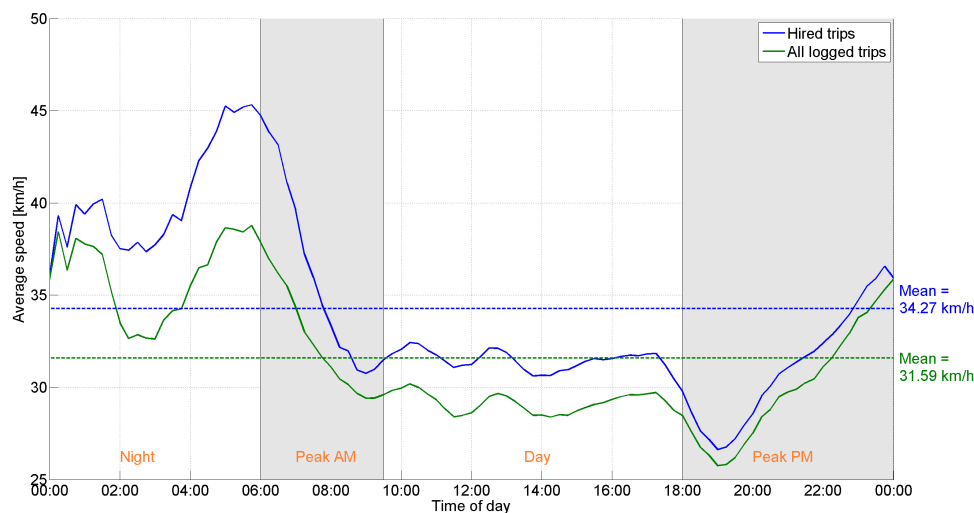


FIGURE 5.8: Average speed distribution subject to time, averaged in steps of 15 minutes over one day

5.3.2 Booking Status Distribution

In Figure 5.9; the status share of all taxis averaged for one day is presented. The distribution reveals a clear disparity between day and night. During night, generally less driving is seen with a maximum of 78 % Log Off share and a minimum of 16 % Hired share at 4 am. During daytime, by contrast, inactive times reach its minimum at 3 pm with less than 5 % Log Off mode. The most productive times take place during peak hours, one at 9 am with 40 % share of Hired status, and one at 7 pm reaching the absolute maximum Hired share of 41 %.

Busy status is seen more often during the day since drivers apparently do not use the taxi for their own purposes during night. Overall, a fairly small On Call status is observed, indicating a rather short waiting time for passengers after requesting a taxi booking. All here discussed points are in line with prior contemplated expectations and thus prove the booking data quality good. The distribution is of value for the status based trip extraction method.

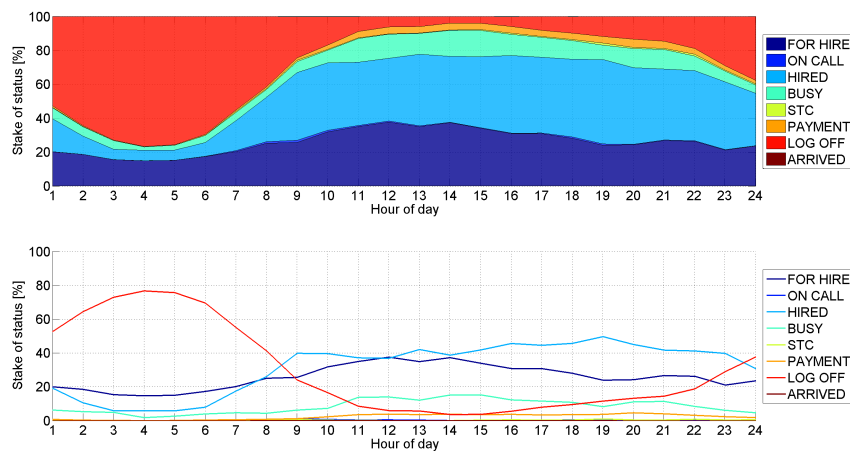


FIGURE 5.9: Status distribution averaged over one day in steps of one hour

A further separation into one- and two-hirer scheme taxis is provided in Appendix E, from which a remarkable fact can be observed that, despite technically more drivers using the same vehicle, the status distribution is almost equally distributed for both schemes.

Figure 5.10 illustrates the overall status distribution disregarding the time of day for all taxis, divided into one- and two-hirer schemes. The Log Off status is included in the upper plot and excluded in the bottom one to gauge the implication of non-active times.

It is apparent that one-hirer scheme taxis generally reveal a higher productivity with a Hired share of 53 %, outperforming the two-hirer scheme taxis with only 44 %. The higher Busy share of 14 % for the latter versus 9 % for the former gives a second hint that hirers may use the taxi more often on personal purpose or have to spend more time for commuting between the agreed shift change locations. Remarkable is the fact that the Log Off share, as perceivable in the upper chart, is not significantly lower for the two-hirer scheme as for one would have expect.

Together with results of daily status distribution in Figure 5.9 and Appendix E, it hence can be concluded that a two-hirer scheme not necessarily implies two shifts with two shift changes each day but can also indicate a taxi alternation on a day-to-day basis. This is of significance for the shift extraction method.

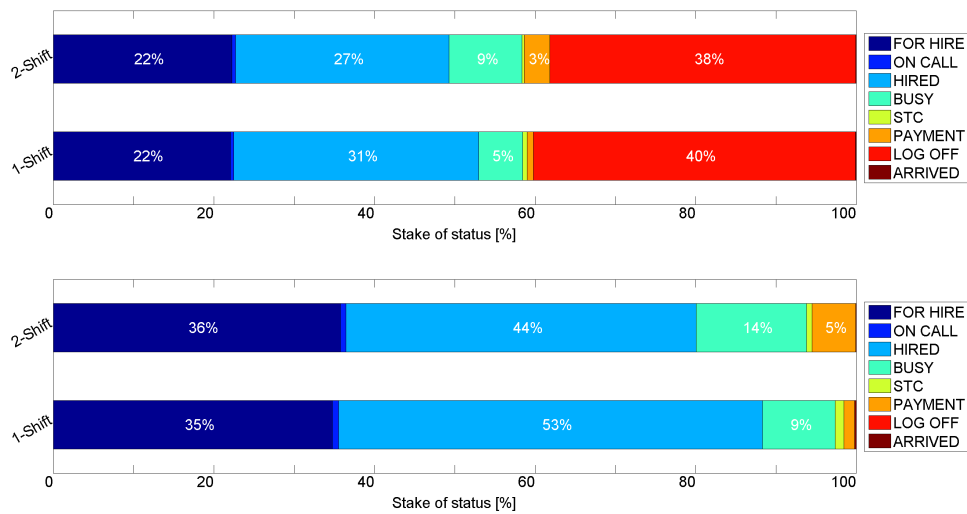


FIGURE 5.10: Status distribution per one- and two-hirer schemes, upper plot including Log Off times and bottom plot excluding Log Off times

5.3.3 Mileage and Distance Analysis

The overall, average daily mileage and its share of hired trips for the recorded set of 20 taxis is depicted in Figure 5.11. A threshold marking the 250 km value is included, which originates from a targeted daily mileage as enforced by the LTA as a criteria for taxi companies to retain their licence. Singapore is comparably strict in issuing taxi licences, requiring not only a minimum daily mileage but also a minimum fleet magnitude.

Noticeable is the high amount of taxis, which seem to target exactly this 250 km threshold. For further investigations, the taxis are clustered into four categories, low, medium, high, and very-high mileage as worked out in Section 5.3.5. In congruence with the expectations, the average daily mileage settles at 306 km, mainly influenced by the very-high mileage cluster as only seven taxis excel the mean value.

Generally can be observed that the productivity, denoted by the length of the coloured bar, is higher for low and medium mileage clusters, which is in line with the results presented in Figure 5.10. Drivers' taxis in the very high mileage cluster seem to use the taxi in a broader sense, and the productivity can be lower because still a high absolute mileage of hired trip is reached.

Figure 5.12 shows the overall daily mileage distribution distinguished by one- or two-hirer scheme. As expected, the one-hirer scheme taxis drive less than its counterpart, but the impact is much lower than anticipated. This again hints to the likelihood that both hirers of the same taxi do not inevitably share their taxi throughout the day with distinct shift changes, which would result in a much higher daily mileage because the taxi is used more frequently. Both distributions are more or less Gaussian shaped, finding their mean value at 260 km for one-hirer and 326 km for two-hirer scheme taxis.

As can be seen in Figure 5.13, the average hired trip distance is distributed equally among different vehicle types and has its mean at around 10.5 km, which is a bit higher than LTA data would suggest (discussion are to follow in Section 5.4). The two luxurious vehicle types (Chrysler in orange and Ssangyong in red) slightly exceed the average trip distance, attributable to the fact that luxurious vehicle are more reasonable for longer trips.

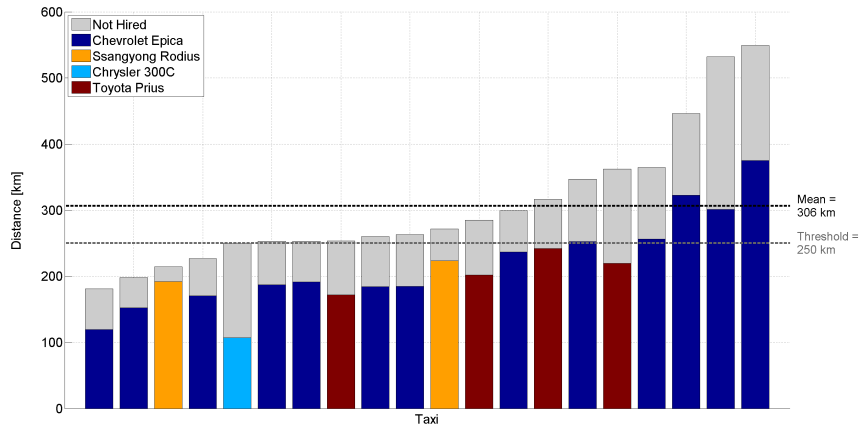


FIGURE 5.11: Average daily mileage and share of hired trips for each vehicle type, ordered ascendantly by mileage

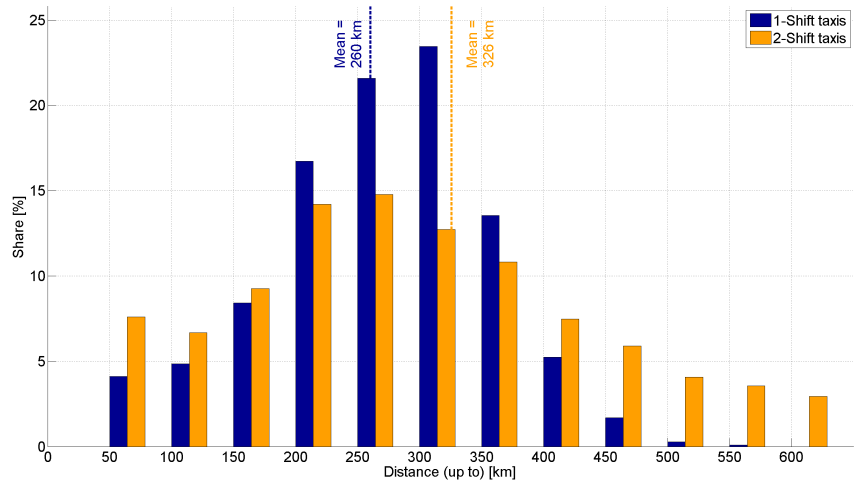


FIGURE 5.12: Daily mileage distribution for one- and two-hirer scheme taxis

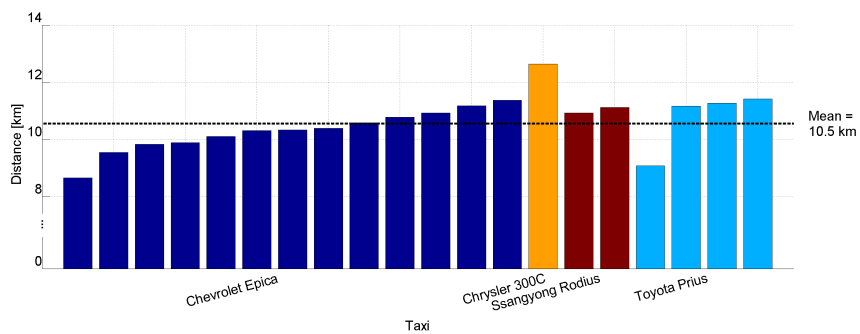


FIGURE 5.13: Average hired trip distance among different vehicle types

Figure 5.14 reveals the distribution of hired trip distances distinguished by one- and two-hirer schemes. Noticeable is the high ratio of short trips with a peak at around 5 km. As assumed, the share of longer trips decreases hyperboloidal and reaches a minimum share of less than 1% for trips longer than 30 km. Nevertheless, a lot more small trips are encountered than expected, leading to an unequal distribution.

An explanation might be given by the fact that Singaporean taxis are comparably cheap in relation to average income; hence, a lot of people can afford taxi journeys. In some cases, taxis even tend to be the preferred means of transportation because public transport infrastructure, which is admittedly increasing dramatically and is of decent quality, does not yet cover all areas as pervasively as desired. For instance, plans attempt to increase the share of MRT rides of roughly 30% in 2013, preliminary by substituting bus journeys, which have the highest share of currently about 60% [52]. Furthermore, Singapore is not big in extent which has a confining affect on trip length.

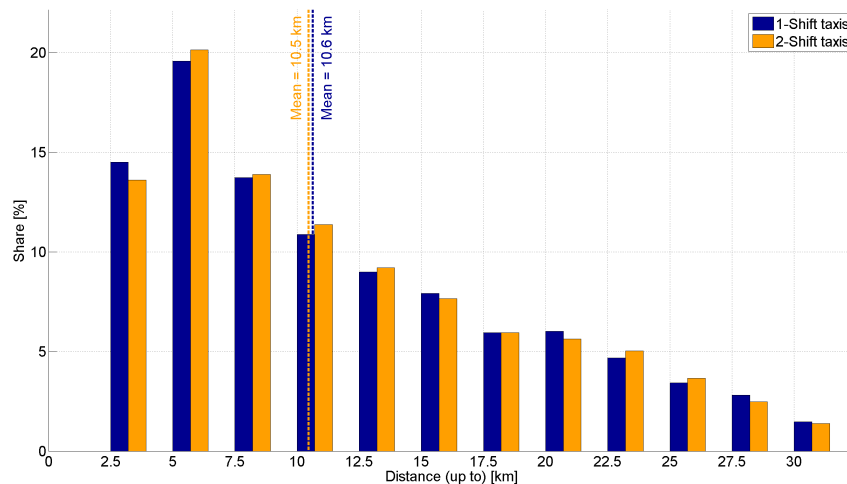


FIGURE 5.14: Hired trip distance distribution for one- and two-hirer scheme

5.3.4 Region related Analysis

A general idea about driver's favoured locations is illustrated by means of a heat map in Figure 5.15. Heat maps are a convenient tool to present distinct features from a data set in an easy conceivable, graphical way. Its construction is explained in detail in Section 4.6.1. As a reminder, the colour code indicates the quantity of occurrences in a predefined cell on a meshed grid. A blue pixel indicates a low occurrence probability, whereas a red pixel marks frequently visited locations. The scale is chosen to be logarithmic to cover the wide range and extensively unequal occurrence distribution.

A video was created which concatenates a number of those heat maps on basis of time depended trips. It shows which roads are used more frequent and which parts of Singapore are more often visited for different times of a day. In this way, it reveals more distinctiveness than one presented plot would be able to. The video can be found on the attached CD.

A peculiarity might be seen in the blurred areas around the CBD and the Southern coastline, former originating from poor GPS quality caused by typically high building and multi-path effects as described in Section 4.2.2, and latter caused by tunnel transits requiring interpolation and affecting the GPS trajectory as apparent.

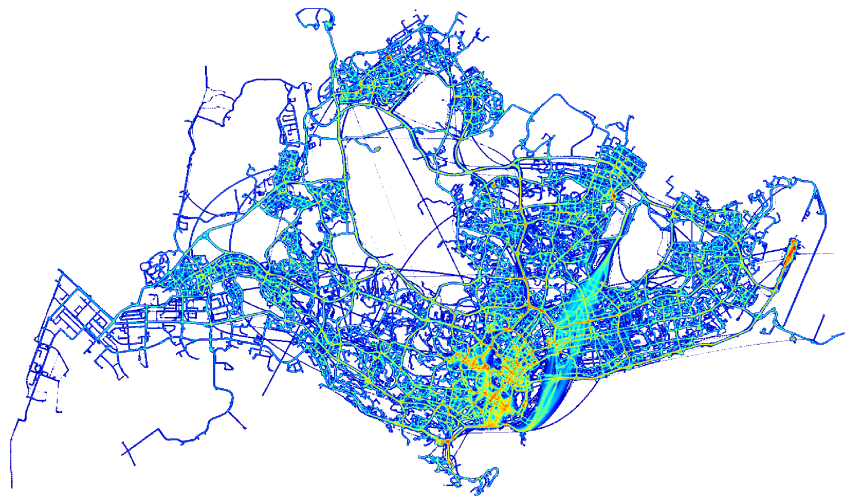
FIGURE 5.15: Occurrence frequency heat map on a 200×300 grid

Figure 5.16 illustrates the distribution of taxi occurrences for trip origins and trip destinations subject to Planning Areas, denoted by the intensity of the colours as referenced in the colour-bar. A trip origin is hereby defined as the location where a trip has started; a trip destination is consecutively defined as the location a trip has ended.

A similar distribution is seen for origins and destinations. This indicates the anticipated driving pattern that drivers tend to start a subsequent trip in the same area, searching a new passenger in the vicinity rather than returning to a fix spot. It reveals a big distinction towards German taxi patterns where passenger pick-ups largely happen at fix locations, namely taxi stands. Additionally noticeable is the higher occurrence in the South-East area, highlighting the airport, CBD, and the amusement quarter Kallang.

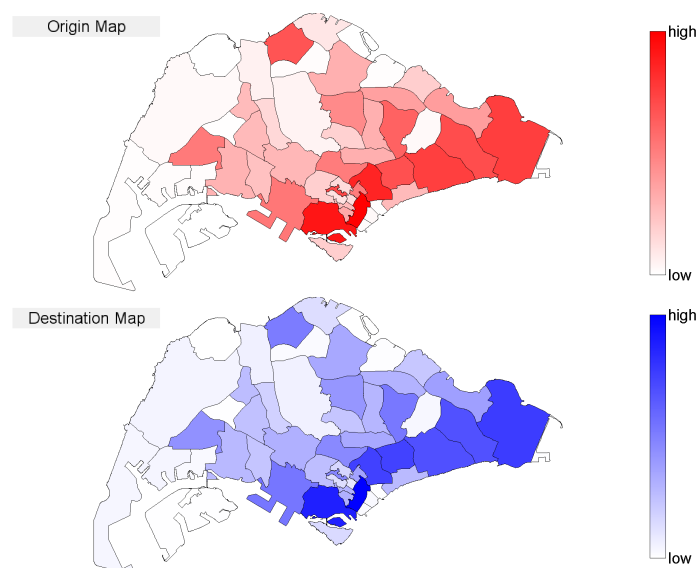


FIGURE 5.16: Origin/Destination map for hired trips, showing origins in the top subplot and destination in bottom subplot, indicating concurrence frequency per Planning Area by colour intensity

A related plot is presented in Figure 5.17, emphasising differences between previously shown origins and destinations distribution. Priorly hidden, it now indeed reveals insightful features for Planning Areas showing, for instance, a high amount of trips starting from the SMRT Woodlands depot and a high amount of trips with destination Kallang.

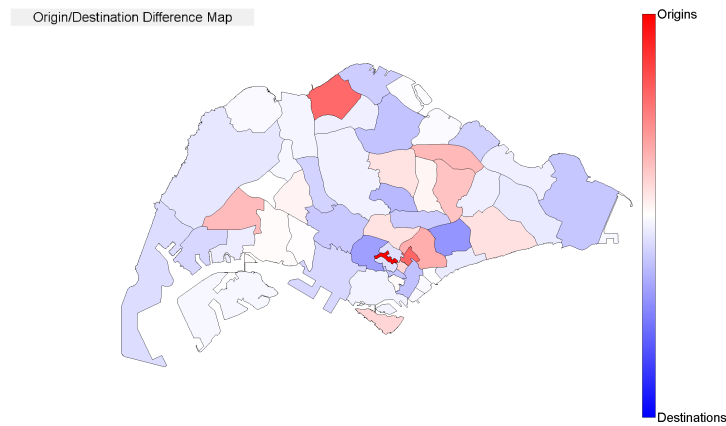


FIGURE 5.17: Origin/Destination difference map for hired trips, showing the difference in origin and destination frequencies per Planning Area by colour intensity

5.3.5 Cluster Analysis

After clustering taxis according to daily mileage into four distinct categories, each is submitted to a separate investigation. This addresses the average total daily mileage, average daily mileage with status Hired, herefrom derived productivity evaluation, and the active times searching for customers. Results are presented in Figure 5.18, from which previous conclusions are affirmed: drivers with a relatively small daily mileage tend to be more productive as they use their small amount of driving to a huge extent for Hired trips.

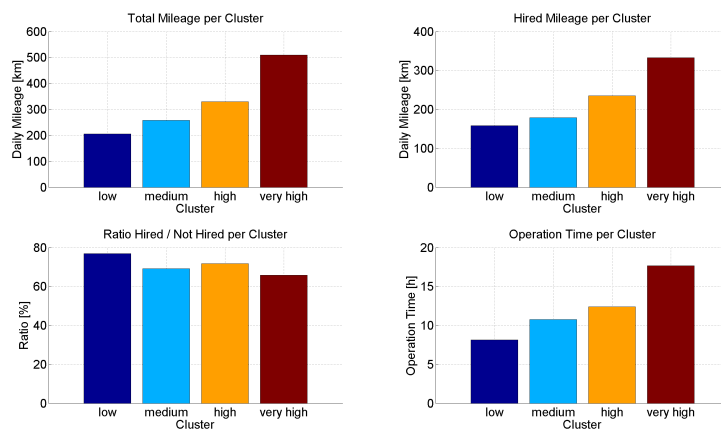


FIGURE 5.18: Cluster statistics displaying total mileage, mileage of Hired trips, ratio hired/not hired, and operation time

5.4 Comparison and Validation with LTA Data

In Table 5.7, statistics of this work’s analysis versus statistical data from LTA sources are presented. The first two rows for one- and two-hirer schemes display the latests accessible data as from December 2013 and was taken from [7]. This source, however does not include statistics for all taxis where the distribution of both hirer-schemes is impactful. To gather missing values, another source [53] is hence queried, reflecting the entire year 2012. It is noted, the term *hirer-scheme*, as used by SMRT, is considered a synonym to *shift*, as given in the queried sources.

There are basically two noticeable things apparent: both LTA data source do not straightforwardly fit together, as the average daily mileage for all taxis exceeds even the maximum of both single values. Conceivable sources of deviation are a change in driving patterns between 2012 and 2013 or a change in means of measuring of daily mileage. The average trip distance however is in line among both sources.

Regarding the comparison of LTA data with evaluated statistics, in principal a considerably good match is observed, especially in average trip distances not exceeding a 10 % deviation. The captured average daily mileage, by contrast, surpasses the provided LTA benchmarks of the first source by 12.0 % and 27.6 %, respectively, whereas it undercuts the values provided in the second source by 7.7 %. The good match in average trip distances confirms developed trip extraction algorithms. The major differences in daily mileage are likely not caused by incorrect programs but are rather due to different data sources presented numbers rest upon and different definitions of hirer schemes and shift systems.

TABLE 5.7: Comparison of evaluated statistics with LTA data

Hirer Scheme	Statistic	LTA Data	Examined Data	Deviation
One-hirer scheme	Average daily taxi trips	20.2	24.6	+21.8 %
	Average mileage per trip	10.1 km	10.6 km	+5.0 %
	Average daily mileage	204.0 km	260.3 km	+27.6 %
Two-hirer scheme	Average daily taxi trips	29.4	31.1	+5.8 %
	Average mileage per trip	9.9 km	10.5 km	+6.1 %
	Average daily mileage	291.1 km	326.1 km	+12.0 %
All taxis	Average daily taxi trips	34.3	29.2	-14.9 %
	Average mileage per trip	9.7 km	10.5 km	+8.2 %
	Average daily mileage	332.5 km	306.8 km	-7.7 %

Chapter 6

Energy Demand Analysis - Methodology

This chapter introduces the proposed energy estimation approaches and outlines the conceptualization of a corresponding energy database, which bases on successfully pre-processed and map-matched taxi GPS trajectories as described in Chapter 4.

In principle, there are two distinct approaches conceivable towards energy estimation: first, using a vehicle model and directly simulate the car’s dynamic behaviour in form of a speed profile, and second, utilizing historical data such as previously recorded GPS trips, environmental, or traffic data to set up a database. Moreover, a combination of both intentions is feasible, exploiting the car’s actual dynamic characteristics on the one hand, enhanced with robust historical background information on the other. The former vehicle model concept is outlined in Figure 6.1.

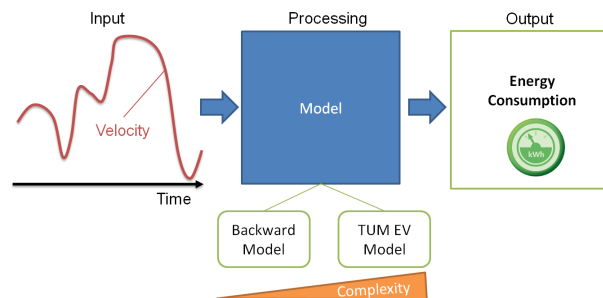


FIGURE 6.1: EV model application scheme

As mentioned earlier, the motivation behind this energy demand analysis is to incorporate obtained results in form of energy estimation programs into a sophisticated taxi simulation in Singapore, addressing the impacts of a high quantity of electric taxis on the power grid. As a requirement and input for developed programs, a wide range of driving profiles’ level of detail is anticipated, including a rather rough estimate of taxi’s chosen route. This is accounted for by a simulation’s natural of abstraction and simplification. The energy demand estimation approaches are therefore tailored to support low sampling rates, and a high leverage of historical and environmental data is utilized.

In detail, the investigation of a great number of real-world GPS trajectories recorded by a subset of 20 SMRT taxis, establishes an energy database - using a static energy map on the one hand and a dynamic driving share approach on the other hand.

As a simplified outline, each road segment of Singapore’s road network is tagged with an average energy consumption and an distribution of average driving shares (distribution of state of driving like idling, cruising, accelerating, or deceleration, see Section 6.3), and additionally different time zones and road types are regarded. When attempting to compute energy consumption for arbitrary trips, this database is harvested by retrieving the corresponding energetic characteristics for each traversed road segment.

6.1 Electric Vehicle Models

Electric vehicle models are pervasive for computing energy demands of driving cycles or real-world driving trips in various application areas, including official consumption assessment tests for new cars. A typical input for a vehicle model is a trip’s dynamic representation in form of a speed and altitude profile subject to time. The combination of both is defined as a *driving profile*.

There are various models available for different simulation purposes. One, for instance, is the PHEM [22] which emphasises on vehicle’s emissions. To meet targeted requirements and to align to the context of electrical vehicles, two in-house developed models are of choice: the TUM CREATE EV Model [54] and the Backward Model (for implementation see Listing M.20 in Appendix M); latter is a derived and simplified version of the former.

To allow for a comparison among both models and to link the model to a distinct vehicle type, a uniform set of vehicle characteristics is applied as indicated in Table 6.1. Most of herein presented values are adopted directly from EVA characteristics, an electric taxi prototype developed and manufactured by TUM CREATE [27]. Unattainable data was, as applicable, taken from state-of-the-art books and papers as corresponding reference is given in the table.

The auxiliary energy demand is distinguished between a base load for on-board power consumers (such as electronics, light, and infotainment systems) and air conditioning, which affects energy consumption in tropical cities like Singapore to a high extent. For both energy consumers, with regard to simplicity, a constant value is assumed, which approximately holds for prevalent conditions of rather constant temperatures and humidity throughout the day and year [56]. Section 6.1.3 outlines a more comprehensive approach suggested for further investigations which substitutes an assumed constant auxiliary power demand with a fine-grained auxiliary model.

To perceive an idea about the impact of recuperation as a major benefit of electric vehicles, Figure 6.2 compares consumed and recovered energy during a simulation on several ARTEMIS standard driving cycles. It is apparent that energy recuperation has a bigger impact under generally slower driving conditions, as for instance prevalent in an urban environment, than it has on motorways with typically higher speeds. This is accounted for by an higher share of Stop & Go traffic and because recuperation only occurs while breaking.

Furthermore, Figure 6.3 illustrates the share of drivetrain power in comparison to auxiliary and air conditioning power demand. Similar to the energy recuperation distribution, the share of auxiliary power demands decreases with faster driving conditions due to a higher influence of air and roll resistance. Remarkable is the high share of 43% spent for auxiliary power, which does not contribute to the actual driving purpose. It furthermore reveals the high importance of energy saving air-conditioning systems as in case of Singapore the humid air first has to be dried before the actual cooling process can be started.

As a conclusion, the electric vehicle model should be capable of predicting energy demands correctly under various driving conditions. Neither recuperation nor auxiliary submodules can be neglected as it influences total energy demand significantly.

TABLE 6.1: Electric vehicle characteristics for EV models

Category	Characteristic	Symbol	Value	Source
Basic Characteristics	Mass	m_{car}	1500 kg	[5]
	Wheel radius	r_{wheel}	0.3 m	[54]
	Rolling resistance coefficient	F_r	0.008	[54]
	Drag coefficient	c_w	0.34	[5]
	Frontal area	A_{front}	2.8 m ²	[5]
	Cabin volume	V_{cabin}	5 m ³	[5]
Dynamic Characteristics	Maximum power	P_{max}	60 kW	[5]
	Maximum torque	M_{max}	223 Nm	[5]
	Top speed	v_{max}	111 km/h	[5]
Passenger Characteristics	Number of passengers	$N_{passenger}$	2	[54]
	Passenger weight	$m_{passenger}$	75 kg	[54]
Battery Characteristics	Battery capacity	$C_{battery}$	50 kWh	[5]
	Battery energy density	$\mu_{battery}$	110 kWh/m ³	[5]
	Initial SOC	SOC_{init}	0.9	[54]
	Lower SOC limit	SOC_{lower}	0.1	[54]
	Usable SOC range	SOC_{usable}	0.8	[54]
Auxiliary Characteristics	Air-conditioning power	P_{aircon}	1500 W	[55]
	Ambient temperature	$T_{ambient}$	27 °C	[56]
	Ambient humidity	$\phi_{ambient}$	84 %	[56]
	Desired cabin temperature	T_{cabin}	23 °C	[54]
	Desired cabin humidity	ϕ_{cabin}	60 %	[54]
	On-board power supply	$P_{auxiliary}$	500 W	[55]
Efficiency Characteristics	Transmission efficiency	$\eta_{transmission}$	0.95	[57]
	Engine efficiency	η_{engine}	0.94	[57]
	Inverter efficiency	$\eta_{inverter}$	0.97	[57]
	Charging efficiency	η_{charge}	0.85	[57]
	Discharging efficiency	$\eta_{discharge}$	0.95	[57]

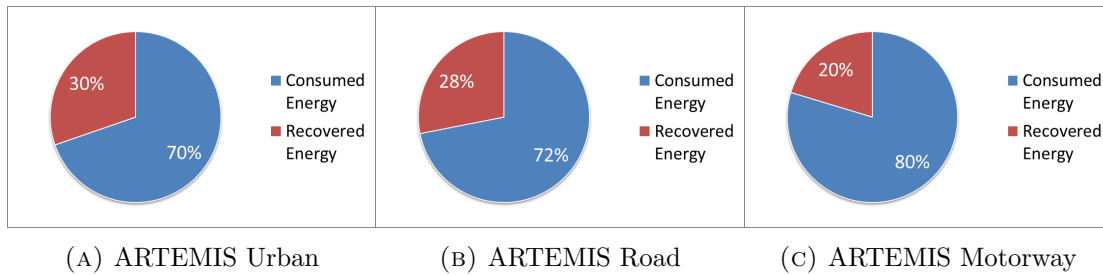


FIGURE 6.2: Share of consumed vs. recovered energy for different ARTEMIS [50] standard driving cycles

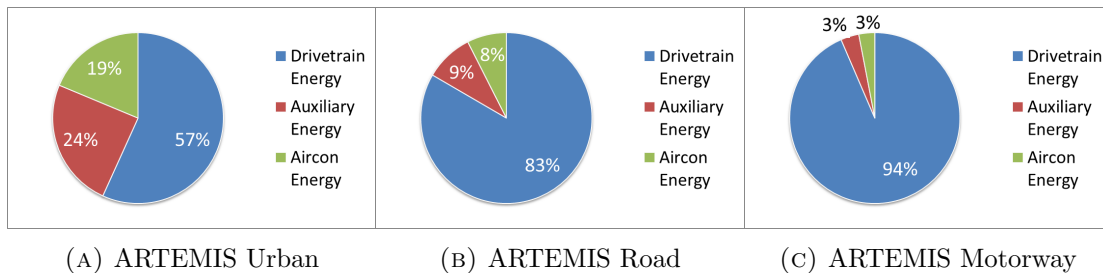


FIGURE 6.3: Share of energy consumers for different ARTEMIS [50] standard driving cycles

6.1.1 TUM CREATE EV Model

The most comprehensive electric vehicle model accessible is the TUM CREATE EV Model [54], which is implemented in Matlab Simulink and thus easily adoptable into the established development environment. An overview of its first layer is given in Figure 6.4, comprising all submodules and basic interconnections. Since the model is exclusively utilized and by no means adjusted respect to its internal structure, a further functional description would go beyond this work's scope. This model is used for result validation of proposed energy estimation approaches and for comparison with the simplified Backward Model.

The model's main advantage is a thoroughly modelled and complete set of submodules, including *Driver*, *Environment*, *Vehicle Dynamics*, *Drivetrain*, *ESS* (Energy Storage System), *Auxiliaries*, and *Controller*, which together regard each possible energy consumer on board an EV. Drawbacks, on the other hand, are its comparable long computation time and sensibility for peaks in speed profile. Both are due to an extensive usage of differential equations as they are commonly employed in Simulink. Moreover, its in-depth modelling character exceeds the requirements for this study's anticipated usage scenario.

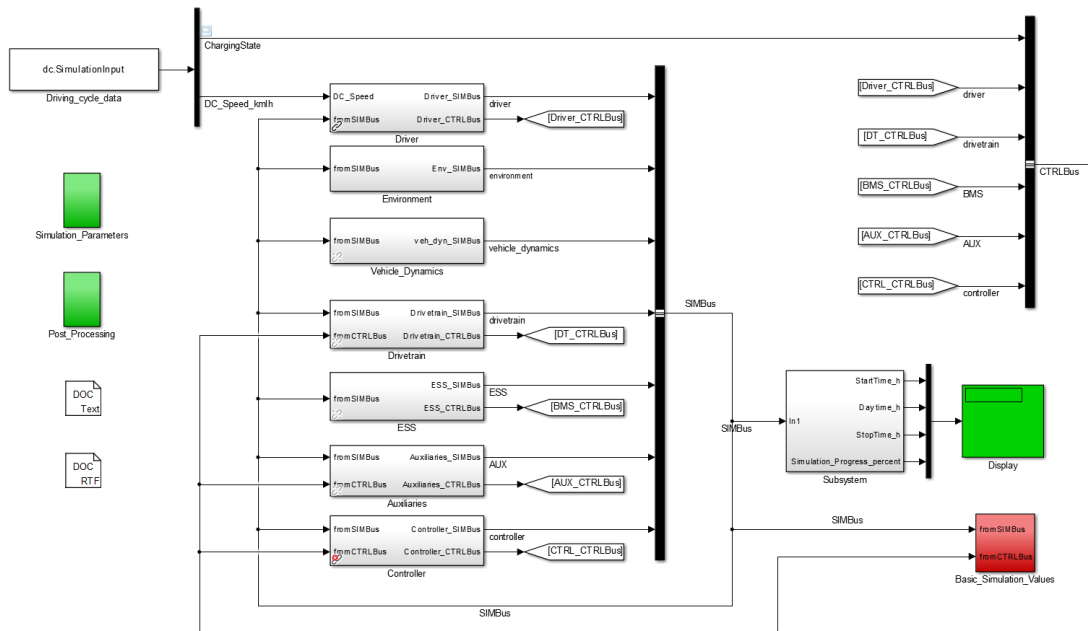


FIGURE 6.4: First layer of the TUM CREATE EV Model, adopted from [54]

6.1.2 Backward Model

Particularly the low computation time of the TUM CREATE EV Model matters to implement a simplified, faster version without limiting the estimation quality. Since it is aimed to take all so far captured trips (more than 77,000, see Table 3.1) into consideration to set up a meaningful database in terms of quality and quantity of covered road segments, it was unavoidable to use a less time consuming model. Feasible for a high amount of trips, robust and less sensitive to dynamic outliers, the purely Matlab based Backward Model is used (for implementation see Listing M.20 in Appendix M).

In its principal functionality, it computes the required power of each submodule for each time step on basis of the trip's speed and altitude profile. Resulting power values are amplified with

underlying characteristic EV efficiency coefficients, and the resulting energy is calculated as the sum of all power shares multiplied with the time spend for the examined trip.

There is no necessity for any control loops as it is only looked into a small portion of two adjacent sample points at each step. This, however, reveals some drawbacks as possible impactful interferences between submodules are omitted, which may would have been covered in control loops. The incorporated submodules are schemed in Figure 6.5.

