Bachelor Thesis

Advanced Driver Assistance System for Bike Detection using Sensor Fusion

Mostafa Hussein Shaaban

Supervisor:
Prof. Dr. rer. nat. Toralf Trautmann

A thesis submitted in fulfilment of the requirements for the degree of Bachelor of Science

in the

Mechatronics Engineering
German University in Cairo

Dresden, August 2014

Dedication

This paper is dedicated to my mother Maha, my father Hussein and my best friend and brother Omar. Actually, all of my work which is done to write my thesis would never have been achieved without Allah's blessings and those people. Their efforts everyday and prayers were a great aid for me to be able to reach what I have reached now. They were always beside me in every obstacle I face during my life, and if I tried to show them how thankful I am, I will never be able to do it.

Another ultimate thanks for my aunts who have been great supporters to me during my life; giving me help, support and advices during my entire educational path. Moreover, I dedicate this also to my lovely deceased grandmother who has always assisted me with her motivations and prayers, and was always telling me: *"Mostafa, you will be the best ever"*, and now I am putting my first steps on the successful life that she always wished me to reach it, and proving that I can be the best.

Last but not least, my final words in this dedication go to my life partner Haidi, who stood beside me and has always been my backbone; she gave me great support to pass any phase in my life and reach any achievement.

Acknowledgement

It has been a great experience working on a subject such as Advanced Driver Assistance Safety System. I am indebted to many people who have influenced and inspired me in my project. Their enthusiasm, help, and support have ultimately led to the completion of this dissertation.

A special thanks goes to my supervisor, Prof. Toralf Trautmann for giving me such opportunity to work on this project in his lab, taking care of my work and continuous support with any questions. He was always pointing me with his suggestions to the right direction.

I am grateful to Dipl.-Ing.(FH) Erik Unger for his guidance and sincere support all throughout the project, and helping me a lot with his great assistance. It has been a pleasure working in such a distinctive project with those people.

At this point I would like to thank all the staff of the Laboratory of Automotive Mechatronics which I have worked in during 6 months, and thanks for HTW-Dresden and the Centre for Applied Research and Technology at Dresden University of Applied Sciences (ZAFT e.V.) for their helpfulness and great working atmosphere.

I would like to express my particular gratitude and deep appreciation to the German University in Cairo for guidance and facilities. The great honour goes to Prof. Dr. Elsayed Ibrahim Imam Morgan, Head of Mechatronics Engineering Department and my internal supervisor whom I had learned from him a lot. And of course I cannot forget to thank all the professors who have changed a lot in my life and gave me great support in my educational life in the university, so these thanks go especially to Prof. Dr. Yasser Higazi, Dean of Faculty of Information Engineering & Technology, Prof. Dr. Ayman Elbadawy, and my beloved deceased professors, Prof. Dr. Mustafa Amer. and Dr. Amin Selim who have taught me a lot.

Final thanks to all my family members and friends who supported me during this period of work.

Declaration of Authorship

I, Mostafa Hussein, hereby declare that this thesis titled, 'Advanced Driver Assistance System for Bike Detection using Sensor Fusion' is my own work and effort and that it has not been submitted anywhere for any award. Where other sources of information have been used, they have been acknowledged.

This thesis is a presentation of my original research work. Wherever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature, and acknowledgement of collaborative research and discussions.

The work was done under the guidance of Professor Toralf Trautmann, at the Laboratory of Automotive Mechatronics, HTW-Dresden.

B. Sc. Mostafa Hussein Shaaban Hussein

In my capacity as supervisor of the candidate's thesis, I certify that the above statements are true to the best of my knowledge.

Prof. Dr. rer. nat. Toralf Trautmann

Dresden, August 31, 2014

GERMAN UNIVERSITY IN CAIRO

Abstract

Engineering and Material Sciences
Mechatronics Engineering

Bachelor of Science Degree

Advanced Driver Assistance System for Bike Detection using Sensor Fusion

by Mostafa Hussein Shaaban

Advanced Driver Assistance System or ADAS are systems used to help the driver in the driving process. When designed with a safe Human-Machine interface, it should increase car safety and more generally road safety. Such systems were trying to play a great role in achieving the top safety for the driver. And according to many statistics and researches done on accidents that happen due to collision between cyclists and cars, my main project is to work on detecting the critical situations for the driver, thus informing him/her about the accident that may happen. Using special sensors in the car itself and by calculating the car dynamics, in addition to using data from the smartphone of the cyclist; one can be able to know the critical situation for the car. Fusion occurs between different sensors to detect this situation; so in this project, radar and GPS data were fused together to detect the case where the cyclist is about to hit the car, thus we can give out a warning. iBeacon was studied and analyzed for the usage in the fusion, but there were some problems in the initialization, so it was not used. The system can prevent accidents between bicycles and cars, especially in the case of door opening, intersections or blindspot. So the future collisions between cyclists and motor vehicles at cornering situations in the crossing areas are to be prevented. Because of their narrow silhouette and often high speed, cyclists are often overlooked by-turning motor vehicles. A driver assistance system for cyclists' detection can reduce this risk. The results throughout the project showed that such a system can work perfectly but with some errors fixation, as it was proved that GPS is not considered a reliable tool because it gives a range of accuracy error. So finally, two systems were created as simulation; an ideal system and a real system which contains the errors from the GPS and the radar. Afterwards, some technical details were added to the real system to be able to overcome the errors and reach the same output as the ideal one.

Contents

Page

Dedication i
Acknowledgement ii
Declaration of Authorship iii
Abstract iv
Table of Contents v
List of Figures ix
List of Tables xiii
Lists of Equations xv
Acronyms xvi
Nomenclature xx

1. Introduction **1**
1.1 Overview 1
1.2 Theoretical Background 4
1.3 Problem Statement 6
1.4 Literature Review 7
1.5 Motivation 10
1.6 Methodology 10

2. Basic Tools and Methods **12**
2.1 Serial Bus Systems 12
2.1.1 Main Idea 12
2.1.2 Communication 13
2.1.3 ISO/OSI Model 14
2.1.4 Three Layer Model 16
2.1.5 Architecture of Serial Bus Systems - Topologies 17
2.1.6 Protocols 18
2.2 Controller Area Network (CAN) 19
2.2.1 Introduction 19
2.2.2 CAN advantages over conventional wiring 20
2.2.3 CAN Classes 20
2.2.4 Communication 21
2.2.5 Data Protocol 29
2.2.6 Signal Analysis 31
2.3 Test Vehicle Citroën C6 32

