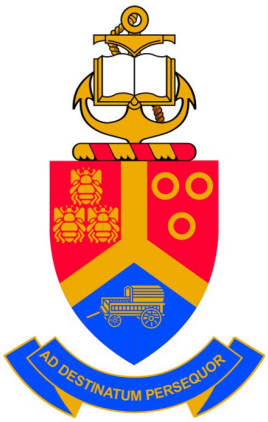

University of Pretoria

Vehicle Stability Analysis of Transport Latencies and Dropped Packets on Wireless Communication of an Off-site Steering Controller

by

Rory Douglas Bennett

Submitted in partial fulfilment of the requirements for the degree

Master of Engineering
(Mechanical Engineering)

in the Faculty of

Engineering, Built Environment, and Information Technology (EBIT)

at the

University of Pretoria,
Pretoria

30 August 2019

Summary

Title:	Vehicle Stability Analysis of Transport Latencies and Dropped Packets on Wireless Communication of an Off-site Steering Controller
Author:	Rory Douglas Bennett
Study Leader:	Dr T.R. Botha
Department:	Mechanical and Aeronautical Engineering, University of Pretoria
Degree:	Masters in Mechanical Engineering

With autonomous vehicles being introduced around the world the possibility of controlling these vehicles from off-site locations presents itself as an opportunity to increase occupant/user safety as well as system efficiency. This applies not only to passenger vehicles but to vehicles in the industrial sector as well. To safely implement off-site control requires an understanding of how vehicle control is affected by larger transport latencies and dropped packets, which are inherent properties of a wireless network. This study aimed to shed some light on these effects on lateral control of a passenger vehicle when the vehicle's controller was placed off-site.

In this study a Linear Quadratic Self-Tuning Regulator (LQSTR) controller, making use of a vehicle model based on auto-regressive theory describing the relationship between the yaw rate and steer angle of a vehicle, was used. A simulation study showed that the controller was able to control the vehicle through a Double-Lane-Change (DLC) -manoeuvre at vehicle speeds of at least $80\ km/h$ while maintaining very low path tracking errors.

With only minor alterations made to the controller's parameters the system was then subjected to transport latencies and packet drops between the vehicle and its controller to simulate off-site vehicle control. It was found that control of the vehicle was lost when latencies between the vehicle obtaining sensor data and a subsequent control action being realised were in excess of $240\ ms$ and packet drop percentages reached 40% to 50%. If no packets were dropped, i.e. 0% packet drop, transport latencies could be as high as $460\ ms$ before control of the vehicle was lost entirely.

Acknowledgements

- Dr. Botha, for his guidance and expertise in helping in the completion of this study.
- Members of the VDG, for their companionship and morning coffee discussions.
- Victoria Bennett, for making all of this possible.
- Savannah Bennett, for being a supportive sibling.

Contents

Contents iii

List of Figures vi

List of Tables viii

1 Introduction 1
- 1.1 ADAS . . . 2
  - 1.1.1 Collision Avoidance Systems . . . 3
- 1.2 The Importance of Understanding Latencies in CA Systems . . . 4

2 Literature Review 7
- 2.1 V2X Technologies . . . 7
  - 2.1.1 V2P . . . 8
  - 2.1.2 V2I . . . 9
  - 2.1.3 V2V . . . 9
- 2.2 WLAN . . . 10
  - 2.2.1 Channels, Bandwidth, and Bit-Rate . . . 10
  - 2.2.2 MANETs and VANETs . . . 11
  - 2.2.3 A Brief History of the 802.11 Protocol . . . 12
- 2.3 WLAN and V2X . . . 12
  - 2.3.1 Cooperative Awareness and Beaconing . . . 13
  - 2.3.2 Latencies in VANETs . . . 13
  - 2.3.3 VANETs and Location Tracking . . . 14
- 2.4 The 5.9GHz Band . . . 14
- 2.5 The 802.11p Standard . . . 15
  - 2.5.1 Beaconing with the 802.11p Standard . . . 16
  - 2.5.2 Enhanced Distributed Channel Access . . . 17
  - 2.5.3 CSMA/CA . . . 17
  - 2.5.4 Readiness of the 802.11p Standard . . . 19

2.5.5 Suggestions for the 802.11p Standard . . . 19
2.5.6 The Nv2 Protocol . . . 20
2.6 Alternatives to the 802.11p Standard . . . 21
2.6.1 LTE . . . 21
2.6.2 WiMax . . . 22
2.6.3 Satellite Broadband . . . 23
2.7 Common Driver Models . . . 23
2.7.1 Human Based Driver Models . . . 24
2.7.2 Vehicle-Based Driver Models . . . 25
2.8 The AutoRegressive Model with eXogenous Inputs . . . 30
2.9 Conclusion . . . 31
2.10 Research Focus . . . 31

3 Latencies in WLANs 33
3.1 Latency Sources in WLANs . . . 34
3.1.1 Propagation Delays . . . 34
3.1.2 Transmission Delays . . . 35
3.1.3 Processing Delays . . . 35
3.1.4 Packet Drops . . . 35
3.2 Simulating Real World Latencies . . . 36
3.2.1 Latency Model Experimental Set-Up . . . 36
3.2.2 Latency Model Experimental Results . . . 38
3.3 Latency Model Generation . . . 43
3.4 Conclusion . . . 44

4 Controller Synthesis and Analysis 45
4.1 The ARX-Model . . . 47
4.1.1 ARX-Model Order . . . 48
4.1.2 ARX-Model Sampling . . . 50
4.1.3 ARX-Model Parameters . . . 51
4.1.4 ARX-Model Conclusion . . . 52
4.2 LQSTR . . . 53
4.2.1 LQSTR Overview . . . 53
4.2.2 State Space Realisation and Implementation . . . 54
4.2.3 LQSTR Preview Times and Error Calculations . . . 56
4.2.4 LQSTR Parameter Values . . . 58
4.3 Validated Simulation Model . . . 59
4.3.1 $\mathbf{4}S_4$ Modification . . . 59
4.3.2 Adams and Simulink . . . 59
4.4 Test Paths and Measures of Accuracy . . . 61
4.4.1 The DLC-Manoeuvre . . . 61
4.4.2 Measures of Path-Following Accuracy . . . 63

4.5 LQSTR Simulation Study . . . 64
4.5.1 DLC Results . . . 64
4.5.2 LQSTR Gain Computation Analysis . . . 68
4.5.3 ARX-Model Parameters' Analysis . . . 70
4.5.4 Comparison of Original and Modified Controllers . . . 71
4.6 Conclusion . . . 73

5 Wireless Simulations **74**
5.1 Controller Alterations for Latency Simulations . . . 75
5.1.1 Frequency Alterations . . . 76
5.2 Latency Model Alterations for Simulations . . . 77
5.3 Wireless Simulations' Results . . . 78
5.3.1 Variable Baseline Latency . . . 79
5.3.2 Variable Packet Drop Percentage . . . 80
5.4 The Effect of Extra System Delays . . . 81
5.4.1 Extra System Delays - Parameter Alterations . . . 82
5.4.2 Extra System Delays - Results . . . 83
5.5 Conclusion . . . 85

6 Conclusion and Recommendations **87**
6.1 Conclusion . . . 87
6.2 Recommendations . . . 89

Appendices **91**

A Vehicle Properties **92**

B Lateral Error Calculation **93**

C Wireless Simulation Results **96**

Bibliography **98**

List of Figures

1.1 Road Traffic Deaths from 2001 until 2013 [WHO, 2015]. 1

2.1 DSRC Spectrum Band and Channels for the USA [Jiang and Delgrossi, 2008]. 15
2.2 Depiction of the Hidden Terminal Problem experienced when using CSMA/CA at the MAC layer. 18
2.3 A comparison of LTE-V2V with the 802.11p standard when increasing awareness range for a beacon size of 190 *bytes* [Cecchini et al., 2017]. 22
2.4 Visual representation of the pure pursuit algorithm presented in [Coulter, 1992]. . . 27
2.5 Visual representation of the kinematic bicycle model [Polack et al., 2017]. 28
2.6 Visual representation of the dynamic bicycle model [Snider et al., 2009]. 29

3.1 Depiction of the locations of the main sources of latencies in a wireless network. . . 34
3.2 Results of the wired latency tests. 41
3.3 Results of the wireless 20 m latency tests. 41
3.4 Results of the wireless 150 m latency tests. 42
3.5 Results of the wireless 700 m latency tests. 42
3.6 Wireless latency model, experimental and artificial results for 700 m. 44

4.1 Vehicle parameter convention showing steer angle and yaw angle of the vehicle. . . . 46
4.2 Calculated LQSTR gains in handling mode - blue; and ride comfort mode - green, during simulation of an increasing sinusoidal path at 80 km/h [Kapp, 2014]. 48
4.3 Lateral acceleration and cross-track error during simulation of a vehicle in handling mode - blue; and ride comfort mode - green, travelling through an increasing sinusoidal path at 80 km/h [Kapp, 2014]. 49
4.4 LQR block diagram for a discrete-time system. 54
4.5 LQSTR block diagram as presented in this study. 57
4.6 Full vehicle model, as simulated in Adams. 60
4.7 DLC course layout, ISO 3888-1 [Botha, 2011]. 62
4.8 Simulated DLC path through DLC boundaries, as presented in Equation 4.28. . . . 63
4.9 Simulated DLC results at 30 km/h in handling mode. 65

4.10 Simulated DLC results at 60 km/h in handling mode. 65
4.11 Simulated DLC results at 100 km/h in handling mode. 66
4.12 Simulated DLC results at 120 km/h in handling mode. 66
4.13 Comparison of RMSEs for handling mode and ride comfort mode suspension settings. 67
4.14 Comparison of iteration counts for vehicle speeds of 30 km/h, 60 km/h, and 120 km/h through the DLC-manoeuvre. 68
4.15 Comparison of the average calculated LQSTR gain (K_{LQR}) for all simulated vehicle speeds. 69
4.16 Comparison of the calculated theta values for vehicle speeds of 30 km/h, 60 km/h and 120 km/h through the DLC manoeuvre. 70
4.17 Comparison of the modified and original controllers' RMSEs through the DLC-manoeuvre. 71
4.18 Original controller DLC-manoeuvre results at 110 km/h [Kapp, 2014]. 72
4.19 Modified controller DLC-manoeuvre results at 110 km/h. 72

5.1 Depiction of the locations where latencies were to be present during simulations. . . 75
5.2 RMSE of the vehicle through the DLC-manoeuvre at various baseline latencies and various packet drop percentages. 79
5.3 Results of the DLC-manoeuvre simulation for various baseline latencies and a packet drop percentage of 0%. 80
5.4 Results of the DLC-manoeuvre simulation for various packet drop percentages and a baseline latency of 0 ms. 81
5.5 RMSE of the vehicle through the DLC-manoeuvre at various baseline latencies and various packet drop percentages, including a steer delay. 83
5.6 Results of the DLC-manoeuvre simulation for various baseline latencies and a packet drop percentage of 0% - including a steer delay. 84
5.7 Results of the DLC-manoeuvre simulation for various packet drop percentages and a baseline latency of 0 ms - including a steer delay. 85

B.1 Calculation of lateral error, system prior to any translations and rotations. 94
B.2 Calculation of lateral error, system after translation. 94
B.3 Calculation of lateral error, system after rotation. 95

List of Tables

3.1 Processing unit used during the latency model generation. 37
3.2 Mikrotik router used for latency model generation. 37
3.3 Statistical summary of latency tests' RTT results. 38
3.4 Latency Model Distribution Values. 43

4.1 LQSTR on-board parameter values. 58
4.2 DLC course width and length values [Botha, 2011]. 61

5.1 Parameter values used when the controller was simulated to be off-site. 76
5.2 Simulated Latency Distribution Values. 77
5.3 Parameter values used when the controller was simulated to be off-site with a steering delay. 82

A.1 Properties of the fully-kitted Land Rover Defender 110 around which the simulation model was built. 92

C.1 Summary of the wireless simulations' results, RMSE for each simulation. 96
C.2 Summary of the wireless simulations' results including a steering delay of 200 ms, RMSE for each simulation. 97

List of Acronyms

Acronyms and Abbreviations	Description
3GPP	3rd Generation Partnership Project
$\mathbf{4}S_{\mathbf{4}}$	Four-State Semi-active Suspension System
ACC	Adaptive Cruise Control
ACK	ACKnowledgements
ADAS	Advanced Driver-Assistance Systems
AP	Access Point
AR	Auto-Regressive
ARX	Auto-Regressive with eXogenous inputs
Bps	Bytes Per Second
bps	Bits Per Second
CA	Collision Avoidance
CAM	Cooperative Awareness Message
CCH	Control CHannel
C-ITS	Cooperative-Intelligent Transport Systems
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access/Collision Detection
D2D	Device-To-Device
deg	Degrees
DLC	Double Lane Change
DSRC	Direct Short-Range Communication
EDCA	Enhanced Distributed Channel Access
EU	European Union
FCC	Federal Communications Commission
GDP	Gross Domestic Product
GNSS	Global Navigation Satellite System
h	Hour
Hz	Hertz
IEEE	Institute of Electrical and Electronics Engineers
ISA	Intelligent Speed Adaptation
LAN	Local Area Network
LEO	Low Earth Orbit

Acronyms and Abbreviations	Description
LQR	Linear Quadratic Regulator
LQSTR	Linear Quadratic Self-Tuning Regulator
LTE	Long-Term Evolution
m	Metre
MAC	Media Access Control
MANET	Mobile Ad-hoc NETwork
MIMO	Multiple-Input Multiple-Output
NGSO	Non-GeoStationary Orbit
NHTSA	National Highway Traffic Safety Administration
NZ	New Zealand
OS	Operating System
PHY	PHYsical
rad	Radians
RAM	Random Access Memory
RMSE	Root Mean-Squared Error
rpm/RPM	Revolutions Per a Minute
RSU	Road Side Unit
RTS	Request-To-Send
RTT	Round-Trip Time
s	Seconds
SCH	Service CHannel
SSA	Safe System Approach
STDMA	Self-organised Time-Division Multiple Access
TC-ITS	Technical Committee-Intelligent Transport Systems
TDMA	Time-Division Multiple Access
UDP	User Datagram Protocol
USA/US	United States of America
V2I	Vehicle-To-Infrastructure
V2P	Vehicle-To-Pedestrian
V2R	Vehicle-To-Roadside
V2V	Vehicle-To-Vehicle
V2X	Vehicle-To-Everything
VANET	Vehicular Ad-hoc NETwork
VDG	Vehicle Dynamics Group
VLEO	Very Low Earth Orbit
VRU	Vulnerable Road User
WAVE	Wireless Access in Vehicular Environments
WHO	World Health Organisation
WiMax	Worldwide interoperability for Microwave access
WLAN	Wireless Local Area Network

List of Symbols

Roman Symbols

Symbol	Description	Units
A, B, C, D	State space matrices	
a, b, c, d	Corresponding state space constants	
cg/CG	Centre of gravity	m
e_{lat}	Straight-line lateral error	m
f_{sample}	ARX-model sample rate	Hz
f_{model}	ARX-model update rate	Hz
$f_{vehicle}$	Vehicle parameter update rate	Hz
f_{send}	Packet transmission rate	Hz
f_{Adams}	Overall simulation frequency	Hz
g	Number of exogenous input terms in the ARX-model	
H	Transfer function of ARX-model	
J_{LQR}	Performance function of the LQSTR controller	
K	Original controller gains	
K_{LQR}	Modified controller gain	
k	Time step	
N	ARX-model sample count	
$\bar{N}$	Pre-compensation gain of the LQSTR controller	
P/p	Ricatti Equation solution	
Q_{LQR}	Set-point weighting matrix	
q	Set-point weighting constant	
R_{LQR}	Control input weighting matrix	
r	Control input weighting constant	
s	Number of regressive terms in ARX-model	
Δt	Change in time	s
u	Exogenous inputs in ARX-model (Steer angle)	deg
Y	Vector of measured yaw rates	
$\widetilde{Y}_t$	Estimated ARX-model value (Yaw rate)	$\deg/s$ or rad/s

Greek Symbols

Symbol	Description	Units
δ	Steer angle at front wheels	deg or rad
ε	Unmodeled residuals in the ARX-model	
η	Steer angle weighting in the ARX-model	
φ	Yaw rate weighting in the ARX-model	
Φ	Regression vector	
ψ	Yaw angle of a vehicle	deg or rad
$\dot{\psi}$	Yaw rate of a vehicle	deg $/s$ or rad/s
$\ddot{\psi}$	Yaw acceleration of a vehicle	deg $/s^2$ or rad/s^2
$\widetilde{\psi}$	Estimated yaw rate of a vehicle	deg $/s$ or rad/s
$\sum$	The sum-of	
τ_{acc}	Yaw angle to yaw rate conversion preview time	s
τ_{lat}	Lateral error preview time	s
τ_{yaw}	Yaw error preview time	s
θ	Parameter vector	

Chapter 1

Introduction

The need for Advanced Driver Assistance Systems (ADAS), technology that increases a vehicles occupants' safety/comfort, and by extension those around the vehicle, can be highlighted by a cursory glance at the statistics published by the World Health Organisation (WHO). In 2013, 1.25 million people died on roads around the world, and although this number seems to have plateaued since 2007, see Figure 1.1, the majority of these deaths are considered preventable. More recent data was not immediately available at the time of conducting this study.

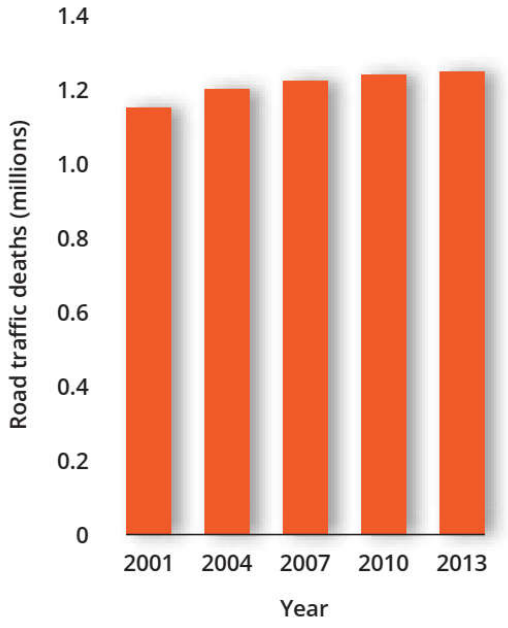

Figure 1.1: Road Traffic Deaths from 2001 until 2013 [WHO, 2015].

These road deaths account for, on average, a 3% loss in a country's total Gross Domestic Product (GDP) on a yearly basis. A further 20 to 50 million people are injured on the roads, with some of these injuries leading to permanent disabilities [WHO, 2015].

The Safe System Approach (SSA), a design methodology shared by the WHO, Safer Journeys in New Zealand (NZ), and others, takes into account human fallibility and vulnerability when designing transport systems. This goes beyond just vehicle design, with the idea that it will reduce death and serious injury on the roads [Journeys, 2018]. Two key pointers of the SSA is that people make mistakes and that all parts of the road system need to be strengthened/improved. The first of these pointers, people make mistakes, will always be present when people are given control. A driver has only a limited peripheral knowledge of a vehicle's states and the states of the vehicle's surroundings. Under normal conditions, this is adequate to maintain safe control of a vehicle, but should either of these states change suddenly, the driver may not be able to adequately react to the change. An example could be the appearance of an obstacle in the road. Very few drivers know exactly when a vehicle is going to lose control, and as such they may not handle the vehicle appropriately when attempting to avoid the obstacle - should they see it. The second point, strengthening/improving of the road system, can be seen as partially accommodating of the first point. By taking into account road users' (drivers/pedestrians) fallibility, systems can be designed and installed that can pick up the slack when one part of the system fails. ADAS has a pivotal role to play in the strengthening of the road system to accommodate for the fallibility of humans as road users. The next section has a quick look at ADAS and some of the technologies being used with a particular focus on collision avoidance systems.

1.1 ADAS

Over recent years secondary safety features in vehicles have been shown to increase the safety of vehicle occupants as well as all other road users. Primary systems are now being produced to further improve on this increased safety. A secondary safety system, also known as a passive system, is activated in response to a crash/accident, protecting road users during and after the incident has occurred, i.e. they do not attempt to prevent the incident from occurring. Primary systems, also known as active systems, aim to prevent/reduce the severity of the accident before it takes place. Both of these systems can fall under ADAS.

ADAS is a broad field that covers many topics, well beyond that of pure vehicle control. A few examples of some ADAS technologies that are already employed by the vehicle manufacturing industry are given below. This list covers only a select few examples that fall under ADAS, showing that ADAS technologies are already a prominent feature that need to be considered in the design of all vehicles.

1. Intelligent Speed Adaptation (ISA): ISA is a system that informs the driver if they are either travelling too fast, with reference to the legal speed limit of an area, or too slow, allowing the driver the opportunity to correct their speed.

2. Emergency Brake Assist: A system installed that attempts to determine whether the driver is actively trying to perform an emergency braking manoeuvre or not. If the system determines that the driver did intend to perform an emergency braking manoeuvre, full emergency braking features are put into effect.

3. Lane Departure Warnings: Has two primary modes. The first is merely a warning, visual or auditory, letting the driver know that their current heading is causing them to drift out of their lane. The second allows the vehicle to automatically correct the heading of the vehicle should unwanted drift be detected.

4. Collision Avoidance (CA): A big part of ADAS. This can vary from alerting the driver to possibly dangerous situations to taking full control of the vehicle to prevent an accident from occurring, the later of which is covered in more detail below.

1.1.1 Collision Avoidance Systems

CA systems aim to prevent crashes entirely, using either lateral or longitudinal vehicle control. CA systems can either take control of the vehicle or provide pre-emptive warnings that alert the driver to possible dangers. A few well known longitudinal and lateral CA systems are highlighted below:

1. Adaptive Cruise Control (ACC), where the cruising speed of the vehicle is adjusted to keep a safe following distance between it and the preceding vehicle without any additional input required from the driver.

2. Downhill-Assist Control, where the vehicle maintains a constant speed when descending steep and/or slippery slopes [Toyota, nd].

3. Steering Assist, where the vehicle takes control of itself to avoid obstacles/hazards it has perceived to be in the vehicle's path. This can be combined with lane departure warning technologies for more involved CA systems.

A single CA system, such as point 3 above, is itself comprised of many smaller subsystems, from visual identification of objects surrounding the vehicle using on-board sensors to path planning and applying a control action that will allow the vehicle to safely avoid the hazard without losing control, and without endangering any other nearby road users. To generate such a CA system requires, amongst other items, a controller that is able to safely steer the vehicle along a desired path. Such a steering controller was designed in this study, with a study on the effects of implementing the controller from off-site, thereby introducing large transport latencies between the vehicle and the off-site controller. The path planning and object identification of such a system went beyond the scope of this study.

1.2 The Importance of Understanding Latencies in CA Systems

Most vehicle control systems are placed within the vehicle. In that aspect it replicates a human driver where there is a minimal delay between determining a suitable control action and physically implementing said action, and although controlling a vehicle from a location other than on-board the vehicle is not a new technology, it still poses quite a challenge. This is because the exact amount of permissible transport latencies between the vehicle and controller depends on many factors, such as: the type of controller being used, the level of accuracy when measuring the vehicle states, the condition of the tyre-road interface, etc.

When making manoeuvres at high speeds, reaction time is of utmost importance. CA systems increase all-round safety by, among other things, improving the reaction time of the vehicle system as a whole, this includes the driver or controller, by potentially creating a larger effective reaction window. Now, when the controller is placed off-site, the transport latencies that are introduced reduce this effective reaction window.

This begs the question: If on-site control is possible, and safer, why have off-site control at all? A possible use for off-site control is for traffic management at, for example, a busy interchange. Self-driving cars, where the controller is on-site, are all planning their own routes and applying their own control. This information can, and most likely would, be shared between all the vehicles to increase cooperative awareness via beaconing, so that each individual vehicle can attempt to make the best decision considering the decisions all the surrounding vehicles are making. In such a scenario it may be preferable to have all the decision making made by one central system, that way one system knows where all the vehicles are and can plan all routes and all control actions accordingly, without the need to wait for multiple smaller systems to make their own decisions and inform surrounding vehicles. This central control system can then optimise the paths each vehicle should take with respect to one another, perhaps reducing the possibility of a collision occurring. Such a central vehicle management system is not viable only to passenger vehicle control scenarios, but can also be used in the industrial sectors, e.g. control of surface mining equipment or construction equipment. Mining vehicle operators are already being taken out of the vehicles they control, i.e. the operators control the vehicles from a remote location [Sasaki et al., 2004]. Such a system not only increases the operators' safety, but also increases efficiency as the operators are in a more comfortable working environment. The next step in such systems would be to go fully autonomous, where human operators are no longer required.

For implementation of large-scale off-site control to be possible, two key ideas need to be realised: the first is that the central control system has to be able to accommodate a large volume of different vehicles at high speeds while maintaining safe and reliable control. Second is that the designed system has to be able to either account for the possibility of larger transport latencies, as well as packet drops, between the central control system and each vehicle, or be designed in such a way that the disturbances caused by the latencies and packet drops are reliably rejected. This study aimed to show the effect that larger transport latencies and packet drops had on the controllability of a vehicle travelling at the common suburban, and mining vehicle, speed limit of $60\ km/h$ (South Africa) when using a platform independent driver model as part of the controller. This was accomplished by determining, through simulations, the maximum transport latency that could be present between the vehicle and its controller, as well as the maximum percentage of packets that could be dropped, before control of the vehicle became unreliable or was lost.

This study is organised in the following manner. Chapter 2 covers the current state of vehicle communications, as most current off-site communications with vehicles are related to supplementary

information the vehicle can use to increase safety, and not actual control of the vehicle. Chapter 3 covers latencies and their causes, along with a generated model that can be used in simulations to recreate real-world latencies. Chapter 4 details the design of the platform independent controller with a section dedicated to the simulation model that was used to conduct simulations. The path that was used for all testing purposes, as well as the measures of accuracy used, is also covered with a simulation study that was carried out to verify the controller's suitability as a robust controller. Chapter 5 covers the wireless simulations that were carried out. Chapter 6 closes of the study with conclusions drawn and recommendations for any future work.

Chapter 2

Literature Review

Following is a brief overview of some of the technologies that are currently being used for wireless vehicular communications. First the main Vehicle-To-Everything (V2X) technologies are described, highlighting the need for such systems with a general scope of each technology given. Next, a semi-in-depth look at Wireless Local Area Network (WLAN) technologies is given, with a particular interest in the Institute of Electrical and Electronics Engineers' (IEEE) 802.11p standard, mentioning its strengths and weaknesses in the wireless vehicular environment. A look at some of the common controller types often found in literature dealing with lateral vehicle control is also presented.

The inclusion of information regarding satellite broadband (Chapter 2.6.3) hints at the true scale of practical implementation of controlling a vehicle from off-site. Such a system, offering world-wide, low latency internet connectivity, could potentially allow for off-site control of vehicles on a global scale. This could even allow areas that do not have the infrastructure, or capital, for the installation of localised vehicle control systems to take advantage of off-site control.

2.1 V2X Technologies

V2X covers a broad range of vehicular communications, the purpose of which is to enhance the safety of road users as well as the ability of decision-making vehicles on the road. This is accomplished by supplementing the information available to the vehicle and/or surrounding infrastructure/road users. The three main V2X technologies, Vehicle-To-Pedestrian (V2P), Vehicle-To-Vehicle (V2V), and Vehicle-To-Infrastructure (V2I), that are already being implemented in some parts of the world

are covered here. Some example cases are given for each, showing some of the current applications where each technology is being utilised. For a more complete history of V2X technologies, see [MacHardy et al., 2018].

2.1.1 V2P

V2P communications are geared more towards warning systems for pedestrians, cyclists, or more generally "those that are not in a vehicle", also known as Vulnerable Road Users (VRUs). This section of V2X technology had been lagging behind other V2X technologies quite extensively but with improvements in smartphones' communication capabilities, particularly Direct Short-Range Communication (DSRC) and Wi-Fi Direct enabled smart-phones, this is changing. DSRC and Wi-Fi Direct technologies enable messages to be sent between devices without the need for a cellular network, greatly reducing the latencies experienced by the messages.

The National Highway Traffic Safety Administration (NHTSA) of the United States of America (USA) stated that in 2014 there were an estimated 5000 pedestrian fatalities, with over 65000 pedestrian injuries, that occurred on the road [Tahmasbi-Sarvestani et al., 2017]. In the European Union (EU), it was estimated that 46% of road deaths in 2016 were those of VRUs. This is the same number of deaths as vehicle occupants, coming from some of the safest roads in the world [Commission, 2017]. These statistics highlight the need for systems that add to the safety of VRUs, beyond the safety added from purely non-cooperative ADAS. Non-cooperative ADAS are systems in which the vehicle makes all the decisions without alerting/informing VRUs that may be affected by the vehicle's decision. A cooperative ADAS goes beyond purely making its own decisions by alerting the VRU to its presence/decision, thereby potentially increasing the safety of the VRU. This increased safety comes about by increasing the VRUs awareness of their surroundings with respect to the vehicle's decision(s).

It has been suggested that by using V2P communication technologies it would be possible to increase VRUs situational awareness by making use of Personal Safety Messages (PSMs) that are broadcast using DSRC/Wi-Fi Direct technologies. This improves upon the purely passive systems that have already been implemented with VRUs in mind, such as pedestrian crumple zones on the hood of vehicles. In line with increasing VRUs situational awareness, V2P frameworks are now being incorporated into ADAS, resulting in cooperative systems.

2.1.2 V2I

V2I communications encompass the exchange of data between a vehicle and a static structure relatively nearby (depending on the technology), commonly referred to as a Road Side Unit (RSU). This has the potential to increase the sensing range of a vehicle beyond that of its on-board systems, e.g. informing vehicles/drivers of environmental changes far further-a-field than what any vehicle can sense on its own as RSUs can be connected to far more expansive networks than vehicles alone.

V2I can also be used for congestion mitigation, where the flow of traffic, for example, around a city is monitored using data collected by multiple RSUs making use of V2I communications. Vehicles are subsequently informed of the best routes to take. V2I also has the potential to help with emergency services. If a vehicle is experiencing problems, it can notify the nearest RSU, which in turn can notify the nearest relevant emergency service, avoiding multi-hop message broadcasting methods - this is where a packet is sent much further from one network node by being re-broadcast multiple times by other surrounding nodes. V2I can also be used in less permanent situations, where an RSU is temporarily installed to alert vehicles to surrounding abnormal road conditions. A well-established V2I system already in-place around the world is automatic tolling.

2.1.3 V2V

V2V communications are primarily used as a way of extending the sensing range of a single vehicle well beyond its own sensors' limits while potentially increasing the decision-making capabilities of individual vehicles [Caveney and Dunbar, 2012]. Such limits are only further reduced when there are many surrounding vehicles. Typical data contained in a V2V communication packet could include: speed, heading, and brake-status of the broadcasting vehicle [Harding et al., 2014], which is sent to surrounding vehicles using DSRC. A packet containing this information would allow the surrounding vehicles to know, for example, the exact braking status of the broadcasting vehicle without having to estimate it using their own sensors, as estimation can lead to inaccurate results and increased delays during decision making. This increased sensing range introduced by V2V communication allows for earlier/faster predictions to be made, granting larger reaction windows to a vehicle when compared to vehicles not making use of V2V communication. An estimate presented by the NHTSA suggests that V2V systems would be able to prevent up to 592000 crashes a year, while saving 1083 lives [Harding et al., 2014].

V2V messages can be used by vehicles in multiple ways, two of which are considered here. First, the vehicle could alert the driver to the onset of a potentially dangerous situation - as contained in the V2V message. This would grant the driver more time to respond, an example of a passive ADAS. Second, the vehicle could take full control to avoid the dangerous situation, an example of an active ADAS. V2V communication is already being utilised by Toyota in Japan [VTS, 2015] where they have Communicating Radar Cruise Control and Emergency Vehicle Notification messages that are broadcast between vehicles on the road. This cruise control expansion allows for vehicles to maintain safe following distances when using cruise control more reliably than a system that only makes use of its own on-board sensors to estimate the states of a preceding vehicle. The Emergency Vehicle Notification messages alert the driver of a vehicle to the presence of an emergency vehicle, allowing the driver to move their own vehicle out of the way should it be necessary.

2.2 WLAN

A WLAN is a collection of devices, called network nodes, that originally needed to wirelessly connect to a central Access Point (AP), enabling the transfer of data between each network node. For the purpose of this study, clarity over the three terms; channel, bandwidth, and bit-rate is given, then a comparison of the two subcategories of WLANs; Mobile Ad-hoc NETworks (MANETs) and Vehicular Ad-hoc NETworks (VANETs), is given. This is followed by a brief history of the WLAN standard, also known as the 802.11 standard, as published by the IEEE. The history presented below is not all-inclusive, it only aims to give the reader a general idea of where the protocol started and where it was at the time of conducting this study. This will help with understanding the choice of WLAN used during the generation of the latency model in Chapter 3, which was used during the simulations of Chapter 5.