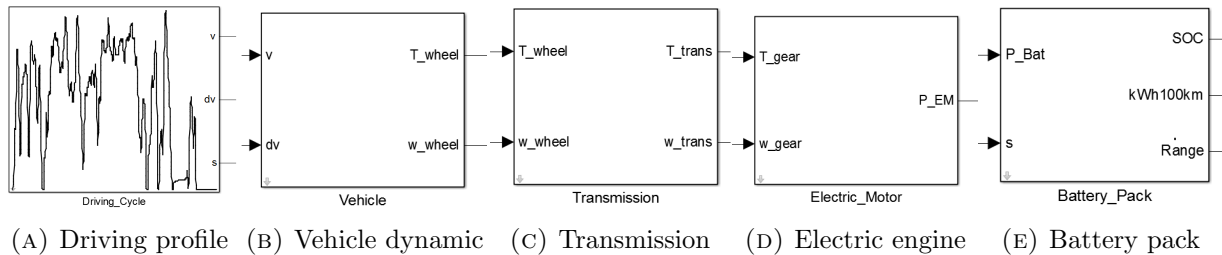


FIGURE 6.5: Backward Model submodules

6.1.3 Model Tailoring and Enhancements

Both models are adjusted to fit into the applicable Matlab environment. The Simulink based TUM CREATE EV Model requires an interface for direct access from Matlab, which in principal passes trip's dynamics from the Matlab workspace into the Simulink workspace.

The Backward Model comes without a gradient model which is meant to reflect the road slopes as part of the vehicle dynamics submodule. This had to be implemented manually. A gradient model computes the quotient of height difference and Euclidean distance between two adjacent sample points and rescales values the height disparity per 100 m.

As mentioned, a vehicle model relies on a set of vehicle characteristics directly influencing the energy consumption. In an initialization step, these characteristics are imported into both models to allow for comparable results and to follow the dynamic behaviour of EVA, from which most of the characteristics are taken from.

Further recommended model enhancements would include the design of an auxiliary model, capable of estimating required auxiliary power in greater detail. Auxiliary consumers comprise on-board power supply, infotainment, heating, and air condition. In case of prevalent climatic conditions in Singapore, air condition has the biggest share of all auxiliary power consumers.

In the current implementation, a constant value of 500 W is assumed for the on-board power supply and infotainment system, and an additional 1500 W for the air conditioning system. An comprehensive auxiliary energy model, though, would use speed, ambient temperature, and sun angle to compute first, the vehicles surface temperature and second, the hereupon derived demanded air conditioning power.

6.2 Static Energy Map Approach

In principal, an energy map is denoted as a database which links average energy demands to an underlying road network. In the pervasive context, each of the 40,425 Singaporean road segments is assigned with a mean energy consumption plus an additionally set of four time dependent values according to the introduced time zones in chapter 4.6.

By energy consumptions, values scaled to the unit of kWh/100 km are denoted to allow for independence of road segment length, especially applicable in the stage of constructing the energy map. Although a huge number of recorded trips is incorporated, it cannot be expected that each road segment has been traversed as frequent as it would be required for a thorough energy inspection. In these cases, it is resorted on road type's mean energy consumptions, which have been obtained as an average of all trip's consumptions on certain road types.

There are two steps enfolded in the static energy map approach: (a) constructing the map on basis of historical data to gain average, road segment linked energy demands; (b) utilizing the map for arbitrary trips to estimate its energy consumption. Firstly, addressing the former, the map is constructed by taking all so far recorded trips into account, analysing those energetically by applying an electric vehicle model, and finally extracting mentioned five energy values. Secondly, addressing the application stage, traversed road segments of a chosen trip are extracted and all corresponding energy values are summed up. An abstract overview of the conceptualization step is delineated in Figure 6.6; and detailed descriptions are to follow in the next sections.

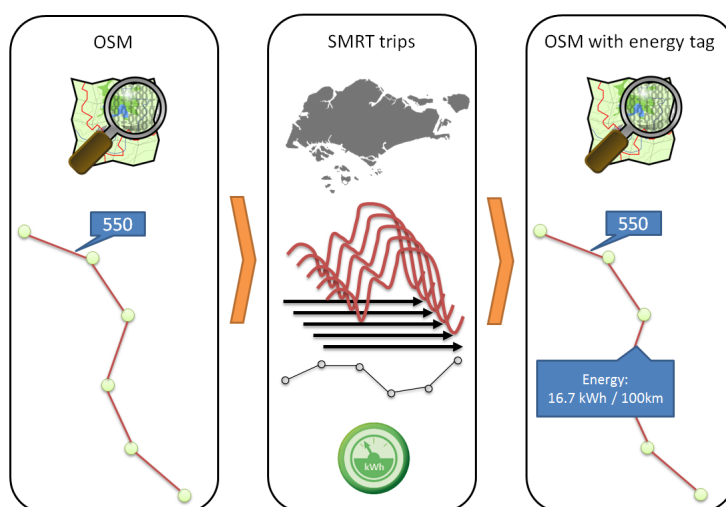


FIGURE 6.6: Principal scheme of the energy map conceptualization

As a by-product, it is targeted to export the previously imported OSM (described in Section 3.2.1) into the same OSM format for a convenient access by commonly used tools like JOSM.

This approach is considered as static by means of implicitly incorporating a set of characteristics, which are not retrospectively adjustable for different vehicle types or for varying environmental conditions. The former is due to fixed vehicle characteristics directly involved into the energy models. The latter is almost unavoidable as environmental conditions are inevitably directly reflected in the GPS trajectories, influencing computed energy consumptions.

After revising results in the evaluation unit of Chapter 7, another restriction arose, expressing the lack of treating particular traffic conditions such as traffic jams. This is due to the sole usage of averaged energy values, which are less perfect capable of reflecting individual scenarios.

6.2.1 Energy Map Construction

The first step of the static energy map approach includes the energy map conceptualization by incorporating recorded, pre-processed, and map-matched trips as established in Chapter 4. The process is subdivided into road segment and road type based construction, both enhanced by

time zone information. As aforementioned, those method do not use absolute, but relative energy values, scaled to the unit of kWh/100 km, to overcome boundary effects which are basically inaccuracies resulting from map-matching.

As described, map-matching tags the corresponding road segments to sample points and inevitably cuts the dynamic information between two adjacent points at segment transitions. When recalling micro trips on a certain road segment, this information bit cannot be recovered, resulting in incomplete dynamics and a generally smaller observed micro trip length compared to the road segment length. Boundary effects are more impactful on smaller segments, when both length do not correspond to a high degree. An example is provided in Figure 6.7, where only five sample points are map-matched to a traversed segment, whose length, as a result, does not fit to the actual segment length.

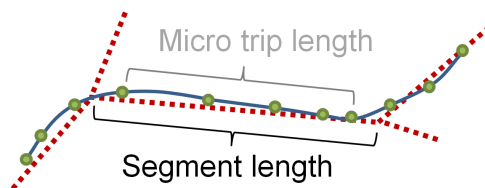


FIGURE 6.7: Boundary effects resulting in mismatch between micro trip length and road segment length

Energy Map by Road Segments

To construct a road segment based energy map, it is proceeded in two steps: first, traversed road segments are retrieved in form of micro trips; second, an energy model is applied on those fetched micro trips. Regarding the first step, another two options are faced: (a) the entire recorded trip set is browsed and micro trips are gathered according to road segment ids, which are tagged to sample points; (b) the established SQL database is directly queried by fetching all micro trips for selected road segment ids. The latter database method is more convenient and faster but was initially not available. That is why this section introduces both methods in the following.

The second step, addressing the application of an energy model, is done for both methods in the same manner and includes the actual computation of energy consumption for micro trip's driving profiles. Due to performance and robustness issues, in this stage the Backward Model is utilized as rationales are discussed in Section 6.1.2. Although it is way simpler than its TUM CREATE EV Model counterpart, evaluation results in Section 7.1 indicate a high compliance of both models.

The method implied in (a) performs at the layer of a shift set, which is basically the result of the entire driving profile analysis program as implemented according to analysis undertaken in Chapter 3. It includes a hierarchical structure comprising taxis, shifts, trips, and sample points. The algorithm skims iteratively through each layer until reaching the bottommost sample point level. It then gathers transitions in sample point's tagged road segment ids. Those transition indices are the actual points at which the traversed road segment has changed: the edge points. Next, it extracts micro trips by cutting along these edge points and hereby finally receives a set of micro trips, of which each traverses only a single road segment.

In a second stage, those micro trips are applied on the chosen energy model. The computed energy value is finally tagged to the road segment id, and described steps are repeated on behalf of all successive trips.

By querying the database directly, as stated in option (b), it can be avoided to immerse into every single sample point as it would have been required in method (a). The power of a database management system is conveniently exploited by fetching all micro trips, which took place on a certain road segment, straightforwardly by using the SQL *Select* statement alike an example query `SELECT idTrip, datetime, speed, altitude FROM DataPoint WHERE idSection = 8`. In the example, most vital trip fields are fetched, including timestamp, speed, and altitude value for road segment with id 8.

The road segment based energy map is in a sense problematic and questionable because it heavily relies on the quality of previously conducted map-matching. Since the tagging of road segment ids to sample points is solely done in the map-matching step, a good quality is crucial for the here proposed approach. Especially the traversed road segment's boundaries are of importance (see Figure 6.7), exacerbating the process of gathering the entire range of dynamics.

A threshold of a minimum amount of five sample points is introduced, determining whether to use a micro trip or not and avoiding a high fluctuation of energy values for very small micro trips. It is assured that this procedure does not dismisses high speeds as those occur in almost every case on longer road segments. The threshold hence implicitly regards all traffic conditions. Yet, the nature of a road segment is its limited length; a high variance is to some extent inevitable because micro trips on road segments tend to be short, anyway. In the examined dataset an average number of about 19 sample points per road segment were obtained.

Enhancement according to Time Zone

All taxi companies in Singapore divide a day strategically into four parts with different applicable surcharges: two peak hours, including commuting traffic home \rightarrow work in the morning (*Peak AM*) and commuting traffic work \rightarrow home in the evening (*Peak PM*), one normal, surcharge free period during daytime (*Day*), and one with a comparable low traffic volume during night (*Night*) and highest surcharge accounting for aggravating working conditions for drivers. Detailed time bounds, surcharges (additional fare as a share of the normal trip fare), and average speeds during depicted periods are listed in Table 6.2. As the diverse surcharge and average speed distribution justifies, the applicable time zone is of high interest and certainly considered impactful for the energy examination.

TABLE 6.2: Time zones characteristics indicating start and end time, applicable taxi surcharge, and average speed

Time Zone	Start	End	Taxi Surcharge	Average Speed
Peak AM	06:00 am	09:30 am	25 %	32.74 km/h
Day	09:30 am	18:00 pm	0 %	31.40 km/h
Peak PM	18:00 pm	24:00 pm	25 %	31.10 km/h
Night	00:00 am	06:00 am	50 %	37.38 km/h

Energy Map by Road Types

The energy map by road types is an eased version of the sophisticated road segment version, and only sets up a database for categories of roads, other than for actual individual roads.

The OSM road network provides 16 different major road types as listed in Table 6.3. Each is displayed and equipped with an average speed value resulting from investigations of all recorded

trajectories and serving as a traffic indicator. A distinct speed distribution among road types is evidence that traffic indeed follows patterns and behaves diverse for different categories.

TABLE 6.3: Road type overview as defined in OSM [58] with introduced clusters and average speeds, gathered from all recorded trips

Cluster	Index	Road Type	Average Speed
Motorway	1	Motorway	69.93 km/h
	2	Trunk	48.10 km/h
	9	Motorway Link	53.58 km/h
	10	Trunk Link	43.65 km/h
Road	3	Primary	33.90 km/h
	4	Secondary	27.24 km/h
	5	Tertiary	26.20 km/h
	11	Primary Link	29.71 km/h
	12	Secondary Link	24.67 km/h
Urban	13	Tertiary Link	23.75 km/h
	6	Unclassified	25.58 km/h
	7	Residential	20.15 km/h
	8	Service	18.85 km/h
	14	Living Street	30.36 km/h
	15	Track	25.73 km/h
	16	Road	-

In principal, the coarse-grained road type energy map is a subset of the fine-grained road segment counterpart. Although it is not claimed that the road type based map performs as well as the high detail road segment one, it is nevertheless expected to see decent results, capable of substituting possibly missing road segment values or even serving as a stand-alone solution.

Moreover, road types are categorized by extracting three clusters Motorway, Road and Urban, which are in line with the ARTEMIS categories of same names. Since OSM merely recommends, rather than obligates the proper usage of those 16 road types [59], the mapper does not have to follow these rules unconditionally, leading to inaccuracies as discussed in Chapter 3. Road types are hence empirically linked to the introduced clusters.

A distribution of road types among Singapore for all recorded trips is illustrated in the bar chart in Figure 6.8. More than 30% of all sample points are linked to Primary categorized roads, which are mostly the main streets or feeder roads from motorways leading into the city and apparently preferred by taxis. Motorways, Secondary, and Residential categorized roads are also used frequently and meet at a share of around 15%. The category Residential denotes streets in the uptown or in residential neighbourhoods.

The actual construction of a road type based energy map follows generally the road segment based counterpart with its steps: (a) retrieval of road type micro trips; (b) application of an energy model. The main difference in step (a) is the gathering of transition indices which are now defined as indices where a change in a road type occurred. Where the road segment based approach tends to result in rather small micro trips risking to omit characteristics in intersection vicinity, it is now worked on typically longer road type micro trips (for comparison see Figure 6.9), covering a wider area of dynamics. This is more robust and results in a lower variance among extracted characteristics. The energy model application in step (b) is identical to the previously described process and hence not recalled again.

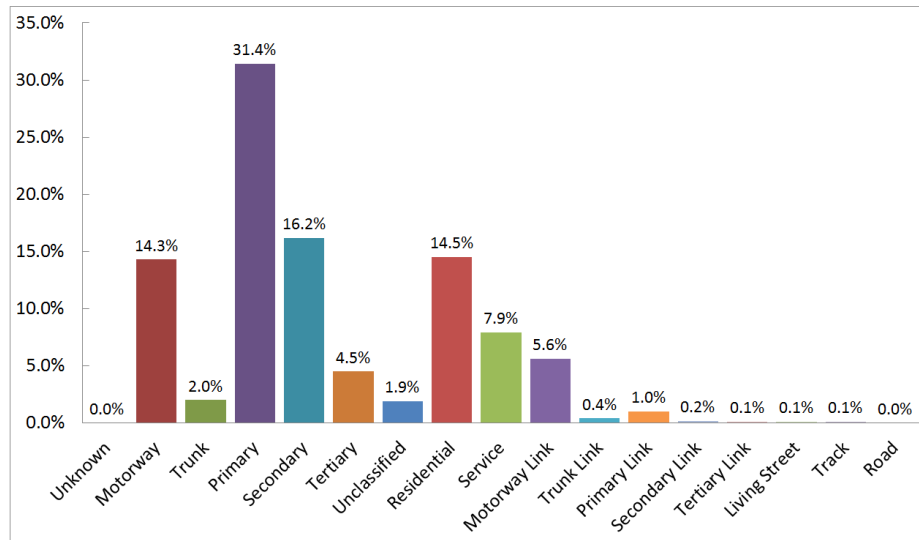


FIGURE 6.8: Road type distribution for all captured trips

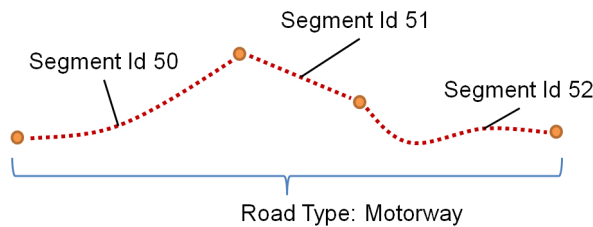


FIGURE 6.9: Road type micro trip length vs. road segment micro trip length

6.2.2 Energy Map Application

Once the energy map database is established, it is possible to derive energy consumptions for arbitrary trips. Those trips can originate either from real-world GPS trajectories, as already used to set up the database, or from simulated trips for a wide range of sampling rates as it will be the anticipated case for the taxi simulation. In line with the energy map construction part, in the application phase, traversed road segments initially have to be extracted and subsequently applied with on database. Figure 6.10 schematically illustrates this application process.

Retrieval of traversed Road Segments

This step is identical to the one in the construction of the database. In both cases, traversed road segments have to be obtained. For this, a good map-matching is required where the majority of points has to be linked correctly as this is the only information for matching the sample points with the underlying road network. In dependence of the sampling rate of input GPS trajectories, either the introduced map-matching tools are applied, or a routing algorithms for estimating the most likely taken route through the road network are utilized. Conceivable is also a combination of both. The ultimate target is to project the input GPS trajectory as good as possible to the road network.

Figure 6.11 compares routes with different estimation sources: The red coloured trajectory displays the trajectory as recorded from the GPS logger (one second sampling period). A high correspondence can be observed for the yellow, normal resolution (60 seconds sampling period)

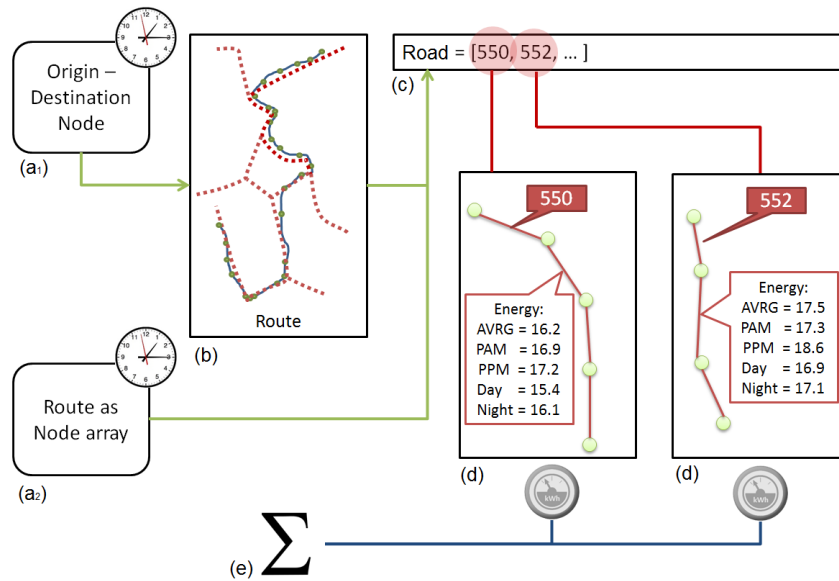


FIGURE 6.10: Schematic overview of energy map application with steps: Retrieval of traversed nodes from origin/destination node (a_1) or directly obtained from real-world trips (a_2); (b) Route reconstruction; (c) Retrieval of traversed road segments; (d) Retrieval of corresponding energy values; (e) Summation of energy values

map-matched route, which essentially follows the original one with just little deviations. The green coloured trajectory shows the estimated route when passing origin and destination positions into a routing algorithm, using a connectivity matrix of Euclidean distance (see Section 3.2.1). This, however, does not take any information despite the first and last sample point into account and results in a different route.

Although considered as a special case where the taxi did not follow the shortest route, it was frequently encountered that these circumstances do occur - often accompanied by an erroneous trip extraction. Moreover, prevailing traffic conditions are not taken into consideration, leading to the fact that the originally chosen route indeed might be the fastest due to avoidance of congested roads. This should be considered when using the energy map solely based on origin and destination locations.

Estimation of Energy Demand

The final step towards energy demand estimation is the actual application of the energy map on arbitrary trips. The extracted route of a trip, including all traversed road segment ids, is already sufficient for this purpose. Here lays the big distinction between construction and application: was the speed value of essence in the former, is the location core in the latter.

For every entailed road segment id a set of energy consumptions is retrieved from the database of all ever conducted micro trips on that specific road. This takes place subject to availability and in accordance to the corresponding time zone. This step is repeated for all other road segment ids.

Since a set of single energy consumptions is not of interest but rather is the average of all historical trips, the mean is computed on-the-fly. By doing so, the energy map is urged to grow simultaneously to its application and allow more values to be added when new data is recorded.

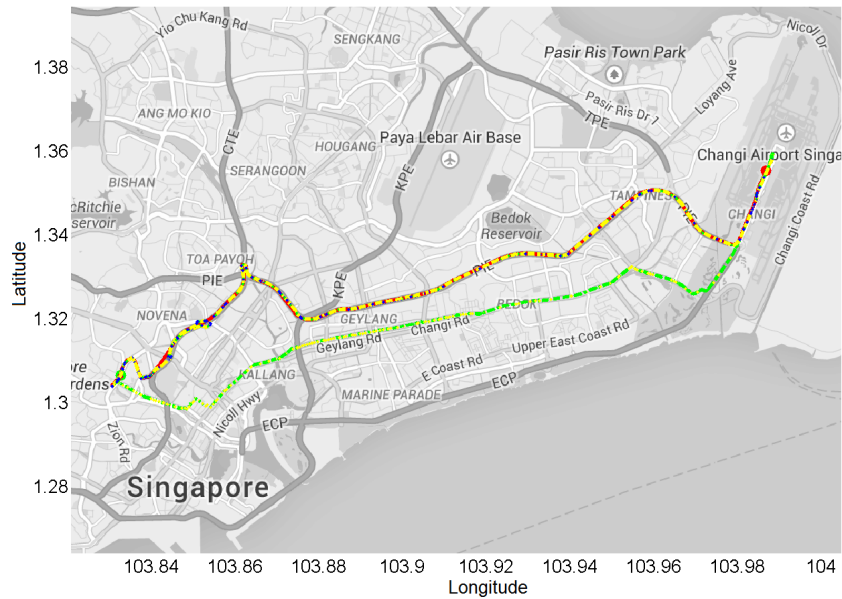


FIGURE 6.11: Comparison between trip, map-matched route and origin/destination route, background image taken from [30]

An alternative approach would be to build the mean consumptions in a preceding step and store its results.

In a final step, all received energy consumptions per road segment are summed up after re-scaling the stored values in relation to the road segment's length. Eventually, the actual consumed energy in the unit of kWh is gained. Missing values for non traversed road segments are substituted with average road type values as obtainable from the road type database.

6.3 Dynamic Driving Share Approach

One downside of the proposed energy map approach in Section 6.2 is its static design. Once, energy values are computed on basis of a certain set of vehicle characteristics and by utilizing an underlying energy model, values are fixed. Moreover, by solely focusing on the mean values, particularities in trips are omitted, which makes this approach merely feasible for a good estimate under normal conditions

The herein suggested more dynamic approach tries to mitigate those drawbacks by focusing on patterns in trips' trajectories, independent of an underlying vehicle type and capable of projecting particularities in traffic conditions. It shall be acknowledged that recorded GPS trajectories indeed are already influenced by vehicle specific parameters since its dynamic behaviour is directly mirrored to the trips. As discussed earlier, it is nevertheless assumed feasible to transfer recorded trips to various vehicle types including the one of an electric taxi.

6.3.1 Driving Share Approach Basics

This section describes the foundations of the driving share approach as adopted from [26]. The principal idea and concept behind this approach is twofold: First, it is assumed that trips are distinguishable into four distinct driving phases: *idling*, *cruising*, *accelerating* and *decelerating*;

each of them addresses a certain combination of power submodules to overcome various applicable driving forces. By including vehicle characteristics and trip duration, power requirements eventually translate into energy demands, which are of particular interest. Second, the driving phases are dependent on speed and moreover unique for different roads.

Driving Share Definition

Driving phases principally express in which dynamic mode or state the vehicle is operated, whether the driver changes speed (accelerating or decelerating), stops the car (idle), or remains at approximately the same speed (cruise). For instance, a trip snippet with a very low acceleration or deceleration, as prevalent on motorways with relatively constant speed, behaves completely different to a snippet with high acceleration or deceleration under a Stop & Go conditions on urban roads.

To reasonably distinguish driving phases, boundary thresholds are defined tolerating small outliers and therefore broaden the driving phase definitions. Since a vehicle is naturally so dynamic constituted that it is either accelerating or decelerating throughout the time, it is required to introduce mentioned thresholds. The four characteristic driving phases are introduced with mentioned thresholds as follows.

Idle The vehicle is not moving, denoted by a speed value below a threshold of 1 km/h. Generic traffic situation are, for instance, waiting in front of a red traffic light, traffic jams, or - particularly in case of taxis - waiting periods at taxi stands.

Cruise The vehicle is driving at constant speed defined by a value between an upper bounded acceleration of 0.15 m/s^2 and a lower bounded deceleration of -0.15 m/s^2 . Examples of traffic situations are long trips on uncongested motorways or trips under smooth high street traffic conditions with constant speed.

Acceleration The vehicle accelerates with a higher value as 0.15 m/s^2 . Suggested traffic situations are starting after a red traffic light or increasing speed for a takeover manoeuvre.

Deceleration The vehicle decelerates with a lower value as -0.15 m/s^2 . Possible traffic situations are stopping for a red traffic light or decreasing speed due to congestion.

The proposed approach focuses on the distribution of those driving phases for a certain trip. In accordance with the paper in [26], which introduces the discussed concept, this distribution is called *driving share*. For instance, a ride on a motorway would, due to its typically smooth traffic flow conditions, have a higher share of cruising time than accelerating or decelerating phases and likely zero idling share. A converse example of a ride on a congested road in an urban area during peak hours, however, would probably reveal a high share of accelerating, decelerating and foremost idling time, while cruising share would be negligible low.

Underlying Energy Model

The total power requirement to overcome driving forces is denoted in 6.1 as a summation of all partial power submodules. Since this work supports an electrical vehicle with recuperation capabilities, $P_{decelerating}$ can be negative which decreases the required power or even allows for recharging the battery.

$$P_{battery} = P_{roll} + P_{air} + P_{accelerating} + P_{decelerating} + P_{grade} + P_{aux} \quad (6.1)$$

Each power contributor is computed using following equations as adopted from [26, p. 259].

$$P_{roll} = m \cdot g \cdot F_r \cdot \bar{v} \quad (6.2)$$

$$P_{air} = \frac{1}{2} \cdot c_w \cdot \rho_{air} \cdot A_{front} \cdot \sqrt{v^3} \quad (6.3)$$

$$P_{accelerating} = m \cdot \bar{a} \cdot \bar{v} \quad (6.4)$$

$$P_{decelerating} = -m \cdot \bar{a} \cdot \bar{v} \quad (6.5)$$

$$P_{grade} = m \cdot g \cdot \alpha_{gradient} \cdot \bar{v} \quad (6.6)$$

$$P_{auxiliary} = const. \quad (6.7)$$

P_{roll}	...	Needed power to overcome rolling resistance [kW]
P_{air}	...	Needed power to overcome air resistance [kW]
$P_{accelerating}$...	Needed power for acceleration [kW]
$P_{decelerating}$...	Gained power from decelerating [kW]
$P_{gradient}$...	Needed power to overcome road gradient [kW]
$P_{auxiliary}$...	Needed power for auxiliary power demand [kW]
\bar{v}	...	Vehicle's average speed [m/s]
\bar{v}^3	...	Vehicle's average cubic speed [m ³ /s ³]
\bar{a}	...	Vehicle's average acceleration and deceleration, respectively [m/s ²]
$\alpha_{gradient}$...	Road gradient [%]
$m, g, F_r, c_w, \rho_{air}, A_{front}$...	Vehicle characteristics (see Table 6.1)

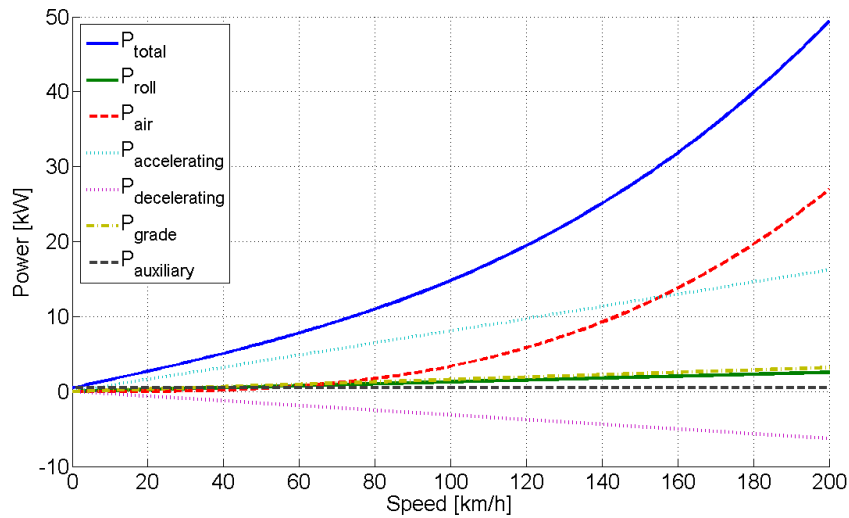


FIGURE 6.12: Impact of driving power submodules subject to speed with a constant acceleration of 0.5 m/s^2 and 1% road gradient, as suggested in [26, p. 260]

As evident from Figure 6.12, P_{roll} and P_{grade} are almost independent from vehicle's speed, whereas the linearly rising $P_{accelerating}$ and $P_{decelerating}$, as well as particular the cubical rising P_{air} have a higher impact at higher speeds. It seems that engine power up to a speed of 100 km/h is mainly spent to overcome acceleration resistance, while a value beyond this threshold predominantly requires engine power to overcome air and acceleration resistance. Especially the cubical dependency should be handled with great care since the driving share approach merely takes the

mean speed as an input for the model equations. To mitigate this possible weakness, a second average speed value \bar{v}^3 is introduced where the cube is calculated prior to extracting the mean.

In the following, all energy consumption contributors subject to power, time, and driving shares are delineated and its equation given. The symbol τ hereby expresses the normalized share of time spend in that particular driving phase.

E_{idling} The car is not moving and hence does not consume any motion related energy. Only the auxiliary devices contribute to its overall power demand.

$$E_{\text{idling}} = P_{\text{auxiliary}} \cdot t \cdot \tau_{\text{idling}} \quad (6.8)$$

E_{cruising} The car runs at constant speed which allows to omit acceleration and deceleration power. Merely roll, air and grade resistance pertain.

$$E_{\text{cruising}} = (P_{\text{roll}} + P_{\text{air}} + P_{\text{grade}} + P_{\text{auxiliary}}) \cdot t \cdot \tau_{\text{cruising}} \quad (6.9)$$

$E_{\text{accelerating}}$ The car raises its speed, thus energy to overcome acceleration related forces is needed. Since the car is still moving, roll, air and grade resistance additionally apply.

$$E_{\text{accelerating}} = (P_{\text{accelerating}} + P_{\text{roll}} + P_{\text{air}} + P_{\text{grade}} + P_{\text{auxiliary}}) \cdot t \cdot \tau_{\text{accelerating}} \quad (6.10)$$

$E_{\text{decelerating}}$ The car reduces its speed and besides the regular motion resistance of moving, roll, air and grade resistance, energy may be utilized by recuperation denoted by a negative value of $P_{\text{decelerating}}$.

$$E_{\text{decelerating}} = (P_{\text{decelerating}} + P_{\text{roll}} + P_{\text{air}} + P_{\text{grade}} + P_{\text{auxiliary}}) \cdot t \cdot \tau_{\text{decelerating}} \quad (6.11)$$

As can be seen in Equations 6.8 to 6.11, each driving phase related energy consumption is computed as a summation of the applicable power values for that specific driving phase and by multiplying those with the total trip time, as well as the share of time spend in this phase. The total energy consumption is finally derived as the addition of all partial energy values in Equation 6.12.

$$E_{\text{total}} = E_{\text{idling}} + E_{\text{cruising}} + E_{\text{accelerating}} + E_{\text{decelerating}} \quad (6.12)$$

6.3.2 Driving Shares for Energy Estimation

As of [26], the fundamental concept to link driving shares with energy consumption is the dependency on speed. It is examined that driving phase distributions change subject to speed and differ for various road types or even reveal diverse characteristics on the level of road segments. This is where the established approach comes into play. The principal idea is to establish a solely speed depended driving share database to compute the energy consumption for arbitrary trips.

In addition to the driving shares, average acceleration and deceleration values subject to speed enrich the method and can directly be passed to the equations of the underlying vehicle model. In the following, those speed dependent average acceleration and deceleration values are named *dynamics*.

The driving share and dynamics distribution is intrinsically depended on various parameters, such as traffic conditions, weather, and driving style - just to name a few. However, aiming at an easy but reasonable implementation, both are kept solely depended on vehicle's speed. Although hereby likely neglecting a number of input factors, this dependency is assumed satisfactory feasible for mirroring the driving situation effectively and sufficiently.

An upcoming restriction might be that the method is only applicable when drawing from short trips because averaged driving shares and dynamics are only reasonable if there exists a high delimitation in driving conditions. One option is to use traversed road segments or traversed road type micro trips, as they are by default short as described in Section 6.3.1. Alike the energy map construction, the objective is to construct a driving share and dynamics database for each road segment and each road type, enhanced by further differentiation into time zones.

Extraction of Driving Share and Dynamics

For the construction of the driving share database, road segment and road type micro trips are utilized, which are extracted as described in Section 6.2.1. As therein likewise introduced, the database is enhanced by treating micro trips according to their time zone, thus distinguishing driving shares temporally. Each of those retrieved micro trips is processed by an algorithm which extracts seven characteristics: driving shares (distribution of the four driving phases), mean acceleration and deceleration, as well as mean speed.

For eased understanding, virtually each recorded second of driving can be classified into one distinct driving phase. Likewise, the dynamics are easily obtainable on a second-by-second basis taking the successor sample point into account.

In detail, the seized driving shares are normalized to 100% and average speed as well as average acceleration and deceleration values are computed. For the two last mentioned values, only driving periods where changes in speed do actually apply are considered, leading to an omission of standstill and cruising times, which would have adversely affected the values.

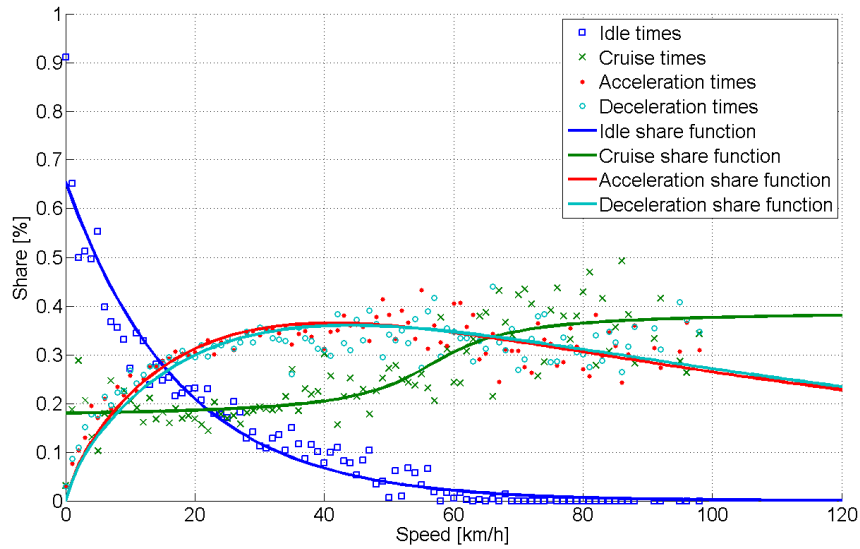
Curve Fitting

On basis of extracted driving shares and dynamics subject to speed, in each case, a curve fitting technique is performed to gather continuous functions describing these dependencies. At first, it is looked into the basic shape of each function by plotting averaged values for each available speed as illustrated in Figure 6.13 for the driving shares and in Figure 6.14 for the dynamic values. These are attached with an empirical evaluated fit function.

For curve fitting, the strength of existing built-in Matlab function *lsqcurvefit* [60] (Least Square Curve Fitting) is exploited. This technique solves nonlinear data fitting problems by minimizing the variance to a predefined function. Therefore, a basic predefined function type is requested for each distinct shape. From 6.13 three different function types can be derived.

The *Idle* share function basically shapes alike a decreasing exponential function, which is shifted to the right following a feasible function as given in Equation 6.13. This shape is suitable because for smaller average speeds the idling share has to increase and eventually intersect the y-axis at unity because in this case the vehicle was not driving at all. For higher speeds, by contrast, the idling share approaches the x-axis and becomes eventually negligible small.

The *Cruise* share function has an Arcus-Tangential form, shifted to the right and top, and is asymptotically constraint by an upper and lower boundary. It follows a possible equation as given in Equation 6.14. This shape is reasonable because a higher share of cruising for higher speeds,

FIGURE 6.13: Curve fitting for *driving shares* functions of four driving phases

$$f_{idle}(x) = a \cdot e^{-b \cdot x} \quad (6.13)$$

- a ... The y-axis intercept where e-function x-value is zero
- b ... The e-function's decreasing speed

where usually few obstacles or few traffic flow changes occur, is more likely. However, there is no big drop or rise as also for small speeds cruising time is conceivable.

$$f_{cruise}(x) = a \cdot \arctan(b \cdot x + c) + d \quad (6.14)$$

- a ... Stretching along y-axis
- b ... Stretching along x-axis
- c ... The x-axis shift
- d ... The y-axis shift

To describe the *Acceleration* and *Deceleration* functions, which both reveal the same function shape, the difference of two decreasing e-functions with variant decreasing speeds is exploited, intersecting the origin and approaching to a lower asymptote as a potential shape according to Equation 6.15. This function type is applicable because the graph has to intersect the origin for the same reason as the idling share has to intersect unity: the car is not driving at all. It increases for higher speeds reaching a peak at roughly 35 km/h, which is equal to the average inter urban speed. This is because at these speeds there is a high probability for Stop & Go traffic. It finally decreases for trips with very high average speed as on motorways with additional relatively small speed changes.

$$f_{acc/dec}(x) = a \cdot (e^{-b \cdot x} - e^{-c \cdot x}) + d \cdot e^{-1/x} \quad (6.15)$$

- a ... The y-axis intercept of resulting function
- b ... Decreasing speed for first e-function
- c ... Decreasing speed for second e-function
- d ... The lower asymptote the resulting function heads to

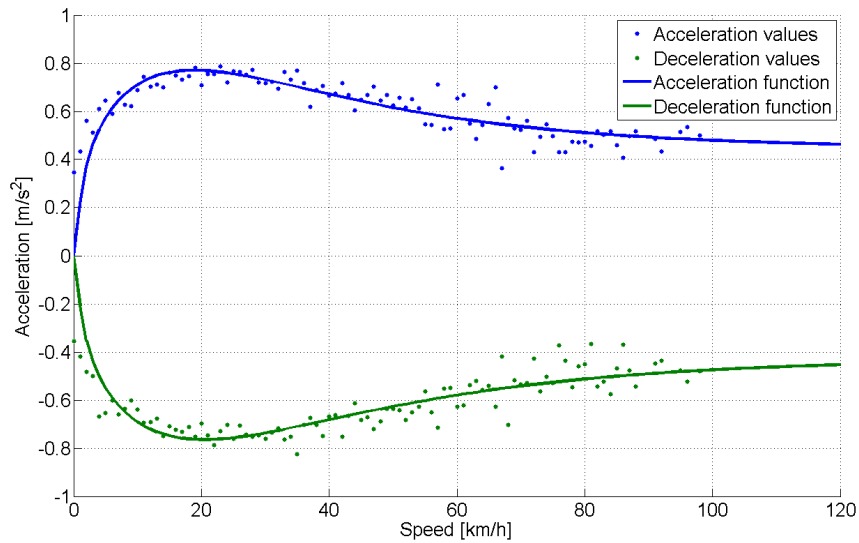


FIGURE 6.14: Curve fitting for *dynamics* functions of acceleration and deceleration

In the dynamics distribution in Figure 6.14, two different function types are gained, which are mirror-symmetric and can be consolidated into one function with contrary signs. When comparing those to the results gained from previous driving share curve fitting, a high accordance to the *Acceleration* and *Deceleration* distribution is observed, and hence the same function are used once more as given in Equation 6.15.