2.3.1 Overview . . . 33
2.3.2 CAN connections . . . 33
2.4 Radar Sensor . . . 34
2.4.1 Introduction . . . 34
2.4.2 Short Range Radar . . . 35
2.4.3 Ultra Wide Band UWB . . . 36
2.4.4 Principle of Operation . . . 37
2.5 Global Positioning System (GPS) . . . 42
2.5.1 Basic Structure and System Overview . . . 42
2.5.2 Theory of Operation . . . 43
2.6 iBeacon . . . 45
2.6.1 Introduction . . . 45
2.6.2 Theory of Operation . . . 45
2.6.3 Signals . . . 46
2.7 FlexDevel Controller . . . 47
2.7.1 Introduction . . . 47

3. Experimental Work **49**
3.1 CAN Messages . . . 49
3.1.1 CANalyzer . . . 49
3.1.2 Setting up a connection . . . 50
3.1.3 Testing CAN-Transmission . . . 51
3.2 Test Vehicle . . . 53
3.3 Radar Readings . . . 54
3.3.1 Connection to PC . . . 54
3.3.2 Logging Data . . . 54
3.3.3 Analyzing Radar Readings . . . 54
3.4 GPS Readings . . . 58
3.4.1 GPS readings from Car . . . 58
3.4.2 GPS readings from Bike . . . 59
3.4.3 Merging the readings . . . 60
3.5 iBeacon Readings . . . 63
3.5.1 Identifying on the application . . . 63
3.6 Simulation . . . 64
3.7 Real Measurements . . . 66
3.7.1 Static Measurements . . . 68
3.7.2 Dynamic Measurements . . . 71
3.8 Finalizing Systems . . . 72
3.8.1 Simulation of the Ideal System . . . 73
3.8.2 Simulation of the Real System . . . 73

4. Results **77**
4.1 Ideal System . . . 77
4.2 Real System . . . 78

5. Conclusion **82**

6. Future Recommendations **84**

Bibliography **89**

Appendix 90

A. Comparison between LIN, CAN and FlexRay Protocols **91**

B. Controller Area Network **92**
B.1 Standardization 92
B.2 Layered Architecture of CAN 93
B.3 Data Protocols 94
B.4 Signal Analysis 102

C. Citroën Car **109**
C.1 Technical Specifications 109
C.2 Electronic Control Units (ECUs) 110

D. Radar Sensor **113**
D.1 Sensor Objectives 113
D.2 Sensor Connectors 114
D.3 Power Supply Requirements 115
D.4 Sensor Address 115
D.5 CAN Interface 116
D.6 Message Content 116

E. FlexDevel Technical Data **122**

F. S500 Mercedes-Benz W221 Car **126**

G. GPS Receiver **127**
G.1 Technology Specifications 127
G.1.1 Physical Dimension 127
G.1.2 Environmental Characteristics 127
G.1.3 Electrical Characteristics 127
G.1.4 Performance 128
G.1.5 Protocal&Interface 128
G.1.6 LED Function 129

H. NMEA Transmitted Messages **130**
H.1 Global Positioning System Fix Data (GGA) 130
H.2 Geographic Position with Latitude/Longitude(GLL) 131
H.3 GNSS DOP and Active Satellites (GSA) 132
H.4 GNSS Satellites in View (GSV) 132

H.5 Recommended Minimum Specific GNSS Data (RMC) 133
H.6 Course Over Ground and Ground Speed (VTG) 134
H.7 ZDASiRF Timing Message 134

List of Figures

1.1 Adaptive Cruise Control [22] . . . 2
1.2 Emergency Brake Assist [9] . . . 2
1.3 Lane Departure Warning [9] . . . 3
1.4 Rear Cross Traffic Alert [9] . . . 3
1.5 The Blind Spot area [19] . . . 4
1.6 ETAC Statistics - Truck Motion . . . 5
1.7 ETAC Statistics - Injuries for different road users . . . 5
1.8 ETAC Statistics - Distribution of accidents . . . 5
1.9 Problems in Lane Change . . . 6
1.10 Bicycles-Cars Accidents [17] . . . 6
1.11 Volvo Blind Spot system (BLIS) . . . 8
1.12 Infiniti Blind Spot system (BSW & BSI) . . . 9
1.13 Blind Spot Mirror . . . 9
1.14 The difference between the rear-mirror with a blind spot mirror and without [41] 9

2.1 ECU Coupling [16] . . . 13
2.2 Bus Networking [16] . . . 13
2.3 ISO/OSI Model [16] . . . 14
2.4 Layer Communication [16] . . . 15
2.5 Peer-to-Peer Communication [16] . . . 15
2.6 Bus Node Architecture [16] . . . 16
2.7 Three Layer Model [16] . . . 16
2.8 Topologies [16] . . . 17
2.9 Protocols [1] . . . 18
2.10 Basic CAN Bus interface [10] . . . 19
2.11 Data Transmission [44] . . . 24
2.12 Structure of a CAN Network . . . 26
2.13 CAN Transceiver [16] . . . 27
2.14 Twisted CAN Network [16] . . . 27
2.15 HS and LS CAN Voltage Levels . . . 28
2.16 Gateway in the OSI Model [28] . . . 29
2.17 Standard Data Frame [16] . . . 29
2.18 Standard Remote Frame [16] . . . 30
2.19 Standard Error Frame [10] . . . 30
2.20 Standard Overload Frame . . . 30
2.21 Example for priority of messages [44] . . . 31

2.22 K-CAN Voltage Levels . . . 32
2.23 PT-CAN Voltage Levels . . . 32
2.24 Citroën C6 [2] . . . 33
2.25 Locations of different ECUs in Citroën C6 . . . 34
2.26 Two Radar Sensors attached in the front bumper of a car . . . 35
2.27 Overview of Automotive Applications [12] . . . 36
2.28 Transmission and Receiving of Pulses . . . 37
2.29 UWB Radar Resolution [32] . . . 37
2.30 Range Measurement [35] . . . 38
2.31 Block Diagram - SLR Range Measurement [35] . . . 38
2.32 Range Accuracy Performance [32] . . . 39
2.33 Antenna Patterns [35] . . . 39
2.34 Angle Measurement [32] . . . 40
2.35 Patterns [35] . . . 40
2.36 Sensor Bearing Capability . . . 40
2.37 Angle Accuracy [35] . . . 41
2.38 Absolute Angle Error [35] . . . 41
2.39 Space Segments around the Earth [13] . . . 42
2.40 Object located somewhere on the Earth using satellites [29] . . . 44
2.41 Longitude and Latitude of an object [31] . . . 45
2.42 Different levels of interactions of iBeacon at each range [25] . . . 47

