ISTANBUL SABAHATTIN ZAIM UNIVERSITY
GRADUATE EDUCATION INSTITUTE
DEPARTMENT OF COMPUTER ENGINEERING

AUTONOMOUS VEHICLE PLATOON MODELING AND CONTROL USING PID AND LINEAR QUADRATIC REGULATOR

Ph.D. DISSERTATION

Alex GUNAGWERA

Istanbul
July, 2022

ISTANBUL SABAHATTIN ZAIM UNIVERSITY
GRADUATE EDUCATION INSTITUTE
DEPARTMENT OF COMPUTER ENGINEERING

AUTONOMOUS VEHICLE PLATOON MODELING AND CONTROL USING PID AND LINEAR QUADRATIC REGULATOR

Ph.D. DISSERTATION

Alex GUNAGWERA

Supervisor
Asst. Prof. Dr. Aydın Tarık ZENGİN

Istanbul
July, 2022

APPROVAL PAGE

This study has been approved in partial fulfillment of the requirements for Ph.D. Degree in Computer Engineering

Chair person of jury…………………………………..….(Signature)

Asst. Prof. Dr. Aydın Tarık ZENGİN

Member of jury…..………………………………………(Signature)

Prof. Dr. Mustafa Nizamettin ERDURAN

Member of jury…..………………………………………(Signature)

Assoc. Prof. Dr. Gökhan ERDEMİR

Member of jury…..……………………………………..(Signature)

Prof. Dr. Ahmet Emin KUZUCUOĞLU

Member of jury…..…………………………………..…(Signature)

Asst. Prof. Dr. Haluk BAYRAM

Approval by

Graduate Education Institute

Signature

Director,

Graduate Education Institute

DECLARATION OF SCIENTIFIC ETHICS AND ORIGINALITY

This is to certify that this PhD dissertation titled "Autonomous Vehicle Platoon Modeling And Control Using PID And Linear Quadratic Regulator" is my own work and I have acted according to scientific ethics and academic rules while producing it. I have collected and used all information and data according to scientific ethics and guidelines on thesis writing of Sabahattin Zaim University. I have fully referenced, in both the text and bibliography, all direct and indirect quotations and all sources I have used in this work.

Signature

Alex GUNAGWERA

July, 2022

PREFACE

I have been intrigued by the concept of self-driving cars since the moment I was introduced to them. A group of self-driving moving in unison while using less fuel, less space, with relatively lower chances of road-accidents, just mind-blowing. In this dissertation, with the aim of fulfilling one of the requirements for graduation in the Department of Computer Engineering in Istanbul Sabahattin Zaim University, this dissertation was written. It presents a relatively cheaper and cost-efficient approach of modeling and controlling an autonomous platoon of vehicles using the PID and LQR controllers. It was the perfect opportunity during the most trying of times so far.

I would like to thank my advisor, Asst. Prof. Dr. Aydın Tarık Zengin for his excellent guidance, support, and patience. I would also like to extend my sincere gratitude to Prof. Dr. Nizamettin Erduran and Assoc. Prof. Dr. Gökhan Erdemir for their invaluable guidance. It was the greatest of honors to work under you. I will forever remain grateful.

My appreciation to all colleagues at the IZUNAR LAB you made my stay and the entire dissertation period memorable.

Special thanks to my family and friends for their uncondition support and encouragement. I also thank TÜBİTAK BİDEB for its support. The programme would have been a lot harder without it.

Alex GUNAGWERA

ABSTRACT

AUTONOMOUS VEHICLE PLATOON MODELING AND CONTROL USING PID AND LINEAR QUADRATIC REGULATOR

Alex GUNAGWERA
Ph.D. Dissertation, Computer Engineering
Supervisor: Asst. Prof. Dr. Aydın Tarık ZENGİN
July - 2022, 74 + XIV pages

With over 50 million cars being produced annually since 2010, the number of vehicles on the roads increases throughout the world. With the increase in vehicle population comes challenges and problems in efficient and safe transportation via roads. Major problems include road accidents, global warming due to emissions, efficient energy utilization and road usage to mention but afew. The autonomous vehicle platooning concept provides both a promising solution and an enhancement approach to intelligent transport systems. Among the benefits this promising concept presents are reduced fuel consumption and, hence emissions, efficient road usage among others. The autonomous vehicle platooning industry, however, still faces a great deal of issues and challenges ranging from string stability, safety guaranties, efficient communication to accurate control. In this study, autonomous vehicle platooning modeling and control using PID and LQR are presented. The algorithms are verified using two scenarios on 3D vehicles simulated used Gazebo and ROS. In the first scenario the platoon leading vehicle mainly accelerated, travelled with constant speed, and decelerated to rest. In the second scenario, however, the leading vehicle travelled with uncertain and constantly varying velocities. Results from both scenarios are then presented and summarized in graph and tabular forms. In both scenarios, platoon stability, especially after the initial transient response was achieved, the following effect for the platoon members in the performed simulations is also guaranteed and a steady state error of *0m* was obtained by both control strategies.

Keywords: Autonomous Vehicles, Autonomous Vehicle Platoons, Control Theory, Intelligent Transport Systems, LQR, PID

ÖZET

LINEAR QUADRATIC REGULATOR VE PID CONTROLLER İLE OTONOM ARAÇ MÜFREZESİNİN MODELLENMESİ VE KONTROLÜ

Alex GUNAGWERA
Doktora Tezi, Bilgisayar Mühendislik
Danışman: Dr. Ögr. Üyesi Aydın Tarık ZENGİN
Temmuz - 2022, 74 + XIV sayfa

2010 yılından bu yana yılda 50 milyonun üzerinde otomobil üretiliyor ve dünya genelinde yollardaki araç sayısı artıyor. Araç popülasyonundaki artışla birlikte karayolu ile verimli ve güvenli ulaşımda zorluklar ve sorunlar ortaya çıkmaktadır. Başlıca sorunlar arasında trafik kazaları, emisyonlardan kaynaklanan küresel ısınma, verimli enerji kullanımı ve yol kullanımı sayılabilir. Otonom araç müfreze konsepti, akıllı ulaşım sistemlerine hem umut verici bir çözüm hem de iyileştirme yaklaşımı sağlar. Bu umut verici konseptin sunduğu faydalar arasında daha az yakıt tüketimi ve dolayısıyla emisyonlar ve diğerlerinin yanı sıra verimli yol kullanımı yer alıyor. Bununla birlikte, otonom araç müfreze endüstrisi, dizilim kararlılığı, güvenlik garantileri, verimli iletişimden doğru kontrole kadar birçok sorun ve zorlukla karşı karşıyadır. Bu çalışmada, PID ve LQR kullanılarak otonom araç müfreze modellemesi ve kontrolü sunuldu. Algoritmalar, iki senaryo kullanılarak Gazebo ve ROS 3B ortamlarında simüle edilerek doğrulandı. İlk senaryoda, müfrezeye liderlik eden araç hızlandı, sabit hızla gitti ve durmak için yavaşladı. İkinci senaryoda, lider araç belirsiz ve sürekli değişen hızlarla yol aldı. Her iki senaryodan elde edilen sonuçlar daha sonra grafik ve tablo formlarında sunuldu ve özetlendi. Her iki senaryoda da, ilk geçici tepki elde edildikten sonra müfreze kararlılığı elde edildi ve takip de garanti edildi. Her iki kontrol stratejisi ile de 0m'lik bir kalıcı hal hatası elde edildi.

Anahtar kelimeler: Akıllı Ulaşım Sistemleri, Kontrol Teorisi, LQR Otonom Araçlar, Otonom Araç Takımları, PID

TABLE OF CONTENTS

APPROVAL PAGE ..... **i**

DECLARATION OF SCIENTIFIC ETHICS AND ORIGINALITY ..... **ii**

PREFACE ..... **iii**

ABSTRACT ..... **iv**

ÖZET ..... **v**

TABLE OF CONTENTS ..... **vi**

LIST OF TABLES ..... **ix**

LIST OF FIGURES ..... **x**

ABBREVIATIONS ..... **xii**

LIST OF SYMBOLS ..... **xiv**

CHAPTER 1
INTRODUCTION **1**

1.1. Autonomous Driving 1

1.2. Vehicle Automation 2

1.3. Platoon Driving 3

1.4. Autonomous Platoon Driving 4

1.5. Motivation 4

CHAPTER 2
LITERATURE REVIEW **7**

CHAPTER 3
BACKGROUND **12**

3.1. ROS 12

3.1.1. What is ROS 12

3.1.2. Communication in ROS 13

3.1.3. Why ROS 13

3.2. Vehicle Modeling 14

3.2.1. Vehicle model categories 15

3.2.2. Simple Vehicle Kinematic Model 17

3.3. Vehicle dynamics 18

CHAPTER 4
METHODOLOGY.. 21

4.1. Platoon Information Flow Models 21

4.2. General Platoon Model and Problem Statement 22

4.3. Simulation Environment and Experimental Setup 27

4.4. PID Controller 34

4.4.1. Brief Introduction 34

4.4.2. PID Formulation 34

4.5. LQR 35

4.5.1. Brief Introduction 35

4.5.2. LQR Derivation and Formulation 36

CHAPTER 5
RESULTS 41

5.1. Scenario 1 41

5.1.1. Results obtained using PID controller 43

5.1.2. Results obtained using LQR controller 45

5.2. Scenario 2 (Uncertain Scenario) 47

5.2.1. PID controller performance 47

5.2.2. LQR controller performance 49

CHAPTER 6
DISCUSSION: COMPARISONS, CONTRIBUTIONS, AND LIMITATIONS 51

6.1. Comparisons 51

6.2. Contribution 53

6.3. Limitations 54

CHAPTER 7
CONCLUSION.. 56

CHAPTER 8
FUTURE WORK 58

REFERENCES 59

CURRICULUM VITAE 71

LIST OF TABLES

Table 3.1: Simple motion description summary 16

Table 3.2: Description of the symbols presented in the vehicle longitudinal model . 20

Table 4.1: Vehicle information 28

Table 5.1: PID Controller parameters 41

Table 5.2: Description of the distinct zones throughout the simulations.................. 42

Table 5.3: Error statistics of the three inter-vehicle distances while using PID 44

Table 5.4: Error statistics of the three iner-vehicle distances while using LQR........ 46

Table 6.1: General comparison of results obtained from the first scenario 51

Table 6.2: General comparison of the results obtained from the second scenario..... 51

Table 6.3: Comparison of results with other methodologies suggested in existing literature.. 52

LIST OF FIGURES

Figure 1.1: Platooon Driving - Lateral and longitudinal control 3

Figure 1.2: Merging and Splitting in AVPS 4

Figure 2.1: Waymo’s Self-driving Vehicle Samples Lulu Chang (2018).................... 9

Figure 2.2: SARTRE Demo SARTRE (2019) 10

Figure 2.3: Platoon Driving; Individuals in Follower(F) Vehicles Can Engage in Other Activities 11

Figure 3.1: Communication in ROS.. 13

Figure 3.2: Roll, pitch, and yaw demonstration 16

Figure 3.3: Car kinematics using the bicycle model approach 17

Figure 3.4: Forces acting on the vehicle on a longitudinally inclined plane.............. 19

Figure 4.1: Common Information follow models in vehicle platoons. 22

Figure 4.2: PID platoon controller model. 24

Figure 4.3: LQR platoon controller model... 24

Figure 4.4: Illustration of the inter-vehicle distance d_i, d_{il}, *LV*, and *F*. 25

Figure 4.5: Simulation environment 27

Figure 4.6: Internode relationships 30

Figure 4.7: General closed loop PID control 34

Figure 5.1: IVD while using the PID controller... 43

Figure 5.2: Error in the IVD while using the PID controller 43

Figure 5.3: Platoon velocity profile while using the PID controller 44

Figure 5.4:IVD while using the LQR algorithm 45

Figure 5.5: Error in the Error while using the LQR controller 45

Figure 5.6: Platoon velocity profile while using the LQR algorithm 46

Figure 5.7: Platoon performace when the LV velocity is continuously varied - uncertain scenario while using the PID controller 47

Figure 5.8: Platoon performace when the LV velocity is continuously varied - uncertain scenario while using the LQR 49

ABBREVIATIONS

Abbreviation	Definition
3D	3 Dimensional
ACC	Adaptive Cruise Control
ADS	Automation Driving Systems
ANN	Artificial Neural Networks
APF	Artificial Intelligence Potential Field
API	Application Programming Interface
AVP(s)	Autonomous Vehicle platoon(s)
AVPS	Autonomous Vehicle platoon Systems
AVs	Autonomated/Autonomous Vehicles
CACC	Cooperative Adaptive Cruise Control
COG	Center of Gravity
CSMA/CD	carrier Sense Multiple Access and collision detection
C-V2X	Cellular Vehicle to Everything
DDTs	Dynamic Driving Tasks
DSRC	Dedicated Short Range Communications
F	Follower Vehicle
FSM	Finite State Machines
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HDV	Heavy-Duty Vehicle
HMD	Head Mount Display
ICR	Instantaneous Center of Rotation
IMU	Inertial Measurement Units
ITS	Intelligent Transport Systems
LIDAR	Light Detection and Ranging

LQR	Linear Quadratic Regulator
LV	Lead Vehicle
MaaS	Mobility as a Service
MAS	MultiAgent System
MPC	Model Predictive Control
NSB	Null-Space-Based
NSC	National Safety Council
OS	Operating System
PAVE	Partners for Automated Vehicle Education
PID	Proportional Integral Derivative
QoS	Quality of Service
RADAR	Radio Detection And Ranging
RF	Radio Frequency
ROS	Robot Operating System
SAE	Society of Automotive Engineers
SLAM	Simultaneous Localization And Mapping
TF	Transform Library
URDF	Universal Robot Description Files/Formats
USDOT	United States Department of Transportation
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
V2X	Vehicle to Everything
VANET	Vehicle Ad hoc Network
XACRO	XML Macro
XML	Extensible Markup Language
YAML	Yet Another Markup Language

LIST OF SYMBOLS

Symbol	Description (Unless stated, otherwise)	Unit
m	Mass	kg
π	Pi	
g	Acceleration due to gravity	ms^{-2}
r	Radius	m
L	Length	m
F	Force	N
t	Time	s
v	Velocity	ms^{-1}
e	Error	m
σ	Steering angle	Radians
θ	Vehicle heading	Radians
R	Radius of rotation	m
$\mathbb{R}$	Set of real numbers	

CHAPTER 1

INTRODUCTION

The transport industry, in particular, and the public, in general are facing problems and issues all over the world today. These issues range from traffic congestion, road accidents, pollution due to emissions, efficient road usage to efficient energy utilization. The impact and magnitude of these issues becomes more pronounced with the increase in the number of vehicles on the roads.

The concept of vehicle platooning (Kavathekar and Chen 2011; Kimura et al. 2017; Kundu et al. 2013; Marino et al. 2009; Tong et al. 2019) presents promising approaches towards not only mitigating the issues facing the transport industry all over the world today, but also enhancing the quality of the transportation experience in general.

With the rapidly growing vehicle population all over the world, that is, over fifty million cars being manufactured every year, and approximately 1.4 billion vehicles actively being used on the roads presently, (Worldometer 2019), solutions to the issues facing the transport industry require immediate attention. As such, the promising solution of vehicle platooning has attracted the attention of governmental bodies, automobile industries, engineers, and academic researchers all over the world. In this chapter, major terms forming the foundation of the autonomous vehicle platooning concept are presented followed by the motivation of this study.

1.1. Autonomous Driving

The terms automated vehicle (Broggi et al. 2013; Chang and Yuan 2018; Z. Huang et al. 2019; Kolb, Nitzsche, and Wagner 2019), autonomous (Bartz 2009; Freddi, Longhi, and Monteriù 2015; Hallé, Laumonier, and Chaib-draa 2004; Szalay et al. 2018) and self-driving (Chakraborty, Yamaguchi, and Datta 2018; Guizzo 2011; Ross 2019; Ruane 2018; Stewart 2018a) refer to packages of technology made up of software and hardware that is capable of, with or without human intervention, executing the Dynamic Driving Tasks (DDTs) as defined by the Society of Automotive Engineers. SAE (international 2016; Jurgen 2011). These terms are sometimes inter-changeably used in literature to mean the same thing. Driverless (McArdle 2018; Sharon L Poczter and Jankovic 2014; Szalay et al. 2018) is also used towards the same end.

In order to avoid misinterpretations, The Society of Automotive Engineers published *j3016*, (SAE 2013) and designated the term automated as the standard and least misleading word for the general control and management of the entire DDTs. DDTs range from tactical such as response to signals, lane maneuvers such as lane joining, or lane change, for or operational, such as speed control via acceleration, braking to mention but a few. As such the overall definition of automated driving should not be confused with collision avoidance systems whose major focus is different (Vahidi and Eskandarian 2003).

1.2. Vehicle Automation

SAE categorizes all Automation Driving Systems into six major levels from level zero through level five depending on how much they automate and execute the DDTs. Level zero incorporates no automation at all, whereas Level five corresponds to full automation of the DDTs. The details of what each automation level entails are stipulated in (international 2016). A summarized graphical version of can also be referred to from SAE's website, (Shuttleworth 2019). In a nutcase, at level zero, the human driver is responsible for all the DDTs, along with all the Object and Event Detection and Response (OEDR).

At level one, either the lateral or longitudinal motion control of the vehicle is automated but not both. The human driver is responsible for all other tasks and subtasks whereas at level two, also known as partial automation, the automation system, while under the monitoring of a human driver is capable of handling both the longitudinal and lateral motion of the vehicle. Level three automation only requires that the human driver responds to any fallback requests, system messages, and notifications as the automation system handles all the other DDTs. Level four automation systems execute the entire DDT and DDT fallback within the specified limits, and it is limit/domain specific. Nothing is particularly expected from the human driver. Automation of this level and beyond is categorized as high-level automation.

Level five, also known as, full automation is the ultimate level of automation. The ADS is capable of handling all DDTs, fallbacks, notifications and the like under all climate and road conditions.

1.3. Platoon Driving

In the concept of platoon driving, a group of one or more vehicles are connected and move as a single object. The connected group of vehicles is generally subjected to the same lateral and, or longitudinal motion control. The concept of lateral and longitudinal motion, as applied to vehicle platoons, is demonstrated in Figure 1.1.

Connection of the platoon members is generally achieved through one or a combination of the following ways: via electric connections especially using Vehicle to Vehicle (V2V) connections as (Choi et al. 2009; Ibrahim et al. 2018; Karnadi, Mo, and Lan 2007; Li and He 2018), via the use of sensors such as LIDAR and radar based connection systems, (Guizzo 2011; Levanon 1988; Levinson et al. 2011; Reutebuch, Andersen, and Mcgaughey 2005). Every platoon has a lead vehicle (*LV*) which is followed by the other follower (*F*) vehicles. The *LV* is normally indexed as the first member of the platoon with or numerically as the zeroth (0^{th}) platoon member. The vehicle immediately after the *LV* is indexed as the second platoon member or 1^{st} platoon member and the notion goes on till the last *F* vehicle indexed as the n^{th} vehicle in a platoon with a total of $n+1$ members – *LV* inclusive. The *LV* has the most information about the route to be traversed and the overall path planning. Other platoon members are generally only concerned with efficiently following the *LV*. They need

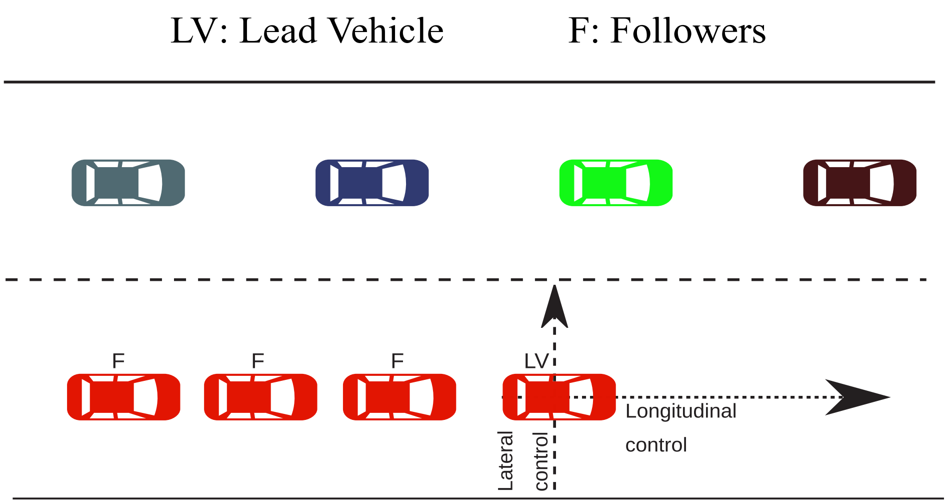

Figure 1.1: Platooon Driving - Lateral and longitudinal control

not know the route or map or any other such information.

1.4. Autonomous Platoon Driving

A collection of one or more Autonomous Vehicles(AV) in a platoon culminates in the so-called autonomous platoon. This implies members in autonomous vehicle platoon (AVP) need to at least have level two ADS incorporated as described above in order to be able to smoothly perform the autonomous vehicle platoon maneuvers. Major platoon maneuvers are merge and split. Other platoon maneuvers may include overtake and overall navigation. Efficient and proper execution of these tasks greatly affects the performance and general success of a platoon. At the moment, however, smooth execution of these tasks is not easy given the fact that platoon is majorly mixed. It may include vehicles manned by humans, some partially automated and so on. So human errors, mechanical and technical failures need to be accounted for as well.