2.2.1 Channels, Bandwidth, and Bit-Rate

To understand a good portion of the 802.11 standards one must understand the meaning of the above terms as there can be some confusion. A channel, or communication channel, refers to a medium that can carry a signal from a transmitter(s) to a receiver(s). For a wired network a channel will more than likely be a copper or fibre optic cable, for WLAN a broadcasting (wireless) channel is used. A channel is defined by its central frequency and how wide the channel is. This channel width, usually

given in Hertz (Hz), is known as the bandwidth of a channel [Gates, 2015]. As an example, if it is stated that the channel is located in the $1.2\ GHz$ band and has a channel width of $20\ MHz$, the central frequency will be $1.2\ GHz$ with the channel making use of $10\ MHz$ on either side of it. Bandwidth can also be defined as the rate of data transfer, but for wireless communication and this study, bandwidth will refer to the width of a signal channel. To relate to V2X, seven channels with a bandwidth of $10\ MHz$ each have been allocated for V2X applications in the $5.9\ GHz$ band, more on this in Chapter 2.4.

There are a few performance measures for WLANs, the important ones for this study are latency, covered in Chapter 3, packet drop rate, and bit-rate. Packet drop rate, often given as a percentage, is the number of packets that, once they have left the transmitter, do not reach the receiver. Bit-rate, as its name suggests, refers to the rate at which data bits can be transmitted through a channel. The higher the bit-rate, the more data can be put on the channel over a shorter period of time.

2.2.2 MANETs and VANETs

A MANET is a subclass of WLAN that is comprised of mobile nodes that do not require an AP for the network to exist. The network itself is self-organising with nodes having the ability to join and leave the network at random, while also being able to change locations relative to one another. This results in a variable network topology, hence "ad-hoc". One of the key characteristics of a MANET is that a node has the potential to act as a sender, receiver, and/or a router. In this context a router refers to a node that can relay messages between nodes that are, perhaps, out of communication range of one another. This means that there is no requirement for permanent infrastructure [Chong et al., nd].

A VANET is a subclass of MANET. It maintains all the characteristics of a MANET, with the added consideration that the network nodes are vehicles. This results in network nodes that are moving a lot faster than the network nodes in a conventional MANET. The non-requirement for an AP in VANETs, and MANETs, gives them the ability to act as a form of emergency broadcast system should standard systems be inoperable, for whatever reason, as they do not require an AP/infrastructure to send messages across the network [Abdelgader and Lenan, 2014].

2.2.3 A Brief History of the 802.11 Protocol

The 802 Project was started by the IEEE as far back as 1980. The first specification, 802.11, was published in 1997 and had bit-rates on the order of $1\ Mbps$ to $2\ Mbps$ while operating in the $2.4\ GHz$ band. Over the years many changes have been made to this (legacy) specification that have allowed bit-rates to increase beyond $1\ Gbps$ (for the 802.11ac specification - published in 2013), while at the same time increasing network reliability and security. For a more complete explanation of the 802.11ac specification see [Cisco, 2018].

One of the major advances in the 802 Project was the inclusion of Multiple-Input Multiple-Output (MIMO) techniques, first published with the 802.11n specification in 2009. Originally this meant that more antennas were used to transmit and receive data over a radio link. This later evolved into multipath propagation where more than one data stream could be sent over a single channel. This meant that the total bit-rate could be scaled up by including more antennas or by making use of multipath propagation [Kim and Lee, 2015].

In 2010 the 802.11p specification was published with amendments to make Wireless Access in Vehicular Environments (WAVE), a vehicular communication system, possible. A more detailed look at the 802.11p standard is presented in Chapter 2.5.

2.3 WLAN and V2X

For obvious reasons cabled network topologies are not suitable for V2X communications due to the high mobility of vehicles/nodes. A cabled connection may be suitable for connections linking infrastructure but as soon as a certain degree of mobility is required, wireless network channels become a requirement. The idea behind V2X at the moment is to create cooperative awareness between vehicles, infrastructure, and other road users. This is achieved through Cooperative Awareness Messages (CAMs) that are sent between nodes (vehicles/RSUs/VRUs) periodically, a process known as beaconing, based on some criterion [Lyamin et al., 2018]. Cooperative awareness, beaconing, why latencies in VANETs are a cause for concern, and using the properties of VANETs to enable location tracking are discussed below.

2.3.1 Cooperative Awareness and Beaconing

One of the essential ideas behind V2X, and more recently ADAS, is cooperative awareness. This is the sharing of information between vehicles in a network, and beyond, to increase the data/information available to individual vehicles well beyond what their own sensors are able to capture at any given time. This sharing of information is largely accomplished via beaconing between vehicles, and between vehicles and infrastructure/VRUs.

Beaconing is where vehicle data is shared with surrounding vehicles. An example of beacon data could be the vehicle's states (velocity, heading, position, etc.) or external information acquired from the vehicle's environmental sensors (obstacles around the vehicle, the condition of the road, weather conditions, etc.). To increase on the safety added by beaconing, beacons can be broadcast in either single-hop or multi-hop fashion. This could theoretically allow a beacon from one vehicle to traverse an entire city in a relatively short period of time - without the need for permanent infrastructure. The size of a beacon varies based on the data that is being sent but tends to be roughly $400\ bytes$ in size [Tomar et al., 2017].

Presently V2I is mainly concerned with safety messages containing information about roadworks and road conditions in the area, as well as infotainment. V2I can however greatly improve on the, currently, limited range of V2V. Where further ranging messages would require the use of multi-hop methods to travel between vehicles across a city using only V2V, V2I would allow a more direct sending system, avoiding the shortfalls of multi-hop networks, such as the increased probability of packet collisions with an increase in vehicle density [Atallah et al., 2015].

2.3.2 Latencies in VANETs

When considering data transmissions in VANETs, information/packet data that is delivered to its destination correctly but after a certain deadline is, for all intents and purposes, useless and has the potential to lead to catastrophic failure of, and within, a system. This holds true for beaconing, where vehicles can be moving at very high speeds, both in unison and relative to one-another. If a beacon is received late, the transmitting vehicle may no longer be near its last location contained in the CAM, and/or all of the sending vehicle's states may have changed.

It is stated in the 802.11p specification that data/packet exchanges should be able to be completed within $50\ ms$ [Khairnar and Kotecha, 2013], this is for pre-crash warning messages, and within

$100\ ms$ for other V2X safety service messages. A more in-depth analysis of latencies and their causes is given in Chapter 3.

2.3.3 VANETs and Location Tracking

[Hossain et al., 2016] and [Jiangfeng et al., 2009] have conducted studies where they made use of V2V and Vehicle-To-Roadside (V2R) communications to accurately pinpoint the location of vehicles within a VANET. Their results showed that such tracking systems were more accurate than the more commonly used Global Navigation System of Satellites (GNSS) alone. VANET tracking methods can be relative: if the RSU that the vehicles are connecting to only calculates the position of the vehicles relative to the RSU's location, or absolute in the global sense: if the RSU has a known GNSS location to which it can compare the relative location of the vehicles. This requirement for an RSU in VANET location tracking is a drawback only in terms of global vehicle positioning as once there are no longer any RSUs within communicating range of the vehicle, the ability to track a vehicle's position is lost. This may make VANET location tracking more suitable as a complimentary system to GNSS, as when driving in built-up areas, or areas with poor satellite visibility, GNSS can become unreliable. During such times VANET tracking can take over as built-up areas can have multiple RSUs creating a dense network for the vehicles to connect to.

2.4 The 5.9GHz Band

In the USA a $75\ MHz$ spectrum in the $5.9\ GHz$ band has been set aside for DSRC with its sole purpose being for V2X communications. The band is broken up into smaller channels, each of them having a bandwidth of $10\ MHz$. There is one Control CHannel (CCH) and 6 Service CHannels (SCHs) [Jiang and Delgrossi, 2008]. The CCH is intended for the transmission of critical information as well as the organisation of the SCHs, achieved by handling the data streams. The SCH is intended to carry less safety critical information.

The band used in the EU has not been specified in exactly the same manner, with the European Cooperative-Intelligent Transport Systems (C-ITS) services operating from $5.855\ GHz$ to $5.875\ GHz$, $5.905\ GHz$ to $5.925\ GHz$, and $63\ GHz$ to $64\ GHz$ [Commission, 2016], while Japan uses the $5.725\ GHz$ - $5.875\ GHz$ band [ARIB, 2001]. Each of these set-ups is different, depending on what the sitting committee decided. For the purpose of clarity, the work referred to in this study has to do

with the DSRC rules and implementations set by the NHTSA of the USA. A representation of the $75\ MHz$ band, comprised of its multiple channels, can be seen in Figure 2.1.

Infotainment will have less of a focus in this $75\ MHz$ spectrum, as it is not a requirement for cooperative awareness and would cause unnecessary clogging up of the network channel where lower latencies are key to safety. Due to infotainment not being a safety related messaging system, it is allowed a higher maximum one-way latency than what is required by CAMs, where CAMs have requirements of $100\ ms$ latency and below, infotainment and traffic management have been marked at $500\ ms$ [Filippi et al., 2016].

The 802.11p standard sets up the rules and regulations for enabling DSRC in the $5.9\ GHz$ band and is discussed in greater detail next.

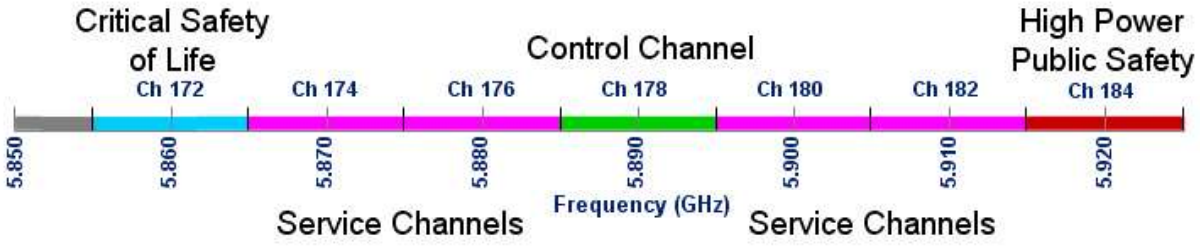

Figure 2.1: DSRC Spectrum Band and Channels for the USA [Jiang and Delgrossi, 2008].

2.5 The 802.11p Standard

The 802.11p standard, published by the IEEE in 2010, is an amendment to the ongoing 802.11 standard with a special focus on VANETs where there is a very high mobility of nodes in an ad-hoc network topology. Due to the ad-hoc nature of VANETs, the 802.11p standard removed the necessity for a central AP that controlled and monitored network access. This alteration means that vehicles are able to quickly, and efficiently, join and leave the network of nodes without a primary node/AP delegating resources.

To realise these changes, the Media Access Control (MAC) and PHYsical (PHY) layers' specifications were altered by the inclusion of the 1609.4 data link layer specification to allow for DSRC. The 1609.4 data link layer is a subset of the IEEE 1609.x standards that were introduced, collectively known as WAVE (USA) or TC-ITS (EU). Other 1604.x standards included security features that enabled secure V2X wireless communication. The MAC layer is responsible for sending data packets

across a shared channel while avoiding collisions with other data packets on the channel [Beal, nd]. The PHY layer consists of the circuitry required to implement physical layer functions, found above the MAC layer. The 1609.x standards support bit-rates of up to $27\ Mbps$. In contrast to more recent standards, where 802.11ac can reach bit-rates in excess of $1\ Gbps$, the bit-rate is relatively low. This has however been deemed adequate as the data that is being shared in VANETs for V2X applications does not consist of excessively large files. Media streaming, and other large file data transfers, would fall under infotainment data transmissions, where higher bit-rates are required. 1609.4 allows for the use of Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA), which is described in more detail in Chapter 2.5.3. For more details on the IEEE 1609.x standards, see [USDT, ndb].

Some of the important technologies implemented in the 802.11p standard, and how they operate, are discussed below. Special emphasis is placed on the CSMA/CA MAC layer protocol to show that it may not be the most suitable option for vehicular communications - especially wireless vehicle control. Although the 802.11p standard was not used in this study (see Chapter 2.5.6), some of its operating principles are highlighted here as it is still considered the de-facto standard. Understanding why the 802.11p standard was not used requires an understanding of these operating principles.

2.5.1 Beaconing with the 802.11p Standard

The nature of beaconing, in terms of the vehicle sending out beacons in a VANET, requires that it be a one-way transmission that does not make use of ACKnowledgements (ACKs). An ACK is used to confirm that a message sent by a transmitter has been received by the intended receiver. If ACKs are being used and the transmitter does not receive an ACK after transmitting a message, it will resend the message until an ACK is received. If a broadcasting vehicle/node were to wait for ACKs from each surrounding vehicle it would need to know exactly how many surrounding vehicles there are. This is problematic in VANETs as vehicles can enter and leave the ad-hoc network quite quickly. A vehicle may receive the broadcasted beacon while in range but before it can return an ACK, it has left the range of the broadcasting vehicle. This results in a broadcasting vehicle that is stuck sending the same message waiting for a vehicle to acknowledge it that is no longer a part of the network [Klingler et al., 2015].

2.5.2 Enhanced Distributed Channel Access

Initially the 802.11p standard catered for single channel operation. The 1609.4 standard expanded this to multichannel operation using Enhanced Distributed Channel Access (EDCA). EDCA is a contention based channel access method that uses time division between the CCH and the SCHs, which can hamper beaconing performance as a side effect of contention based channel access is the possibility of unbounded delays. The SCHs can also be combined into single $20\ MHz$ channels by combing two $10\ MHz$ channels, creating a larger available bandwidth.

2.5.3 CSMA/CA

CSMA/CA is used by the MAC layer to avoid, rather than react to, packet collisions within the network - unlike Carrier Sense Multiple Access/Collision Detection (CSMA/CD). If the channel is sensed to be "busy", a random back-off period is chosen during which the sending node remains idle. Once the back-off period has ended the node senses the medium/channel again to check if it is idle. This process is repeated until the node is able to send the message. Although the 802.11p standard is quite mature there are still problems associated with using CSMA/CA at the MAC layer that need to be addressed before full scale adoption can take place safely and effectively. These problems are the hidden terminal problem and the possibility of unbounded delays, both of which are discussed in more detail below.

Nodes that are a part of a WLAN each have their own effective communication range. Once beyond this range, the ability to send and receive messages to and from the node becomes unreliable at best. Since there is no set structure for a WLAN, it is possible for nodes that are a part of the same network to not be within direct communicating range of every other node in the network. This is known as the hidden terminal problem, and it is a major concern for any WLAN, especially those that use CSMA/CA at the MAC layer. In Figure 2.2, the AP is within communicating range of Nodes A, B, and C (the coloured circles show the effective communication range of the corresponding node with the black circle indicating the communication range of the AP). Note that the AP could also represent a node, as would be the case in an IEEE 802.11p DSRC scenario that does not require an AP to set-up a network. If one considers 802.11p's CSMA/CA as the MAC protocol in use, when Node B wants to send a message to the AP it would first "listen" to the wireless channel, allowing it to ascertain whether another node is currently making use of the wireless channel or not. As Node A

and the AP are within effective communication range of Node B, Node B can determine if either the AP or Node A are currently using the channel to transmit a signal. If the channel is sensed to be busy with a transmission then Node B will choose a random back-off time, where it essentially goes into standby. Once this back-off time has passed, Node B will listen to the channel again. This process will continue until the channel is sensed idle, at which point Node B will send its message for the AP to receive. The hidden terminal problem comes in as a result of Node C. There is no way for Node B to sense whether Node C is transmitting as Node C's effective communication range falls short of Node B. Node B may determine that the channel is idle and start to transmit its message, while in fact Node C is busy transmitting. This results in a packet collision and can result in corrupted data at the AP (or any receiving node). It is possible to enable Request To Send/Clear To Send (RTS/CTS) for 802.11p to help combat the hidden terminal problem, but a scheme making using of RTS/CTS will immediately have higher latencies due to the sending and receiving of messages before sending the actual message.

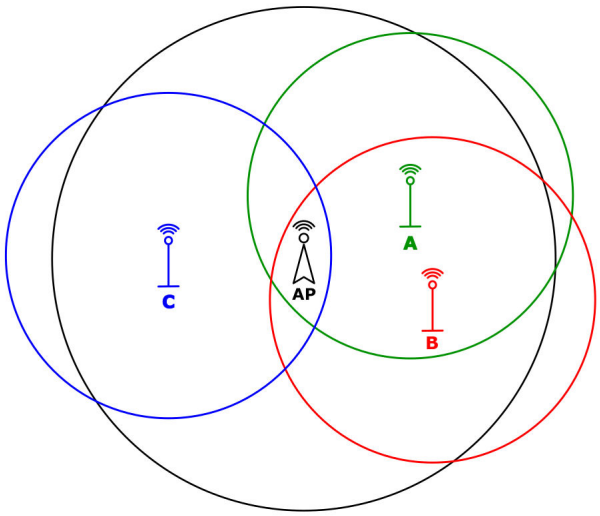

Figure 2.2: Depiction of the Hidden Terminal Problem experienced when using CSMA/CA at the MAC layer.

During high traffic periods the original CSMA/CA MAC protocol can result in unbounded delays, where there is no limit to how long a message can take before it is sent from the transmitting node. Since the CSMA/CA protocol first senses whether the channel is idle or not before sending a message, a permanently busy channel would prevent any messages from being sent. An example of this would be the presence of an interference signal, operating in the same $5.9\ GHz$ band, that prevents all

nodes in an area from sending messages as they will always sense the channel as busy. Another situation that could result in unbounded delays would be a highly congested network. This can result in one node never getting the chance to send a beacon as there is always another node using the channel. This heavy congestion effect was studied in [Khairnar and Kotecha, 2013], where it was shown in simulation that upwards of 50% of packets were dropped on a highway of 1200 vehicles when using the CSMA/CA MAC protocol. A dropped packet referred to packets that were not sent from the transmitting node as by the time the channel was sensed to be idle, the next packet had arrived for sending.

2.5.4 Readiness of the 802.11p Standard

The 802.11p standard is more ready for large scale deployment, when compared to other possible standards (Chapter 2.6), due to how long it has been around. The $5.9\ GHz$ band has already been allocated and manufacturers have started incorporating DSRC enabling hardware devices into their vehicles [Yoshida, 2014], while other manufacturers are very close to including the hardware as standard features in their vehicles [VTS, 2018].

Extensive field trials have already been conducted in the USA [USDT, nda] and Europe [simTD, 2018] that show the maturity, and aptness, of the 802.11p standard for V2X applications. From this it appears that for now the 802.11p standard shall remain the de-facto standard for V2X applications.

2.5.5 Suggestions for the 802.11p Standard

It has been suggested that to combat some of the problems with 802.11p, a Self-organised Time-Division Multiple Access (STDMA) approach be used. This works on a self-organised schedule basis, where the nodes in the network create a schedule that they must follow for sending information. This means that all nodes within a network are guaranteed a time slot whereby they can send a transmission that will have a lower probability of being involved in a collision. It was shown in the simulations of [Sjoberg et al., 2011] that although the hidden terminal problem is not a major limiting factor for VANETs, STDMA out-performs CSMA/CA in most aspects due to the synchronisation of message sending.

[Khairnar and Kotecha, 2013] suggested that CSMA/CA may not be suitable for real-time applications due to the uncertainty present in its delays. They also found that in high vehicle den-

sity periods, STDMA was far superior to the CSMA/CA approach originally suggested, due to the scheduling of packet sends. The real-time problem of CSMA/CA has been noted by others [Abdelgader and Lenan, 2014], who suggested a Time-Division Multiple Access (TDMA) approach similar to that of STDMA, where TDMA requires an AP that coordinates all the other nodes of the network, in the case of VANETs the nodes represent the vehicles within communicating range of the AP.

2.5.6 The Nv2 Protocol

The Nv2 protocol is a proprietary protocol developed by Mikrotik for their wireless routers that makes use of TDMA at the MAC layer, thereby requiring one node of the network to act as an AP to set-up the schedule [Gupta, 2012]. In terms of V2X, the AP would most likely be an RSU managing traffic at, for example, a busy intersection. This AP creates the schedule for all of the connected nodes granting each node a timeslot during which it is allowed to transmit a message to the AP, removing the hidden terminal problem from the network. This schedule is then periodically shared with all the nodes. Should a new node join the network, or the location of already connected nodes change in such a way as to affect the propagation delays on the channel, a new schedule is calculated by the AP. This new schedule is then re-transmitted throughout the network.

By making use of a schedule, the nodes in the network do not need to make use of ACKs or RTS/CTS messages, reducing overhead, increasing throughput, and adding more control over the latency experienced by each node. Although similar in some aspects to the 802.11 standard released by the IEEE, to make use of the Nv2 protocol requires that all connecting nodes in a network make use of Mikrotik's RouterOS. This means that a device operating with the 802.11 set of standards will not be able to connect to a Nv2 AP. This drawback of the Nv2 protocol, in terms of interoperability, would only present itself in situations where other proprietary protocol devices, namely devices making use of the 802.11p standard, tried to connect to the Nv2 network for two reasons: the first being that the Nv2 protocol operates in the same band ($5.9\ GHz$) as the 802.11p standard, which would result in interference between the devices. Secondly, since Nv2 makes use of a schedule, a device using the 802.11p standard would not be able to assimilate into the network without alterations to the standard as 802.11p does not make use of scheduling.

The 802.11 standard published by the IEEE has been shown to be an incredibly mature standard,

offering many different solutions for countless scenarios, with 802.11p being specifically designed with V2X in mind. With that said, there are still problems with the 802.11p standard, the most detrimental being the noted possibility of unbounded delays due to interference signals and the hidden terminal problem. As a result, the 802.11p protocol was not used in this study when generating the latency model. Following the suggestions of using a TDMA/STDMA MAC layer protocol instead of a CSMA/CA MAC layer protocol to help combat the hidden terminal problem and possibility of unbounded delays, the Nv2 protocol of Mikrotik was used. The exact protocol used however was not of major concern as it was found during simulations that the transport latencies had to be artificially increased quite substantially beyond what was naturally experienced in the wireless network before control was lost.

2.6 Alternatives to the 802.11p Standard

Long-Term Evolution (LTE), Worldwide interoperability for Microwave access (WiMax), and satellite broadband present themselves as possible (future) alternatives to the current de-facto 802.11p standard. A brief discussion of each of these alternatives follows in the form of comparisons with the 802.11p standard. Due to satellite broadband only recently being marked with latencies low enough for V2X communication, the information presented aims to show the benefits of satellite broadband from a V2X communications perspective.

2.6.1 LTE

The LTE standard used by mobile devices, also known as 4G, was deemed inadequate for V2V communications due to its lack of Device-To-Device (D2D) communicating ability. This changed with the release of LTE-12. Release 12 introduced side-link, a D2D communication structure that bypassed the cellular network. Release 14 in 2016 further improved on this with 3rd Generation Partnership Project (3GPP) aiming for more V2V capabilities. This release (release 14) was known as LTE-V2V, and marked the beginning of the 5G standardization in 3GPP.

It has been shown that when the radius of awareness of network nodes is increased beyond a certain size LTE-V2V out-performs the 802.11p standard. This was most likely due to a larger radius of awareness having a higher number of hidden terminals present, a shortcoming of CSMA/CA. LTE-V2V appears to handle the hidden terminal problem quite well [Cecchini et al., 2017]. Figure 2.3

shows how, in simulation, LTE-V2V outperforms 802.11p as the awareness radius is increased for both in-coverage and out-of-coverage scenarios.

One of the primary advantages that LTE-V2V has over the 802.11p standard is that it already has mass deployed infrastructure. There are very few places that do not have cell phone reception. Infotainment systems and insurance companies already make use of LTE-V2V to send and receive media, and monitor driving behaviour respectively.

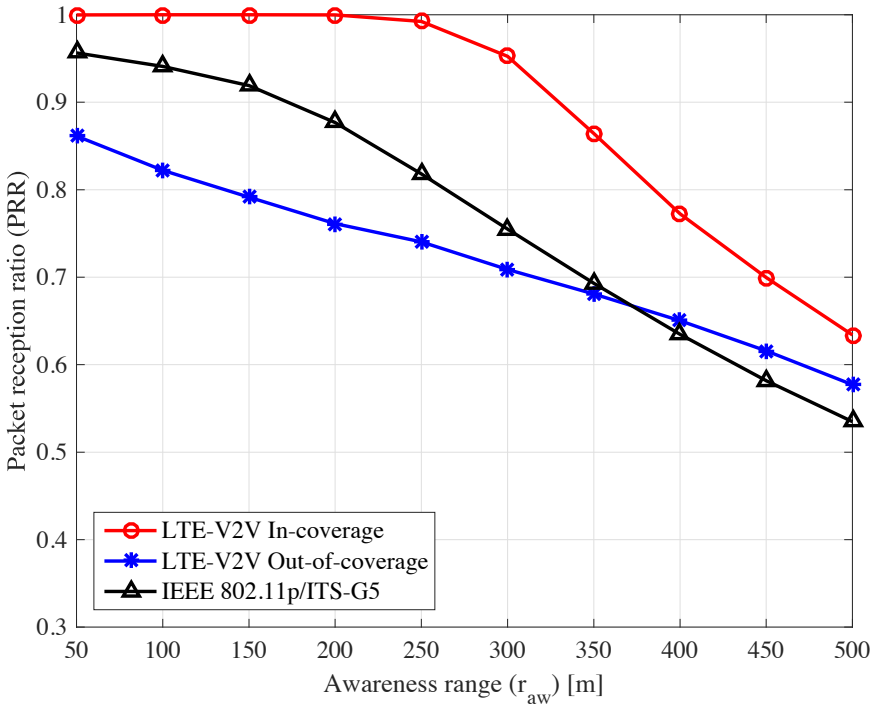

Figure 2.3: A comparison of LTE-V2V with the 802.11p standard when increasing awareness range for a beacon size of 190 *bytes* [Cecchini et al., 2017].

2.6.2 WiMax

WiMax, like WLAN, is based on standards set by the IEEE. Where WLAN is related to the 802.11 family of standards, WiMax belongs to the 802.16 family of standards. The idea of WiMax dates back to the 1990s, when it was realised that there was going to be a massive increase in data traffic around the world and that wired networks were deemed too expensive for last mile deployment. The 802.16 family of standards was first published in 2001 with an initial offering of data speeds of 30 $Mbps$ to 40 $Mbps$ [silicon, 2018]. As with 802.11, there were many amendments made to the original

standard over the years.

One of the main features of WiMax is its range, with data rates of up to $1\ Gbps$ over distances of up to roughly $50\ km$ [Rouse, 2005]. Such data rates can only be reached between stations that are fixed, which is one of the reasons WiMax is not being used for VANETs at the moment. The 802.11p standard does have a much larger operating range than the other 802.11 standards, requiring an operating range of around $1\ km$ [Abdelgader and Lenan, 2014], but it is still well below that of 802.16. Other 802.11 standards generally cap out at around $100\ m$ depending on the exact standard being utilised [RFWirelessWorl, nd]

2.6.3 Satellite Broadband

SpaceX has recently received approval from the Federal Communications Commission (FCC) to launch two massive satellite constellations, one in Low Earth Orbit (LEO) and the other in Very Low Earth Orbit (VLEO), in Non-GeoStationary Orbit (NGSO). The purpose of these constellations, collectively dubbed Starlink, will be to provide the world with robust, high speed, broadband services [Albulet, 2017]. One of the key features of the VLEO constellation will be its high speeds of up to a reported $1\ Gbps$ with incredibly low latencies of $25\ ms$ to $35\ ms$, faster than current copper cable networks. This will be made possible largely due to the lower orbits of the VLEO constellation.

Satellite broadband that is already in place makes use of higher altitude constellations, with the reason being that fewer satellites are needed to cover a larger area with services if they are higher up, resulting in latencies of $600\ ms$ or more [arsTECHNICA, 2018]. These higher latencies make current broadband systems inadequate for V2X communications when considering the $50\ ms$ to $100\ ms$ latency requirements of CAMs.

2.7 Common Driver Models

Presented below are two common driver model classes that are used by controllers to enable accurate and stable path following of a vehicle. The first is a class of driver model based on human characteristics and behavioural patterns, designed to mimic the human driver as closely as possible. The second class of driver model presented is based more on information about the vehicle than the driver. This second class is broken up into three sub-classes: geometric, kinematic, and dynamic, each of which is briefly looked at showing that neither of these models is always better than the other two as each

of them experiences a compromise between low speed accuracy and high speed stability for different reasons.

2.7.1 Human Based Driver Models

A human based driver model has its operating principles based on the characteristics exhibited by a human driver when controlling a vehicle. The controller is designed to react in the same way as a human (driver) would in any given situation. This goes beyond just steering and throttle control of the vehicle. A human driver is capable of path planning, identifying objects surrounding the vehicle, judging the intentions of surrounding vehicles, as well as many other tasks. A lot of the tasks that a human driver completes while controlling a vehicle can be considered beyond that of pure vehicle control, path planning for example, but at the same time the outputs of these extra tasks can be used to further improve on the driver's control of the vehicle. Such an example would be adjusting the look-ahead distance based on the path that is to be followed. For a straight path the driver can make use of a larger look-ahead distance, but a path filled with close-knit sharp turns requires a smaller look-ahead distance. This only hints at the true scale of trying to create a driver model based on human characteristics.

One of the key features human drivers have over many artificial models is their ability to adapt to changes in the vehicle's dynamics and the environment surrounding the vehicle. This level of adaptability is difficult to reproduce in human driver models, and even more so in vehicle-based driver models. Many common controllers are subject to a compromise between low speed tracking accuracy and high-speed stability, a compromise that humans are not subject to, up to a certain point. A human based driver model should aim to avoid this compromise if it is to truly mimic the ability of a human driver. To do this requires a complete understanding of the characteristics of a human driver, some of which are highlighted below [Macadam, 2003]. Even from this incomplete list one can see that modelling a human driver requires a very involved system, made up of many smaller subsystems. From visual and auditory cues to delays between each subsystem, the versatility of the human driver is not a simple task to recreate.

1. Time delays: Introduced by delays in the firing of neurons, muscle reaction times, as well as processing delays, to name a few. This characteristic can be seen as a major short-fall of human drivers when compared to models that are designed to minimise delays in every aspect. Some

sources of delays are counteracted by the human driver by using a look-ahead approach, or previewing information available to them.

2. Visual information: Considered the most important characteristic used by a human driver when controlling a vehicle, where the driver is able to extract velocity and position independently from a scenario.

3. Motion influences: The vestibular system and kinaesthetic channels of the human driver allow for better control performance when supplementing other characteristics.

4. Auditory input: Seen as more of a supplement to the visual channel, where the auditory inputs reinforce the decisions based on visual inputs.

5. Prediction: The ability of a human driver to anticipate future events is one of the more difficult characteristics to replicate in a model. Partly due to the fact that a lot of predictions are based on past experiences/learned behaviour.

An all-encompassing human based driver model should not exclude the effects of vehicle dynamics. In reality the driver and the vehicle are in a constant feedback loop with the driver reacting to changes in the vehicle, and the vehicle reacting to the actions of the driver. When constructing a human driver model, there are also human characteristics that designers will avoid. One such example is the ability of a human driver to be distracted from the task of controlling a vehicle, a highly undesirable trait in controllers as well as humans controlling a vehicle.

2.7.2 Vehicle-Based Driver Models

The three main vehicle-based driver models are geometric, kinematic, and dynamic. [Snider et al., 2009], using the three driver models presented here, conducted a series of tests where each model was used with a controller to follow a series of paths. He found that of the three models, none of them performed the best in all scenarios. For low speeds, resulting in kinematic motion of the vehicle, a simpler model such as the bicycle kinematic model proved adequate. At moderate speeds, the linearised dynamic bicycle model was able to capture the dynamics of the vehicle well enough to allow for accurate path following. This simple linearised dynamic model, however, did not perform very well at very low speeds or tight manoeuvres at very high speeds. The geometric model, making

use of the Stanley method, performed on-par with the other two models until the vehicle speed was increased significantly.

Geometric Driver Models

The simplest form of driver model discussed here. The dynamics of the vehicle being controlled are not included in the geometric formulation and neither are the kinematic equations of motion of the vehicle. An example of such a model is the pure pursuit algorithm [Coulter, 1992], a tracking algorithm that calculates the curvature required to get a vehicle from its current position to a required position. This curvature is calculated using the geometric relationship between the vehicle's current position, the desired position, and the arc of a circle, depicted in Figure 2.4.

The important steps for the general implementation of the pure pursuit algorithm are:

1. Obtain the current location of the vehicle.
2. Find the nearest path point to the vehicle.
3. Find the goal point at some distance ahead of the vehicle.
4. Calculate the curvature that will get the vehicle from its current point to the goal point.
5. Set the steering angle of the vehicle so the vehicle follows the calculated curvature.