3.1 CANalyzer linking between different stations on the CAN-Bus [15] . . . 49
3.2 CANalyzer different functions . . . 50
3.3 CANcaseXL . . . 51
3.4 Testing Signals . . . 52
3.5 Mercedes-Benz S500 . . . 53
3.6 Radar Logged Data . . . 54
3.7 Plot of Radar Objects (1 Sensor) . . . 55
3.8 Plot of Radar Objects (2 Sensors) . . . 55
3.9 Different Levels of Warnings . . . 56
3.10 Rear lights turn on when the bike is coming . . . 57
3.11 Proteus Design for the artificial LEDs . . . 57
3.12 Artificial LEDs in Mirror . . . 58
3.13 GPS Car Logged Data . . . 59
3.14 Screenshot from the Application . . . 60
3.15 GPS Bike Logged Data . . . 60
3.16 The route on Google Maps . . . 61
3.17 The route on Google Earth . . . 61
3.18 Car and Bike intersection point . . . 62
3.19 MATLAB plot of the GPS readings . . . 63
3.20 GPS Accuracy . . . 63
3.21 iBeacon Detection Application . . . 64
3.22 Part of the Simulink Model used . . . 65
3.23 Simulink Model - Car and Bikes positions . . . 65

3.24 Simulink Model Results 66
3.25 Error - Difference in timestamps 67
3.26 UTC of GPS reading with Errors (Arrows) 67
3.27 Path on Google Earth for Static Measurements 68
3.28 Difference in timestamps between iPad and Receiver - Static 69
3.29 Simulink Model Results - Static Measurement 70
3.30 Screenshot from the video - Static 70
3.31 Path on Google Earth for Dynamic Measurements 71
3.32 Difference in timestamps between iPad and Receiver - Dynamic 71
3.33 Error in the UTC 72
3.34 Ideal System - Simulation 73
3.35 Real System - GPS Update Rate Error 74
3.36 Real System - GPS Update Rate Solution 75
3.37 Real System - GPS Position Error 76

4.1 Routes Plot - Ideal System 77
4.2 Ideal System - 3D Simulation 78
4.3 Real System - 3 meters error 79
4.4 Real System - 4 meters error 79
4.5 Real System - 5 meters error 80
4.6 Comparison between Ideal and Real Systems 81

6.1 Garmin 19x HVS .vs. GPS Receiver - Track Log Accuracy 84
6.2 Comparison of conventional sensor and wide-angle laser radar 85

B.1 Standard and Implementation 93
B.2 Layered Architecture of CAN 94
B.3 Data Frame in Standard and Extend Format 95
B.4 Data Frame 95
B.5 Arbitration Field 96
B.6 Data Length Code 97
B.7 DLC and Data Field 98
B.8 CRC Field 98
B.9 ACK Field 99
B.10 Physical Transfer of a Data Frame in Standard Format 99
B.11 Remote Frame 100
B.12 Error Frame 100
B.13 Overload Frame 101
B.14 Different Types of Signals [44] 102
B.15 Byte Order for different processors 107
B.16 Significance of bits 107

C.1 ECUs on CAN IS 112
C.2 ECUs on CAN CAR 112
C.3 ECUs on CAN CONF 112

D.1 Short Range Radar Sensor [4] 113
D.2 Sensor Side View looking into connector mating face 115
D.3 Sensor Orientation 115
D.4 Definition for Sign of Target Bearing 120

E.1 FlexDevel Conroller - Article 3-042-P01 122
E.2 FlexDevel Block Diagram 124
E.3 FlexDevel Layout Diagram 125

G.1 GR-213 GPS Receiver 127

List of Tables

2.1 Can Classes 21
2.2 Difference between Parallel and Serial Data Transmission 22
2.3 Data Transmission 23
2.4 Data Transmission Sequence 24
2.5 Proximity States of an iBeacon 47

A.1 Comparison between different protocols 91

B.1 Coding of the number of data bytes by the Data Length Code 97
B.2 Different Number Systems 103
B.3 Window's Signal Example 104
B.4 Binary-Decimal Conversion 105
B.5 Large number conversion example 105
B.6 Conversions of Number Systems 106

C.1 Technical characteristics of Citroën C6 109
C.2 Identification of the components of the test vehicle 110

D.1 Sensor Parameters 113
D.2 Connector Parts 114
D.3 Sensor Connector Pin Usage 115
D.4 Power Supply Requirements 115
D.5 Sensor Address 115
D.6 Bit-Timing Parameters for CAN Bus 116
D.7 RDU CAN Communication Messages (n: Sensor Number) 116
D.8 Content of Sync Message 117
D.9 Content of Command Message 117
D.10 Content of Target Messages 118
D.11 Calculation Rules for unfiltered values 119
D.12 Content of Error management-Message 120
D.13 Content of Status-Message 120

E.1 Electrical Characteristics 122
E.2 Physical Characteristics 123
E.3 Environmental Conditions 123

F.1 Mercedes-Benz S500 CAN BUS 126

H.1 NMEA-0183 Output Messages 130
H.2 GGA Data Format 130
H.3 Position Fix Indicator 131
H.4 GLL Data Format 131
H.5 GSA Data Format 132
H.6 Mode 1 132
H.7 Mode 2 132
H.8 GSV Data Format 133
H.9 RMC Data Format 133
H.10 VTG Data Format 134
H.11 ZDA Data Format 134

List of Equations

2.1 Target Azimuth Angle Ratio 40

2.2 Target Angle Calculation 41

2.3 Distance of object on Earth from satellite 44

2.4 x, y and z coordinates of object from the center of the earth 45

B.1 Number of possible variants from bits 104

Acronyms

GUC	**G**erman **U**niversity in **C**airo
HTW	**H**ochschule für **T**echnik und **W**irtschaft
ZAFT	**Z**entrum für **a**ngewandte **F**orschung und **T**echnologie e.V.
FSD	**F**ahrzeug**s**ystem**d**aten GmbH
ADAS	**A**dvanced **D**river **A**ssistance **S**ystem
NHTSA	**N**ational **H**ighway **T**raffic **S**afety **A**dministration
ETAC	**E**uropean **T**ruck **A**ccident **C**ausation
EC	**E**uropean **C**ommission
BLIS	**B**lind **S**pot **I**nformation **S**ystem
BSW	**B**lind-**S**pot **W**arning
BSI	**B**lind-**S**pot **I**ntervention
High-Tech	**H**igh **Tech**nology
Low-Tech	**Low Tech**nology
LED	**L**ight **E**mitting **D**iode
C6	Vehicle Upper-class model from Citroen
MATLAB	**Mat**rix **Lab**oratory
Radar	**Ra**dio **D**etection **a**nd **R**anging