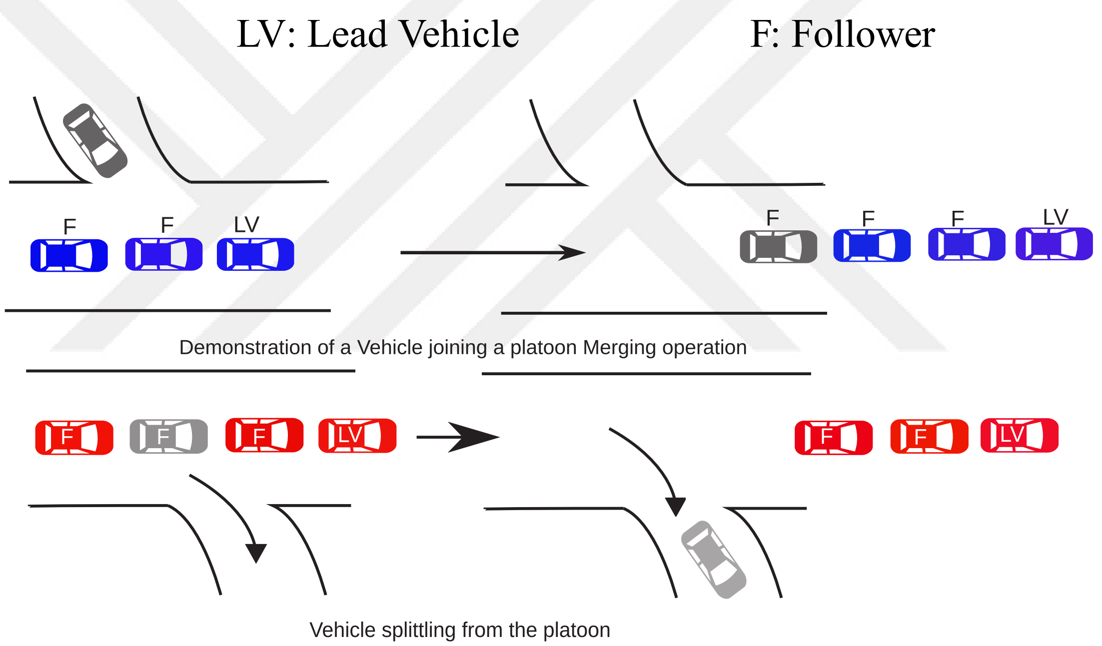

Figure 1.2: Merging and Splitting in AVPS

Figure 1.2: provides an illustration of the merging and splitting maneuvers of AVPS.

1.5. Motivation

The numerous benefits presented by AVPS coupled with its cross-disciplinary reach have made the field of AVs in general, and AVPS in particular a trending topic all over the world. AVPS are also applicable to variety of applications from different fields thereby expanding its outreach even further. Such fields include the military, underwater research, academics and so much more. The benefits it promises range from societal to business oriented. The societal benefits include the reduction of emissions culminating from the fact that AVP member are able to travel at relatively shorter distances compared to those when all vehicles are controlled by humans – without error or risk of accidents assuming all factors are kept constant.

Vehicle platoons also maximize the efficiency of road utilization. As an example, two trucks travelling in a platoon with a headway spacing of 0.3 seconds or equivalently travelling at a speed of 60mph reduced the road space occupied by the two trucks to forty four meters from eighty two meters (that is by about 46%) as calculated by (Janssen et al. 2015).

AVPS save time and also improve people's quality of life. For people like such as the color blind, physically incapacitated and also the elderly, driving becomes no longer an issue. According to (United Nations D. o. E. a. SA 2017), workers' bulk is also lightened especially those dealing in transportation. Time that would be spent while driving can be more effectively used on other tasks such as resting, reading news and so on, as depicted in Figure 2.3.

AVPS increases road safety. (Belcher et al. 2018) reported that the United States Department of Transportation observed that more than 90% of the traffic accidents resulted from human error, furthermore, that eliminating the human-factor from the conditions would minimize accidents by approximately 80%.

From the business point of view, AVPS benefits include minimization of the fuel consumed by platoon members, (Brandt et al. 2010). Reduction in expenses made on labor is another advantage especially in Automated Highway Systems (AHS) which require drivers to drive for long hours.

Despite the variety of merits facilitated by AVPS, the tremendous research carried out, a multitude issues still face and bar the implementation and deployment of AVPS. Such issues include security risks, safety such as; threat of hackers hacking into AVPS since they mainly depend on Wi-Fi for communication, (Blum and Eskandarian 2004; Marques, Casimiro, and Calha 2009). Another issue is the anxiety, distrust, and insecurity of passengers when it comes to AVs. For instance news of accidents caused by or involving AVs (CENTER 2017; Claybrook and Kildare 2018; Miller 2015; Stewart 2018b).

AVPS are also expensive to build, maintain and control. The extra requirements of AVs are expensive thereby deterring potential investments. Such requirements range from hardware equipment such as sensors to software that will safely control the AVs.

With this background, yet another relatively cost-efficient approach of modeling and controlling AVPS is presented in this study. The proposed approach retrieves data obtained from sensors and passes the information within to either of PID or LQR controller with the main aim being maintaining a desired inter-vehicle gap between platoon members, ensuring user safety by making sure no collisions among platoon members occur. Other important aspects of AVPS such as string stability and user comfort are also taken into account.

CHAPTER 2

LITERATURE REVIEW

Owing to the significance its potential, the concept of AVPS has attracted a lot of attention from various fields. As such, tremendous research has been conducted related to the field. In this chapter, a review, though not exhaustive, of the most prominent research, projects, and work pertaining to AVPS is conducted.

A multitude of studies focusing on the control of autonomous vehicle platoons was carried out. The research focusing on this area presents methods aimed at the smooth running and operation of autonomous vehicle platoons under various circumstances. The proposed approaches include; Cooperative Adaptive Cruise control (Bayuwindra et al. 2020; Deng 2016; Han et al. 2013; Naus et al. 2010; Ucar, Ergen, and Ozkasap 2017; Wang et al. 2018), Adaptive Cruise Control (ACC), (Liu et al. 2008; Richtel and Dougherty 2015; Sivaji and Sailaja 2013), Model predictive controllers, (Z. Huang et al. 2019; Ibrahim et al. 2018; V. Jain et al. 2018) among others. ACC is an enhance form of vehicle speed control where the corresponding vehicle observes a given inter-vehicle distance between itself and its predecessor by accelerating and decelerating appropriately. CACC incorporates communication on ACC systems thereby provided better performance,(Anayor, Gao, and Odekunle 2018; Naus et al. 2010; Shladover et al. 2015).

(Z. Huang et al. 2019) joined the MPC and Artificial Intelligence (AI) concepts to provide a solution to simultaneous path planning and motion control. They used simulations to verify their simulations. (J. Zhao, Ecole, and Lille 2011) present an alternative aimed at the reduction of overall carbon emissions and improve safety while travelling at high speeds. (Mena-Oreja, Gozalvez, and Sepulcre 2019), examine relationship between mixed traffic, maximum platoon length, and safe gaps. The Augmented Reality (AR) technic is employed to ease life of welfare workers and the elderly in narrow environments such corridors by (Kimura et al. 2017).

Energy efficiency, (Deng 2016; Van De Hoef, Johansson, and Dimarogonas 2018; Jia et al. 2016; Nemeth et al. 2012; Wu, Wu, and Wang 2019) is one of the backbones of good platoon performance in AVPS. A linear programming approach to find the optimal number of platoons, energy required by the *LV*, and duration required for gathering members was presented by (Hadded et al. 2018).

The problem of fuel-efficient coordination for large platoons comprising of trucks considering the destination of the platoon members, deadlines of their arrivals to the destinations, departure times and so forth was investigated by (Van De Hoef, Johansson, and Dimarogonas 2018).

Optimally energy efficient platoon maneuver execution was explored in (Blum and Eskandarian 2004). There work proposed a shortest-path-based algorithm to realize energy efficient execution of the merging and splitting maneuvers of platoons.

Another problem sensitive yet crucial aspect to AVPS, in particular and AVs in general is communication(Choi et al. 2009; Hu et al. 2020; R. H. Huang et al. 2018; Ioannou and Xu 1994; Mei et al. 2018; Su and Ahn 2016a; Zou and Li 2019). Efficient, robust, and reliable communication has the capability to exponentially increase platoon performance. On the contrary, poor communication does not only reduce platoon performance, but could also affect platoon safety. It is thus, no surprise that a plethora of studies were performed, aimed at improving platoon communication (among platoon members, or with surroundings). The major protocol used for platoon communication is the IEEE 802.11(M. Jain and Saxena 2017; Khaksari and Fischione 2012), and its variations with the most prominent ones including: at 5GHz, the IEEE 802.11a, the IEEE 802b at 2.4GHz, and a relatively newer protocol designed for the Vehicle Ad hoc Network (VANET), (Kaur et al. 2018; Singh and Agrawal 2014; Su and Ahn 2016b; Tonguz et al. 2007), the IEEE 802.11p.

Figure 2.1: Waymo’s Self-driving Vehicle Samples Lulu Chang (2018)

Since Wi-Fi based communication systems are susceptible to attacks. Concerns about security pose another issue affecting AVPS. Security issues and various attack types likely to face VANET were meticulously investigated by (Chaurasia, Verma, and Tomar 2011).

The USDOT also required that vehicles to be sold with effect from 2023 be equipped with V2V based DSRC (Tsugawa et al. 2001; Ucar, Ergen, and Ozkasap 2017) systems thereby providing another way of standardizing and enhancing communication in AVPS. China also decided on using Cellular-Vehicle to Everything (C-V2X), (Qi and MacH 2019) and DSRC, with their plan have deployed C-V2X by 2020 in the majority of their vehicles, (Estopace 2019).

A V2V, (Choi et al. 2009; Zhang et al. 2018), based vehicle positioning approach for systems that do not require vision sensors was proposed by (Shen et al. 2018). (Marino et al. 2009) presented a Null-Space Based approach to control multi robot behavior as a solution for communication-based robot patrol.

Positioning and orientation are of paramount importance to the successful routing and navigation of an AVPS from a given source to a desired destination. Many approaches and tools have been designed and proposed to achieve this goal. The most prominent among these tools are the Global Positioning Satellite (GPS), (Chakraborty, Yamaguchi, and Datta 2018; Omar et al. 2016) and the Global Navigation Satellite System (|GNSS), (Chakraborty, Yamaguchi, and Datta 2018; Xiao Chen et al. 2018; mann-Wellenhof, Bernhard and Lichtenegger, Herbert and Wasle 2007)(Chakraborty, Yamaguchi, and Datta 2018). Especially in cases where the *LV* is an AV as well, centralized, and decentralized approaches for achieving positioning and navigation of the platoon were investigated

Figure 2.2: SARTRE Demo SARTRE (2019)

(Freddi, Longhi, and Monteriù 2015; Hallé, Laumonier, and Chaib-draa 2004; Stilwell and Bishop 2001; Zaher, Madeleine El and Gechter, Franck and Hajjar, Mohammad and Gruer 2016).

Objected detection (Xiaozhi Chen et al. 2015; Redmon et al. 2016; Tseng and Jan 2018) through sensors followed by positioning and routing approaches were also suggested in literature (LeCun et al. 2005; McAllister et al. 2017). Final motion generating steps can be implemented by means such as application of neuro evolution algorithms (Fortin et al. 2012; A. K. Jain, Mao, and Mohiuddin 1996; Koutník et al. 2013), supervised learning as done by (Bojarski et al. 2016), reinforcement learning (El Sallab et al. 2017) among others. Major data and information acquisition methods majorly include sensors such as cameras (Xiao Chen et al. 2018; Gallego et al. 2019; Gaspar, Winters, and Santos-Victor 2000), LIDAR (Reutebuch, Andersen, and McGaughey 2005), and RADAR (Levanon 1988; Sivaji and Sailaja 2013; Ziegler et al. 2014).

How autonomous vehicles interact with humans (passengers, pedestrians, and drivers) is a topic worth investigating given the fact that global fully autonomous transportation is not yet in effect. AVPS, therefore, are inevitably required to interact with other humans and sometimes accidents may occur (Boudette 2016; Miller 2015).

Indoor platoon systems operable in narrow environments as well, and those that lighten the workload of employees in welfares were proposed, (Kimura et al. 2017; Sugano et al. 2016; Sugano, Okajima, and Matsunaga 2015).

To date, projects, field tests, and demonstrations were carried out in order to prove that AVPS are an achievable goal, not just a dream. Furthermore, overall, general progress in the field of AVs and AVPS has been realized. Prominent among such projects include; the AV car demonstration organized by SAE (Visnic 2018), Nvidia's project (Shapiro 2017), and Tesla's tests (Kessler 2015). Culminating from the google self-driving car, the Waymo project (Team-Waymo 2019). Figure 2.1: shows sample vehicles developed under the Waymo project. Another impressively work was SARTRE, whose demonstration is shown in Figure 2.2, (SARTRE Consortium 2012), the Hyundai's project (Hyundai 2019), ARGO AI (Team 2019) both of which aim at producing AV products and ultimately providing Mobility as a Service (MaaS), which may prove more fruitful for all engaging parties (Global 2019).

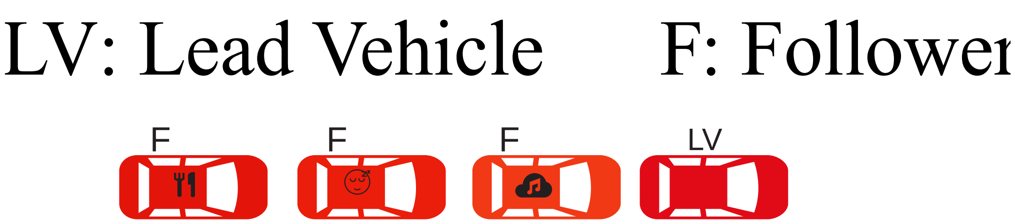

Figure 2.3: Platoon Driving; Individuals in Follower(F) Vehicles Can Engage in Other Activities

Figure 2.3**:** provides a visual demonstration of how time that would be spent on driving can alternatively be used for other productive work in AVPS.

CHAPTER 3

BACKGROUND

In this chapter, general background of the major tools used in this study are presented. Basic underlying concepts applied are also presented and explained in this chapter.

3.1. ROS

In this section, the Robot Operating System introduction is done. A general overview of ROS, its environment, reasons why it is preferred, and working structure are also explained.

3.1.1. What is ROS

The Robot Operating System is an open-source framework for robot software development. The main goal of ROS to provide a relatively simple and standard way of achieving common robotic tasks such as navigation, path planning (Kolb, Nitzsche, and Wagner 2019), control and monitoring of low-level devices such as motors and acutators(Chitta, Marder-Eppstein, Meeussen, Pradeep, Tsouroukdissian, et al. 2017), simulations (Koenig and Howard 2004) etc. Currently, there are two major versions of ROS; ROS1 and ROS2. In this study, ROS1 was used and is the one being referred to henceforth unles stated otherwise. ROS provides Operating System (OS)-like functionality for robots. It should be noted that ROS is ***NOT*** an OS according to the traditional definition of an operating system i.e., with major goals of scheduling and managing processes. ROS runs on top of an operating system (mostly Linux/Ubuntu) but all devices with a woring network connection are capable of running software that can communicate with ROS nodes. This feature fascilitates communication of Personal Computers (PCs) and embedded devices with, possibly, different architecture (heterogenous) over a structured layer, (Quigley 2009). Data processing and calculations are performed within nodes. Nodes make up the basic blocks of a ROS environment. On top of processing data, nodes can also publish, receive, or do more data manipulations.

3.1.2. Communication in ROS

Communication in ROS is fascilitated through messages. These messages are passed over topics. Nodes in ROS communicate over these topics. The ROS Master, initialized at the start-up of the ROS server, manages communication among all the nodes. All nodes register with the master at startup. Nodes in ROS are able to subscribe or publish to topics. Generally, there's one publisher and there maybe one or more subscribers to a given topic. The topic serves as a name for a given stream of messages.

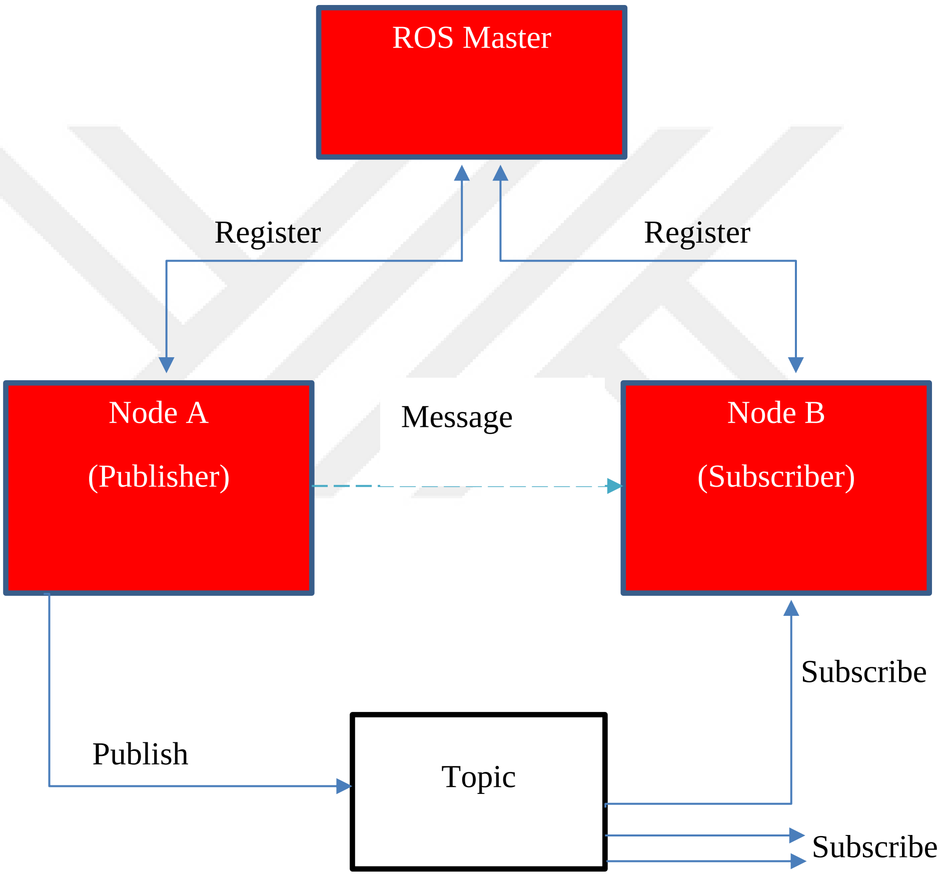

Figure 3.1: Communication in ROS

3.1.3. Why ROS

Continously writing and implementing logging systems, basic navigation logic, communication protocols among devices and procedures, visualization, and debugging systems and so on and so forth, gets tedious over time. In addition to providing all the above functions readily implemented, ROS readily provides the following:

- Hardware Abstraction
- Code Reuse
- Low-level device control
- Message-passing between processes
- ROS is modular with fairly easy package management. This enables users to easily add new and more specific functionality to already existing packages and modules as deemed necessary. ROS applications typically comprise of various nodes each performing a specific, and mostly, single task e.g., speed control, data aquisition, data multiplexing.
- A variety of tools and client API libraries in form of packages owing to its large community, continous support and being open source. Parties benefit from packages written by one another thereby eliminating continous re-inventing of the wheel.
- ROS supports various architecture i.e. simultaneous development across multiple hardware architectures using a multitude of supported programming languages is a great advantage in ROS. E.g., Nodes of the same application could be written on different computers, microprocessors such as arduino, android smartphones etc. using different programming languages such as Python, C++, MATLAB to mention but a few.

The details of the major ROS tools, packages and other components utilized in this study will be presented in the subsequent corresponding sections.

3.2. Vehicle Modeling

Vehicle modeling is no straight forward feat. It comprises of nonlinear vehicle components such as tyres along with a multitude of parameters ranging from vehicle mass, tyre-road friction coefficient to mention but a few.

Vehicle dynamics modeling is fundamentally centered around forces generated by the interaction between the tyres and the road surface in conjunction with the various mechanical components of the vehicle.

Many tyre modeling approaches have been suggested in literature. These include; (Pacejka, 2006)'tyre vehicle behavior, tyre modeling using finite element method presented by (Gim, 1990, 1991a, 1991b).

Vehicle Dynamics modeling: To accurately model vehicle dynamics, the choice of the list of mechanical parts to be put under consideration is of great importance i.e., steering angle, anti-roll bar, suspension geometry, tyres, etc. Complexity and accuracy of the model are directly dependent on the list of such chosen vehicle components.

Normally a tradeoff/compromise between accuracy and complexity is inevitable. However, the major deciding factor between complexity and accuracy of the model is the ultimate goal to be achieved by the model implementation. (Day 1995, Pham 1997, Hingwe 1997, Nouveliere 2002).

Some of the most complex vehicle models presented in literature include (Lowndes, 1998) with 28 DoF, (Addi, 2005) presents a rigid body 18 DoF dynamics model. To date, 6 DoF models are considered sufficient and are generally accepted almost everywhere. This is because they capture the 6 principal movements including.