Parameter Extraction

By applying curve fitting, Matlab generates a set of parameters a , b , c and d (c and d only where applicable) which explicitly define the underlying functions. Therefore, the problem is translated from a huge number of fixed driving shares and dynamics to a small set of more flexible parameters for speed depended functions. In a later stage, those can be used to recover the original values, which in turn are input for the underlying energy model.

It should be noted that this approach suffers from a high fluctuation and uncertainty especially when working on short road segment or road type micro trips. For instance, a quantity of less than five points per micro trip or less than 30 total recorded micro trips per traversed segment may not reveal any characteristic pattern at all. On top of that, different driving situations are included by default, such as intersections, traffic lights, congestions, and traffic jams.

It is inevitably operated on a set of averaged values, which only reveal their full power if extracted from a huge dataset. An evidence for this fact can be retrieved from Figure 6.15, where for a specific road segment all actual input points for performed curve fitting are displayed for driving shares and dynamics.

6.3.3 Driving Share Application

The previous constructed database can be applied on arbitrary trips: firstly, by retrieving corresponding driving shares and dynamics from the database, and secondly, by computing the hereupon required energy consumption. This approach is best suitable for recorded real-world trips which reveal detailed or moderate dynamic information, as the average speed is essential for the proposed method.

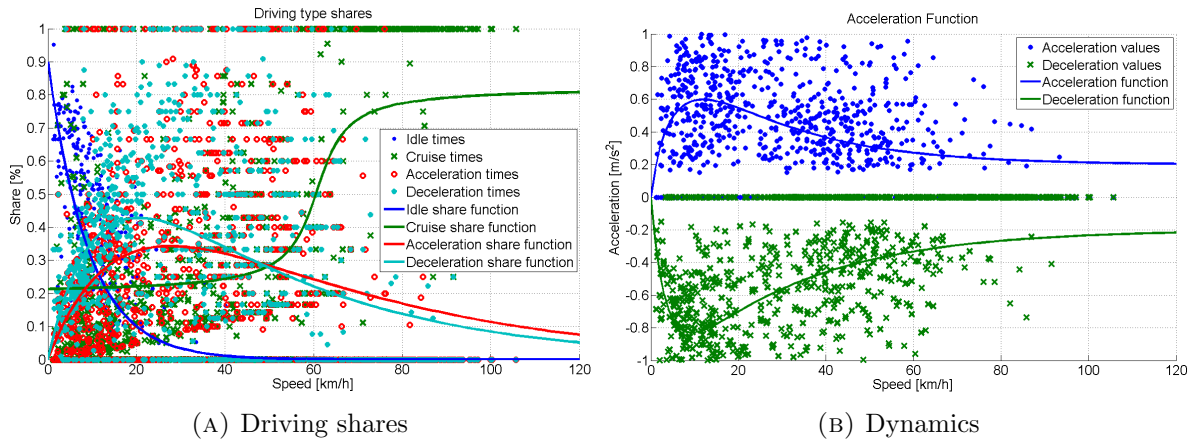


FIGURE 6.15: Curve fitting contemplating all values

Nevertheless, there are application scenarios conceivable where information about traversed road segments or road types is sufficient. In this case, dynamic information like speed and road gradient is not received from the trip itself but from previous established databases. Examples are already given in Section 3.2.2, where LTA provided speed band sensors are queried, or in Section 6.2.1, where average speed values for road types are depicted.

In Figure 6.16, a scheme about the application process is visualized, including the gathering of average speed per road segment and the retrieval of speed dependent driving share functions per road segment.

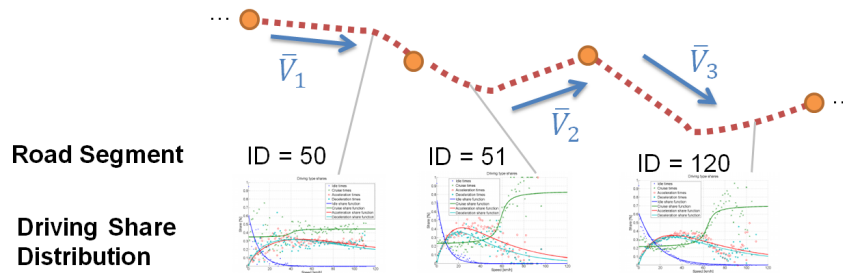


FIGURE 6.16: Driving share application scheme

Retrieval of Driving Shares and Dynamics

Based on the chosen scenario, the retrieval of driving shares is diverse. In case of real-world trips, the first step comprises the gathering of traversed road segments. Hence, a previously carried out map-matching is required. A successful map-matching tags each sample point with a corresponding road segment id which is passed as a key to the driving share database, and the herein stored driving shares and dynamics are retrieved - one set per ever carried out trip.

Since a set of driving shares and dynamics is not of major benefit, but rather are the parameters of underlying functions describing its distribution subject to time, the introduced curve fitting technique is now executed. An on-the-fly curve fitting is not suggested (the retrieval of average energy values in the energy map approach was, by contrast) because it would cost too much time. Hence this step is outsourced prior to the actual application of the method in its objected usage

context. Received function parameters are additionally stored, and curve fitting is repeated on reception of new data.

In addition to the values received from the database, the micro trip itself has to be extracted, comprising all connected sample points on that specific road segment, in order to compute the average speed (as the single input for the driving shares and dynamics functions). In case of nonexistent speed values, like simulated trips of low resolution or routed trips, it is possible to obtain dynamic values from previous mentioned data sources as a proxy. At this point, in both cases, the function parameters and the average speeds are obtained, which both can easily be handed to the actual functions from Equation 6.13, 6.14 and 6.15 for computing the resulting driving shares and dynamics.

6.4 Instant Approaches

Next to the presented static and dynamic approaches, a third category *instant* is established, which aims at instantaneously addressing the trip's dynamics distribution (without using an energy model) and is solely applicable for trips of a high sampling rate.

This instant category comprises two approaches: a driving share and a driving feature approach. The former is already known from the dynamic category where the distribution of the four driving phases and mean acceleration/deceleration is considered. The latter is a novel attempt, aiming at taking a set of 16 different features into account, as drawn from examined feature extraction in Section 4.5. Hereupon, clusters are established reflecting characteristic driving patterns, and certain energy values are linked to each. Both approaches employ recorded trips of high sampling rates and do not require any information about traversed road segments. Thus, they are the only approaches which are independent of the - to times - error prone map-matching step.

Both approaches utilize micro trips as obtained from a micro trip extraction algorithm in Section 4.5. To recall, it is aimed to divide the entire interested trip into smaller portions (the actual micro trips) with distinct characteristics, following the rules of smooth transitions among two adjacent parts. In the best case, the original trip resembles a re-assembled set of extracted micro trips and no intermediate gaps are produced. In reality, this is hardly achievable since the smooth transition rules typically enable only the inclusion of a trip's subset. In these cases, gap indices where no micro trip was extracted are located, and characteristics are interpolated by taking the two adjacent micro trip characteristics into account.

6.4.1 Instant Driving Share Approach

Extracted micro trips are directly investigated with respect to driving shares and dynamics. No previously established database is queried, but the resulting values are directly applied on the underlying model.

For each micro trip the average speed, acceleration, and deceleration is computed. Those values plus the EV characteristics are then directly passed to the equations of the driving share model introduced in Section 6.3.3.

This approach reveals a rather rough estimate about energy consumptions because important dynamic details are omitted by merely considering average values of micro trips of about 210 seconds. However, it is helpful in validating the addressed model, which differs from the one utilized in the more comprehensive EV models.

6.4.2 Instant Driving Feature Approach

In the instant driving feature approach, it is principally assumed that a set of characteristic features indicate a certain energy consumption if the features have been chosen accordingly and the hereupon established clusters as distinct among each other. Evidence for this attempt is given in [12], where the energy consumption even influenced the construction of clusters to a high degree.

To classify micro trips with respect to clusters, for each micro trip a feature vector is extracted, including the characteristics presented in the methodology section in Table 4.1. It is then attempted to find the best matching cluster denoted by the minimum Euclidean distance to the cluster centroid. Subsequently, previously computed average energy consumptions are gathered for all micro trips belonging to the matching cluster, and these steps are repeated for all remaining micro trips.

The importance of normalizing feature vectors in the clustering and classification phase should be highlighted. As a measure to identify and compute cluster's centroids, the built-in Matlab function `kmeans` and the Euclidean distance are used. The Euclidean distance is particularly sensible to vector's magnitude and therefore requires normalization. Each feature vector of the fetched micro trips is normalized by applying the same standard deviations and mean values as gathered in the clustering step.

Figure 6.17 provides an insight into the 18 eventually established clusters and its centroids for two examples with two highlighted features, each. The clusters are displayed in different colour, and a distinct distribution with marked boundaries is perceivable.

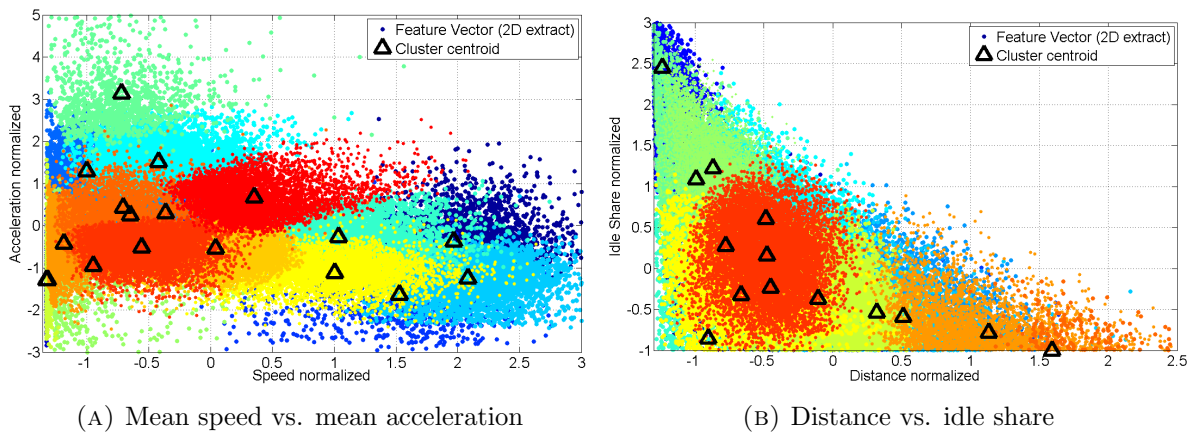


FIGURE 6.17: Extract of established 18 clusters of driving feature approach

Chapter 7

Energy Demand Analysis - Results and Evaluation

This chapter summarizes and evaluates results of the application of the utilized energy models and energy estimation approaches as introduced in the previous Chapter 6. For evaluation purposes, a representative set of total 795 trips is selected, following proceedings discussed in Section 7.2.1. To gain an insight into these trips, an extract of nine trips is displayed in Figure 7.1. The representative trips basically reflect the overall distribution in terms of status, duration, and length, and vary in its energy consumptions to derive insightful statements from evaluation.

The first section compares both energy models the TUM CREATE EV Model and the Backward Model among each other and derives consequences for its usage within herein established approaches. The second section concentrates on the evaluation of developed energy estimation approaches against a reference vehicle model for different input data sampling rates. All energy values are reported in the scaled dimension of kWh/100 km to ease comparison and enable independence from trip length.

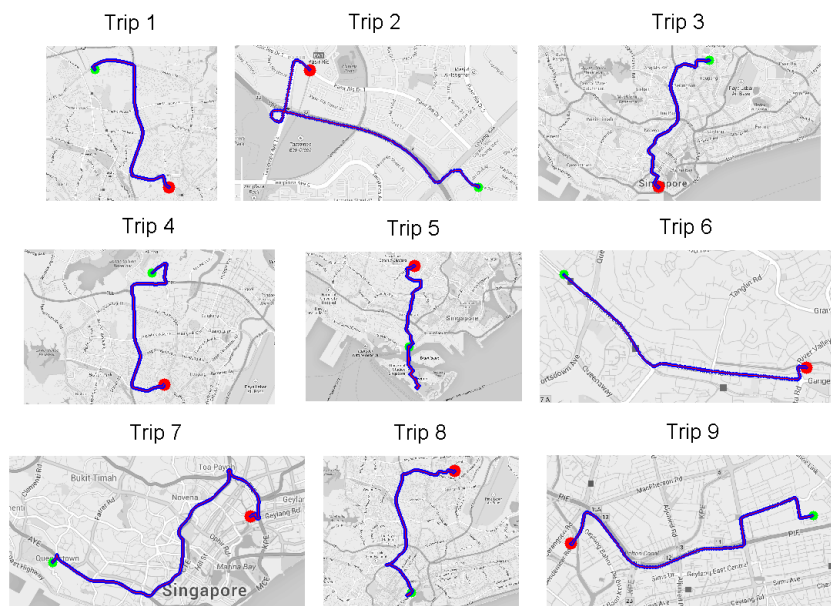


FIGURE 7.1: Extract of representative trips for validation, background images taken from [30]

7.1 Comparison of Energy Models

In this work, electrical vehicle models are utilized twofold: first, as a foundation for energy consumption calculation as part of the static energy map; second, to validate results among all energy estimation approaches. For this purpose, a confirmation and prove is needed that the incorporated models are of good quality and reliably predict energy consumptions correctly.

Since there is no access to real-world energy consumption data, such as trials on real EVs or CAN BUS retrievable energetic information as accessible for newer cars, the validation solely relies on a vehicle model as a reference - the so called ‘ground truth’. For this task, the TUM CREATE EV Model is utilized as it underwent a number of validations and was assessed as qualitatively highly valuable. It has been developed approximately since two years and preliminary used for the EVA taxi consumption estimation, and it is utilized among a number of different project in various RPs. In the following, the term *reference vehicle model* always denotes the TUM CREATE EV Model.

According to [61], the average energy consumption per 100 km for the one of the latest fully electric powered vehicle BMW i3 is 16.8 kWh (combined for city and motorway). Since this car is appreciably comparable to the characteristics of the EVA taxi prototype, most of the vehicle characteristics are taken from, it is reasonable to use this value as a benchmark and approximate estimation reference. Note that the actual vehicles attached with GPS loggers are non-electric and hence show a slightly different dynamic behaviour, particularly in terms of acceleration where the electric engine has the advantage of providing torque linearly right from the beginning - even for low rotational speeds.

The reference model has some drawbacks in terms of computation time and robustness as discussed earlier in Section 6.1. For the conceptualization of the energy map, thus, the simplified Backward Model is employed. To prove its operational capability, both models are compared on a subset of nine trips of three categories, taken from the total set of representative trips. Results are listed in Table 7.1. It is evident that deviations are very low and a high compliance can be observed throughout the clusters with deviation not exceeding 5 %.

Road Cluster	TUM CREATE EV Model	Backward Model	Deviation
Urban	17.34 kWh	16.73 kWh	-3.6 %
	17.15 kWh	16.85 kWh	-1.8 %
	18.99 kWh	18.57 kWh	-2.3 %
Road	17.03 kWh	16.71 kWh	-1.9 %
	16.11 kWh	16.75 kWh	+3.8 %
	16.54 kWh	16.03 kWh	-3.2 %
Motorway	15.99 kWh	15.83 kWh	-1.0 %
	15.02 kWh	14.74 kWh	-1.9 %
	14.45 kWh	14.84 kWh	+2.7 %

TABLE 7.1: Comparison of energy models for roadtype clusters with energy values scaled to energy values per 100 km

A further investigation result is illustrated in Figure 7.2, where all 795 trips have been applied on both models. In the left-hand chart, the Backward Model estimation is placed on the x-axis, whereas the y-axis entails the reference vehicle model calculations. It is apparent that the values in principal follow the ideal diagonal line displayed as the ‘optimum’. This is considered as ideal

because both estimation sources would meet at the same value. Another two lines representing the $\pm 5\%$ and $\pm 10\%$ deviation borders are marked. Moreover, most of the values lay within the bounds of $\pm 5\%$ deviation, revealing a high accordance. Only a little number of outliers can be seen where the reference model estimated a value which is too high. However, those are negligible.

The right-hand chart shows the trip distance comparison, which can have a biasing effect of scaled values shown in the left-hand chart. However, in this case a high compliance is observed, too.

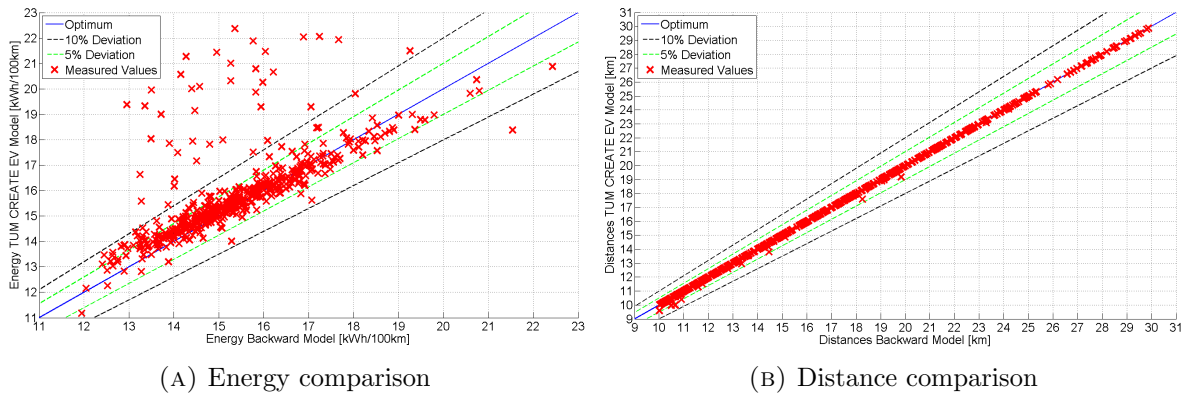


FIGURE 7.2: Comparison of energy models for all representative trips

7.2 Environment and Scope of Evaluation

One could say that a vehicle model is perfectly sufficient for the energy estimation task as it is already a great predictor and is even used as the reference ground truth. However, the actual high resolution trip's dynamics have to be available because no historical, traffic, or environmental data is leveraged.

Thus, the established energy demand estimation approaches have their right to exist; a detailed trip's trajectory with a high dynamics resolution (speed, acceleration, and altitude) cannot implicitly be expected as an input for developed programs - particularly in the context of a simulation. If so, it would be perfectly adequate to rely on the introduced vehicle models since a detailed trip with known dynamics is easily processable by those. Moreover, elaborated approaches are faster and more tailored to be used in the context of a more abstract simulation.

The evaluation's objective entails also a performance investigation of various sampling rates or periods for input trips. For this purpose, five variants are chosen: 10 s, 60 s, 180 s, 300 s, and sampling period 'zero' as solely using origin and destination of a trip. Note that for the prior database's conceptualization only high resolution trips were taken into consideration to allow for extracting as many detailed energetic characteristics about Singapore's road network as possible. The application now assumes that a database has already been established.

The approaches which link energy characteristics to the underlying road network lead to further helpful applications, which escalate the usage scenarios from just this work's case to many others. For instance, they could be used for energy economic routing or trip recommendations, especially for electric vehicles with an interest in using eco-friendly routes to save scarce energy.

Since a representative subset of taxis is chosen, which mirrors the whole taxi fleet accordingly, the taxi trajectories give a very good insight into the actual overall traffic situation as a representative. Furthermore, taxis are virtually on the road throughout the time. Taken the status

into consideration, they also reflect people’s behaviour and commuting patterns subject to time and location.

Approaches and implementations underlie a high impact of input sampling rates. The sampling rate particularly affects map-matching, but also influences the extraction of driving shares, characteristic driving features, and the quality of vehicle model’s energy computation. Figure 7.3 gives an idea about different sampling rates as possible simulation inputs. It basically affects the resolution of small dynamic changes, which are of great importance for a fair energy estimation.

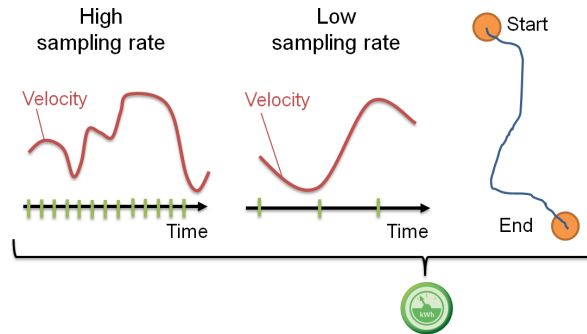


FIGURE 7.3: Overview of different sampling rates

It should once more be highlighted that there is a high impact of chosen parameters for developed programs. Most of those are derived empirically from investigations. There is not just a high dependency between energy demand estimations and chosen vehicle characteristics for the EV models; also the set of program parameters influences previous conducted processing steps. A full list can be obtained from Appendix H.

7.2.1 Evaluation Environment

The evaluation environment virtually describes what is required for the process of evaluation, which data is concerned, and which established databases are queried. The current implementation stores the road segment specific characteristics (average energy values for the energy map, and driving share and dynamics function parameters in case of driving share approach) in a Matlab hash map format. The map’s key corresponds in each case to the road segment id. With this preliminary set-up, it is possible to transfer characteristics into the established SQL database (see Appendix C), where all trips and the road network are already available. For instance, the Road Segment table in the database could easily be extended with fields denoting driving shares and energy values.

In the following sections, the choice and gathering of representative trips for all assessments is initially described, followed with a description about incorporated databases emphasizing on its utilization.

Representative Dataset for Validation

To derive meaningful statements from the following validation procedure, it is vital to work on a representative subset of all taxi trips to cover a wide range of various driving situations, vehicle types, and styles by different drivers. To allow for outliers and to enhance validity, a relatively big validation set is employed. A number of 960 trips was targeted, which are composed as follows:

For each tracked taxi ($n_{taxi} = 20$), for each time zone Peak AM, Day, Peak PM, Night ($n_{time} = 4$), and for each highway cluster Motorway, Road, Urban ($n_{cluster} = 3$), four trips ($n_{trip} = 4$) were chosen. This sums up to $n_{trips} = n_{taxi} \cdot n_{time} \cdot n_{cluster} \cdot n_{trip} = 20 \cdot 4 \cdot 3 \cdot 4 = 960$ trips. However, due to partial coverage of several road types in combination with certain time zones, only an excerpt of 795 trips could be used.

The determination of representative trips follows basically the trip length and status distribution. To derive meaningful results, which are not disturbed by factors arising on small trips (e.g., poor map-matching or few distinct driving characteristics), the focus is set on trips between 10 km and 30 km. Regarding the status, the overall distribution is alluded, and 75 % trips of status Hired and 25 % of status For Hire are taken. All other statuses were disregarded with respect to its few occurrences.

The energetic examination of representative trips, applied on the reference vehicle model, results in a mean energy demand of 15.71 kWh/100 km and is listed with additional characteristics in Table 7.2. The mean is slightly lower than the BMW i3 benchmark and possibly accounted for different EV characteristics and energy consumption measurement procedures.

TABLE 7.2: Characteristics of representative trips as used for validation, energy demand values are stated per 100 km

Characteristic	Mean	Min	Max	5%	95%
Distance	17.10 km	10.01 km	29.88 km	10.47 km	27.11 km
Energy Demand	15.71 kWh	11.59 kWh	37.98 kWh	13.27 kWh	18.50 kWh

Utilized Databases

Inputs for the following evaluation steps are the road segment characteristics (average energy, driving shares and dynamics function parameters), which are stored in Matlab hash maps. Additionally, the average energy per road type and driving shares and dynamics function parameters per road type, which are stored in common Matlab arrays, are fetched. Both source types were previously conceptualized by using all recorded trips in the period from 8th June to 9th September.

Table 7.3 lists all established approaches which are evaluated and validated in this unit. By including time zones, the differentiation between Peak AM, Day, Peak PM, and Night is taken into account and an increase in estimation quality is expected. As discussed later, this effect did not ensue because the time zones taken from taxi companies' definitions do not reflect the actual traffic situation accordingly.

TABLE 7.3: Validation categories segmented into static, dynamic, and instant approaches

Static	Energy Map	Road Segment	Mean
			Time Zone
		Road Type	Mean
			Time Zone
Dynamic	Driving Share	Road Segment	Mean
			Time Zone
		Road Type	Mean
			Time Zone
Instant	Driving Share		
	Features		

7.2.2 Application Set-Up

The evaluation is done respect to primary three emphasises: (a) the comparison of estimated energy demands of established approaches with the reference vehicle model, (b) the comparison among the estimation approaches; (c) the dependency of input trips' sampling rate and its influence on energy demand estimation quality.

The alternation of different sampling rates reflects different scenarios and mimes various input trip resolutions. There might only be the origin and destination location available on whose basis an energy demand has to be estimated; or conceivable is a trajectory with three minute sampling period as prevalent in the booking data. Also 60 seconds, as used in the map-matching step, or a high resolution of one second alike the recorded raw trajectories is a possible input for the programs in the context of a simulation. As a reminder, at low sampling rates navigation methods are used to determine traversed road segments.

The representative trips are re-sampled with the desired sampling rate, top-down by using the highest sampling rate and reducing it stepwise artificially by omitting intermediate points. In this way, lower rates are simulated as they would have been received from input data. The erected database, however, is not touched in this stage and a high resolution database is still harvested.

As already introduced in the corresponding sections, the application process of the three main approaches' categories is diverse and recapitulatory recalled as follows.

Static Approach Map-matching → Retrieval of traversed road segment list → Retrieval of corresponding time zone (→ Retrieval of all recorded energy values per road segment) → Extraction of average energy values per road segment → Summation of all partial energy consumptions

Dynamic Approach Map-matching → Retrieval of traversed road segment list → Retrieval of corresponding time zone (→ Retrieval of all recorded driving shares and dynamics per road segment) → Extraction of function parameters for driving share and dynamics functions → Application of energy model with gained driving shares and dynamics

Instant Approach Extraction of micro trips → Extraction of driving features or driving shares → Clustering of feature vectors or extraction of driving share and dynamics → Summation of all received or computed energy consumptions per micro trip

The steps in brackets denote those which are not necessary if average energy consumption (in case of energy map) and function parameters (in case of driving shares) have already been extracted in a preceding step.

For the results presented in the next sections, box plots are utilized, which are a convenient and insightful graphical representation of deviations between a set of reference and a set of estimated values. In this work's case, the reference values are energy consumptions computed by the reference vehicle model, and the estimated ones come from implemented approaches.

A box plot basically consists of four features: the mean value, two rectangles representing a certain share of values, and outliers. The mean deviation is displayed as a solid horizontal line. The range in which the best 25 % to 75 % of all values lie is illustrated with a blue rectangle, and the area which covers the best 5 % to 95 % of all values is depicted by the black vertical line. Red crosses mark outliers. A 2σ outlier threshold is aimed to be achieved; therefore the 'Whisker' is set to 0.9826 according to [62] and explained in Figure 7.4.

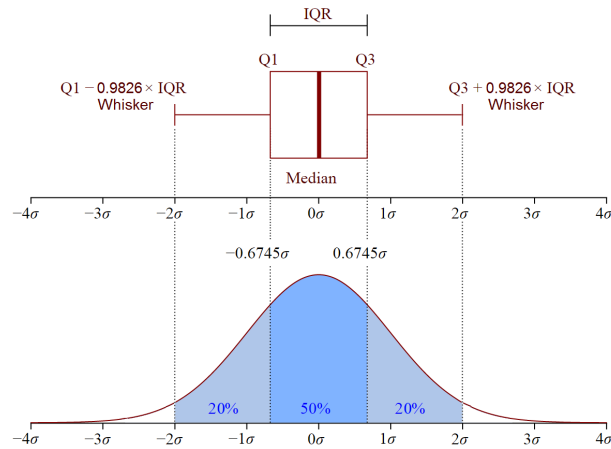


FIGURE 7.4: Box plot basics where the box denoted by *IQR* entails the best 50% values, and the red line marks values within a certain deviation boundary (in this work's case it is set to 2σ), adopted from [63]

7.3 Evaluation Results

This section presents the main results of the application of implemented energy estimation approaches and the quality of its corresponding established databases when applied on the introduced representative trip set. First, a compressed table summarising the major box plot results is presented, comparing all approaches at a glance. Second, it is continued with discussing dependencies between sampling rate and its impact of estimation quality. In a later stage, each approach is analysed in detail, presenting akin detailed box plots and attempting to explain deviations and remarkable effects.

Table 7.4 compares specific values extracted from each box plot and expresses the overall approaches' estimation quality. In each, as anticipated, the road segment based versions performs better than the road type ones due to its finer-grained nature. The inclusion of time zone consideration does not improve results to the expected extent or even worsens the outcomes. One reason might be the fact that the time zones, as chosen by the taxi companies, are mainly defined for the application of different surcharges and do not reflect the actual traffic conditions and the corresponding required energy demand accordingly. Both instant approaches perform decent, whereby the instant driving share approach almost reaches its dynamic counterpart.

TABLE 7.4: Comparison of energy approach results segmented into approaches' versions

Approach	Method	Vers.	Mean	Deviation Best 50 %	Deviation Best 90 %
Energy Map	Road Segment	Mean	-8.8 %	13.1 % _[-6.9%,+6.2 %]	37.3 % _[-19.7%,+17.6 %]
		Time	-8.7 %	13.3 % _[-6.7%,+6.6 %]	38.3 % _[-19.3%,+19.0 %]
	Road Type	Mean	+1.7 %	14.1 % _[-6.5%,+7.6 %]	41.9 % _[-20.0%,+21.9 %]
		Time	+1.4 %	14.1 % _[-6.7%,+7.4 %]	41.5 % _[-20.3%,+21.2 %]
Driving Share	Road Segment	Mean	+9.3 %	9.2 % _[-4.3%,+4.9 %]	26.8 % _[-13.0%,+14.8 %]
		Time	+8.5 %	9.4 % _[-4.9%,+4.5 %]	27.7 % _[-14.0%,+13.7 %]
	Road Type	Mean	+13.9 %	9.9 % _[-4.7%,+5.2 %]	29.1 % _[-14.2%,+14.9 %]
		Time	+13.7 %	9.9 % _[-4.9%,+5.0 %]	29.1 % _[-15.4%,+13.7 %]
Driving Share			+3.1 %	13.4 % _[-7.1%,+6.3 %]	30.3 % _[-12.3%,+18.0 %]
Driving Features			-9.7 %	14.7 % _[-6.3%,+8.4 %]	38.0 % _[-19.3%,+18.7 %]

When comparing values to those from lower or higher sampling rates, a deteriorating effect for all approaches can be observed as evident from Figure 7.5. Only the range of the best 50% trips are picked for displaying.

For the graph's construction, the trip representatives are re-map-matched with different sampling rates. The estimation quality of both dynamic approaches does indeed decrease with lower sampling rate, but they are still utilizable as results are considerably good. This is accounted for by the fact that map-matching quality is fairly constant for a wide range of sampling rates as shown before in the map-matching evaluation unit in Section 5.2. Since the extraction of road segments and hence the retrieval of road segment characteristics correlates to the map-matching, this effect was to a certain extent expected.

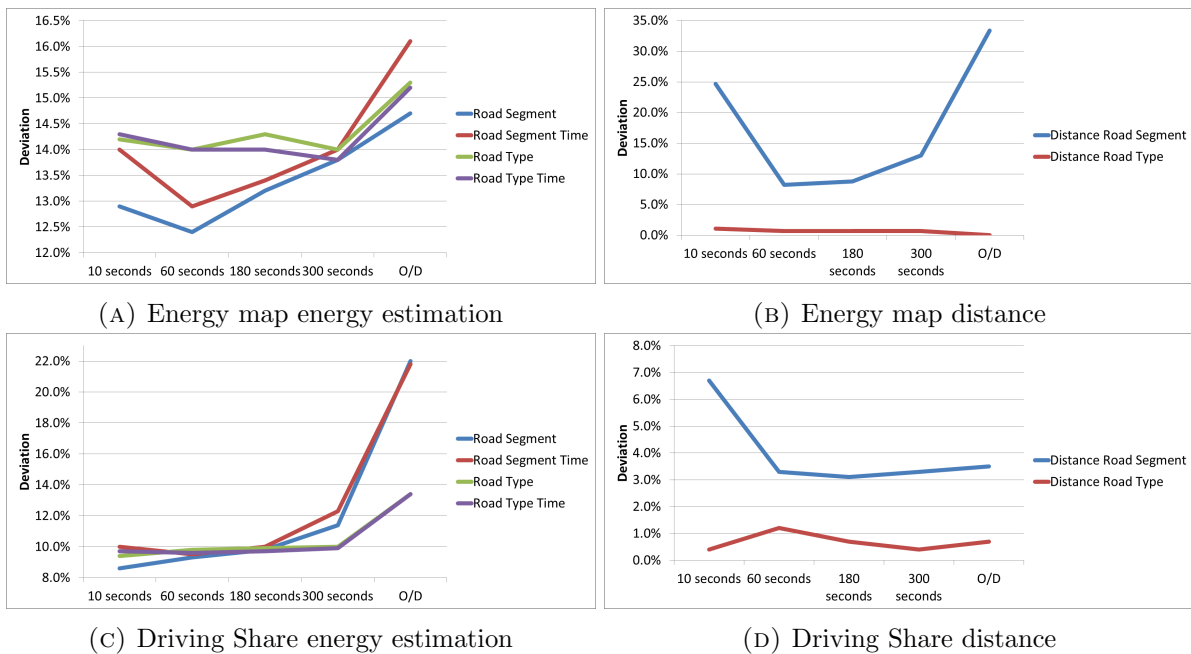


FIGURE 7.5: Energy demand and distance estimation for different sampling rates

7.3.1 Results for Energy Map Approach

Table 7.5 lists the results of the road type energy map depicting the mean energy demand per 100 km and per time zone. Regarding the mean values, several features can be observed. Energy consumption is generally lower on roads with typically higher speeds. That is, the Motorway cluster with Motorway and Trunk consumptions settle below 15 kWh/100 km, followed by the Road cluster with Primary, Secondary, and Tertiary at around 17 kWh/100 km, and reaches its peak for the Urban cluster with Unclassified, Residential, and Service at average 19 kWh/100 km. Living Street, Track, and Road are disregarded since, according to Figure 6.8, a very low occurrences of less than 0.1% is observed.

The five road types attributed with *Link* constitute connections between roads of same or different types. They are generally lower for the cluster Motorway because these links are often tagged in the transition between a road with higher speed and a road with lower speed; thus, the share of the vehicle decelerating overtops. For Links of cluster Road, by contrast, values are increasing as it is often the case that those links are mapped where a transition to a road with higher speed is present, causing a high share of acceleration.

TABLE 7.5: Energy consumption per road type and time zone in units of kWh/100 km

Mean	Peak AM	Day	Peak PM	Night	Id	Road Type
14.90	14.62	14.77	14.65	16.52	1	Motorway
14.82	15.79	14.49	14.79	15.23	2	Trunk
16.19	16.26	16.27	16.16	15.89	3	Primary
17.60	17.28	17.52	17.80	17.75	4	Secondary
17.17	17.89	17.08	17.06	16.93	5	Tertiary
17.77	16.64	17.12	18.82	20.62	6	Unclassified
18.86	18.38	18.97	19.12	18.19	7	Residential
20.36	21.03	20.41	20.39	18.87	8	Service
14.83	15.05	14.59	14.63	16.03	9	Motorway Link
13.91	14.92	14.07	13.42	13.49	10	Trunk Link
22.06	21.93	22.25	22.21	21.03	11	Primary Link
21.20	20.51	21.00	22.15	19.96	12	Secondary Link
18.63	17.37	18.50	18.78	24.04	13	Tertiary Link
14.17	12.85	13.69	15.89	13.53	14	Living Street
21.73	20.55	20.94	23.20	22.18	15	Track
-	-	-	-	-	16	Road

By immersing into the different time zones, this work's proposed approaches principally got confirmed regarding the expected outcomes. Since a high correlation between average speed and energy consumption is seen, two features should be highlighted: for the Motorway road type, average energy consumption increases during night as roads are free and taxis can drive at higher speeds. For an EV this usually means higher consumption as the impact of the cubic speed dependent air resistance rises. For the Primary road type, on the other hand, a slightly decreasing demand is observed, which is probably not due to an actual change of average speed but rather due to more uncongested traffic conditions during night, leading to more efficient driving.