USB	**U**niversal **S**erial **B**us
GPS	**G**lobal **P**ositioning **S**ystem
CAN	**C**ontroller **A**rea **N**etwork
CAN-DB	**CAN D**ata**b**ase
CANH	**CAN H**igh Line
CANL	**CAN L**ow Line
iOS	Apple **O**perating **S**ystem
ECU	**E**lectronic **C**ontrol **U**nit
OBD	**O**n-**B**oard **D**iagnostics
ISO	**I**nternational **S**tandardisation **O**rganisation
OSI	**O**pen **S**ystem **I**nterconnection
LIN	**L**ocal **I**nterconnect **N**etwork
MAC	**M**edium **A**ccess **C**ontrol
LLC	**L**ogical **L**ink **C**ontrol
PLS	**P**hysical **L**ayer **S**ignaling
PMA	**P**hysical **M**edium **A**ttachement
PMS	**P**hysical **M**edium **S**pecification
UTP	**U**nshielded **T**wisted **P**air
CPU	**C**entral **P**rocessing **U**nit
ID	**Id**entifier
RX	**R**eceive
TX	**T**ransmit
DLC	**D**ata **L**ength **C**ode

CRC	**C**yclic **R**edundancy **C**heck
ACK	**Ack**nowledgement
SOF	**S**tart **O**f **F**rame
RTR	**R**emote **T**ransmission **R**equest
LSB	**L**east **S**ignificant **B**yte
lsb	**L**east **S**ignificant **B**it
MSB	**M**ost **S**ignificant **B**yte
msb	**M**ost **S**ignificant **B**it
NRZ	**N**on-**R**eturn to **Z**ero Coding
ABS	**A**ntilock **B**raking **S**ystem
EDS	**E**lectronic **D**ata **S**ystems
SRR	**S**hort **R**ange **R**adar
UWB	**U**ltra **W**ide **B**and
SLR	**S**equential **L**obing **R**adar
EIRP	**E**ffective **I**sotropic **R**adiated **P**ower
ACC	**A**utonomous **C**ruise **C**ontrol
CW	**C**ontinuous **W**ave
HRR	**H**igh **R**ange **R**esolution
RDU	**R**adar **D**istribution **U**nit
DOD	U.S **D**epartment **O**f **D**efense
MEO	**M**edium **E**arth **O**rbit
MCS	**M**aster **C**ontrol **S**tation

PWM	**P**ulse **W**idth **M**odulation
PC	**P**ersonal **C**omputer
SNR	**S**ignal-to-**N**oise **R**atio
GUI	**G**raphical **U**ser **I**nterface
UTC	**C**oordinated **U**niversal **T**ime
NMEA	**N**ational **M**arine **E**lectronics **A**ssociation
HVS	**H**ospitality **V**oice **S**ervices

Nomenclature

Symbol	Definition	SI-Unit
a	Acceleration	m/s^2
B	Transfer Rate; BaudRate	$kBit/s; KBaud$
I	Current	A
m	Mass	kg
P	Power	W
T	Torque	Nm
r	Radius	m
R	Resistance	Ω
t	Time	s
V	Voltage	V
f	Frequency	Hz
v	Velocity - Car Speed	$m/s - km/h$
θ	Theta (Angle)	$^\circ$
ω	Revolutions per minute	rpm
V	Volume	cm^3
-	Litre (Volume of one kilogram)	l

-	Frequency Response	dB
-	Power Level	dBm
-	Horse Power	Hp

CHAPTER 1

Introduction

1.1 Overview

Advanced driver assistance systems are systems developed to augment vehicle systems for safety and enhanced driving. Safety features are intended to stay away from collisions and accidents by offering technologies that makes the driver aware to prospective troubles, or to prevent collisions by implementing safeguards and taking over control of the vehicle These technologies offer drivers with necessary information, automate hard or cyclic tasks, and lead to an overall augment in car safety. Some of these technologies have been applied in cars for a long time ago, and they have proven to be more useful systems which resulted to a great enhance and improve in the driving experience and a better overall road safety.
Nevertheless, a lot of ADAS are right on the cutting edge of rising automotive technologies. Some of these systems will have the staying power to stick around, and you can expect to see at least a few of them in your next car. Others may fizzle and disappear or be replaced by better implementations of the same basic idea.
Based on intelligent sensor technology, driver assistance systems constantly examine the automobile surroundings as well as the driving behaviour to notice potentially critical situations at an early stage. Thus, these systems notify and vigorously support the driver and, if necessary, interfere automatically in an attempt to prevent a collision or to lessen the consequences of the accident.

Some examples of such systems are [34], [9], [6]:

1. **Adaptive Cruise Control (ACC)**

   This advanced driver assistance technology is particularly practical on the highway. With this system, a vehicle will automatically slow down or speed up in reaction to the actions of the vehicle in front of it. So for example, the system has a safety distance to maintain, and if the car is very near to the front car, so the system automatically slows down your car thus to keep the safety distance as it is, or speeds up in case the distance between the two cars is very big.

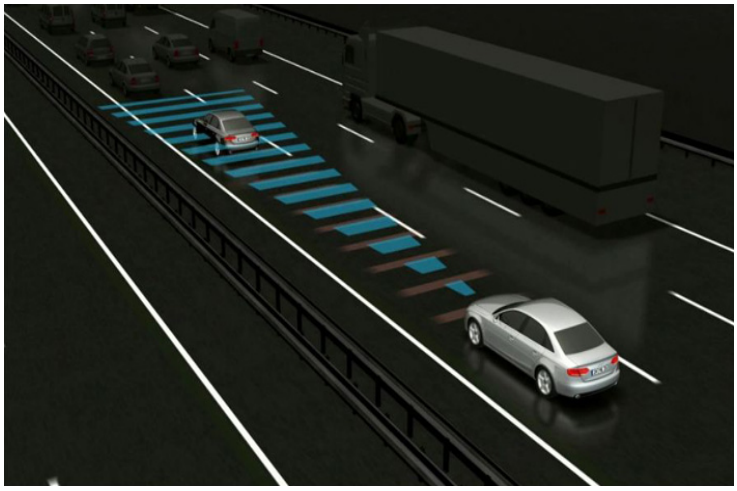


Fig. 1.1: Adaptive Cruise Control [22]

2. **Emergency Brake Assist (EBA)**

   This system detects dangerous traffic situations and ensures best braking. It can significantly decrease stopping distance, thereby extenuating an accidents harshness once it becomes obvious.