The three translations along the *X*, *Y*, and *Z* axes and the three rotational motions about the 3 axes i.e., roll, pitch, and yaw motions. Figure 3.2 demonstrates these rotational and translational motions in 3D space.

3.2.1. Vehicle model categories

Vehicle models can be categorized into three major subcategories: longitudinal, lateral and coupled both longitudinal and lateral vehicle models. Generally, the longitudinal acceleration, lateral acceleration, and yaw rate can be derived by applying Newton's second law of motion and putting into account the forces acting on the front and rear wheels combined with the torques on the rear wheels.

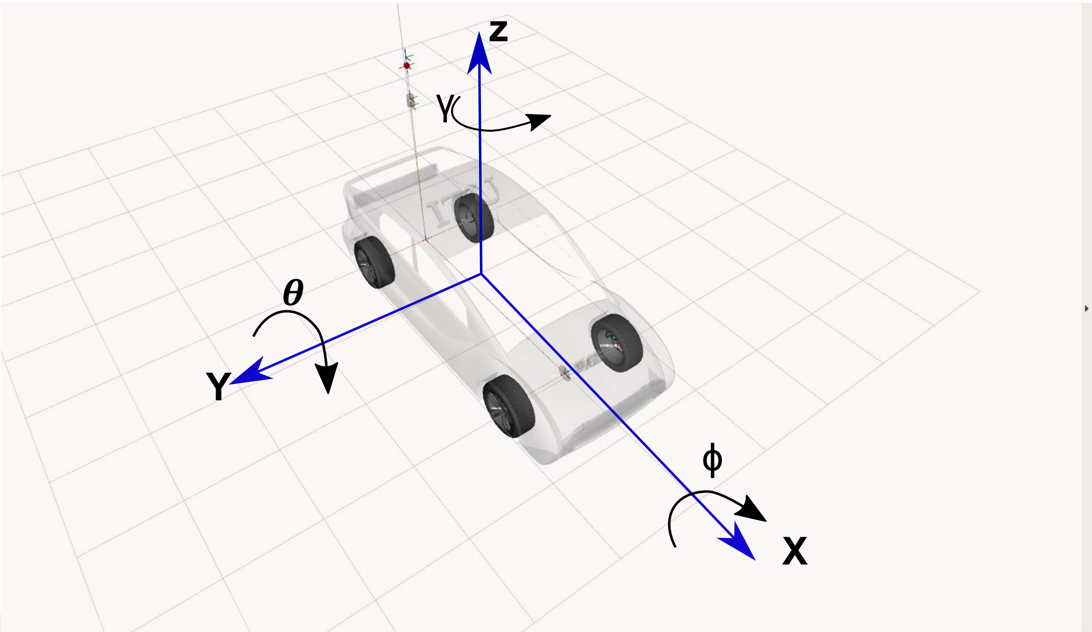

Figure 3.2: Roll, pitch, and yaw demonstration

Figure 3.2 illustrates the roll, pitch, and yaw motions of a vehicle in 3D space.

Table 3.1 below summarizes the description of possible motions of a vehicle in a 3D plane using the car coordinate space presented in Figure 3.2.

Table 3.1: Simple motion description summary

Translation	**Direction**
Longitudinal	along x-axis
Lateral	along the y-axis
Vertical	along the z-axis
Rotations	**Direction**
Roll angle (ϕ)	rotation about the z-axis
Pitch angle(θ)	Rotation about x-axis
Yaw angle(γ):	rotation about the y-axis

3.2.2. Simple Vehicle Kinematic Model

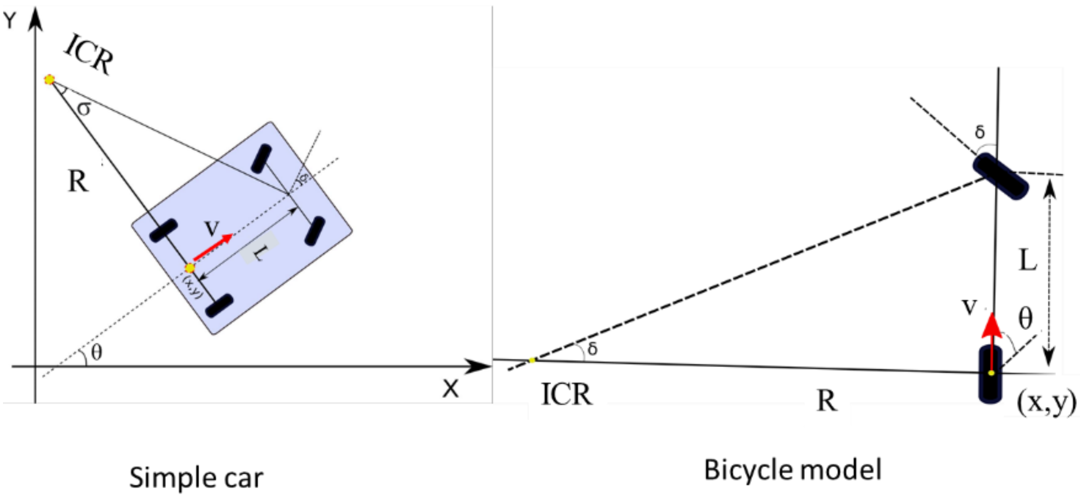

Figure 3.3: Car kinematics using the bicycle model approach

Figure 3.3 shows the bicycle model used to derive the car kinematics. A simple car has three degrees of freedom, but the velocity space at any configuration is only two-dimensional. The general car kinematics can, thus be expressed as shown in equations (3.1) - (3.3) about the rear wheel axle.

$$\dot{x} = v\cos(\theta) \tag{3.1}$$

$$\dot{y} = v\sin(\theta) \tag{3.2}$$

$$\dot{\theta} = \frac{v}{L}\tan(u_{\Phi}) \tag{3.3}$$

Where: v is the velocity of the vehicle, σ is the steering angle, R is the radius of rotation, and $u_{\Phi} \in [-\sigma_{max}, \sigma_{min}], \sigma_{max} < \frac{\pi}{2}$ and *(x,y)* are the coordinates of the rear wheel on the bicycle model of the vehicle as shown in Figure 3.3 above. (LaValle 2006; Rajamani 2011). θ is the heading of the vehicle.

3.3. Vehicle dynamics

Consider a vehicle as one rigid body moving along an inclined road as shown in Figure 3.4. Major forces include gravitational, rolling resistance, and aerodynamic drag forces. The general dynamic equations of the vehicle can be obtained as follows:

From Newton's second law of motion:

$$\vec{F} = m\vec{a} \tag{3.4}$$

Applying it in the longitudinal and lateral directions, we get the following general equations.

$$ma_x = \sum F_x \tag{3.5}$$

$$ma_y = \sum F_y \tag{3.6}$$

$$J\dot{\zeta} = l_f F_f - \ l_\zeta F_\zeta \tag{3.7}$$

Where:

- m is the mass of the car
- a_x , a_y are the longitudinal and lateral accelerations respectively
- F_x , F_y are foces acting in the longitudinal and lateral orientations respectively
- J is the moment of inertia
- ζ is the yaw rate during turning
- l_f and l_r are the lengths from COG to front and rear axes
- F_f and F_r are the lateral forces on the front and rear axes

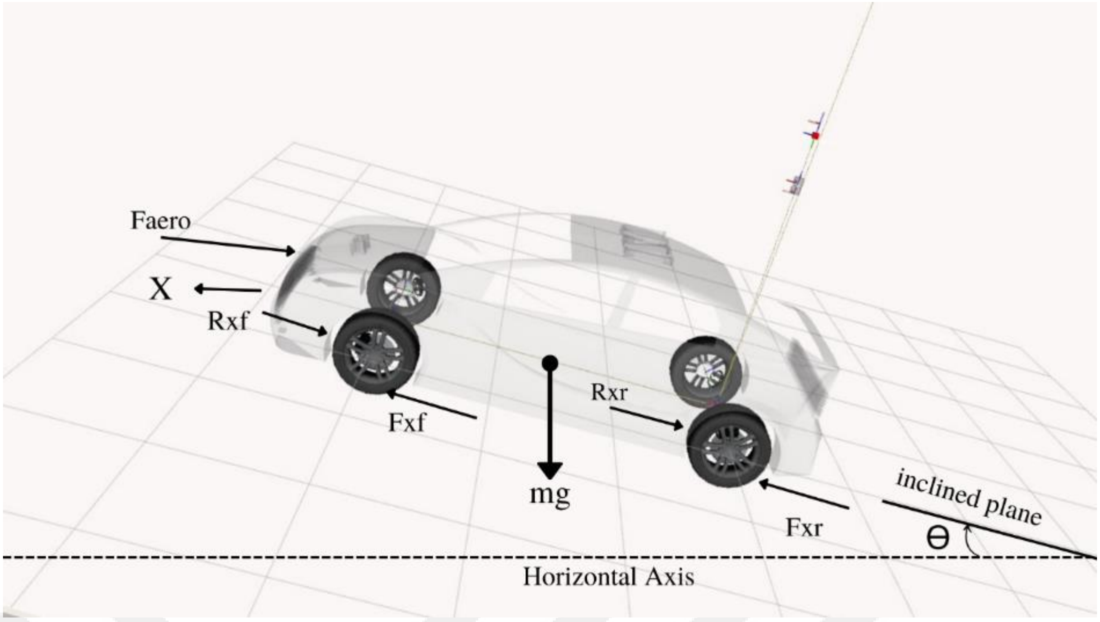

Figure 3.4: Forces acting on the vehicle on a longitudinally inclined plane

Balancing forces in the vehicle's longitudinal direction, which is the main focus of this study, we can write:

$$m\ddot{x} = F_{xf} + F_{xr} - R_{xf} - R_{xr} - mgsin(\theta) - F_{aero} \tag{3.8}$$

Since longitudinal control is the main focus of this study, the details of lateral control are skipped here i.e., equation (3.6), its derivations and details.

Figure 3.4 demonstrates the action of the major forces on the car on an inclined plane or road surface (Rajamani 2011). The meaning of the symbols used is presented in Table 3.2.

Table 3.2: Description of the symbols presented in the vehicle longitudinal model

Expression	Meaning
F_{xf}	Longitudinal force exerted on front tyres
F_{xr}	Longitudinal force exerted on rear tyres
F_{aero}	Total longitudinal force due to the aerodynamic drag
R_{xf}	Rolling resistance force exerted on the front tyres
R_{xr}	Rolling resistance force exerted on the rear tyres
m	Vehicle mass
g	Gravitational acceleration
θ	Road inclination angle

CHAPTER 4

METHODOLOGY

This chapter presents the methods implemented and applied in this study. A brief overview and formulation of each of the PID and LQR approaches is presented as well.

4.1. Platoon Information Flow Models

The transmission of information from one vehicle to another in a platoon is of paramount importance to the behavior of the platoon members, in particular, and the overall performance of the platoon, in general. The amount, the source(s) and accuracy of the information shared ultimately defines platoon behavior. The term platoon information flow topology is usually used to define the direction, source and receiver of the information within a platoon, (Han et al. 2013; Wang et al. 2018; Zheng et al. 2014).

With Vehicle to Vehicle (V2V) communication, (Kundu et al. 2013; Li and He 2018; Zhang et al. 2017; S. Zhao et al. 2015) in place, a number of information flow topologies/models have been developed. Information received, and/or transmitted is utilized in the speed, (Ali Memon, Jumani, and Larik 2012; Deng 2016; Hu et al. 2020) and/or inter-vehicle spacing (Hu et al. 2020; Willke, Tientrakool, and Maxemchuk 2009; Yu, Guo, and Lei 2018) control by platoon members.

A platoon generally comprises of a Leader Vehicle (*LV*), and a total of *n* follower (*F*) vehicles. In most cases, the *F* vehicles are equipped with decentralized controllers (Freddi, Longhi, and Monteriù 2015; Ghasemi, Kazemi, and Azadi 2013; Hallé, Laumonier, and Chaib-draa 2004) that handle the overall control of the corresponding platoon member. The *LV* is indexed and occasionally referred to as the 0^{th} vehicle, and, counting upstream, the last *F* vehicle becomes the $(n\text{-}1)^{th}$ vehicle.

There are numerous information flow models for platoon control. The behavior of individual platoon members is directly affected by the information model used in the platoon. Thus, platoon behavior is also, ultimately, influenced by the platoon model(s) implemented.

Figure 4.1 shows some of the most commonly used information flow models in AVPs. These models include:

1) The Predecessor-Following (PF) model, labelled as *i*.
2) The Predecessor-Leader-Following (PLF) model, *ii*.
3) The Bidirectional (BD) model, *iii*.
4) Bidirectional Leader (BDL) model, *iv*.
5) Two Predecessors Following (TPF) model, *v*.
6) Two Predecessors Leader Following (TPLF) model, *vi*.

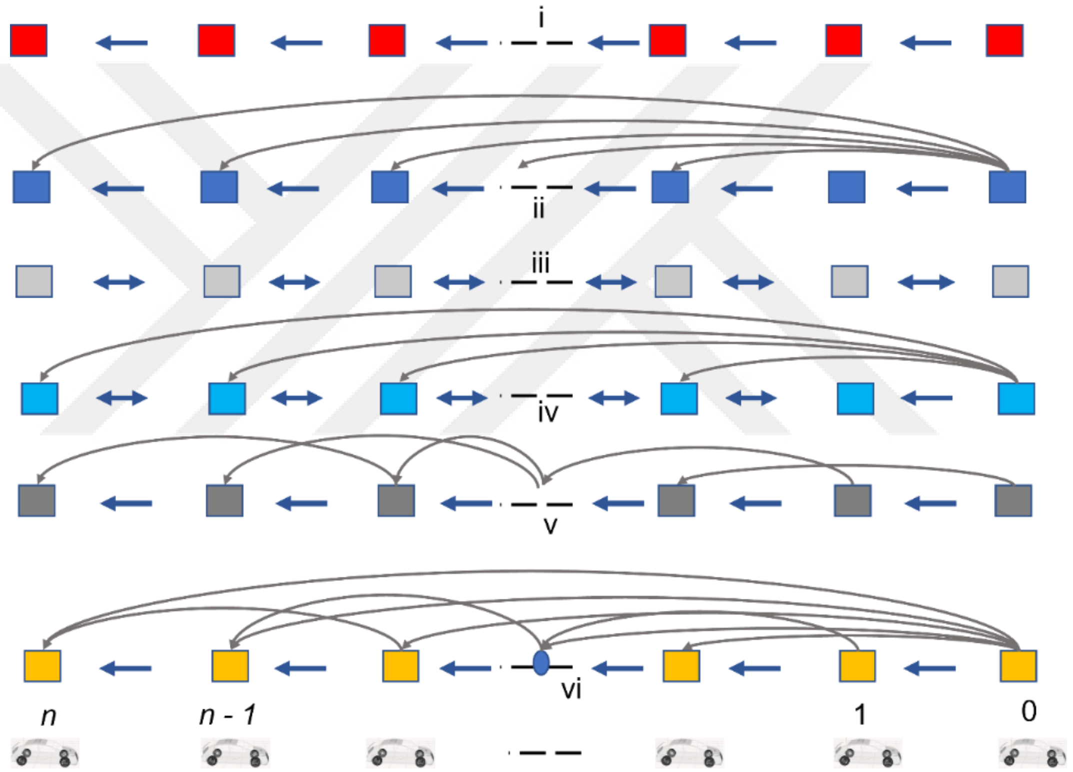

Figure 4.1: Common Information follow models in vehicle platoons.

4.2. General Platoon Model and Problem Statement

The controlled platoon comprises four vehicles in total. The Leader Vehicle (LV) and three Follower *F* vehicles. The platoon model considered in this study is based on the predecessor-leader following (*PLF*) communication model described by (Swaroop and Hedrick 1999; Wang et al. 2018; Zheng et al. 2014). We design a PID controller and an LQR to aid the control of the longitudinal distance between vehicles. Generally, both algorithms take as input the current inter-vehicle distance between vehicles (d_i to the preceding vehicle and d_{il} to the *LV*) and return as output a velocity reference with which the corresponding *F* Vehicle needs to travel to achieve the desired distance (*D*) to the preceding vehicle and D_{il} to the *LV*. Figure 4.2 depicts data acquisition, the specific parameters used by the PID algorithm, and how, ultimately, the reference velocity of the corresponding F_i vehicle is calculated from the joint outputs of the PID controllers where $x_i(t)$, $y_i(t)$ are the corresponding vehicle's world coordinates. Similarly, the LQR algorithm, Figure 4.3 uses the same world coordinates, $x_i(t)$ and $y_i(t)$ as the position state estimate at time, *t*, of the *F* vehicle in question relative to its predecessor(s). We thus state the general problem as:

Given:

$$\begin{aligned} d_i &= D \pm E_i \\ d_{il} &= D_{il} \pm E_{il} \end{aligned} \tag{4.1}$$

Where,

$$\begin{aligned} E_i &= D - d_i \\ E_{il} &= D_{il} - d_{il} \end{aligned} \tag{4.2}$$

Subjected to:

$$E_i = E_{il} \leq E_{thresh} \tag{4.3}$$

The main objective is to minimize $|E_i|$.

Therefore, the primary purpose of the PID and LQR controllers is to reduce the errors, E_i and E_{il} and drive them as close to $0m$ as possible.

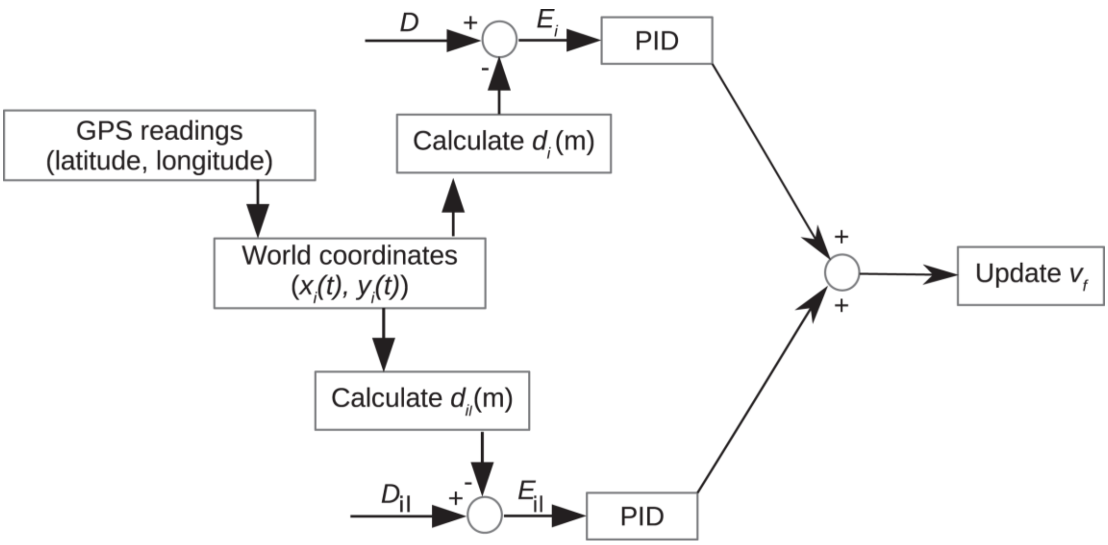

Figure 4.2: PID platoon controller model.

So the best-case scenario at any point in the simulations is to have E_i=E_{il}=$0m$, especially during the steady-state. For evaluation purposes, analyzing the results of E_i is sufficient since Ei is directly proportional to E_{il}, that is,

$$E_i = k \cdot E_{il} \tag{4.4}$$

E_i = k· E_{il}. Where k is the constant of proportionality. k = 1 in this case. Thus, the exclusion of the values of E_{il} and, incidentally, d_{il} from the graphs and tables for brevity.