This shows that none of the kinematics or dynamics of the vehicle are required. In fact, the curvature is calculated using no vehicle parameters at all. While this makes the geometric driver model very easy to implement, it limits the ability of the controller to very small operating regions.

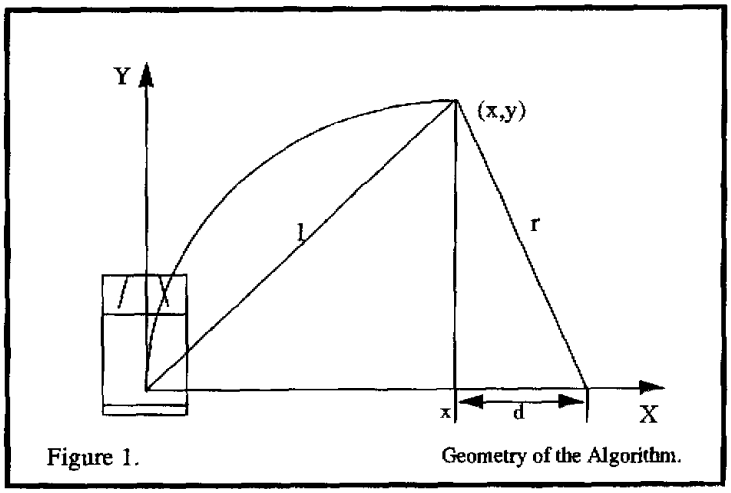

Figure 2.4: Visual representation of the pure pursuit algorithm presented in [Coulter, 1992].

[Snider et al., 2009] showed that if the controller gains that were selected for low speed operation were again used during higher speed manoeuvres, vehicle control was lost. Conversely, high speed gains could not adequately control the vehicle at lower speeds, highlighting the compromise between high speed stability and low speed accuracy. Extensions to this very simple pure pursuit model incorporate the geometry of the vehicle, the wheelbase and the steer angle of the vehicle for example, such as the simplified Ackerman geometric model.

Kinematic Driver Models

The kinematic driver model is an extension to the geometric driver model, but still a simplification of the real-world vehicle. Under the assumption of no slip at the tyre-road interface, the kinematic equations of motion for a vehicle in planar motion are used to generate a model. A common form of the kinematic driver model is the bicycle model, shown in Figure 2.5, which itself is a simplification of a four-wheeled vehicle.

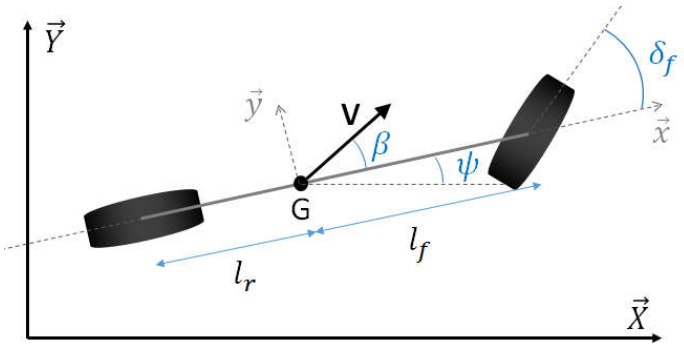

Figure 2.5: Visual representation of the kinematic bicycle model [Polack et al., 2017].

The two front wheels are combined into one wheel, as are the two back wheels, resulting in a model that only has two wheels, hence the name. The steer angle of both front wheels is assumed the same, in contrast to that of an Ackermann Steering system where the front right and left wheels are able to achieve different steer angles when going through a turn. To model such a set-up (with Ackermann Steering) more accurately would require a slightly more complex model than the bicycle model, e.g. a four-wheeled model.

Although controllers making use of kinematic driver models tend to perform better than their geometric counterparts, the assumption of there being no slip at the tyre-road interface results in the same compromise between low speed accuracy and high-speed stability. This is as a result of a vehicle travelling at lower speeds experiencing only trace amounts of slip, in accordance with the no slip assumption, but as the speed of the vehicle is increased slip starts to affect the response of the vehicle substantially.

Dynamic Driver Models

The kinematic model is based on the assumption that the tyres do not experience slip between the tyre surface and the terrain. This assumption, while valid at moderate speeds on good terrain performing simple manoeuvres, becomes invalid at higher speeds, on slippery terrain, and when manoeuvres require large lateral accelerations to be performed. Therefore, to account for the tyre slip it is essential to take tyre forces into account, which is accomplished using the dynamic driver model. Again, a simplified bicycle model can be used, shown in Figure 2.6, or a full vehicle model

that takes into account all the dynamics of the vehicle, including the dynamic equations of motion. The dynamic model introduces the effect of forces, forces which are often non-linear functions of vehicle states. Due to the computational complexity added by the inclusion of the vehicle dynamics, often certain dynamic relations are linearised. This linearisation can impact the performance of a controller using the dynamic driver model quite severely.

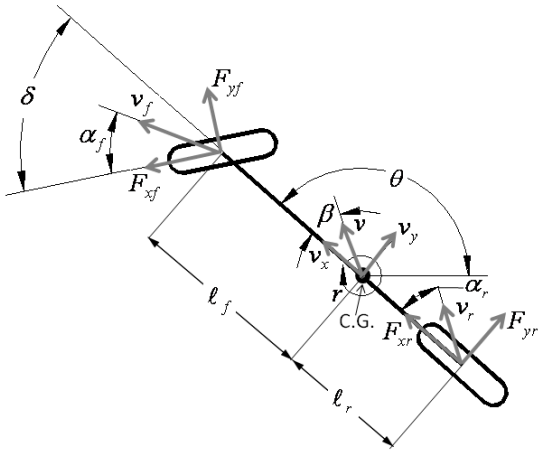

Figure 2.6: Visual representation of the dynamic bicycle model [Snider et al., 2009].

Although it is possible to fully characterise all the dynamic forces present on a vehicle, this is a very time-consuming task that can be computationally expensive. A change in the vehicle's parameters has the propensity to alter the state of the vehicle entirely. For example, a change in a vehicle's mass can be measured with relative ease, however this will also affect the centre-of-gravity and mass moment of inertia of the vehicle as well as all other forces which are functions of the vehicle's mass. This is especially true for load-carrying mining vehicles. Not only are the vehicle forces affected by the mass of the load, the load distribution can also play a major role. Other vehicle parameters are also very difficult to measure which directly affect the vehicle handling. These include road friction and changes in tyre properties. Due to parameters varying during the course of the vehicle's life, modelling these aspects as constants may significantly reduce performance of the driver model in certain conditions. The non-linear vehicle dynamics also significantly increase computational complexity and ability to use the model in driver models. To reduce the computational load of the dynamic model certain simplifying methods can be used, one of which is linearisation.

An example of a dynamic driver model is one in which the tyre-road interface is incorporated into

the model. These equations describing the tyre-road interface are often linearised [Snider et al., 2009]. A dynamic model making use of a linearised tyre model can be expected to perform well while the vehicle is operating in the linear region of the tyre but should the speed of the vehicle increase or the path being followed contain tight corners, the tyre will no longer be operating in its linear region which would likely lead to a loss of control. This loss of control is a result of the linearised dynamic equations no longer being representative of the true dynamics of the tyre.

[Snider et al., 2009] found that the dynamic driver model coupled with some optimal control method performed reasonably well in a path tracking simulation. However, the model made use of certain simplifications and linearization techniques that restricted its operating region. The tracking simulation was only successful if the speed of the vehicle was maintained within a range that was neither too low nor too high, while the path needed to be relatively smooth. This limited operating region came about as a result of the simplifications and linearisation used. The linearisation assumed the tyres of the vehicle would be operating in their linear region, but at higher vehicle speeds, or sharp turns, the tyres were no longer operating in their linear regions. In conclusion, reducing the complexity of the model results in a reduced controller operating region.

2.8 The AutoRegressive Model with eXogenous Inputs

In Chapter 2.7 it was shown that the three driver models: geometric, kinematic, and dynamic, all experience some form of compromise between low- and high-speed control. For the dynamic driver model this compromise comes about when reducing the complexity of the model, through either simplification or linearisation. The use of linearisation in the dynamic driver model reduces the models effective operating region as the model is only valid under certain conditions.

The AutoRegressive model with eXogenous inputs (ARX-model) of [Kapp, 2014] addresses this problem by updating the vehicle model based on the vehicle's states at previous time steps. This formulation removes the necessity of a fully characterised vehicle model being available a priori by calculating the required parameters as the vehicle is travelling. Should the vehicle be travelling at low speeds with lower lateral accelerations, the vehicle model is adjusted accordingly, and should the vehicle's speed increase resulting in an increase in its lateral accelerations, the vehicle model will automatically adapt. This grants the controller access to a dynamic model that is not limited to a set operating region, removing the compromise between low speed accuracy and high-speed stability.

2.9 Conclusion

With regard to how the channels in the $75\ MHz$ band have been organised by the NHTSA, it was shown that there are two main groups. The first being the SCHs, for transmitting less safety critical data, and the CCH, for transmitting more safety critical data. Neither of these channels have been explicitly specified to handle real-time off-site vehicle control, which, in terms of occupant safety, and subsequent level of importance, ranks above both the SCHs and the CCH. For this reason, when it comes to wireless vehicle control, it may be more suitable to set aside a separate channel for off-site vehicle control applications.

The other options presented as suitable future alternatives to the 802.11p protocol are LTE-V2V and satellite broadband solutions, with the satellite broadband option not quite mature enough as lower latency offering constellations have yet to be put in orbit, and the LTE-V2V standards not quite fully developed yet. Of these two, satellite broadband will undoubtedly draw a lot of attention in the future. It has the potential to offer a worldwide network without the need for ground-based infrastructure, meaning no RSUs, and fewer dead-zones where there is no available network. It would also be able to give vehicles/VRUs access to far more powerful computing capabilities than what individual on-board processors will ever be able to offer.

Four vehicle-based driver models were presented, three of which were known to suffer from a compromise between low-speed accuracy and high-speed stability. The fourth driver model was presented as a solution to this compromise, and as a result was used in this study. It had already been shown to enable robust control over a wide range of operating conditions, including while travelling on a loose gravel road, without the need to have pre-determined parameter values. The specific mathematics of the model are presented in Chapter 4.

2.10 Research Focus

Although vehicle control is a well-documented field of study, with many controllers existing for all kinds of operating conditions, almost all of the work presented is aimed either at a system with an on-board controller (self-driving cars) or, an off-site controller that is under complete human supervision, and for autonomous off-site control most of the work is centred around underwater vehicles [Triantafyllou and Grosenbaugh, 1991]. With the increase in capabilities of ADAS, and the move

towards self-driving vehicles, systems such as Starlink and the like present an immense opportunity to provide such upgrades (ADAS, self-driving) on a global scale.

Before such global upgrades can be realised, a complete understanding, or as complete as possible, of how vehicles respond in such latency-laden environments is required. One step towards this understanding is determining how much latency can be present between reading a vehicle's states and applying an off-site-calculated control action, as well as the amount of data that can be lost as packet drops between the vehicle and its controller. This study aims to shed some light on the effect these latencies and packet drops can have on the system as a whole, while presenting a controller that is able to accommodate said packet drops and latencies.

Chapter 3

Latencies in WLANs

Latency in a network is unavoidable. The act of carrying information/data from one location to another with current technologies will always have a minimum latency greater than zero. This is true for any data transfer that occurs, even between sensors and their controllers. Networks that make use of fibre optic cables, exhibiting incredibly low latencies, still experience transport latencies of roughly $490\ \mu s$-per-$100\ km$ [Miller, 2012]. A lot of these smaller latencies, i.e: within sensor networks, can be safely ignored or are small enough to be assumed zero. When it comes to WLANs however, transport latencies between nodes are not so safe to ignore. This is especially true in the case of VANETs, where latencies can mean the difference between life or death. This is only exacerbated when considering vehicle control via a wireless network, as opposed to on-board control. For this reason it is important to understand what causes these latencies, as well as the technologies available to reduce them.

The following chapter has a look at what causes latencies in WLANs, and by extension VANETs, as well as some methods that are currently being used to reduce them where possible. This is followed by the experimental procedure followed to gather latency data of a WLAN making use of the Nv2 protocol, the results of which were used to generate a latency model to be used during simulations. It was found that the assumption of a constant latency did not accurately capture the true nature of latencies in a WLAN using the Nv2 protocol. Finally, the resulting latency model is presented with conclusions that were drawn.

3.1 Latency Sources in WLANs

The latency a packet experiences travelling through a network does not come from a single source, rather it is a combination of many smaller latencies, added at different sections of the network. The four main sources of these smaller latencies in a WLAN are discussed next. Figure 3.1 shows where each of these individual sources of latency occur, considering the set-up that was used during the generation of the latency model. The green rectangles are where propagation delays occur, along the wired and wireless channels, the magenta circles are where the processing delays occur (on the Mikrotik routers), and the grey ellipsoids are where the transmission delays occur, between the Mikrotik routers and the wireless channel, the processing units and the wired channels, and the Mikrotik routers and the wired channels.

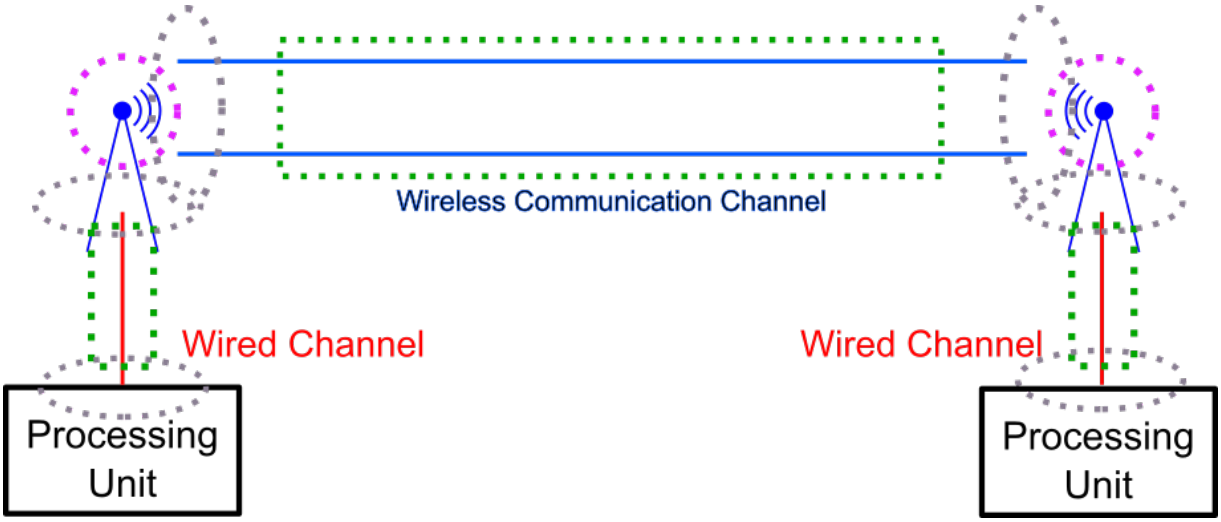

Figure 3.1: Depiction of the locations of the main sources of latencies in a wireless network.

3.1.1 Propagation Delays

The propagation delay is the amount of time it takes for information to travel across a channel, for VANETs this is the air between the transmitter and receiver. In an ideal scenario the speed of propagation is the speed of light, or $3 \times 10^8\ m/s$. Since VANETs are located on Earth, for the time being, and the channel is not the vacuum of space but rather the lower atmosphere, the propagation speed will always be lower than this value. There is very little that can be done to decrease propagation delay, it is an effect that needs to be considered and designed for in WLANs.

Fortunately, the effects of propagation delays when distances between transmitters and receivers are relatively small (up to a few kilometres) are largely negligible. Only when working with larger distances do propagation delays start to truly impact the network's performance [Lopez-Aguilera et al., 2010].

3.1.2 Transmission Delays

The time it takes to put a packet of information on a channel, the channel being the air for a WLAN or a copper wire in a Local Area Network (LAN), is known as the transmission time. Put more elegantly, the time between the first data bit leaving the transmitting node and the last data bit leaving the transmitting node. Transmission time is a function of packet size and the channel's bit-rate, as shown in Equation 3.1.

$$\text{Transmission Time} = \frac{\text{Packet Size}}{\text{Bit-rate}} \tag{3.1}$$

3.1.3 Processing Delays

Before a packet of information can be put onto a channel it needs to be processed. In a standard home network, comprised of computers and - usually - one router, this processing of the packet occurs on the computer, before being sent to the router, and again on the router, before being sent to its destination. Once the packet has been received by the router bit-level errors can be checked - this is to ensure that no transmission errors occurred when the packet was put onto the channel. The destination of the packet must also be determined, by reading the header of the packet, before it can be sent to its destination. If the packet is encrypted then the encryption process, as well as the decryption process, adds to the total processing delay.

To relate this to the wireless latency tests conducted in the generation of the latency model, there were four locations for processing delays during a packet's life-cycle: At each of the processing units, as the messages were encrypted and decrypted for analysis, and at each of the Mikrotik routers, as they were analysed to determine their required destination.

3.1.4 Packet Drops

Packet drops in networks that make use of ACKs are another source of latency, as more time is spent resending a dropped packet, adding to the packet's overall experienced latency. Since the connection

between the processing units and the Mikrotik routers made use of the User Datagram Protocol (UDP) and the Mikrotik routers used the Nv2 protocol, resending of dropped packets did not occur during all tests conducted in this study. Both UDP and the Nv2 protocol do not make use of ACKs. This meant that if a packet was dropped, it did not have a latency value as the packet never arrived at a receiver since the transmitter never attempted to resend the packet.

3.2 Simulating Real World Latencies

As mentioned in Chapter 2.5.6, the Nv2 protocol by Mikrotik was to be used in this study and as such, a latency model needed to be generated to conduct simulations that were more representative of the real-world. Following is a brief description of the experimental procedure followed to generate said latency model. The resulting latency model was used in simulations to determine the effect, if any, the Nv2 protocol had on the subsequent controllability of the vehicle when simulating an off-site controller.

3.2.1 Latency Model Experimental Set-Up

Using two Mikrotik routers and two processing units, a series of latency tests were conducted to determine what real-world latencies had to be recreated for simulation purposes. First some hardware specific details are given, followed by the procedure carried out to acquire latency data of the network.

Latency Model Hardware Details

The details of the processing units and Mikrotik routers that were used during the acquisition of network latency data are presented here. The properties of the two processing units are summarised in Table 3.1, with the properties of the Mikrotik routers given in Table 3.2. The Mikrotik routers making use of the Nv2 protocol were set up to operate near the $5.9\ GHz$ band, $5.805\ GHz$ to be exact, with a channel bandwidth of $20\ MHz$ - much like a single SCH, and a minimal TDMA period size of $1\ ms$.

Latency Model Experimental Procedure

The tests were conducted using stationary placements for the AP and the node, with a clear line of site between the two (the AP being a requirement of the Nv2 protocol). The distance between the two

placements started at $5\ m$ and was gradually increased for each test until the distance reached $700\ m$. A direct wired test was also conducted as a baseline test, the two processing units were connected with an Ethernet cable directly, with no intermediate Mikrotik routers. Each test lasted 30 minutes and was run three times at each distance to ensure the results obtained were truly representative of a network making use of the Nv2 protocol. A UDP set-up was used between the processing units and the Mikrotik routers, much like the Nv2 protocol this meant that there was no resending of dropped packets, since UDP is a connectionless protocol that does not wait for an acknowledgement.

Table 3.1: Processing unit used during the latency model generation.

Property	**Value**
Model	Raspberry Pi 3 Model B
Processor	Broadcom BCM8387 chipset (1.2GHz)
RAM	1GB
OS version	Linux 4.9.68
Real-Time patch	rt56-v7
Ethernet connection	100 $Mbps$

Table 3.2: Mikrotik router used for latency model generation.

Property	**Value**
Model	Groove 52 HPn
Processor nominal frequency	600 MHz
Wireless frequency	2.4 GHz/5 GHz
Ethernet connection	100 $Mbps$
Protocol	Nv2
Period length	1 ms

One processing unit was set up to transmit $100\ Byte$ packets at a rate of $100\ Hz$ (one every $10\ ms$), while the other processing unit was set to receive the packet and return it immediately to the original processing unit. The path along which a single packet travelled through the latency test network is described here for clarity. The packet was sent from the first processing unit to the first Mikrotik router, acting as the AP, along a wired channel. The first Mikrotik router then processed the packet and sent it to the second Mikrotik router, acting as the node, across the wireless channel.

The second Mikrotik router then processed the packet and sent it to the second processing unit along a wired channel. The packet was then analysed by the second processing unit, to ensure it was a part of the tests, and returned to the first processing unit following the same steps, but in reverse.

Due to the difficulty in synchronising the two processing units' clocks with each other, the round-trip latency/round-trip time (RTT) was measured instead of measuring the one-way latency. In WLANs there is no guarantee that one-way latency $= \frac{RTT}{2}$, as asymmetrical transmission times can occur, so RTT was used as the baseline for the latency calculations. To calculate the RTT, the $100\ Byte$ message included a time stamp of when the message was sent. Upon the message returning to the original processing unit, this original time stamp was compared with the current time on the processing unit and from this comparison the RTT was acquired. Thus the RTT value included the time to package, send, and unpack (decode) the message on both processing units. Any latencies referred to for the remainder of this chapter will be in reference to the RTT, however, in Chapter 5 latencies were added on a one-way basis.

3.2.2 Latency Model Experimental Results

The results of the wired (Figure 3.2), $20\ m$ wireless (Figure 3.3), $150\ m$ wireless (Figure 3.4), and $700\ m$ (Figure 3.5) wireless tests are given, with a statistical summary of all the tests conducted given in Table 3.3.

Table 3.3: Statistical summary of latency tests' RTT results.

Test	**Mean** $[ms]$	**Standard Deviation** $[ms]$	**Maximum** $[ms]$	**Minimum** $[ms]$
Wired	0.5437	0.0099	0.6988	0.5263
5m Wireless	2.9171	0.3979	8.0924	1.9975
20m Wireless	2.9298	0.4451	15.2744	1.9748
150m Wireless	4.8076	2.5105	41.6886	2.1291
500m Wireless	4.9331	2.4948	47.7081	1.9608
700m Wireless	5.0187	2.7865	77.1714	2.0012

When looking at the minimum latency it can be seen that, when comparing Figures 3.2 and 3.3, the addition of the wireless routers resulted in an RTT of at least $1.9\ ms$ - an addition of $1.4\ ms$. The general pattern appeared to be that as the distance between the antennas was increased, the mean RTT increased, and so did the noise around the mean RTT. When looking at the histograms of the results, another pattern emerged. Along with an increase in the mean RTT and the surrounding

noise, the distribution of the RTT changed. Although this change was not very clear when comparing the wired and $20\ m$ wireless results, it became evident when comparing the $20\ m$ wireless results with the $150\ m$ wireless results. In the $20\ m$ wireless results the RTT distribution can be likened to a normal distribution centred around a single mean value. However, for the $150\ m$ wireless results, a single distribution would not be able to accurately capture the nature of the RTT distribution. Rather, the single distribution was split into three distributions, each of which was centred around its own mean value. These mean values did not correspond to the single mean value of the entire dataset, shown with the red dashed line.

The three distinct distributions (not just random scattering), as well as the minimum $1.9\ ms$ RTT value, came about as a result of the TDMA schedule set up by the Mikrotik routers. The processing unit that sent out the packets initially was likely not synchronised with the TDMA schedule, i.e. when the packet left the processing unit, it would have had to wait at the router until the router was allowed to send off another packet according to the schedule. Since the TDMA schedule had a period size of $1\ ms$, where the Mikrotik router would send out a message every millisecond, and the rate at which packets were sent from the processing unit was $100\ Hz =$ every $10\ ms$, the two systems remained out-of-sync throughout the tests. This would also happen on the packet's return trip.

Mathematically, if the mean of the wired tests, roughly $550\ \mu s$, is considered, the addition of the Mikrotik routers added two more transmission times, from the router acting as the AP to the wireless channel, and from the router acting as the node to the wireless channel. The amount of time added by each of these transmissions, based on a $100\ Byte$ packet size and a bit-rate of $4\ Mbps$ (the packet size used in determining the latency model and roughly the bit-rate of the Nv2 protocol), was calculated as $200\ \mu s$ using Equation 3.1. This resulted in an additional $400\ \mu s$ added to the roughly $550\ \mu s$ and combined with a $1\ ms$ out-of-sync waiting period gave the roughly $1.95\ ms$ minimum delay experienced by the addition of the routers.

The exponential decay noticed in the histograms of Figures 3.4 and 3.5 suggest that the probability a packet missed a chance to be sent from the router according to the TDMA schedule reduced drastically with each TDMA period completion as the majority of the packets only missed the scheduled send time once.

When comparing the $150\ m$ wireless results with the $700\ m$ wireless results, there was not much of a change, the three distributions were roughly centred at the same mean values. For the $700\ m$

wireless results, the beginnings of a forth distribution set can be seen around $14\ ms$ to $15\ ms$.

For all tests conducted, from wired to $700\ m$, there were no lost packets. This showed that the Nv2 protocol, coupled with the Mikrotik routers, was very reliable at getting all the required data across the wireless channel, even without ACKs. This meant that any simulations done to represent packet loss would require an entirely artificial dropped packet parameter as no model could be obtained from the experiments.

The transmission time between the Mikrotik routers and the wireless channel, $200\ \mu s$, was inconsequential, even when compared to the latency results from the wireless latency test of Figure 3.3, where the mean Round-Trip Time was calculated to be $2.9298\ ms$. If a one-way latency that was $2/3$ that of the RTT was assumed, the transmission time would comprise just over 10% of the total one-way latency. For the wired channel tests, Figure 3.2, where the bit-rate was $100\ Mbps$, the same $100\ Byte$ packet would have a transmission time of $8\ \mu s$, a mere 1.5% of the mean RTT of $0.5437\ ms$. A slightly larger one-way latency of $2/3$ of the RTT was selected based on the results of [Wellnitz and Wolf, 2010], where the difference between one-way latency and RTT$/2$ was found to be as large as one third.

When comparing Figures 3.2 and 3.3, it was shown that the addition of the wireless Mikrotik routers alone added roughly $2.4\ ms$ of delay to the mean RTT results (from $500\ \mu s$ to $2.9\ ms$ when going from a direct wired channel, no intermediate nodes, to a $20\ m$ wireless channel using the Mikrotik routers respectively). This showed that the majority of the latency came from the routers themselves. This was due to the router's processing delays. The routers had to handle the packets before they could send them along the channel (wired/wireless) and this handling of the packets contributed the most to the latency experienced in the sending and receiving of packets.

To ensure there were as few delays on the processing units as possible, their Linux-based Debian Operating Systems (OS) had a real-time patch, this allowed for more reliable interrupts than a standard OS without a real-time patch. The response time of each processing unit was then tested and found to average $114\ \mu s$, with a maximum of $138\ \mu s$, while having a schedule priority of 95. The results of the response time tests, when compared to the wireless latency tests (Figures 3.3 to 3.5), showed that the response times of the processing units were insignificant, and did not affect the results of the latency model in any noticeable way.

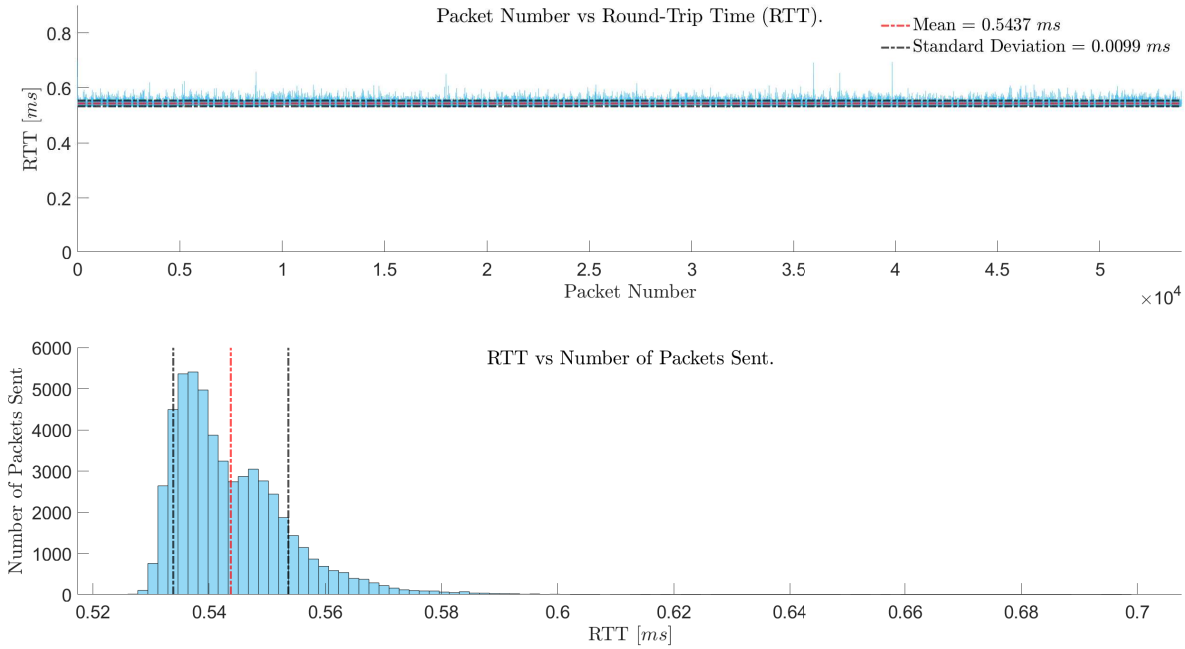

Figure 3.2: Results of the wired latency tests.

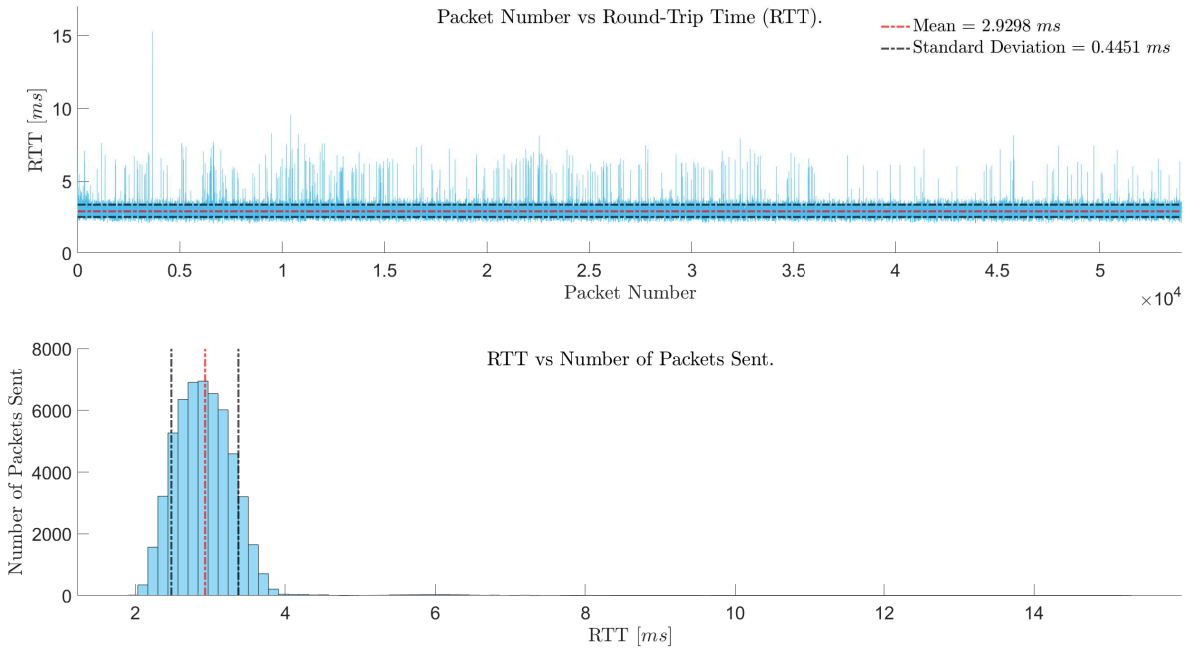

Figure 3.3: Results of the wireless 20 m latency tests.