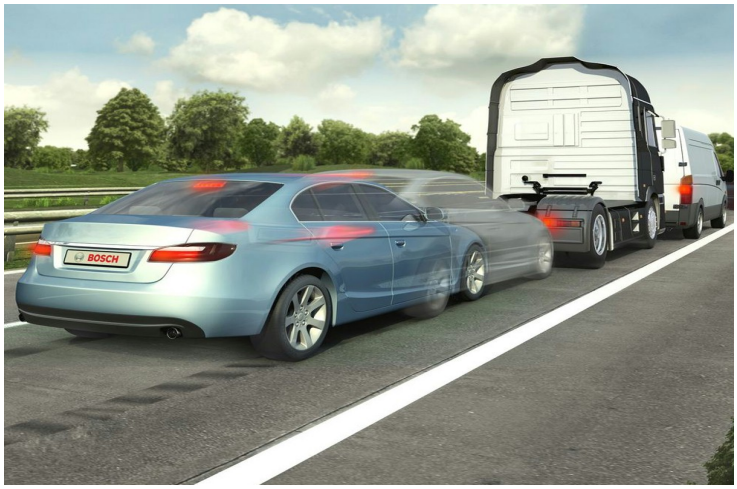


Fig. 1.2: Emergency Brake Assist [9]

3. **Lane Departure Warning (LDW)**

   Lane Departure Warning helps to frustrate unintended drift out of the obvious lane. It alerts the driver with acoustical or haptic warnings before his vehicle is about to leave the lane.

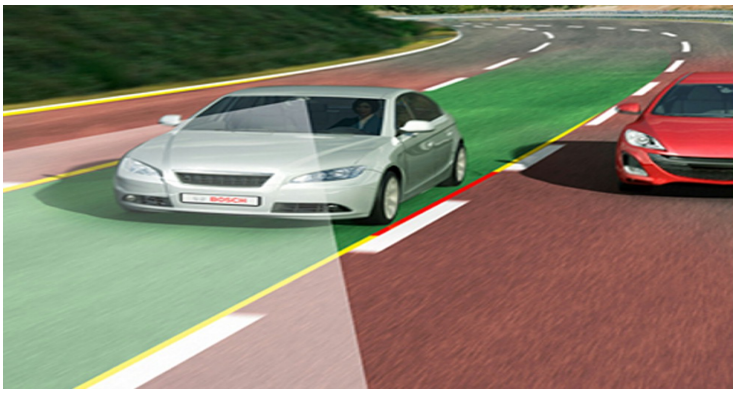


Fig. 1.3: Lane Departure Warning [9]

4. **Rear Cross Traffic Alert (RCTA)**

   Backing out of a parking space at right angles to the road can frequently be a challenge. Rear cross traffic alert can make backing out of parking spaces easier. This system can help to avoid accidents when reversing out of a parking space. The system uses two mid-range radar sensors in the back of the vehicle. They calculate and read the distance, speed and predictable driving path of vehicles detected in cross traffic. When vehicles are detected, the system generates an audible and/or visual warning to alert the driver to the impending risk of collision.

Fig. 1.4: Rear Cross Traffic Alert [9]

And there are much more from these systems; each performing a different function which at the end reaching the basic idea of ADAS, and this idea is to reach the top safety for the drivers on the road as well as the top comfortable driving process a person can have. However, all of these systems are mainly developed for this reason, and most of them are dealing with car-car accidents, but there are no such systems that are mainly working on preventing accidents with cyclists. Thus here comes the idea of this paper, which is to develop a system that tries to decrease collisions between cars and bicycles.

1.2 Theoretical Background

Collisions with automobiles are considered to be one of the greatest fears people have about cycling. In fact, this fear is the biggest barrier to getting more people on bikes. Many people want to ride, but the feeling of unsafety prevents them from doing so.

Regardless of whether the accident is a solo accident which is involving in a defect in the road or a trail or it is a collision with an automobile, most of the accidents are the result of somebody's negligence; either the driver, the bicycle or the local government which is responsible for the condition of the roads. However, there is a great percentage of bikes-automobiles collisions which occur with no one is in charge of it, and these accidents occur due to an existing problem called "The Blind Spot".

Blind Spot is the area around the vehicle that cannot be observed by the driver; either in front or by use of mirrors. The most frequent place for a blind spot is the offside quarter. Your peripheral vision can see from ahead to both sides. Your rear view can see behind, but only as far as the rear window will allow. However, there are some spots between your immediate side and the rear view mirror cuts out. This is called the Blind Spot [36].

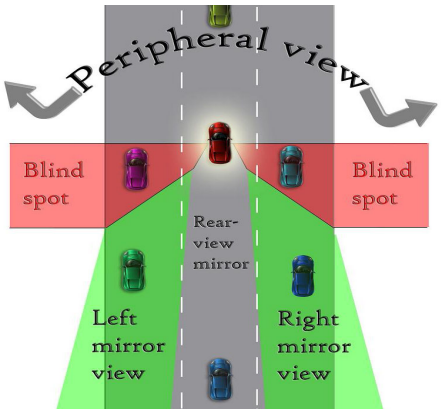


Fig. 1.5: The Blind Spot area [19]

The blind spot is one of the most significant troubles in automotive design. According to the National Highway Traffic Safety Administration (NHTSA) in Washington D.C, there are approximately 840,000 side-to-side blind spot collisions every year in the United States and about 300 of them lead to mortalities.

Another unique study, the European Truck Accident Causation (ETAC) study, was launched by the European Commission (EC) and the International Road Transport Union (IRU) from 2004 to 2006 to identify the main causes of accidents involving trucks.

They have proved that among 30 accidents, they have studied, occurring in an intersection and involving at least one vulnerable road use (pedestrian or a two-wheel vehicle) that blind spots is the main cause of 47% of them. And 2/3 of these accidents are fatal accidents.

And here are some statistics from the study [43]:

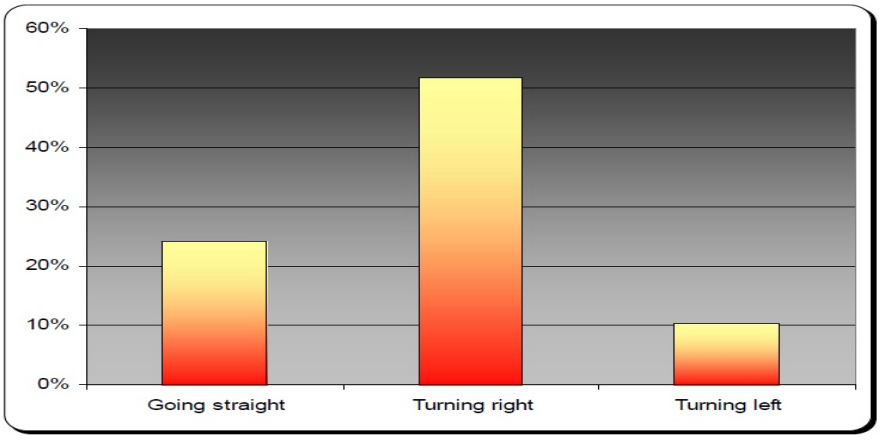


Fig. 1.6: Truck Motion [43]