In Figure 7.6 a box plot depicting the energy estimation performance respect to the benchmark of the reference vehicle model is presented. The left-hand chart is divided into four sections: both leftmost tackling the road segment based and the two rightmost tackling the road type based energy map. Each is extended with a time zone version. The right-most chart, by contrast, shows the distance deviation for both approach versions.

In principal, the more detailed road segment based version performs better than the road type counterpart. Both suffer from a biased mean, which is negative for road segments and positive for road types. Best performance is observed from the leftmost box plot with a deviation of roughly $\pm 8\%$ for the best 50% of examined trips, which surprisingly yields slightly better results than the distinguishing into time zones. This is an indication that the incorporated time zones, which are in line with the taxi companies' definition, do not satisfactorily support the estimation quality. Results in Section 4.6 already pointed to that issue; particularly drops and rises in the speed over time distribution in Figure 5.8 do not align to the time zones.

By analysing the distribution in detail as done in Figure 7.7, a problematic characteristic is observed that the values do not match over the whole range of reference model's estimated energy consumptions. Although moderate results are obtained regarding the average deviation depicted in the box plot, a good match is only considered for rather 'normal' energy consumption around 16 kWh/100 km under average conditions. The resulting line does not shape along the optimal diagonal line, but it rather resembles a parallel line to the x-axis.

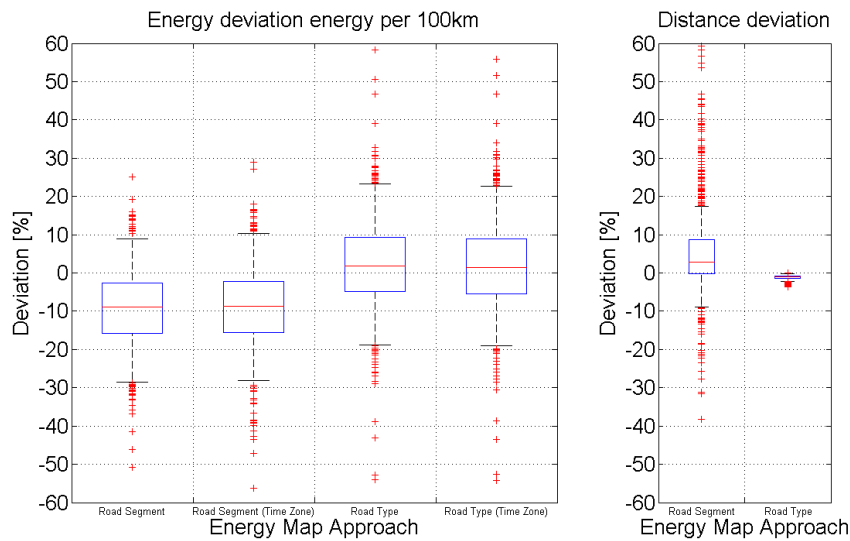


FIGURE 7.6: Energy estimation results for different versions of the energy map approach applied on a representative trip set

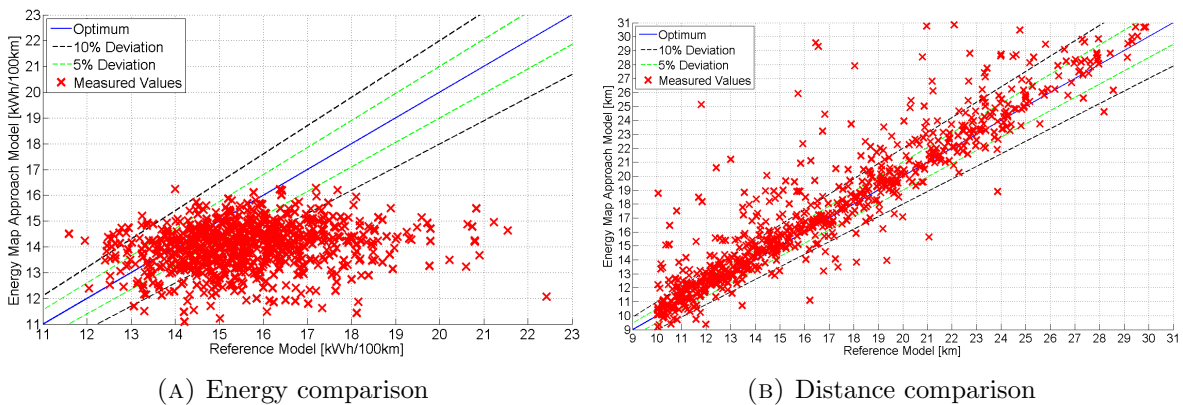


FIGURE 7.7: Evaluation of road segment based energy map

The main reason for this observation is the energy map's lack to support particular traffic situations due to its sole foundation on mean energy values. For normal trips a high fit is observed, which, however, would have also be gained by estimating the energy consumption as a constant value. Hence, no real added value is gained. As a result, a road segment based energy map tends to over-simplify particular traffic situations and rather only gives an estimate under normal conditions.

7.3.2 Results for Driving Share Approach

Shortcomings of the energy map approach are mitigated in the driving share approach. In a direct comparison to the previous discussed box plot, smaller boxes are observed and indicate a better, lower deviation in results as evidence from Figure 7.8.

In line with observations from the energy map evaluation, the road segment version of this approach outperforms the road type version due to its more tailored nature. The best result yields the road segment version without the distinguishing of time zones, reaching an estimation quality of roughly $\pm 5\%$ for the best 50% of examined trips. Counter-intuitively, time zones do

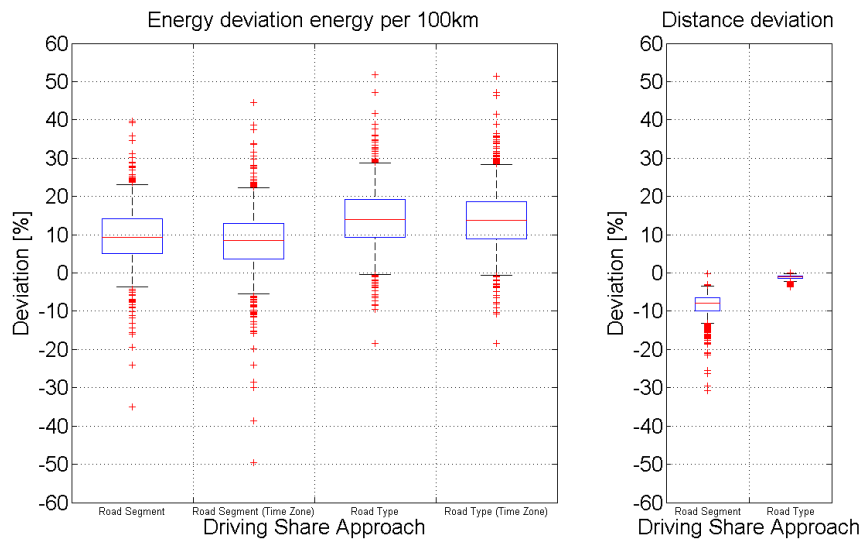
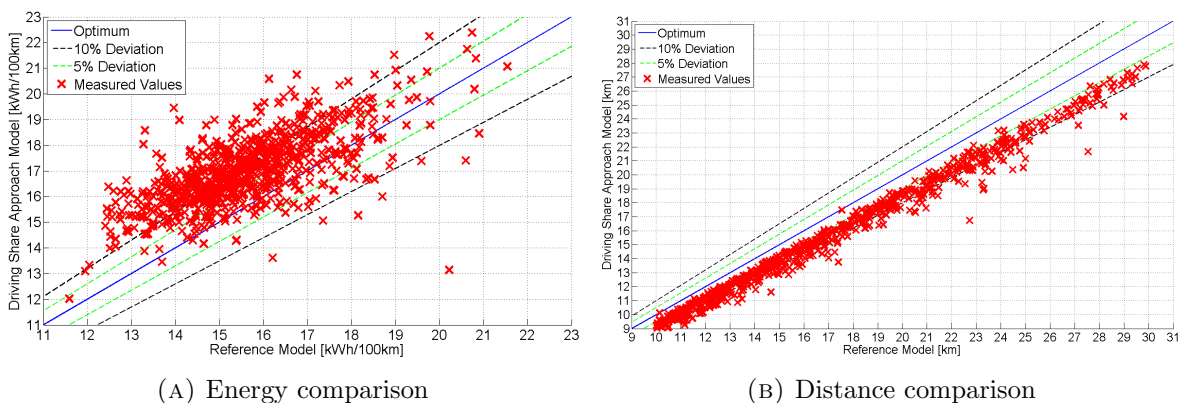


FIGURE 7.8: Energy estimation results for different versions of the driving share approaches applied on a representative trip set

not enhance quality to a great extent or even do not improve those at all, which points to the same issue of unaligned time zone separation as discussed earlier.

In this case, however, for all versions the mean value is biased and settles at a too high value at around +10% and hence does not align to the predicted energy demand. One reason is suggested that the different underlying models, the reference vehicle model on the one hand and the driving share model on the other hand, do have different sources and thus interpreting characteristics or emphasizing submodules in a different manner.

The measurement points in the chart in Figure 7.9 align very well along a line with the same slope and shape alike the ideal one, indicating a high correlation between both results sets and the substantially good prediction quality of the dynamic driving share approach. However, the same bias in the mean is reflected. As mentioned, this is likely to be caused by the underlying model and the difference to the compared reference model.



(A) Energy comparison

(B) Distance comparison

FIGURE 7.9: Evaluation of road segment based driving share approach

The driving share approach's foundation is determined by differences in driving shares and dynamics distributions for different road types and road segments. In the conceptualization

phase, function parameters describing dependencies between speed and mentioned distributions are extracted for each road type and each road segment.

The charts in Figure 7.10 illustrate distributions for three different road types (Motorway, Primary, and Service) with distinct function shapes. As expected, driving shares on roads of category Motorway are predominantly determined by cruising time, whereas idling time is barely present. The dynamics distribution reflect this issue similarly with relatively low acceleration and deceleration values throughout the speed range. Roads of category Primary are defined by considerably high acceleration and deceleration values and shares. Also Service roads follow this pattern but decline more sudden for higher speeds.

Overall, the charts reveal a fair level of distinctness confirming the basic driving share concept and is even more diverse when considering single road segments instead of road types.

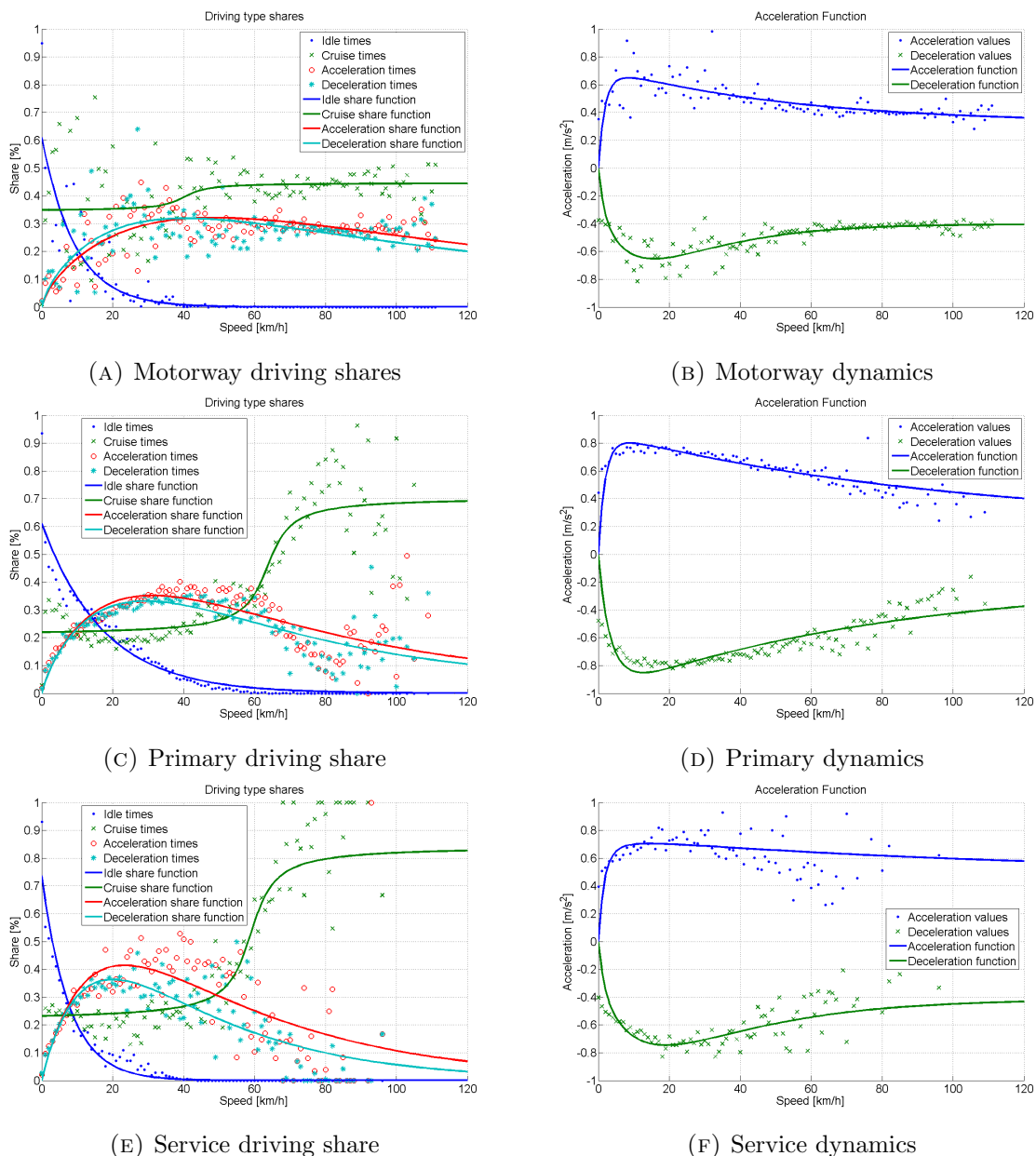


FIGURE 7.10: Driving share and dynamics per road type for *Motorway*, *Primary* and *Service*

7.3.3 Results for Instant Approaches

The instant approaches do not claim to be the main outcome of this work, but they rather gave valuable feedback during development of the driving share approach with its unproven underlying model and are a good starting point for possible further investigations into driving features as they incorporate more dynamic characteristics than the driving share.

Results of both application variations are plotted in Figure 7.11 and 7.12. Former evaluates the instant driving share approach and latter the driving feature approach.

In Figure 7.11, the box plot in the left-hand chart and the comparison plot in the right-hand chart both indicate an admirably good estimation quality. Prior to application, the trip was cut into micro trips of which driving shares and dynamics were derived. Those are then directly handed to the underlying energy model equations, calculating the demander energy. It turns out that the extracted averaged characteristics are already sufficient to estimate energy consumption of a moderate quality without requiring a preceding map-matching step.

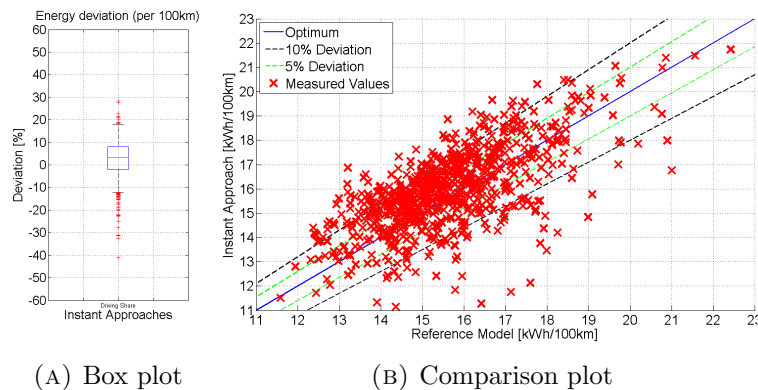


FIGURE 7.11: Instant driving share approach evaluation

The latter plots in Figure 7.12 display evaluation results of the driving feature approach, which aimed to extract feature vectors from micro trips and to classify those according to 18 established clusters with distinct energy consumptions. This approach seems to underestimate the variety of driving characteristics and reveals a rather big deviation. It basically follows the optimal diagonal trend line as seen from the right-hand chart, but it does not convince with results. For future work it is suggested to extent the amount of clusters or to apply weights to the vector to control each feature's impact within a feature vector.

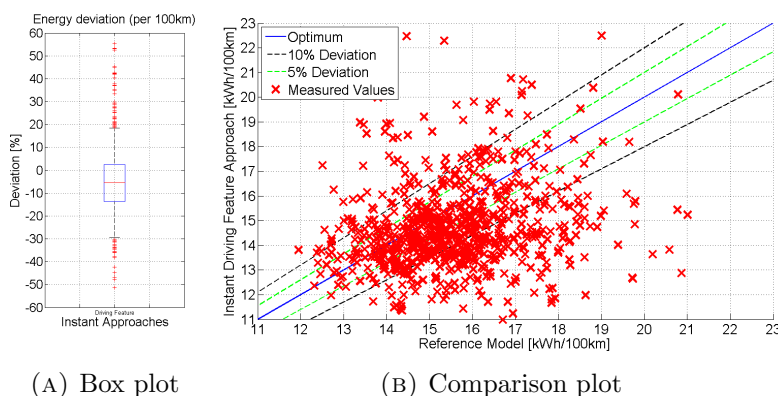


FIGURE 7.12: Instant driving feature approach evaluation

Chapter 8

Conclusion and Outlook

This thesis focused on two parts: driving profile and energy demand analysis. In this chapter, both parts are concluded with focus on its utilization opportunities and implications for further work. In the objective stated in the introduction, this thesis' ultimate goal was the conceptualization of reliable energy demand estimation programs for electrical vehicles for different sampling rates of trips. The integration into the overarching project context was defined as the role of a supplier for a comprehensive taxi simulation in tropical mega-cities.

As the main outcome of the driving profile analysis unit, a database entailing all recorded trips was established. It not just contains the raw data as directly gathered from the GPS loggers, but entails trips which already underwent various processing steps like: filtering, interpolation, trip extraction, shift extraction, map-matching, and zone matching.

The trip's quality is appropriate for the subsequently conducted energy demand analysis, and statistical evaluations show high compliance with LTA provided data and empirically derived expectations. Furthermore, for future work, the database is capable of serving multiple purposes and of supporting several projects in the context of electro-mobility in Singapore - perhaps as part of future projects of TUM CREATE.

Especially the fact that extracted trips' each sample points are linked with a sophisticatedly estimated road segment and implicitly with a road type, both part of the underlying OSM road network, makes it valuable for analysing traffic patterns directly projected to environmental conditions in Singapore. Since Singapore's entire road network, its zones, and auxiliary data such as taxi stands are available in the database, this work established a data compound, conveniently accessible from just one single source. This might be a peculiarity of present work as a leveraged emphasise of various data sources make data mining more reliable.

The cooperation with SMRT has been solidified and is about to be extended, likely aided due to satisfying statistical evaluations from this thesis. SMRT is seen as a strategic partner for upcoming projects in the field of electro-mobility or urban transportation in general, which are already in draft.

Statistics turned out to match appreciatively with various LTA data sources, indicating a good logging and booking data quality, as well as a plausible implementation of algorithms incarnated in developed Matlab programs. It also reveals a fit in chosen program parameters, especially for the more empirically based procedures like trip and shift extraction. Also the driving profile preparation, the decent map-matching, enhanced filtering, and re-assignment of correct altitude values from the Google Elevation API contributed to a high quality of uploaded trips.

As a summary of energy demand analysis' results, the outcome of the driving share approach

shall be highlighted, meeting the energy consumption computed by the reference vehicle model to a great extent and shows high compliance. Driving shares do not just express the average energy consumptions on road segments as the static energy map does, but it also take dynamics in dependency of the average speed per road segment into account and in this way, projects particular traffic conditions way better. The driving shares have previously been derived from a huge set of trips. It provides opportunities for applications for various sampling rates as long as the average speed can be determined in the targeted granularity. It is furthermore extendable with upcoming recordings and adjustable to different vehicle types.

Areas for potential have been uncovered during implementations as outlined in the regarded chapters and summarized in the following. The concept of road segment based analysis was initially grounded on the energy map approach, which was expected to perform best due to its anticipated well fitting and tailored design to the energy models. However, a rather moderate deviation in energy values was observed, underperforming the driving share approach. New data from GPS trajectories are about to be incorporated into the current database which will stepwise enhance energy estimation quality.

An additional challenge for further work is the examined mismatch of employed time zones and actual traffic patterns subject to time, which did not reveal the anticipated enhancement in energy estimation quality. Results of all energy estimation approaches shall furthermore be uploaded into the established database for a uniform data access. Moreover, an implementation of auxiliary submodules for employed EV models are conceivable in future improvements.

As a final summary, in the driving profile analysis a database of in-detail trips enhanced by several processing steps and multiple data sources is established, which has great potential and is considered useful also for related projects. The analysis revealed insightful statistics enhancing developed programs and is of great value. On the energy demand analysis side, results show that a utilization of energy approaches in a simulation context is feasible due to a good estimation quality. This applies particularly for the driving share approach with less than $\pm 14\%$ deviation for the best 90% trips. Moreover, this approach enables good expandability, usability, and adaptability since it is independent from a specific vehicle and specific traffic conditions. All approaches are expected to become more accurate with additionally recorded data.

Appendix A

Logging Data Example

TABLE A.1: Logging data example

Id	Tag	Date	Time	Latitude	Longitude	Height	Speed	Heading
2000	T	140901	24959	01.285260N	103.847203E	52	40	120
2001	T	140901	25000	01.285208N	103.847291E	52	42	122
2002	T	140901	25001	01.285260N	103.847563E	52	37	124
2003	T	140901	25002	01.285216N	103.847683E	53	29	130
2004	T	140901	25003	01.285163N	103.847735E	53	39	124
2005	T	140901	25004	01.285041N	103.848093E	53	40	121
2006	T	140901	25005	01.284928N	103.848243E	53	40	119
2007	T	140901	25006	01.284869N	103.848346E	54	37	118
2008	T	140901	25007	01.284768N	103.848271E	54	40	118
2009	T	140901	25008	01.284704N	103.848308E	54	39	118
2010	T	140901	25009	01.284646N	103.848395E	54	45	117
2011	T	140901	25010	01.284583N	103.848496E	55	43	117
2012	T	140901	25011	01.284513N	103.848621E	55	47	118
2013	T	140901	25012	01.284466N	103.848718E	56	47	118
2014	T	140901	25013	01.284416N	103.848828E	57	45	116
2015	T	140901	25014	01.284374N	103.848928E	58	41	115
2016	T	140901	25015	01.284309N	103.849060E	59	51	117
2017	T	140901	25016	01.284293N	103.849184E	60	49	123
2018	T	140901	25017	01.284160N	103.849263E	61	44	129
2019	T	140901	25018	01.284050N	103.849361E	62	50	154
2020	T	140901	25019	01.283960N	103.849424E	63	45	148
2021	T	140901	25020	01.283865N	103.849504E	64	50	144
2022	T	140901	25021	01.283766N	103.849594E	65	48	144
2023	T	140901	25022	01.283661N	103.849668E	65	50	145
2024	T	140901	25023	01.283546N	103.849729E	66	51	149
2025	T	140901	25024	01.283426N	103.849793E	67	49	151

Appendix B

Booking Data Example

TABLE B.1: Booking data example

Taxi License	Latitude	Longitude	Status	Date and Time
SHF323S	1.362683	103.7447	FOR HIRE	9/8/2014 12:08
SHC4866G	1.335307	103.8628	BUSY	9/8/2014 12:08
SHB5902J	1.314898	103.9095	HIRED	9/8/2014 12:08
SHB1013B	1.36554	103.8907	HIRED	9/8/2014 12:08
SHF52Z	1.27775	103.8087	FOR HIRE	9/8/2014 12:08
SHB167Y	1.304678	103.9215	HIRED	9/8/2014 12:08
SHB164E	1.320099	103.8528	PAYMENT	9/8/2014 12:08
SHB5566X	1.372853	103.9317	HIRED	9/8/2014 12:08
SHB1133M	1.373368	103.8822	FOR HIRE	9/8/2014 12:08
SHF290C	1.318837	103.8626	FOR HIRE	9/8/2014 12:09
SHB1078P	1.349168	103.7097	FOR HIRE	9/8/2014 12:11
SHB1188E	1.304898	103.764	PAYMENT	9/8/2014 12:11
SHB5121X	1.351087	103.7319	BUSY	9/8/2014 12:11
SHB1210Z	1.294433	103.7861	HIRED	9/8/2014 12:11
SHD6282Y	1.306355	103.8493	HIRED	9/8/2014 12:11
SHF61Y	1.34079	103.8475	FOR HIRE	9/8/2014 12:11
SHB1269E	1.346933	103.9602	LOG OFF	9/8/2014 12:11
SHC4809Y	1.437782	103.8376	LOG OFF	9/8/2014 12:11
SHB1238U	1.35954	103.8829	HIRED	9/8/2014 12:11
SHF323S	1.362677	103.7447	FOR HIRE	9/8/2014 12:11
SHC4866G	1.335293	103.863	BUSY	9/8/2014 12:11
SHB5902J	1.308844	103.9117	HIRED	9/8/2014 12:11
SHB1013B	1.369155	103.8944	HIRED	9/8/2014 12:11
SHF52Z	1.272677	103.8115	FOR HIRE	9/8/2014 12:11
SHB167Y	1.317193	103.9636	HIRED	9/8/2014 12:11

Appendix C

Taxi Database Structure

Shift	Trip	DataPoint	StandstillPeriod	Vehicle	VehicleType	TaxiType	Status
idShift	idTrip	idDataPoint	idStandstillPeriod	idVehicle	idVehicleType	idTaxiType	idStatus
idVehicle	idStatus	idTrip	idStandstillLocation	idVehicleType	idTaxiType	name	name
	idShift	idSection	idDataPoint1	idHomeSubZone	name		
	idDataPointStart	idSubZone	idDataPoint2	licensePlate			
	idDataPointEnd	datetime	duration[h]	hirers			
	distance [km]	location					
	revenue [SGD]	altitude [m]					
		speed [km/h]					
Section	SubZone	BorderSubZone	PlanningArea	BorderPlanningArea	Region	BorderRegion	BorderPoint
idSection	idSubZone	idBorderSubZone	idPlanningArea	idBorderPlanningArea	idRegion	idBorderRegion	idBorderPoint
idNode1	idPlanningArea	idBorderPoint	idRegion	idBorderPoint	polygon	idBorderPoint	location
idNode2	polygon	idSubZone	polygon	idPlanningArea	name	idRegion	
idStreetType	name	sequence	name	sequence		sequence	
idway							
Node	StreetType	StandstillLocation	StandstillType				
idNode	idStreetType	idStandstillLocation	idStandstillType				
location	name	idStandstillType	name				
idOSM		idSubZone					
		idAuxiliary					
		location					

FIGURE C.1: Taxi database overview

Appendix D

Shift Set Structure

20x7 table

	1 shifts	2 licenseNumber	3 vehicleType	4 shiftType	5 homeAddress	6 homeRegion	7 statistics
1	59x4 table		'Chevrolet Epica'	2		'North'	19x6 table
2	14x4 table		'Toyota Prius'	1		'East'	19x6 table
3	52x4 table		'Chevrolet Epica'	2		'North'	19x6 table
4	46x4 table		'Chevrolet Epica'	1		'West'	19x6 table
5	58x4 table		'Toyota Prius'	2		'North-East'	19x6 table
6	46x4 table		'Toyota Prius'	1		'West'	19x6 table
7	63x4 table		'Chevrolet Epica'	2		'North-East'	19x6 table
8	65x4 table		'Toyota Prius'	1		'North-East'	19x6 table
9	70x4 table		'Ssangyong Rodius'	1		'West'	19x6 table
10	58x4 table		'Chevrolet Epica'	2		'North'	19x6 table
11	49x4 table		'Ssangyong Rodius'	1		'East'	19x6 table
12	52x4 table		'Chevrolet Epica'	1		'East'	19x6 table
13	51x4 table		'Chrysler 300C'	2		'North'	19x6 table
14	70x4 table		'Chevrolet Epica'	2		'North-East'	19x6 table
15	62x4 table		'Chevrolet Epica'	2		'West'	19x6 table
16	7x4 table		'Chevrolet Epica'	2		'West'	19x6 table
17	49x4 table		'Chevrolet Epica'	1		'North-East'	19x6 table
18	58x4 table		'Chevrolet Epica'	2		'North-East'	19x6 table
19	34x4 table		'Chevrolet Epica'	2		'South'	19x6 table
20	43x4 table		'Chevrolet Epica'	1		'West'	19x6 table

FIGURE D.1: Shift set structure overview

59x4 table

	1 trips	2 shiftStart	3 shiftEnd	4 duration
1	37x7 table	'01-Aug-2014 10:53:04'	'01-Aug-2014 18:09:30'	7.2739
2	17x7 table	'01-Aug-2014 21:00:42'	'02-Aug-2014 00:20:59'	3.3381
3	29x7 table	'02-Aug-2014 07:16:15'	'02-Aug-2014 17:18:37'	10.0394
4	32x7 table	'02-Aug-2014 17:39:23'	'03-Aug-2014 03:50:25'	10.1839
5	24x7 table	'03-Aug-2014 07:10:50'	'03-Aug-2014 17:57:01'	10.7697
6	52x7 table	'04-Aug-2014 07:11:51'	'04-Aug-2014 16:59:38'	9.7964
7	26x7 table	'04-Aug-2014 17:11:48'	'05-Aug-2014 00:18:30'	7.1117
8	37x7 table	'05-Aug-2014 06:56:55'	'05-Aug-2014 17:12:48'	10.2647
9	28x7 table	'05-Aug-2014 17:12:50'	'05-Aug-2014 23:43:00'	6.5028
10	44x7 table	'06-Aug-2014 06:54:31'	'06-Aug-2014 18:04:42'	11.1697
11	8x7 table	'06-Aug-2014 22:17:19'	'06-Aug-2014 23:54:56'	1.6269
12	52x7 table	'07-Aug-2014 06:44:37'	'07-Aug-2014 17:06:39'	10.3672
13	27x7 table	'07-Aug-2014 17:07:05'	'07-Aug-2014 23:58:16'	6.8531
14	64x7 table	'08-Aug-2014 06:58:20'	'08-Aug-2014 18:42:31'	11.7364
15	1x7 table	'09-Aug-2014 00:26:45'	'09-Aug-2014 00:28:43'	0.0328
16	57x7 table	'09-Aug-2014 07:07:50'	'09-Aug-2014 18:28:51'	11.3503
17	48x7 table	'10-Aug-2014 07:28:50'	'10-Aug-2014 16:50:54'	9.3678
18	33x7 table	'10-Aug-2014 17:35:35'	'11-Aug-2014 03:38:02'	10.0408
19	53x7 table	'11-Aug-2014 07:18:14'	'11-Aug-2014 18:19:59'	11.0292
20	30x7 table	'12-Aug-2014 07:04:10'	'12-Aug-2014 17:06:48'	10.0439
21	44x7 table	'12-Aug-2014 17:49:05'	'13-Aug-2014 04:29:43'	10.6772
22	36x7 table	'13-Aug-2014 07:21:01'	'13-Aug-2014 17:18:47'	9.9628
23	30x7 table	'13-Aug-2014 17:18:56'	'14-Aug-2014 00:42:56'	7.4000
24	46x7 table	'14-Aug-2014 07:02:20'	'14-Aug-2014 17:05:20'	10.0500
25	27x7 table	'14-Aug-2014 17:07:26'	'15-Aug-2014 00:23:07'	7.2614
26	42x7 table	'15-Aug-2014 06:56:50'	'15-Aug-2014 17:18:22'	10.3589
27	32x7 table	'15-Aug-2014 17:18:24'	'16-Aug-2014 00:59:50'	7.6906
28	30x7 table	'16-Aug-2014 07:02:44'	'16-Aug-2014 17:11:33'	10.1469
29	26x7 table	'16-Aug-2014 17:20:26'	'17-Aug-2014 01:20:54'	8.0078
30	39x7 table	'17-Aug-2014 08:15:24'	'17-Aug-2014 17:12:20'	8.9489

FIGURE D.2: Trip set structure overview

37x7 table

	1 trip	2 tripStart	3 tripEnd	4 duration	5 distance	6 status	7 taxistand	
1	1x1 Trip	'01-Aug-2014 10:53:04'	'01-Aug-2014 11:01:35'	8.5167	2.2458	'HIRED'	[]	
2	1x1 Trip	'01-Aug-2014 11:07:47'	'01-Aug-2014 11:37:02'	29.2500	6.4284	'HIRED'	1x2 cell	
3	1x1 Trip	'01-Aug-2014 11:37:03'	'01-Aug-2014 11:49:04'	12.0167	8.1450	'HIRED'	[]	
4	1x1 Trip	'01-Aug-2014 11:49:06'	'01-Aug-2014 11:50:41'	1.5833	0.2317	'FOR HIRE'	[]	
5	1x1 Trip	'01-Aug-2014 11:50:44'	'01-Aug-2014 12:05:04'	14.3333	8.0133	'HIRED'	[]	
6	1x1 Trip	'01-Aug-2014 12:05:06'	'01-Aug-2014 12:09:06'		4	1.1265	'FOR HIRE'	[]
7	1x1 Trip	'01-Aug-2014 12:09:08'	'01-Aug-2014 12:33:54'	24.7667	4.7749	'HIRED'	1x2 cell	
8	1x1 Trip	'01-Aug-2014 12:33:55'	'01-Aug-2014 12:55:26'	21.5167	7.9483	'HIRED'	[]	
9	1x1 Trip	'01-Aug-2014 12:55:28'	'01-Aug-2014 12:59:18'	3.8333	1.2675	'HIRED'	[]	
10	1x1 Trip	'01-Aug-2014 12:59:20'	'01-Aug-2014 13:00:00'	0.6667	0.0520	'FOR HIRE'	[]	
11	1x1 Trip	'01-Aug-2014 13:00:02'	'01-Aug-2014 13:06:18'	6.2667	2.0991	'ON CALL'	[]	
12	1x1 Trip	'01-Aug-2014 13:06:21'	'01-Aug-2014 13:31:46'	25.4167	8.4763	'HIRED'	[]	
13	1x1 Trip	'01-Aug-2014 13:31:57'	'01-Aug-2014 13:35:10'	3.2167	0.9099	'FOR HIRE'	[]	
14	1x1 Trip	'01-Aug-2014 13:35:12'	'01-Aug-2014 13:44:02'	8.8333	4.6736	'HIRED'	[]	
15	1x1 Trip	'01-Aug-2014 13:44:04'	'01-Aug-2014 13:50:03'	5.9833	2.6710	'FOR HIRE'	[]	
16	1x1 Trip	'01-Aug-2014 13:50:12'	'01-Aug-2014 14:04:14'	14.0333	6.6066	'HIRED'	[]	
17	1x1 Trip	'01-Aug-2014 14:04:15'	'01-Aug-2014 14:13:00'	8.7500	3.1127	'HIRED'	1x2 cell	
18	1x1 Trip	'01-Aug-2014 14:13:03'	'01-Aug-2014 14:15:40'	2.6167	0.1161	'FOR HIRE'	[]	
19	1x1 Trip	'01-Aug-2014 14:15:49'	'01-Aug-2014 14:29:39'	13.8333	12.8911	'HIRED'	[]	
20	1x1 Trip	'01-Aug-2014 14:29:40'	'01-Aug-2014 14:39:12'	9.5333	4.3785	'HIRED'	[]	
21	1x1 Trip	'01-Aug-2014 14:39:14'	'01-Aug-2014 14:44:00'	4.7667	1.5874	'FOR HIRE'	[]	
22	1x1 Trip	'01-Aug-2014 14:44:03'	'01-Aug-2014 15:07:13'	23.1667	14.9139	'HIRED'	[]	
23	1x1 Trip	'01-Aug-2014 15:07:20'	'01-Aug-2014 15:14:33'	7.2167	1.0559	'FOR HIRE'	[]	
24	1x1 Trip	'01-Aug-2014 15:29:28'	'01-Aug-2014 15:32:20'	2.8667	0.8230	'FOR HIRE'	[]	
25	1x1 Trip	'01-Aug-2014 15:32:22'	'01-Aug-2014 15:46:29'	14.1167	6.6286	'HIRED'	[]	
26	1x1 Trip	'01-Aug-2014 15:46:41'	'01-Aug-2014 15:50:18'	3.6167	1.4801	'FOR HIRE'	[]	
27	1x1 Trip	'01-Aug-2014 15:50:25'	'01-Aug-2014 16:00:35'	10.1667	10.1900	'HIRED'	[]	
28	1x1 Trip	'01-Aug-2014 16:00:37'	'01-Aug-2014 16:21:27'	20.8333	7.3917	'FOR HIRE'	[]	
29	1x1 Trip	'01-Aug-2014 16:21:29'	'01-Aug-2014 16:27:47'	6.3000	2.9231	'HIRED'	1x2 cell	
30	1x1 Trip	'01-Aug-2014 16:27:49'	'01-Aug-2014 16:43:58'	16.1500	9.1359	'FOR HIRE'	[]	