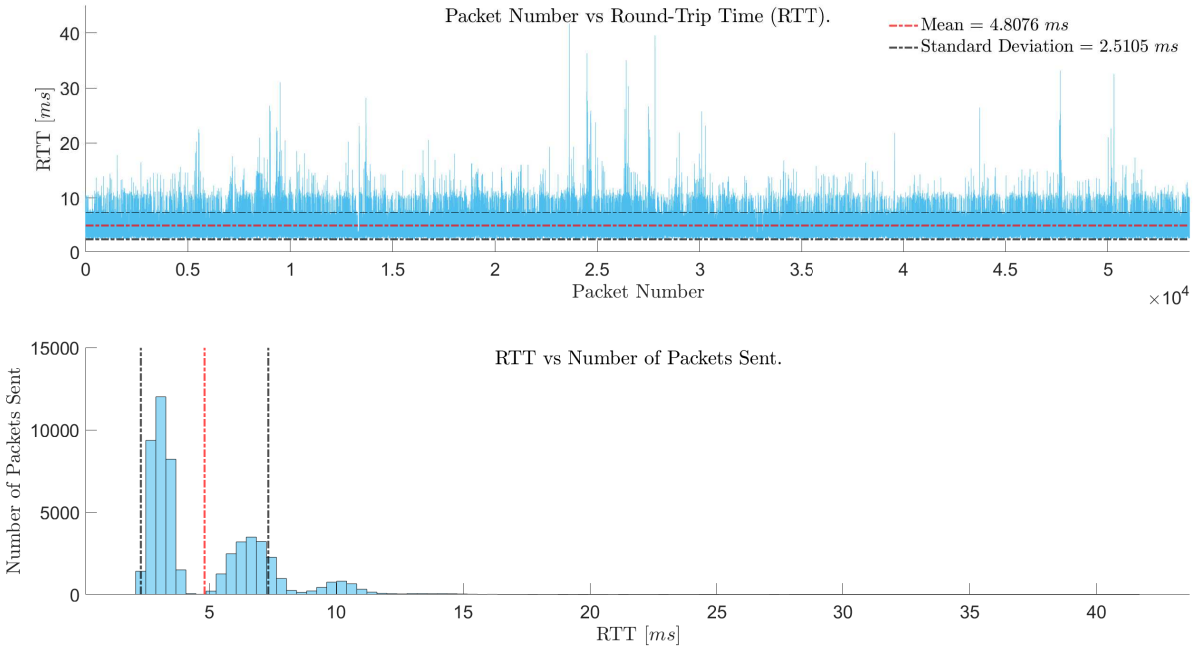

Figure 3.4: Results of the wireless 150 m latency tests.

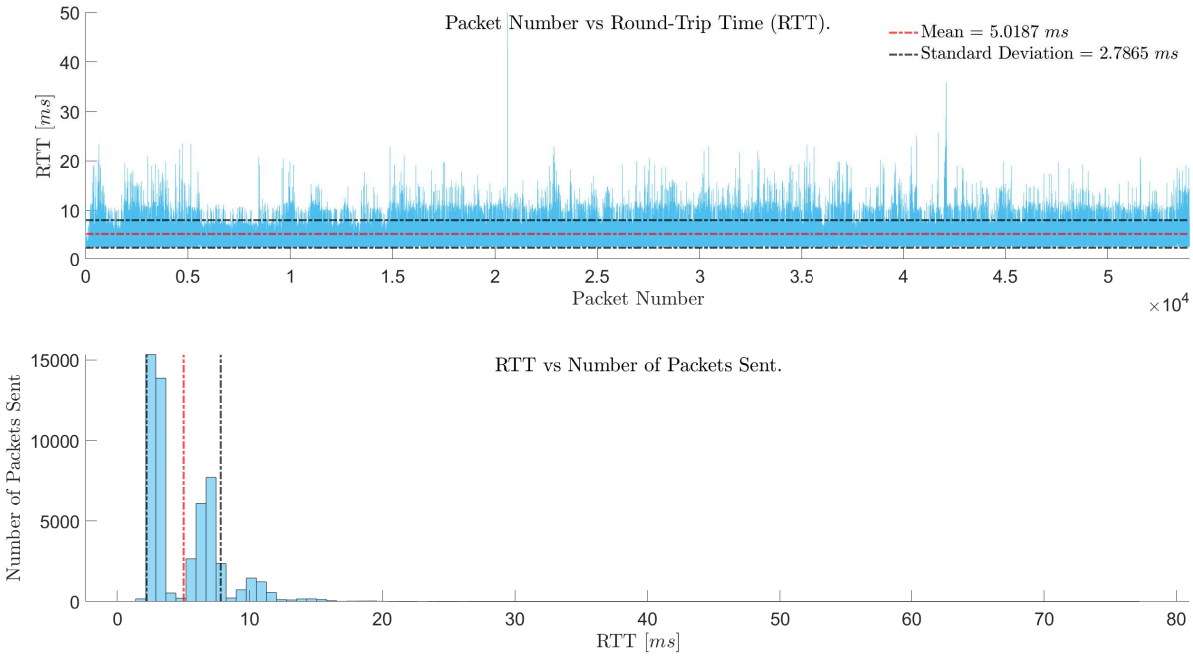

Figure 3.5: Results of the wireless 700 m latency tests.

3.3 Latency Model Generation

Using the results of the previous section, a latency model was generated. For the generation of the latency model, it was decided that the $700\ m$ wireless tests' results would be used exclusively as it had the greatest variation - when compared to the smaller distance tests. This distance was also the closest to the $1\ km$ operating range requirement for the 802.11p standard. Unfortunately, due to space constraints, a distance greater than $700\ m$ between the two Mikrotik routers was not possible when conducting the latency model experiments.

The data of the $700\ m$ test was split in to four distributions. The first was packets that had an RTT of between $0\ ms$ and $4.5\ ms$, the second distribution between $4.5\ ms$ and $8.5\ ms$, and the third distribution between $8.5\ ms$ and $12.5\ ms$. The final distribution was packets that had an RTT larger than $12.5\ ms$, this group was comprised of only 1.4316% of the total packets sent and was clipped from the data set. The mean, standard deviation, and percentage of packets in each distribution was calculated assuming a normal distribution for each, the results of which are given in Table 3.4.

Table 3.4: Latency Model Distribution Values.

Distribution	**Range** $[ms]$	**Mean** $[ms]$	**Standard Deviation** $[ms]$	**Size** $[\%]$
1	$0 < x < 4.5$	2.9113	0.3661	55.4073
2	$4.5 <= x < 8.5$	6.7057	0.6715	35.3963
3	$8.5 <= x < 12.5$	10.4102	0.7877	7.7648

Using these values, a model was generated that would select a latency randomly from one of the three distributions. Figure 3.6 shows a comparison of the clipped wireless $700\ m$ latency results and an artificially generated data set. Although not an exact match, with the experimental distributions not being perfectly normal and the artificial results having a few values below $1.9ms$, the latency model was able to accurately reproduce the real-world results.

If the experiments were to be conducted with larger distances between the Mikrotik routers, the latency model could easily be extended, by the addition of more normal distributions, as Figure 3.5 shows the beginnings of a fourth distribution. This would cause a change in the percentages of packets in each distribution, with packets being shifted towards the right. The tests conducted only considered a clear line of site with stationary nodes. For a more complete/accurate model, the effects

of vegetation/obstructions as well as moving nodes would need to be considered. However, for the purpose of this study the simple model presented was deemed adequate.

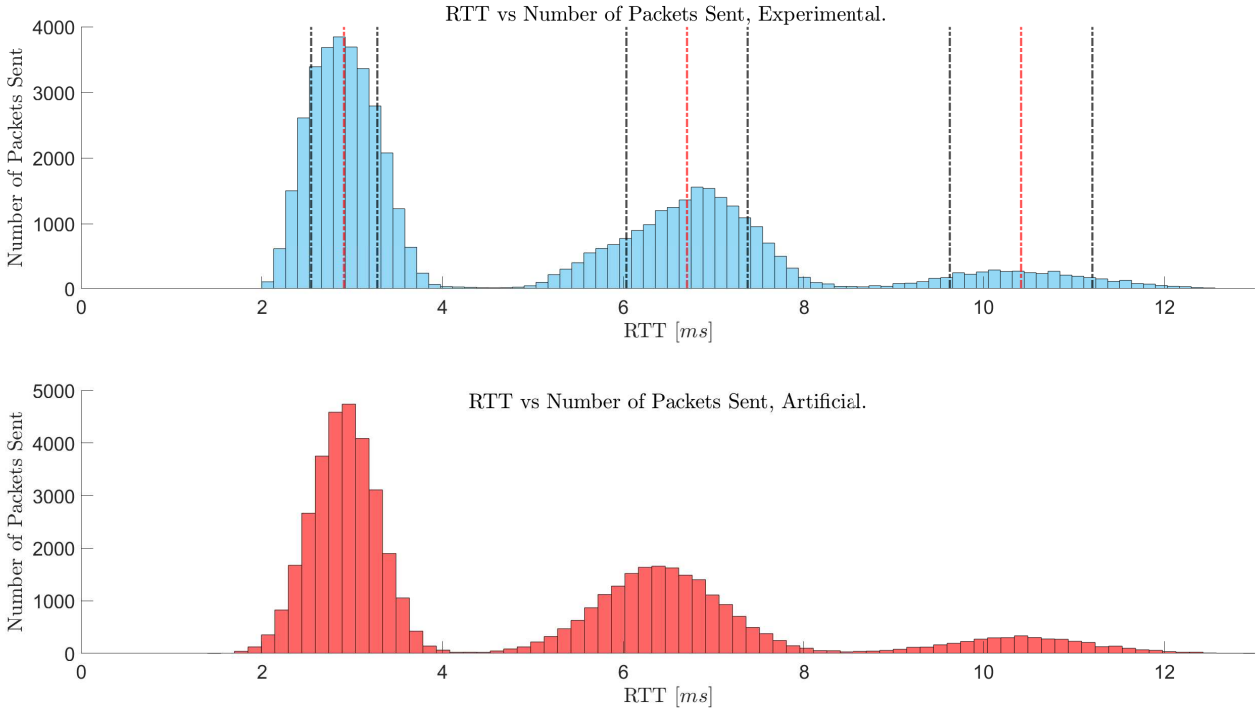

Figure 3.6: Wireless latency model, experimental and artificial results for 700 m.

3.4 Conclusion

The common causes of latencies in a network were looked at and it was shown that in the network used to generate the latency model the majority of the RTT was introduced due to processing delays on the Mikrotik routers. The results of the 700 m tests showed a mean RTT of $5.0187\ ms$ with a maximum RTT value of $77.1714\ ms$. From all the packets sent during the 700 m tests, just over 1.4% had RTT values greater than $12.5\ ms$. It has been shown that to accurately simulate latencies in a network using the Nv2 protocol, a model that was slightly more complex than a constant latency was needed. Such a model has been presented and was shown to accurately reproduce real-world RTT latencies in a network making use of Mikrotik routers using the Nv2 protocol.

Chapter 4

Controller Synthesis and Analysis

The following chapter details the theory behind the construction of the controller that was used during simulations. A controller that was robust in its handling of unknown changes in the environment, while remaining stable, was sought, to be used for on-board control and off-site control. The off-site controller was subjected to both transport latencies as well as packet drops, where an ideal scenario had no packet drops but a minimum transport latency greater than zero, as shown in Chapter 3.2.

It was believed that a controller meeting these criteria, robust in changes in environment and stable, would be capable of safely and effectively handling the vehicle in a larger range of uncertainties, which are inherently present when working with transport latencies and packet drops. The controller designed and tested in [Kapp, 2014], henceforth referred to as the original controller/ARX-model, was shown to be very robust while maintaining a fairly high path following accuracy as its driver model - the ARX-model - was platform independent and updated itself while the vehicle was being controlled. It was later shown to handle transport latencies quite well in [Kapp, 2017], with its ability to handle permanent packet drops - where there is no resending of a dropped packet - not yet investigated. Modifications were made to the original controller, henceforth referred to as the modified controller/ARX-model, which are detailed in this chapter, the majority of these changes were made to the driver model used by the controller.

This chapter is organised as follows: First, the ARX-model as how it was originally described to model the vehicle is covered. Next, the implementation of an LQSTR-controller using the state-space method is covered, followed by a brief explanation of the parameters used by the controller.

Information relating to the validated simulation model is then presented. This is followed by the details of the test path that was used as well as the measures of accuracy that were used to judge the effectiveness of the controller. The effectiveness of the modified controller is then looked at, this includes a comparative study whereby the original controller was compared to the modified controller, showing the improvements and shortfalls of the changes made.

Figure 4.1 shows the naming convention used when designing the controller, where ψ is the yaw angle of the vehicle measured from the X-axis, δ is the steer angle of the vehicle's front wheels measured from the X'-axis, X and Y are the axes of the global coordinate system, and X' and Y' are the local coordinate axes with zeros at the centre-of-gravity of the vehicle.

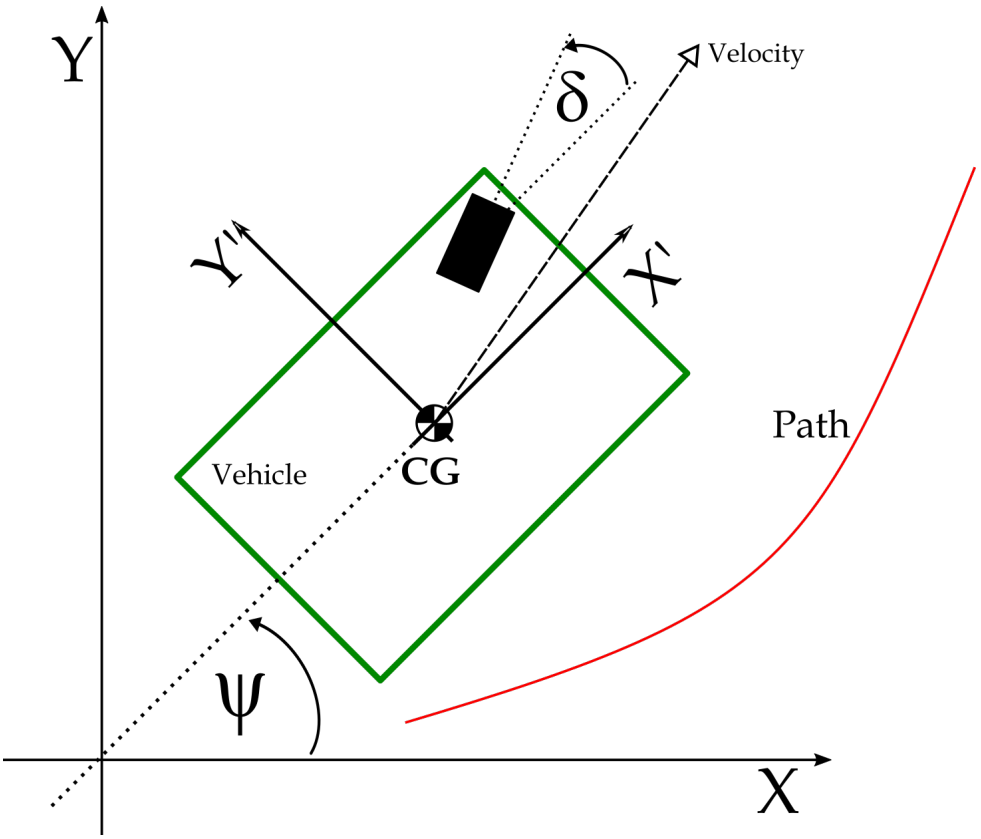

Figure 4.1: Vehicle parameter convention showing steer angle and yaw angle of the vehicle.

A positive angle direction was measured as anti-clockwise from the respective X-axis. It was assumed that both front wheels experienced the same steer angle during a turn. The yaw angle of the vehicle (ψ) was used to describe the vehicle's heading with respect to some reference heading: the X-axis in Figure 4.1 and simulations. The yaw angle itself was not explicitly used during model

generation, rather the yaw rate ($\dot{\psi}$) was used, which is the rate of change of the yaw angle, along with the steer angle of the vehicle (δ).

4.1 The ARX-Model

Primarily used in economics, an Auto-Regressive (AR) -model is a model in which the future values of a time-varying random process are determined using a weighted sum of the past values of the process. In [Kapp, 2014], this method was used to model the plant (vehicle) instead of linearising around set operating points. To expand on the AR-model, the effect of exogenous inputs (external excitation variables) was included, resulting in an ARX-model. It was noted in [Thoresson et al., 2014] that the relationship between the steer rate (the rate of change of the steer angle) of a vehicle's front wheels and the vehicle's yaw acceleration, and by extension the steer angle and yaw rate, could be modelled using the Magic Tyre Formula. It was this relationship, between the steer angle of a vehicle and its yaw rate, that was used as the time-varying random process to be estimated by the ARX-model.

Although ARX-modelled estimates are more suited for linear systems, whereas a vehicle's response is generally non-linear, if a small enough time-step is used, then an approximately linear system between the time-steps is realised. The general equation of the expanded ARX-model is given in Equation 4.1, where $\widetilde{Y}_t$ is the estimated value of the model, Y_{t-i} are the autoregressive terms, s is the number of autoregressive terms, i.e. the number of previous time-steps' values to be used in the calculation of the current model estimate, u_{t-i} are the exogenous input terms, g is the number of exogenous input terms, φ and η are weighting variables, and ε_t represents the unmodeled residuals that are invariably present in most real processes.

$$\widetilde{Y}_t = \sum_{i=1}^{s} \varphi_{i-1} Y_{t-i} + \sum_{i=0}^{g} \eta_i u_{t-i-n} + \varepsilon_t \tag{4.1}$$

This time domain equation can be represented in the discrete domain by the transfer function shown in Equation 4.2, where the unmodeled residuals are dropped, the numerator represents the exogenous inputs, and the denominator the previous samples.

$$H(z) = \frac{Y(z)}{U(z)} = \frac{\sum_{i=0}^{g} \eta_i z^{-(n+i)}}{\sum_{i=0}^{s} \varphi_i z^{-i}} \tag{4.2}$$

4.1.1 ARX-Model Order

In the design of the original controller, it was suggested that there be two discrete poles present in the transfer function of Equation 4.2 for the following reasons: the first pole would allow the model to capture the dynamics of the vehicle when the tyre was operating in the linear tyre region while the second pole increased the dependency of the model on previous samples to accommodate operation in the non-linear tyre region. However, while recreating the results obtained with the original controller it was noted that the weighting that was calculated for the second pole (K_2) was relatively close to zero when compared to the first pole (K_1) throughout an increasing sinusoidal manoeuvre, shown in Figure 4.2. The second weighting even remained close to zero when the lateral acceleration of the vehicle past the vehicle's limit, which was known to be roughly $0.8\ g \approx 7.8\ m/s^2$ - shown in Figure 4.3 at around $190\ m$. This lower gain value suggested that the model's dependency on previous samples was much lower than initially thought.

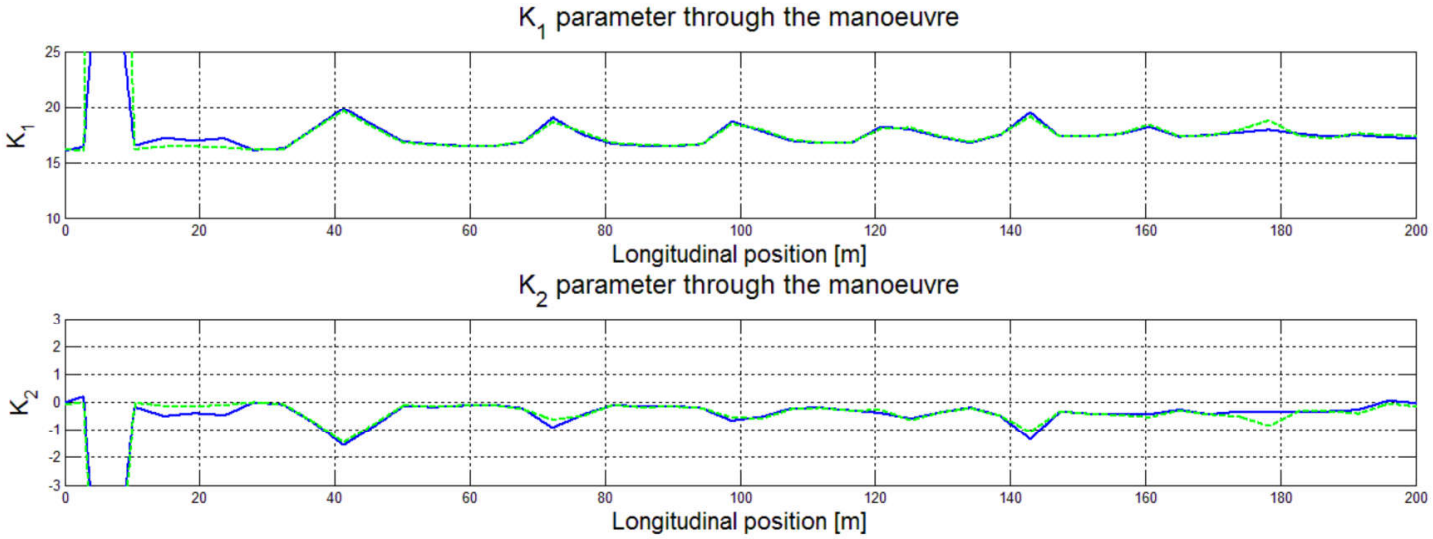

Figure 4.2: Calculated LQSTR gains in handling mode - blue; and ride comfort mode - green, during simulation of an increasing sinusoidal path at 80 km/h [Kapp, 2014].

This lead to the second pole being removed from the transfer function for the modified controller. The input dependency (zeros/numerator) of the modified controller was not altered and left at one sample. This simplification of the ARX-model greatly increased the speed at which the system could solve for the parameter values, covered in Chapter 4.1.3, and the LQSTR gains, covered in Chapter 4.2. The reduced model transfer function is given in Equation 4.3. The model delay n of the input

dependency was again left as one, as in the original controller. This delay in the exogenous input variable was intended to absorb any higher order dynamics of the system as well as any unknown time delays that were likely to be present in the steering system.

$$H(z) = \frac{\eta_0 z^{-n}}{1 + \varphi_0 z^{-1}} \tag{4.3}$$

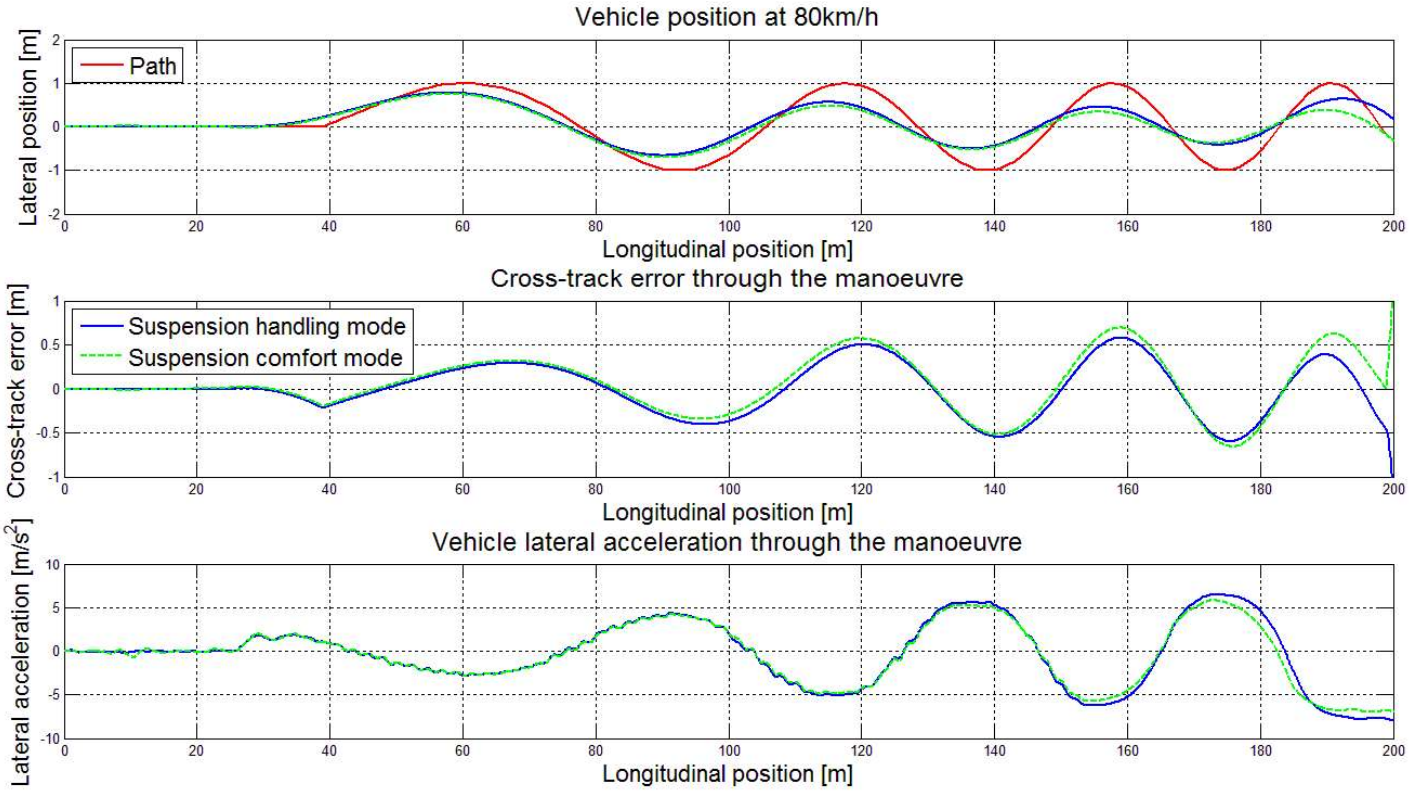

Figure 4.3: Lateral acceleration and cross-track error during simulation of a vehicle in handling mode - blue; and ride comfort mode - green, travelling through an increasing sinusoidal path at 80 km/h [Kapp, 2014].

Equation 4.3 can be re-written in the difference form, shown in Equation 4.4, where $\widetilde{y}(k)$ represents the estimated yaw rate of the vehicle at the current time-step, $y(k-1)$ represents the measured yaw rate of the vehicle at the previous time-step, $u(k-n) \rightarrow u(k-1)$ represents the steering angle of the vehicle at the previous time-step, while φ_0 and η_0 are the weighting parameters on the previous time-step's yaw rate and previous time-step's steer angle respectively.

$$\widetilde{y}\ (k) = \varphi_0 y\,(k-1) + \eta_0 u\,(k-n) \tag{4.4}$$

Equation 4.4 will be used in the form presented in Equation 4.5.

$$\widetilde{y}_{\ k} = \varphi_0 y_{k-1} + \eta_0 u_{k-1} \tag{4.5}$$

For ease of understanding Equation 4.5 is rewritten in Equation 4.6 using the standard symbols for yaw rate and steer angle presented in the list of symbols, where $\dot{\widetilde{\psi}}_k$ represents the estimated yaw rate at the current time-step as a function of the measured yaw rate and measured steer angle of the vehicle at the previous time-step multiplied by their respective weighting variables.

$$\dot{\widetilde{\psi}}_k = \varphi_0 \dot{\psi}_{k-1} + \eta_0 \delta_{k-1} \tag{4.6}$$

4.1.2 ARX-Model Sampling

The ARX-model sampling frequency of the modified controller was kept at $20\ Hz$ as the original ARX-model had already shown that it was capable of capturing the yaw natural frequency of the vehicle around which the simulation model was based [Botha, 2011]. These sample points were used to estimate the ARX-model parameters. The frequency at which the ARX-model itself was updated remained at $5\ Hz$.

A simulation study was conducted to determine what sample size (N) was needed for the ARX-model parameter estimations. The results of the study showed that for speeds of $30\ km/h$ to $100\ km/h$ a minimum of 3 samples was required to accurately model the yaw rate - steer angle relationship of the vehicle. These 3 samples spanned $0.15\ s$ at the sample rate of $20\ Hz$. Although 3 samples was deemed adequate for simulation purposes, where the simulation data was not polluted with noise, a larger sample size of 10 was chosen to allow for any future experimental implementation where measurements would be noisy. It was believed that a larger sample size, spanning a longer time period, would be able to help combat the effect of noisy measurements. This resulted in a data set that spanned $0.5\ s$ and was half that of the original ARX-model's sample size. It was assumed that the vehicle would not undergo any significant changes during this $0.5\ s$ time-frame, which would cause instabilities and likely lead to a loss of stable vehicle control. This study also showed that in simulations, sample sizes from 3 to 20 samples performed roughly the same in terms of handling

the vehicle. Anything smaller resulted in either a loss of control of the vehicle or inadequate path following. These simulations for analysing the effect of sample size were conducted assuming an on-board controller, with no packet losses and transport latencies of $0\ ms$.

4.1.3 ARX-Model Parameters

The principle of Linear Least Squares Estimation was used to solve for the ARX-model (weighting) parameters φ_0 and η_0, the method presented by [Van de Geer et al., 2008] is summarised here. Linear Least Squares Estimation was used to estimate the required parameters for the yaw rate estimation equation (Equation 4.5) by minimising the squared differences between a set of observed data points and a set of expected values. By re-writing Equation 4.5 in vector form for the 10 samples, Equation 4.7 is returned.

$$Y = \Phi\theta \tag{4.7}$$

Y is a vector of measured output values with k representing the time-step and N the total number of samples (Note that these are measured outputs, not estimated outputs):

$$Y = \begin{bmatrix} y_k \\ \vdots \\ y_{k-N} \end{bmatrix} \tag{4.8}$$

Φ is the regression vector:

$$\Phi = \begin{bmatrix} y_{k-1} & u_{k-1} \\ \vdots & \vdots \\ y_{k-N-1} & u_{k-N-1} \end{bmatrix}, \tag{4.9}$$

and θ the parameter vector, comprised of the ARX-model parameters, also called the least squares estimator:

$$\theta = \begin{bmatrix} \varphi_0 \\ \eta_0 \end{bmatrix} \tag{4.10}$$

The least squares estimator θ minimises the squared error shown in Equation 4.11, where y_i is the measured (actual) output, $\widetilde{y}_i$ is the estimated output, and N is the total sample size of 10.

$$\sum_{i=1}^{N} (y_i - \widetilde{y}_i)^2 \tag{4.11}$$

The solution of the least squares estimator is given in Equation 4.12.

$$\theta = \left(\Phi'\Phi\right)^{-1} \Phi' Y \tag{4.12}$$

It is possible to extend the above by including weightings for each sample set, thereby making use of the weighted least squares estimation technique. This approach, and its subsequent effectiveness, was deemed beyond the scope of this study.

4.1.4 ARX-Model Conclusion

The original ARX-model was shown to accurately capture the yaw response of the vehicle through a path-tracking simulation. The changes made, resulting in the modified ARX-model, were not so excessive as to warrant another tracking simulation, and as such it was assumed that the yaw response of the vehicle would still be captured adequately by the modified model. The primary alterations to the original model were a reduction in model order from 2 to 1, and a reduction in sample size from 20 to 10 samples. An important feature of this ARX-model is its platform independence. As the vehicle parameters evolve the ARX-model captures these changes allowing the model to be used between different platforms with very few modifications to the overall controller.

An important note: Because Equation 4.12 has a matrix inversion operation, it is necessary that the matrix that is being inverted not be close to singular. This presents a problem: When the vehicle's yaw rate is stabilising the $\Phi'\Phi$ matrix of Equation 4.12 is a poorly scaled matrix that is close to singular which causes inversion problems resulting in incorrect θ parameter calculations that lead to incorrect LQSTR gain values, see Chapter 4.2.2 for more on the LQSTR gain calculations. The same would likely occur during experimental implementations, where the measured signals are polluted by noise. To combat this problem it was decided that the ARX-model parameters would only be solved for if the Φ matrix had a Euclidean norm greater than 0.2, otherwise the previous model values - and subsequent gain values - would be used. This limiting factor corresponded to steer angle and yaw rate measurements in $radians$ and $radians/s$ respectively. It was noted that this limiting value for the Euclidean norm of the Φ matrix was only suitable if the yaw rate was stabilising towards zero. If the yaw rate was stabilising towards any other value the $\Phi'\Phi$ matrix still tended

towards a singular/poorly scaled matrix but the limiting value of 0.2 for the Euclidean norm was no longer adequate, suggesting that either a variable Euclidean norm be used or a different approach be followed to address the issue of poorly scaled matrices. For the purpose of this study the yaw rate stabilised to a value of zero, and as a result a constant limiting Euclidean norm of 0.2 was used.

4.2 LQSTR

In [Kapp, 2014] three control methods were compared, each making use of the ARX-model to represent the "plant" in a control scheme, the "plant" being the vehicle that would convert some steering angle into a yaw rate. Of these three controllers (Model Predictive Controller, Indirect self-tuning regulator, and an LQSTR), the aptly named LQSTR was found to be the most promising as it exhibited stable control at higher speeds as well as accurate control at lower speeds, overcoming the shortfall that many controllers experience. Modifications were made to the controller that allowed for even higher stable path following speeds with improved path following accuracy at all speeds considered. These modifications to the LQSTR controller follow, first with an overview of what the LQSTR controller is. Next the state space implementation is looked at, followed by a section on the errors used to obtain a reference signal for the controller. Finally, a summary of all the parameters used by the LQSTR controller is given.