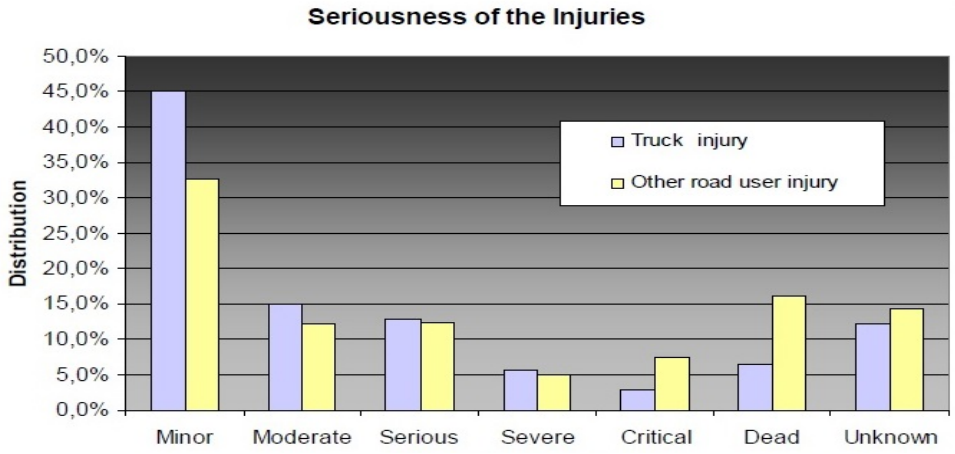


Fig. 1.7: Injuries for different road users [43]

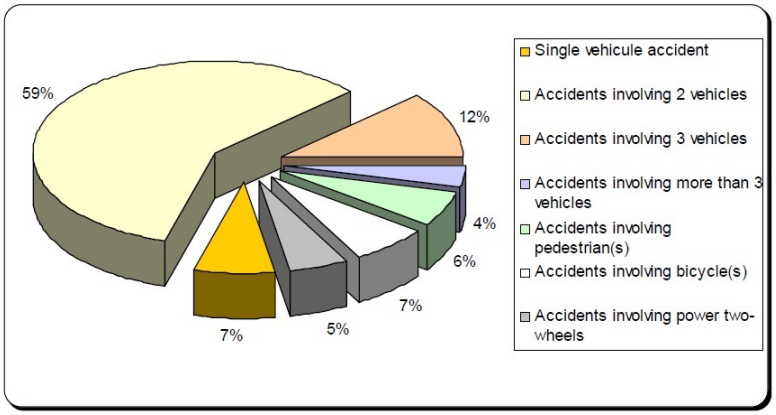


Fig. 1.8: Distribution of accidents [43]

You can't avoid what you can't see, and the blind spot keeps you from seeing things that you desperately need to avoid. If you make a lane change while there's a car lurking invisibly just off your rear bumper, you'll probably be able to catch the error in time to prevent it. But you might also hit the other car or force it off the road (because the driver of that other car can see you, and will wildly attempt to avoid you), and at worst you could trigger a multiple pile-up of cars urgently trying to get themselves out of the way of your boneheaded maneuver.

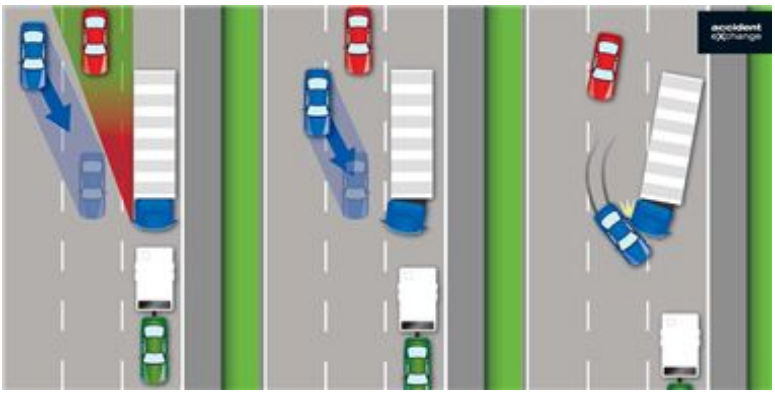


Fig. 1.9: Blind Spot during lane change [20]

1.3 Problem Statement

So our main problem is that accidents still exist and are increasing daily with a high percentage, especially, bicycles-cars accidents. Pedestrians, cyclists and motorcyclists are considered vulnerable road users because they have no external frame to protect them and thus at a higher risk of injury and death than vehicle occupants.

Fig. 1.10: Bicycles-Cars Accidents [17]

With clear significant data and statistics from a research done by Dr.-Ing. Jean Emmanuel Bakaba from the German Insurance Association in Vienna, 2013 about bicycles collisions in

Germany during the past years; he showed us how big the problem is for cycling in general and how can cyclists' life be exposed to danger [7].

That's why cyclists-automobiles crashes are still considered a major problem and remain to be a serious concern for all the people worldwide.

1.4 Literature Review

Many research institutes and car companies tried to find a solution for obtaining safety for cyclists during driving in the road.

They were divided into two groups; a group which was working as making general researches about bicycles-cars accidents and analyzing the main reasons that cause such accidents including negligence of drivers, alcohol drinking, age groups, the absence of helmets and also hit-and-run, however, the other group which was mainly the car companies was working on a main problem which is the blind spot.

Both groups had their statistics for such crashes and submitted results. The first group's results was mainly dealing of ways to avoid these accidents as observing the rules, riding with situational awareness, and wearing safety equipments like helmets, and all of these results were kind of safety precautions for both the car's driver and the cyclists that they have to abide and respect, as well as, some advices for the governments to increase this awareness and improving the condition of the roads.

On the other hand, the second group submitted results showing how to make the blind spot less dangerous. In fact, there are several things that can be done in order to prevent blind-spot accidents or at least, decrease them. Over the last few years, major car companies and research institutes have come up with several ideas that can alert you for the presence of another car or a bike in your blind spot. Some of these ideas where high-tech and rely on the latest technology of the usage of different sensors and image processing (camera detection), however the others were low-tech which were relying on something very simple as the improved rear-view mirrors.

Each auto manufacturer approaches the issue of blind spot recognition in a somewhat diverse way; however we can inexactly assemble the new blind spot screening innovations into two separate classes: active and passive [33]:.