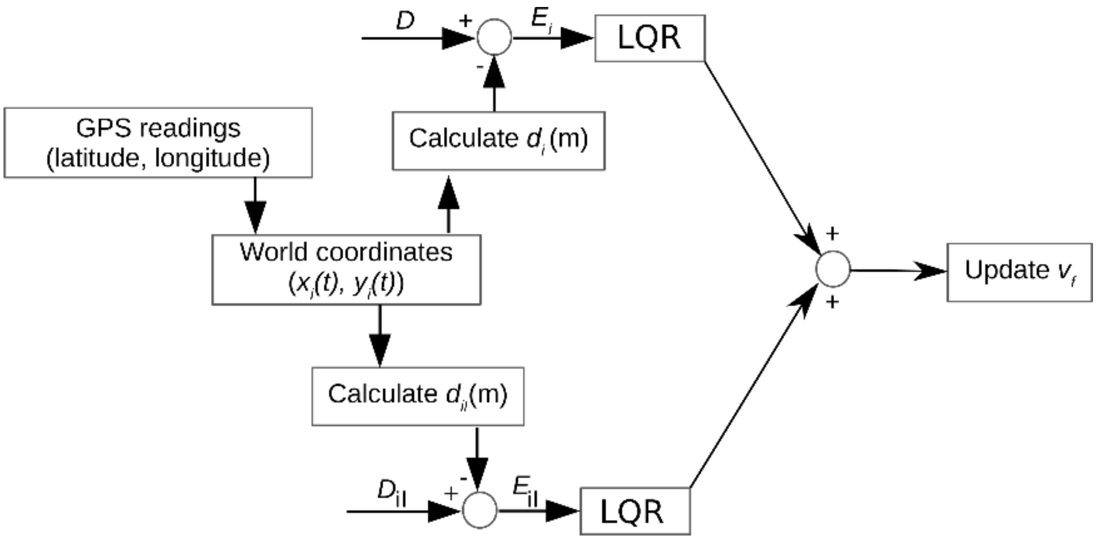

Figure 4.3: LQR platoon controller model

$\forall F_i$, $i \in \{1, 2, 3\}$ where d_i is the i^{th} inter-vehicle distance between the i^{th} F vehicle and the preceding vehicle, d_{il} is the inter-vehicle distance between the i^{th} F vehicle and the

LV, D is the desired inter-vehicle distance between the i^{th} F vehicle and its corresponding predecessor, D_{il} is the desired inter-vehicle distance between the i^{th} F vehicle and the LV, E_i is the error between the i^{th} inter-vehicle distance and the desired distance, D and E_{il} is the error between d_{il} and D_{il}. E_{thresh} is the maximum and minimum threshold value beyond which any of the errors should not exceed. This constraint ensures that F vehicles should not fall more than E_{thresh} behind the preceding vehicle, i.e., $E_i \leq E_{thresh}$ and $E \leq E_{thresh}$. It also ensures that F vehicles do not get more than E_{thresh} closer to the preceding vehicle and, incidentally, the LV, i.e., $E_i \geq - E_{thresh}$ and $E_{il} \geq - E_{thresh}$. Every F vehicle runs the corresponding control algorithm. Applying the platoon stability definition provided by (Seiler, Pant, and Hedrick 2004), the steady-state error transfer function can be written as

$$H(s) = \frac{E_i(s)}{E_{i-1}(s)} \tag{4.5}$$

This implies that platoon stability is locally guaranteed under the condition that $||H(s)||_\infty \leq 1$, and $h(t) > 0$ where $h(t)$ yields the error propagation impulse response of the i^{th} vehicle as per the ζ_2 norm as defined by (Oncu et al. 2013). ζ_∞ extends this notion throughout the whole platoon to ensure that overshoots do not occur as the signals propagate up the string, hence global stability.

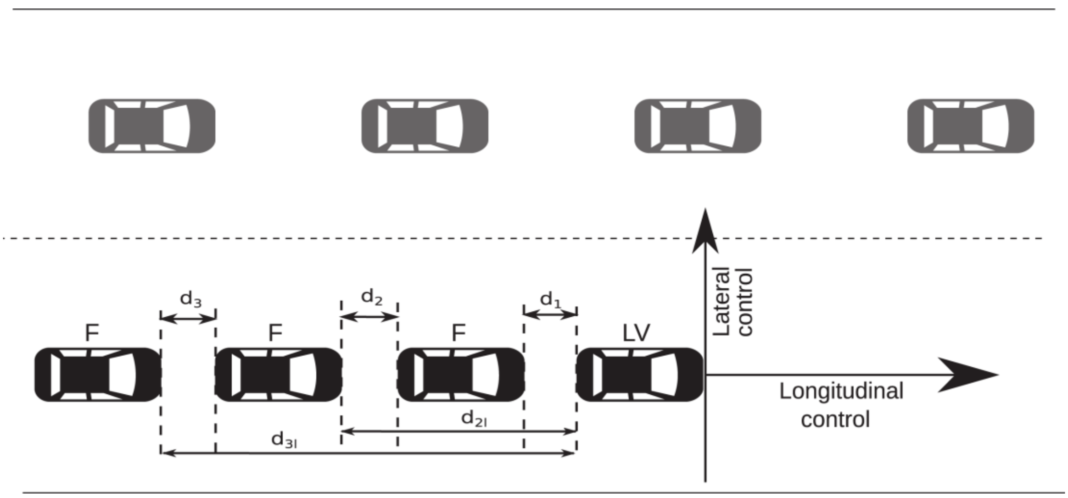

Figure 4.4: Illustration of the inter-vehicle distance d_i, d_{il}, LV, and F.

Figure 4.4 demonstrates the platoon setup. LV, is the Leading vehicle, also referred to as the root node of the platoon and is generally indexed as the first member of the platoon. The F labels depict the follower vehicles. The distance from one vehicle's head to the preceding or LV vehicle's rear is referred to as the inter vehicle gap/distance in this study. The control algorithm aims at ascertaining a constant inter-vehicle gap with all vehicles' velocity sufficiently approximately equal to that of the *LV* in the platoon using only the distance calculated from the data provided by the onboard GPS sensors of the vehicles. Errors in the inter-vehicle gap should be bounded, and there should be no collisions among platoon members in the worst-case scenario at all times during the simulation. In the simulation, constraint parameters were set as; E_{thresh} = 5m, D = 7m. At the start of the simulation, GPS data measurements are retrieved asynchronously if available from every vehicle's onboard GPS sensor. From this data, the relative intervehicle gap is calculated and forwarded to the control algorithm in question, which in turn returns the reference velocity with which the corresponding *F* vehicle's velocity is updated with the aim to achieve the desired inter-vehicle gap between F and the preceding vehicle. The desired inter-vehicle distance, *D*, is set as the PID algorithm's setpoint and is also one of the LQR algorithm's reference states' elements, whereas the intervehicle gap estimate, calculated from the data measurements provided by the GPS sensors, is provided as the feedback for the control algorithms. Similarly, the inter-vehicle gap between a given *F* vehicle and *LV*, with the exception of the second *F* vehicle, is controlled using d_{il} as the feedback and D_l as the setpoint of the PID – desired state's value for the LQR algorithm. Then the algorithms are executed in a closed loop.

4.3. Simulation Environment and Experimental Setup

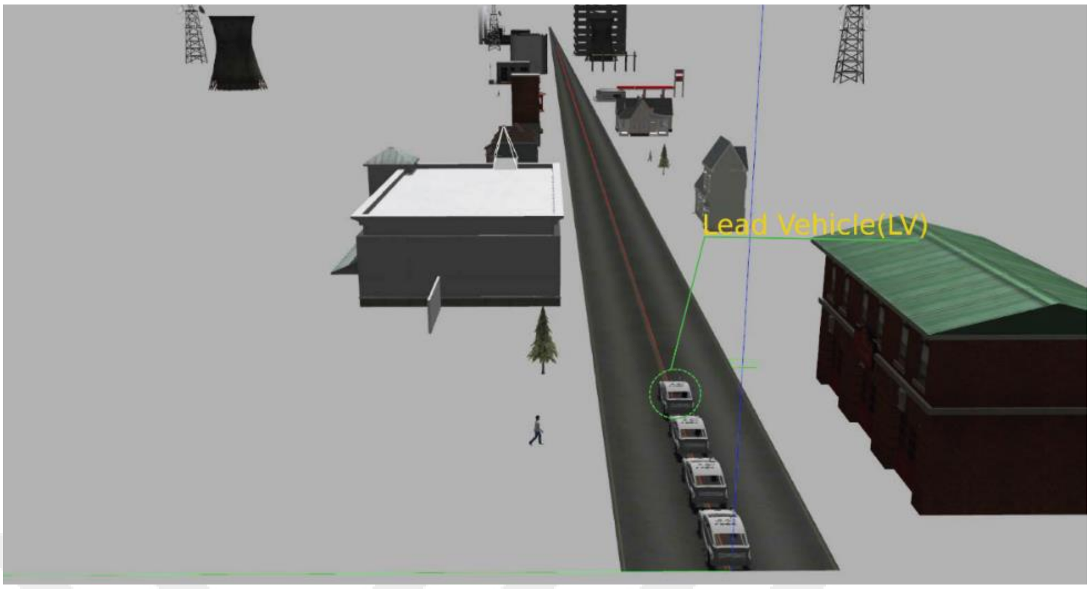

Figure 4.5: Simulation environment

3D vehicles were designed and modeled using gazebo and ROS environments. Visuals of the 3D vehicle objects were created using the gazebo platform, whereas vehicle motion was handled and controlled by the nodes implemented using the ROS framework(Quigley 2009). Figure 4.5 shows the environment in which the simulations were performed and monitored. The vehicles move forward along the road during simulation to preserve the presented platoon formation and the desired inter-vehicle distance. We performed simulations under the assumptions and constraints that:

- The roads are straight and have got no slope so that longitudinal control of the platoon remains the focus of the platoon.
- Communication is wireless, that is, over Wi-Fi (IEEE 802.11), and each vehicle is only allowed to communicate to the preceding vehicle.
- Overtake and reversing maneuvers are not allowed in the platoon.

Table 4.1: Vehicle information presents some of the major vehicle model parameters and their values.

Table 4.1: Vehicle information

Vehicle Attribute	Value
Vehicle Mass	1823.0kg
Vehicle Length	5m
Vehicle Width	1.89m
Vehicle Height	1.480m
Wheel Radius	0.34m
Wheelbase	2.95m
Wheel Width	0.225m
Drag coefficient	0.27 cd

The platoon comprises four vehicles with-not obligatorily homogeneous attributes. The vehicle attributes are presented in Table 1. We approximated these attributes in reference to a small vehicle such as the Hyundai Genesis-2014 presented by Edmunds (2014). The vehicle attributes are used in the description of the dynamics and kinematics of the joints that ultimately make up the vehicle model. ROS obtains dynamic and kinematic information of the robot by parsing Universal Robot Description Files/Formats (URDF) (Newman 2017), which are based on the Extensible Markup Language (XML). From this information, ROS conveniently calculates and generates a robot description and stores it in the ROS parameter server. From this server, all information concerning the robot is then available for processing and manipulation. A robot is defined as a set of rigid kinematic links connected by joints in a 3-dimensional (3D) world within the URDF files. The joints connecting the robot links can be one of the following six types (Angeli 2018):

- planar - allows motion in a plane perpendicular to the axis.
- floating - allows motion for all six degrees of freedom.
- continuous - a hinge joint; rotates around the axis with no upper and lower limits.

- fixed - not an actual joint since it cannot move. All degrees of freedom are locked. Its configuration requires not the axis, calibration, dynamics, safety control, or limits.
- prismatic - a joint that slides along the axis and has a limited range specified by upper and lower limits.

Alternatively, ROS provides the XML Macro (XACRO) files which enable the construction of shorter, more readable, and easier to manage XML files. XACROs can perform essential arithmetic evaluations. They provide the feature of code reuse since they can access macros from external files-so xacro files have the ability to extend other xacro files. This feature simplifies the modeling of more complex robots e.g., vehicle parts; wheel, body, steering, etc., can be described in individual xacro files, and then combined in a single xacro file, say vehicle.xacro. The vehicle.xacro file is then expanded to generate the final URDF file used by ROS. ROS stores the robot information as a robot description in the ROS parameter server from which one can access this information using any higher-level programming language supported by ROS. Such languages include Python, C++ to mention but a few. Furthermore, the xacro files are also capable of parsing arguments defined in YAML configuration files (Conley 2009). The ability to parse arguments enable the use of variables in the evaluation of arithmetic operations. We pass the vehicle attributes presented in Table 4.1 as arguments to the xacro files since this format allows freedom of modification for the various vehicles, i.e., we can spawn simultaneously spawn multiple robots with similar or different attributes using the same robot template with varying values. Gazebo is an open-source robotics simulator developed by the University of California and Willow Garage. ROS and gazebo interact intimately to facilitate robot control and simulation. Gazebo offers an option to launch the simulator as a ROS node. Thus, information published via ROS messages and topics becomes available for utilization to the gazebo simulator. Figure 4.6 shows the generalized structure of the nodes and their connections. The *platoon_controller* node is the center of our inter-vehicle distance control algorithm. Within this node is the PID control algorithm which takes as input the GPS data containing coordinate measurements of the preceding vehicle and the *LV*. The *F* vehicle running the algorithm then calculates its distance to the preceding vehicle and the *LV* and passes the result to the control algorithm. The output controller(s) becomes the velocity reference of the *F* vehicle in question. GPS data is generated by a GPS sensor attached to each vehicle, and the data is published by every vehicle's corresponding *vehicle/gps* node. The *F* vehicle only needs to subscribe to the topic to which the preceding vehicle publishes its data. Each vehicle knows its node name, and all data is namespaced with the vehicle node name followed by the topic name to which the data is published. E.g., the *LV* (first vehicle of the platoon) is named *vehicle1*. Its GPS data is published to the topic named *vehicle1/gps*. The vehicle immediately behind the *LV* is named *vehicle2* and its corresponding GPS data is published to *vehicle2/gps* and so forth. Assigning names to vehicle node names occurs during the platoon formation process, during the generation of the vehicle description

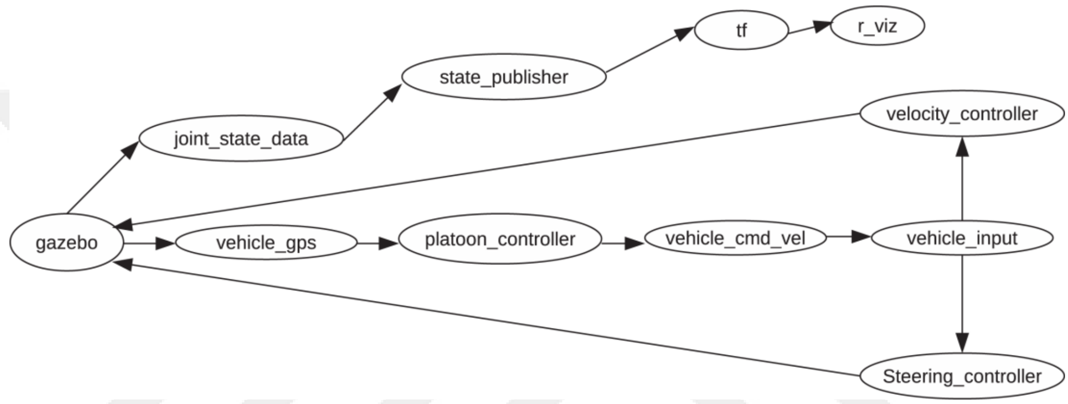

Figure 4.6: Internode relationships

information for ROS, and right before the generated robot models spawn in the gazebo simulator.

The *platoon_controller* publishes PID output velocity references to the respective *vehicle_cmd_vel* nodes of the corresponding vehicles via separate *vehicle_cmd_vel* topics. The *vehicle_input node* consumes this velocity reference data and prepares the different velocity components from this data, i.e., either angular or linear velocity. The *steering_controller* node handles the angular velocity component, whereas the *velocity_controller* node handles the linear velocity component.

Equation (4.9) and equation (4.10) provide the main formulae which we employ to calculate the velocity of the vehicles. ROS control is a rather helpful ROS package utility for robot control in ROS. It provides ready implemented joint control loops such as PID loop, Sinesweep to ensure joint frequency control, to mention but a few. The utility further aids in the abstraction of the robot hardware through hardware interfaces for all the uniquely defined robot joints. The *velocity_controller* and *steering_controller* nodes implement the interfaces and provide the inputs to the ROS control utility. This utility abstracts the direct interaction of our application with the vehicle resources such as actuators/motors, sensors, etc. (Chitta, Marder-Eppstein, Meeussen, Pradeep, Rodríguez Tsouroukdissian, et al. 2017) provide more information about the details of the utility's method of operation, configuration, and the like. The *r_viz*, *state_publisher*, *joint_state_data*, and the *tf* nodes are important utility nodes. The *r_viz* node is from the *r_viz* ROS package, which enables robot visualization in real-time. Using this node, we can check the status of each vehicle. The *joint_state_data* node holds data of the joints of the robot. We can monitor joint states at any given time. This data is made available by the ROS control package. The *state_publisher* node enables us to publish this data as we wish.

The *tf* node provides access to functionalities offered by the Transform library (TF) (Foote 2013). This library provides a standard way to track data in different 3D coordinate frames in real-time since robots generally have multiple constantly changing coordinate frames over time. These frames include the world frame, base frame, etc. This node makes it possible for us to track any robot frame we are interested in at any moment in time. The gazebo node facilitates connection to the gazebo simulator, which we use to visualize the entire platoon during simulation.

Equation (4.6), equation (4.7), and equation (4.10) represent the inertia matrices of the vehicle main body, vehicle wheel and the vehicle steering as defined in the xacro files which are ultimately evaluated to derive the underlying lower-level vehicle dynamics:

$$BodyInertia = \begin{matrix} \frac{(W^2 + M_h^2) \cdot M}{12} & 0 & 0 \\ 0 & \frac{(L^2 + M_l^2) \cdot M}{12} & 0 \\ 0 & 0 & \frac{(W^2 + L^2) \cdot M}{12} \end{matrix} \tag{4.6}$$

$$WheelInertia = \begin{matrix} \frac{R^2 \cdot M}{2} & 0 & 0 \\ 0 & \frac{R^2 \cdot M}{4} + \frac{W^2 \cdot M}{12} & 0 \\ 0 & 0 & \frac{R^2 \cdot M}{4} + \frac{W^2 \cdot M}{12} \end{matrix} \tag{4.7}$$

$$SteeringInertia = \begin{matrix} \frac{R^2 \cdot M}{4} + \frac{L^2 \cdot M}{12} & 0 & 0 \\ 0 & \frac{R^2 \cdot M}{2} & 0 \\ 0 & 0 & \frac{R^2 \cdot M}{4} + \frac{L^2 \cdot M}{12} \end{matrix} \tag{4.8}$$

where: W is the vehicle width, M_h is vehicle height, L is the vehicle length, M is the vehicle mass and R is the wheel radius. We use such matrices to also define the limits and values of the vehicle joints in to have the 3D vehicle models spawned in the Gazebo platform move realistically in our 3D world. Further details of low-level designs and control are left out for brevity since they are not the main focus of this study. We calculate the velocity to the rear left and right wheel actuators as follows:

$$V_{rb}, V_{lb} = \frac{v_x}{r} \tag{4.9}$$

Equation (4.19) yields the velocity components for the left *(V_{lb})* and right *(V_{rb})* rear wheels of the vehicle. Where v_x is the x component of the linear input velocity, and r is the radius of the vehicle's wheel. Finally:

$$V = \frac{V_{rb} + V_{lb}}{2} \cdot r \tag{4.10}$$

V is published to a PID controller whose result is then directly issued to the joint links controlling the vehicle wheels. Control of steering is beyond the scope of this study and thus left out for brevity. Initially, all vehicles are at rest (0 m/s) and are at an inter-vehicle distance of 1m apart. The inter-vehicle distance is measured from the head of the following vehicle to the rear of the preceding vehicle. The GPS sensors are, however, placed at the corresponding vehicle's center of mass as (Redondo et al. 2018) recommend. The *LV* starts accelerating uniformly at 0.1m/s^2 at a time, $t > 0s$ with a time-step of 0.1s until it finally reaches a pre-defined velocity reference, $V_1 = 22 \pm 0.001$ m/s. It then moves with constant velocity, V_1 m/s for a pre-defined duration, $T_1 \approx 55$s, after which period, the *LV* starts to gradually decelerate at a rate of 0.1 m/s^2 at time intervals of 0.1s until it reaches velocity $V_2 = 10 \pm 0.001$ m/s. The *LV* moves with constant velocity, V_2 m/s for a duration of $T_2 \approx 74$s, after which it linearly decelerates to velocity $V_3 = 5 \pm 0.001$m/s and maintains V_3 for $T_3 = 74$s. Ultimately, the *LV* decelerates to rest. We terminate the simulation after the entire platoon reaches 0 m/s, i.e., at rest. The follower (*F*) vehicles accelerate and decelerate accordingly, with their reference velocities being the outputs returned by their respective control algorithms. Each *F* vehicle simultaneously controls its distance to both the preceding vehicle and the *LV* - except the second, *F*, vehicle whose preceding vehicle happens to be the *LV*. We, additionally, include a different scenario (uncertain scenario) during which the *LV's* velocity is uncertain and changes continuously. That is, the velocity of the *LV* is never constant. Given that acceleration and deceleration of the *LV* can be viewed as disturbances in a platoon according to (Zheng et al. 2014) the *LV* occasionally exhibits random, abrupt and sharp accelerations and/or decelerations during this scenario. This behavior is intended to simulate miscellaneous real-life occassions where the *LV* maybe required to make an abrupt deceleration or acceleration For instance, to either dodge an obstacle or prevent collision. At the end of this scenario, the *LV* decelerates with a relatively larger deceleration magnitude. This enables us to evaluate how the platoon responds to such extreme scenarios. We present the platoon's performance during this scenario in the Results section. Vehicles publish their ROS messages at an average rate of *≈10Hz* relatively slower than the suggested maximum frequency of *33Hz*, in (Institute 2019), because we expect the signal transmission to the GPS receivers to be slower and, in some cases, unstable. We also configure the GPS with an update rate of 0.1s. We only update feedback to the controllers when the corresponding GPS sensors publish new measurement data. We tune the GPS with the following settings: a standard deviation of the additive Gaussian noise to the position of 0.01 for the latitude, longitude, and altitude, the standard deviation of the relative velocity error in GPS readings is 0.1m/s.

4.4. PID Controller

This section briefly introduces the Proportional, Integral, and Derivative (PID) controller along with its general formulation and the meaning of the major parameters of the algorithm. Figure 4.7 shows the general closed loop of a PID.