4.2.1 LQSTR Overview

Linear Quadratic Regulator (LQR) control is a design methodology in which controller gains are calculated by optimally solving (minimising) some chosen cost function [Triantafyllou and Hover, 2003]. Ideally, LQR control requires full-state feedback for optimal control, however it is often not possible to obtain all the state measurements needed for full state feedback. LQSTR is an extension to LQR, where the gain matrix changes with changes in the adaptive plant model. For the LQSTR controller, the open loop plant is estimated using the ARX-model, where the ARX-model estimates the state of the vehicle (its yaw rate) at the next time-step using a combination of the previous time steps' yaw rate as well as the previous time steps' steering angle.

The general LQR block diagram with full-state feedback is shown in Figure 4.4, where r represents the reference signal that is to be followed, the required yaw rate. u is the input to the vehicle system, a required steer angle. A, B, and C are the state-space matrices (D is omitted from the diagram

as there is no feed-forward signal). K_{LQR} is the calculated LQR gain matrix with k the time-step. $\bar{N}$ is the pre-compensation gain matrix. y is the output of the system, the actual yaw rate. The red square encloses the open-loop plant that is estimated by the ARX-model.

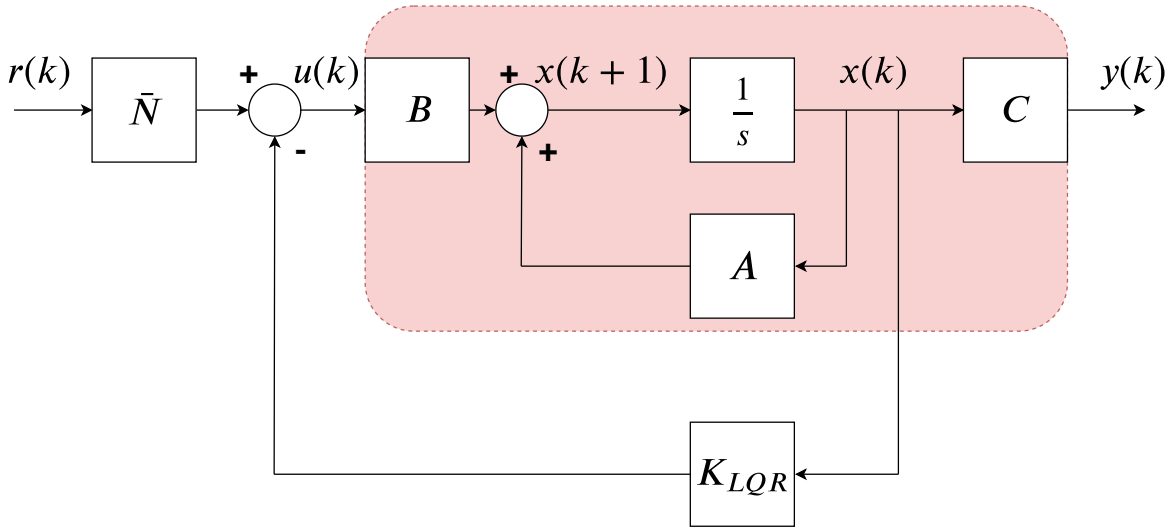

Figure 4.4: LQR block diagram for a discrete-time system.

4.2.2 State Space Realisation and Implementation

Two gain matrices needed to be solved for, the optimal gain matrix K_{LQR}, and the pre-compensation gain matrix $\bar{N}$. To solve for the optimal gain matrix K_{LQR} in Figure 4.4, the state space method was used to represent the plant/vehicle. The state space representation of the ARX-model (Equation 4.5) is given in Equation 4.13 in its standard form. To prevent confusion with variables and their meanings the standard u, as the input to the controller of Figure 4.4, is replaced with δ, the exogenous steer angle input of the ARX-model.

$$\begin{aligned} X_k &= AX_{k-1} + B\delta_{k-1} \\ Y_k &= CX_k + D\delta_{k-1} \end{aligned} \tag{4.13}$$

Where

$$X_k = [\widetilde{y}_k] \tag{4.14}$$

$$A = [\varphi_0] \tag{4.15}$$

$$B = [\eta_0] \tag{4.16}$$

$$C = [1] \tag{4.17}$$

$$D = [0] \tag{4.18}$$

When solving for the LQR gain matrix, the quadratic performance function with infinite horizon of Equation 4.19 was used, this allowed for weighting on both the set-point (Q_{LQR}) and control input (R_{LQR}).

$$J_{LQR} = \sum_{n=0}^{\infty} \left[\left(\dot{\psi}_{sp} - \dot{\psi} \right)^T Q_{LQR} \left(\dot{\psi}_{sp} - \dot{\psi} \right) + \delta^T R_{LQR} \delta \right] \tag{4.19}$$

The optimal feedback control law that minimises J_{LQR} is given in Equation 4.20, with $\delta_{required}$ the input to the plant/vehicle ($u(k)$ of Figure 4.4).

$$\delta_{required} = \dot{\psi}_{required} \bar{N} - K_{LQR} X_{k-1} \tag{4.20}$$

Where K_{LQR} was calculated with:

$$K_{LQR} = R_{LQR}^{-1} B^T P \tag{4.21}$$

P was solved for using the dynamic algebraic Ricatti Equation:

$$P = Q_{LQR} + A^T \left(P - PB \left(R_{LQR} + B^T PB \right)^{-1} B^T P \right) A \tag{4.22}$$

For the discrete case, the above was converted to the iterative solution in Equation 4.23. Since all matrices involved had been reduced to single elements a computationally simpler version was used, where the lower-case singular values correspond to the upper-case matrices presented thus far. The iteration limit was left at 1000 iterations, as in [Kapp, 2014].

$$p_{k+1} = q + a^2 p_k - \frac{(apb)^2}{r + b^2 p_k} \tag{4.23}$$

As shown, since there were no longer any matrices, there were no more matrix inversions involved in solving for the dynamic Ricatti Equation, greatly reducing the complexity as well as the required iteration count.

In Figure 4.4, it can be seen that the output $y(k)$ is not compared to the reference input $r(k)$ at any point. For this reason, a pre-compensation gain ($\bar{N}$) was added. For a continuous-time system, the pre-compensation gain is calculated as presented in [Michigin et al., ndb], however for a discrete-time system the controller designer usually uses their better judgement to choose a constant pre-compensation gain value [Michigin et al., nda]. During the simulation studies it was found that a pre-compensation gain making use of the discrete-time formulation was more suitable, for both on-board and off-site control. The value of $\bar{N}$ was calculated based on the speed of the vehicle, and thus remained constant throughout the simulations - which were conducted at constant specific speeds. The equation used is given in Table 4.1.

4.2.3 LQSTR Preview Times and Error Calculations

The reference signal, or required yaw rate, was a combination of two different yaw angles that, once summed together, were converted to a required yaw rate. These two yaw angles came in the form of two errors: the heading error and the lateral error, as depicted in Figure 4.5, where the complete block diagram of the LQSTR controller used in this study is shown. The dashed lines and blocks indicate measurements that were needed to update the ARX-model, which in turn was used to calculate the gain K_{LQR}. The $\dot{\psi}_{required}$ value corresponds to the reference input r of Figure 4.4, while $\dot{\psi}_k$ corresponds to y.

The heading error was used to align the vehicle with the path so that the path and the vehicle were parallel, while the lateral error, also known as the cross-track error, was used to remove any offsets that may have arisen due to drift while the vehicle was being controlled. The calculation of the heading error, presented in Equation 4.24, is trivial: the heading, or yaw angle, of the vehicle was compared with the desired heading at a preview time of τ_{yaw} ahead of the vehicle.

$$\psi_{heading\_error} = \psi_{required} - \psi_{actual} \tag{4.24}$$

The calculation of the lateral error is non-trivial. All the steps with corresponding transformation images are shown in Appendix B, with the final result for the lateral error given in Equation 4.25, where $y_{B'}$ and $x_{B'}$ are the transformed coordinates of the preview point and are given by Equation B.2.

$$e_{lat} = y_{B'} \cos(-\psi_{actual}) + x_{B'} sin(-\psi_{actual}) \tag{4.25}$$

The result of Equation 4.25 is a straight line distance between the vehicle's location and its desired location at some preview time τ_{lat} ahead of the vehicle. This was then converted to an angle using the `atan2d()` function in Matlab.

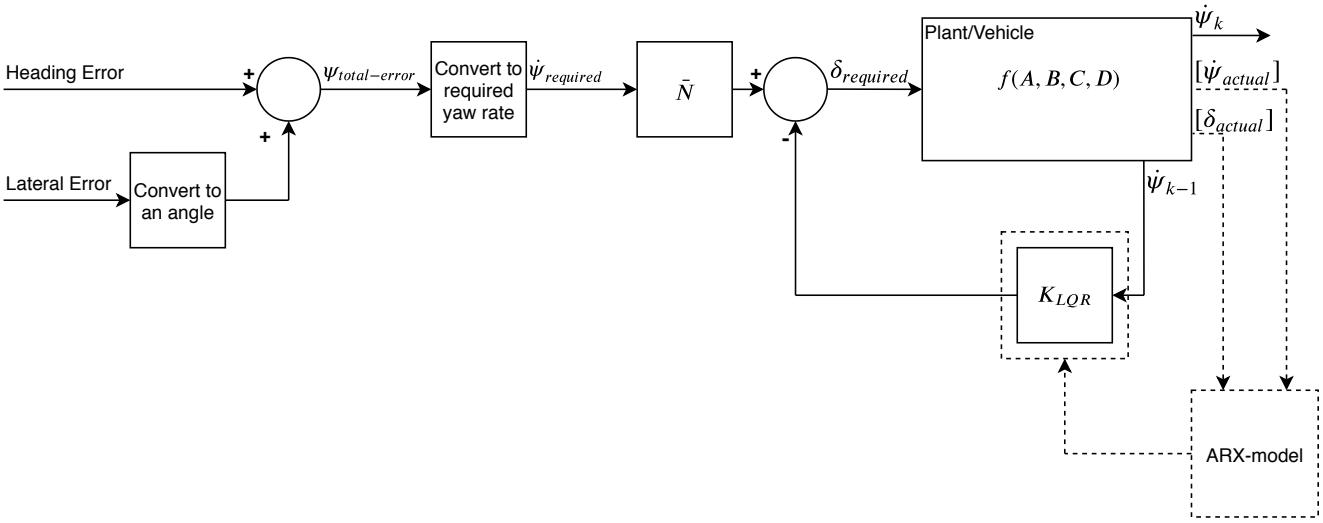

Figure 4.5: LQSTR block diagram as presented in this study.

After the individual errors were calculated, and converted to an angle if required, they were combined into a single yaw error (ψ_{total_error}). Then, using the general equation for rotation of a rigid body in plane motion (Equation 4.26), the total yaw error was converted into a required yaw rate ($\dot{\psi}_{required}$).

$$\Delta\psi = \frac{d\psi}{dt}\Delta t + \frac{1}{2}\frac{d^2\psi}{dt^2}\Delta t^2 \tag{4.26}$$

When the yaw acceleration term is assumed to be zero, the equation reduces to Equation 4.27 when solving for the desired yaw rate, where τ_{acc} is the preview time used for converting the yaw angle into a yaw rate.

$$\frac{d\psi}{dt} = \dot{\psi}_{required} = \Delta\psi\Delta t = \psi_{total\_error}\tau_{acc} \tag{4.27}$$

4.2.4 LQSTR Parameter Values

The completed LQSTR controller required 11 parameters, including those used in the ARX-model solutions, the values of which are summarised in Table 4.1, along with the frequency of the simulation software $f_{Simulink}$. $V_{vehicle}$ represents the speed of the vehicle in km/h. The values presented are for the simulations done under the assumption of an on-board controller ($0\ ms$ latency and 0% packet drop).

Table 4.1: LQSTR on-board parameter values.

Parameter	Value	Parameter	Value
Q_{LQR}	1	$N_{samples}$	10
R_{LQR}	3	$\Phi_{norm-limit}$	0.2
τ_{lat}	$\frac{V_{vehicle}}{180} + 0.2333$	f_{sample}	$20\ Hz$
τ_{yaw}	0.6	f_{model}	$5\ Hz$
τ_{acc}	1	$f_{vehicle}$	$100\ Hz$
$\bar{N}$	$-\frac{0.3}{90}V_{vehicle} + 1.3$	$f_{Simulink}$	$1000\ Hz$

τ_{lat} is the preview time used for the lateral error (e_{lat}) calculation, τ_{yaw} is the preview time used for the heading error (ψ_{error}) calculation, and τ_{acc} is the preview time used when converting the combined heading and lateral errors into a required yaw rate ($\dot{\psi}_{required}$). Although the lateral error preview time was based on the vehicle's speed, the controller was still able to function using a constant τ_{lat}, but basing the preview time on the speed of the vehicle resulted in a controller with better performance metrics.

The simulation frequency $f_{Simulink}$ was set to run at $1000\ Hz$ in Simulink. The reason for this was to allow signal transport latencies of as little as $1\ ms$ to be used ($1/1000 = 0.001\ s = 1\ ms$). The simulation environment required all subsystems to have a sampling frequency that was a multiple of the simulation frequency, thus a $1000\ Hz$ simulation frequency allowed for the $100\ Hz$ vehicle

parameters' sampling frequency $f_{vehicle}$, $20\ Hz$ ARX-model parameters' sampling frequency f_{sample}, $5\ Hz$ ARX-model update frequency f_{model}, and, if necessary, a $1\ ms$ change in transport latencies.

4.3 Validated Simulation Model

A fully validated simulation model based off of a Land Rover Defender 110 was made available to conduct the simulation studies. The Land Rover made use of a modified suspension system, which is discussed in more detail below. The simulation model included an Adams (Multibody Dynamics Simulation Solution software by MSC) model that was linked with Simulink (a component of Matlab).

4.3.1 $4S_4$ Modification

The Land Rover made use of a Four-State Semi-active Suspension System ($4S_4$), which is a hydro-pneumatic suspension system that had been designed and developed by [Els, 2007]. Each suspension strut of the vehicle could have its damper and spring stiffness changed for a ride more suited to comfort or a ride more suited to handling. These two settings are referred to as ride comfort mode and handling mode respectively. As the modes' names suggest, this resulted in ride comfort mode being more comfortable to the occupants while handling mode was less comfortable but resulted in a ride with less chance of roll-over as handling ability was increased. This was achieved in the following manner: in ride comfort mode, the $4S_4$ was set to low suspension stiffness and damping, while in handling mode the system was set to high suspension stiffness and damping.

During testing of the proposed controller in simulation studies, both ride comfort mode and handling mode settings were used at varying vehicle speeds. The reason for this was that the difference in vehicle properties varied greatly between ride comfort mode and handling mode. As in [Kapp, 2014], being able to control the vehicle in both modes without changes to the controller parameters showed a certain level of robustness on the controller's part. For the wireless simulations, the vehicle's $4S_4$ was left purely in handling mode.

4.3.2 Adams and Simulink

A 16 degree-of-freedom non-linear model was developed by [Uys et al., 2006], [Thoresson et al., 2007], and [Cronje et al., 2008]. The MSC.Software ADAMS (2017) environment was used to simulate the

multi-body dynamics of the Land Rover using experimentally obtained vehicle parameter values. Included in the model were the non-linear $4S_4$ characteristics, done so using adiabatic process theory to model the pneumatic suspension and look-up tables for the dampers and bump stops. Figure 4.6 shows the graphical representation of the MSC ADAMS model. The magenta spheres represent the occupants of the vehicle. The Pacejka '89 tyre model of [Bakker et al., 1989] was used to model the tyre road interface using model parameters determined by [Thoresson et al., 2007]. The properties of the Land Rover, on which the simulation model was based, is given in Appendix A.

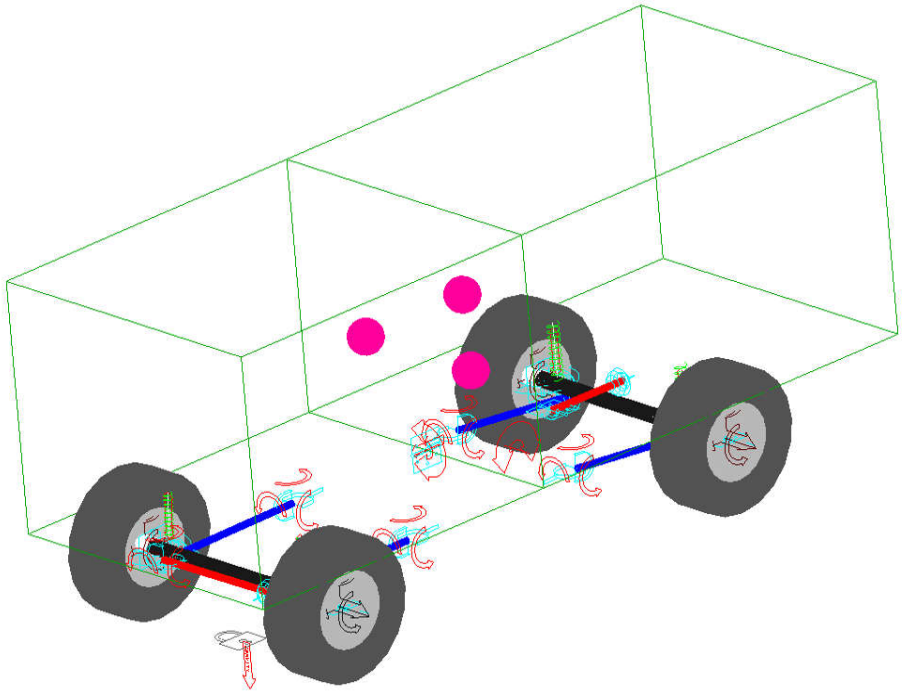

Figure 4.6: Full vehicle model, as simulated in Adams.

The longitudinal, lateral, and self-aligning moment characteristics of the tyre model were included. For the suspension kinematics of the front and rear portions of the vehicle model, solid axles and joints were modelled, similar to that of the physical test platform. The Adams model was then linked with Simulink (Matlab, 2017b), which allowed for efficient solving of the non-linear $4S_4$ by making use of options in the Matlab environment.

A full vehicle model validation study was carried out by [Botha, 2011], where the lateral motions of the simulation model were compared with the actual lateral motions of the platform when driving through a series of DLC-manoeuvres (ISO 3888-1, as in Chapter 4.4.1). In conclusion it was

determined that there was a good correlation between the platform and the model, resulting in a simulation model that could be used to reliably test driver models prior to testing them on the vehicle platform.

4.4 Test Paths and Measures of Accuracy

The specifications of a path used to test the controller's ability to handle a vehicle are presented as the DLC-manoeuvre below. The equations used to generate a path for simulation purposes are also presented. This is followed by two performance measures that were used to judge the path following accuracy of the vehicle through the DLC-manoeuvre: The maximum cross-track/lateral error and the Root Mean Squared Error (RMSE).

4.4.1 The DLC-Manoeuvre

To validate the controller presented in this study, a DLC-manoeuvre was conducted in simulations, the properties of which are given in Figure 4.7 with the relevant widths and lengths given in Table 4.2. The ISO 3888-1 Double-Lane-Change test was designed to evaluate the handling performance of a vehicle.

Table 4.2: DLC course width and length values [Botha, 2011].

Section	**Length** $[m]$	**Width**	**Actual Value** $[m]$
1	15	$1.1 \times$ vehicle width $+ 0.25m$	2.23
2	30	NA	NA
3	25	$1.2 \times$ vehicle width $+ 0.25m$	2.41
4	25	NA	NA
5	15	$1.3 \times$ vehicle width $+ 0.25m$	2.59
6	15	$1.3 \times$ vehicle width $+ 0.25m$	2.59
Lane Offset	3.5	NA	NA

It has been criticised due to it not only testing the physical vehicle, but also the ability of the driver [VECHI.CO, nd]. A more skilled driver is able to complete the DLC-manoeuvre at higher speeds, which makes it difficult to compare different results when differently skilled drivers are used

to complete the manoeuvre. However, this also makes the DLC-manoeuvre excellent for testing driver models, as a better driver model is likely to show improved performance through the manoeuvre.

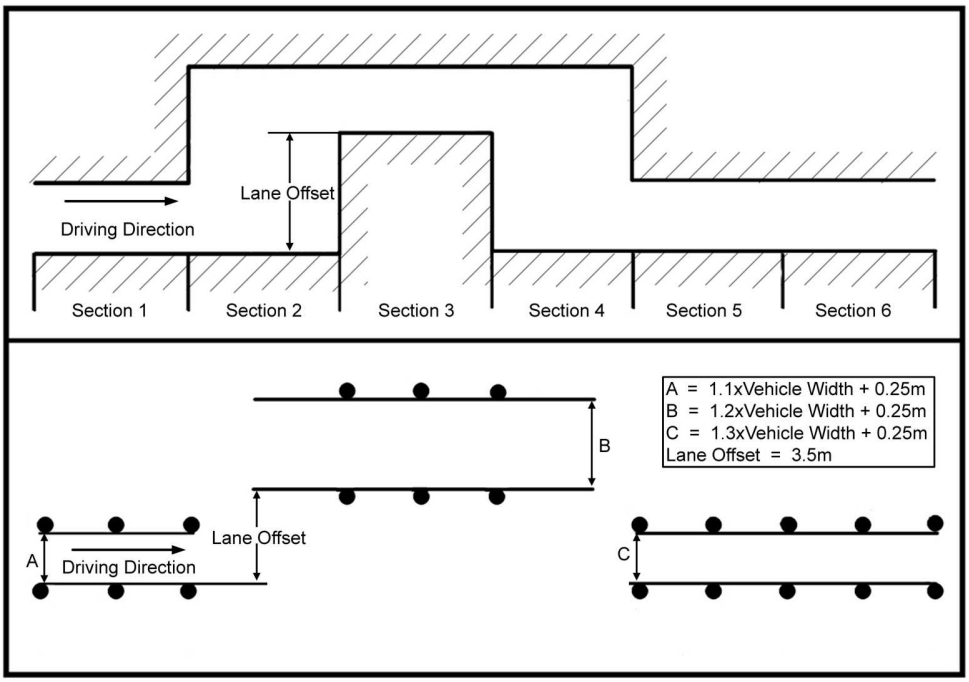

Figure 4.7: DLC course layout, ISO 3888-1 [Botha, 2011].

The equations presented in [Genta, 1997] were used to generate a path for the vehicle to take that simulated the path a vehicle would theoretically aim to take while attempting to traverse the manoeuvre. They are presented below in Equation 4.28 with the results plotted in Figure 4.8, where Y is the lateral position and X is the longitudinal position.

$$Y = \begin{cases} 0 & \text{for } X < 15 \\ \frac{3.5}{2}\{1 - cos(\frac{\pi}{30}[X - 15])\} & \text{for } 15 \leq X < 45 \\ 3.5 & \text{for } 45 \leq X < 70 \\ \frac{3.5}{2}\{1 + cos(\frac{\pi}{30}[X - 70])\} & \text{for } 70 \leq X < 100 \\ 0 & \text{for } 100 \leq X \end{cases} \tag{4.28}$$

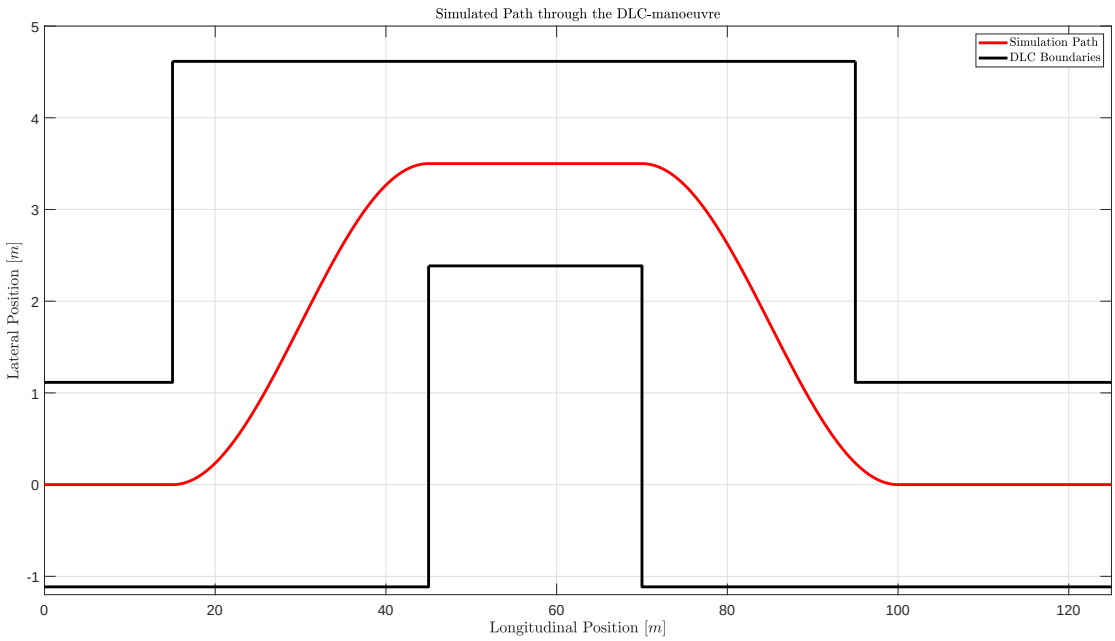

Figure 4.8: Simulated DLC path through DLC boundaries, as presented in Equation 4.28.

4.4.2 Measures of Path-Following Accuracy

Two methods were used to judge the accuracy of the path taken by the vehicle during simulations. The first was the maximum cross-track/lateral error. This was calculated using Equation 4.29, where cg_{y_i} is the y-coordinate of the centre of gravity of the vehicle, y_{path_i} is the y-coordinate of the required path directly above/below the centre of gravity of the vehicle calculated without preview times, and K is the total number of points on the path. A higher maximum cross-track error through the DLC-manoeuvre suggests a lower path following performance but this is only a localised error. It is possible that the vehicle could traverse the DLC-manoeuvre more accurately overall and still have a larger maximum cross-track error. For this reason, a second method of accuracy measure was used.

$$\begin{aligned} [\text{cross-track error}]_i &= y_{path_i} - cg_{y_i} \\ \implies \max([\text{cross-track error}]) &= \max(y_{path_i} - cg_{y_i}) \text{ for } i = 1, 2, 3...K \end{aligned} \tag{4.29}$$

The second method used was the RMSE presented in Equation 4.30, where cg_{x_i} and cg_{y_i} are the x- and y-coordinates of the centre of gravity of the vehicle, y_{path_i} and x_{path_i} are the x- and y-coordinates of the nearest path point to the vehicle calculated without preview times, and N is the total number of points in the path travelled by the vehicle. The RMSE can be seen as the total

deviation of the vehicle's path from the desired path in a single number: the higher the RMSE, the lower the path following accuracy of the vehicle, while the lower the RMSE, the higher the path following accuracy of the vehicle.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \left(\min \left(\sqrt{(y_{path_i} - cg_{y_i})^2 + (x_{path_i} - cg_{x_i})^2} \right) \right)^2}{N}} \tag{4.30}$$

4.5 LQSTR Simulation Study

A simulation study was conducted to determine how well the proposed modified controller faired against the original controller. It had already been shown that the original controller was able to stably control the simulated vehicle model from speeds of $30\ km/h$ up to $115\ km/h$, with the modifications made allowing for speeds in excess of $120\ km/h$.

4.5.1 DLC Results

A series of DLC-manoeuvre tests were conducted in the simulation environment of Matlab using the proposed modified controller. It was found that the modified controller was able to stably control the vehicle up to speeds of at least $150\ km/h$ while maintaining very low RMSEs. At speeds greater than $80\ km/h$ the vehicle started cutting corners, thereby not successfully navigating the DLC in terms of staying within the cones - which is physically impossible for the vehicle at such speeds. The results of the DLC-manoeuvres at $30\ km/h$, $60\ km/h$, $100\ km/h$, and $120\ km/h$ follow in Figures 4.9, 4.10, 4.11, and 4.12 respectively. These results were obtained with handling mode settings for the suspension system. It can be seen that as the speed of the vehicle increased, so did the maximum lateral acceleration of the vehicle. The cross-track error plots exhibit the same general pattern for all speeds simulated, with only the magnitude changing with the increase in speed. The maximum cross-track errors generally occurred as the vehicle was leaving the first lane change while the maximum lateral acceleration was generally experienced as the vehicle left the second lane change. This was however not the case for the $60\ km/h$ simulation, where the maximum cross-track error occurred as the vehicle entered the second lane change and the maximum lateral acceleration was experienced as the vehicle entered the first lane change.

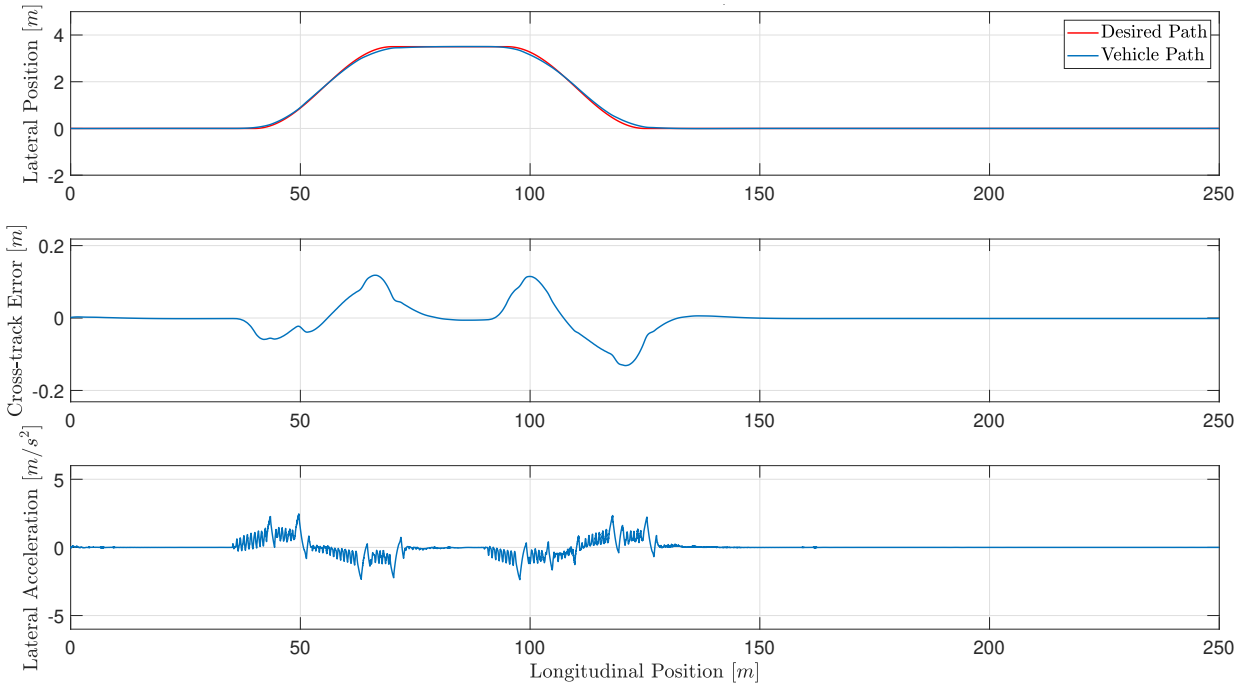

Figure 4.9: Simulated DLC results at 30 km/h in handling mode.