- **Active:** This type of monitoring mainly used electronic detection devices mounted on both sides of the car that sends out electromagnetic waves or takes computer processed images with a digital camera and analyzes them, thus giving you a kind of alarm signal. When one of these devices detects an object coming with a speed towards your car, it gives this signal. This signal could be some flashing lights or by making some audible sounds. And here are some of the famous auto manufacturers that already worked in this technology:

– **Volvo:** As their great care and concern that Volvo feels about driving safety, they introduced an automated blind spot detection system called Blind Spot Information System (BLIS). This system uses cameras placed in the car's side rear view mirrors and a computer processed the image from those cameras to see if an object is coming towards the driver car and may hit it while changing lanes for example. The newer Volvo BLIS systems use radar and are mounted in the rear of the car which can send electromagnetic waves that bounce off solid objects and return an echo indicating that they are existing, and thus giving some flashing lights or glowing LEDs for the driver [21].

Fig. 1.11: Volvo Blind Spot system (BLIS)

– **Infiniti:** This Company actually did a new unique intervention which is two built-in systems. The first is called Blind-Spot Warning (BSW) which works quite similar as the other carmaker's system work. It uses radar to detect vehicles approaching from the side of the cars and some blinking LEDs in the rear view mirrors to indicate the cars in the blind spot. And for more safety for the car driver, Infiniti added the second system to this detection system which is the Blind-Spot Intervention (BSI). This system will start gently applying the brakes on the side of the car away from the danger, causing the vehicle to slowly go swerve back towards the safe lane and preventing it to deviate to the danger one. Probably you can override this delicate prod in the event that you push the wheel hard enough, yet it wouldn't be a great thought to do so [24], [26], [27].

Fig. 1.12: Infiniti Blind Spot system (BSW & BSI)

And of course other big companies as AUDI, Ford and BMW developed such systems which work on the blind spot detections, but all of them are mainly working to avoid car-car crashes while changing lanes.

- **Passive:** This type of class as stated before is the low-tech group. Many car manufacturers are offering you the alternative to place a special convex mirror in the corner of your external rear-view mirror that can be able to see the areas where the normal rear-view ones cannot do it. So this gives the driver a wider vision angle so that he can be able to detect any object passing by his blind spot and so taking his precautions of avoiding this object.

Fig. 1.13: Blind Spot Mirror

Fig. 1.14: The difference between the rear-mirror with a blind spot mirror and without [41]

1.5 Motivation

After checking all of these statistics and revealing the main problem behind the danger for cyclists, and seeing all the possible solutions for different people or companies who tried to decrease the rate of the collisions, I got motivated to work in such a project.

My project idea was to implement an Advanced Driver Assistance Safety system for the cars to provide safety for both driver and cyclists. It is not working only on collisions due to blind spots, however, it is working on decreasing the collisions between bicycles and automobile in general, either it was due to blind spot or due to negligence.

This system will be using different techniques to achieve the critical situation for the cyclist when he can get a crash with the car.

I will be using three approaches for doing this:

- Position and velocity of the car and the bicycle using GPS measurements
- Detection of objects near the car using radar sensors
- Getting the distance between the bicycle and the car using iBeacon

These three approaches are merged together and give us the detection if the bicycle is in a critical situation or not, and according to the result, warnings and alarms will be given for the driver to take an action.

And this is considered to be the main objective of this project.

1.6 Methodology

1. Understanding the car electronics of the Citroen C6 car which will be used in this project.
2. Understanding the different CAN interfaces that are built-in inside the car
3. Learning how to send and receive signals on the car's CAN Bus
4. Trying to control special functions inside the car as the Horn, Flash Lights, and doors using CAN messages
5. Understanding the theory of operation of the Radar Sensor
6. Analyzing objects detected from the Radar Sensor
7. Filtering the objects to get the object that is in a critical situation with the car
8. Understanding the theory of operation of the GPS

9. Taking GPS measurements from the car and the bike
10. Analyzing these measurements
11. Filtering the readings from both the car and the bicycle to detect which object is in critical situation with the car
12. Understanding the theory of operation of the iBeacon
13. Taking measurements between iBeacon and the iOS device
14. Working on a Simulink Model to merge all of these data and finally gives the alarm for the critical object that will make a collision with the car
15. Implementing this model in real life (Car)

CHAPTER 2

Basic Tools and Methods

2.1 Serial Bus Systems

2.1.1 Main Idea

The late history of vehicles is portrayed by intensive electronification systems. What make car manufacturers work in this, are the continuous demanding wishes for drivers to obtain more safety and comfort in the automobile, in addition to more stringent exhaust emissions legislation. Furthermore, competition and cost pressure play an important role into the demanding of developing and producing a high-tech product in car industries.

At the start of electronification, the electronic structural planning was embodied by ECUs that were working independently from each other. Presently, it was recognized that the coordination of these electronic control units and frameworks had the potential for tremendously enhancing and developing vehicle usefulness. Throughout the span of time, electronic capacities discovered their route into the vehicles that couldn't be actualized without profoundly data exchange between ECUs.

Thus it was decided to implement an active network system which connects all ECUs in the automobile together. In the beginning, the networking was implemented conventionally, i.e. each signal which is transmitted between different ECUs goes an assigned electrical line. But unfortunately, this was not an effective method because the electronic systems in the car was requiring more intensive networking to be well implemented which lead to high expenses for wiring systems. In order to get out from this impasse, a communication channel (BUS) was implemented to join a number of electronic control units where a form of bit-serial exchange of data is transferred through this channel.

And from this point, serial bus systems were invented. Serial bus systems allow smooth data exchange between electronic control modules in the automobile and represent the backbone of modern electronic architectures. Nowadays, huge serial bus systems perform their job as an essential part of a modern electronic architecture in an automobile. The benefit behind using such systems than the basic wiring systems were to reduced costs, space re-

quirements, weight and errors in implementing these wires. Not only that, but it allowed a simplified project planning and installation in the car. Moreover, they fulfil the requirements of reliable data exchange in real-time and assure high flexibility in making changes or extensions [16].