FIGURE D.3: Trip structure overview

Appendix E

Additional Statistics

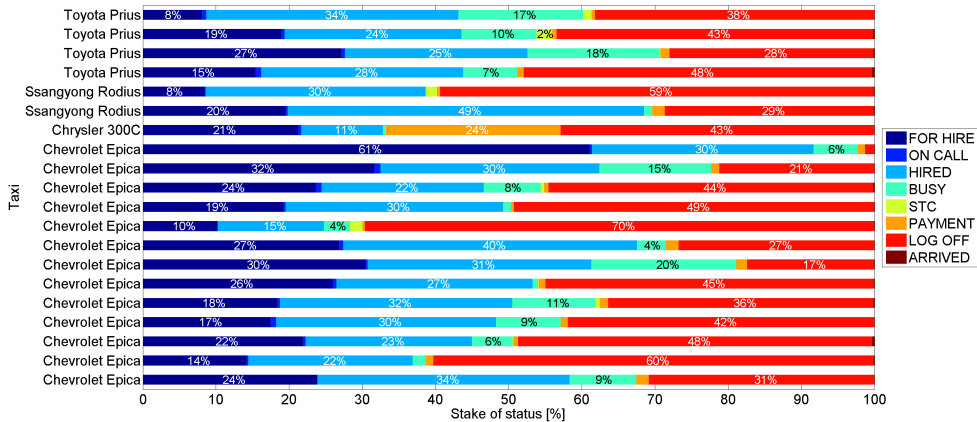


FIGURE E.1: Average status distribution for all recorded 20 taxis

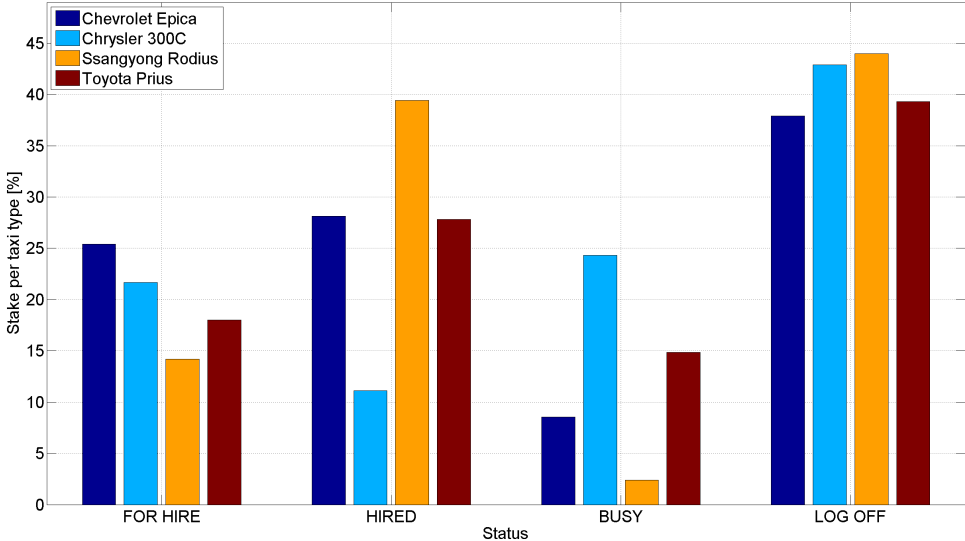
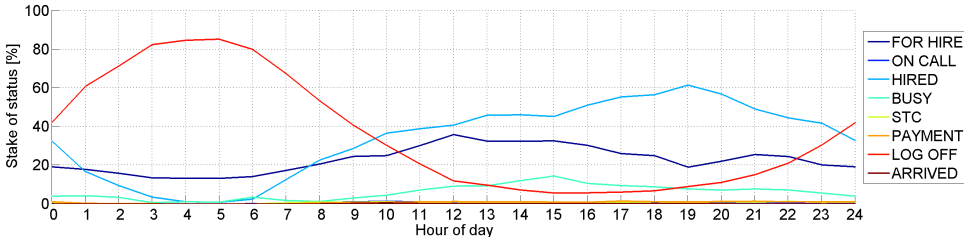
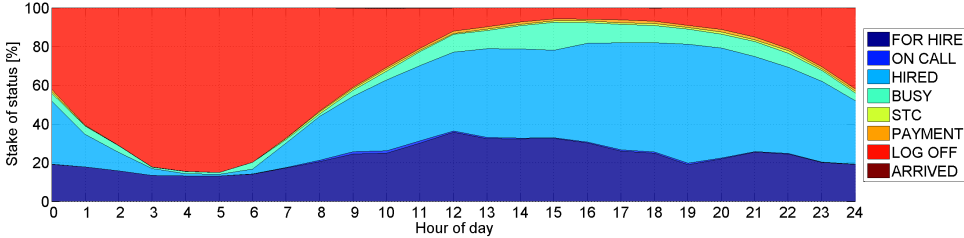
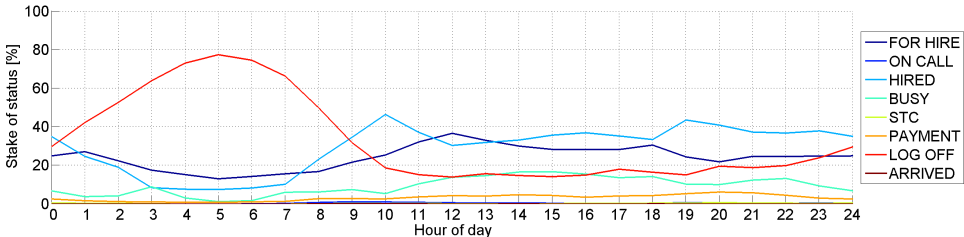
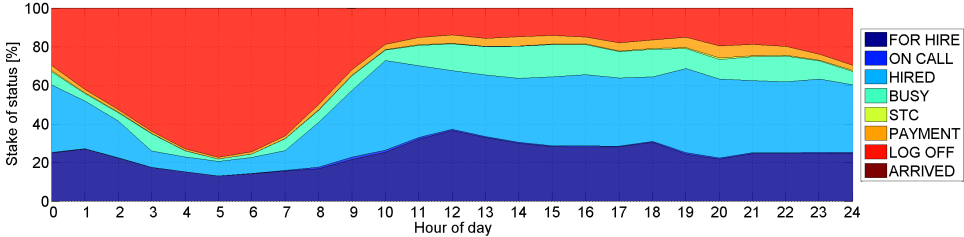


FIGURE E.2: Average status share per taxis type

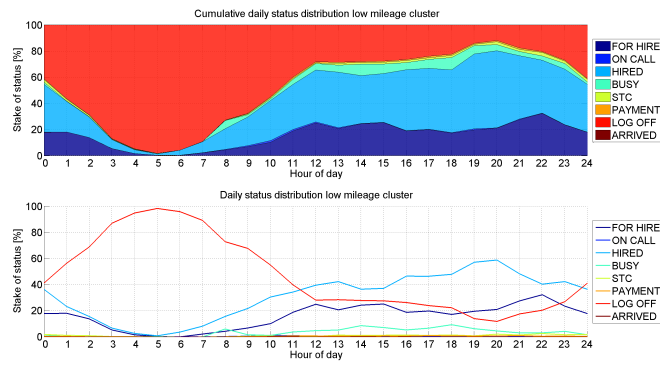


(A) One-hirer scheme taxis

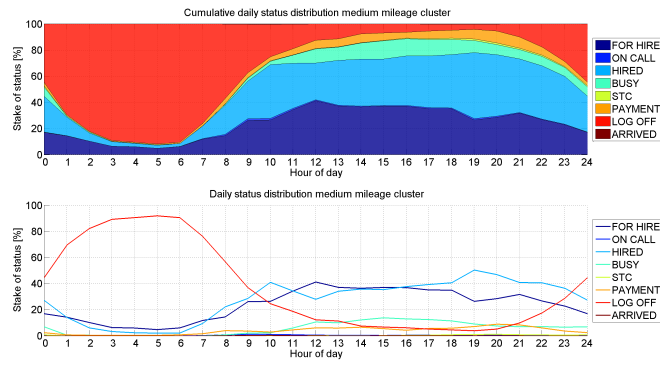


(B) Two-hirer scheme taxis

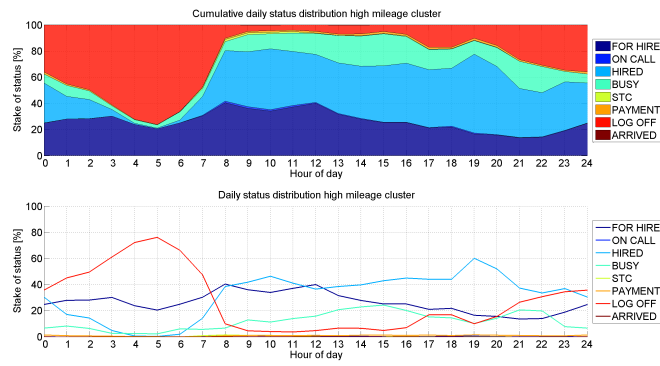
FIGURE E.3: One- and two-hirer scheme status distribution averaged over one day in steps of one hour



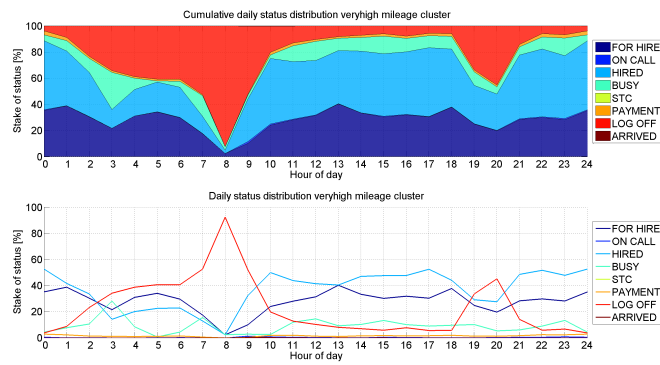
(A) Low mileage cluster



(B) Medium mileage cluster



(C) High mileage cluster



(D) Very high mileage cluster

FIGURE E.4: Daily booking status distribution for clusters *low*, *medium*, *high* and *very high*

Appendix F

Shift Set Statistics

TABLE F.1: Trip set statistics June

Feature	Mean	Min	Max	5%	95%
Shift Duration [h]	7.21	0.02	25.64	0.96	12.09
Shift Mileage [km]	170.28	0.00	554.83	17.13	317.69
Hired Status [%]	55.30	0.00	100.00	0.00	85.24
For-Hire Status [%]	30.18	0.00	100.00	0.00	55.92
Other Status [%]	14.52	0.00	100.00	0.00	50.25
Trip Duration [min]	13.27	0.50	1332.75	1.79	33.30
Trip Length [km]	6.59	0.00	108.52	0.15	22.47
Trip Velocity [km/h]	29.71	0.00	119.64	0.00	85.38
Hired Trip Duration [min]	17.76	0.53	661.97	4.32	36.72
Hired Trip Length [km]	10.58	0.00	71.16	1.26	26.37
Hired Trip Velocity [km/h]	35.70	0.00	119.60	0.00	88.73
Altitude [m]	19.10	-0.27	87.16	8.19	34.26
Altitude Incline [%]	0.00	-16.39	16.49	-1.60	1.60
Velocity Start of Trip [km/h]	1.00	0.00	28.21	0.00	6.30
Velocity End of Trip [km/h]	1.41	0.00	89.98	0.00	5.87
Stand still period distribution [%]	30.34	0.00	100.00	5.82	73.68
Pause duration [min]	29.87	5.02	178.48	5.45	104.22
Energy Consumption [kWh/100km]	18.01	7.89	78.78	13.16	29.11
Hired Energy Consumption [kWh/100km]	16.28	9.94	72.50	13.34	20.12

TABLE F.2: Trip set statistics July

Feature	Mean	Min	Max	5%	95%
Shift Duration [h]	7.07	0.01	25.51	0.79	11.68
Shift Mileage [km]	163.35	0.03	531.35	12.07	304.40
Hired Status [%]	56.30	0.00	100.00	0.00	87.98
For-Hire Status [%]	27.80	0.00	100.00	0.00	51.23
Other Status [%]	15.90	0.00	100.00	0.00	64.03
Trip Duration [min]	13.40	0.52	886.92	1.78	34.17
Trip Length [km]	6.55	0.00	224.39	0.16	22.81
Trip Velocity [km/h]	29.25	0.00	119.62	0.00	84.95
Hired Trip Duration [min]	18.26	0.52	886.92	4.25	38.66
Hired Trip Length [km]	10.59	0.00	224.39	1.21	26.36
Hired Trip Velocity [km/h]	34.72	0.00	119.48	0.00	88.04
Altitude [m]	19.36	-19.10	124.27	8.42	34.83
Altitude Incline [%]	0.00	-16.30	16.02	-1.59	1.60
Velocity Start of Trip [km/h]	1.00	0.00	29.74	0.00	6.01
Velocity End of Trip [km/h]	1.39	0.00	100.94	0.00	5.88
Stand still period distribution [%]	28.75	0.00	100.00	5.47	70.97
Pause duration [min]	27.95	5.00	179.57	5.42	94.74
Energy Consumption [kWh/100km]	17.94	7.51	79.48	13.01	29.20
Hired Energy Consumption [kWh/100km]	16.30	10.01	79.48	13.23	20.37

TABLE F.3: Trip set statistics August

Feature	Mean	Min	Max	5%	95%
Shift Duration [h]	7.17	0.01	19.34	0.67	11.60
Shift Mileage [km]	165.62	0.00	509.57	15.05	316.62
Hired Status [%]	57.75	0.00	100.00	0.00	92.32
For-Hire Status [%]	27.24	0.00	100.00	0.00	56.15
Other Status [%]	15.01	0.00	100.00	0.00	57.68
Trip Duration [min]	13.00	0.52	535.08	1.73	33.77
Trip Length [km]	6.40	0.00	81.84	0.08	22.77
Trip Velocity [km/h]	29.43	0.00	143.44	0.00	84.77
Hired Trip Duration [min]	18.30	0.52	483.37	4.33	39.00
Hired Trip Length [km]	10.50	0.00	81.84	1.05	26.89
Hired Trip Velocity [km/h]	34.38	0.00	143.44	0.00	87.56
Altitude [m]	19.24	-16.29	85.92	8.15	34.35
Altitude Incline [%]	0.00	-15.84	17.02	-1.61	1.61
Velocity Start of Trip [km/h]	1.04	0.00	36.10	0.00	6.14
Velocity End of Trip [km/h]	1.39	0.00	134.15	0.00	5.97
Stand still period distribution [%]	29.57	0.00	100.00	5.19	79.30
Pause duration [min]	28.75	5.00	179.85	5.49	96.94
Energy Consumption [kWh/100km]	18.03	8.03	79.76	13.07	29.42
Hired Energy Consumption [kWh/100km]	16.38	9.09	75.40	13.30	20.51

Appendix G

GPS Logger Manual



GPS Data recording



1. Purpose of this research project

SMRT and TUM CREATE, a Singaporean research institute (www.tum-create.edu.sg), agreed to start a research project to analyse the potential of electric taxis in Singapore. Therefore a detailed analysis of Singaporean taxi driving profiles is essential. In order to record these driving profiles, GPS loggers will be installed in taxis.

There exists a contract between SMRT and TUM CREATE, claiming that the recorded data is confidential, will be exclusively used for scientific purposes, and is not allowed to be forwarded to third parties.

2. Functionality of GPS loggers

GPS loggers are able to record their position very precisely by processing satellite's signals. They won't send any signals at all, so no electronic devices inside the car can be disturbed by their presence.

Because of the long recording period (6 month), they need an external power supply, which can be easily provided by connecting them with the vehicle's cigarette lighter. Since their energy demand is very low, the case, that the car's battery may be discharged until

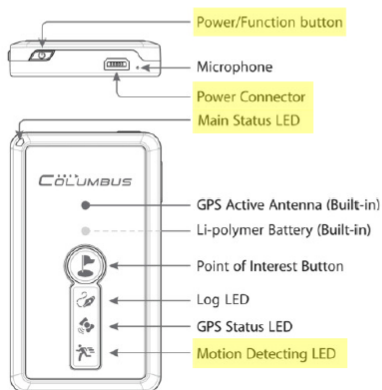
it won't be possible to start it anymore, can be excluded.

There will be a big amount of data recorded during this project, which can't be stored completely on the logger's memory card. That's why the memory card will be exchanged every month.

[...]

4. Operation

The GPS logger, installed in the vehicles, is shown on the picture below.



3

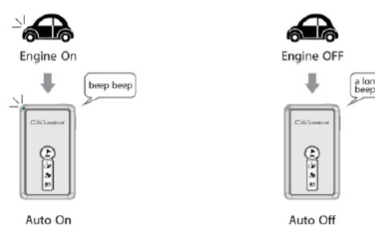
Beginning of shift - Before starting the car:

For the power supply, the logger needs to be connected with the cigarette lighter via cable. At the logger the cable is connected with the **Power Connector** (see picture).

The logger must not be removed from the place of installation.

After starting the car:

If the cigarette lighter is only supplied with power when the engine is running, the GPS logger should switch on automatically.



4

In this case, there will be two short beeps and the **Main Status LED** turns green. If this is not the case, the logger needs to be switched on manually. Therefore the **Power/Function button** has to be pushed for several seconds, until two short beeps occur and the **Main Status LED** turns green.

If it's not possible to switch on the logger, please check all cable connections from the cigarette lighter to the logger – unplug and plug every connection. Afterwards, try to switch on the logger again.

[...]

End of shift – after engine switching off

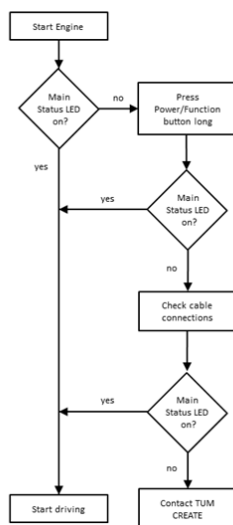
If the power supply of the cigarette lighter is disconnected when the car is off, the logger will switch off automatically – there will be one long beep and the **Main Status LED** turns off. If the logger doesn't switch off automatically, it has to be switched off by pushing the **Power/Function button** for several seconds until there is long beep and the **Main Status LED** turns off.

Please make sure that the logger is always switched on when starting the shift and switched off after finishing

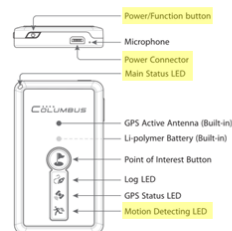
[...]

5. Quick guide

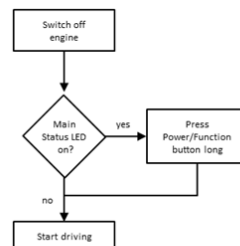
Beginning of shift



GPS Logger



End of shift



5

6

Appendix H

Program Parameters

Monthly Chain

TEST_RUN = true (true, if only a test folder is used instead of real taxis)
TEST_TAXI_NUMBER = 20 (The taxi number to be used in test mode)
ONLY_ONE_DAY = false (Consider only first day)
ONLY_ONE_TAXI_RUN = false (Only consider one taxi)
ONLY_TAXI_NUMBER = 1 (The taxi in one day mode)
CHAIN_START = 1 (First taxi index)
CHAIN_END = 20 (Last taxi index)
SHALL_ONLY_IMPORT = false (true, if only importing is demanded)
SHALL_ONLY_COMBINE = false (true, if only combination of logging and booking is demanded)
SHALL_IMPORT_LOGGING_FROM_CSV = true (true, if import of logging data is demanded)
SHALL_IMPORT_BOOKING_FROM_XLS = true (true, if import of booking data is demanded)
SHALL_COMBINE_BOOKING_AND_LOGGING = true (true, if combination of logging and booking is demanded)
SHALL_PREPROCESS = true (true, if pre-processing is demanded)
SHALL_MATCH_SUBZONES = false (true, if trips shall be matched with Subzones)
SHALL_MAP_MATCH = false (true, if trips shall be matched with road network)
SHALL_SAVE_IMPORT_FILES = true (true, if imported files shall be saved)
SHALL_SAVE_TRIP_FILES = true (true, if processed trips shall be saved)
SHALL_SAVE_COMBINATION_FILES = false (true, if combination data set shall be saved)
SHALL_SAVE_RAW_TRIPS = false (true, if unprocessed trips shall be saved)
SHALL_SAVE_FILTERED_TRIPS = false (true, if filtered trips shall be saved)
SHALL_SAVE_MAP_MATCHED_TRIPS = false (true, if map-matched trips shall be saved)

Booking Set

GAP_FOR_LOG_OFF_INTERPOLATION = 30 (Time in minutes to substitute gaps in booking data with Log Off)

Feature Classification

NUM_OF_CLUSTERS = 3 (Number of clusters for driving feature clustering)

Trajectory Cleaning

CROP_NUMBER_OF_ELEMENTS_START = 10 (Number of sample points to be erased on device start)

Combination

MAX_DISTANCE_THRESHOLD = 500 (Allowance of deviation in metres between both GPS coordinates)

MAX_TIME_DIFF_THRESHOLD = 60 (Allowance of deviation in seconds between both timestamps)

Elevation Map

GRID_SIZE_Y = 200 (Size of elevation map grid in x-axis)

GRID_SIZE_X = 300 (Size of elevation map grid in y-axis)

INTERPOLATION_FACTOR = 20 (Factor to increase elevation map meshed grid granularity)

NUMBER_OF_LAYERS = 20 (Number of layer in the contour map)

Energy Validation

MIN_TRIP_LENGTH = 10 (Min length of trips for representative set)

MAX_TRIP_LENGTH = 30 (Max length of trips for representative set)

TRIPS_PER_SEGMENT = 4 (Number of targeted trips per category)

WHISKER = 0.9826 (Whisker to enable a 2σ box plot distribution)

Evaluation Map

GPS_DEVIATION_THRESHOLD = 20 (Assumed standard deviation for GPS for statistics)

MIN_PAUSE_DURATION = 5 (Duration considered as a pause)

STANDSTILL_VELOCITY_THRESHOLD = 1 (Max speed to be considered as standstill)

Trajectory Filtering

ACCELERATION_OFFSET = 4 (Threshold for acceleration filtering)

DECELERATION_OFFSET = -6 (Threshold for deceleration filtering)

MIN_NUMBER_OF_ELEMENTS = 30 (Minimum sample point required to be contained in a trip)

CROP_NUMBER_OF_ELEMENTS_START_AND_END = 0 (Number of sample points to be erased at trip's edges)

MAX_TIME_DIFFERENCE = 0.5 (Minimum time between two sample points)

MIN_TIME_OF_TRIP = 30 (Minimum time of one trip)

MAX_VELOCITY_THRESHOLD = 200 (Threshold for speed filtering)

GAUSSIAN_WINDOW = 5 (Filter Gaussian window for possible smoothing with Gaussian filtering)

GPS Plots

DEFAULT_VISUALIZATION_DISTANCE = 200 (Distance from sample point to map extracts bored)

Trajectory Interpolation

WEIGHT_FACTOR_ELEVATION_MAP = 1 (Weight denoting share of logger and elevation map height value)

Map Attachments

MIN_DRIVING_POINTS_ON_WAY = 5 (Minimum points on a way to extract energetic characteristics)

Map-Matching

HIGHEST_PROBABILITY = 1 (Norm factor for uniform highest values among all functions)

INTERPOLATION_METHOD = 'pchip' (Interpolation method for measurement point interpolation)

INTERPOLATION_FACTOR = 5 (Factor determining number of interpolated point between two sample points)

CANDIDATE_POINTS = 3 (The number of candidate per road segment)

STAKE_OF_USED_POINTS = 60 (The stake of taken point for map-matching)

DEFAULT_DEVIATION = 50 (Assumed deviation for GPS signals)

MEAN = 0 (Assumed bias of GPS signal)

Micro Trip Extraction

STOP_TRESHHOLD = 1 (Minimum speed considered as a stop)

MIN_DRIVING_POINTS_ON_HIGHWAY = 2 (Minimum sample points on a certain road type)

MICRO_TRIP_TARGET = 210 (Targeted micro trip length in seconds)

MICRO_TRIP_TOLERANCE = 30 (Tolerance band in seconds for micro trip extraction)

MIN_ACCELERATION_REGARDED_AS_CRUSINING = 0.15 (Minimum acceleration considered as cruising)

ZERO_VELOCITY_TRESHHOLD = 1 (Minimum speed considered as standstill)

OSM

MIN_DISTANCE_VALUE = 0.01 (Minimum distance between two nodes for connectivity matrix)

UNKNOWN = 0 (Unknown road type)

MOTORWAY = 1 (Motorway or highway)

TRUNK = 2 (Inner city motorway)

PRIMARY = 3 (Main road)

SECONDARY = 4 (Smaller main road)

TERTIARY = 5 (Smallest main road)

UNCLASSIFIED = 6 (Unclassified road)

RESIDENTIAL = 7 (Residential area road)

SERVICE = 8 (Service roads for public buildings)

MOTORWAY_LINK = 9 (Link from/to a motorway)

TRUNK_LINK = 10 (Link from/to a trunk)

PRIMARY_LINK = 11 (Link from/to a Primary road)

SECONDARY_LINK = 12 (Link from/to a Secondary road)

TERTIARY_LINK = 13 (Link from/to a Tertiary road)

LIVING_STREET = 14 (Residential road with restricted speed limit)

TRACK = 15 (Rural roads)

ROAD = 16 (General road type)

GRID_SIZE_Y = 200 (Grid size in x-direction for OSM grid class)

GRID_SIZE_X = 300 (Grid size in y-direction for OSM grid class)

Trip Set

MAX_DISTANCE_TO_TAXILOCAATION = 50 (Maximum distance to taxi stand for taxi stand localisation)
 MIN_STOP_DURATION = 60 (Minimum time in seconds considered as waiting at a taxi stand)
 MAX_VELOCITY_FOR_STOP = 10 (Maximum speed threshold for taxi stand waiting)
 MAX_OUTLIER_GAP = 5 (Maximum allowed outliers for previous established constraints)

Shift Extraction

MAX_SHIFT_DURATION = 12 (Maximum targeted shift duration)
 TOLERANCE_AJDUSTMENT_BAND = 5 (Tolerance band for shift duration)
 PERCENTAGE_OF_MIN_POINTS = 0.20 (Minimum points in a period considered as normally active)
 EDGES = 14401 (Resolution of shift pattern analysis (minute wise))
 NOISE_EDGES_REDUCTION = 5 (Smooth factor for reducing noise and flattening the distribution)
 ONE_SHIFT_CHANGE_THRESHOLD = 3 (After his time in hours ineffectiveness a shift change is forced)
 MIN_TRIPS_PER_SHIFT = 5 (Minimum trips per shift for post-processing)
 MAX_TIME_BETWEEN_SHIFTS = 3 (Maximum time between shifts in hours for post-processing)
 SHIFT_CHANGE_THRESHOLD = 2 (Maximum time between two shifts on one day for shift change)
 TIME_THRESHOLD = 10 (Threshold in seconds a car has to be stopped for considering as a break)
 TOLERANCE_BAND_OUTLIERS = 10 (Possible outliers for break detection)
 MIN_STOP_TIME = 240 (Minimum stop time in seconds)

Singapore

NORTH = 1.4728372 (Upper north latitude of Singapore)
 SOUTH = 1.1867929 (Bottom south latitude of Singapore)
 WEST = 103.6106791 (Leftmost west longitude of Singapore)
 EAST = 104.0871962 (Rightmost east longitude of Singapore)
 DISTANCE_NORTH_TO_SOUTH = 31806.67495559416 (Distance in metre from north to south)
 DISTANCE_WEST_TO_EAST = 52968.77840846990 (Distance in metre from north to south)
 LAT_DISTANCE_NORTH_TO_SOUTH = 0.2860443 (Distance in spherical coordinates from north to south)
 LON_DISTANCE_WEST_TO_EAST = 0.4765171 (Distance in spherical coordinates from west to east)
 NORTH_SOUTH_DISTANCE_PER_LAT = 111194.9266445588 (Conversion factor from degree to metre for lat.)
 WEST_EAST_DISTANCE_PER_LON = 111158.1901435854 (Conversion factor from degree to metre for long.)
 DISTANCE_PER_DEGREE_MEAN = 111176.5583940721 (Conversion factor from degree to metre mean)
 LAT_DISTANCE_TO_LON_DISTANCE_RATIO = 1.66588566876 (Ratio of north-south and west-east extent)
 CENTER_LAT = 1.32981505 (Singapore center latitude)
 CENTER_LON = 103.84893765 (Singapore center longitude)

Trip

FILTER_COEFFICIENT = 30 (Filter for heading computation for smoothing)
 MIN_SPATIAL_EXTENT = 500 (Minimum average distance for trip considered as 'driving')
 VELOCITY_THRESHOLD = 3 (Minimum average speed for trip considered as 'driving')

Trip Extraction

MIN_TRIP_LENGTH_TO_CONSIDER_LONG_HIRED_TRIPS = 9000 (Minimum distance to apply extraction)
 MIN_TRIP_DISTANCE_TO_CUT = 3000 (Minimum distance for candidates to be allowed to cut)
 TIME_OFFSET = 300 (Time in seconds of inactiveness to cut a trip)
 VELOCITY_STOP_THRESHOLD = 0 (Minimum speed considered as a stop for stop based extraction)
 TOLERANCE_BAND_OUTLIERS = 30 (Maximum outliers for established constraint)
 MAX_DISTANCE_TO_TAXILOCAATION = 50 (Maximum distance to taxi stand for taxi stand localisation)
 MIN_STOP_DURATION = 180 (Minimum stop duration considered as waiting period)
 MAX_VELOCITY_FOR_STOP = 3 (Maximum speed considered as a stop)
 MIN_DISTANCE_TO_LAST_CUT_POINT = 1000 (Distance in metre between two trips for trip exaction)
 MIN_DISTANCE_TO_START_POINT = 200 (Tolerance distance for most distant point analysis)
 MIN_STOP_FOR_PASSENGER_SECONDS = 10 (Time in seconds required for passenger pickup)
 MOVING_WINDOW_SIZE = 10 (Moving window size to detect U-turns)
 UTURN_DEGREE = 180 (Targeted U-turn heading change in degree)
 UTURN_DETECTION_THRESHOLD_DEGREE_TOLERANCE = 20 (Tolerance band for U-turn detection in degree)
 RANGE_IN_METER = 200 (Minimum distance of the period a U-turn was detected)

Appendix I

Matlab Data Folder

ALLSETS All booking and logging sets so far merged and prepared to be used within the programs

AUXMODEL Lookup tables for temperature, time, and sun angle in Singapore and derived roof and door temperatures as inputs for a planed auxiliary energy model

BOOKINGSET All booking data combined on monthly basis for all taxis

DRIVINGSHARE Driving share sets for all micro trips and all road type, both including time zone information

ELEVATION Elevation map for Singapore obtained from Google Elevation API

ENERGY Energy has maps for all micro trips and all road type, both including time zone information

EVALUATION Evaluated statistics for each monthly shift set

LTA Sensor data fetched at various times, containing speed bands of a number of speed sensors provided from LTA

MAPATTACHEMENT Map attachments for energy, driving share, and average speed which is attached to the OSM map for each section and contains the main outcome of the energy demand analysis

MODEL EV characteristics which are used throughout the programs for each model to allow for comparison and uniformity among each other

OSM The OSM road network representation in Matlab, including hashed map containers, section based links, section length, and grid assignments, plus auxiliary files required for building step-wise OSM map representation in Matlab and raw files exported from OSM website

SHIFTSET Cell array based representation of shifts, each including a set of trips

SHIFTSETTABLE Shift set in a table based format including statistical attachments

SMRT Folder structure for the set of SMRT taxis including all logging CSV and booking XLSM files plus temporary MAT cache files

STATISTICS Colour map for unified plot colours in statistic al evaluations

TAXIS List of all taxis attached with GPS loggers

TAXISTANDS List of taxi stands, pickup bays, and bus stops and its raw data gathered from data.gov.sg and processed with QGIS and finally imported to Matlab

TRIPSET Set of trips for each taxi containing a number of tips on monthly basis

ZONES List of regions, planning areas and subzones and its raw data gathered from data.gov.sg and processed with QGIS and finally imported to Matlab

Appendix J

Matlab Source Folder

AUXMODEL Air conditioning lookup table implementation for further implementations of EV models

BOOKING Booking class with fields longitude, latitude, time, status ◊ Interpolation of status from booking data to one minute sampling rate ◊ Status class containing all statuses, string to status conversion, and status to string conversion

CLEAN Cleaning of boundaries where GPS points are not within Singapore's boundary ◊ Cleaning of device starts shortly after device has been started ◊ Time order cleaning to correct time order of GPS points of a recorder trajectory

CLASSIFICATION Classification of driving features and clustering into categories ◊ Retrieval of clusters where a feature set belongs to

COLOR Unique colours throughout the plots

COMBINE Combination of logging and booking data based on time and location ◊ Validation check of combination results by evaluating time and spatial difference of received booking and logging data set

DATABASE Connection to database using credentials ◊ Database query template for SQL queries

DRIVINGSHARE Dynamics entailing acceleration and deceleration parameters, fit functions definitions, fit function parameters, and thresholds ◊ DrivingShare comprising idle, cruise, acceleration, and deceleration share parameters, fit functions, fit function parameters, and thresholds ◊ DrivingShareParameter holds both, Dynamics and DrivingShare instances ◊ Computation of energy based on driving shares for micro trips extracted from whole trip ◊ DrivingShare and Dynamics are computed using mean velocity ◊ Computation of energy based on whole trips ◊ Extraction of Dynamics from a given set of micro trips ◊ Extraction of DrivingShare values from a given set of micro trips ◊ Extraction of share parameters as a combination of Dynamics and DrivingShare ◊ Extraction of share parameters for road types based on road type micro trips using time zones ◊ Definition of fit functions for each driving share ◊ Functions for computing contributing driving power submodules

ELEVATION Construction of meshed grid ◊ Plotting elevation map, in addition contour and gradient ◊ Parsing of map from Google API queries

ENERGY Comparison of all used models ◊ Computation of energy for each models ◊ Extraction of energy values for road type micro trips using time zones ◊ Retrieval of time zones for given trip ◊ Retrieval of time zone set where each cell array contains all trips happened in a certain time ◊ Computation of energy for road types for energy map approach ◊ Computation of energy for a series of traversed road segments for energy map approach ◊ Computation of energy for an origin and destination node for energy map approach ◊ Used default energy value for cases where no value could be retrieved

ENERGYAPPLICATION Comparison of distances for different distance calculation methods ◊ Comparison of driving share application using all methods from driving share approach ◊ Comparison of energy map application using all methods from energy map approach

ENERGYVALIDATION Retrieval of a number of representative trips ◊ Validation of both approaches driving share and energy map

- EVALUATION** Statistical application on different functions receiving mean, max, min, 5% percentile and 95% percentile ◊ Evaluation methods for about 15 different characteristics
- FILTER** Application of all trip set filter on data ◊ Application of all trip specific filter on data ◊ Filtering for acceleration either GPS position or logger speed based ◊ Filtering for speed either GPS position or logger speed based ◊ Filtering sample point count based as a certain number of sample point per trip is required ◊ Filtering where origin and destination of a trip are reduced by an amount of sample points ◊ Filter for a certain minimum trip time is required ◊ Acceleration and speed computation ◊ Trip length computation based on GPS position
- GOOGLE** Class preliminary for plotting Google Maps
- GPS** GPS position as a latitude and longitude class representation ◊ GPS boundary as a boundary object mainly for plotting purpose
- IMPORT** Booking data import based on XLSM sheets ◊ Logging data import reading folder CSV files and converting files to Matlab format also applying data format ◊ Merge of booking data since multiple sources are conceivable
- INTERPOLATION** Gap interpolation application ◊ Computation of slope or gradient ◊ Altitude assignment ◊ Interpolation of big gaps by using velocity method ◊ Re-sampling to ensure time conformity of one second sampling rate
- LTA** Fetching of sensor data from server ◊ Plotting of sensor data
- MAPATTACHEMENTS** Container based adding of values for road segments (energy map and driving share approach) ◊ Conceptualization of the attachment map ◊ Conceptualization by using data fetched from database ◊ Retrieval of a set of trips for given link from DB ◊ Update of an existing value in map containers
- MAPMATCHING** Attaching of road segment ids to appropriate sample points ◊ Computation of observation probability ◊ Computation of Transmission probability ◊ Finding of best matching sequence concerning weights ◊ Retrieval of adjacent indices from map ◊ Retrieval of candidate points in a certain range ◊ Retrieval of all road segment candidates ◊ Retrieval of candidate point for a link ◊ Retrieval of an average version of observed point ◊ Map-matching of a trip ◊ Map Matching class which holds basic configurations ◊ Map-matching of a road segment based algorithm which transforms the OSM fields from 'way' to 'section' ◊ Map-matching of a whole trip set ◊ Plotting of the comparison for matching evaluation ◊ Plotting of the comparison for matching evaluation linked with actual corresponding points ◊ Plotting of fitting sequence ◊ Plotting of sections, ways ◊ Post-processing by reattaching points to ways and omitting wrongly matched ways ◊ Shortening of connectivity matrix to enable fast runtime ◊ Computation of temporal analysis
- MICROTRIP** Extraction of feature set form micro trips ◊ Extraction of features for one micro trip ◊ Extraction of highway micro trips analysing highway changes ◊ Extraction of link based micro trips analysing road segment changes ◊ Feature Set class denoting all field for one feature set ◊ Finding of best cut point for trip according to micro trips ◊ Retrieval of candidates for velocity inspection ◊ Retrieval of candidates for acceleration inspection
- OSM** Trip distance computation based on road segment length ◊ Extraction of connectivity matrices ◊ Retrieval of closest start and end node for a trip ◊ Distance calculation between two nodes ◊ Highway, Node, One-way, Speed, Section, and Way as basic classes for OSM constructs ◊ OSM, OSMap, OSMapGrid, OSMapGridSection denoting the actual classes a OSM is represented ◊ According Parser classes to transform one OSM class into another ◊ Post-processing for connectivity matrix to remove flawed entries
- REGIONS** Import of Planning Areas, Regions and Subzones ◊ Plotting of various zones
- ROUTING** Comparison of different routing or navigation methods ◊ Distance computation based on GPS sample point latitude and longitude values ◊ Retrieval of traversed rad segment from a route consisting of OSM nodes ◊ Retrieval of all latitude and longitude values from a trip ◊ Retrieval of a route for start and end node ◊ Plotting of a route