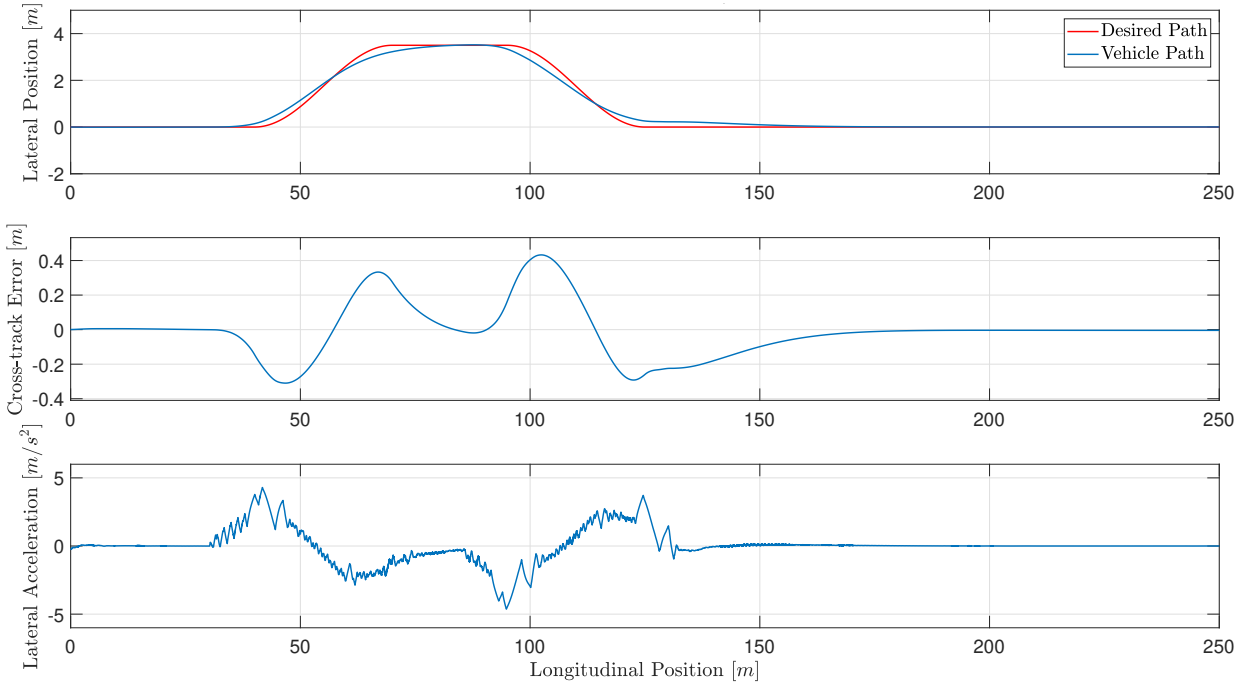

Figure 4.10: Simulated DLC results at 60 km/h in handling mode.

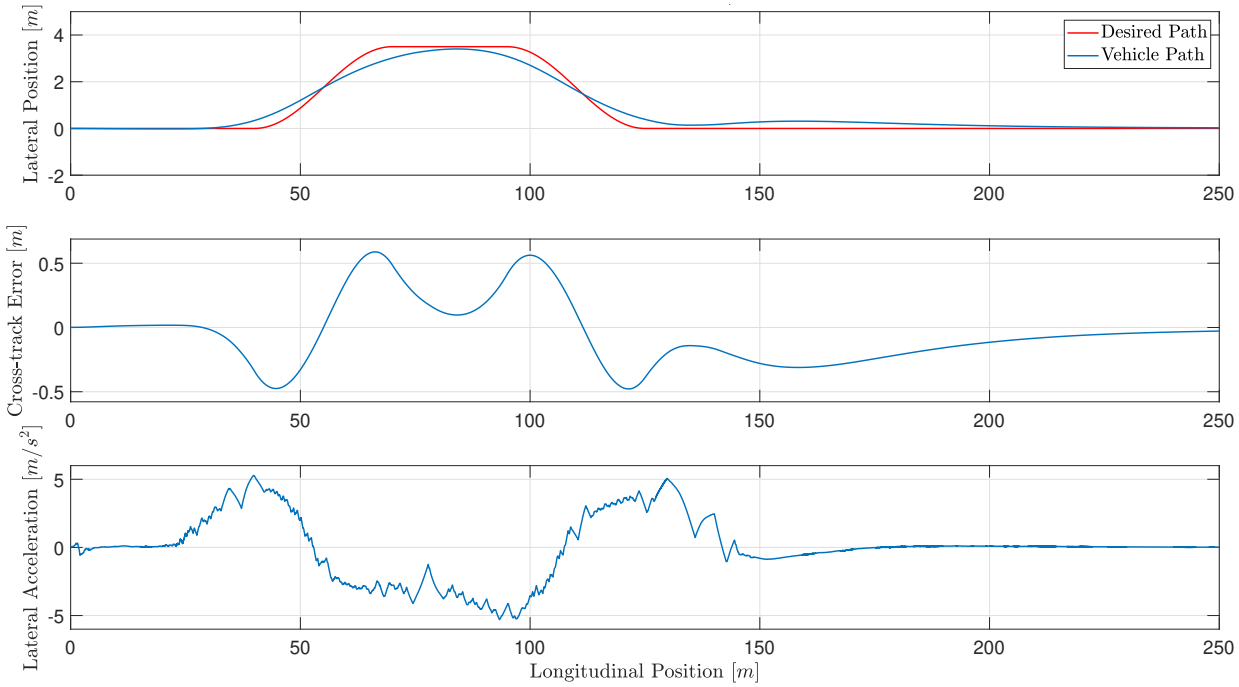

Figure 4.11: Simulated DLC results at 100 km/h in handling mode.

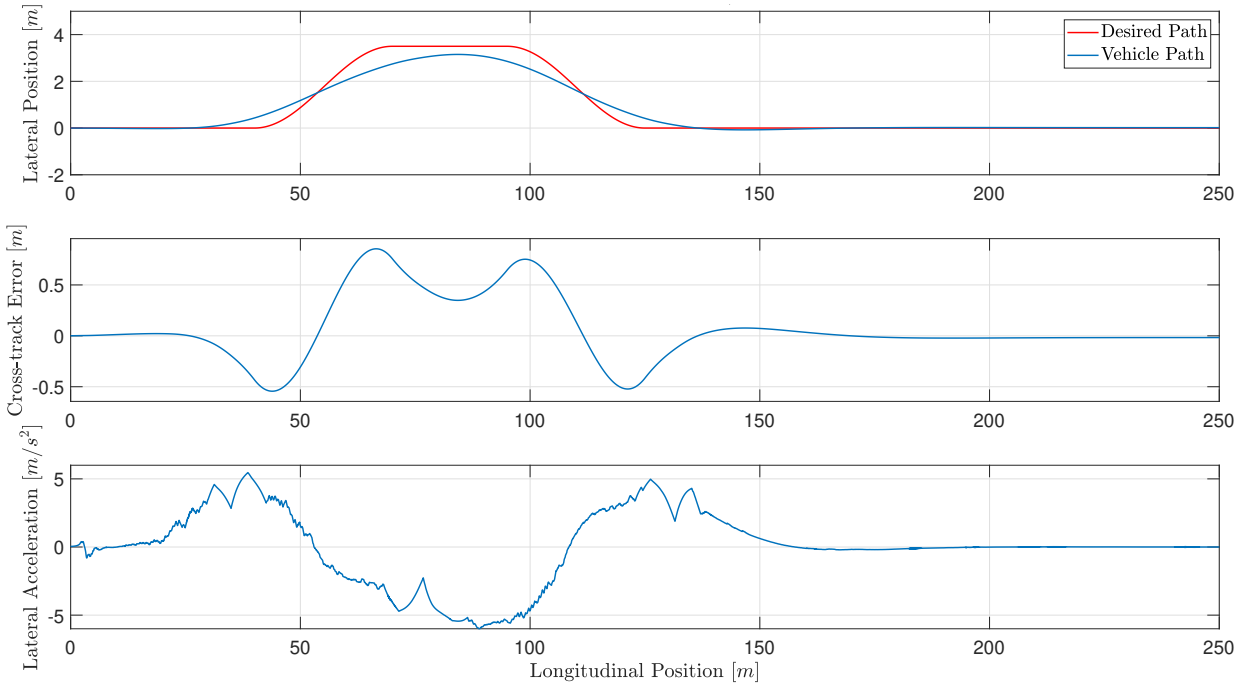

Figure 4.12: Simulated DLC results at 120 km/h in handling mode.

As the speed was increased from $30\ km/h$ to $100\ km/h$, the vehicle took longer to return to the desired path after the second lane change. As the vehicle left the second lane change it would oscillate once before crossing the desired path. This changed between the $100\ km/h$ and $120\ km/h$ simulation. At $120\ km/h$ the vehicle was simulated to cross over the desired path at roughly $140\ m$ as a slight overshoot. This change over from undershooting to overshooting the desired path after the second lane change occurred at a speed of about $110\ km/h$, shown in Figure 4.19, and resulted in a spike in the RMSE values shown in Figure 4.13. When looking at the path taken by the vehicle there was almost no overshoot as the vehicle left the first lane change. This was most likely due to τ_{lat} being a function of vehicle speed. The equation for τ_{lat} presented in Table 4.1 resulted in higher vehicle speeds corresponding to larger preview times, thereby sending larger required yaw rates to the controller. A constant preview time is usually used to adjust the preview distance, as was the case in the original controller. This constant preview time results in a variable preview distance that is a function of the vehicle's speed; by the relationship between time, speed, and distance. By basing the lateral preview time τ_{lat} in this study on the vehicle's speed, a larger variation in the preview distance was realised between lower and higher vehicle speeds. This contributed substantially to the improved performance of the controller. A comparison of the RMSEs for the handling mode and ride comfort mode settings of the active suspension is given in Figure 4.13.

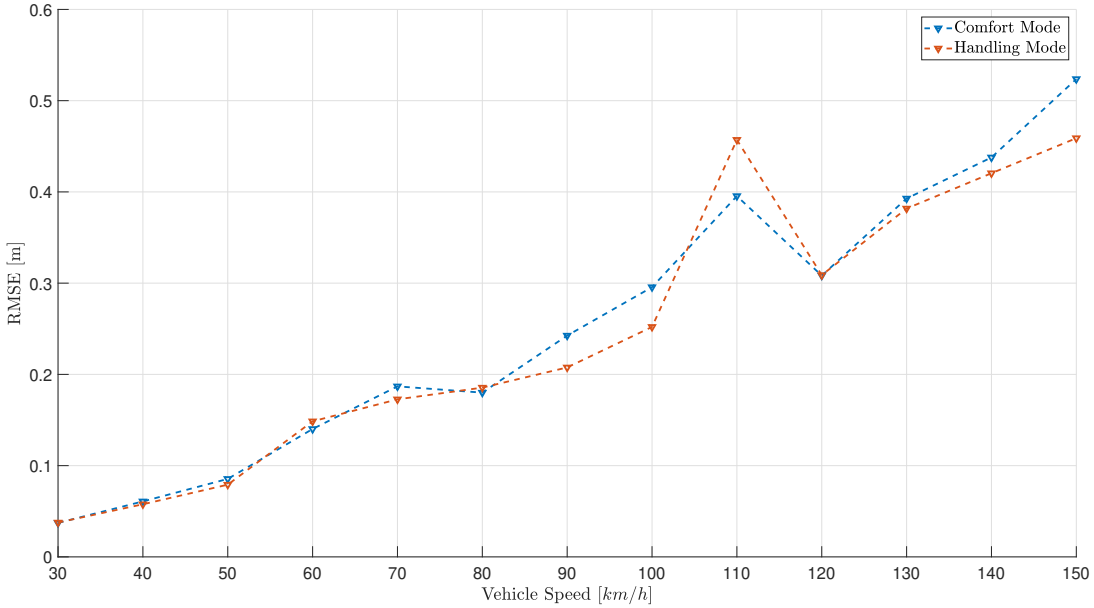

Figure 4.13: Comparison of RMSEs for handling mode and ride comfort mode suspension settings.

This comparison is shown to emphasise the level of robustness the controller exhibited. The controller performed about the same in handling mode and ride comfort mode at all simulated speeds. The differences in the RMSE values for both ride comfort and handling mode suspension settings were negligible, with a roughly linear relationship between RMSE and vehicle speed being present for both suspension settings. Even the bump in the RMSE values at $110\ km/h$ occurred for both suspension settings. This was the only instance where the ride comfort suspension settings were used, the rest of this study made use of the handling mode settings for simulations, with the exception of the comparison between the original and modified controller in Chapter 4.5.4.

4.5.2 LQSTR Gain Computation Analysis

When looking at the iteration counts for solving the dynamic Ricatti Equation in Figure 4.14, it can be seen that the iterations fell well below the limit of 1000 iterations for all speeds considered. The average iteration count was calculated without considering iteration counts of zero.

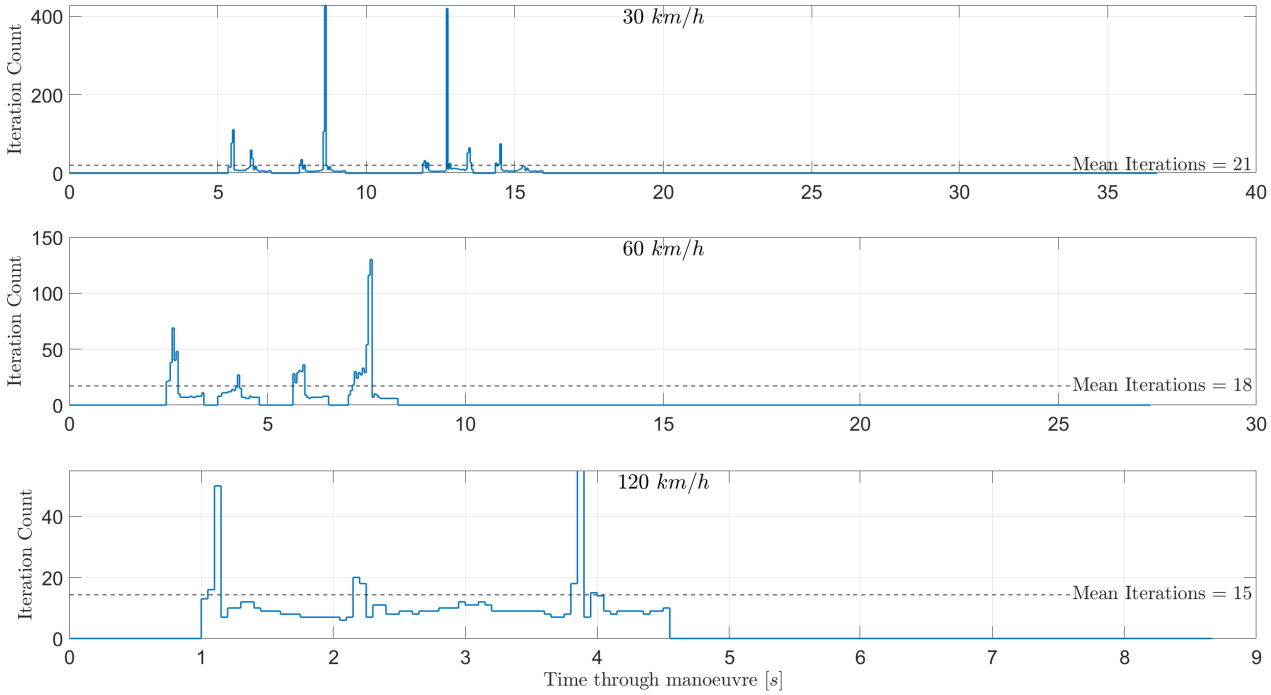

Figure 4.14: Comparison of iteration counts for vehicle speeds of 30 km/h, 60 km/h, and 120 km/h through the DLC-manoeuvre.

The maximum iteration count shown occurred at a speed of 30 km/h, 430 iterations, well below

the imposed limit. The maximum iteration count for $120\ km/h$ was 310. This showed that it was a bit difficult to predict exactly how many iterations there would be at any given ARX-model update but it could be confidently stated that even if measurements were to be polluted with noise, the iteration count should fall well below the limit of 1000 iterations, especially for a vehicle speed of $60\ km/h$.

Figure 4.15 shows the mean calculated LQSTR gain K_{LQR} through the manoeuvre at all the simulated speeds from $30\ km/h$ to $150\ km/h$, as well as the calculated pre-compensation gain $\bar{N}$ for comparative reasons. The means were calculated starting from the beginning of the simulation (corresponding to $0\ m$ travelled) up until just after leaving the second lane change (corresponding to $150\ m$ travelled). The mean gain values calculated were rough estimates of the mean value during the manoeuvre, values after the vehicle had stopped calculating new gains were not included in the calculations. From the figure there appears to be no special relationship between the optimal LQSTR gain value and the vehicle's speed, other than the calculated gain value falling just below a value of one for all simulated speeds. The plot of the pre-compensation gain shows that as the speed of the vehicle was increased, the effect of the reference signal was decreased as a result of the decrease in the pre-compensation gain.

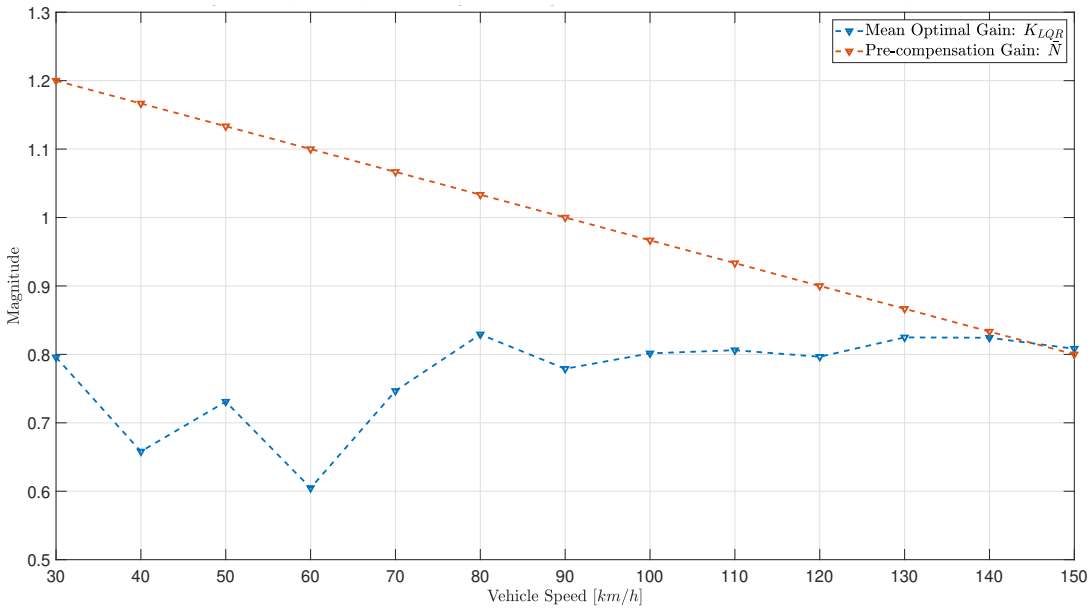

Figure 4.15: Comparison of the average calculated LQSTR gain (K_{LQR}) for all simulated vehicle speeds.

4.5.3 ARX-Model Parameters' Analysis

In Figure 4.16 the relationship between the current yaw rate and that of the previous yaw rate and previous steering angle is depicted. It shows that the model solved in such a way that the current yaw rate was more dependent on the previous steering angle (corresponding weighting variable: $\theta(2) = \eta_0$) than it was on the previous yaw rate (corresponding weighting variable: $\theta(1) = \varphi_0$). This was especially true at lower speeds, where at $60\ km/h$ the weighting on the previous yaw rate generally stayed below one, while the weighting on the previous steering angle was generally above one.

When looking at the $30\ km/h$ results of the figure, the afore mentioned relationship between the current yaw rate and the previous yaw rate and steer angle appears unclear. However, as the speed of the vehicle was increased the pattern appeared more pronounced - as shown by the results of the $120\ km/h$ plot. This was most likely due to the increased angular momentum the vehicle experienced at higher vehicle speeds while traversing the DLC-manoeuvre. The sections of the graphs where both weightings are zero correspond to when no new LQSTR gain values were calculated.

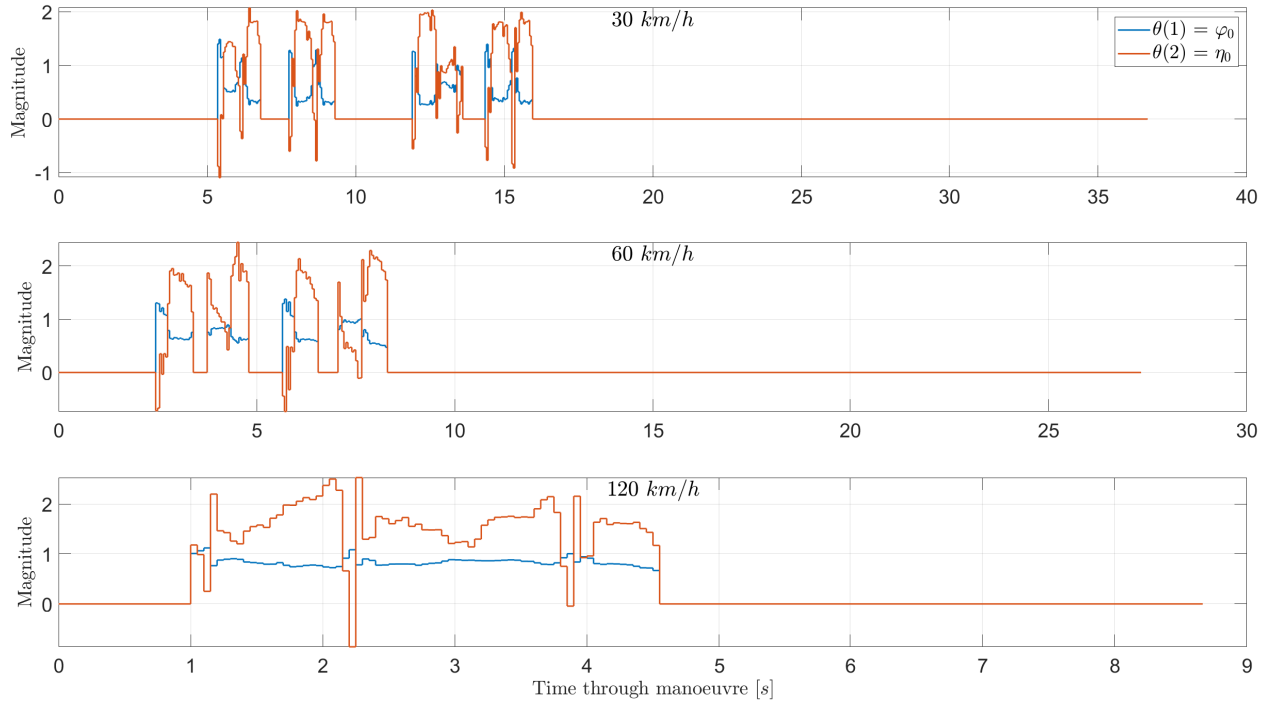

Figure 4.16: Comparison of the calculated theta values for vehicle speeds of 30 km/h, 60 km/h and 120 km/h through the DLC manoeuvre.

4.5.4 Comparison of Original and Modified Controllers

It has been shown that the modified controller was able to robustly control the vehicle up to speeds of $150\ km/h$ in simulations. A comparison of the RMSEs of the modified and original controller for speeds from $30\ km/h$ up to $150\ km/h$ is given in Figure 4.17, with Figures 4.18 and 4.19 comparing the results of the $110\ km/h$ DLC-manoeuvre.

The RMSE comparison of Figure 4.17 clearly shows that the modified controller was able to control the vehicle up to higher speeds and maintain a higher accuracy, in terms of RMSE, at all speeds except: $90\ km/h$ and $100\ km/h$ for the modified controller using ride comfort mode settings. The comparison of the DLC-manoeuvre completed at a speed of $110\ km/h$ showed that although the modified controller maintained a lower RMSE for the entire manoeuvre, the original controller was able to return a slightly smaller maximum cross-track error, at least for a vehicle speed of $110\ km/h$. The lower RMSE value obtained by the modified controller was attributed to the vehicle returning to its required path, after completing the DLC-manoeuvre, faster than the original controller. This was accompanied by lower lateral accelerations from the modified controller.

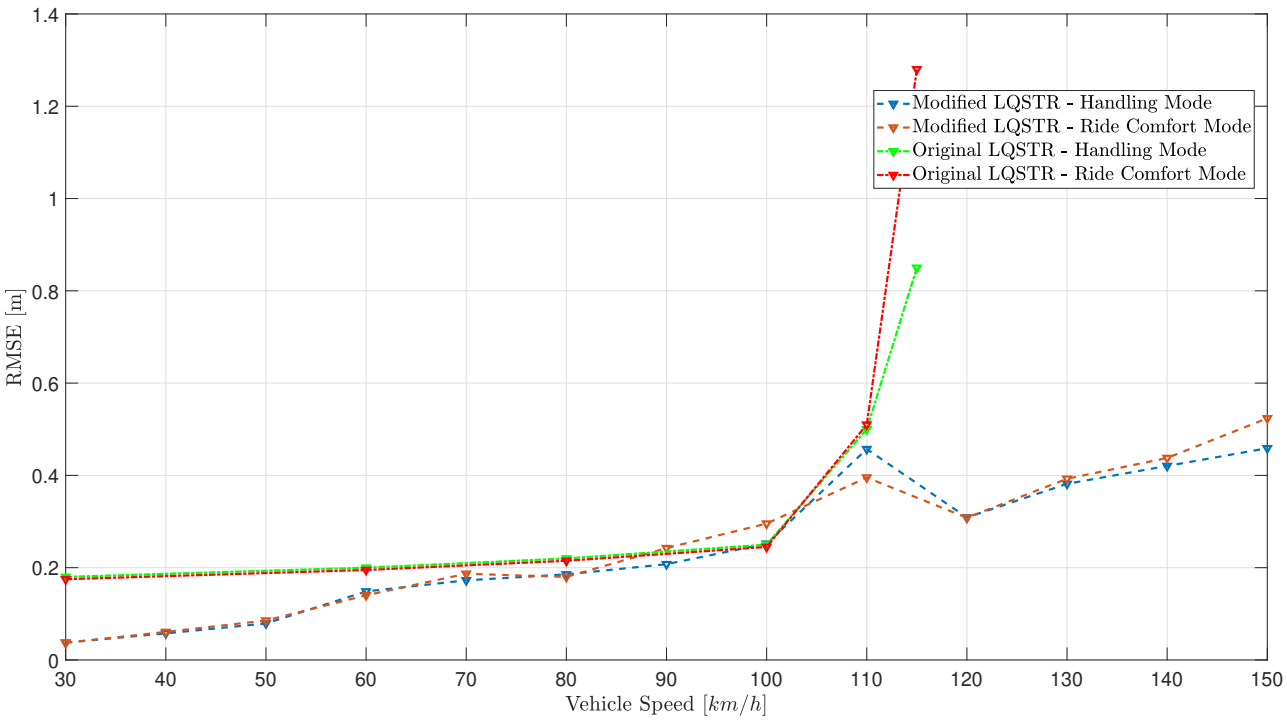

Figure 4.17: Comparison of the modified and original controllers' RMSEs through the DLC-manoeuvre.

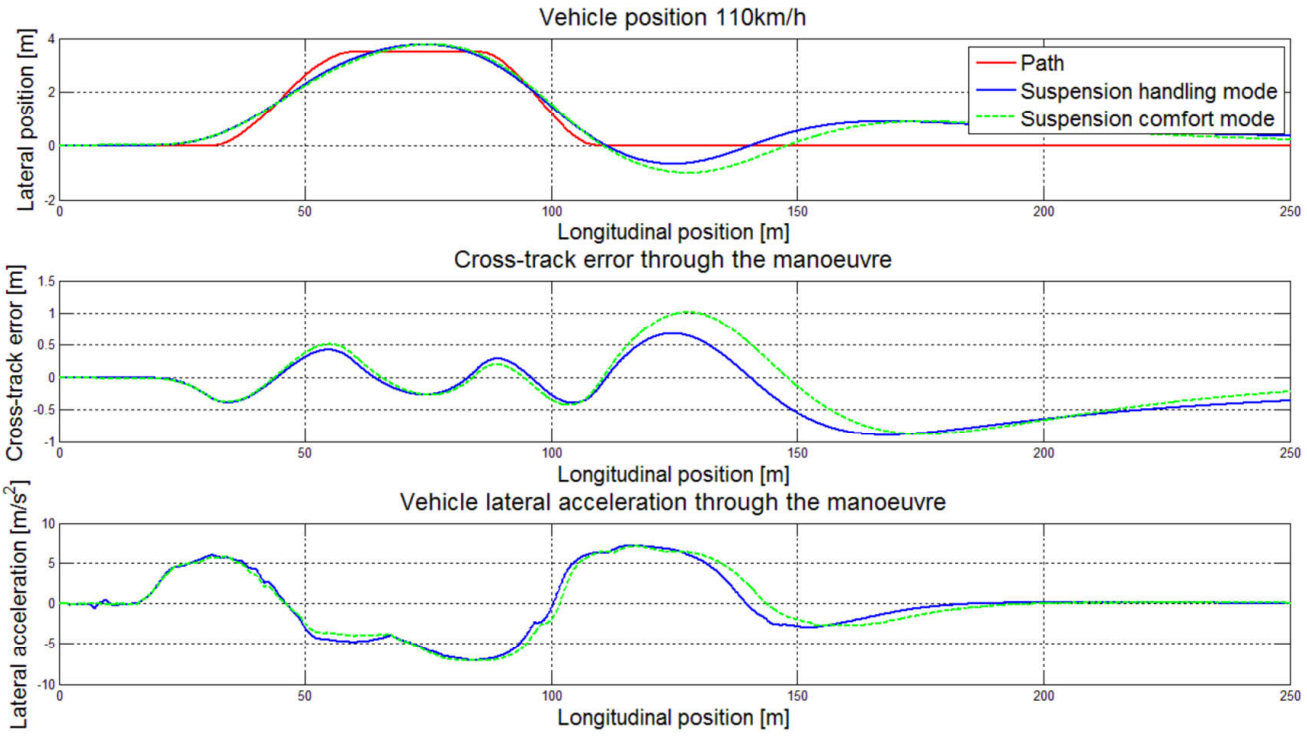

Figure 4.18: Original controller DLC-manoeuvre results at 110 km/h [Kapp, 2014].

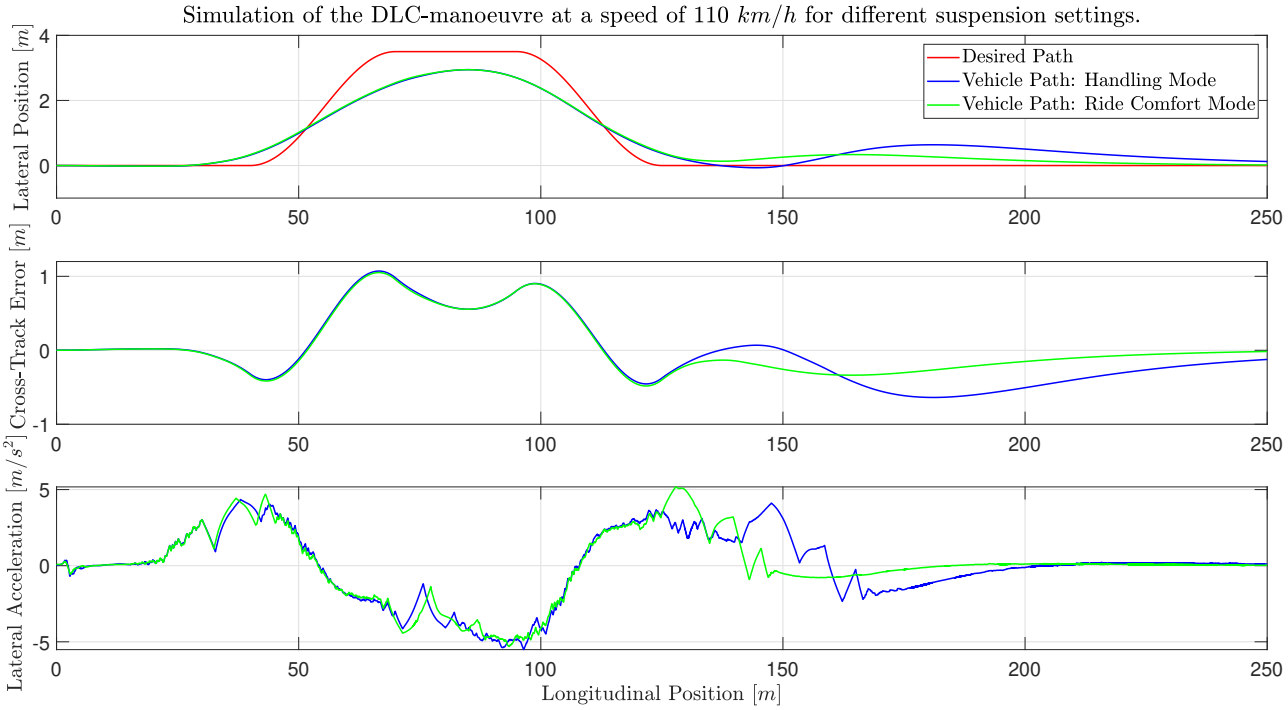

Figure 4.19: Modified controller DLC-manoeuvre results at 110 km/h.

4.6 Conclusion

A modification to the controller presented by [Kapp, 2014] has been presented. It was shown that the modified controller improved the all-round performance of the original controller, while at the same time reducing computational complexity, both while the vehicle was travelling at lower speeds ($30\ km/h$) and at higher speeds ($120\ km/h$ to $150\ km/h$). Not only did the modified controller perform better, it also required far less computational effort due to the smaller sample size and the removal of the matrix inversion in the dynamic Ricatti Equation. The reduction in computational requirements was seen as quite an important feature for the modified controller in terms of V2X. If an RSU is to take control of multiple vehicles at the same time it cannot spend unnecessary amounts of time solving for one vehicle's parameters, gains, and subsequent required control actions. This would lead to larger latencies between measuring vehicle states and applying a calculated control action, leading to an increased risk of control loss.