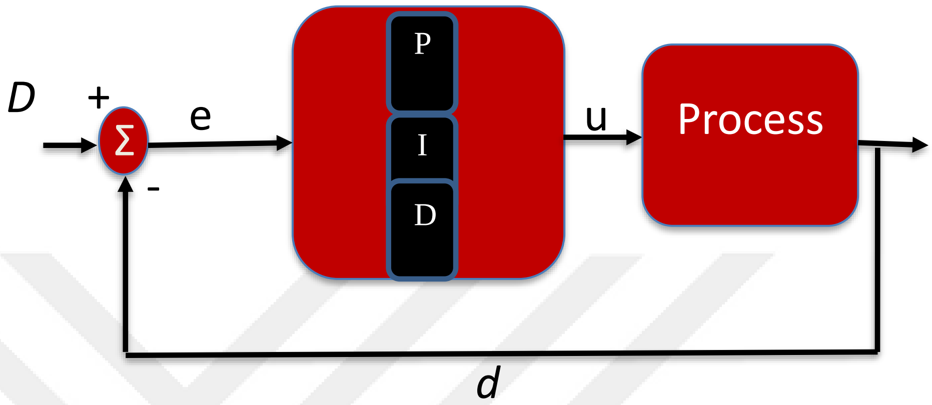

Figure 4.7: General closed loop PID control

4.4.1. Brief Introduction

In this section, a brief introduction and overview of the Proportional Integral and Derivative(PID) control (Åström and Hägglund 1995; Ogata 2010) concept is presented. The basic PID control algorithm, formulation and implementation are also discussed in this section.

All over the world today, most feedback loops are controlled using the PID controller or its modifications - be it as a stand alone controller or as a part of the overall control package. The PID controller is arguably the most common control method owing most of its fame to its general applicability to most control systems. This is more so the case for control systems whose mathematical model is unknown, thereby nullifying the application of analytical control and design methods (Ogata 2010).

4.4.2. PID Formulation

The goal of a PID controller is to have the difference between the Desired value (*D*), and the measure process value (*d*) as close to zero as possible. *D* is optionally referred to as the SetPoint(*SP*) The difference between *D* and *d* is referred to as the error(*e*). The standard PID formula can be written as:

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau)d\tau + k_D \frac{de(t)}{dt} \tag{4.11}$$

Where;

k_P : is the proportional gain

k_D : is the derivative gain

k_I : is the integral gain

d(t) : the change in time

u(t) : the system control variable.

e(t) : is the error between the desired value and the actual value at a given time, *t*.

PID control is all about driving the error, *e*, to *0*. The value *d* is also, referred to as the feedback value. This is usually measure or estimated with the aid of sensors and is sometimes also referred to as the process measurement. It is re-fed back into the control system, hence the term feedback and depending on the error a new control input is determined so as to minimize the error(Åström et al. 2006; Chang and Yuan 2018; Ogata 2010). A control system that determines the control input based on the so-called feedback value is referred to as a closed-loop control system.

4.5. LQR

In this section, the Linear Quadratic Regulator is briefly introduced and presented. Related variations, parameters and what they stand for are also explained in this section.

4.5.1. Brief Introduction

In this section, a brief introduction and overview of the (Linear Quadratic Regulator) LQR optimization concept is presented. The basic LQR optimization algorithm, formulation and implementation are also discussed in this section. The optimal control theory is a mathematical optimization approach towards providing solutions for system control. The LQR is a subcategory of the optimal control theory. In optimal control, the main aim is to calculate a control for a given system – usually dynamic – that optimizes the system's corresponding cost function (Anderson and Moore 2007; Heydari and Balakrishnan 2012; Smallwood and Sondik 1973).

LQR deals with controlling a system, whose dynamics are expressed using a combination of *Linear* differential equations, with a corresponding *Quadratic* cost function at an optimal cost, in general, and *minimum,* in particular. This is referred to as solving the *LQ* problem. The *Linear Quadratic Regulator* is thus one of the approaches to solving the *LQ* problem.

In regulation control, weights are provided that aid in the minimization of the corresponding cost function. The cost function is the total of all errors in the observed measured values relative to the desired values. LQR is a feedback loop control algorithm objectively concerned with minimizing the cost function. Its regulation is performed around a predefined reference state.

4.5.2. LQR Derivation and Formulation

The LQR algorithm can be divided into two major categories depending on the time, generally referred to as the *horizon*, over which the optimization is performed. This can horizon can be one of finite or infinite (Bertsekas 2019; Bertsekas and Shreve 1996). Furthermore, the formulation is done depending on whether the finite or infinte horizon LQR algorithm is to be applied to a discrete or continous system

(Bertsekas 2012; Bertsekas and Shreve 1996; Carlson, Haurie, and Leizarowitz 2012; Garg et al. 2011; Heydari and Balakrishnan 2012; Milano 2018).

Continuous-time formulations are done assuming the state equation of the continous-time linear system as provided in equation (4.12), provided that

$x \in \mathbb{R}^n$ $is\ the\ state\ of\ the\ system\ in\ question\ and\ u \in$
$\mathbb{R}^m$ $is\ the\ control\ input\ vector.$

A and *B* are the state and input matrices respectively. They are constants whereas *x* and *u* are functions – of time in this case.

$$\dot{x} = Ax + Bu \tag{4.12}$$

a) Infinite-horizon continuous LQR formulation

The cost of a continuous system provided in equation (4.12) can be generally formulated as:

$$J = \int_0^\infty (x^T Q x + u^T R u + 2x^T N u) dt \tag{4.13}$$

It should be noted that the term in N is almost always left out for most dynamic systems. With the feedback gain, *K*, given as:

$$K = R^{-1}(B^T P \ + N^T) \tag{4.14}$$

and solving the algebraic Riccati equation (ARE), (Bertsekas 2012), for P as:

$$A^T P \ + PA \ - \ (PB + N)R^{-1}(B^T P + N^T) + Q = 0 \tag{4.15}$$

The control variable which gives the minimum value of the cost function defined in equation (4.10) can be given as:

$$u = -Kx \tag{4.16}$$

b) Infinite-horizon discrete-time LQR formulation

The corresponding discrete-time LQR version can be formulated as follows.

The corresponding state equation of discrete-time linear system becomes:

$$x_{k+1} = Ax_k + Bu_k \tag{4.17}$$

The associated cost function can be evaluated as:

$$J = \sum_{0}^{\infty}(x_k^T Q x_k + u_k^T R u_k + 2x_k^T N u_k) \tag{4.18}$$

Defining *K* as:

$$K = (R + B^T P B)^{-1}(B^T P A + N^T) \tag{4.19}$$

Following, calculating for *P* from the discrete-time algebraic Ricatti Equation (DARE) as:

$$P = A^T P A - (A^T P B + N)(B^T P B + R)^{-1}(N^T + B^T P A) + Q \tag{4.20}$$

The control can finally be obtained as:

$$u_k = -K x_k \tag{4.21}$$

c) Finite-horizon continuous time LQR formulation

Considering the system represented by equation(4.12), the corresponding cost function evaluation for finite horizon LQR can be formulated as:

$$J = x^T(t_1)F(t_1)x(t_1) + \int_0^{\infty}(x^T Q x + u^T R u + 2x^T N u)dt \tag{4.22}$$

Defining the gain *K* as:

$$K = R^{-1}(B^T P(t) \ + N^T) \tag{4.23}$$

and *P* from solving the corresponding ARE as:

$$A^T P(t) \ + P(t)A \ - \ (P(t)B + N)R^{-1}(B^T P(t) + N^T) + Q = -\dot{P(t)} \tag{4.24}$$

Subjected to the boundary condition(BC),

$$P(t_1) = F(t_1) \tag{4.25}$$

The control variable can be calculated by:

$$u = -Kx \tag{4.26}$$

d) Finite-horizon discrete time LQR formulation

The finite-horizon discrete time cost function for a system described in equation (4.17) can generally be formulated as:

$$J = x_{H_i}^T Q x_{H_i} + \sum_{0}^{H_i-1} (x_k^T Q x_k + u_k^T R u_k + 2x_k^T N u_k) \tag{4.27}$$

Where H_i is the time horizon.

Now, calculating the feedback gain K as:

$$K_k = (R + B^T P_{k+1} B)^{-1} (B^T P_{k+1} A + N^T) \tag{4.28}$$

Where P_k can be found through iteratively solving the following Riccati Equation.

$$P_{k-1} = A^T P_k A - (A^T P_k B + N)(B^T P_k B + R)^{-1}(N^T + B^T P_k A) + Q \tag{4.29}$$

With the boundary condition,

$$P_N = Q \tag{4.30}$$

The the control variable can be evaluated as:

$$u_k = -K_k x_k \tag{4.31}$$

Results presented were obtained using the formulation from this sub-category. The ***x*** term in the control variable equations; equation (4.16), equation (4.21), equation (4.26), and equation (4.31) can, in some cases like ours, be more conveniently expressed as the error between the desired state and current system state. For instance, taking the $\boldsymbol{x}$ term in equation (4.31) as $\bar{x} = x_{k,a} - x_d$ where $x_{k,a}$ is the actual system state and x_d is the desired state.

Now, the major 15 states of a vehicle are presented in equation (4.32) below.

$$states = \left(X, Y, Z, roll, pitch, yaw, \dot{X}, \dot{Y}, \dot{Z}, r\dot{o}ll, pi\dot{t}ch, y\dot{a}w, \ddot{X}, \ddot{Y}, \ddot{Z}\right) \quad (4.32)$$

After specifying the vehicle model equations provided in section 3.2 into the ROS URDF files, we get access to this state information via the vehicle odemetry topic after *proper setup* at any desired point in time. For longitudinal control, $X, Y, \dot{X}, and\ \dot{Y}$ are sufficient. $Y\ and\ \dot{Y}$, like the other states are not paramount to platoon control but may prove useful in the verification of correctness and accuracy. We obtain *X, Y, and Z* measurements from the GPS sensor and use the states provided by ROS as a part of the verification process during simulations.

CHAPTER 5

RESULTS

In this chapter, results from the performed simulations for all the scenarios under both controllers are presented. The results are followed by plot descriptions and explanations.

5.1. Scenario 1

We performed simulations by periodically varying the *LV's* reference velocity seamlessly to V_1 = 22 m/s, V_2 = 10 m/s, and V_3 = 5 m/s. This is realized by publishing velocity commands to the LV's velocity topics via ROS. Reference velocities of the F vehicles were determined by the PID controllers. The desired inter-vehicle distance, D, in the simulations was set to 7 m.

Table 5.1: PID Controller parameters

Parameter	Value
P	0.07
I	0.00005
D	0.08

V_1, V_2, V_3, *and D* were arbitrarily chosen. Table 5.1 shows the PID gains used by the PID controllers. We break down the simulation and entire motion of the platoon into eight distinct zones represented by letters A through H in the graphed results for the first scenario. These zones are different color coded for a more straightforward distinction and clarity. These colors, in brief, represent red: the zone where *LV* is accelerating; white: zone where the LV is moving with constant velocity or at rest; green: zone in which the *LV* is decelerating. Table 5.2 provides a more detailed description of the zones. This section presents the results obtained from the simulations.

Table 5.2: Description of the distinct zones throughout the simulations

Zone	Color	Explanation
A	Red	LV Accelerating from 0 to 22 m/s
B	White	LV moving with constant velocity, 22 m/s
C	Green	LV decelerating from 15 to 10 m/s
D	White	LV moving with constant velocity, 10 m/s
E	Green	LV decelerating from 10 to 5 m/s
F	White	LV moving with constant velocity, 5 m/s
G	Green	LV decelerating from 5 m/s to rest
H	White	All vehicles at rest with 0 m/s.

5.1.1. Results obtained using PID controller

In this section, graphs of the results obtained using the PID algorithm are presented.

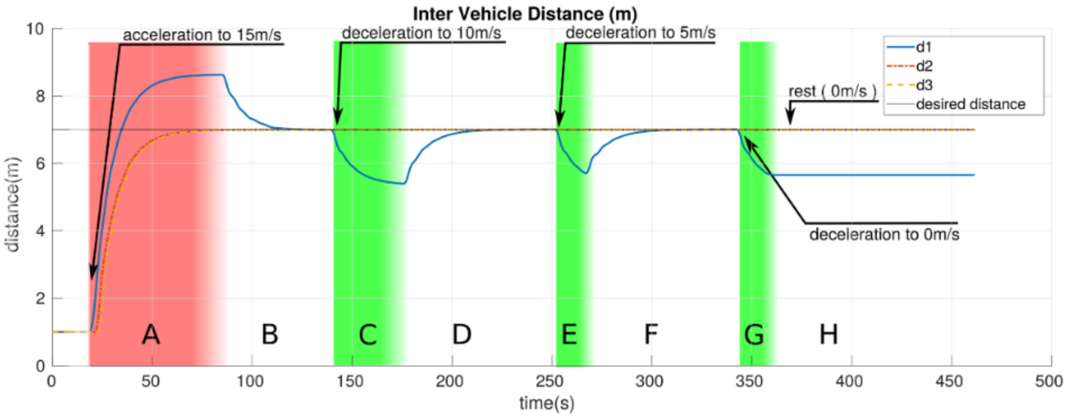

Figure 5.1: IVD while using the PID controller

In Figure 5.1, the change in the inter-vehicle distance of platoon members over simulation time while using PID control is shown. The maximum, and only, overall overshoot is exhibited in d_1 in this scenario. Neither overshoots nor oscillations are observed in d_2 and d_3 throughout the simulations in this scenario.

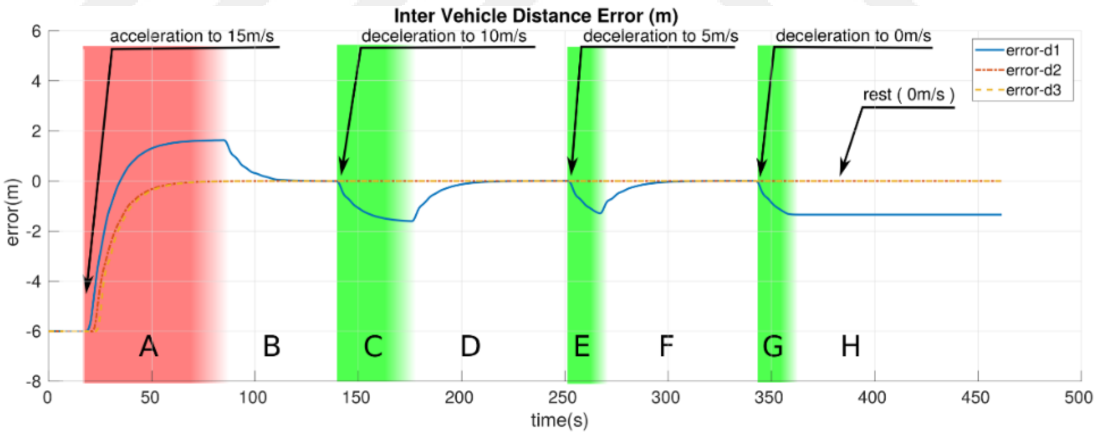

Figure 5.2: Error in the IVD while using the PID controller

The graph in Figure 5.2 shows how the corresponding error in the inter-vehicle distance between corresponding platoon members changes over time during the simulation in reference to the desired inter-vehicle distance. It can be noted that F_1 generally experiences much greater errors in comparison to F_2 and F_3 as it only gets information from one vehicle (*LV*) whereas F_2 and F_3 receive information from two different sources (preceding vehicle and *LV*).

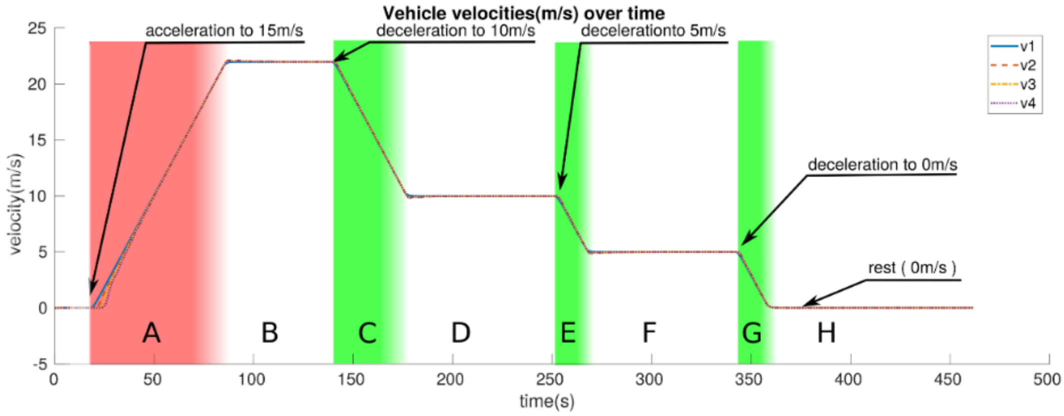

Figure 5.3: Platoon velocity profile while using the PID controller

The plot in Figure 5.3 presents the respective velocities of the platoon members throughout the simulation.

The parameter gains that yielded the results obtained using the PID controller are provided in Table 5.1. The parameters were selected through trial and error.

Table 5.3: Error statistics of the three inter-vehicle distances while using PID

d_i	**d_1**		**d_2**		**d_3**	
Zone	**Std(m)**	**Var(m^2)**	**Std(m)**	**Var(m^2)**	**Std(m)**	**Var(m^2)**
A	0.3876	0.1502	0.3701	0.1370	0.4344	0.1887
B	0.2410	0.0581	0.0014	0.0000	0.0024	0.0000
C	0.3890	0.1513	0.0006	0.0000	0.0002	0.0000
D	0.2359	0.0557	0.0000	0.0000	0.0001	0.0000
E	0.4284	0.1835	0.0013	0.0000	0.0001	0.0000
F	0.2028	0.0411	0.0002	0.0000	0.0001	0.0000
G	0.4737	0.2244	0.0001	0.0000	0.0001	0.0000

Table 5.3 provides statistics of the error in the inter-vehicle distances at different zones throughout the simulation when PID control is applied. The statistics presented are the standard deviation (std) and the variance (var) of the inter-vehicle distances within the specified regions.

5.1.2. Results obtained using LQR controller

Graphs demonstrating the performance of the LQR are presented in this section

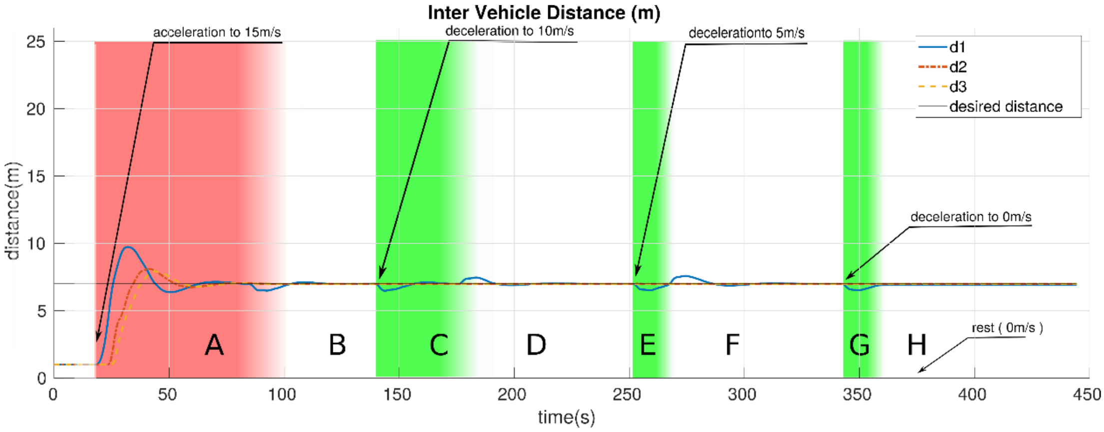

Figure 5.4:IVD while using the LQR algorithm

The graph in Figure 5.4 shows the change in the inter-vehicle distance of platoon members over simulation time while using LQR control. The graph shows that d_1, experiences the maximum overshoot followed by d_2 and d_3 shows the minimum overshoot. The graph also exhibits mild transient oscillations in IVD till the system finally settles.

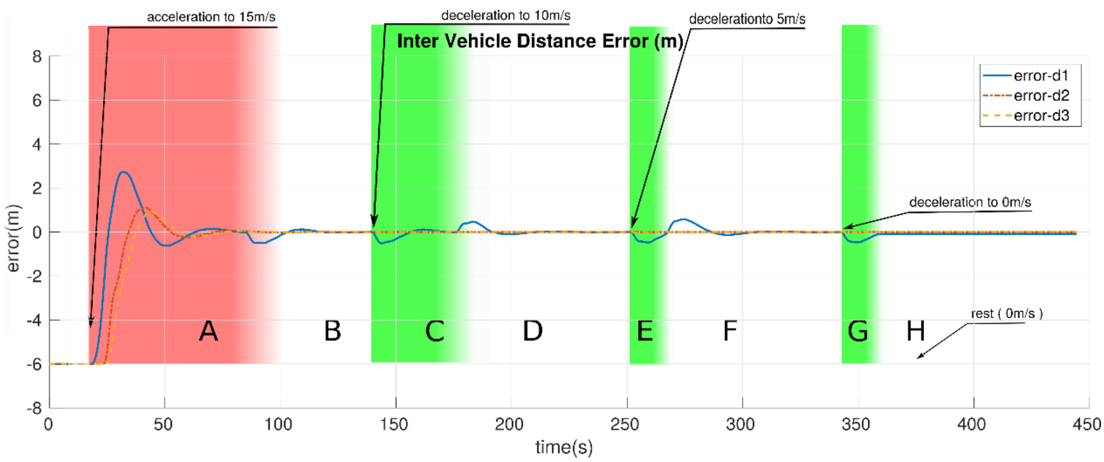

Figure 5.5: Error in the Error while using the LQR controller

The graph in Figure 5.5 depicts the observed IVD error among platoon members under LQR control. Ignoring errors resulting from the initial platoon position, it can be observed that the error in d_1 exhibits the maximum transient error (*2.73m*), and maximum error overall followed by the maximum error in d_2 (*1.12m*) and finally the maximum error in d_3 (*0.86m*).