SETS Attaching of shift set with additional information ◊ Attaching of trip set with additional information
◊ Conversion of shift set into table based format ◊ Creation of a booking set from all booking data per taxi ◊ Creation of a shift set from all shifts per taxi ◊ Creation of a trip set from all shifts per taxi ◊ Finding of taxi stand location in the vicinity of a point ◊ Merge of two or more booking sets ◊ Merge of data into trip sets ◊ Merge of more than one trip set

SHIFT Adjustment of shift change candidates based on location ◊ Analysis of shift change candidates
◊ Extraction of shift change candidates based on patterns ◊ Extraction of shifts ◊ Search for driving periods in trips ◊ Search for standstill periods ◊ Retrieval of best fitting shift change pattern in candidates ◊ Retrieval of break start and end time for pattern recognition ◊ Retrieval of time histogram for active taxi driving times

SINGAPORE Singapore boundaries and extent definitions

TAXISTANDS Import of bus stops, pick-up bays, and taxi stands

TRIP Computation of heading based on GPS coordinates ◊ Computation of trip length based on GPS coordinate, road segment length, and speed based from logger fields ◊ Computation of speeds based on GPS coordinates ◊ Extraction of daily trips from a set of trips ◊ Retrieval of all trips with status Hired ◊ Retrieval of all trips where taxi was actually driving ◊ Retrieval of day start and end time ◊ Retrieval of distances to a GPS coordinate in metres

TRIPEXTRACTION Prove for valid status change cut candidate ◊ Cutting of trips into two or more stints ◊ Trip extraction long hired trip based ◊ Trip extraction status based ◊ Trip extraction stop based ◊ Trip extraction taxi stand based ◊ Trip extraction time based ◊ Finding best suitable cut point ◊ Finding the best point in terms of low speed ◊ Retrieval of possible cut candidates ◊ Retrieval of most distant indices for trip excerpt ◊ Retrieval of relevant status changes ◊ Retrieval of stop indices ◊ Retrieval of of time in driving mode ◊ Retrieval of U-turn candidates ◊ Plotting of best suitable candidates ◊ Validation of U-turn candidates

WAYMATCHING Basically an old approach matching ways merely by location ◊ Plot functions for way matching

XML Export of road segment with energy tags to OSM XML file ◊ Export of all road segments to an OSM XML file ◊ Export function into OSM format for opening with JOSM ◊ Extraction of mean values for each road segment

ZONEMATCHING Matching of sample points to zones

Appendix K

Regions, Planning Areas, and Subzones in Singapore

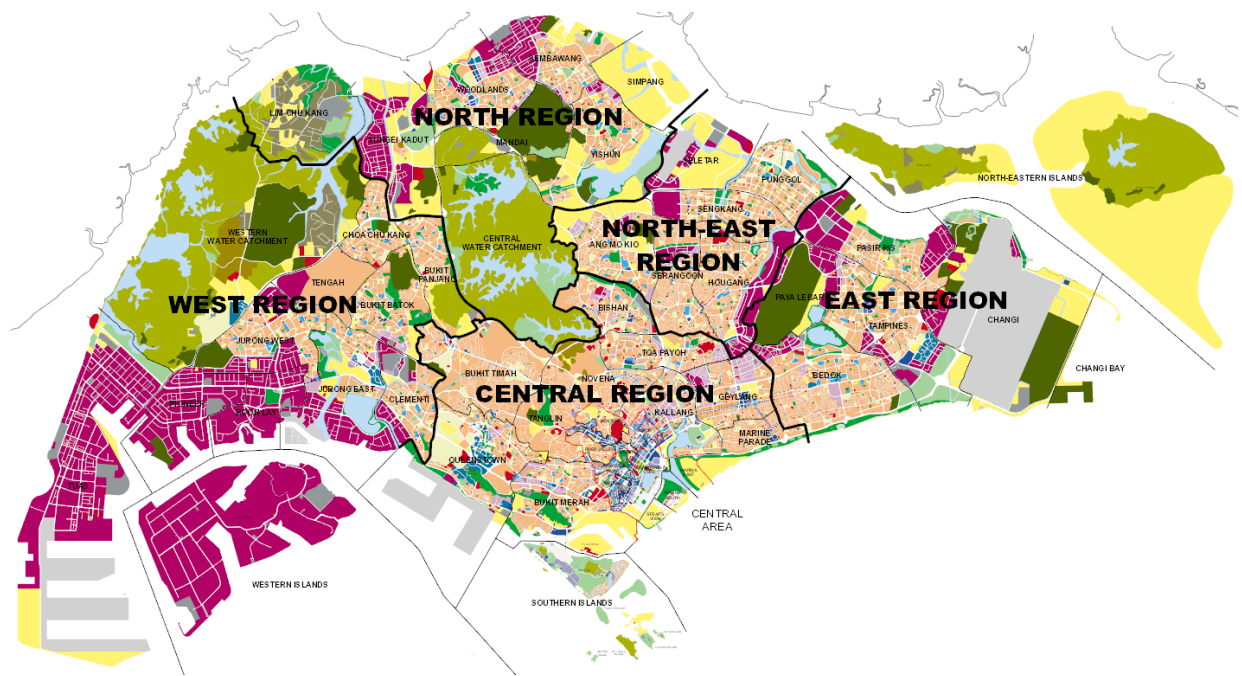


FIGURE K.1: Singapore Regions, Planning Areas, and Subzones, taken from [64]

Appendix L

Taxi Stands in Singapore

TABLE L.1: Taxi stands in Singapore, taken from [2]

Id	Code	Location	Type
<i>SECTOR A (Orchard)</i>			
1	A01	Orchard Road along driveway outside Lucky Plaza	Stand
2	A02	Bideford Road - driveway of Paragon Shopping Centre	Stand
3	A03	Mt Elizabeth Road at York Hotel	PU/DO
4	A04	Mt Elizabeth Road at Paragon Medical Centre	PU/DO
5	A05	Mt Elizabeth Road at Mt Elizabeth Hospital	Stand
6	A06	Cairnhill Road outside Cairnhill Place	Stop
7	A07	Orchard Link at Meritus Mandarin Hotel	PU/DO
8	A08	Orchard Turn outside Wisma Atria Shopping Centre	Stand
9	A09	Orchard Turn along driveway outside Ngee Ann City	Stand
10	A10	Grange Road at Cathay Cineleisure	PU/DO
11	A11	Orchard Road outside Centrepoint	Stand
12	A12	Cuppauge Road at Starhub Centre	Stop
13	A13	Kramat Lane outside Le Meridien Singapore Shopping Centre	Stand
14	A14	Cavenagh Road at Holiday Inn Parkview	Stop
15	A16	Handy Road outside Plaza Singapura	Stand
16	A17	Handy Road near Cathay Building	Stand
17	A18	Penang Road along driveway outside Park Mall	Stand
18	A19	Clemenceau Avenue in front of Haw Par Glass Centre	Stand
19	A20	Penang Road after Istana Park	Stand
20	A21	Killiney Road outside Orchard Central	Stand
<i>SECTOR B (Bugis)</i>			
21	B01	Prinsep Street at Hotel Rendezvous	PU/DO
22	B02	Selegie Road outside Parklane Shopping Mall	Stand
23	B03	Short Street near La Salle Arts College	Stand
24	B04	Bencoolen Street at Summerview Hotel	Stop
25	B05	Bencoolen Street at Bayview Hotel	Stop
26	B06	Bencoolen Street at SMU School of Econ & Social Sciences	Stop
27	B07	Bencoolen Street at SMU School of Information Systems	Stop
28	B08	Queen Street at SMU Li K Shing Library	Stand
29	B09	Queen Street at PA Staff Club	Stand
30	B10	Waterloo Street at Stamford Arts Centre	Stand
31	B11	North Bridge Road along driveway outside Bugis Junction Tower	Stand
32	B12	Tan Quee Lan Street near Heritage Place	Stand
33	B13	Beach Road near Shaw Towers	Stand
34	B14	Purvis Street outside Shop House No 24	Stand
35	B15	Victoria Street at Hotel Grand Pacific	PU/DO
36	B16	Bras Basah Road at Carlton Hotel	PU/DO
37	B17	Bencoolen Street at Masjid Bencoolen	Stop
<i>SECTOR C (City Hall)</i>			
38	C01	River Valley Road outside Clarke Quay	Stand
39	C02	River Valley Road outside Liang Court Shopping Centre	Stand

40	C03	River Valley Road at Novotel Clarke Quay	PU/DO
41	C04	Tan Tye Place at Clarke Quay	Stand
42	C05	Canning Rise near Registry of Marriages	Stand
43	C06	Armenian Street in front of Peranakan Museum	Stand
44	C07	Hill Street outside Funan DigitaLife Mall	Stand
45	C08	Coleman Street along driveway outside Peninsula Excelsior Hotel	Stand
46	C09	North Bridge Road outside Funan DigitaLife Mall	Stop
47	C10	Old Parliament Lane Behind Victoria Theatre	Stand
48	C11	Connaught Drive at The Padang	Stand
49	C12	Coleman Street at The Adelphi	Stop
50	C13	North Bridge Road outside Capitol Building	Stand
51	C14	Stamford Road along driveway outside Raffles City	Stand
52	C15	North Bridge Road along driveway outside Raffles City	Stand
53	C16	Bras Basah Road at The Fairmont	PU/DO
54	C17	Stamford Road along driveway of Swissotel The Stamford	PU/DO
55	C18	Temasek Boulevard at Suntec Tower 1 & 2	Stand
56	C19	Temasek Boulevard at Suntec Tower 3 & 4	Stand
57	C20	Raffles Boulevard at Millenia Walk	Stand
58	C21	Temasek Avenue at Millenia Tower	Stand
59	C22	Raffles Boulevard along driveway outside Marina Square	Stand
60	C23	Raffles Boulevard at Marina Mandarin Hotel	PU/DO
61	C24	Raffles Avenue at The Esplanade	Stand
62	C25	Raffles Boulevard at Suntec City	Stand
63	C26	Percival Road at round-about of Fort Canning Park	Stop

SECTOR D (Chinatown)

64	D01	Teo Hong Road near Outram MRT Station Exit H	Stand
65	D02	Neil Road at Chinatown	Stand
66	D03	Pearl's Hill Terrace outside Pearl's Centre	Stand
67	D04	Kreta Ayer Road opposite Oriental Plaza	Stand
68	D05	Park Crescent outside People's Park Complex	Stand
69	D06	Upper Cross Street outside OG Building	Stand
70	D07	Havelock Square outside People's Park Centre (New Market Road)	Stand
71	D08	New Bridge Road along driveway outside Chinatown Point	Stand
72	D09	New Market Road outside Ministry of Manpower	Stop
73	D10	Magazine Road outside Central Mall	Stand
74	D11	Merchant Road at Riverside Point	Stand
75	D12	Tew Chew Street at Central	Stand
76	D13	Upper Circular Road at The Riverwalk	Stop
77	D14	Carpenter Street Opposite Safra Town Club	Stand
78	D15	North Canal Road opposite Hong Lim Park	Stand
79	D16	South Bridge Road outside Sri Mariamman Temple	Stand
80	D17	Sago Lane Beside Beside Duddha Tooth Relic Temple Museum	Stand
81	D18	Temple Street in front of Grand Court Restaurant	Stand
82	D19	Mosque Street in front of Dragon Brand Bird's Nest Building	Stand
83	D20	Club Street at City State Apartments	Stand
84	D21	Temple Street beside Lucky Chinatown	Stop

SECTOR E (Raffles Place / Tanjong Pagar)

85	E01	Pickering Street at Great Eastern Centre	PU/DO
86	E03	Cross Street at China Square	Stop
87	E04	South Canal Road near OCBC Centre	Stop
88	E05	Market Street outside Golden Shoe Car Park	Stand
89	E06	Amoy Street behind Thian Hock Keng Temple	Stand
90	E07	Telok Ayer Street outside PWC Building	Stand
91	E08	Cecil Street outside GB Building (before McCallum St)	Stop
92	E09	Cecil Street outside Keck Seng Tower	Stop
93	E10	Cecil Street outside Market Street Car Park	Stop
94	E11	D'almeida Street outside Bharat Building	Stop
95	E12	Fullerton Road at One Fullerton Building	Stop
96	E13	Battery Road outside Straits Trading Building	Stop
97	E14	Robinson Road before McCallum Street outside Singapore Post	Stop
98	E15	Robinson Road outside Robinson Point	Stop
99	E16	Raffles Quay near Telegraph Street	Stop

100	E17	Raffles Quay at One Raffles Quay	Stand
101	E18	Shenton Way outside SGX Centre	Stop
102	E19	Shenton Way outside DBS Building	Stop
103	E20	Shenton Way outside MAS Building	Stop
104	E21	Cook Street behind Murray Terrace Food Alley	Stand
105	E22	Duxton Road at Shophouse No 61 (Spinners Pub)	Stand
106	E23	Hoe Chiang Road at Keppel Tower	Stop
107	E24	Tanjong Pagar Road at Amara Hotel	Stop
108	E25	Tanjong Pagar Road outside Tanjong Pagar Plaza	Stand
109	E26	Peck Seah Street near Red Dot Traffic	Stand
110	E27	Bernam Street outside Fuji Xerox Towers	Stand
111	E29	Anson Road at M Hotel	PU/DO
112	E30	Anson Road along driveway outside International Plaza	Stand
113	E31	Robinson Road at Capital Tower	PU/DO
114	E32	Collyer Quay at Hitachi Towers	Stop
115	E33	Circular Road at Shophouse No 5	Stand
116	E34	Tanjong Pagar opposite Mac-Nels Building	Stand
117	E35	Lorong Telok at Archipelago Brewery Building	Stand
118	E36	Raffles Quay outside Lau Pa Sat	Stop
119	E37	Collyer Quay outside OUE Bayfront	Stop
120	E38	Tanjong Pagar at Shophouse No. 68	Stand
121	E39	Cross Street outside Telok Ayer DTL Station	Stand

SECTOR F (CENTRAL)

1	F01	Ang Mo Kio Ave 6 outside Yio Chu Kang MRT Station	Stand
2	F02	Ang Mo Kio Ave 8 outside Ang Mo Kio MRT Station	Stand
3	F03	Bishan Place outside Bishan Junction 8 shopping Centre	Stand
4	F04	Bishan Road opposite Bishan MRT Station (towards Ang Mo Kio)	Stand
5	F05	Bishan Road outside Bishan MRT Station (towards Toa Payoh)	Stand
6	F06	Toa Payoh Central outside Toa Payoh Community Library	Stand
7	F07	Toa Payoh Lorong 1 outside Blk 109 (near Braddell MRT Station)	Stand
8	F08	Toa Payoh Lorong 2 in front of Blk 175	Stand
9	F09	Toa Payoh Lorong 6 along driveway of OrangeTee Building	Stand
10	F10	Upper Thomson Road along driveway of Thomson Plaza	Stand
11	F11	Whampoa Drive outside Blk 90 food centre	Stand
12	F12	Balestier Road outside Balestier Hospital (Parkway Medical)	Stand
13	F13	Buangkok View outside Block 1 (Institute of Mental Health)	Stand
14	F14	Bendemeer Road outside Blocks 25/27 (Bendemeer Shopping Mall)	Stand
15	F15	Kallang Road along driveway of ICA building	Stop
16	F16	Serangoon Road outside Boon Keng NEL Station (Ent B)(opp Blk 102)	Stand
17	F17	Serangoon Road outside Serangoon Plaza	Stand
18	F18	Kitchener Road along driveway of Park Royal Hotel	Stop
19	F19	Jalan Sultan Road near Sultan Plaza	Stand
20	F20	Beach Road along Driveway of Golden Mile Complex	Stand
21	F21	Republic Ave outside Nicoll Highway CCL Station (Ent A)	Stand
22	F23	Hastings Road Outside Tekka Mall	Stand
23	F24	Rochor Canal Road outside Sim Lim Square	Stop
24	F25	Bukit Timah Road along driveway outside Balmoral Plaza	Stand
25	F26	Bukit Timah Rd outside Little India NEL Station (Ent A)	Stand
26	F27	Race Course Road outside Little India NEL Station (Ent C)	Stand
27	F28	Race Course Road outside Farrer Park NEL Station(Ent C)	Stand
28	F29	Race Course Road outside Farrer Park NEL Station (Ent D)	Stand
29	F30	Harbourfront Walk beside Harbourfront Centre	Stand
30	F31	Telok Blangah Crescent near Radin Mas Community Club	Stand
31	F32	Telok Blangah Rd opposite Harbourfront NEL Station (Ent D)	Stand
32	F33	Hospital Drive outside Singapore General Hospital (Block 1)	Stand
33	F34	Outram Road outside Outram Park MRT Station	Stand
34	F35	Outram Road along driveway of Holiday Inn Atrium & Shopping Centre	Stand
35	F36	Bukit Ho Swee Crescent along driveway of Tiong Bahru Plaza	Stand
36	F37	Seng Poh Road outside Tiong Bahru Market	Stand
37	F38	Bukit Merah Central outside Blk 165	Stand
38	F39	Bukit Purmei Ave outside Block 109	Stand
39	F40	Tiong Bahru Rd outside Redhill MRT Station	Stand

40	F41	Commonwealth Ave outside Commonwealth MRT Station (Alexandra Rd)	Stand
41	F42	Commonwealth Ave outside Commonwealth MRT Station (Clementi)	Stand
42	F43	Commonwealth Ave outside Queenstown MRT Station (Clementi)	Stand
43	F44	Commonwealth Ave outside Queenstown MRT Station (Alexandra)	Stand
44	F45	Queensway outside Queenstown Neighbourhood Police Centre	Stand
45	F46	North Bouna Vista Road outside Buona Vista CCL Station (Holland Road)	Stand
46	F47	Jervois Road near Embassy of Malaysia along Holt Road	Stand
47	F48	Chatsworth Road outside Embassy of Republic of Indonesia	Stop
48	F49	Angullia Park outside Liat Towers	Stand
49	F50	Claymore Road outside Orchard Towers	Stop
50	F51	Napier Road along Driveway of Gleneagles Hospital	Stand
51	F52	Scotts Road along driveway of Far East Plaza	Stand
52	F53	Scotts Road outside Newton MRT Station (towards Orchard)	Stand
53	F54	Scotts Road outside Newton MRT Station (towards Newton Circus)	Stand
54	F55	Tanglin Road along driveway of Tanglin Shopping Centre	Stand
55	F56	Tanglin Road outside Tanglin Mall	Stand
56	F57	Cluny Park Road outside Botanic Gardens CCL Station	Stand
57	F58	Toa Payoh Rise outside Caldecott CCL Station	Stand
58	F59	Marina Bay Cruise Centre Singapore	Stand

SECTOR G NORTH-EAST

59	G01	Potong Pasir Ave 1 outside Potong Pasir NEL Station (Ent A)	Stand
60	G02	Upper Serangoon Road outside Potong Pasir NEL Station (Ent B)	Stand
61	G03	Upper Serangoon Road outside Woodleigh NEL Station (Ent B)	Stand
62	G04	Serangoon Ave 3 outside Blk 267	Stand
63	G05	Serangoon Central outside Serangoon CCL Station (Ent E)	Stand
64	G06	Upper Serangoon Road outside Serangoon NEL Station (Ent C)	Stand
65	G07	Serangoon Central outside Serangoon NEL Station (Ent A)	Stand
66	G08	Serangoon Ave 3 outside Lorong Chuan CCL Station (Ent A)	Stand
67	G09	Yio Chu Kang Rd outside Kovan Centre	Stand
68	G10	Upper Serangoon Road outside Kovan NEL Station (Ent C)	Stand
69	G11	Upper Serangoon Road outside Kovan NEL Station (Ent A)	Stand
70	G12	Hougang Central outside Hougang NEL Station (Ent B)	Stand
71	G13	Sengkang Central outside Buangkok NEL Station (Ent A)	Stand
72	G14	Sengkang Central outside Buangkok NEL Station (Ent B)	Stand
73	G15	Punggol Road outside Rivervale Plaza	Stand
74	G16	Sengkang East Way outside Sengkang NEL Station (Ent D)	Stand
75	G17	Sengkang East Way outside Sengkang NEL Station	Stand
76	G18	Sengkang Square outside Sengkang NEL Station (Ent C)	Stand
77	G19	Punggol Central outside Punggol NEL Station (Ent A)	Stand
78	G20	Punggol Central outside Punggol NEL Station (Ent C)	Stand
79	G21	Seletar Airport Passenger Terminal at Seletar West Camp	Stop
80	G22	Pasir Ris Central outside Pasir Ris MRT Station	Stand
81	G23	Pasir Ris Central Street 3 outside White Sands Shopping Mall	Stand

SECTOR H (NORTH-WEST)

82	H01	Ghim Moh Road outside Blk 19	Stand
83	H02	Holland Drive outside Buona Vista Swimming Complex	Stand
84	H03	Holland Ave outside Holland Road Shopping Centre	Stand
85	H04	Holland Road along driveway outside Cold Storage Jelita	Stand
86	H05	Farrer Road outside Farrer Road CCL Station	Stand
87	H06	Upper Bukit Timah Road along driveway of Bukit Timah Shopping Centre	Stand
88	H07	Yishun Ave 2 outside Khatib MRT Station	Stand
89	H08	Yishun Ave 2 outside Yishun MRT Station	Stand
90	H09	Yishun Ave 5 opp Blk 151 near Yishun MRT Station	Stand
91	H10	Yishun Central along driveway of Northpoint Shopping Centre	Stand
92	H11	Deptford Road at Carpark outside Sembawang Wharves	Stand
93	H12	Sembawang Road along driveway outside Sembawang Shopping Centre	Stand
94	H13	Sembawang Way outside Sembawang MRT Station	Stand
95	H14	Woodlands Ave 7 outside Admiralty MRT Station	Stand
96	H15	Woodlands Square (East End) outside Woodlands MRT Station (Ent C)	Stand
97	H16	Woodlands Square (West End) outside Woodlands MRT Station (Ent B)	Stand
98	H17	Woodlands Centre Road outside Woodlands Train Checkpoint Complex	Stand
99	H18	Woodlands Ave 3 outside Marsiling MRT Station	Stand

100	H19	Woodlands Road outside Kranji MRT Station	Stand
101	H20	Jumbo Jet Drive at Singapore Turf Club	Stop

SECTOR I (EAST)

102	I01	Changi Village Rd outside Blk 5	Stand
103	I02	Changi South Ave 1 along driveway of Singapore EXPO	Stand
104	I03	Tampines Ave 4 outside Tampines Mall	Stand
105	I04	Simei Street 6 outside Eastpoint Shopping Mall	Stand
106	I05	New Upper Changi Road outside Tanah Merah MRT station (Ent A)	Stand
107	I06	New Upper Changi Road outside Tanah Merah MRT station (Ent B)	Stand
108	I07	New Upper Changi Rd along driveway outside Blk 209	Stand
109	I08	New Upper Changi Rd outside Bedok MRT station (towards Siglap)	Stand
110	I09	Bedok North Street 3 outside Blk 539	Stand
111	I10	Bedok Reservoir Road outside Sheng Siong Hypermart	Stand
112	I11	Bayshore Road outside The Bayshore	Stand
113	I12	Marine Parade Road outside Mandarin Gardens	Stand
114	I13	Marine Terrace outside Blk 54	Stand
115	I14	Marine Parade Road outside Blk 71	Stand
116	I15	Marine Parade Road along driveway of Parkway Parade	Stand
117	I16	Mountbatten Road along driveway of Katong Shopping Centre	Stand
118	I17	Sims Ave East outside Kembangan MRT station (towards Bedok)	Stand
119	I18	Sims Ave near Geylang Serai Malay Village	Stand
120	I19	Sims Ave near Tanjong Katong Complex	Stand
121	I20	Tanjong Katong Road outside City Plaza	Stand
122	I21	Tanjong Katong Road outside Lion City Hotel	Stand
123	I22	Paya lebar Rd outside Paya Lebar CCL Station (Ent C)	Stand
124	I23	Eunos Road 8 outside Paya Lebar MRT Station (Singapore Post Centre)	Stand
125	I24	Geylang East Central outside Geylang Polyclinic	Stand
126	I25	Geylang Lorong 25A outside Aljunied MRT Station	Stand
127	I26	Bartley Rd outside Bartley CCL Station (Ent A)	Stand
128	I27	Old Airport Road outside Dakota CCL Station (Ent A)	Stand
129	I28	Stadium Walk opposite Singapore Indoor Stadium	Stand
130	I29	Stadium Boulevard outside Stadium CCL Station (Ent B)	Stand

SECTOR J (WEST)

131	J01	Bukit Batok West Ave 5 outside Bukit Gombak MRT Station	Stand
132	J02	Choa Chu Kang Ave 4 near Choa Chu Kang MRT Station	Stand
133	J03	Choa Chu Kang Ave 4 outside Yew Tee MRT Station	Stand
134	J04	Pasir Panjang Rd outside Haw Par Villa	Stand
135	J05	Commonwealth Ave West outside Dover MRT Station (Clementi)	Stand
136	J06	Commonwealth Ave West outside Dover MRT Station (Queenstown)	Stand
137	J07	Commonwealth Ave West outside Clementi MRT Station	Stand
138	J08	Jurong Gateway Road outside Jurong East MRT Station	Stand
139	J09	Jurong Gateway Road in front of JCube (U/C)	Stand
140	J10	Jurong West Ave 1 outside Blk 492	Stand
141	J11	Boon Lay Way outside Chinese Garden MRT Station	Stand
142	J12	Boon Lay Way outside Lakeside MRT Station	Stand
143	J13	Boon Lay Way outside Boon Lay MRT Station (Jurong East)	Stand
144	J14	Boon Lay Way outside Boon Lay MRT Station (Tuas)	Stand
145	J15	Boon Lay Way outside Jurong Point Shopping Mall	Stand
146	J16	Jurong West St 63 outside Pioneer MRT Station (Ent B)	Stand
147	J17	Joo Koon Circle outside Joo Koon MRT Station (Ent A)	Stand
148	J18	Singapore Discovery inside the carpark	Stand
149	J19	Pasir Panjang Road outside Haw Par Villa CCL Station	Stand

Appendix M

Matlab Code Excerpts

```
%% This script is used to import and process a monthly dataset for one taxi.
%% All logging files to import shall be put in a path subfolder logs. The
%% booking data file shall be put into the path folder. Intermediate stages
%% are stored into the path folder.

% Author: Sascha Moecker
% Contact: sascha.moecker@live.de

%#ok<*>UNRCH>

%% Chain attributed, properties and constants
clc;
clear all;
startup;

L.trace('runPrepareChain', sprintf('Starting monthly taxi chain ...'));

% MONTH = '06_june';
% MONTH = '07_july';
% MONTH = '08_august';
MONTH = '09_september';
% MONTH = 'TEST';

% Import and load properties
SHALL_ONLY_IMPORT = false;
SHALL_ONLY_COMBINE = false;

SHALL_IMPORT_LOGGING_FROM_CSV = true;
SHALL_IMPORT_BOOKING_FROM_XLS = true;

% Processing properties
SHALL_COMBINE_BOOKING_AND_LOGGING = true;
SHALL_PREPROCESS = true;

SHALL_MATCH_SUBZONES = false;
SHALL_MAP_MATCH = false;

% Save properties
SHALL_SAVE_IMPORT_FILES = true;
SHALL_SAVE_TRIP_FILES = true;

SHALL_SAVE_COMBINATION_FILES = false;

SHALL_SAVE_RAW_TRIPS = false;
SHALL_SAVE_FILTERED_TRIPS = false;
SHALL_SAVE_HEIGHT_INTERPOLATED_TRIPS = false;
SHALL_SAVE_MAP_MATCHED_TRIPS = false;
SHALL_SAVE_ATTRIBUTED_TRIPS = false;

% Test and debug properties
TEST_RUN = true;
TEST_TAXI_NUMBER = 20;

% One day
ONLY_ONE_DAY = false;

% One taxi
ONLY_ONE_TAXI_RUN = false;
ONLY_TAXI_NUMBER = 1;

% At which position should the chain start
CHAIN_START = 1;
CHAIN_END = 20;

%% Load required data Loading previously established data and auxiliary input
L.trace('runPrepareChain', sprintf('Loading required data ...'));

% ElevationMap Constructed elevation map of Singapore for height interpolation
```

```

L.trace('runPrepareChain', sprintf('Loading elevation map ...'));
load('data/elevation/elevationMap60000InterpolatedFiltered.mat');

% Taxistands List of taxi stand locations in Singapore
L.trace('runPrepareChain', sprintf('Loading taxistands, pickup-bays and busstops ...'));
load('data/taxistands/taxistands.mat');
load('data/taxistands/pickupbays.mat');
load('data/taxistands/busstops.mat');

% Zones - Cell array of zone structs denoting seperations
L.trace('runPrepareChain', sprintf('Loading regions, planning areas and subzones ...'));
load('data/zones/regions.mat');
load('data/zones/planningAreas.mat');
load('data/zones/subzones.mat');
load('data/zones/subzones_ex.mat');

% OSM coastline map ?OSM map denoting waterfront borders of Singapore
L.trace('runPrepareChain', sprintf('Loading OSM coastline map ...'));
load('data/osm/osmCoastlineMap.mat');

% Aux
L.trace('runPrepareChain', sprintf('Loading colormap for stats ...'));
load('data/statistics\cmap.mat');

% Energy
L.trace('runPrepareChain', sprintf('Loading EVCharacteristics for energy estimation ...'));
load('data\model\EVCharacteristics.mat');

% OSM Map OSM map of Singapore including all nodes and ways
if SHALL_MAP_MATCH
    L.trace('runPrepareChain', sprintf('Loading OSM map grid section ...'));
    load('data/osm/osmMapGridSectionDistance.mat');

    L.trace('runPrepareChain', sprintf('Loading connectivity matrix ...'));
    load('data/osm/connectivityMatrixDistance.mat');
end

% Taxis liscence numbers All tracked and participating taxis including taxi type
L.trace('runPrepareChain', sprintf('Loading taxis liscence numbers ...'));
load('data/taxis/taxis.mat');

%% Only one taxi (determind by ONLY_TAXI_NUMBER) is considered
if ONLY_ONE_TAXI_RUN
    taxis = taxis(ONLY_TAXI_NUMBER, :);
end

%% Test run uses test data of taxi TEST_TAXI_NUMBER
if TEST_RUN
    TAXI_REAL_NAME = taxis(TEST_TAXI_NUMBER, 1);
    taxis = taxis(TEST_TAXI_NUMBER, :);
    taxis{1} = 'TEST';
end

```

LISTING M.1: Preparation chain

```

%% This script performs all trip set extraction methods
for i = CHAIN_START:min(CHAIN_END, size(taxis, 1))

    %% Basic setup Set Folders for taxi and month
    TAXI = taxis(i, 1);

    L.info(['runTripSetChain <', num2str(i), '>', ...
        sprintf('Run monthly chain for taxi <%s> in <%s> with loop index <%d>', TAXI, MONTH, i)]);

    path = ['data/smrt/', TAXI, '/', MONTH];
    bookingPath = ['data/smrt/booking/', MONTH];
    L.trace('runTripSetChain', sprintf('Path set to <%s>', path));
    L.trace('runTripSetChain', sprintf('Booking path set to <%s>', bookingPath));

    %% Import or load logging data
    if SHALL_IMPORT_LOGGING_FROM_CSV
        L.trace(['runTripSetChain <', num2str(i), '>', ...
            sprintf('Importing logging data ...')]);
        data = importLoggingData([path, '/logs']);

        if isempty(data)
            % Continue with next taxi, since no data was fetched
            L.trace(['runTripSetChain <', num2str(i), '>', ...
                sprintf('Skipping taxi, since no data could be imported')]);
        else
            %% Save logging data
            if SHALL_SAVE_IMPORT_FILES
                L.trace(['runTripSetChain <', num2str(i), '>', ...
                    sprintf('Saving imported logging data ...')]);
                save([path, '/data.mat'], 'data');
            end
        end
    else
        %% Load logging data from saved files
        L.trace(['runTripSetChain <', num2str(i), '>', ...
            sprintf('Loading logging data ...')]);
        try
            load(['data/smrt/', TAXI, '/', MONTH, '/data.mat']);
        catch
        end
    end
end

```

```

catch
    L.trace(['runTripSetChain <', num2str(i), '>'], ...
        sprintf('Could not load logging data, skipping taxi ...'));
end
end

%% Import or load booking data
if SHALL_IMPORT_BOOKING_FROM_XLS
    L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Importing booking data...'));
    if TEST_RUN
        dataBooking = importBookingDataFromOneSheet(bookingPath, TAXI_REAL_NAME, MONTH);
    else
        dataBooking = importBookingDataFromOneSheet(bookingPath, TAXI, MONTH);
    end

    %% Save booking data
    if isempty(dataBooking)
        L.trace(['runTripSetChain <', num2str(i), '>'], ...
            sprintf('Skipping taxis booking data, since no booking data could be imported'));
    else
        if SHALL_SAVE_IMPORT_FILES
            L.trace(['runTripSetChain <', num2str(i), '>'], ...
                sprintf('Saving imported booking data ...'));
            save([path, '/dataBooking.mat'], 'dataBooking');
        end
    end
else
    %% Load booking data from saved files
    L.trace(['runTripSetChain <', num2str(i), '>'], ...
        sprintf('Loading booking data ...'));
    try
        load(['data/smrt/', TAXI, '/', MONTH, '/dataBooking.mat']);
    catch
        L.trace(['runTripSetChain <', num2str(i), '>'], ...
            sprintf('Could not load booking data, skipping taxi'));
    end
end

%% Only import mode forces to skip the next stages
if SHALL_ONLY_IMPORT
    L.trace(['runTripSetChain <', num2str(i), '>'], ...
        sprintf('Only import mode is on, continue to next available taxi'));
    continue;
end

%% Check if we have data
if isempty(data) || isempty(dataBooking)
    % Continue with next taxi, since no data was fetched
    L.trace(['runTripSetChain <', num2str(i), '>'], ...
        sprintf('Skipping taxi, since no logging data or booking data available'));
    continue;
end

%% Only one day is considered
if ONLY_ONE_DAY
    dailyData = extractDailyTrips(data);
    data = dailyData{1};
end

%% Combination of booking and logging data
if SHALL_COMBINE_BOOKING_AND_LOGGING
    L.trace(['runTripSetChain <', num2str(i), '>'], ...
        sprintf('Combination ...'));
    [data, combinationIndices] = combineDatasets(data, dataBooking);

    %% Cleaning data
    L.trace(['runTripSetChain <', num2str(i), '>'], ...
        sprintf('Cleaning and first trip set filtering...'));
    [data, combinationIndices] = applyCleaning(data, combinationIndices);
    [data, combinationIndices] = applyTripSetFilters(data, combinationIndices);

    %% Save combined data
    if SHALL_SAVE_COMBINATION_FILES
        L.trace(['runTripSetChain <', num2str(i), '>'], ...
            sprintf('Saving combined data ...'));
        save([path, '/dataCombined.mat'], 'data');
        L.trace(['runTripSetChain <', num2str(i), '>'], ...
            sprintf('Saving combination indices ...'));
        save([path, '/combinationIndices.mat'], 'combinationIndices');
    end
else
    %% Load booking data from saved files
    L.trace(['runTripSetChain <', num2str(i), '>'], ...
        sprintf('Loading combination data ...'));
    try
        load(['data/smrt/', TAXI, '/', MONTH, '/dataCombined.mat']);
        load(['data/smrt/', TAXI, '/', MONTH, '/combinationIndices.mat']);
    catch
        L.trace(['runTripSetChain <', num2str(i), '>'], ...
            sprintf('Could not load combination data or combination indices, skipping taxi ...'));
        continue;
    end
end

%% Only import mode forces to skip the next stages
if SHALL_ONLY_COMBINE
    L.trace(['runTripSetChain <', num2str(i), '>'], ...