Chapter 5

Wireless Simulations

In this chapter the modifications made to the on-board controller to simulate off-site control are presented first. Next the results of the simulations carried out under various baseline latencies and packet drop percentages are looked at. Finally, conclusions are drawn from the results of the simulations. The latency model used in this chapter was discussed in Chapter 3.2.

Figure 5.1 shows in a broader sense where transport latencies would be present in a real world scenario, and subsequently where they were to be simulated during simulations. There would be a latency experienced between the vehicle and the base station and between the base station and the vehicle. When referring to the latency applied, a latency of $x\ ms$ refers to $x\ ms$ between a packet leaving the vehicle and a subsequent control action returning to the vehicle. For simulation purposes, latencies were added on a one-way basis - between the vehicle and the base station/controller, and again between the base station and the vehicle. This meant that if a latency of $100\ ms$ was applied, $50\ ms$ was applied on the path between the vehicle and the base station, while the other $50\ ms$ was applied on the path between the base station and the vehicle. It must be mentioned that the path on which the latency occurs does not matter, what matters is the amount of time between the vehicle sending a packet X and the vehicle receiving a calculated required steer angle based on the contents of packet X.

Based on the experimental results obtained by [Kapp, 2017], vehicle instabilities only started to present themselves once one-way latencies, between vehicle and controller and then between controller and vehicle, were in excess of $50\ ms$, i.e. when the system was subjected to a total latency of $100\ ms$.

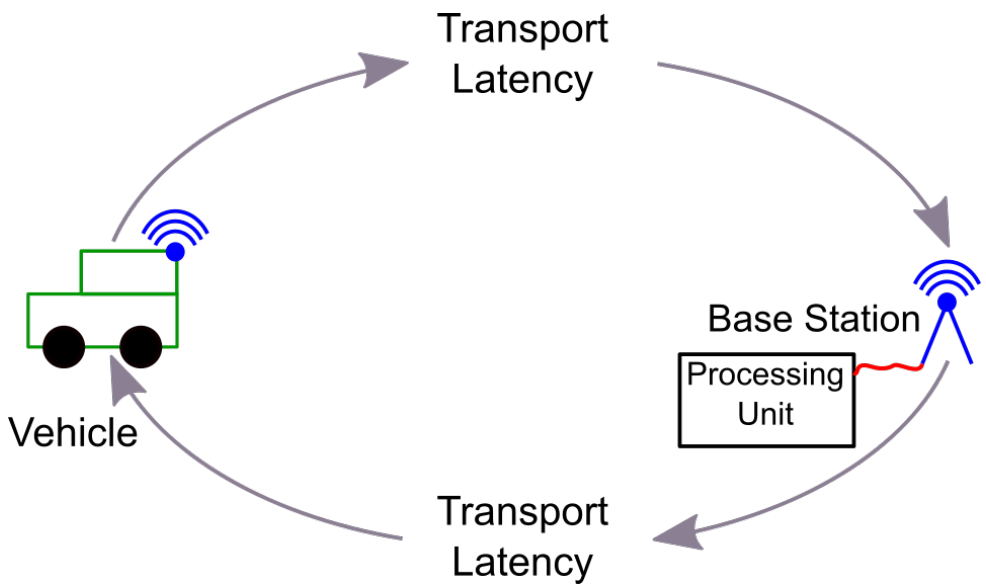

Figure 5.1: Depiction of the locations where latencies were to be present during simulations.

This was well beyond even the RTTs of the 700 m latency-model-generation tests, meaning that when higher latencies were desired a baseline latency needed to be added to the model latency. Since simulations did not have to physically send packets over a wireless channel - thereby adding the processing delays of the routers, the baseline latency had the extra modelled latency added to it immediately.

5.1 Controller Alterations for Latency Simulations

The only modifications made to the controller to account for latencies being present was the addition of a new parameter: the transmission rate, or the rate at which packets were sent from the vehicle to the base station, and the removal of the ARX-model update rate, discussed below. Table 5.1 again summarises the values used for the simulations, with the addition of the new parameter. The preview time τ_{lat} was calculated as $0.5667\ s$ for a vehicle speed of $60\ km/h$ using the equation in Table 4.1. The pre-compensation gain was calculated using the same method as that used during on-board simulations, the equation of which can be found in Table 4.1, resulting in a constant value of 1.1 for a vehicle speed of $60\ km/h$. Since the vehicle speed remained at a constant $60\ km/h$ during simulations, a constant pre-compensation value of 1.1 was used.

Table 5.1: Parameter values used when the controller was simulated to be off-site.

Parameter	Value
Q_{LQR}	1
R_{LQR}	3
τ_{lat}	0.5667 s
τ_{yaw}	0.6 s
τ_{acc}	1 s
$\bar{N}$	1.1
f_{send}	50 Hz

Parameter	Value
$N_{samples}$	10
$\Phi_{norm-limit}$	0.2
f_{sample}	20 Hz
f_{model}	N/A
$f_{vehicle}$	100 Hz
$f_{Simulink}$	1000 Hz

5.1.1 Frequency Alterations

A new frequency was added, the rate at which packets were sent from the vehicle, or f_{send}, and was chosen to be $50\ Hz$. A small simulation study was conducted to determine the effect of message transmission rates from the vehicle to the base station/controller. The transmission rate from a base station back to a vehicle would be fully-dependent on how long it took the processing unit to calculate a required steer angle and the transmission rate from the vehicle to the base station, as once the required steer angle was calculated it would be sent back to the vehicle immediately. From the study it was determined that a transmission rate of at least $10\ Hz$ was needed up to speeds of $100\ km/h$ while $20\ Hz$ was required for $120\ km/h$, control of the vehicle was lost if the transmission rate was below these values.

The chosen transmission rate of $50\ Hz$ was half that used by [Kapp, 2017], where $100\ Hz$ was used, but well above the $20\ Hz$ lower limit. A benefit of using this lower transmission rate is that there would be less traffic on the network channel. Since there was only one vehicle being controlled in the simulation of a wireless network the amount of traffic was not a major limiting factor, but for large-scale implementation, the less congestion on the channel there is, the better.

When the controller was simulated to be on the vehicle the ARX-model update frequency (f_{model}) ran at a set rate of $5\ Hz$. This update rate was changed when latencies were added so that the

ARX-model updated whenever a data packet was received by the base station. This meant that if a packet was simulated to have dropped, the model would not update as there were no new values to build a model from. It also meant that at most the model would update at $50\ Hz$, while the minimum would depend on the number of packets dropped as well as the distribution of dropped packets.

An important note on the order in which packets arrive when being transmitted: The transmission rate of $50\ Hz$ resulted in a packet being sent from the vehicle every $20\ ms$. This gap between packets being sent was larger than that of the latencies added to the network by the Mikrotik routers, which was at most $15\ ms$ for a round-trip, meaning that the chances of a packet being received out of order were very slim, as by the time the next packet was sent, the previous packet would have already arrived at its destination. Thus, out of order packets were not considered in simulations. In a real-world application out-of-order packets can be addressed by including a form of incremental packet number in the message, which is used to help maintain packet order.

5.2 Latency Model Alterations for Simulations

For implementation during the simulations, the mean values, standard deviations, and packet percentages around each normal distribution were altered slightly, the results of which are given in Table 5.2.

Table 5.2: Simulated Latency Distribution Values.

Distribution	**Mean** $[ms]$	**Standard Deviation** $[ms]$	**Size** [%]
1	3	0.3661	56
2	7	0.6715	34
3	11	0.7877	10

The mean values for the first and second distributions were rounded-up, while the third means' distribution was rounded-down. The percentages were changed to be easier to work with, with the first distribution containing 56% of the packets, the second distribution containing 34% of the packets, and the third distribution containing 10% of the packets.

5.3 Wireless Simulations' Results

A series of simulations were run to study the effect that transport latencies had when simulating control of a vehicle from an off-site controller, the effect of dropped packets was also included in the simulations where the total number of dropped packets was a percentage of the total number of packets sent. A summary of all of the simulations' results is given in Table C.1 in Appendix C. Figure 5.2 plots the results of the simulations where control of the vehicle was not lost, at packet drop percentages of 0%, 10%, 20%, 30%, and 40%. The baseline latency value represents the total milliseconds that the packet was simulated to be held back: a combination of the path between the vehicle and the controller, and between the controller and the vehicle. Added to this value was a latency value drawn from the latency model of Chapter 3.2, this was done on each of the paths. A certain percentage of the packets sent in either direction, from the vehicle to the controller or the controller back to the vehicle, were dropped independently, i.e. if slightly too many packets were dropped from the vehicle to the controller, it did not affect the amount of packets dropped from the controller to the vehicle.

Figure 5.2 highlights that up to a baseline latency of $100\ ms$, up to 40% of packets could be dropped before total control of the vehicle was lost, while higher baseline latencies had lower corresponding packet drop percentages before control was lost. Although higher baseline latencies were simulated without total loss of control, the resultant RMSE values were deemed inadequate to be considered reliable control. Note in Figure 5.2 that for the baseline latency of $160\ ms$, the simulation was only able to complete if there was a packet drop percentage of 0%.

The summary of the RMSE results presented in Table C.1 suggest a "safe" operating region of up to $100\ ms$ baseline latency where up to 40% of packets can be dropped. Larger baseline latencies, up to $200\ ms$, resulted lower maximum allowable packet drop percentages where safe control was only realised if no packets were dropped.

Following are two analyses, one looking at the effect of maintaining a constant packet drop percentage while increasing the baseline latency, and the next looking at the effect of maintaining a constant baseline latency while increasing the packet drop percentage.

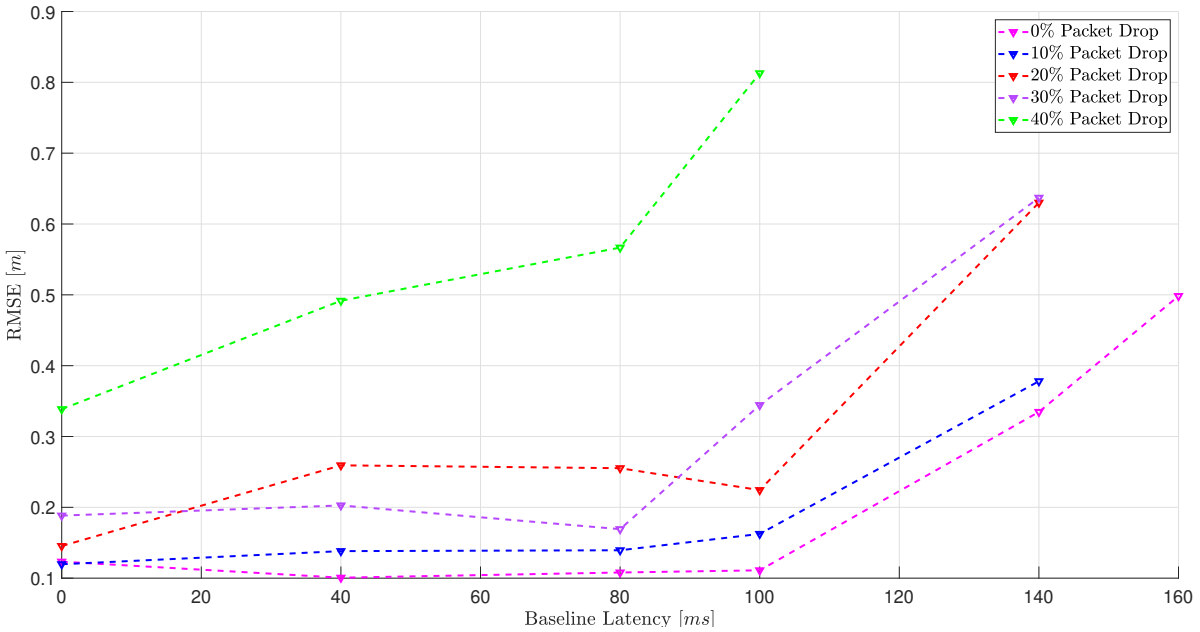

Figure 5.2: RMSE of the vehicle through the DLC-manoeuvre at various baseline latencies and various packet drop percentages.

5.3.1 Variable Baseline Latency

Figure 5.3 shows the effect of increasing the baseline latency while maintaining a constant packet drop percentage (0%). It appears that as the latency was increased, the oscillatory behaviour of the vehicle also increased, this is highlighted in the lateral acceleration plot. Control of the vehicle was generally lost when the oscillations were large enough to cause lateral accelerations beyond the safe handling limit of the vehicle ($7.8\ m/s^2$). The oscillations appeared to increase gradually as the baseline latency was increased.

Another important observation highlighted in Figure 5.3 can be seen when considering the $0\ ms$ latency plot (blue). When looking at the lateral acceleration, before the vehicle had entered the first lane-change it was oscillating from side to side yet when it left the second lane change it returned to a steady state operation. This was due to the initial LQSTR gain value (K_{LQR}) of 1. This was not chosen as an optimal value and resulted in the oscillations presented. However, after the second lane-change, the controller was making use of an optimally calculated gain, a value nearer to 0.6, which resulted in a much smoother response from the vehicle. Starting with a lower initial K_{LQR} value - closer to the calculated optimal value of 0.6 - may have resulted in improved performance

from the controller. It is possible that entering the DLC-manoeuvre with a sub-optimal gain resulted in loss of control of the vehicle at lower baseline latencies, however this was not tested here.

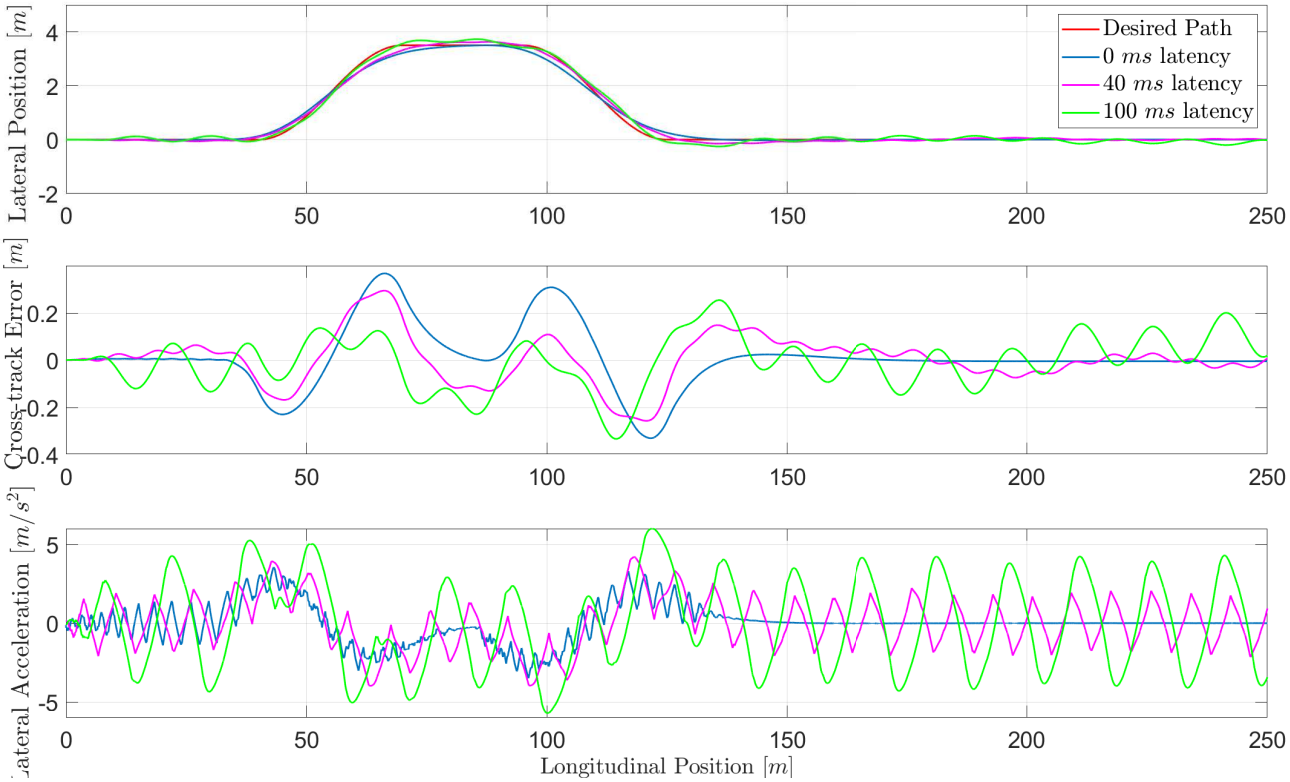

Figure 5.3: Results of the DLC-manoeuvre simulation for various baseline latencies and a packet drop percentage of 0%.

5.3.2 Variable Packet Drop Percentage

Figure 5.4 shows the effect of increasing the percentage of packets dropped while maintaining a constant baseline latency ($0\ ms$). In contrast to the results of Figure 5.3, where an increase in baseline latency caused a smooth increase in oscillations, an increase in packet drop percentage resulted in more unpredictable behaviour from the vehicle, highlighted by the sudden increase in cross-track error of the 40% packet drop plot at around $195\ m$. This was due to the random nature of the packet drops. If packets were dropped with a completely uniform distribution a more predictable pattern, such as the increasing baseline latency simulations, may have been present. Unfortunately, packet drops in networks are generally not uniform, but rather clumped together.

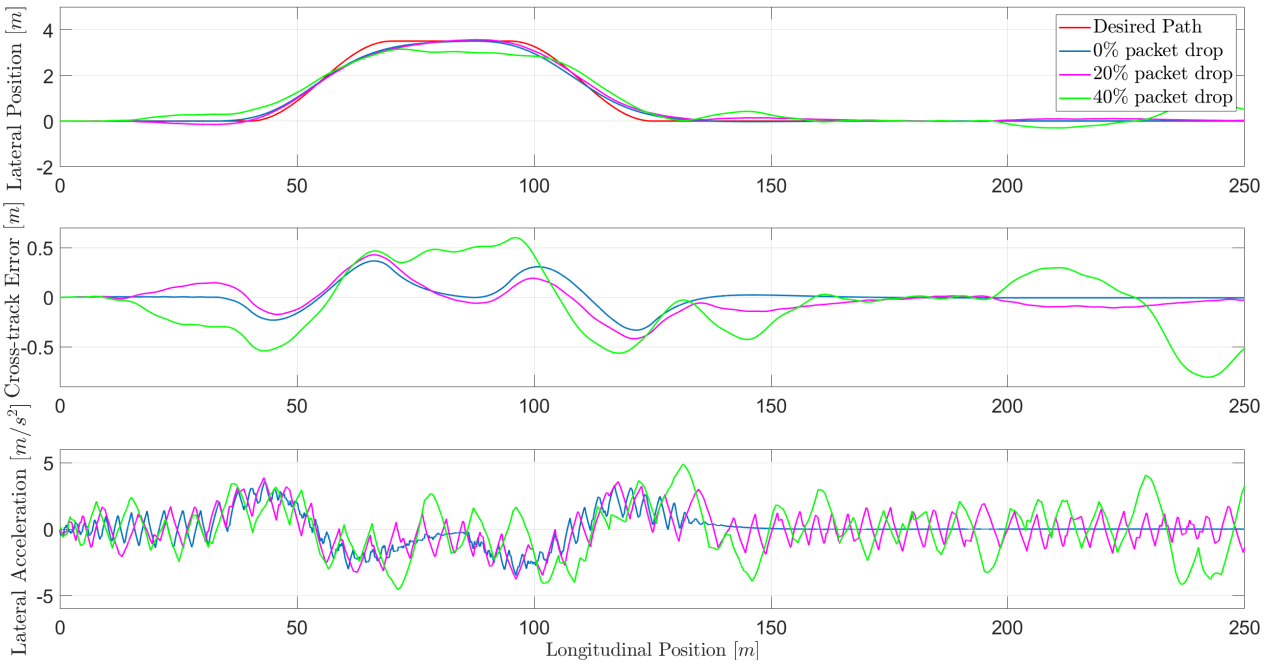

Figure 5.4: Results of the DLC-manoeuvre simulation for various packet drop percentages and a baseline latency of 0 ms.

5.4 The Effect of Extra System Delays

The simulation results presented thus far were conducted under the assumption that there were no extra delays in the system, i.e. the only delays present in the simulation were the transport delays. In practice this is almost never the case. To quantify all the delays present in a system was beyond the scope of this study, however it was known that when a steering robot was installed on the Land Rover 110 - around which the simulation model was built - there was a steering delay on the order of 200 ms. That is, between the steering controller receiving a required steer angle for the front wheels of the vehicle and the front wheels actually reaching that angle, there was a delay of 200 ms. This posed a potential problem: The LQSTR-controller had been shown to have a "safe" operating range of up to 100 ms, meaning that if this controller were to be implemented on the Land Rover 110 it would not be able to safely handle the vehicle as the starting latency of the system would be greater than the maximum "safe" latency of the system. For this reason an extra set of simulations were run with the incorporation of the 200 ms steering delay. Presented below is a section detailing some alterations that were made to the controller to allow for stable and accurate vehicle control

when a steering delay was present, followed by the results of the simulations. Unfortunately the controller making use of these altered parameters showed poor path following ability if used under the assumption of no steering delay.

5.4.1 Extra System Delays - Parameter Alterations

After the steering delay of $200\ ms$ was added to the simulation environment, a new set of controller parameters were calculated, shown in Table 5.3. Some of the parameter values were changed, when compared to Table 5.1, with an additional parameter added. The additional parameter was $K_{Lateral-Error}$, this was a gain applied to the lateral error before it was added to the yaw error when calculating the required yaw rate. By reducing the effect of the lateral error ($K_{Lateral-Error} < 1$) the oscillatory response associated with higher latencies was reduced almost completely. This corresponded with the lateral error gain that was applied by [Kapp, 2014] during experimental testing and was also likely the cause for the steady-state offset that occurred during the experiments.

Table 5.3: Parameter values used when the controller was simulated to be off-site with a steering delay.

Parameter	**Value**	**Parameter**	**Value**
Q_{LQR}	1	$N_{samples}$	10
R_{LQR}	25	$\Phi_{norm-limit}$	0.2
τ_{lat}	$0.4\ s$	f_{sample}	$20\ Hz$
τ_{yaw}	$0.6\ s$	f_{model}	N/A
τ_{acc}	$1\ s$	$f_{vehicle}$	$100\ Hz$
$\bar{N}$	0.4	$f_{Simulink}$	$1000\ Hz$
f_{send}	$50\ Hz$	$K_{Lateral-Error}$	0.1

The total error effect on the controller was also reduced by decreasing the pre-compensation gain $\bar{N}$ to 0.4. Again, this helped with reducing the oscillatory response of the system at higher latencies. To account for both of these decreased error effects, R_{LQR} was increased substantially from 3 to 25. There was a fine balance between decreasing the effects of the errors and increasing the weighting

on the control input, where an incorrect balance lead to either unstable oscillations or a system that did not attempt to follow the required path accurately. It was found that a slightly reduced τ_{lat} of $0.4\ s$ benefited the system as well.

Based on the results obtained without the added steering delay the initial K_{LQR} value was changed from 1 to 0.1 to prevent the vehicle from entering the first lane change while oscillating.

5.4.2 Extra System Delays - Results

Figure 5.5 summarises the results obtained with the altered parameter values, indicating that the system was able to maintain accurate and stable path following at greater baseline transport latencies and greater packet drop percentages when compared to the system that was designed under the assumption of zero steering delay.

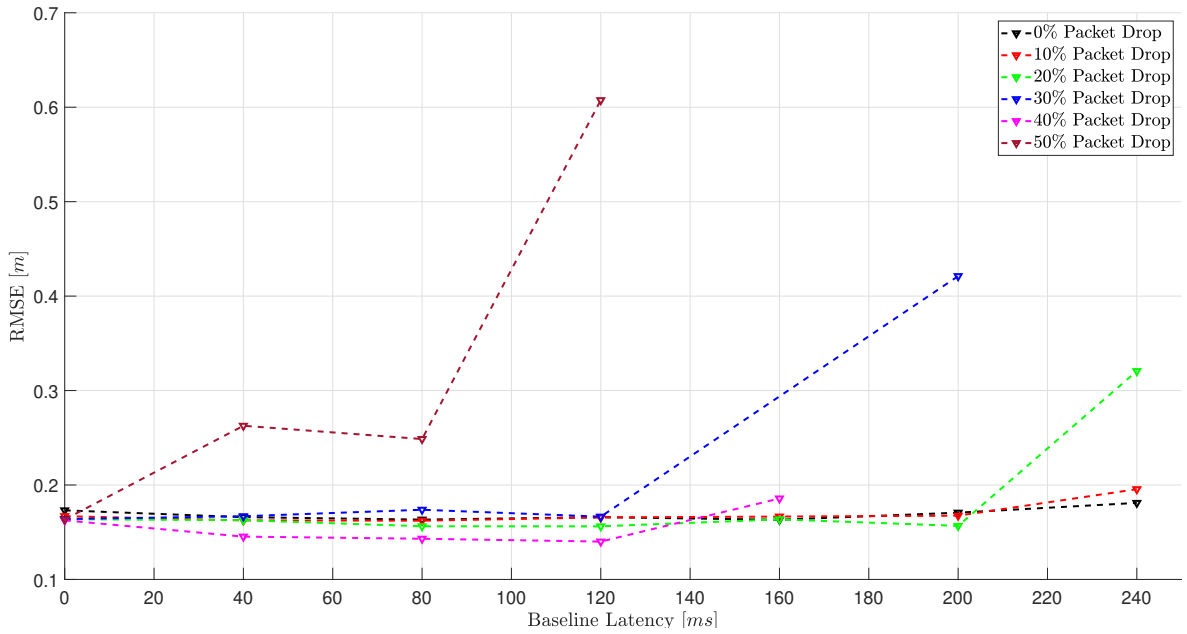

Figure 5.5: RMSE of the vehicle through the DLC-manoeuvre at various baseline latencies and various packet drop percentages, including a steer delay.

It can be seen that as the baseline latency was increased, the RMSE tended to first decrease, then increase for all packet drop percentages considered. This was attributed to the delayed system's vehicle path shifting slightly to the left, aligning it more with the desired path - this can be seen in Figure 5.3 when looking at the $0\ ms$ case and the $80\ ms$ case. When considering an increasing

packet drop percentage while maintaining a constant baseline percentage there was not a clear-cut pattern, except that the RMSE increased drastically between 40% and 50% packet drop percentages.

The tabulated summary is given in Table C.2 in Appendix C. Comparing Tables C.1 and C.2 shows that the alterations made to the parameters allowed for adequate vehicle control over a much larger variation in baseline latencies and packet drop percentages. The alterations in the parameters allowed for baseline latencies of up to $260\ ms$ - with a 0% packet drop percentage - and packet drop percentages of up to 60% - for baseline latencies of up to $80\ ms$. The "safe" operating region was increased to packet drop percentages of 40% to 50% for baseline latencies below $120\ ms$ to $140\ ms$.

Figure 5.6 shows the effect of increasing the baseline latency while maintaining a packet drop percentage of 0%, while Figure 5.7 shows the effect of maintaining a constant baseline latency of $0\ ms$ while increasing the packet drop percentage.

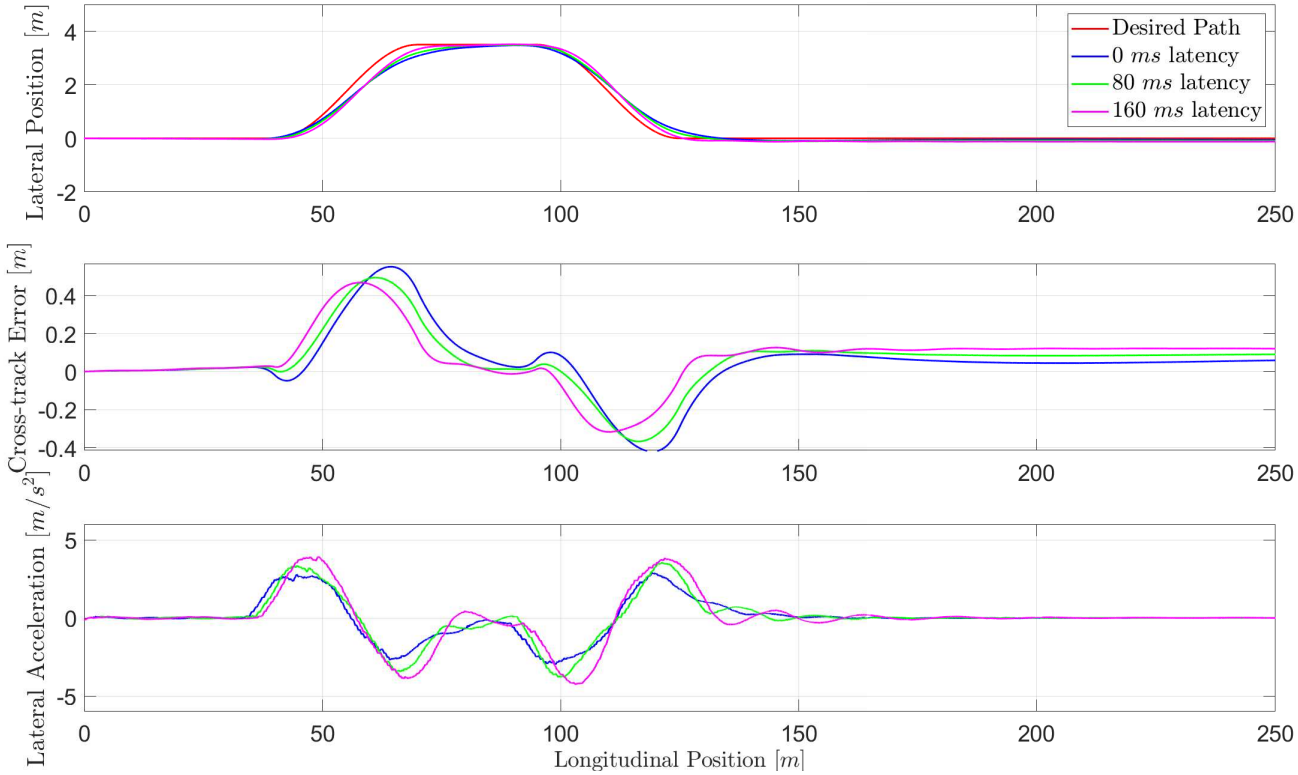

Figure 5.6: Results of the DLC-manoeuvre simulation for various baseline latencies and a packet drop percentage of 0% - including a steer delay.

Much like the results of the previous section, increasing the baseline latency while maintaining a 0% packet drop percentage resulted in a gradual increase in the oscillatory response of the vehicle,

but due to the reduced effect of the errors on the system the magnitude of the oscillations were less severe. This can be seen when looking at the $160\ ms$ case, where the oscillations had largely died-out by $200\ m$. Figure 5.7, where the packet drop percentage was increased and the baseline latency was maintained at $0\ ms$, shows a similar pattern to Figure 5.6 - this is in contrast to the results of the previous section, where an increase in packet drop percentage lead to a more sudden loss of control. Again this was attributed to the reduced error effect on the system.

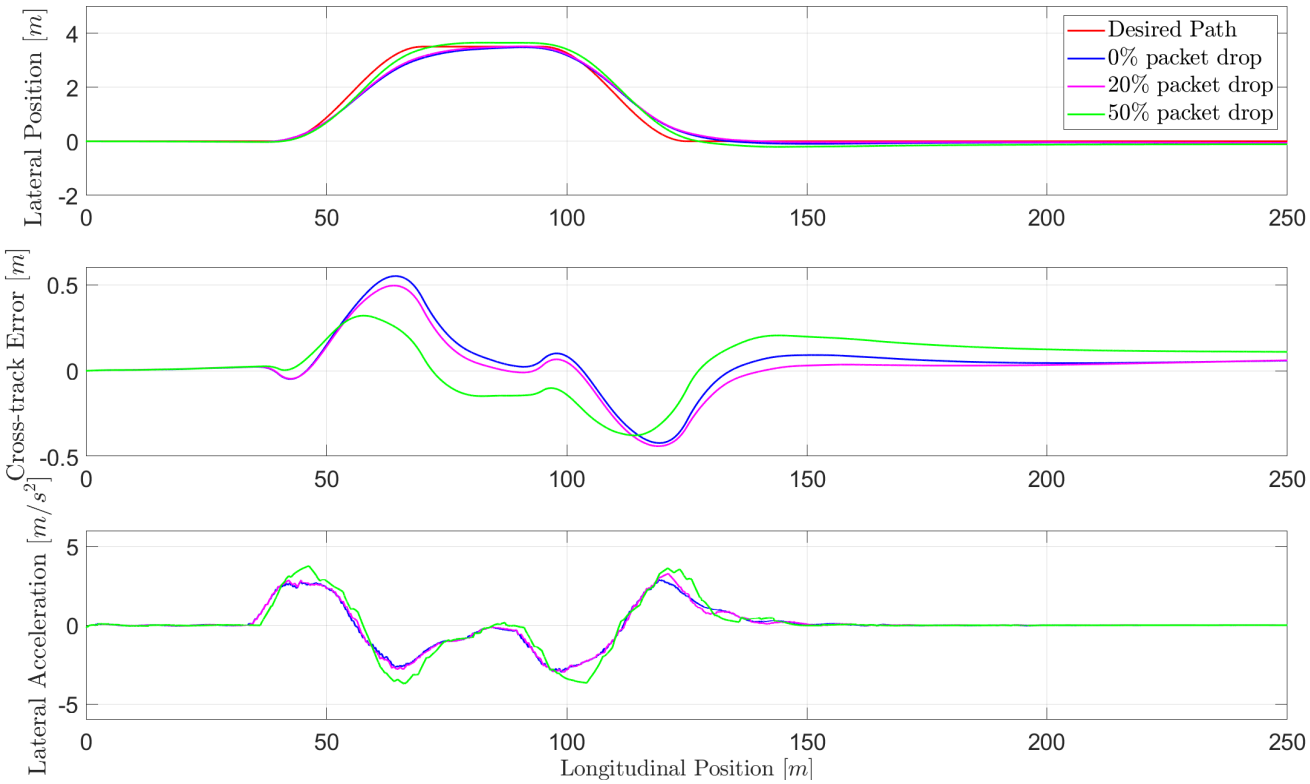

Figure 5.7: Results of the DLC-manoeuvre simulation for various packet drop percentages and a baseline latency of $0\ ms$ - including a steer delay.