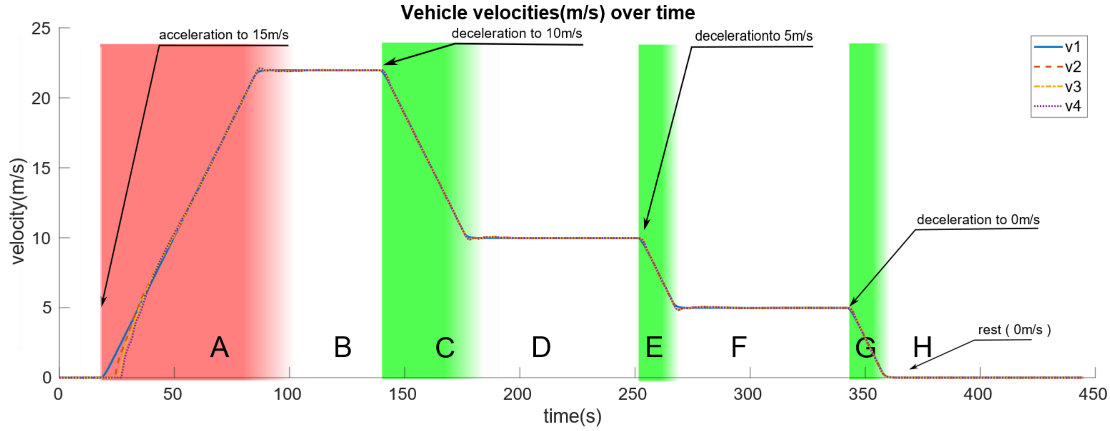

Figure 5.6: Platoon velocity profile while using the LQR algorithm

In Figure 5.6, the velocity profile of the platoon members during the simulations under LQR control is shown.

Table 5.4: Error statistics of the three iner-vehicle distances while using LQR

d_i	**d_1**		**d_2**		**d_3**	
Zone	**Std(m)**	**Var(m^2)**	**Std(m)**	**Var(m^2)**	**Std(m)**	**Var(m^2)**
A	2.0129	4.0519	0.4625	0.2139	0.0881	0.0078
B	1.0644	1.1329	0.7581	0.5747	1.0677	1.1400
C	0.0204	0.0004	0.0065	0.0000	0.0040	0.0000
D	0.1937	0.0375	0.0076	0.0001	0.0076	0.0001
E	0.2085	0.0435	0.0001	0.0000	0.0044	0.0000
F	0.2058	0.0423	0.0007	0.0000	0.0030	0.0000
G	0.0115	0.0001	0.0000	0.0000	0.0002	0.0000

Table 5.4 provides statistics of the error in the inter-vehicle distances at different zones throughout the simulation when PID control is applied. The statistics presented are the standard deviation (std) and the variance (var) of the inter-vehicle distances within the specified regions.

5.2. Scenario 2 (Uncertain Scenario)

During this scenario, platoon performance of each controller is examined when the *LV* velocity profile is uncertain. The *LV* velocity is continuously varied throughout the simulation period.

5.2.1. PID controller performance

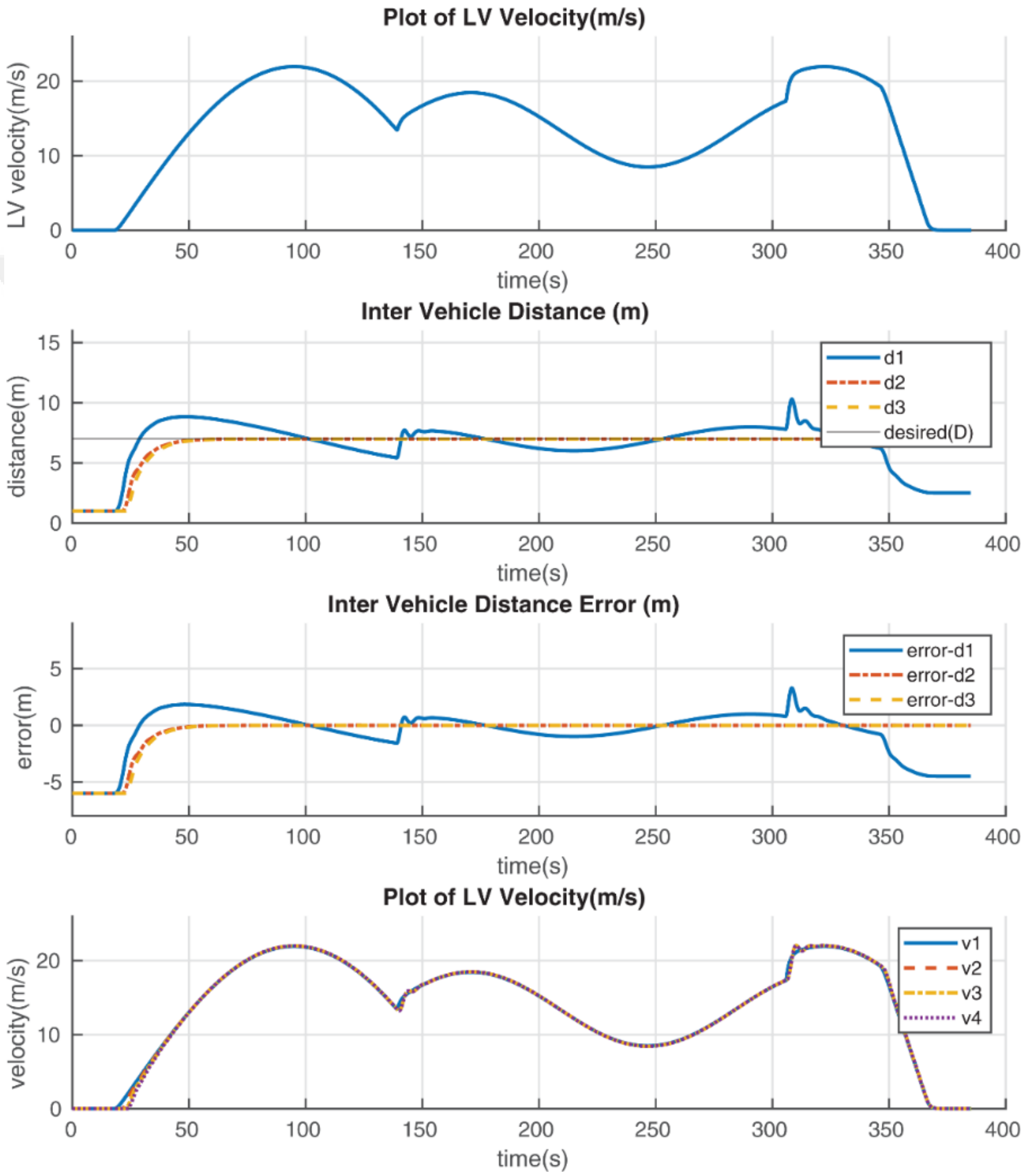

Figure 5.7: Platoon performace when the LV velocity is continuously varied - uncertain scenario while using the PID controller

Figure 5.7 shows the results of the proposed controller's performance when the *LV's* velocity is continuously changing while the platoon is being controlled using a PID controller. From top to bottom, in Figure 5.7 is the LV's *velocity* plot throughout the scenario, followed by the platoon's inter-vehicle distance (*IVD*), *error,* and *velocity*. The maximum and minimum observed errors during this scenario are *3.30 m* and *-4.50 m*, respectively. The maximum acceleration magnitude observed in this scenario was $2\ m/s^2$ observed when the elapsed time was *306s*. The LV accelerated from 17m/s to 21 m/s in *2s*. It can also be observed that the *LV* decelerates relatively fastly towards the end of the scenario till it comes to rest. That is, from the period when the elapsed time is 348s till the LV comes to rest. The *LV* decelerates from 19.5 m/s to 0 m/s in 17 s. It should be noted that the error measurements used in the calculation of these statistics are those values logged after the first *F* vehicle observes an inter-vehicle gap of 7 m to the *LV* for thefirst time. That is, d_i = 7 m for the first time. Error values prior to this time are not taken into account during the calculation of statistics in order to mitigate the influence of the initial conditions to the performance evaulation of the proposed controller. The initial conditions, that is, all vehicles being 1m apart were set arbitrarily. This serves two purposes, i.e., ensures that there is no reverse motion in the platoon and also enables us to monitor how well and how fast the system recovers from an inital error situation. The error plots, therefore, register starting error values of *-6 m* in all the inter vehicle distances, d_i, between vehicles. The negative sign means that the vehicles are 6m *closer* to each other than desired inter-vehicle distance. A positive sign in the error thus means that the vehicles are further from each other than desired.

5.2.2. LQR controller performance

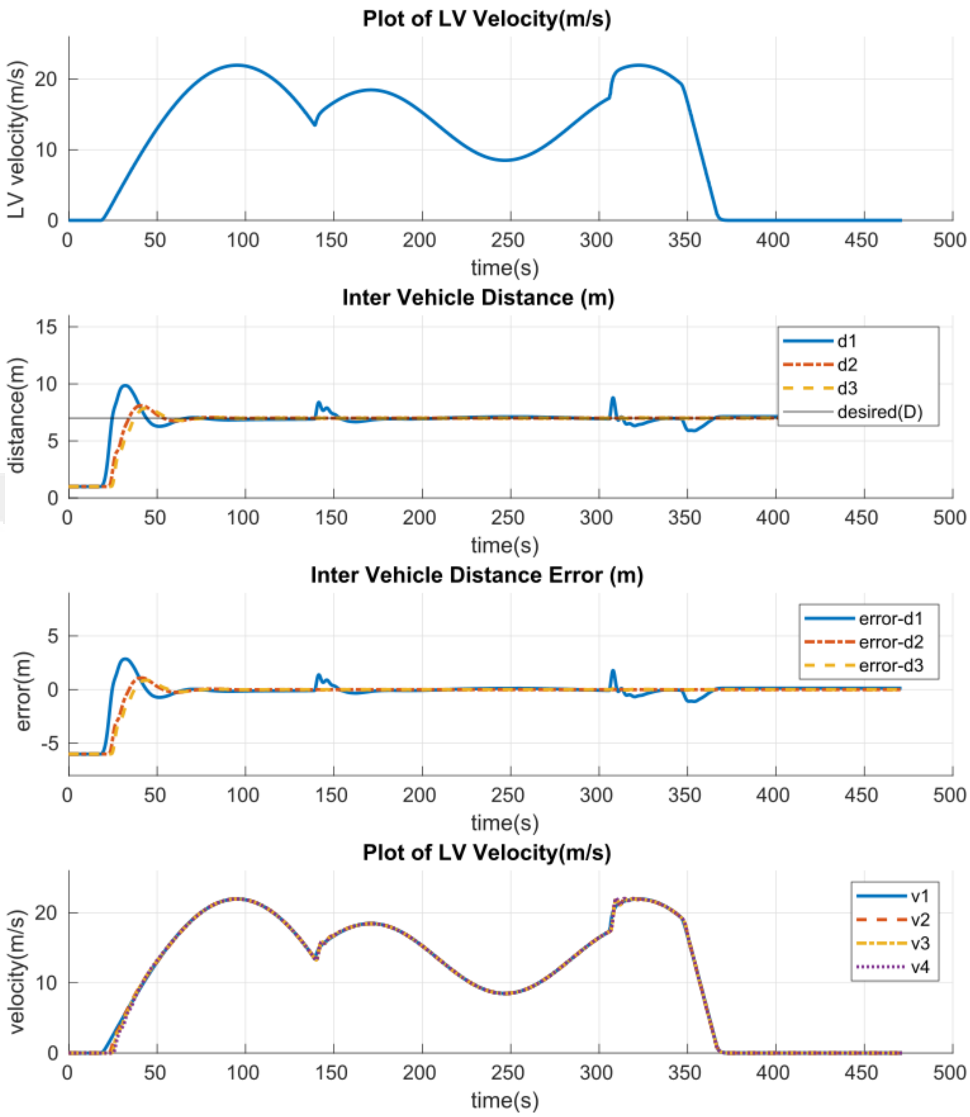

Figure 5.8: Platoon performace when the LV velocity is continuously varied - uncertain scenario while using the LQR

Figure 5.8 shows the results of the proposed controller's performance when the *LV's* velocity is continuously changing while the platoon is being controlled using a PID controller. From top to bottom, in Figure 5.8 is the *LV's velocity* plot throughout the scenario, followed by the platoon's inter-vehicle distance (*IVD*), *error*, and *velocity*. The maximum and minimum observed errors during this scenario are *2.86 m* and *-1.11 m*, respectively. The maximum acceleration magnitude observed in this scenario was *2* m/s^2 observed when the elapsed time was *306s*. The LV accelerated from 17 m/s to

21 m/s in *2s*. It can also be observed that the *LV* decelerates relatively fastly towards the end of the scenario till it comes to rest. That is, from the period when the elapsed time is 348s till the LV comes to rest. The *LV* decelerates from 19.5 m/s to 0 m/s in 17s. It should be noted that the error measurements used in the calculation of these statistics are those values logged after the first *F* vehicle observes an inter-vehicle gap of 7m to the *LV* for thefirst time. That is, d_i = 7m for the first time. Error values prior to this time are not taken into account during the calculation of statistics in order to mitigate the influence of the initial conditions to the performance evaulation of the proposed controller. The initial conditions, that is, all vehicles being 1m apart were set arbitrarily. This serves two purposes, i.e., ensures that there is no reverse motion in the platoon and also enables us to monitor how well and how fast the system recovers from an inital error situation. The error plots, therefore, register starting error values of *-6m* in all the inter vehicle distances, d_i, between vehicles. Again, negative sign means that the vehicles are 6m *closer* to each other than desired inter-vehicle distance. Also again, positive sign in the error thus means that the vehicles are further from each other than desired. The main parameters used by the LQR controllers to obtain results presented in this chapter are provided below:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \delta t \cdot cos(\theta) & 0 \\ \delta t \cdot sin(\theta) & 0 \\ 0 & \delta t \end{bmatrix}, Q = \begin{bmatrix} 1650 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, R = \begin{bmatrix} 15 & 0 \\ 0 & 0 \end{bmatrix}$$

where δt is the change in time and θ is the orientation of the corresponding vehicle.

CHAPTER 6

DISCUSSION: COMPARISONS, CONTRIBUTIONS, AND LIMITATIONS

In this section, we compare and contrast, first, the performance of the two presented controllers for the two described scenarios. A summary of the results is tabulated for each scenario and then a description of the results is provided, followingly. Secondly, a comparison with other methodologies suggested in literature is made. The first scenario, being more relatable to the scenarios considered from the selected literature, is utilized for the comparisons made with other methodologies.

6.1. Comparisons

Table 6.1: General comparison of results obtained from the first scenario

	Method	**PID**	**LQR**
Scenario 1	Max. Overshoot(m)	1.81	2.73
	Max. Undershoot(m)	-1.90	-0.62

It can be noted from Table 6.1 that during the first scenario, LQR showed the overall maximum overshoot while PID exhibited the overall maximum undershoot. The negative signs on the undershoot show that the F vehiles were closer to the corresponding preceding vehicles than desired.

Table 6.2: General comparison of the results obtained from the second scenario

	Method	**PID**	**LQR**
Scenario 2	Max. Overshoot(m)	3.30	2.86
	Max. Undershoot(m)	-4.50	-1.11

It is observed from Table 6.2 that during the second scenario, PID control yielded the both the maximum overshoot and undershoot throughout the entire simulation.

It can be observed from the results provided that all the algorithms guarantee the following effect in the platoon while maintaining the desired inter-vehicle distance since $E_i \leq E_{thresh}$ almost everywhere in all scenarios under the control of both LQR and PID algorithms. During the first scenario, it can be observed from Figure 5.4 and Figure 5.5 that LQR control exhibits relatively more transient oscillations before it settles down than PID results in Figure 5.1 and Figure 5.2.

Furthermore, following from the definition of string stability provided by (Öncü 2014), string stability is guaranteed in this study since overshoots do not occur and amplify as signals propagate up the string. Ignoring the initial errors culminating from initial conditions, string stability can be proven using equation (4.5) almost everywhere, in the case of PID and at steady-state in the case of LQR – right after the initial transient response.

Table 6.3: Comparison of results with other methodologies suggested in existing literature

Approach	Max-transient error(m)	Steady-state error(m)	Communication criteria
Proposed (our) PID-approach	1.90	0	Local and global
Proposed LQR approach	2.73	0	Local and global
(Ali, Garcia, and Martinet 2013)	~3.2	~2.5	Local and global

Table 6.3 presents the comparison of our controller's performance with that of a different approach proposed by (Ali, Garcia, and Martinet 2013). They suggest a variable inter-vehicle distance control approach where they define the time headway term proportional to the difference between the velocity of the vehicle, and an additional term, hereafter, referred to as ρ, which is shared by all platoon members. Thus, the spacing error is evaluated. The additional term, ρ, is also used as a control parameter for their modified constant time headway (CTH) algorithm. Ali et al.'s control approach uses both local and global control. In Table 6.3, local control is represented by l, whereas global control is represented by g. In local control, data from neighboring platoon members are utilized, whereas in global control, data from, at the very least, the LV is necessary. We use global and distributed control in our approaches. To evaluate the performance of their approach, they vary the velocity of

the LV three times, as well, to V1′ ≈ 14 m/s, V2′ ≈ 19 m/s, and V3′ ≈ 16 m/s. They perform simulations with both MATLAB and TORCS. MATLAB is used to represent a perfect world, and TORCS, a 3D, simulation environment, is used to represent a rather more realistic environment since it provides more features. We utilize the results obtained using the rather more realistic TORCS environment in the comparison provided in Table 6.3.

(Long, Tian, and Cheng 2020) proposed a distributed model predictive model for the longitudinal control of truck platoons. Their model considers the state of the LV. In their study, platoon members transition from cruise control (CC), adaptive cruise control (ACC), and cooperative adaptive cruise control (CACC). ACC and CACC constitute the most advanced platoon controllers that exist today. They use MATLAB to evaluate the performance of their model.

We present a summarized comparison of the performance of our approaches and the ACC phase of their study. Throughout the simulations, (Long, Tian, and Cheng 2020) had the LV accelerate to ≈16 m/s, then move with constant velocity after that. They experienced a maximum transient response error of ≈37 m before the system finally settled and had the range error converge to 0 m. Our proposed controllers i.e., PID and LQR, had overall maximum absolute transient errors of 1.90m and 2.73m respectively, after which they converged to 0m at steady state.

6.2. Contribution

In this work, computationally less demanding and scaling approaches to aid realize the control of longitudinal inter-vehicle distance in AVPS that only utilize the vehicles' onboard GPS sensors and a connection to Wi-Fi are proposed. We trade-off accuracy and precision for cost – financial and computational – and ease of setup. There are two major categories of platoon control: centralized (global) and decentralized (local). In the centralized control, a single management unit is responsible for the main data processing, control calculations and the transmission of control commands to the platoon members. Under decentralized control, each individual unit in the platoon handles its own data processing and runs its control algorithm using its computational resources. The proposed approaches are relatively more computationally cost-efficient because the controllers presented are all decentralized and run onboard each AV within the platoon. That is, each vehicle handles its own processing of the data retrieved from the GPS sensor, extracting the relevant position information and forwarding it to the corresponding control algorithm, unlike in approaches based on centralized control as presented in (Milutinović and Lima 2006), for instance. In the approaches presented, no matter how large (in terms of number of platoon members) the platoon becomes, data processing and running the control algorithm costs remain constant unlike centralized approaches. These approaches thus scale with platoon size. That is, data processing requirements do not change as the number of vehicles in the platoon grows unlike in centralized setups for instance methods proposed in (Milutinović and Lima 2006; Sacone et al. 2021) where relatively more powerful computation resources are required at the main processing unit.

Additionally, the proposed approaches are financially cost-efficient and relatively simpler to implement yet providing considerably good real-time performance. All that is required for this algorithm is a single GPS sensor per platoon member for data acquisition. This is relatively much simpler to setup and configure in comparison to sensor fusion approaches that require a minimum of two sensors (e.g., GPS and IMU) such as the one presented by (Sukkarieh, Nebot, and Durrant-Whyte 1999). Setting up, calibrating, and configuring sensors in a sensor fusion system is relatively more complex, financially, and computationally more resource demanding as well.

All inter-vehicle distance information is extracted from measurements taken by the vehicle's onboard GPS sensors, thereby reducing data acquisition and computation resources. Thus, extra data acquisition and computational resources that would be used for longitudinal inter-vehicle gap control can be allocated to other, relatively more demanding platoon applications. Furthermore, this work differs from those that only perform numeric simulations such as (Ali Memon, Jumani, and Larik 2012; Heydari and Balakrishnan 2012) by not only using generated GPS data but also by applying the algorithm to the 3D vehicle models of the platoon simulated using ROS and the Gazebo platform to mimic the real world as much as possible.