```

```

        sprintf('Only combine mode is on, continue to next available taxi'));
        continue;
    end

    %% Extraction of trips from raw data
    L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Extraction ...'));
    data = applyTripExtraction(data, combinationIndices, taxistands, pickupbays, busstops);
    data = applyTripSetFilters(data);

    %% Save raw trips
    if SHALL_SAVE_RAW_TRIPS
        L.trace(['runTripSetChain <', num2str(i), '>'], ...
            sprintf('Saving extracted raw trips ...'));
        save([path, '/tripsRaw.mat'], 'data');
    end

    %% Filtering raw trip to retrieve reasonably filtered trips
    if SHALL_PREPROCESS
        L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Filtering ...'));
        data = applyTripSetFilters(data);
        data = applyTripSpecificFilters(data);
        data = applyTripSetFilters(data);

        if SHALL_SAVE_FILTERED_TRIPS
            L.trace(['runTripSetChain <', num2str(i), '>'], ...
                sprintf('Saving filtered trips ...'));
            save([path, '/tripsFiltered.mat'], 'data');
        end

        %% Interpolation of erased entries in trips to ensure frequency conformity
        L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Height Interpolation ...'));
        data = heightInterpolation(data, elevationMap);

        if SHALL_SAVE_HEIGHT_INTERPOLATED_TRIPS
            L.trace(['runTripSetChain <', num2str(i), '>'], ...
                sprintf('Saving height interpolated trips ...'));
            save([path, '/tripsHeightInterpolated.mat'], 'data');
        end

        L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Resampling Interpolation ...'));
        data = resamplingInterpolation(data);
    end

    %% Match subzone to points
    if SHALL_MATCH_SUBZONES
        L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Subzone-Matching ...'));
        data = matchZonesEx(data, subzones_ex);
    end

    %% Save preprocessed trips
    if SHALL_SAVE_TRIP_FILES
        L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Saving processed trips ...'));
        save([path, '/trips.mat'], 'data');
    end

    %% Map Matching
    if SHALL_MAP_MATCH
        %% Map match to road network
        L.trace(['runTripSetChain <', num2str(i), '>'], sprintf('Map-Matching ...'));
        data = mapMatchSection(data, osmMapGridSection, connectivityMatrix);

        if SHALL_SAVE_MAP_MATCHED_TRIPS
            %% Save map matched trips
            L.trace(['runTripSetChain <', num2str(i), '>'], ...
                sprintf('Saving map matched trip ...'));
            save([path, '/tripsMapMatched.mat'], 'data');
        end
    end

    %% Save trips with attributes
    if SHALL_SAVE_ATTRIBUTED_TRIPS
        data = attachTripsWithAttributes(data);
        save([path, '/tripsAttributes.mat'], 'data');
    end

end

%% Build and save Sets of all taxis
% Trip set
L.trace('runTripSetChain', sprintf('Creating trip set ...'));
tripSet = createTripSet(MONTH, taxis);

L.trace('runTripSetChain', sprintf('Saving trip set ...'));
save(['data/tripset/tripSet', MONTH, '.mat'], 'tripSet');

% Booking set
L.trace('runMonthlyTaxiChain', sprintf('Creating booking set ...'));
bookingSet = createBookingSet(MONTH, taxis);

L.trace('runMonthlyTaxiChain', sprintf('Saving booking set ...'));
save(['data/bookingSet/bookingSet', MONTH, '.mat'], 'bookingSet');

```

LISTING M.2: Trip set chain

```

%% This script performs all map matching steps
L.trace('runMapMatchingChain', sprintf('Loading OSM map grid section ...'));
load('data/osm/osmMapGridSectionDistance.mat');

L.trace('runMapMatchingChain', sprintf('Loading connectivity matrix ...'));
load('data/osm/connectivityMatrixDistance.mat');

tripSet = mapMatchTripSet(tripSet, osmMapGridSection, connectivityMatrix);
tripSet = matchZonesExTripSet(tripSet, zones);

```

LISTING M.3: Map-matching chain

```

%% This script performs all shift set extraction methods
L.trace('runShiftSetChain', sprintf('Creating shift set ...'));
shiftSet = createShiftSet(tripSet);
L.trace('runShiftSetChain', sprintf('Attaching shift set with extracted attributes ...'));
shiftSet = attachShiftSetWithAttributes(shiftSet);
L.trace('runShiftSetChain', sprintf('Attaching shift set with statistics ...'));
shiftSet = attachShiftSetWithStatistics(shiftSet, EVCharacteristics);

L.trace('runShiftSetChain', sprintf('Attaching shift set with taxistand locations ...'));
shiftSet = findTaxistandLocations(shiftSet, taxistands);

L.trace('runShiftSetChain', sprintf('Saving shift set ...'));
save(['data/shiftSet/shiftSet', MONTH, '.mat'], 'shiftSet');

% Shift set table
L.trace('runShiftSetChain', sprintf('Convert shift set to table ...'));
shiftSetTable = convertShiftSetToTable(shiftSet);

L.trace('runShiftSetChain', sprintf('Saving shift set table ...'));
save(['data/shiftSetTable/shiftSetTable', MONTH, '.mat'], 'shiftSetTable');

% Statistics
L.trace('runShiftSetChain', sprintf('Perform shift set evaluation ...'));
statistics = evaluateShiftSet(shiftSet, EVCharacteristics);

L.trace('runShiftSetChain', sprintf('Saving evaluated statistics ...'));
save(['data/statistics/evaluation', MONTH, '.mat'], 'statistics');

```

LISTING M.4: Shift set chain

```

function data = importLoggingData(path)
%IMPORT Imports logging data from given path

L = evalin('base', 'L');
HAEDER_ROW_COUNT = 1;

% Time offset between GMT time and Singapore time
TIME_OFFSET = datenum([0 1 0 8 0 0]);
CSV_TYPE = 'CSV';

filePaths = getAllFiles(path);
L.trace('importLoggingData', sprintf('Got total of <#> file(s)', numel(filePaths)));

% Create empty data array
data = {};

% Erase files from list which do not satisfy requirements for logging data
eraseIndices = [];
for i = 1:numel(filePaths)
    path = filePaths{i};
    type = path(end-2:end);
    if ~strcmp(type, CSV_TYPE)
        L.trace('importLoggingData', sprintf('Will skip file <#>', path));
        eraseIndices = [eraseIndices, i];
        continue;
    end
end
filePaths(eraseIndices) = [];

for i = 1:numel(filePaths)
    path = filePaths{i};

    L.trace('importLoggingData', sprintf('Read trip <#> at <#>', i, path));

    % Read CSV file and assign fields to variables
    [date, time, lat, lon, altitude, velocity, heading] = ...
        textread(path, ...
            '%s %s %u %u %s %s %s %s %s', ...
            'headerlines', HAEDER_ROW_COUNT, ...
            'delimiter', ',');

    trip = Trip;

    %% Time
    dateTime = [num2str(date), num2str(time, '%06i')];
    pattern = datevec(dateTime, 'yymmdd HHMMSS', 1960);

```

```

time = datenum(pattern) + TIME_OFFSET;

trip.time = time;

%% LatLon
lat = (cellfun(@(x) single(str2double(x(1:end-1))), lat));
trip.lat = lat;
lon = (cellfun(@(x) single(str2double(x(1:end-1))), lon));
trip.lon = lon;

%% Altitude
altitude = (cellfun(@(x) int16(str2double(x)), altitude));
trip.altitude = altitude;

%% Velocity
velocity = (cellfun(@(x) int16(str2double(x)), velocity));
trip.velocity = velocity;

%% Heading
heading = (cellfun(@(x) int16(str2double(x)), heading));
trip.heading = heading;

%% Optional Fields
trip.setupOptionalFields();

% Assign trip to data set
data{1, 1} = trip;
end

L.trace('importLoggingData', sprintf('Got <#d> trips', length(data)));

end

```

LISTING M.5: Import of logging data

```

function dataBooking = importBookingDataFromOneSheet(path, taxi, month)
%IMPORT Imports booking data from one Excel sheet respect to the queried
%taxi licence plate number

L = evalin('base', 'L');
XLS_TYPE = 'xls';
XLSM_TYPE = 'xlsm';

filePaths = getAllFiles(path);
L.trace('importBookingData', sprintf('Got total of <#d> file(s)', numel(filePaths)));

dataBooking = {};

% Erase files from list which to not satisfy requirements for booking data
eraseIndices = [];
for i = 1:numel(filePaths)
    path = filePaths{i};
    [~, ~, type] = fileparts(path);
    type = type(2:end);
    if ~strcmp(type, XLS_TYPE) && ~strcmp(type, XLSM_TYPE)
        L.trace('importBookingData', sprintf('Will skip file <#s>', path));
        eraseIndices = [eraseIndices, i];
        continue;
    end
end
filePaths(eraseIndices) = [];

for i = 1:numel(filePaths)
    path = filePaths{i};

    L.trace('importBookingData', sprintf('Read booking file <#d> at <#s>', i, path));

    % Read xls file and store it in cell array of numeric elements and
    % string elements
    [numValues, stringValues] = xlsread(path, taxi);

    if isempty(numValues) || isempty(stringValues)
        L.trace('importBookingData', 'No value found for matching taxi, will skip file');
        continue;
    end

    % Create a new Booking object
    booking = Booking;

    %% LatLon
    booking.lat = numValues(:, 1);
    booking.lon = numValues(:, 2);

    %% Time
    time = [];
    for j = 1:length(stringValues)
        try
            time(j, 1) = datenum(stringValues(j, 5), 'dd/mm/yyyy HH:MM:SS AM');
        catch
            time(j, 1) = datenum(stringValues(j, 5), 'dd/mm/yyyy');
        end
    end
    booking.time = time;

```

```

%% Status
booking.status = stringValues(:, 4);

monthDigit = month(3:4);
monthDigit = strrep(monthDigit, '0', '');
monthDigit = str2double(monthDigit);

yearDigit = month(1:2);
yearDigit = strrep(yearDigit, '0', '');
yearDigit = ['20', yearDigit];
yearDigit = str2double(yearDigit);

% Filter those indices which are in the mnth range.
% It uses 2014 year for easing import. Also, since the dates of the CSV
% files are in UTC time and the booking dates are in Singaporean time,
% we add a one day tolerance filed to the booking data threshold
indices = find(booking.time >= datenum([yearDigit monthDigit 1 0 0 0]) ...
    & booking.time < datenum([yearDigit monthDigit+1 2 0 0 0]));

if ~isempty(indices)
    booking = booking.getExtraction(indices(1):indices(end));
    booking.sort();
else
    L.warn('importBookingData', ...
        sprintf('No booking data could be fetched for given month <%s>', month));
    booking = [];
end

% Assign booking object to data
dataBooking(i, 1) = booking;
end

L.trace('importBookingData', sprintf('Merge <%d> booking data into one booking', length(dataBooking)));
dataBooking = mergeBookings(dataBooking);

L.trace('importBookingData', sprintf('Got <%d> booking data', length(dataBooking)));
end

```

LISTING M.6: Import of booking data

```

function data = filterVelocityBasedLogger(data)
%PREPROCESSTRIPS Filters all trips in data according to velocity outliers

L = evalin('base', 'L');

% km/h
MAX_VELOCITY_THRESHOLD = 200;

L.trace('filterVelocityBasedLogger', ...
    sprintf('Filter values exceeding velocity of <%dkm/h>', MAX_VELOCITY_THRESHOLD));

for i = 1:length(data)
    countBefore = length(data(i).time);

    trip = data(i);

    j = 1;
    while j < length(trip.time) - 1
        j = dynamicFilterVelocityLogger(trip, j, MAX_VELOCITY_THRESHOLD);
    end

    countAfter = length(data(i).time);

    L.trace('filterVelocityBasedLogger', ...
        sprintf('Removed <%d/%d> entries of trip <%d>', ...
            countBefore - countAfter, countBefore, i));
end
end

```

LISTING M.7: Filter for velocity of logger field

```

function j = dynamicFilterVelocityLogger...
    (trip, j, threshold)
%DYNAMICS Filters sample points in terms of thresholds for acceleration,
%deceleration and velocity

% Compute current velocity based on lat lon changes
velocity = trip.velocity(j);

if abs(velocity) > threshold
    % Compute velocity happening when we skip next velocity
    velocitySkipNext = trip.velocity(j+2);

    % Compute velocity happening when we skip current velocity
    velocitySkipCurrent = trip.velocity(j+1);

```



```

% If both values while skipping current or next exceed boundaries,
% delete both entries and skip backstep steps back restarting the
% algorithm
if abs(velocitySkipCurrent) > threshold ...
    && abs(velocitySkipNext) > threshold
    trip.deleteEntries([j, j+1]);
    % Jump to first point
    j = 1;
else
    % Else erase the value which caused the error. This means, if the
    % skipped value now lays within boundaries, remove the other one
    if abs(velocitySkipCurrent) <= threshold
        trip.deleteEntries(j);
    end
    if abs(velocitySkipNext) <= threshold
        trip.deleteEntries(j+1);
    end
    j = max(1, j - 1);
end
else
    j = j + 1;
end
end
end

```

LISTING M.8: Dynamic filter algorithm

```

function data = filterTimestampBased(data)
%FILTERTIMESTAMPBASED Filters values which do not obey the timestamp
%constraints or have the same timestamp

L = evalin('base', 'L');

MAX_TIME_DIFFERENCE = datenum([0 1 0 0 0 0.5]);

L.trace('filterTimestampBased', ...
    sprintf('Remove entries where the timestamp equals'));

countBefore = cellfun(@(x) length(x.time), data);

% Compute time difference per trip
difference = cellfun(@(x) diff(x.time), data, 'UniformOutput', false);

% Erase indices where time difference is smaller than limit
eraseIndices = cellfun(@(x) find(x < MAX_TIME_DIFFERENCE), difference, 'UniformOutput', false);

countRemoved = cellfun(@(x) length(find(x~=0)), eraseIndices);

% Actual removing of entries
cellfun(@(x, y) x.deleteEntries(y), data, eraseIndices);

for i = 1:length(data)
    L.trace('filterTimestampBased', ...
        sprintf('Removed <%d/%d> entries for trip <%d>',...
            countRemoved(i), countBefore(i), i));
end
end

```

LISTING M.9: Filter for timestamps

```

function data = resamplingInterpolation(data)
%RESAMPLING Interpolates missing value of trips in data or replaces
%misarranged sample points

L = evalin('base', 'L');

% Resampling rate
ONE_SECOND = datenum([0 1 0 0 0 1]);

% Define different interpolation strategies for differenc tpuposes
INTERPOLATION_METHOD_SPATIAL = 'pchip';
INTERPOLATION_METHOD_DYNAMICS = 'pchip';
INTERPOLATION_METHOD_TAGS = 'nearest';

L.trace('resamplingInterpolation', ...
    sprintf('Resample dataset by interpolating missing values'));

for i = 1:length(data)
    trip = data{i};

    % Skip this step due to uncertainty
    [timeForVelocity, velocities] = interpolateBigGaps(trip);

    timeForVelocity = trip.time;
    velocities = trip.velocity;

    % Compute the interplated time defined by one-second interval
    timeInterpolated = trip.time(1):ONE_SECOND:trip.time(:end);

```

```

% Interpolate each element with sampling rate for spatial information
latInterpolated = interp1(trip.time, double(trip.lat), timeInterpolated, ...
    INTERPOLATION_METHOD_SPATIAL);
lonInterpolated = interp1(trip.time, double(trip.lon), timeInterpolated, ...
    INTERPOLATION_METHOD_SPATIAL);

% Interpolate fields for dynamics
velocityInterpolated = interp1(timeForVelocity, double(velocities), timeInterpolated, ...
    INTERPOLATION_METHOD_DYNAMICS);
velocityInterpolated(velocityInterpolated < 0) = 0;

altitudeInterpolated = interp1(trip.time, double(trip.altitude), timeInterpolated, ...
    INTERPOLATION_METHOD_DYNAMICS);
headingInterpolated = interp1(trip.time, double(trip.heading), timeInterpolated, ...
    INTERPOLATION_METHOD_DYNAMICS);

% Interpolate discrete fields mainly using nearest neighbour method
statusInterpolated = interp1(trip.time, double(trip.status), timeInterpolated, ...
    INTERPOLATION_METHOD_TAGS);
wayInterpolated = interp1(trip.time, double(trip.way), timeInterpolated, ...
    INTERPOLATION_METHOD_TAGS);
highwayInterpolated = interp1(trip.time, double(trip.highway), timeInterpolated, ...
    INTERPOLATION_METHOD_TAGS);
subzoneInterpolated = interp1(trip.time, double(trip.subzone), timeInterpolated, ...
    INTERPOLATION_METHOD_TAGS);

% Assign interpolated values to trip. Here we define the used data
% format as it is the last preprocessing step
trip.time = timeInterpolated';

trip.lat = single(latInterpolated');
trip.lon = single(lonInterpolated');

trip.velocity = single(velocityInterpolated');
trip.altitude = single(altitudeInterpolated');
trip.heading = int16(headingInterpolated');

trip.status = int8(statusInterpolated');
trip.way = int32(wayInterpolated');
trip.highway = int8(highwayInterpolated');
trip.subzone = int16(subzoneInterpolated');
end
end

```

LISTING M.10: Interpolation of trips

```

function data = heightInterpolation(data, elevationMap)
%HEIGHTINTERPOLATION Interpolates the height by matching the position of
%each point to a grid of Singapore. This grid is attached with an elevation
%value received from Google Elevation Api

L = evalin('base', 'L');

% For now, only use values from elevation map
WEIGHT_FACTOR_ELEVATION_MAP = 1;

L.trace('heightInterpolation', ...
    sprintf('Interpolating altitude values by using Google elevation values with weight factor <math>\times</math>', ...
    WEIGHT_FACTOR_ELEVATION_MAP));

% Get lat and lon range
lats = linspace(Singapore.SOUTH, Singapore.NORTH, elevationMap.GRID_SIZE_INTERPOLATED_Y - ...
    elevationMap.INTERPOLATION_FACTOR + 1);
lons = linspace(Singapore.WEST, Singapore.EAST, elevationMap.GRID_SIZE_INTERPOLATED_X - ...
    elevationMap.INTERPOLATION_FACTOR + 1);

for i = 1:length(data)
    L.trace('heightInterpolation', ...
        sprintf('Interpolate height for trip <math>i</math>', i));

    for j = 1:length(data{i}.time)

        % Get nearest index in x and y direction for current point
        [xIndex, yIndex] = min(abs(lons - data{i}.lon(j)), ...
            abs(lats - data{i}.lat(j)));

        actualDistanceX = lons(xIndex) - data{i}.lon(j);
        actualDistanceY = lats(yIndex) - data{i}.lat(j);

        actualPoint = [data{i}.lon(j), data{i}.lat(j)];

        shiftX = 0;
        if actualDistanceX > 0
            shiftX = 1;
        end

        adjacentIndices(1, 1) = xIndex - shiftX;
        adjacentIndices(2, 1) = xIndex + 1 - shiftX;
        adjacentIndices(3, 1) = xIndex - shiftX;
        adjacentIndices(4, 1) = xIndex + 1 - shiftX;
    end
end

```

```

shiftY = 0;
if actualDistanceY > 0
    shiftY = 1;
end

adjacentIndices(1, 2) = yIndex + 1 - shiftY;
adjacentIndices(2, 2) = yIndex + 1 - shiftY;
adjacentIndices(3, 2) = yIndex - shiftY;
adjacentIndices(4, 2) = yIndex - shiftY;

adjacentPoints = [lons(adjacentIndices(:, 1))', lats(adjacentIndices(:, 2))'];

distances = pdist2(single(adjacentPoints), single(actualPoint), 'euclidean') ...
    * Singapore.DISTANCE_PER_DEGREE_MEAN;

distanceIsZeroIndex = find(distances == 0);
if distanceIsZeroIndex
    elevation = elevationMap.matrixInterpolatedFiltered(adjacentIndices(distanceIsZeroIndex, 2), ...
        adjacentIndices(distanceIsZeroIndex, 1));
else
    weights = distances / min(distances);
    weights = 1 ./ weights;
    weights = weights / sum(weights);

    elevations = diag(elevationMap.matrixInterpolatedFiltered(adjacentIndices(:, 2), ...
        adjacentIndices(:, 1)));
    elevation = sum(weights .* elevations);

    data{i}.altitude = single(data{i}.altitude);
end

elevation = single(elevation);

% Update value according to weighted average
data{i}.altitude(j) = elevation;
end
end
end

```

LISTING M.11: Altitude assignment

```

function [status, timeInterpolated] = getInterpolatedStatus(dataBooking)
%GETINTERPOLATEDBOOKINGDATA Interpolates status gained from a booking set.
%The booking set contains gaps where the taxi was not driven and also small
%gaps which do not correspond to the normal 3 minute frequency. This
%function targets to interpolate each values to a one-minute sequence to
%use it for histograms

TRANSFORM_TO_MINUTE_FACTOR = 24 * 60;
MIN_GAP_IN_MINUTES_FOR_LOG_OFF_INTERPOLATION = 30;

% The time we want to interpolate
ONE_MINUTE = datenum([0 1 0 0 1 0]);

% Get all status as Status objects
status = [];
status = [status; arrayfun(@(x) Status.getFromString(x), dataBooking.status)];

% Get all times of booking data
times = dataBooking.time;

gapIndices = find((diff(times) * TRANSFORM_TO_MINUTE_FACTOR) ...
    > MIN_GAP_IN_MINUTES_FOR_LOG_OFF_INTERPOLATION);

% Interpolate last known indx before gap as Log Off
status(gapIndices) = Status.LOG_OFF;

% Interpolate first known indx after gap as Log Off.
% This enables the interpolation to set the values in between to LOG OFF
status(gapIndices + 1) = Status.LOG_OFF;

% Do some cleaning of flawed time order of booking data
status(diff(dataBooking.time) <= 0) = [];
times(diff(dataBooking.time) <= 0) = [];

% Compute the targeting time which uses one minute sequence
timeInterpolated = dataBooking.time(1):ONE_MINUTE:dataBooking.time(:end);

% Teh ctual interpolation using the nearest neighbor
status = status(floor(interp1(times, 1:length(times), timeInterpolated)));

end

```

LISTING M.12: Interpolation of booking set

```

function [data, combinationIndices] = combineDatasets(data, dataBooking)
%COMBINEDATASETWINDOWED Combines logging with booking dataset by attaching logging
%dataset with status information. It uses time information from booking
%data set and searches robustly for an equivalent in the logging dataset.

L = evalin('base', 'L');

% Allow a band of 30 seconds for logging and booking data
SECONDS_TOLERANCE = 30;
TOLERANCE_OFFSET = datenum([0 1 0 0 0 SECONDS_TOLERANCE]);

combinationIndices = {};
combinationCounter = 0;
combinationCounterTotal = 0;
dataLengthCounter = 0;

L.trace('combineDatasets', ...
    sprintf('Combine booking with logging data with tolerance band of <%ds>', ...
        SECONDS_TOLERANCE));

if isempty(dataBooking)
    L.trace('combineDatasets', 'Booking data is empty, skipping combination');
    return;
end

% Setup combination indices
for i = 1:length(data)
    trip = data(i);
    combinationIndices(i, 1)(1:length(trip.time), 1) = 0;
end

for i = 1:length(data)
    trip = data(i);

    L.trace('combineDatasets', ...
        sprintf('Combining trip <%d>', i));

    % Find all indices of logging dataset where time of booking dataset
    % equals time in logging dataset
    indices = arrayfun(@(x) ...
        find((x >= trip.time - TOLERANCE_OFFSET)...
            & (x <= trip.time + TOLERANCE_OFFSET)),...
        dataBooking.time, ...
        'UniformOutput', false);

    % Get indices as vector values
    indices = cell2mat(indices);

    % Find all indices of booking dataset where time of logging dataset
    % equals time in booking dataset
    bookingIndices = arrayfun(@(x) ...
        find((x >= dataBooking.time - TOLERANCE_OFFSET)...
            & (x <= dataBooking.time + TOLERANCE_OFFSET)),...
        trip.time, ...
        'UniformOutput', false);

    % Get indices as vector values
    bookingIndices = cell2mat(bookingIndices);

    % Validate index results
    isCheckOk = checkCombineDataset(indices, bookingIndices, trip, dataBooking, false);
    if ~isCheckOk
        L.warn('combineDatasets', ...
            sprintf('Booking data does not fit to logging data at trip <%d>', i));
    end

    % Extract status strings from booking data
    statusStrings = dataBooking.status(bookingIndices);

    % Goes through all booking indices and apply its status to the trip
    for j = 1:length(bookingIndices)
        switch j
            case 1
                % If we are at the first index, fill start of trip up to
                % first index
                trip.status(1:indices(j)) = ...
                    Status.getFromStrings(statusStrings(j));
            case length(bookingIndices)
                % If we already reached the last statusString, add the same
                % status till the end of the trip
                trip.status(indices(j):length(trip.time)) = ...
                    Status.getFromStrings(statusStrings(j));
            otherwise
                % Fill gap between current and successor
                trip.status(indices(j):indices(j + 1)) = ...
                    Status.getFromStrings(statusStrings(j));
        end
        % Add the status to combinationIndices where we actually rely
        % had the information that the status has changes. This is
        % required for the trip cutting step.
        combinationIndices(i, 1)(indices(j), 1) = Status.getFromStrings(statusStrings(j));
    end
    combinationCounter = combinationCounter + length(unique(bookingIndices));
end

L.trace('combineDatasets', ...

```

```

    sprintf('Combined <#d/#d> booking data to logging data', ...
    combinationCounter, length(dataBooking.time));

% Some measurement vars
dataLengthCounter = dataLengthCounter + length(dataBooking.time);
combinationCounterTotal = combinationCounterTotal + combinationCounter;

L.trace('combineDatasets', ...
    sprintf('Total combined <#d/#d> booking data to logging data', ...
    combinationCounterTotal, dataLengthCounter));

end

```

LISTING M.13: Interpolation of booking set

```

function dataExtracted = extractTripsStatusBased(data, combinationIndices, taxistands, shallPlot)
%EXTRACTTRIPS Extract trips if there is a change in status. A status change
%indicates that a taxi trip has ended or is about to start or e.g. a break
%has started. Since each trip has its own status, this is a crucial step.

L = evalin('base', 'L');

if nargin < 4
    shallPlot = false;
end

dataExtracted = {};

for dataIndex = 1:length(data)
    trip = data(dataIndex);
    tripCombinationIndices = combinationIndices(dataIndex);

    % Get cut matrix based on status changes
    candidateCutPoints = getRelevantStatusChangeIndices(trip);

    % Check and estimate the right cut position in the area of all cut
    % points
    cutPoints = checkStatusCutPoint(trip, candidateCutPoints, tripCombinationIndices, taxistands, shallPlot);
    cutPoints = unique(cutPoints);

    edgeIndices = [0; cutPoints; length(trip.time)];

    L.trace('extractTripsStatusBased', sprintf('Trace <#d> is cut into <#d> trips', ...
    dataIndex, length(edgeIndices) - 1));

    % Cut trips and assign them to new data set
    dataExtracted = cutTrips(data, dataExtracted, dataIndex, edgeIndices);
end

% Apply set filters since it can happen that an extraction is less than a
% minimum of entries
dataExtracted = applyTripSetFilters(dataExtracted);

% Since we cut trips not necessarily at the change indices of a status
% change, but on a checked or estimated one, we need to convey status
% information from one trip to another.
for j = 1:length(dataExtracted)
    tripExtracted = dataExtracted{j};
    changeIndex = getRelevantStatusChangeIndices(tripExtracted);
    if ~isempty(changeIndex)
        tripExtracted.status(1:changeIndex(:end)) = tripExtracted.status(changeIndex(:end) + 1);
    end
end

L.trace('extractTripsTimeBased', sprintf('Got <#d> trips', length(dataExtracted)));

end

```

LISTING M.14: Status based trip extraction

```

function cutPoints = ...
    checkStatusCutPoint(trip, candidateCutPoints, tripCombinationIndices, taxistands, shallPlot)
%CHECKSTATUSCUTPOINT Takes a set of candidate cut points for a given trip.
%Those cut points should originate from a status changed trip extraction
%analysis. To estimate where the status change exactly occurred, thresholds
%methods are taken into account: Where did the taxi stop, where was a
%turn and which point was most distant to the direct route

if nargin < 5
    shallPlot = false;
end

cutPoints = [];
for i = 1:length(candidateCutPoints)

    % Find the right step back index gathered from the combination indices
    % array from combination step
    realStatusChangeIndices = find(diff(tripCombinationIndices));
    realStatusChangeIndices = realStatusChangeIndices(candidateCutPoints(i) ...

```

```

- realStatusChangeIndices > 0);

if length(realStatusChangeIndices) < 1
    realStatusChangeIndices = 1;
end
if length(realStatusChangeIndices) < 2
    realStatusChangeIndices = [1; realStatusChangeIndices];
end
tripStepBackIndex = floor(mean([realStatusChangeIndices(end), realStatusChangeIndices(end-1)]));

% Get the extraction of trip in specific window
tripExtraction = trip.getExtraction(tripStepBackIndex:candidateCutPoints(i));

% Call urn estimation algo
urnEstimationIndices = getUrnEstimationIndices(tripExtraction);

% Call stop estimation algo
stopEstimationIndices = getStopEstimationIndices(tripExtraction);

% Call most distance estimation algo
mostDistantEstimationIndex = getMostDistantEstimationIndices(tripExtraction);

% Include urn, and stop estimation into the fitting algorithm for the
% best cut point
estimatedCutPointExtraction = ...
    findBestFittingCutPoint(...
        tripExtraction, ...
        urnEstimationIndices, ...
        stopEstimationIndices, ...
        mostDistantEstimationIndex);

% Tailor cut Point to our indices of the whole trip
estimatedCutPoint = tripStepBackIndex + estimatedCutPointExtraction;

% Finally add it to our array
cutPoints = [cutPoints; estimatedCutPoint];

if shallPlot
    % Plots
    handle = figure;
    maxfig(handle, 1);
    hold on;
    plot(tripExtraction.lon(estimatedCutPointExtraction), ...
        tripExtraction.lat(estimatedCutPointExtraction), ...
        'c.', 'MarkerSize', 80, 'DisplayName', 'Best fitting cut point');
    plot(tripExtraction.lon(urnEstimationIndices), ...
        tripExtraction.lat(urnEstimationIndices), ...
        'y.', 'MarkerSize', 60, 'DisplayName', 'Urn candidates');
    plot(tripExtraction.lon(stopEstimationIndices), ...
        tripExtraction.lat(stopEstimationIndices), ...
        'm.', 'MarkerSize', 40, 'DisplayName', 'Stop candidates');
    plot(tripExtraction.lon(mostDistantEstimationIndex), ...
        tripExtraction.lat(mostDistantEstimationIndex), ...
        'k.', 'MarkerSize', 30, 'DisplayName', 'Most distant candidates');
    tripExtraction.plot();
    xlabel('Lon');
    ylabel('Lat');
    title('Status changed trip cut candidates');
    axis equal;
    legend show;
end
end
end

```

LISTING M.15: Validation of status change point

```

function shifts = extractShifts(data)
%EXTRACTSHIFTSBULK Extracts shifts from a set of trips contained in data by
%analysing reoccurring patterns

% A pause more than 3 hours is considered as a shift change
ONE_SHIFT_CHANGE_THRESHOLD = 3;
MIN_TRIPS_PER_SHIFT = 5;
MAX_TIME_BETWEEN_SHIFTS = ONE_SHIFT_CHANGE_THRESHOLD;
shifts = [];

% Find a common pattern for shift change for each data trip
[~, shiftChangeIndices, ~, subzoneChanges] = extractShiftChangeCandidates(data, false);

% Compute gaps between times
starts = cellfun(@(x) x.time(1), data);
ends = cellfun(@(x) x.time(1), data);

gaps = abs(ends(1:end-1) - starts(2:end)) * 24;

% Find the location of gaps where we can possible cut shifts
gapIndices = find(gaps >= ONE_SHIFT_CHANGE_THRESHOLD);
gapIndices = [0; gapIndices; length(data)];

for i = 1:length(gapIndices) - 1
    % Get the shift candidate by cutting at the gap index
    shiftCandidate = {data{gapIndices(i) + 1:gapIndices(i+1)}};
end

```

```

% Analyse cut shift and probably cut it in more detail by taken usual
% shift change behavior into account
shiftCandidates = analyseShiftCandidate(shiftCandidate, shiftchangesIndices, subzoneChanges);

shifts = [shifts; shiftCandidates];
end

% Throughout the preprocess it is possible that empty shift sets are create.
% Erase those.
shifts = shifts(~cellfun('isempty', shifts));

% Here we adjust the trips in order to have not very small shifts which are
% very aadjacent to other shifts
adjustedShift = [];
i = 1;
while i <= length(shifts)
    shift = shifts(i);
    if length(shift) < MIN_TRIPS_PER_SHIFT
        previousShift = [];
        nextShift = [];

        if i > 1
            previousShift = shifts(i - 1);
        end

        if i < length(shifts)
            nextShift = shifts(i + 1);
        end

        if ~isempty(previousShift)
            timeToPreviousShift = (shift{1}.time(1) - previousShift(end).time(;end)) * 24;
        else
            timeToPreviousShift = inf;
        end

        if ~isempty(nextShift)
            timeToNextShift = (nextShift{1}.time(1) - shift(end).time(;end)) * 24;
        else
            timeToNextShift = inf;
        end

        if timeToPreviousShift < MAX_TIME_BETWEEN_SHIFTS
            adjustedShift(end) = [previousShift; shift];
            i = i + 1;
        else
            if timeToNextShift < MAX_TIME_BETWEEN_SHIFTS
                adjustedShift(end + 1, 1) = [shift; nextShift];
                i = i + 2;
            else
                adjustedShift(end + 1, 1) = shift;
                i = i + 1;
            end
        end
    else
        adjustedShift(end + 1, 1) = shift;
        i = i + 1;
    end
end
shifts = adjustedShift;
end

```

LISTING M.16: Shift extraction

```

function connectivityMatrix = extractConnectivityMatrix(osmMap)
%EXTRACTCONNECTIVITYMATRIX Extracts the connectivity matrix from an osmMap
%object.

connectivityMatrix = sparse([]);

% Iterate over all ways
for i = 1:length(osmMap.ways)

    OsmParser.L.trace('connectivityMatrix', ['Extracting connectivity for way <', num2str(i), '>']);

    % Get all nodes contained in way
    nodeIds = osmMap.ways(i).nodeIds;

    % Iterate over all nodes in way
    for j = 1:length(nodeIds)
        if osmMap.node2IndexMap.isKey(num2str(nodeIds(j)))
            % Get current node index
            nodeIndex = osmMap.node2IndexMap(num2str(nodeIds(j)));
        else
            warning(['Key ', num2str(nodeIds(j)), ' not present in node2IndexMap']);
            continue;
        end

        % Check neighbourhood
        if j < length(nodeIds)
            % Get next node index
            if osmMap.node2IndexMap.isKey(num2str(nodeIds(j+1)))

```



```

    shouldPlot = false;
end

if nargin < 5
    shouldPlot = false;
end

for i = range
    try
        parfor_progress;
    catch
    end;

    % This array stores the graph of closest points and ways to backtrack
    % sequence in a later stage
    candidateGraph = {};
    closestNodesGraph = {};
    closestWaysGraph = {};

    % The Fs and Ft array contains the weights for each possible path in
    % spatial and temporal manner
    Fs = {};
    Ft = {};
    trip = data(i);

    L.trace('mapMatch', sprintf('Map match trip <%d/%d> ', i, length(data)));

    % Retrieve a set of point to be used for map matching. Avoid using each
    % point for computation enhancements
    pointIndicesToMatch = [1:MapMatching.STAKE_OF_USED_POINTS:length(trip.time), length(trip.time)];
    pointIndicesToMatch = unique(pointIndicesToMatch);

    for j = pointIndicesToMatch
        L.trace('mapMatch', sprintf('Map match point no <%d/%d> ', j, length(trip.time)));

        observedPoint = getObservedMeanPoint(trip, j, false);
        observedVelocity = median(trip.velocity(max(1, j - MapMatching.STAKE_OF_USED_POINTS):j));

        % Get location on grid
        [latIndex, lonIndex] = osmMapGridSection.getGridLocation(observedPoint);

        ways = [];
        % Start with a range of 1 to allow finding more adjacent ways
        range = 1;
        while length(ways) < MapMatching.CANDIDATE_POINTS
            range = range + 1;
            % Retrieve neighbouring indices
            indices = getAdjacentIndices(osmMapGridSection, latIndex, lonIndex, range);
            % Retrieve all possible ways
            ways = getAllWayCandidates(indices, osmMapGridSection);
        end;

        % Retrieve all candidate points projected on way
        [candidatePoints, closestNodes, closestWays] = ...
            getAllCandidatePoints(ways, observedPoint, osmMapGridSection, MapMatching.CANDIDATE_POINTS);

        % Add points to graph to backtrack matched sequence
        candidateGraph(end+1) = candidatePoints;
        closestNodesGraph(end+1) = closestNodes;
        closestWaysGraph(end+1) = closestWays;

        % Compute observation probability
        N = computeObservationProbability(observedPoint, candidatePoints);
        N = repmat(N, MapMatching.CANDIDATE_POINTS, 1);

        % Compute transmission probability
        if j > 1
            observedPointPrevious = ...
                [trip.lat(max(1, j-MapMatching.STAKE_OF_USED_POINTS)), ...
                 trip.lon(max(1, j-MapMatching.STAKE_OF_USED_POINTS))];

            % Apply the transmission prob. algorithm to gain the assumed
            % prob. that the route followed the path between previous
            % observed and current observed point
            V = computeTransmissionProbability( ...
                observedPointPrevious, observedPoint, ...
                candidatePointsPrevious, candidatePoints, ...
                closestNodesPrevious, closestNodes, ...
                osmMapGridSection, connectivityMatrix);

            % Compute temporal analysis function
            temporalDistance = temporalAnalysisFunction( ...
                osmMapGridSection, ...
                closestNodesPrevious, closestNodes, ...
                observedVelocityPrevious, observedVelocity);
        else
            % For the first point in j just use ones to only apply the
            % observation probability
            V = ones(MapMatching.CANDIDATE_POINTS, MapMatching.CANDIDATE_POINTS);
            temporalDistance = ones(MapMatching.CANDIDATE_POINTS, MapMatching.CANDIDATE_POINTS);
        end

        % Compute Spatial analysis function
        Fs(end+1) = N .* V;

        % Attach temporal analysis function value
        Ft(end+1) = temporalDistance;
    end
end

```

```

    % Store previous observed point for next transmission probability
    % for next point to matched
    candidatePointsPrevious = candidatePoints;
    closestNodesPrevious = closestNodes;
    observedVelocityPrevious = observedVelocity;
end

L.trace('mapMatch', 'Find best matching sequence ...');

% Compute best matching sequence
sequence = findMatchedSequence(Fs, Ft);

% Plot results
if shouldPlot
    maxfig(figure, 1);
    hold on;
    plot(trip.lon(pointIndicesToMatch), trip.lat(pointIndicesToMatch), '.b', 'MarkerSize', 60);
    plotMatchedSequence(osmMapGridSection, candidateGraph, closestNodesGraph, sequence, Fs);
    cellfun(@(x) plotWays(x, osmMapGridSection), closestWaysGraph);
end

% Attach sequenced ways ids and highway tags to points
matchedWayIds = attachTagsToPoints(osmMapGridSection, connectivityMatrix, ...
    trip, pointIndicesToMatch, ...
    closestWaysGraph, closestNodesGraph, sequence, shouldPlot);

% Conduct a post processing step which readjusted wrongly matched ways
postProcessMapMatching(osmMapGridSection, ...
    matchedWayIds, trip, shouldPlot);

if shouldPlot
    trip.visualize;
    hold on;
    plotMatchedWays(trip, osmMapGridSection);
end
end
end