5.5 Conclusion

The conclusion presented in this chapter is aimed towards the simulations conducted that included a $200\ ms$ delay in the steering system of the vehicle. It is believed that these results would be more representative when compared to real-world implementations. The results of these simulations bode well for future implementation of such off-site control systems as being able to control a vehicle safely when up to 60% of the packets are lost means that networks affected by the hidden terminal

problem, or by intermittent interference signals, may still be able to safely control a vehicle to within a relatively high degree of accuracy. With a safe operating latency of $120\ ms$ to $140\ ms$ with a maximum packet drop percentage of 40% to 50%, vehicles could theoretically be safely controlled over very large networks, which are inherently ladened with many hidden terminals and where relatively high latencies are bound to occur with higher chances of packet drops.

Chapter 6

Conclusion and Recommendations

The following chapter briefly highlights the key points of this study, with conclusions drawn from the results obtained. This is followed by a section covering recommendations for any future work that may be carried out in succession to this study.

6.1 Conclusion

The importance of ADAS has been highlighted with a specific focus on the ability of lateral CA systems to greatly increase road user safety. To realise lateral CA systems when all aspects of control are present on board the vehicle is an ongoing research topic, adding the effect of transport latencies and packet drops to such systems only further complicates the matter. This further complication may be seen as unnecessary when the majority of systems in place at the moment are all onboard, but off-site control presents an immense opportunity to the world of autonomous self-driving vehicles (industrial and/or passenger) for a variety of reasons - increased sensor range, central control monitoring, etc..

This study aimed to quantify, through simulations, the effects of transport latencies and dropped packets on the lateral controllability of a vehicle that was being wirelessly controlled from off-site. To enable real-world representative simulations required the generation of a latency model that accurately captured the properties of a real-world network, a controller that could handle a large amount of uncertainties in the controlling environment, and a fully-validate vehicle simulation model. Only lateral control was investigated through simulations of a DLC-manoeuvre executed at the common suburban speed limit of $60\ km/h$ using the simulation model of a Land Rover Defender 110.

The chosen protocol for the wireless network was the Nv2 protocol by Mikrotik which made use of TDMA at the MAC layer, which is seen as a possible solution - along with STDMA - to the hidden terminal problem and the possibility of unbounded delays - both present in networks that make use of the standard CSMA/CA MAC layer protocol. It was found that to simulate latencies experienced in an Nv2 protocol network, a three-mean-normal-distribution was required. This latency model, combined with an additional baseline latency, was used during simulations to model real-world latencies between a vehicle and its controller more accurately.

The controller used was based on a previously tested LQSTR-controller with modifications made to decrease computational complexity while increasing path following ability. These modifications included reducing the model order to one and altering some of the parameter values. The reduction in model order resulted in single element state-space matrices which meant that, when solving for the ARX-model's parameters, there were no longer any matrix inversions. The lateral error preview time was changed to be speed dependent, meaning that the lateral error preview distance became a function of the square of speed, resulting in a preview distance with a greater variation than a preview distance only based on speed. It was shown that the modifications made to the controller resulted in stable path following at speeds of up to $150\ km/h$. Accurately traversing the DLC-manoeuvre at such high speeds was only possible if some corners were cut, and as such only speeds of up to $80\ km/h$ were deemed adequate in both stability and accuracy when the controller was simulated to be onboard.

Off-site latency simulations under the assumption of zero steering delay showed that there was a "safe" operating region, with baseline latencies from $0\ ms$ to $100\ ms$ and packet drop percentages from 0% to 40%. Control of the vehicle was lost, or deemed unsatisfactory, at packet drop percentages greater than 40% while for packet drop percentages of 0%, the vehicle could still be satisfactorily controlled up to baseline latencies of $140\ ms$. If a steering delay of $200\ ms$ was incorporated into the simulation, minor alterations to the controller's parameters were required to increase the "safe" operating range to $120\ ms$ to $140\ ms$ baseline latency with 40% to 50% of packets dropped. It was assumed that packet order was maintained throughout the simulation, even at higher baseline latencies. Of these results the maximum allowable packet drop percentage deserves special attention. Realising safe control when 50% of the information (packets) is lost suggests that even highly congested networks - riddled with hidden terminals and possibly interfering signals - could still allow stable off-site vehicle control.

An important result from the simulations conducted with a steering delay of $200\ ms$ is that the total latency that the system could handle in simulations was the $260\ ms$ transport latency plus the $200\ ms$ steering delay - for a packet drop percentage of 0%. This suggests that a vehicle can be adequately controlled if there is nearly $0.5\ s$ of latency between the vehicle acquiring its states and it realising a calculated control action. As shown in Chapter 3, this is a latency well beyond that which is experienced by modern wireless networks.

6.2 Recommendations

The following list is comprised of possible ideas for future work that could be carried out as an extension to the work conducted in this study. It is believed that before fully-autonomous off-site control can be implemented in public, or private, sectors the items in this list, amongst others, will need to be studied in great detail.

- The effect that the distribution of packet drops has on vehicle control needs to be studied further, as a scenario where all dropped packets occur one after the other is likely to fail sooner than a scenario where the dropped packets are uniformly distributed.
- A study into the effect that packet order has on the lateral controllability of the system should be conducted. Although it is quite simple to prevent out of order packets in a network by making use of timestamps - thereby treating out-of-order packets as dropped packets, the effect they have on the system may produce interesting results. It is believed that the simulations presented in this study represent a best-case scenario, i.e. using the information contained in out of order packets would only increase the instability of the system.
- Preliminary work showed that altering some of the LQSTR controller parameters could increase the maximum allowable transport latency of the system. Further study would be required to find exactly what the maximum allowable transport latency value is.
- Experimentally confirming the simulation results. The experimental environment is fraught with unknowns, extra delays, and noisy measurements that are not included in the simulation model. An experimental study may reveal that an entirely different set of controller parameters are required for safe and accurate vehicle control.

- To incorporate the lateral controller of this study into a larger system - comprised of longitudinal control - would require an analysis into the effect of transport latencies and packet drops on said longitudinal controller. Only then can the full effect of transport latencies and packet drops on off-site autonomous vehicle control start to be realised.

- Although protocols making use of CSMA/CA at the MAC layer have already been shown to underperform when used on congested networks, it may be prudent to build a latency model around the 802.11p protocol as, for the time being, it is the industry standard. The same should be done when LTE-V2V (5G) is formally rolled-out as it could eventually stand to replace 802.11p as the de-facto standard in wireless vehicle communication systems.

- The effect of moving nodes should be incorporated in into the latency model. Since nodes in a VANET are generally moving - at speeds of up to $120\ km/h$ - the effect this motion has on a packet's latency could possibly be of importance if the distribution results of Chapter 3 change significantly.

- As with the effect of moving nodes, the effect that an obscured line-of-site has on a packet's latency and drop probability should also be quantified.

- As shown in Chapter 5.4, to accommodate the steering delay of $200\ ms$ the controller's parameters had to be altered. This lead to a controller that could not adequately handle the vehicle if the steering delay was not present. It may be possible to set up some of the parameters as functions of the system's latency. To realise such a formulation would require further study.

Appendices

Appendix A

Vehicle Properties

Table A.1 contains the properties of the Land Rover platform.

Table A.1: Properties of the fully-kitted Land Rover Defender 110 around which the simulation model was built.

Property	**Value**
Total Mass	2047 kg
Sprung Mass	1576 kg
Mass Moment of Inertia	2057 kgm^2
Yaw Moment of Inertia	2475 kgm^2
Roll Moment of Inertia	744 kgm^2
Front Axle to CG	1.54 m
Rear Axle to CG	1.25 m
Track Width	1.486 m
Distance Between Left and Right $\mathbf{4}S_{\mathbf{4}}$ Struts	1.009 m
Front Roll Centre Height	0.3985 m
Rear Roll Centre Height	0.517 m
Roll Centre to CG	0.14 m
Front Tyre Cornering Stiffness	36821 N/rad
Rear Tyre Cornering Stiffness	36822 N/rad

Appendix B

Lateral Error Calculation

All the steps required to return Equation 4.25 are presented below, with accompanying Figures: B.1, B.2, and B.3. Figure B.1 shows the original system where ψ_{actual} is the current yaw angle of the vehicle through its centre line. P is the preview point τ_{lat} seconds ahead of the vehicle, calculated using the current speed of the vehicle and its current yaw angle. CG is the centre-of-gravity of the vehicle in the (X, Y) plane. e_{lat} is the lateral error of the vehicle, between the preview point and the path, which is normal to the centre line of the vehicle that passes through the CG and point P. ψ_{pe} is the yaw angle error of the vehicle between the preview point, calculated with τ_{lat}, and point B on the path.

The first step is to translate the system such that the CG becomes the origin, with the new x-axis aligned to the centre-line of the vehicle, the result of this translation is shown in Figure B.2, where the prime variables correspond to their non-prime counterparts of Figure B.1.

Next, the procedure for rotating a point through an angle about the origin of [Owen, 1999] was used. The idea is to rotate P′ and B′ such that P′ lies on the x-axis. This required a rotation angle of $-\psi_{actual}$. A positive rotation for a counter-clockwise rotation was assumed.

This rotation results in Figure B.3. Since e_{lat} is normal to the line CG′-P′, and by extension CG′-P″, e_{lat} has only a y-component equal to the magnitude of B″. The calculation of $y_{B''}$ is thus given by Equation B.1, where the coordinates of B′ are given by Equation B.2..

$$y_{B''} = y_{B'} \cos(-\psi_{actual}) + x_{B'} sin(-\psi_{actual}) \tag{B.1}$$

$$y_{B'} = y_B - y_{CG}$$
$$x_{B'} = x_B - x_{CG} \tag{B.2}$$

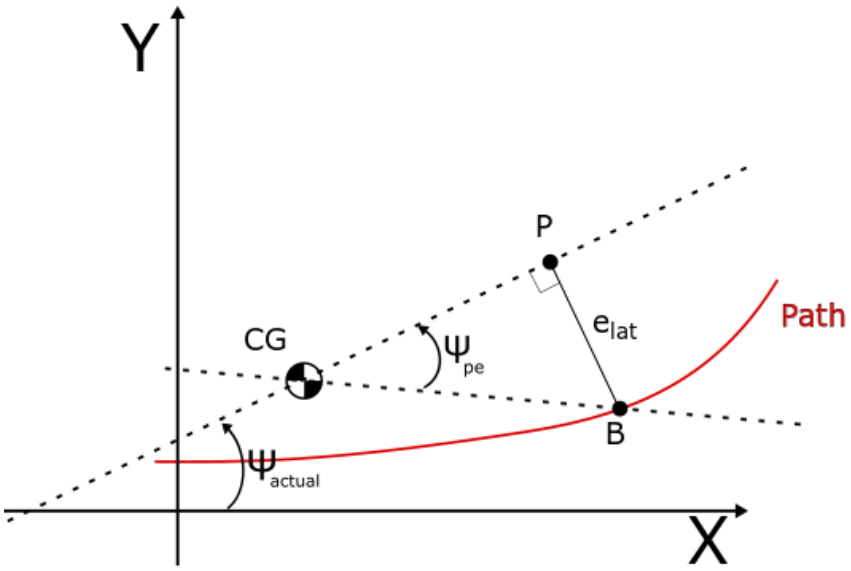

Figure B.1: Calculation of lateral error, system prior to any translations and rotations.

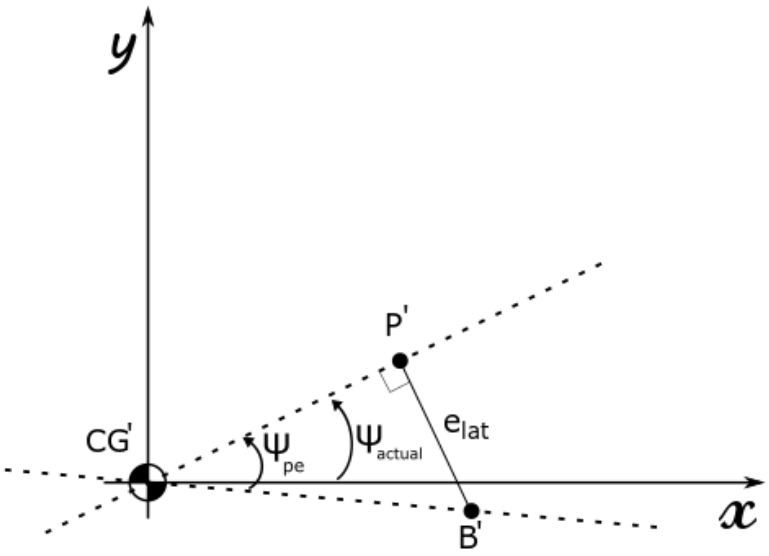

Figure B.2: Calculation of lateral error, system after translation.

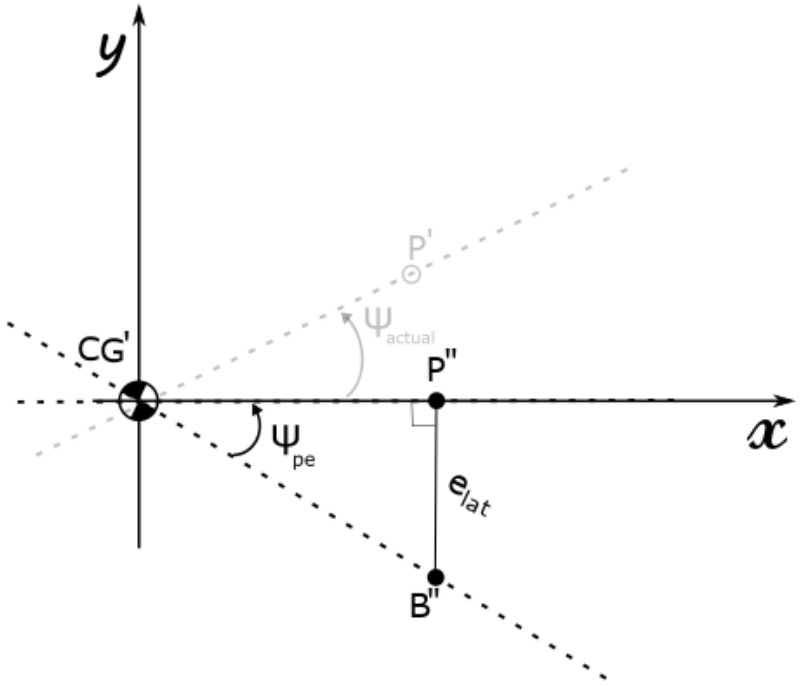

Figure B.3: Calculation of lateral error, system after rotation.

Appendix C

Wireless Simulation Results

A summary of all the latency simulation results is presented in Table C.1. The blank cells represent simulations that failed, either through a loss of control or an RMSE value that was deemed too large for control to be considered successful.

Table C.1: Summary of the wireless simulations' results, RMSE for each simulation.

		Percentage of Packets Dropped [%]				
		0	**10**	**20**	**30**	**40**
Baseline Latency [ms]	**0**	0.1233	0.1197	0.1453	0.1883	0.3385
	20	0.1265	0.1199	0.2551	0.2299	0.3464
	40	0.1007	0.1382	0.2593	0.2027	0.4916
	60	0.1026	0.1571	0.1797	0.3624	0.5639
	80	0.1080	0.1394	0.2552	0.1692	0.5667
	100	0.1111	0.1625	0.2243	0.3443	0.8128
	120	0.2464	0.2725	0.3576	0.7937	\
	140	0.3347	0.3780	0.6301	0.6371	\
	160	0.4983	\	\	\	\
	180	0.5837	\	\	\	\
	200	6.3128	\	\	\	\

An RMSE greater than 2 generally suggested an inability to safely control the vehicle, except for

the simulation of 200 ms baseline latency with a 0% packet drop percentage, where control was only lost right at the end of the simulation.

Table C.2 summarises all the wireless simulations' results that were carried out with an added steering delay of 200 ms. The same format is used as in Table C.1, where a blank cell indicates a simulation that failed, either due to the vehicle slipping out or the RMSE being too large.

Table C.2: Summary of the wireless simulations' results including a steering delay of 200 ms, RMSE for each simulation.

		Percentage of Packets Dropped [%]						
		0	**10**	**20**	**30**	**40**	**50**	**60**
Baseline Latency [ms]	**0**	0.1731	0.1670	0.1637	0.1639	0.1626	0.1634	0.2808
	20	0.1682	0.1654	0.1626	0.1610	0.1534	0.2733	0.4232
	40	0.166	0.1625	0.1628	0.1669	0.1453	0.2627	0.4575
	60	0.1683	0.1629	0.1564	0.1721	0.1493	0.2435	0.5791
	80	0.1633	0.1622	0.1563	0.1738	0.1431	0.2487	0.8381
	100	0.1687	0.1673	0.1606	0.1633	0.1378	0.3097	\
	120	0.1659	0.1657	0.1562	0.1665	0.1401	0.6075	\
	140	0.1600	0.1664	0.1620	0.1772	0.1495	\	\
	160	0.1632	0.1666	0.1636	0.1978	0.1857	\	\
	180	0.1706	0.1733	0.1613	0.2386	0.3767	\	\
	200	0.1707	0.1675	0.1567	0.4211	\	\	\
	220	0.1775	0.1801	0.1814	\	\	\	\
	240	0.1810	0.1955	0.3206	\	\	\	\
	260	0.2382	0.5007	\	\	\	\	\

Bibliography

[Abdelgader and Lenan, 2014] Abdelgader, A. and Lenan, W. (2014). The physical layer of the ieee 802.11 p wave communication standard: the specifications and challenges. In *Proceedings of the world congress on engineering and computer science*, volume 2, page 71.

[Albulet, 2017] Albulet, M. (2017). Spacex v-band non-geostationary satellite system. https://licensing.fcc.gov/myibfs/download.do?attachment_key=1190019. [Online; accessed 2018-11-20].

[ARIB, 2001] ARIB (2001). Dedicated short-range communication system. http://www.arib.or.jp/english/html/overview/doc/5-STD-T75v1_0-E2.pdf. [Online; accessed 2018-11-18].

[arsTECHNICA, 2018] arsTECHNICA (2018). Spacex hits two milestones in plan for low-latency satellite broadband. https://arstechnica.com/information-technology/2018/02/spacexs-satellite-broadband-nears-fcc-approval-and-first-test-launch/. [Online; accessed 2018-11-20].

[Atallah et al., 2015] Atallah, R., Khabbaz, M., and Assi, C. (2015). Throughout analysis of ieee 802.11 p-based multi-hop v2i communications. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2015 IEEE 16th International Symposium on a*, pages 1–6. IEEE.

[Bakker et al., 1989] Bakker, E., Pacejka, H. B., and Lidner, L. (1989). A new tire model with an application in vehicle dynamics studies. *SAE transactions*, pages 101–113.

[Beal, nd] Beal, V. (n.d.). Mac layer - media access control layer. https://www.webopedia.com/TERM/M/MAC_layer.html. [Online; accessed 2018-11-18].

[Botha, 2011] Botha, T. (2011). High speed autonomous off-road vehicle steering. Master's thesis, University of Pretoria.

[Caveney and Dunbar, 2012] Caveney, D. and Dunbar, W. B. (2012). Cooperative driving: Beyond v2v as an adas sensor. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 529–534. IEEE.

[Cecchini et al., 2017] Cecchini, G., Bazzi, A., Masini, B. M., and Zanella, A. (2017). Performance comparison between ieee 802.11 p and lte-v2v in-coverage and out-of-coverage for cooperative awareness. In *Vehicular Networking Conference (VNC), 2017 IEEE*, pages 109–114. IEEE.

[Chong et al., nd] Chong, C. Y., Wee, R. S. K., Lian, S. S., and Hui, T. J. (n.d.). Mobile ad hoc networking. https://www.dsta.gov.sg/docs/default-source/dsta-about/dh02200607-mobile-ad-hoc-networking.pdf?sfvrsn=2. [Online; accessed 2018-11-18].

[Cisco, 2018] Cisco (2018). 802.11ac: The fifth generation of wi-fi. https://www.cisco.com/c/dam/en/us/products/collateral/wireless/aironet-3600-series/white-paper-c11-713103.pdf. [Online; accessed 2018-11-19].

[Commission, 2016] Commission, E. (2016). C-its platform final report january 2016. https://ec.europa.eu/transport/sites/transport/files/themes/its/doc/c-its-platform-final-report-january-2016.pdf. [Online; accessed 2018-11-18].

[Commission, 2017] Commission, E. (2017). 2016 road safety statistics: What is behind the figures? http://europa.eu/rapid/press-release_MEMO-17-675_en.htm. [Online; accessed 2018-11-18].

[Coulter, 1992] Coulter, R. C. (1992). Implementation of the pure pursuit path tracking algorithm. Technical report, Carnegie-Mellon UNIV Pittsburgh PA Robotics INST.

[Cronje et al., 2008] Cronje, P. H. et al. (2008). *Improving off-road vehicle handling using an active anti-roll bar*. PhD thesis, University of Pretoria.

[Els, 2007] Els, P. S. (2007). The ride comfort vs. handling compromise for off-road vehicles. *Journal of Terramechanics*, 44(4):303–317.

[Filippi et al., 2016] Filippi, A., Moerman, K., Daalderop, G., Alexander, P. D., Schober, F., and Pfliegl, W. (2016). Why 802.11 p beats lte and 5g for v2x. *A white paper by NXP Semiconductors, Cohda Wireless, and Siemens*, 21.

[Gates, 2015] Gates, A. (2015). Getting familiar with wi-fi channels? wlan back to basics. https://blog.aerohive.com/getting-familiar-with-wi-fi-channels-wlan-back-to-basics/. [Online; accessed 2018-11-18].

[Genta, 1997] Genta, G. (1997). *Motor vehicle dynamics: modeling and simulation*, volume 43. World Scientific.

[Gupta, 2012] Gupta, S. (2012). N-streme and nv2. https://mum.mikrotik.com//presentations/IN12/soumil.pdf. [Online; accessed 2018-11-18].

[Harding et al., 2014] Harding, J., Powell, G., Yoon, R., Fikentscher, J., Doyle, C., Sade, D., Lukuc, M., Simons, J., and Wang, J. (2014). Vehicle-to-vehicle communications: Readiness of v2v technology for application. Technical report, n.i.

[Hossain et al., 2016] Hossain, M. A., Elshafiey, I., and Al-Sanie, A. (2016). Cooperative vehicular positioning with vanet in urban environments. In *Applied Electromagnetics (APACE), 2016 IEEE Asia-Pacific Conference on*, pages 393–396. IEEE.

[Jiang and Delgrossi, 2008] Jiang, D. and Delgrossi, L. (2008). Ieee 802.11 p: Towards an international standard for wireless access in vehicular environments. In *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pages 2036–2040. IEEE.

[Jiangfeng et al., 2009] Jiangfeng, W., Feng, G., Fei, Y., and Shaoxuan, S. (2009). Design of wireless positioning algorithm of intelligent vehicle based on vanet. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 1098–1102. IEEE.

[Journeys, 2018] Journeys, S. (2018). The safe system approach. https://www.saferjourneys.govt.nz/about-safer-journeys/the-safe-system-approach/. [Online; accessed 2018-11-21].

[Kapp, 2014] Kapp, M. (2014). Robust high speed autonomous steering of an off-road vehicle. Master's thesis, University of Pretoria.

[Kapp, 2017] Kapp, R. A. (2017). Wireless vehicle-in-the-loop platform for development of driver-assistance systems. Master's thesis, University of Pretoria.

[Khairnar and Kotecha, 2013] Khairnar, V. D. and Kotecha, K. (2013). Performance of vehicle-to-vehicle communication using ieee 802.11 p in vehicular ad-hoc network environment. *arXiv preprint arXiv:1304.3357*.

[Kim and Lee, 2015] Kim, J. and Lee, I. (2015). 802.11 wlan: history and new enabling mimo techniques for next generation standards. *IEEE Communications Magazine*, 53(3):134–140.

[Klingler et al., 2015] Klingler, F., Dressler, F., and Sommer, C. (2015). Ieee 802.11 p unicast considered harmful. In *Vehicular Networking Conference (VNC), 2015 IEEE*, pages 76–83. IEEE.

[Lopez-Aguilera et al., 2010] Lopez-Aguilera, E., Casademont, J., and Cotrina, J. (2010). Propagation delay influence in ieee 802.11 outdoor networks. *Wireless networks*, 16(4):1123–1142.

[Lyamin et al., 2018] Lyamin, N., Vinel, A., Jonsson, M., and Bellalta, B. (2018). Cooperative awareness in vanets: On etsi en 302 637-2 performance. *IEEE Transactions on Vehicular Technology*, 67(1):17–28.

[Macadam, 2003] Macadam, C. C. (2003). Understanding and modeling the human driver. *Vehicle system dynamics*, 40(1-3):101–134.

[MacHardy et al., 2018] MacHardy, Z., Khan, A., Obana, K., and Iwashina, S. (2018). V2x access technologies: Regulation, research, and remaining challenges. *IEEE Communications Surveys & Tutorials*.

[Michigin et al., nda] Michigin, Mellon, C., and Mercy, D. (n.d.a). Inverted pendulum: Digital controller design. http://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum &section=ControlDigital. [Online; accessed 2018-11-21].

[Michigin et al., ndb] Michigin, Mellon, C., and Mercy, D. (n.d.b). Inverted pendulum: State-space methods for controller design. http://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum &section=ControlStateSpace. [Online; accessed 2018-11-21].

[Miller, 2012] Miller, K. (2012). Calculating optical fiber latency. https://www.m2optics.com/blog/bid/70587/Calculating-Optical-Fiber-Latency. [Online; accessed 2018-11-21].

[Owen, 1999] Owen, G. S. (1999). 2d rotation. https://www.siggraph.org/education/materials/HyperGraph/modeling/mod_tran/2drota.htm. [Online; accessed 2018-12-05].

[Polack et al., 2017] Polack, P., Altché, F., d'Andréa Novel, B., and de La Fortelle, A. (2017). The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? In *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pages 812–818. IEEE.

[RFWirelessWorl, nd] RFWirelessWorl (n.d.). 802.11 vs 802.16 — difference between 802.11 and 802.16. http://www.rfwireless-world.com/Terminology/802-11-versus-802-16.html. [Online; accessed 2018-11-20].

[Rouse, 2005] Rouse, M. (2005). 802.16. https://searchmobilecomputing.techtarget.com/definition/80216. [Online; accessed 2018-11-20].

[Sasaki et al., 2004] Sasaki, T., Miyata, T., and Kawashima, K. (2004). Development of remote control system of construction machinery using pneumatic robot arm. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 1, pages 748–753. IEEE.

[silicon, 2018] silicon (2018). Tales in tech history: Wimax. https://www.silicon.co.uk/networks/tales-tech-history-wimax-227889. [Online; accessed 2018-11-20].

[simTD, 2018] simTD (2018). Projects. https://www.eict.de/en/projects/. [Online; accessed 2018-11-20].

[Sjoberg et al., 2011] Sjoberg, K., Uhlemann, E., and Strom, E. G. (2011). How severe is the hidden terminal problem in vanets when using csma and stdma? In *Vehicular Technology Conference (VTC Fall), 2011 IEEE*, pages 1–5. IEEE.

[Snider et al., 2009] Snider, J. M. et al. (2009). Automatic steering methods for autonomous automobile path tracking. *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08*.

[Tahmasbi-Sarvestani et al., 2017] Tahmasbi-Sarvestani, A., Mahjoub, H. N., Fallah, Y. P., Moradi-Pari, E., and Abuchaar, O. (2017). Implementation and evaluation of a cooperative vehicle-to-pedestrian safety application. *IEEE Intelligent Transportation Systems Magazine*, 9(4):62–75.

[Thoresson et al., 2014] Thoresson, M. J., Botha, T. R., and Els, P. S. (2014). The relationship between vehicle yaw acceleration response and steering velocity for steering control. *International Journal of Vehicle Design*, 64(2-4):195–213.

[Thoresson et al., 2007] Thoresson, M. J. et al. (2007). *Efficient gradient-based optimisation of suspension characteristics for an off-road vehicle*. PhD thesis, University of Pretoria.

[Tomar et al., 2017] Tomar, R., Prateek, M., and Sastry, H. G. (2017). Analysis of beaconing performance in ieee 802.11 p on vehicular ad-hoc environment. In *Electrical, Computer and Electronics (UPCON), 2017 4th IEEE Uttar Pradesh Section International Conference on*, pages 692–696. IEEE.

[Toyota, nd] Toyota (nd). Hill-assist control (hac)/downhill-assist control (dac). . [Online; accessed 2020-02-04].

[Triantafyllou and Grosenbaugh, 1991] Triantafyllou, M. S. and Grosenbaugh, M. A. (1991). Robust control for underwater vehicle systems with time delays. *IEEE Journal of Oceanic Engineering*, 16(1):146–151.

[Triantafyllou and Hover, 2003] Triantafyllou, M. S. and Hover, F. S. (2003). *Maneuvering and control of marine vehicles*, chapter 19, pages 92–98. Massachusetts of Institute of Technology.

[USDT, nda] USDT (n.d.a). Intelligent transportation systems - connected vehicle safety pilot. https://www.its.dot.gov/research_archives/safety/cv_safetypilot.htm. [Online; accessed 2018-11-19].

[USDT, ndb] USDT (n.d.b). The its standards program. https://www.standards.its.dot.gov/Factsheets/Factsheet/80. [Online; accessed 2018-11-18].

[Uys et al., 2006] Uys, P., Els, P., Thoresson, M., Voigt, K., and Combrinck, W. (2006). Experimental determination of moments of inertia for an off-road vehicle in a regular engineering laboratory. *International Journal of Mechanical Engineering Education*, 34(4):291–314.

[Van de Geer et al., 2008] Van de Geer, S. A. et al. (2008). High-dimensional generalized linear models and the lasso. *The Annals of Statistics*, 36(2):614–645.

[VECHI.CO, nd] VECHI.CO (n.d.). Iso double lane change test. http://www.vehico.com/index.php/en/applications/iso-lane-change-test. [Online; accessed 2018-11-20].

[VTS, 2015] VTS, I. (2015). First toyota cars to include v2v and v2i communication by the end of 2015. http://sites.ieee.org/connected-vehicles/2015/09/30/first-toyota-cars-to-include-v2v-and-v2i-communication-by-the-end-of-2015/. [Online; accessed 2018-11-18].

[VTS, 2018] VTS, I. (2018). Volkswagen group will roll-out ieee 802.11p standard in volume models from 2019. http://sites.ieee.org/connected-vehicles/2018/02/15/volkswagen-group-will-roll-out-ieee-80211p-standard-in-volume-models-from-2019/. [Online; accessed 2018-11-19].

[Wellnitz and Wolf, 2010] Wellnitz, O. and Wolf, L. (2010). On latency in ieee 802.11-based wireless ad-hoc networks. In *IEEE 5th International Symposium on Wireless Pervasive Computing 2010*, pages 261–266. IEEE.

[WHO, 2015] WHO, W. H. O. (2015). *GLOBAL STATUS REPORT ON ROAD SAFETY 2015*.

[Yoshida, 2014] Yoshida, J. (2014). Nxp beats qualcomm, gets first v2v design win. https://www.eetimes.com/document.asp?doc_id=1324052. [Online; accessed 2018-11-19].