6.3. Limitations

The approach presented in this study does have limitations. The first limitation of this algorithm stems from the fact that it is mainly based on GPS sensors. GPS, in reality, is affected by high-frequency faults culminating from multipath errors that occur when signals bounce off surfaces before they can reach the sensor receivers. The position fix, therefore, gets affected as the signals are delayed. Another rarer cause of GPS faults happens when one of the satellites used by the sensor receiver gets blocked and, as a result, has to be compensated by signals received from a different satellite. The position fix received by the GPS sensor is affected by the geometry of the satellites from which the sensor gets signals. So, such changes in configurations of the satellite observed by the sensor receiver affect the position fix finally reported by the GPS receiver.

High-frequency faults and multipath make the accuracy of GPS sensor, and ultimately, the efficiency of our algorithm heavily environment-dependent, making it more accurate and preferable in open space areas than in underground passages, enclosed environments, or places with tall buildings such as skyscrapers. The algorithm can be incorporated into indoor environments or closed environments by replacing the GPS technology with higher precision localization tools and, or sensors such as beacon technology illustrated by (Siegwart and Nourbakhsh 2004; Surian et al. 2019). We meticulously analyzed the impact of factors such as delays in V2V communication, delays in signal propagation from the PID controllers to the vehicle acuators on the performance of the proposed approach. We also analyzed the impact of GPS sensor lags and the random abrupt acceleration or deceleration of the LV due to miscellaneous factors such as having to stop at the traffic lights, inaccuracies resulting from following a human controlled vehicle and the like in (Gunagwera and Zengin 2022).

Autonomous vehicle platooning applications in urban areas involve high precision dependent maneuvers that require about 0.02 m accuracy to guarantee safety, among other requirements - such as lane-keeping/changing on busy streets, overtaking operations, to mention but a few. In such applications, a 0.5m error is pretty significant. This limitation will be rectified in our future works, as well.

CHAPTER 7

CONCLUSION

At the beginning of this study, a brief review of autonomous vehicles and autonomous vehicle platoons is made. Thorough definition of the major terms in and surrounding the field of AVPs is presented followed by the current state-of-the-art. Factors paramount to the successful performance of AVPs are reviewed. These factors range from efficient communication among platoon members, energy efficiency of the autonomous vehicle platoon as a whole, to – but not limited to – safety guarantee by the platoon to both the platoon members and the surroundings. Furthermore, current progress towards the realization of autonomous vehicle platoons, issues and challenges still barring the realization of AVPs in the world today are discussed.

We then present the modeling and control of a cost-efficient autonomous vehicle platoon as a step toward increasing the applicability and operability of the concept of AVPs in both indoor and outdoor intelligent transport systems and cross-disciplinary fields such as mobile robotics, electrical and industrial engineering to mention, but a few. Decentralized (local) algorithms are used to control the longitudinal inter-vehicle distance of an AVP comprising of four members using both PID and LQR controllers. In the presented approaches, each vehicle handles its own data processing and the running of the control algorithms. Data required by the control algorithms is provided by GPS sensors, thus, making the proposed algorithm cost-efficient both financially and computationally. The proposed approach takes as the main input to the PID and LQR controllers, the updated inter-vehicle distance between a follower vehicle and the preceding and/or leader vehicle. This distance is calculated from the data measured and provided by the vehicles' onboard sensors. The controllers return the reference velocity with which the corresponding follower vehicle should move to achieve the desired inter-vehicle distance. 3D simulations using Gazebo and ROS are additionally used to aid the verification and monitoring of the performance of the predefined autonomous vehicle platoon under two main scenarios. The first scenario had the AVP get in motion from an erroneous initial state. The *LV*, accelerated, moved with constant velocity and at long last decelerated to rest. Response of the control algorithms incorporated within the *F* vehicles was then monitored. In the second scenario, we had the *LV*, once again, accelerate with the platoon in an erroneous initial state except this time the velocity of the *LV*, was continuously varied and totally uncertain to the rest of the platoon members. General response and performance of the controllers incorporated within the *F* vehicles was graphed and the results' summary tabulated and discussed in the results section. The platoon control methods applied in this study, utilize both local and global communication, thus, all platoon members are required to have a connection to Wi-Fi.

The proposed approaches ensure maintenance of platoon formation and no collisions among platoon members. Furthermore, after the transient response to the leader vehicle's acceleration, the standard deviation of the inter-vehicle distance error was kept under 7%, when using PID and under 29% when using LQR, of the desired inter-vehicle distance throughout the entire simulation period while using both approaches. The system achieved a *0m* error at the steady state after the initial transient response. Both approaches achieved individual vehicle stability and string stability after the transient response as well. In the first scenario, the PID controller displayed a maximum overshoot of 1.81m while the LQR controller displayed 2.73m as the maximum overshoot. The PID and LQR controllers showed -1.90m and -0.62m as maximum undershoots, respectively. During the second scenario, the maximum overshoot and undershoot of the PID controller were 3.30m and -4.50m respectively while the maximum overshoot and undershoot of the LQR algorithm were 2.86m and -1.11m respectively.

However, the proposed method is mainly suitable for open environments since GPS accuracy is susceptible to high-frequency errors resulting from multipath and collision of GPS signals with surfaces before they reach the receiver. Applicability of the approach can be extended to closed and underground environments if GPS is replaced with high precision localization equipment such as position beacons installed in the target environments.

CHAPTER 8

FUTURE WORK

Lateral control of autonomous vehicle platoons with the aid of the approaches presented in this study is among the immediate future works. Implementation of lateral control alongside a longitudinally controlled autonomous vehicle platoon yields a fully autonomous vehicle platoon system with relatively wider applicability.

Increasing the number of sensors to complement the GPS sensor, thereby providing more data is another topic for our future works. Fusing data from more sensors such as the IMU, LIDAR, RADAR, and Camera with the ultimate aim of improving controller accuracy and, incidentally, enhancing platoon performance is a future work objective of ours.

REFERENCES

Ali, Alan, Gaetan Garcia, and Philippe Martinet. 2013. “Minimizing the Inter-Vehicle Distances of the Time Headway Policy for Urban Platoon Control with Decoupled Longitudinal and Lateral Control.” In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013),* , 1805–10.

Ali Memon, Zeeshan, Saleh Jumani, and Javed Larik. 2012. “Longitudinal Control of a Platoon of Road Vehicles Equipped with Adaptive Cruise Control System.” 31(3): 475–94. http://oaji.net/articles/2016/2712-1454749706.pdf.

Anayor, Chikamso, Weinan Gao, and Adedapo Odekunle. 2018. “Cooperative Adaptive Cruise Control of A Mixture of Human-Driven and Autonomous Vehicles.” *Conference Proceedings - IEEE SOUTHEASTCON* 2018-April: 1–3.

Anderson, Brian D O, and John B Moore. 2007. *Optimal Control: Linear Quadratic Methods*. Courier Corporation.

Angeli, Richard. 2018. “ROS Wiki: URDF/XML/Joint.” http://wiki.ros.org/urdf/XML/joint.

Åström, Karl Johan et al. 2006. 461 ICRA 2009 *Advanced PID Control*. ISA-The Instrumentation, Systems and Automation Society.

Åström, Karl Johan, and Tore Hägglund. 1995. *PID Controllers: Theory, Design, and Tuning*. ISA-The Instrumentation, Systems and Automation Society.

Bartz, Daniel. 2009. “Autonomous Cars Will Make Us Safer.” https://www.wired.com/2009/11/autonomous-cars/.

Bayuwindra, Anggera, Jeroen Ploeg, Erjen Lefeber, and Henk Nijmeijer. 2020. “Combined Longitudinal and Lateral Control of Car-like Vehicle Platooning with Extended Look-Ahead.” *IEEE Transactions on Control Systems Technology* 28(3): 790–803.

Belcher, Scott, Edward Merlis, Justin McNew, and Morgan Wright. 2018. *Roadmap To Vehicle Connectivity*.

Bertsekas, Dimitri. 2012. 1 *Dynamic Programming and Optimal Control: Volume I*. Athena scientific.

———. 2019. *Reinforcement Learning and Optimal Control*. Athena Scientific.

Bertsekas, Dimitri, and Steven E Shreve. 1996. 5 *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific.

Blum, Jeremy, and Azim Eskandarian. 2004. “The Threat of Intelligent Collisions.” *IT professional* 6(1): 24–29.

Bojarski, Mariusz et al. 2016. “End to End Learning for Self-Driving Cars.” *arXiv preprint arXiv:1604.07316*: 1–9. http://arxiv.org/abs/1604.07316.

Boudette, Neal E. 2016. “Autopilot Cited in Death of Chinese Tesla Driver.” *The New*

York Times. https://www.nytimes.com/2016/09/15/business/fatal-tesla-crash-in-china-involved-autopilot-government-tv-says.html (October 15, 2019).

Brandt, Adam C, Edwin A Frame, Robert W Warden, and San Antonio. 2010. TFLRF-404 *SAE J1321 TESTING USING M1083A1 FMTVS INTERIM REPORT by Force Projection Technologies*.

Broggi, Alberto et al. 2013. “Extensive Tests of Autonomous Driving Technologies.” *IEEE Transactions on Intelligent Transportation Systems* 14(3): 1403–15.

Carlson, Dean A, Alain B Haurie, and Arie Leizarowitz. 2012. *Infinite Horizon Optimal Control: Deterministic and Stochastic Systems*. Springer Science \& Business Media.

CENTER, PEW RESEARCH. 2017. *Many Americans Would Be Hesitant to Use Various Automation Technologies*. USA. https://www.pewresearch.org/internet/2017/10/04/automation-in-everyday-life/pi_2017-10-04_automation_0-02/.

Chakraborty, Tanmay, Shingo Yamaguchi, and Soumya Kanti Datta. 2018. “Sensor Fusion and Adaptive Cruise Control for Self Driving Platoon.” *2018 IEEE 7th Global Conference on Consumer Electronics, GCCE 2018* (Gcce): 634–35.

Chang, Chen, and Zou Yuan. 2018. “Combined Longitudinal and Lateral Control of Vehicle Platoons.” *2017 International Conference on Computer Systems, Electronics and Control, ICCSEC 2017*: 848–52.

Chaurasia, Brijesh Kumar, Shekhar Verma, and G. S. Tomar. 2011. “Attacks on Anonymity in VANET.” *Proceedings - 2011 International Conference on Computational Intelligence and Communication Systems, CICN 2011*: 217–21.

Chen, Xiao et al. 2018. “Integration of Low-Cost GNSS and Monocular Cameras for Simultaneous Localization and Mapping.” *Sensors (Switzerland)* 18(7): 1–18.

Chen, Xiaozhi et al. 2015. “3D Object Proposals for Accurate Object Class Detection.” *Advances in Neural Information Processing Systems* 2015-Janua: 424–32.

Chitta, Sachin, Eitan Marder-Eppstein, Wim Meeussen, Vijay Pradeep, Adolfo Rodríguez Tsouroukdissian, et al. 2017. “Ros_control: A Generic and Simple Control Framework for ROS.” *The Journal of Open Source Software* 2(20): 456.

Chitta, Sachin, Eitan Marder-Eppstein, Wim Meeussen, Vijay Pradeep, Adolfo Rodr\'\iguez Tsouroukdissian, et al. 2017. “Ros_control: A Generic and Simple Control Framework for ROS.” *The Journal of Open Source Software* 2(20): 456.

Choi, Youjun Choi Youjun, Daejun Kang Daejun Kang, Sunjong Lee Sunjong Lee, and Youngmin Kim Youngmin Kim. 2009. “The Autonomous Platoon Driving System of the on Line Electric Vehicle.” *2009 Iccas-Sice*: 3423–26.

Claybrook, Joan, and Shaun Kildare. 2018. “Autonomous Vehicles: No Driver… No Regulation?” *Science* 361(6397): 36–37.

Conley, Ken. 2009. “YAML Overview.” http://wiki.ros.org/YAML Overview.

Deng, Qichen. 2016. "A General Simulation Framework for Modeling and Analysis of Heavy-Duty Vehicle Platooning." *IEEE Transactions on Intelligent Transportation Systems* 17(11): 3252–62.

Estopace, Eden. 2019. "First Commercial Deployment of C-V2X Technology Eyed in China by 2020." https://futureiot.tech/first-commercial-deployment-of-c-v2x-technology-eyed-in-china-by-2020/ (October 21, 2019).

Foote, Tully. 2013. "Tf: The Transform Library." In *Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference On,* Open-Source Software workshop, , 1–6.

Fortin, Félix Antoine et al. 2012. "DEAP: Evolutionary Algorithms Made Easy." *Journal of Machine Learning Research* 13: 2171–75.

Freddi, Alessandro, Sauro Longhi, and Andrea Monteriù. 2015. "Nonlinear Decentralized Model Predictive Control for Unmanned Vehicles Moving in Formation." *Information Technology and Control* 44(1): 89–97.

Gallego, Guillermo et al. 2019. "Event-Based Vision: A Survey." *arXiv preprint arXiv:1904.08405*: 1-. http://arxiv.org/abs/1904.08405.

Garg, Divya et al. 2011. "Direct Trajectory Optimization and Costate Estimation of Finite-Horizon and Infinite-Horizon Optimal Control Problems Using a Radau Pseudospectral Method." *Computational Optimization and Applications* 49(2): 335–58.

Gaspar, José, Niall Winters, and José Santos-Victor. 2000. "Vision-Based Navigation and Environmental Representations with an Omnidirectional Camera." *IEEE Transactions on Robotics and Automation* 16(6): 890–98.

Ghasemi, A, R Kazemi, and S Azadi. 2013. "Stable Decentralized Control of a Platoon of Vehicles With Heterogeneous Information Feedback." *IEEE Transactions on Vehicular Technology* 62(9): 4299–4308.

Global, Deloitte. 2019. *2019 Deloitte Global Automotive Consumer Study | Advanced Vehicle Technologies and Multimodal Transportation*. https://www2.deloitte.com/content/dam/Deloitte/us/Documents/manufacturing/us-global-automotive-consumer-study-2019.pdf.

Guizzo, Erico. 2011. "How Google's Self-Driving Car Works." *IEEE Spectrum Online* 18(7): 1132–41.

Gunagwera, Alex, and Aydın Tarık Zengin. 2022. "Longitudinal Inter-Vehicle Distance Control of Autonomous Vehicle Platoons Subjected to Internal and External Disturbances." *Balkan Journal of Electrical and Computer Engineering* 10(1): 75–84.

Hadded, Mohamed, Jean Marc Lasgouttes, Fawzi Nashashibi, and Ilias Xydias. 2018. "Platoon Route Optimization for Picking up Automated Vehicles in an Urban Network." *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC* 2018-Novem: 1987–92.

Hallé, Simon, Julien Laumonier, and Brahim Chaib-draa. 2004. “A Decentralized Approach to Collaborative Driving Coordination.” *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*: 453–58.

Han, Shi Yuan, Yue Hui Chen, Lin Wang, and Ajith Abraham. 2013. “Decentralized Longitudinal Tracking Control for Cooperative Adaptive Cruise Control Systems in a Platoon.” *Proceedings - 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013*: 2013–18.

Heydari, Ali, and Sivasubramanya N Balakrishnan. 2012. “Approximate Closed-Form Solutions to Finite-Horizon Optimal Control of Nonlinear Systems.” In *2012 American Control Conference (ACC),* , 2657–62.

Van De Hoef, Sebastian, Karl Henrik Johansson, and Dimos V. Dimarogonas. 2018. “Fuel-Efficient En Route Formation of Truck Platoons.” *IEEE Transactions on Intelligent Transportation Systems* 19(1): 102–12.

Hu, Junyan et al. 2020. “Cooperative Control of Heterogeneous Connected Vehicle Platoons: An Adaptive Leader-Following Approach.” *IEEE Robotics and Automation Letters* 5(2): 977–84.

Huang, Ren Hung, Ben Jye Chang, Yueh Lin Tsai, and Ying Hsin Liang. 2018. “Mobile Edge Computing-Based Vehicular Cloud of Cooperative Adaptive Driving for Platooning Autonomous Self Driving.” *Proceedings - 2017 IEEE 7th International Symposium on Cloud and Service Computing, SC2 2017* 2018-Janua: 32–39.

Huang, Zichao, Duanfeng Chu, Chaozhong Wu, and Yi He. 2019. “Path Planning and Cooperative Control for Automated Vehicle Platoon Using Hybrid Automata.” *IEEE Transactions on Intelligent Transportation Systems* 20(3): 959–74.

Hyundai. 2019. “Hyundai Motor Group and Aptiv to Form Autonomous Driving Joint Venture.” https://org1-www.hyundai.com/worldwide/en/news/news-room/news/hyundai-motor-group-and-aptiv-to-form-autonomous-driving-joint-venture-0000016315?pageNo=1&searchKey=&rowCount=6&type[]=RES&listPageUrl=news.release.all (October 12, 2019).

Ibrahim, Amr et al. 2018. “Co-Simulation Framework for Control, Communication and Traffic for Vehicle Platoons.” *Proceedings - 21st Euromicro Conference on Digital System Design, DSD 2018*: 352–56.

Institute, European Telecommunications Standards. 2019. 1 *ETSI TR 103 299 V2.1.1 In- Telligent Transport Systems (ITS); Cooperative Adaptive Cruise Control (CACC);* https://cdn.standards.iteh.ai/samples/45854/7b3e6d0d3c014cb995408e4a8a0a87d5/ETSI-TR-103-299-V2-1-1-2019-06-.pdf.

international, S A E. 2016. “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles.” *SAE International,(J3016).*

Ioannou, P, and Z Xu. 1994. “THROTTLE AND BRAKE CONTROL SYSTEMS FOR AUTOMATIC VEHICLE FOLLOWING.” *I V H S Journal* 1(4): 345–77.

https://doi.org/10.1080/10248079408903805.

Jain, Anil K., Jianchang Mao, and K. M. Mohiuddin. 1996. “Artificial Neural Networks: A Tutorial.” *Computer* 29(3): 31–44.

Jain, Monika, and Rahul Saxena. 2017. “Overview of VANET: Requirements and Its Routing Protocols.” In *2017 International Conference on Communication and Signal Processing (ICCSP)*, , 1957–61.

Jain, Varun et al. 2018. “Prediction Based Framework for Vehicle Platooning Using Vehicular Communications.” *IEEE Vehicular Networking Conference, VNC* 2018-Janua: 159–66.

Janssen, Robbert, Han Zwijnenberg, Iris Blankers, and Janiek de Kruijff. 2015. “Truck Platooning.” *Driving The Future of Transportation, TNO*.

Jia, Dongyao et al. 2016. “A Survey on Platoon-Based Vehicular Cyber-Physical Systems.” *IEEE Communications Surveys and Tutorials* 18(1): 263–84.

Jurgen, R. 2011. “Design and Analysis of an Adaptive Real-Time Advisory System for Improving Real World Fuel Economy in a Hybrid Electric Vehicle (2010-01-0835).” In *Electric and Hybrid-Electric Vehicles: Overviews and Viewpoints,* SAE, 35–45. https://ieeexplore.ieee.org/document/8522869.

Karnadi, Feliz Kristianto, Zhi Hai Mo, and Kun Chan Lan. 2007. “Rapid Generation of Realistic Mobility Models for VANET.” *IEEE Wireless Communications and Networking Conference, WCNC*: 2508–13.

Kaur, Rajdeep et al. 2018. “Network (VANET).” *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)* (Icoei): 884–89.

Kavathekar, Pooja, and YangQuan Chen. 2011. “Vehicle Platooning: A Brief Survey and Categorization.” In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, , 829–45.

Kessler, Aaron M. 2015. “Elon Musk Says Self-Driving Tesla Cars Will Be in the Us by Summer.” *The New York Times*: B1. https://www.nytimes.com/2015/03/20/business/elon-musk-says-self-driving-tesla-cars-will-be-in-the-us-by-summer.html?_r=0.

Khaksari, Mohammadreza, and Carlo Fischione. 2012. “Performance Analysis and Optimization of the Joining Protocol for a Platoon of Vehicles.” In *5th International Symposium on Communications Control and Signal Processing, ISCCSP 2012*, IEEE, 1–6.

Kimura, Ryota, Nobutomo Matsunaga, Hiroshi Okajima, and Gou Kotaki. 2017. “Design of Virtual Platoon Control System Using Augmented Reality to Assist Welfare Vehicle Users.” *International Conference on Control, Automation and Systems* 2017-Octob(Iccas): 330–35.

Koenig, Nathan, and Andrew Howard. 2004. “Design and Use Paradigms for Gazebo, an Open-Source Multi-Robot Simulator.” In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No.*

04CH37566), , 2149–54.

Kolb, Julius K., Gunter Nitzsche, and Sebastian Wagner. 2019. “A Simple yet Efficient Path Tracking Controller for Autonomous Trucks.” *IFAC-PapersOnLine* 52(8): 1–6. https://doi.org/10.1016/j.ifacol.2019.08.088.

Koutník, Jan, Giuseppe Cuccu, Jürgen Schmidhuber, and Faustino Gomez. 2013. “Evolving Large-Scale Neural Networks for Vision-Based Reinforcement Learning.” *GECCO 2013 - Proceedings of the 2013 Genetic and Evolutionary Computation Conference*: 1061–68.