```

LISTING M.18: Map-matching algorithm

```

function data = matchZonesEx(data, zones)
%MATCHZONES Matches each point of trip to a certain subzone
% id from zones array

L = evalin('base', 'L');

for i = 1:length(data)
    L.trace('matchZones', sprintf('Matching trip <%d/%d> to zones', i, length(data)));

    trip = data(i);
    point = [trip.lon, trip.lat];

    for k = 1:length(zones)
        in = inpoly(point, [zones{k}.Longitude, zones{k}.Latitude]);
        trip.subzone(in) = k;
    end

    % Adopted from database implementation
    % count consecutive number of 0 values
    min0 = 60;

    idSubZone = trip.subzone;
    idDataPoint = 1:length(trip.subzone);

    n0 = 0;
    i_assign = false(length(idSubZone), 1);

    for i = 1:length(idSubZone)
        if idSubZone(i) == 0
            n0 = n0 + 1;
        else
            if n0 > 0
                if n0 <= min0
                    i_assign(i-n0:i-1) = true;
                end
            end
            n0 = 0;
        end
    end

    if n0 > 0
        if n0 <= min0
            i_assign(i-n0+1:end) = true;
        end
    end

    % Check if interpolation shall be done
    if any(i_assign)

        % Remove values from original data
        id_in = double(idDataPoint(~i_assign));
    end
end

```

```

sz_in = double(idSubZone(~i_assign));

% Interpolate subzone ids
idSubZone_int = interp1(id_in, sz_in, idDataPoint, 'nearest', 'extrap');
trip.subzone = int16(idSubZone_int');
end
end
end

```

LISTING M.19: Matching of Subzones to sample points

```

function [Econs, drivePower, Distance, spCons] = ...
    CalculateEnergyConsumptionEVCharacteristics(dynamics, EVCharacteristics)
% =====
% Function: CalculateEnergyConsumption
% Autor: Pablo Lopez Hidalgo pablo.hidalgo@tum-create.edu.sg (inputs
% from Theresa Knoblauch), changes from Sascha Moecker
% Date: 30.07.2014
% Version: 1.10
% =====
% Calculate the energy consumption for one vehicle type
% Input: speed: 2 dimensional vector 1st column: Time [s]
%         2nd column speed values [m/s] the used speed in this file should
%         already be filtered
%         vehicle: vehicle is a struct containing the vehicle parameters m, f_r, c_w, A, lambda
%
% Output: Cumulated Energy Consumption of Vehicle, last value is also the
%         total energy consumed for one trip
%
% requires: File with driving cycle data
% =====

% Obtain values from uniform EV Characteristics
g = EVCharacteristics.g;
rho = EVCharacteristics.rho;
eta_dt = EVCharacteristics.etaMotor * EVCharacteristics.etaInverter ...
    * EVCharacteristics.etaTransmission;

eta_charge = EVCharacteristics.etaCharging;
eta_discharge = EVCharacteristics.etaDischarging;
P_AUX = EVCharacteristics.powerAux;

m = EVCharacteristics.mass;
fR = EVCharacteristics.tireRollingResistance;
cw = EVCharacteristics.dragCoefficient;
A = EVCharacteristics.frontalArea;
lambda = EVCharacteristics.rotationalMassFactor;

% Slope
time = dynamics(:, 1);
velocity = dynamics(:, 2);
alpha = dynamics(:, 3);

% Acceleration
a_grd = gradient(velocity, time);

% Power submodules
P_L = 1/2 * rho * cw * A .* velocity.^ 3;
P_k = m * lambda .* a_grd .* velocity;
P_r = m * g * fR .* velocity;
P_s = m * g .* sin(alpha) .* velocity;
Paux = ones(length(velocity),1) .* P_AUX;

% Drivetrain power
P_road = P_L + P_k + P_r + P_s;

% Efficiency consideration
P_road(P_road >= 0) = 1 / eta_dt .* P_road(P_road>=0);
P_road(P_road < 0) = eta_dt .* P_road(P_road<0);

% Drivetrain power distinguished by charge and discharge
drivePower(P_road >= 0,1) = (P_road(P_road>=0) + Paux(P_road >= 0)) ./ 1000 * 1 / eta_discharge;
drivePower(P_road < 0,1) = (P_road(P_road<0) - Paux(P_road < 0))./ 1000 * eta_charge;

Econs = cumsum(0.5 .* (drivePower(1;end-1, 1) + drivePower(2;end, 1)) .* (time(2;end, 1) ...
    - time(1;end-1, 1)) ./ 3600);
Econs = [0; Econs];

Distance = cumsum(0.5 .* (dynamics(2;end, 2) + dynamics(1;end-1, 2)) .* ((dynamics(2;end, 1) ...
    - dynamics(1;end-1, 1))) ./ 1000);
spCons = Econs(end) / Distance(end) * 100;
end

```

LISTING M.20: Backward Model implementation

```

function [consumedEnergy, energyPer100Km, distance] = ...
    computeEnergyTumEVModel(trip, EVCharacteristics)

```

```

%COMPUTEENERGY Prepares dataset and applies the vehicle model on
%velocity/altitude basis

% We have to apply a smottohing on our velocity vector since the TUM EV
% model does not deal with to high peaks in the v/t diagram
% tripPrepared.velocity = smooth(double(trip.velocity), SMOOTH_VALUE);

SMOOTH_VALUE = 5; %#ok<NASGU>

% Global variable declaration for SIMULINK
global dc;
global sp;

% Preload the system for inititalization
load_system('VehicleModel');

% Initiate Simulation Parameters
[dc, sp] = InitParameters(trip);

assignin('base', 'dc', dc);
assignin('base', 'sp', sp);

set_param('VehicleModel', 'InitFcn', '');

%Simulate to get Reference Values
sim('VehicleModel');

set_param('VehicleModel', 'InitFcn', 'StartSimulationScript');

% Format results to our known format
energyPer100Km = kWh100km.Data(;end);
consumedEnergy = consumedEnergyPerTrip.Data(;end);
distance = distanceTraveled.Data(;end);

end

```

LISTING M.21: Energy computation for TUM CREATE EV Model

```

function [consumedEnergy, energyPer100Km, distance] = ...
    computeEnergyForSections(osmMapGridSection, energyMap, linkMicroTrips)
%COMPUTEENERGYFORSTARTANDENDNODE Computes energy consumption based on
%established energy map

energyValues = [];
distanceValues = [];
for i = 1:length(linkMicroTrips)
    linkMicroTrip = linkMicroTrips(i);
    sectionId = mode(linkMicroTrip.way);
    section = osmMapGridSection.sectionMap(num2str(sectionId));

    if energyMap.isKey(num2str(sectionId));
        energyPer100kmMeaned = mean(energyMap(num2str(sectionId)));
        % Get from kWh per 100km to kWh consumed for this particalr
        % section
        energyPerSection = energyPer100kmMeaned / (100 * 1000) * section.length;
    else
        energyPerSection = getDefaultEnergyValue(section);
    end

    energyValues = [energyValues; energyPerSection];
    distanceValues = [distanceValues; section.length];
end

consumedEnergy = sum(energyValues);

distance = sum(distanceValues) / 1000;
energyPer100Km = 100 / distance * consumedEnergy;

```

LISTING M.22: Energy computation for road segment based energy map approach

```

function [consumedEnergy, energyPer100Km, distance] = ...
    computeEnergyDrivingShare(trip, EVCharacteristics, drivingShareParameters)
%COMPUTEENERGYDIRVINGSHARE Computes energy consumption based on driving
%share and average acc and dec for a certain trip

time = (trip.time(;end) - trip.time(1)) * 24 * 60 * 60;

velocity = mean(trip.velocity);
velocityCubicMean = (mean(trip.velocity.^ 3)) ^ (1/3);
gradient = mean(methodIncline((trip)));

% Get share and acc dec values for vertain velocity
drivingShares = drivingShareParameters.getDrivingShare(velocity);
accDecValues = drivingShareParameters.getAccDecValue(velocity);

acceleration = accDecValues(1);

```

```

deceleration = accDecValues(2);

% Compute all power values for different driving force contribution
eRoll = getERoll(velocity, EVCharacteristics);
eAir = getEAir(velocity, EVCharacteristics);
eAcc = getEAcc(velocity, acceleration, EVCharacteristics);
eDec = getEDec(velocity, deceleration, EVCharacteristics);
eGrade = getEGrade(velocity, gradient, EVCharacteristics);
eAux = getEAux(EVCharacteristics);

% Retrieve the weighted energy value for our driving situation
energy = getTotalEnergyShareBased(drivingShares, eRoll, eAir, eAcc, eDec, eGrade, eAux) * time;

distance = cell2mat(computeTripLengthVelocityBased({trip})) / 1000;
consumedEnergy = energy;
energyPer100Km = energy / distance * 100;
[~, b, ~] = computeEnergyPabloModel(trip, EVCharacteristics);

end

```

LISTING M.23: Energy computation for generic driving share approach

```

function [consumedEnergy, energyPer100Km, distance] = ...
    computeEnergyDrivingShareForSections(linkMicroTrips, shareParameterMap, EVCharacteristics)
%COMPUTEENERGYDRIVINGSHARE Compute energy consumption based on driving
%share and average acc and dec for a certain trip

distance = 0;
consumedEnergy = 0;

for i = 1:length(linkMicroTrips)
    trip = linkMicroTrips{i};

    wayId = mode(trip.way);

    if shareParameterMap.isKey(num2str(wayId))
        drivingShareParameters = shareParameterMap(num2str(wayId));
        [energy, ~, ~] = ...
            computeEnergyDrivingShare(trip, EVCharacteristics, drivingShareParameters);
    else
        section.length = cell2mat(computeTripLengthVelocityBased({trip}));
        energy = getDefaultEnergyValue(section);
    end

    distance = distance + cell2mat(computeTripLengthVelocityBased({trip})) / 1000;
    consumedEnergy = consumedEnergy + energy;
end

energyPer100Km = consumedEnergy / distance * 100;

end

```

LISTING M.24: Energy computation for road segment based driving share approach

```

function mapAttachement = updateMapAttachementValues(microTrip, mapAttachement, EVCharacteristics)
%GETMAPATTACHEMENTVALUES Updates map attachements in form of average energy
%values for energy map or driving shares and dynamics in case of driving
%share approach

MAX_ENERGY_VALUE = EVCharacteristics.maxPower;

% The actual application of our car model
% [~, energyPer100Km, ~] = computeEnergyStaticModel(vehicleCharacterisitcs, trip, wayRange);
[~, energyPer100Km, ~] = computeEnergyPabloModel(microTrip, EVCharacteristics);

% Continue to next way if energy exceed limits. This can be caused
% by a unregular energy apttern.
if abs(energyPer100Km) > MAX_ENERGY_VALUE
    return;
end

% Prepare the key
key = num2str(mode(microTrip.way));

% Continue with next key if we have a zero as key. This case means,
% we haven't mapped a way to the point
if strcmp(key, '0')
    return;
end

% The average speed
averageSpeed = mean(microTrip.velocity);

acceleration = getAcceleration(double(diff(microTrip.velocity)), diff(microTrip.time));
accelerations = acceleration(acceleration >= AccDecValue.ACCELERATION_THRESHOLD);
decelerations = acceleration(acceleration <= AccDecValue.DECELERATION_THRESHOLD);

```

```

if isempty(accelerations)
    accelerations = 0;
end

if isempty(decelerations)
    decelerations = 0;
end

meanAcc = mean(accelerations);
meanDec = mean(decelerations);

% drivingShareParameters = extractShareParameters({trip}, false);
shares = extractDrivingShareValues(microTrip);
drivingShareMapValue = [averageSpeed, meanAcc, meanDec, shares];

% Attach gathered values and keys to our maps
mapAttachement.energyWayMap = addToMapContainer(mapAttachement.energyWayMap, ...
    key, energyPer100Km);
mapAttachement.averageSpeedMap = addToMapContainer(mapAttachement.averageSpeedMap, ...
    key, averageSpeed);
mapAttachement.drivingShareMap = addToMapContainer(mapAttachement.drivingShareMap, ...
    key, drivingShareMapValue);

timeZoneSet = getTimeZoneSpecificSet({microTrip});
timePeriodIndex = find(~cellfun(@isempty,timeZoneSet));
switch timePeriodIndex
    case PeakHours.PEAK_AM_INDEX
        mapAttachement.energyWayMapPeakAM = addToMapContainer(mapAttachement.energyWayMapPeakAM, ...
            key, energyPer100Km);
        mapAttachement.averageSpeedMapPeakAM = addToMapContainer(mapAttachement.averageSpeedMapPeakAM, ...
            key, averageSpeed);
        mapAttachement.drivingShareMapPeakAM = addToMapContainer(mapAttachement.drivingShareMapPeakAM, ...
            key, drivingShareMapValue);
    case PeakHours.DAY_INDEX
        mapAttachement.energyWayMapDay = addToMapContainer(mapAttachement.energyWayMapDay, ...
            key, energyPer100Km);
        mapAttachement.averageSpeedMapDay = addToMapContainer(mapAttachement.averageSpeedMapDay, ...
            key, averageSpeed);
        mapAttachement.drivingShareMapDay = addToMapContainer(mapAttachement.drivingShareMapDay, ...
            key, drivingShareMapValue);
    case PeakHours.PEAK_PM_INDEX
        mapAttachement.energyWayMapPeakPM = addToMapContainer(mapAttachement.energyWayMapPeakPM, ...
            key, energyPer100Km);
        mapAttachement.averageSpeedMapPeakPM = addToMapContainer(mapAttachement.averageSpeedMapPeakPM, ...
            key, averageSpeed);
        mapAttachement.drivingShareMapPeakPM = addToMapContainer(mapAttachement.drivingShareMapPeakPM, ...
            key, drivingShareMapValue);
    case PeakHours.NIGHT_INDEX
        mapAttachement.energyWayMapNight = addToMapContainer(mapAttachement.energyWayMapNight, ...
            key, energyPer100Km);
        mapAttachement.averageSpeedMapNight = addToMapContainer(mapAttachement.averageSpeedMapNight, ...
            key, averageSpeed);
        mapAttachement.drivingShareMapNight = addToMapContainer(mapAttachement.drivingShareMapNight, ...
            key, drivingShareMapValue);
end
end

```

LISTING M.25: Conceptualization of road segment attachments

```

function featureSet = extractFeaturesFromMicroTrip(microTrip, EVCharacteristics)
%EXTRACTFEATURESFROMMICROTRIP Feature extraction from micro trips based on
%established 16 features

STOP_TRESHHOLD = 1;

stopIndices = find(microTrip.velocity < STOP_TRESHHOLD);
velocitiesWithoutStops = microTrip.velocity;

if ~isempty(stopIndices)
    stopCuts = find(diff(stopIndices) > 1);
    stopCuts = [0; stopCuts; length(stopIndices)];
    stopLength = [];

    for i = 1:length(stopCuts) - 1
        stopLength = [stopLength; stopCuts(i + 1) - stopCuts(i) + 1 - 1];
    end

    stopNumber = length(find(diff(stopIndices) > 1)) + 1;
    velocitiesWithoutStops(stopIndices) = [];
else
    stopNumber = 0;
    stopLength = 0;
end

shares = extractDrivingShareValues(microTrip);

celocityCmp = getVelocity(diff(microTrip.lat), diff(microTrip.lon), diff(microTrip.time));
accelerationCmp = getAcceleration(diff(celocityCmp), diff(microTrip.time(1:end-1)));
acceleration = accelerationCmp;

% acceleration = getAcceleration(diff(microTrip.velocity), diff(microTrip.time));

```

```
[~, energyPer100Km, ~] = computeEnergyPabloModel(microTrip, EVCharacteristics);

% Create an empty feature set instance
featureSet = FeatureSet;

featureSet.distance = cell2mat(computeTriplengthVelocityBased({microTrip})) / 1000;

featureSet.meanVelocityTotal = mean(microTrip.velocity);
featureSet.meanVelocityExcludingStops = mean(velocitiesWithoutStops);

featureSet.maxVelocity = max(microTrip.velocity);
featureSet.minVelocity = min(microTrip.velocity);

featureSet.shareIdle = shares(DrivingShare.IDLE);
featureSet.shareCruise = shares(DrivingShare.CRUISE);
featureSet.shareAcc = shares(DrivingShare.ACCELERATION);
featureSet.shareDec = shares(DrivingShare.DECELERATION);

featureSet.meanAcceleration = mean(acceleration(acceleration > 0.15));
featureSet.meanDeceleration = mean(acceleration(acceleration < -0.15));

featureSet.stopRate = stopNumber / featureSet.distance;
featureSet.meanStopTime = mean(stopLength);

featureSet.modeHighwayType = Highway.getBestMatchingCluster(microTrip.highway);

featureSet.timePeriod = getTimeZoneForTrip( microTrip);
featureSet.energyConsumption = energyPer100Km;

end
```

LISTING M.26: Driving feature extraction for instant approach

```
function drivingShare = extractDrivingShare(microTrips, shallPlot)
%EXTRACT Takes a set of trips or micro trip in data and detects the driving
%shares of each

L = evalin('base', 'L');

drivingShare = DrivingShare;

% Setup a share matrix consisting of MAX_SPEED number of cells
[shares{1:DrivingShare.MAX_SPEED}] = deal([]);
shares = shares';

% Iterate over all trips in data
for i = 1:length(microTrips)
    if ~mod(i, 100) || i == 1
        L.trace('extractDrivingShare', sprintf('Extracting driving share for micro trip <%d>', i));
    end
    microTrip = microTrips(i);

    % Build a trip regarded diving type array
    tripDrivingTypes = [];
    for j = 1:length(microTrip.time) - 1
        drivingType = DrivingShare.getDrivingType(microTrip.velocity(j), microTrip.velocity(j + 1));
        tripDrivingTypes(end+1) = drivingType;
    end

    % Get the mean velocity for this trip. Floor it, so we actually have
    % the velocity + 1. Thus, index 1 equals velocity of zero
    meanedVelocity = floor(mean(microTrip.velocity)) + 1;

    % Get the histogram of driving shares for our trip
    drivingShareHistogram = histc(tripDrivingTypes, DrivingShare.DRIVING_STATES);

    % Assign the gathered histogram normalized to the gathered mean
    % valocity
    shares{meanedVelocity, 1} = ...
        [shares{meanedVelocity, 1}; ...
        drivingShareHistogram / sum(drivingShareHistogram)];
end

% Take all share values for each velocity and mean the containing share
% distribution
sharesMeaned = cellfun(@(x) mean(x, 1), shares, 'UniformOutput', false);

idles = getTimeSharesForDrivingType(shares, DrivingShare.IDLE);
cruises = getTimeSharesForDrivingType(shares, DrivingShare.CRUISE);
accs = getTimeSharesForDrivingType(shares, DrivingShare.ACCELERATION);
decs = getTimeSharesForDrivingType(shares, DrivingShare.DECELERATION);

idlesMeaned = getTimeSharesForDrivingType(sharesMeaned, DrivingShare.IDLE);
cruisesMeaned = getTimeSharesForDrivingType(sharesMeaned, DrivingShare.CRUISE);
accsMeaned = getTimeSharesForDrivingType(sharesMeaned, DrivingShare.ACCELERATION);
decsMeaned = getTimeSharesForDrivingType(sharesMeaned, DrivingShare.DECELERATION);

idles = idlesMeaned;
cruises = cruisesMeaned;
accs = accsMeaned;
decs = decsMeaned;

% Compute the fitting curve for idle times
```

```

[drivingShare.idleParameters, ~] = ...
    lsqcurvefit(DrivingShare.IDLE_FUNCTION, DrivingShare.IDLE_INIT_PARAM, ...
    idles(:, 1), idles(:, 2), ...
    DrivingShare.IDLE_LB, DrivingShare.IDLE_UB);

xIdles = cruises(:, 1);
fitIdlesCurve = DrivingShare.IDLE_FUNCTION(drivingShare.idleParameters, xIdles);

% Compute the fitting curve for crusing times
[drivingShare.cruiseParameters, ~] = ...
    lsqcurvefit(DrivingShare.CRUISE_FUNCTION, DrivingShare.CRUISE_INIT_PARAM, ...
    cruises(:, 1), cruises(:, 2), ...
    DrivingShare.CRUISE_LB, DrivingShare.CRUISE_UB);

xCruise = cruises(:, 1);
fitCruiseCurve = DrivingShare.CRUISE_FUNCTION(drivingShare.cruiseParameters, xCruise);

% Compute the fitting curve for acceleration times
[drivingShare.accParameters, ~] = ...
    lsqcurvefit(DrivingShare.ACC_FUNCTION, DrivingShare.ACC_INIT_PARAM, ...
    accs(:, 1), accs(:, 2), ...
    DrivingShare.ACC_LB, DrivingShare.ACC_UB);

xAccs = accs(:, 1);
fitAccsCurve = DrivingShare.ACC_FUNCTION(drivingShare.accParameters, xAccs); %ok<*NASGU>

% Compute the fitting curve for deceleration times
[drivingShare.decParameters, ~] = ...
    lsqcurvefit(DrivingShare.DEC_FUNCTION, DrivingShare.DEC_INIT_PARAM, ...
    decs(:, 1), decs(:, 2), ...
    DrivingShare.DEC_LB, DrivingShare.DEC_UB);

xDecs = decs(:, 1);
fitDecsCurve = DrivingShare.DEC_FUNCTION(drivingShare.decParameters, xDecs);

if shallPlot
    % Plot the results
    maxfig(figure, 1);
    hold on;

    % First plot each meaned datapoint for driving type distribution
    plot(idles(:, 1), idles(:, 2), '.', ...
        cruises(:, 1), cruises(:, 2), 'x', ...
        accs(:, 1), accs(:, 2), 'o', ...
        decs(:, 1), decs(:, 2), '*');
    % Plot the fitting curves in same diagram
    % plot(xIdles, fitIdlesCurve, xCruise, fitCruiseCurve, xAccs, fitAccsCurve, xDecs, fitDecsCurve);

    xValues = 0:120;
    crvIdle = DrivingShare.IDLE_FUNCTION(drivingShare.idleParameters, xValues);
    crvCruise = DrivingShare.CRUISE_FUNCTION(drivingShare.cruiseParameters, xValues);
    crvAcc = DrivingShare.ACC_FUNCTION(drivingShare.accParameters, xValues);
    crvDec = DrivingShare.DEC_FUNCTION(drivingShare.decParameters, xValues);

    plot(xValues, crvIdle, xValues, crvCruise, xValues, crvAcc, xValues, crvDec);
    ylim([0,1]);

    title('Driving type shares');
    xlabel('Speed [km/h]');
    ylabel('Share [%]');
    grid on;

    legend('Idle times', 'Cruise times', 'Acceleration times', 'Deceleration times', ...
        'Idle share function', 'Cruise share function', 'Acceleration share function', ...
        'Deceleration share function');
end
end

```

LISTING M.27: Driving share extraction for driving share approach

```

function accDecValue = extractAccDecValue(microTrips, shallPlot)
%EXTRACTACCDECSHARE Takes a set of trips or micro trip in data and detects
%the dynamics shares of each

L = evalin('base', 'L');

accDecValue = AccDecValue;

% Setup a share matrix consisting of MAX_SPEED number of cells
[accDecValues{1:AccDecValue.MAX_SPEED}] = deal([]);
accDecValues = accDecValues';

% Iterate over all trips in data
for i = 1:length(microTrips)
    if ~mod(i, 100) || i == 1
        L.trace('extractAccDecShare', sprintf('Extracting acceleration share for micro trip <%d>', i));
    end

    microTrip = microTrips(i);
    meanedVelocity = floor(mean(microTrip.velocity)) + 1;

```



```

% Get all accelerations and extract those accs and decs which exceed
% the deinfed threshold
acceleration = getAcceleration(double(diff(microTrip.velocity)), diff(microTrip.time));
accelerations = acceleration(acceleration >= AccDecValue.ACCELERATION_THRESHOLD);
decelerations = acceleration(acceleration <= AccDecValue.DECELERATION_THRESHOLD);

if isempty(accelerations) || isempty(decelerations)
    continue;
end

meanAcc = mean(accelerations);
meanDec = mean(decelerations);

accDecValues{meanedVelocity, 1} = ...
    [accDecValues{meanedVelocity, 1}; ...
    meanAcc, meanDec];
end

% Take all share values for each velocity and mean the containing share
% distribution
accDecValuesMeaned = cellfun(@(x) mean(x, 1), accDecValues, 'UniformOutput', false);

accs = getTimeSharesForDrivingType(accDecValues, 1);
decs = getTimeSharesForDrivingType(accDecValues, 2);

accsMeaned = getTimeSharesForDrivingType(accDecValuesMeaned, 1);
decsMeaned = getTimeSharesForDrivingType(accDecValuesMeaned, 2);

accs = accsMeaned;
decs = decsMeaned;

% Compute the fitting curve for acceleration times
[accDecValue.accParameters, ~] = ...
    lsqcurvefit(AccDecValue.ACC_FUNCTION, AccDecValue.ACC_INIT_PARAM, ...
    accs(:, 1), accs(:, 2), ...
    AccDecValue.ACC_LB, AccDecValue.ACC_UB);

xAccs = accs(:, 1);
fitAccsCurve = AccDecValue.ACC_FUNCTION(accDecValue.accParameters, xAccs); %#ok<NASGU>

% Compute the fitting curve for deceleration times
[accDecValue.decParameters, ~] = ...
    lsqcurvefit(AccDecValue.DEC_FUNCTION, AccDecValue.DEC_INIT_PARAM, ...
    decs(:, 1), decs(:, 2), ...
    AccDecValue.DEC_LB, AccDecValue.DEC_UB);

xDecs = decs(:, 1);
fitDecsCurve = AccDecValue.DEC_FUNCTION(accDecValue.decParameters, xDecs);

if shallPlot
    % Plot the results
    maxfig(figure, 1);
    hold on;

    plot(accs(:, 1), accs(:, 2), '.', ...
        decs(:, 1), decs(:, 2), 'x');
    % Plot the fitting curves in same diagram
    % plot(xAccs, fitAccsCurve, xDecs, fitDecsCurve);

    xValues = 0:120;
    crvAcc = AccDecValue.ACC_FUNCTION(accDecValue.accParameters, xValues);
    crvDec = AccDecValue.DEC_FUNCTION(accDecValue.decParameters, xValues);
    plot(xValues, crvAcc, xValues, crvDec);

    ylim([-1,1]);

    title('Acceleration Function');
    xlabel('Speed [km/h]');
    ylabel('Acceleration [m/s^2]');
    grid on;

    legend('Acceleration values', 'Deceleration values', ...
        'Acceleration function', 'Deceleration function');
end
end
end

```

LISTING M.28: Dynamics extraction for driving share approach

Bibliography

- [1] TUM CREATE. Master Thesis proposal: Project: Driving Profile Analysis based on GPS Trajectories, 1/10/2013. URL http://www.tum-create.edu.sg/viewdownload.aspx?file=jobs/pdf&ofile=5026_641.pdf.
- [2] Taxis In Singapore. Taxi Stands in CBD. URL <http://www.taxisingapore.com/taxi-stands/>.
- [3] Ricardo Maia, Marco Silva, Rui Araújo, and Urbano Nunes. Electric vehicle simulator for energy consumption studies in electric mobility systems. In *Integrated and Sustainable Transportation System (FISTS), 2011 IEEE Forum on*, pages 227–232, 2011.
- [4] Eric Sortomme and Mohamed A. El-Sharkawi. Optimal charging strategies for unidirectional vehicle-to-grid. *Smart Grid, IEEE Transactions on*, 2(1):131–138, 2011.
- [5] TUM CREATE. EVA by TUM CREATE – Electric Taxi for Tropical Megacities: Technical Data, 21/9/2014. URL <http://www.eva-taxi.sg/technical-data.php>.
- [6] LTA Singapore. Taxi Availability Results-Aug 2014 (Pass-Fail), 2014. URL http://www.lta.gov.sg/content/dam/ltaweb/corp/PublicTransport/files/TA_Results.pdf.
- [7] LTA Singapore. Taxi Info for LTA Website 2014, . URL http://www.lta.gov.sg/content/dam/ltaweb/corp/PublicationsResearch/files/FactsandFigures/taxi_info_2014.pdf.
- [8] LTA Singapore. Taxi Info for LTA Website 2013. . URL http://www.lta.gov.sg/content/dam/ltaweb/corp/PublicationsResearch/files/FactsandFigures/taxi_info_2013.pdf.
- [9] U.S. Government. GPS.gov: GPS Overview: Official U.S. Government information about the Global Positioning System (GPS) and related topics. URL <http://www.gps.gov/systems/gps/>.
- [10] Guochang Xu. *GPS: Theory, algorithms, and applications*. Springer, Berlin and New York, 2nd ed edition, 2007. ISBN 978-3-540-72714-9.
- [11] Jon Froehlich and John Krumm. Route prediction from trip observations, 2008.
- [12] Michael Hörner. Entwicklung einer Software zur Erstellung einer räumlich-zeitlichen Energie-Bedarfs-Karte für Elektrofahrzeuge in Singapur, 2013.

- [13] Jae-seok Yang, Seung-pil Kang, and Kyung-soo Chon. The map matching algorithm of GPS data with relatively long polling time intervals. *Journal of the Eastern Asia Society for Transportation Studies*, 6:2561–2573, 2005.
- [14] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. Map-matching for low-sampling-rate GPS trajectories. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 352–361, 2009.
- [15] Barlow, Tim J and Latham, S and McCrae, IS and Boulter, PG. *A reference book of driving cycles for use in the measurement of road vehicle emissions*. 2009.
- [16] Fabian Viets. Automatisierte Erstellung von Fahrzyklen: Master Thesis, 2013.
- [17] Birgit Pattberg. DLR - Institute of Transportation Systems - SUMO – Simulation of Urban Mobility, 2014. URL http://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/.
- [18] Ricardo Machado Maia, Marco Silva, Urbano Nunes. Electric Vehicle Simulator for Energy Consumption Studies in Electric Mobility Systems. 2011.
- [19] Karen L. Butler, Mehrdad Ehsani, and Preyas Kamath. A Matlab-based modeling and simulation package for electric and hybrid electric vehicle design. *Vehicular Technology, IEEE Transactions on*, 48(6):1770–1778, 1999.
- [20] Steve Miller. Hybrid-Electric Vehicle Model in Simulink - File Exchange - MATLAB Central, 22/9/2014. URL <http://www.mathworks.com/matlabcentral/fileexchange/28441-hybrid-electric-vehicle-model-in-simulink>.
- [21] A. Florescu, H. Turker, S. Bacha, and E. Vinot. Energy management system for hybrid electric vehicle: Real-time validation of the VEHLIB dedicated library. In *Vehicle Power and Propulsion Conference (VPPC), 2011 IEEE*, pages 1–6, 2011.
- [22] M. Zallinger, J. Tate, and S. Hausberger. An instantaneous emission model for the passenger car fleet. In *16th International Transport and Air Pollution Congress*, 2008.
- [23] Kanok Boriboonsomsin, Matthew J. Barth, Weihua Zhu, and Alexander Vu. Eco-Routing Navigation System Based on Multisource Historical and Real-Time Traffic Information. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1694–1704, 2012. ISSN 1524-9050. doi: 10.1109/TITS.2012.2204051.
- [24] Matthew Barth, Feng An, Theodore Younglove, C. Levine, G. Scora, Marc Ross, and Thomas Wenzel. *Development of a comprehensive modal emissions model*. National Cooperative Highway Research Program, Transportation Research Board of the National Academies, 2000.
- [25] Stefan Grubwinkler, Maria Kugler, and Markus Lienkamp. A system for cloud-based deviation prediction of propulsion energy consumption for EVs. In *Vehicular Electronics and Safety (ICVES), 2013 IEEE International Conference on*, pages 99–104, 2013.
- [26] Karin Kraschl-Hirschmann and Martin Fellendorf. Estimating energy consumption for routing algorithms. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 258–263, 2012.

-
- [27] TUM CREATE. EVA by TUM CREATE – Electric Taxi for Tropical Megacities: Home, 21/9/2014. URL <http://www.eva-taxi.sg/index.php>.
- [28] Vicory Technology Co., Ltd. Columbus V-990 Features, 4/3/2013. URL http://www.cbgps.com/v990/index_en.htm.
- [29] Reinhard Sellmair. Anforderungsanalyse für einen elektrischen Taxibetrieb: Diploma Thesis, 2011.
- [30] Google Developers. Static Maps API V2 Developer Guide - Google Maps Image APIs — Google Developers, 7/10/2014. URL <https://developers.google.com/maps/documentation/staticmaps/>.
- [31] Michael Bichlmeier. Hybride Antriebe im Automobilbereich. 2011.
- [32] OSM. OpenStreetMap - About page, 2014. URL <http://www.openstreetmap.org/about>.
- [33] OSM. Stats - OpenStreetMap Wiki, 2014. URL <http://wiki.openstreetmap.org/wiki/Stats>.
- [34] Mathworks. File Exchange - MATLAB Central, 2014. URL <http://www.mathworks.com/matlabcentral/fileexchange/>.
- [35] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, et al. *Introduction to algorithms*, volume 2. MIT press Cambridge, 2001.
- [36] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
- [37] NIIT GIS Pte. Ltd. Singapore Government Data, 2014. URL <http://data.gov.sg/home.aspx>.
- [38] Land Transport Authority. MyTransport.SG:Data Mall: Traffic Related Data, 24/9/2014. URL http://www.mytransport.sg/content/mytransport/home/dataMall.html#Traffic_Related.
- [39] Department of Statistics Singapore. Statistics Singapore - Geospatial Data, 2014. URL http://www.singstat.gov.sg/statistics/browse_by_theme/geospatial.html.
- [40] Google Developers. The Google Elevation API - Google Maps API Web Services — Google Developers, 23/9/2014. URL <https://developers.google.com/maps/documentation/elevation/>.
- [41] Google Developers. Google Developers Console, 6/10/2014. URL <https://console.developers.google.com/project>.
- [42] Roger W. Sinnott. Virtues of the Haversine. *Sky and telescope*, 68:158, 1984.
- [43] Chris Veness and 2002-2005. GIS FAQ Q5.1: Great circle distance between 2 points, 27/7/2014. URL <http://www.movable-type.co.uk/scripts/gis-faq-5.1.html>.

- [44] Kenneth L. Judd. *Numerical methods in economics*. MIT, Cambridge, Mass., 1998. ISBN 9780262100717.
- [45] Frederick N. Fritsch and Ralph E. Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238–246, 1980.
- [46] Edward J. Krakiwsky, Clyde B. Harris, and Richard V. C. Wong. A Kalman filter for integrating dead reckoning, map matching and GPS positioning. In *Position Location and Navigation Symposium, 1988. Record. Navigation into the 21st Century. IEEE PLANS'88.*, IEEE, pages 39–46, 1988.
- [47] W. T. Hung, H. Y. Tong, C. P. Lee, K. Ha, and L. Y. Pao. Development of a practical driving cycle construction methodology: A case study in Hong Kong. *Transportation Research Part D: Transport and Environment*, 12(2):115–128, 2007. ISSN 13619209.
- [48] Mathworks. k-means clustering - MATLAB kmeans. URL <http://www.mathworks.com/help/stats/kmeans.html>.
- [49] Volker Schwarzer and Reza Ghorbani. Drive cycle generation for design optimization of electric vehicles. *Vehicular Technology, IEEE Transactions on*, 62(1):89–97, 2013.
- [50] Michel Andre. Real-world driving cycles for measuring cars pollutant emissions—Part A: The ARTEMIS European driving cycles. *Report Inrets-LTE*, 411:97, 2004.
- [51] Department of Statistics Singapore. Yearbook of Statistics, 2014. URL http://www.singstat.gov.sg/publications/publications_and_papers/reference/yearbook_2014/yos2014.pdf.
- [52] LTA Singapore. Transport Statistics in Brief. 2013.
- [53] LTA Singapore. LTA Statistics in Brief: Facts and Figures, . URL http://www.lta.gov.sg/content/dam/ltaweb/corp/PublicationsResearch/files/FactsandFigures/Stats_in_Brief_2013.pdf.
- [54] Fares Maximilian Mrad Agua. Electric Vehicle Modeling and Simulation: Bachelor Thesis, 2012.
- [55] R. Farrington and J. Rugh. Impact of vehicle air-conditioning on fuel economy, tailpipe emissions, and electric vehicle range. In *Earth Technologies Forum*, 2000.
- [56] National Environment Agency. Weather Statistics 1869-2013, 22/9/2014. URL <http://app2.nea.gov.sg/weather-climate/climate-information/weather-statistics>.
- [57] Andreas Jossen and Wolfgang Weydanz. *Moderne Akkumulatoren richtig einsetzen: 36 Tabellen*. Ubooks, Neusäß, 1. Aufl edition, 2006. ISBN 978-3939359111.
- [58] OSM. Key:highway - OpenStreetMap Wiki, 26/9/2014. URL <http://wiki.openstreetmap.org/wiki/Key:highway>.
- [59] Open Street Map. Key:highway - OpenStreetMap Wiki: Good practice for highway tag usage, 30/10/2014. URL <http://wiki.openstreetmap.org/wiki/Key:highway>.

-
- [60] Mathworks Documentation. Solve nonlinear curve-fitting (data-fitting) problems in least-squares sense - MATLAB lsqcurvefit, 2/10/2014. URL <http://www.mathworks.com/help/optim/ug/lsqcurvefit.html?refresh=true>.
- [61] U.S. Department of Energy. Energy Efficiency & Renewable Energy — Compare Side-by-Side. URL <http://www.fueleconomy.gov/feg/Find.do?action=sbs&id=35207&id=35279>.
- [62] Stackoverflow. Matlab - Value of Whisker in boxplot for 2 sigma coverage. URL <http://stackoverflow.com/questions/11698876/value-of-whisker-in-boxplot-for-99-7-coverage>.
- [63] Wikipedia. Boxplot vs PDF - Box plot - Wikipedia, the free encyclopedia, 10/10/2014. URL <http://en.wikipedia.org/w/index.php?oldid=617605869>.
- [64] Urban Redevelopment Authority. Viewing Planning Boundaries, 19/9/2014. URL https://www.ura.gov.sg/uramaps/?config=config_preopen.xml&preopen=Planning%20Boundaries&pbIndex=2.