Kundu, Sanjiban et al. 2013. “Vehicle Speed Control Algorithms for Eco-Driving.” *2013 International Conference on Connected Vehicles and Expo, ICCVE 2013 - Proceedings* (c): 931–32.

LaValle, Steven M. 2006. *Planning Algorithms*. Cambridge university press.

LeCun, Yann et al. 2005. “Off-Road Obstacle Avoidance through End-to-End Learning.” *Advances in Neural Information Processing Systems*: 739–46.

Levanon, Nadav. 1988. *Radar Principles*. New York: Wiley-Interscience. https://ui.adsabs.harvard.edu/abs/1988wi...book.....L.

Levinson, Jesse et al. 2011. “Towards Fully Autonomous Driving: Systems and Algorithms.” *IEEE Intelligent Vehicles Symposium, Proceedings* (Iv): 163–68.

Li, Yongfu, and Changpeng He. 2018. “Connected Autonomous Vehicle Platoon Control Considering Vehicle Dynamic Information.” *2018 37th Chinese Control Conference (CCC)*: 7834–39.

Liu, Kevin et al. 2008. “Precision Relative Navigation Solution for Autonomous Operations in Close Proximity.” *Record - IEEE PLANS, Position Location and Navigation Symposium*: 1246–51.

Long, Mohan, Guangfeng Tian, and Hao Cheng. 2020. “Longitudinal Control for Truck Platooning.” In *2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI)*, , 418–23.

mann-Wellenhof, Bernhard and Lichtenegger, Herbert and Wasle, Elmar. 2007. *GNSS-Global Navigation Satellite Systems: GPS, GLONASS, Galileo, and More*. Springer Science and Business Media.

Marino, Alessandro et al. 2009. “A Fault-Tolerant Modular Control Approach to Multi-Robot Perimeter Patrol.” *2009 IEEE International Conference on Robotics and Biomimetics, ROBIO 2009*: 735–40.

Marques, Luís, António Casimiro, and Mário Calha. 2009. “Design and Development of a Proof-of-Concept Platooning Application Using the HIDENETS Architecture.” *Proceedings of the International Conference on Dependable Systems and Networks*: 223–28.

McAllister, Rowan et al. 2017. “Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning.” *IJCAI International Joint Conference* on *Artificial Intelligence*: 4745–53.

McArdle, Megan. 2018. "How Safe Are Driverless Cars? Unfortunately, It's Too Soon to Tell." *The Washington Post*. https://www.washingtonpost.com/opinions/no-driverless-cars-arent-far-safer-than-human-drivers/2018/03/20/5dc77f42-2ba9-11e8-8ad6-fbc50284fce8_story.html#comments-wrapper.

Mei, Jie et al. 2018. "Joint Radio Resource Allocation and Control for Vehicle Platooning in Lte-V2v Network." *IEEE Transactions on Vehicular Technology* 67(12): 12218–30.

Mena-Oreja, Jesús, Javier Gozalvez, and Miguel Sepulcre. 2019. "Effect of the Configuration of Platooning Maneuvers on the Traffic Flow under Mixed Traffic Scenarios." *IEEE Vehicular Networking Conference, VNC* 2018-Decem: 0–3.

Milano, Politecnico D I. 2018. "A Study of an LQR Drifting Controller for RC Cars."

Miller, Ben. 2015. "First Driverless Car Crash Study: Autonomous Vehicles Crash More, but Injuries Are Less Serious." *Government Technology*. https://www.govtech.com/fs/First-Driverless-Car-Crash-Study-Autonomous-Vehicles-Crash-More-Injuries-Are-Less-Serious.html (October 15, 2019).

Milutinović, Dejan, and Pedro Lima. 2006. "Modeling and Optimal Centralized Control of a Large-Size Robotic Population." *IEEE Transactions on Robotics* 22(6): 1280–85.

Naus, Gerrit et al. 2010. "Cooperative Adaptive Cruise Control, Design and Experiments." *Proceedings of the 2010 American Control Conference, ACC 2010* (1): 6145–50.

Nemeth, Balazs, Alfred Csikos, Istvan Varga, and Peter Gaspar. 2012. "Road Inclinations and Emissions in Platoon Control via Multi-Criteria Optimization." *2012 20th Mediterranean Conference on Control and Automation, MED 2012 - Conference Proceedings*: 1524–29.

Newman, Wyatt. 2017. *A Systematic Approach to Learning Robot Programming with ROS*. CRC Press.

Ogata, Katsuhiko. 2010. *Modern Control Engineering*. 5th ed. Prentice hall Upper Saddle River, NJ.

Omar, Halima Mansour, Zhang Yanzhong, Zhang Bo, and Haris Ubaid Gul. 2016. "Integration of GPS and Dead Reckoning Navigation System Using Moving Horizon Estimation." In *Proceedings of 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2016*, IEEE, 553–56.

Öncü, S. 2014. *String Stability of Interconnected Vehicles : Network- Aware Modelling, Analysis and Experiments*. Technische Universiteit Eindhoven.

Oncu, Sinan, Nathan van de Wouw, W. P. Maurice H. Heemels, and Henk Nijmeijer. 2013. "String Stability of Interconnected Vehicles under Communication Constraints." *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*:

2459–64.

Qi, Yinan, and Tomasz MacH. 2019. “QoS and Coverage Aware Dynamic High Density Vehicle Platooning (HDVP).” In *IEEE Vehicular Technology Conference*, IEEE, 1–5.

Quigley, M. 2009. “ROS: An Open-Source Robot Operating System.” In *ICRA 2009*,.

Rajamani, Rajesh. 2011. *Vehicle Dynamics and Control*. Springer Science \& Business Media.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. “You Only Look Once: Unified, Real-Time Object Detection.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, , 779–88.

Redondo, Jonatan Pajares et al. 2018. “VEHIOT: Design and Evaluation of an IoT Architecture Based on Low-Cost Devices to Be Embedded in Production Vehicles.” *Sensors (Switzerland)* 18(2).

Reutebuch, Stephen E, Hans-erik Andersen, and Robert J Mcgaughey. 2005. “Tec Ti on and Ra Ngin g (LI DAR) : An Tmerg I Ng Res Lource Inve Ntory.” *Journal of Forestry* 103(6): 286-292.

Reutebuch, Stephen E, Hans-Erik Andersen, and Robert J McGaughey. 2005. “Light Detection and Ranging (LIDAR): An Emerging Tool for Multiple Resource Inventory.” *Journal of Forestry* 103(6): 286–92.

Richtel, Matt, and Conor Dougherty. 2015. “Google’s Driverless Cars Run into Problem: Cars with Drivers.” *New York Times*: n.p. https://www.eila.univ-paris-diderot.fr/_media/user/claire_dupuy/google_cars_text.pdf%0Ahttp://www.nytimes.com/2015/09/02/technology/personaltech/google-says-its-not-the-driverless-cars-fault-its-other-drivers.html?hp&action=click&pgtype=Homepage&module=mi.

Ross, Philip E. 2019. “Taxis without Drivers-or Steering Wheels.” *IEEE Spectrum* 56(1): 33–34.

Ruane, Laura. 2018. “SAE International Holds Public Self-Driving SUV Demo at Babcock Ranch near Fort Myers.” *Fort Myers News Press*. https://www.news-press.com/story/news/2018/12/13/public-self-driving-car-demo-comes-babcock-ranch-east-fort-myers/2263328002/.

Sacone, Simona, Cecilia Pasquale, Silvia Siri, and Antonella Ferrara. 2021. “Centralized and Decentralized Schemes for Platoon Control in Freeway Traffic Systems.” In *2021 60th IEEE Conference on Decision and Control (CDC)*, , 2665–70.

SAE, Taxonomy. 2013. “Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems-J3016.” *Society of Automotive Engineers: On-Road Automated Vehicle Standards Committee; SAE Pub. Inc., Warrendale, PA, USA*.

El Sallab, Ahmad, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. 2017. “Deep Reinforcement Learning Framework for Autonomous Driving.” *IS and T*

International Symposium on Electronic Imaging Science and Technology: 70–76.

SARTRE Consortium. 2012. “Safe Road Trains for the Environment (SARTRE).” https://www.roadtraffic-technology.com/projects/the-sartre-project/ (October 12, 2019).

Seiler, P, A Pant, and K Hedrick. 2004. “Disturbance Propagation in Vehicle Strings.” *IEEE Transactions on Automatic Control* 49(10): 1835–42.

Shapiro, Danny. 2017. “Self-Driving Car Demo Like No Other: The Road to an Unforgettable Cup of Coffee.” *Nvidia Blog*. https://blogs.nvidia.com/blog/2017/01/04/bb8-ces/ (October 14, 2019).

Sharon L Poczter, and Luka M. Jankovic. 2014. “The Google Car: Driving Toward A Better Future?” *Journal of Business Case Studies (JBCS)* 10(1): 7–14.

Shen, Zhongyi et al. 2018. “Performance Analysis of Extended Sensor Sharing in Vehicular Ad Hoc Networks.” *2018 International Symposium on Antennas and Propagation (ISAP)*: 1–2.

Shladover, Steven E., Christopher Nowakowski, Xiao Yun Lu, and Robert Ferlis. 2015. “Cooperative Adaptive Cruise Control: Definitions and Operating Concepts.” *Transportation Research Record* 2489(November 2014): 145–52.

Shuttleworth, Jennifer. 2019. “SAE Standards News: J3016 Automated-Driving Graphic Update.” *SAE International*.

Siegwart, Roland, and Illah R Nourbakhsh. 2004. *Introduction to Autonomous Mobile Robots*. USA: Bradford Company.

Singh, Surmukh, and Sunil Agrawal. 2014. “VANET Routing Protocols: Issues and Challenges.” In *2014 Recent Advances in Engineering and Computational Sciences (RAECS)*, , 1–5.

Sivaji, V V, and M Sailaja. 2013. “Adaptive Cruise Control Systems for Vehicle Modeling Using Stop and Go Manoeuvres.” *International Journal of Engineering Research and Applications* 3(4): 2453–56. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.433.9908&rep=rep1&type=pdf.

Smallwood, Richard D, and Edward J Sondik. 1973. “The Optimal Control of Partially Observable Markov Processes over a Finite Horizon.” *Operations research* 21(5): 1071–88.

Stewart, Jack. 2018a. “Tesla’s Autopilot Was Involved in Another Deadly Car Crash.” *Transportation, Wired*. https://www.wired.com/story/tesla-autopilot-self-driving-crash-california/ (October 25, 2019).

———. 2018b. “Why People Keep Rear-Ending Self-Driving Cars.” *Wired*. https://www.wired.com/story/self-driving-car-crashes-rear-endings-why-charts-statistics/ (September 15, 2019).

Stilwell, Daniel J, and Bradley E Bishop. 2001. *Decentralized Guidance , Navigation*

, and Control for Platoons of Cooperating UUVs.

Su, Dongliang, and Sanghyun Ahn. 2016a. "Autonomous Platoon Formation for VANET-Enabled Vehicles." *2016 International Conference on Information and Communication Technology Convergence, ICTC 2016*: 247–50.

———. 2016b. "Autonomous Platoon Formation for VANET-Enabled Vehicles." *2016 International Conference on Information and Communication Technology Convergence, ICTC 2016*: 247–50.

Sugano, Tatsuya, Hiroshi Okajima, and Nobutomo Matsunaga. 2015. "Robust and Precise Platoon Driving Control of Welfare Vehicles along Wheel Track by Using Model Error Compensator." *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*: 256–61.

Sugano, Tatsuya, Hiroshi Okajima, Satoshi Samura, and Nobutomo Matsunaga. 2016. "Narrow Space Driving of Welfare Vehicles Using Robust Platoon Control with Adaptive Way Point Tracking." *2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan, SICE 2016*: 1436–41.

Sukkarieh, Salah, Eduardo M Nebot, and Hugh F Durrant-Whyte. 1999. "A High Integrity IMU/GPS Navigation Loop for Autonomous Land Vehicle Applications." *IEEE Transactions on Robotics and Automation* 15(3): 572–78.

Surian, Didi et al. 2019. "Tracking a Moving User in Indoor Environments Using Bluetooth Low Energy Beacons." *Journal of Biomedical Informatics* 98: 103288. https://www.sciencedirect.com/science/article/pii/S1532046419302072.

Swaroop, DVAHG, and J Karl Hedrick. 1999. "Constant Spacing Strategies for Platooning in Automated Highway Systems."

Szalay, Zsolt et al. 2018. "Development of a Test Track for Driverless Cars: Vehicle Design, Track Configuration, and Liability Considerations." *Periodica Polytechnica Transportation Engineering* 46(1): 29–35.

Team-Waymo. 2019. "WAYMO." https://waymo.com/mission/ (October 11, 2019).

Team, ARGO AI. 2019. "ARGO AI." https://www.argo.ai/ (October 15, 2019).

Tong, Wang, Azhar Hussain, Wang Xi Bo, and Sabita Maharjan. 2019. "Artificial Intelligence for Vehicle-to-Everything: A Survey." *IEEE Access* 7: 10823–43.

Tonguz, Ozan et al. 2007. "Broadcasting in VANET." *2007 Mobile Networking for Vehicular Environments, MOVE*: 7–12.

Tseng, Yu-ho, and Shau-shiun Jan. 2018. "Combination of Computer Vision Detection and Segmentation for Autonomous Driving." : 1047–52.

Tsugawa, S. et al. 2001. "A Cooperative Driving System with Automated Vehicles and Inter-Vehicle Communications in Demo 2000." *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*: 918–23.

Ucar, Seyhan, Sinem Coleri Ergen, and Oznur Ozkasap. 2017. "Security Vulnerabilities of Autonomous Platoons." : 1–4.

United Nations D. o. E. a. SA, Population Division. 2017. "World Population Prospects: The 2017 Revision, Key Findings and Advance Tables"." *The Department of Economic and Social Affairs: UN*.

Vahidi, Ardalan, and Azim Eskandarian. 2003. "Research Advances in Intelligent Collision Avoidance and Adaptive Cruise Control." *IEEE Transactions on Intelligent Transportation Systems* 4(3): 132–53.

Visnic, Bill. 2018. "SAE Brings Autonomy to the Public in New Demonstration Program." *SAE International*. https://www.sae.org/news/2018/05/sae-demo-days-in-florida.

Wang, Ziran et al. 2018. "A Review on Cooperative Adaptive Cruise Control (CACC) Systems." *arXiv preprint arXiv:1809.02867*: 2884–91. https://arxiv.org/ftp/arxiv/papers/1809/1809.02867.pdf.

Willke, Theodore L, Patcharinee Tientrakool, and Nicholas F Maxemchuk. 2009. "A Survey of Inter-Vehicle Communication Protocols and Their Applications." *IEEE Communications Surveys Tutorials* 11(2): 3–20.

Worldometer. 2019. "Annual Global Car Production." : 1. https://www.worldometers.info/cars/.

Wu, Dapeng, Jiaxing Wu, and Ruyan Wang. 2019. "An Energy-Efficient and Trust-Based Formation Algorithm for Cooperative Vehicle Platooning." *2019 International Conference on Computing, Networking and Communications, ICNC 2019*: 702–7.

Yu, Xiaohai, Ge Guo, and Hongbo Lei. 2018. "Longitudinal Cooperative Control for a Bidirectional Platoon of Vehicles with Constant Time Headway Policy." *Proceedings of the 30th Chinese Control and Decision Conference, CCDC 2018*: 2427–32.

Zaher, Madeleine El and Gechter, Franck and Hajjar, Mohammad and Gruer, Pablo. 2016. "An Interaction Model for a Local Approach to Vehicle Platoons." *International Journal of Vehicle Autonomous Systems* 13(2): 91–113. https://www.inderscienceonline.com/doi/pdf/10.1504/IJVAS.2016.078760.

Zhang, Congchi, Yunpeng Zang, Jose Angel Leon Calvo, and Rudolf Mathar. 2017. "A Novel V2V Assisted Platooning System: Control Scheme and MAC Layer Designs." In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, , 1–7.

———. 2018. "A Novel V2V Assisted Platooning System: Control Scheme and MAC Layer Designs." *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC* 2017-Octob: 1–7.

Zhao, Jin, Other Ecole, and Centrale De Lille. 2011. "To Cite This Version : Contribution à La Commande d ' Un Train de Véhicules Intelligents."

Zhao, Siyang et al. 2015. "Vehicle to Vehicle Communication and Platooning for EV with Wireless Sensor Network." *2015 54th Annual Conference of the Society of Instrument and Control Engineers of Japan, SICE 2015*: 1435–40.

Zheng, Yang et al. 2014. “Influence of Information Flow Topology on Closed-Loop Stability of Vehicle Platoon with Rigid Formation.” *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*: 2094–2100.

Ziegler, Julius et al. 2014. “Making Bertha Drive-an Autonomous Journey on a Historic Route.” *IEEE Intelligent Transportation Systems Magazine* 6(2): 8–20.

Zou, Cunming, and Hongxing Li. 2019. “Event-Driven Connected Vehicular Platoon Control with Mixed Time-Varying Delay.” *IEEE Access* PP: 1–1. https://ieeexplore.ieee.org/document/8758226/.

CURRICULUM VITAE

A. PARTICULAR DETAILS

Alex Gunagwera

Adres: Istanbul Sabahattin Zaim University, Istanbul, Turkey

Contact:

B. EDUCATION UPDATE

Master Degree: Istanbul Sabahattin Zaim University, 2017

C. LANGUAGES:

English: Official language/Very good (Fluent),

Turkish: Good (Fluent)

D. PUBLICATIONS

Cevik, T., Gunagwera, A. and Cevik, N., 2015. A survey of multimedia streaming in wireless sensor networks: progress, issues and design challenges. *arXiv preprint arXiv:1512.03565*.

Cevik T, Gunagwera A. A directional multicasting-based architecture for wireless sensor networks. *International Journal of Electronics*. 2019 Oct 3;106(10):1441-62.

Cevik T, Gunagwera A. Coverage and energy efficiency optimization for randomly deployed multitier wireless multimedia Sensor Networks. *International Journal of Communication Networks and Information Security*. 2018 Apr 1;*10(1)*:28-36.

Gunagwera, A. and Kiani, F., 2019. Ultimate indoor navigation: a low cost indoor positioning and intelligent path finding,

Gunagwera, A., & ZENGİN, A. T. (2022). Longitudinal inter-vehicle distance control of autonomous vehicle platoons subjected to internal and external disturbances. *Balkan Journal of Electrical and Computer Engineering*, 10(1), 75-84.

Gunagwera, A., & Zengin, A. T. (2022). A longitudinal inter-vehicle distance controller application for autonomous vehicle platoons. *PeerJ Computer Science*, 8, e990.

E. ACCOMPLISHMENTS

- A couple of high honor and honor certificates for academic success awarded every end of academic term (semester).

- Certificates of completion in; Web Engineering (With highest distinction on a MOOC course), Microprocessors and embedded systems, programming, mathematics, Deeplearning etc.

- Medals in football, chess (with certificate) and table tennis.

F. Experience and Major Projects

- Metaheuristic Optimization and natural, clever algorithms

- Some major projects (not in class) include:
  - Database optimization
  - Quad-copter development project.

- Worked on a project to make a quadcopter (I was assigned to the algorithm development section).

- **Worked** in a team on a project. Department: Research and Development team (R&D) – internship.

  - Public speaking

- Data recovery
- Teaching (Programming, Mathematics, English(privatelessons) and Physics)
- Academic writing
- A couple of in and out-course assigned projects, including ecommerce, web development…
- Freelance projects (e-commerce websites and web, and mobile application development)
- Paper Based Examination Typist and printer manager in high school.
- Currently ongoing/Recent projects:
    - Multi-tier randomly deployed sensor Network energy optimization and analysis with intensive simulation (Also part of thesis for the Master's programme.) – finished.
    - Multi-cast routing with Multi-beam directional antennas. Software development (Purely JAVA and MATLAB). – Finished.
    - Mobile navigation system development (Android). Writing final report – Finished.
    - Built a few simple java games such as: Tetris, car race, etc. (practice)
    - Big data and data analysis using python, R, and/or MATLAB with sample cases on analysing the trend in the number of foreign students in Turkey – prior to the corona pandemic.
    - Deep and Machine learning: with sample practical cases in consumer preferences in logistic patterns – Finished.
    - Control theory and applications: ongoing research with theoretical and practical sample applications on the control of autonomous land vehicle platoons, mechanical components etc. – ongoing

G. SKILLS AND TECHNICAL EXPERTISE

- ***Programming Languages and skills***:
    - Java (Proficient)
    - Latex(Intermediate – prior experience)
    - MATLAB (Proficient)
    - PHP (Proficient with prior experience)
    - Database Systems (MySQL) – Proficient
    - Web development (Prior Experience)
    - Racket (intermediate)
    - Python (Proficient)
    - HTML, CSS, Javascript (Proficient)
    - Android Development (Prior Experience)
    - Micro-controllers and embedded systems (Prior Experience).
- ***Technologies And Frameworks***
    - Robotics Operating System (ROS). (Proficient with experience).
    - JQuery, Angular, Vue, Bootstrap, etc. (Proficient).
    - Stack J2EE (JSP, EJB, JMS, JPA) (Prior Experience)
    - Visual Studio (Prior experience/comfortable)
    - MapReduce (Beginner)
- ***Operating Systems***
    - Windows(Proficient with experience)
    - Linux (Proficient with experience)
    - Android
- **VCSs**
    - Git (Proficient)