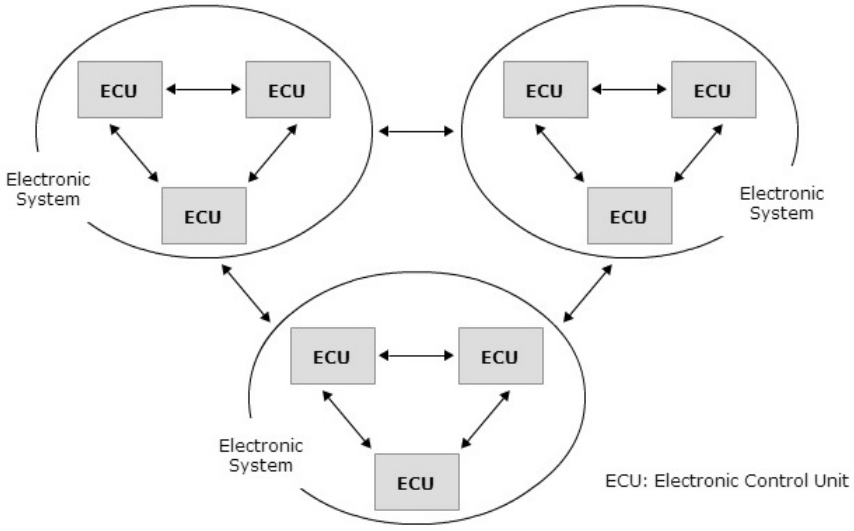


Fig. 2.1: ECU Coupling [16]

2.1.2 Communication

Simply, two ECUs (bus nodes) are needed to be interconnected in a serial communication system via a bus. Significant communication structures and rules are needed to ensure that the bus nodes can effectively allow data exchange without any troubles [16].

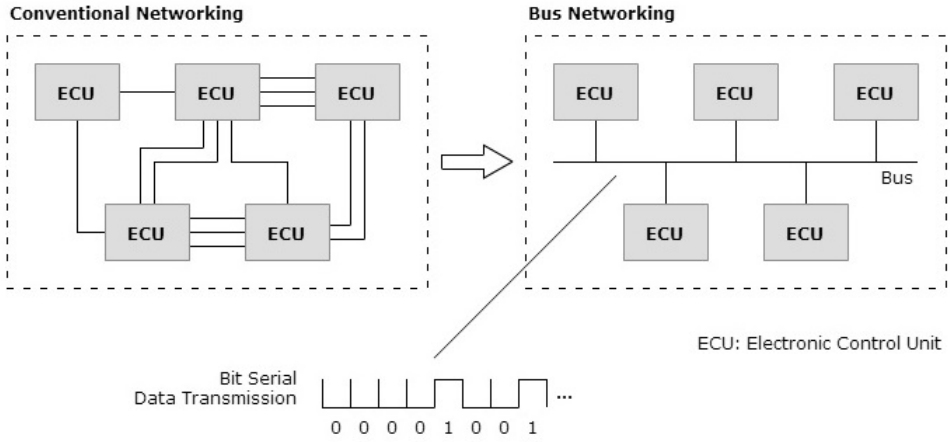


Fig. 2.2: Bus Networking [16]

2.1.3 ISO/OSI Model

The ISO/OSI communication model (Open System Interconnection) published in 1983 by ISO (International Standardisation Organisation) makes a significant contribution here. As can be seen in the figure ISO/OSI Model, this model organizes the communication process, with its numerous and complex tasks, into seven clearly defined layers that build upon one another (seven layer model) and defines communication between the layers [16].

Since for serial data exchange between electronic control units in the automobile, it is primarily the two lowest layers that are relevant (bit transmission and data protection layers), and functions of the unconsidered layers can be added to the uppermost layer (application layer), this reduces the seven layer model to a three layer model.

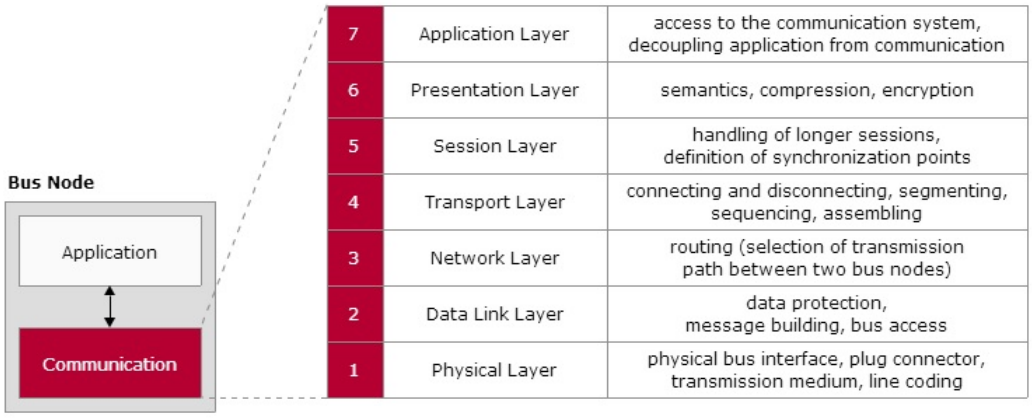


Fig. 2.3: ISO/OSI Model [16]

The communication process is modeled into layers that are built one over the other. This is an essential for working out measures for the individual layers or interfaces and decreases intricacy, since one layer may be changed out without influencing the layers above or underneath it. The communication layers, from bottom to top, are the Physical Layer, Data Link Layer, Network Layer, Transport Layer, Session Layer, Presentation Layer and Application Layer.

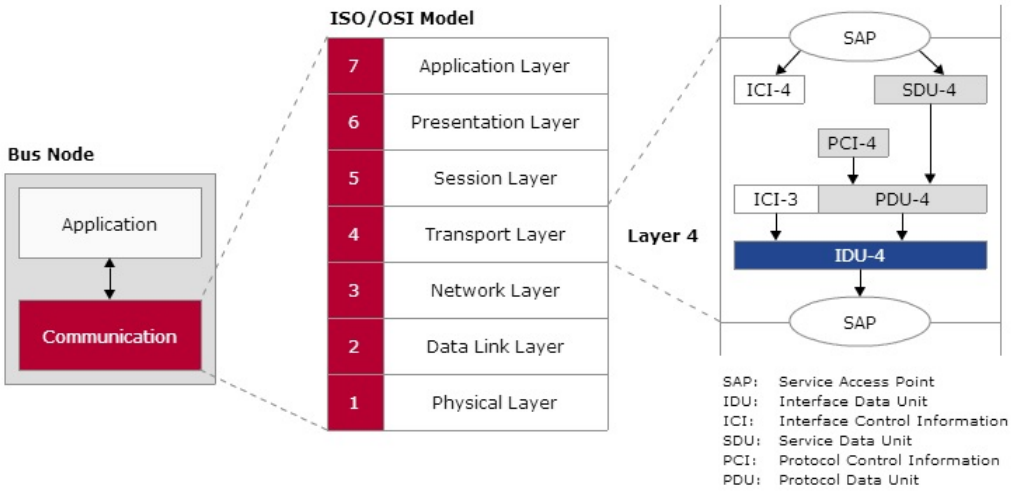


Fig. 2.4: Layer Communication [16]

The ISO/OSI model describes communication between communication partners as Peer-to-Peer communication, in which individual layers of the sender communicate with corresponding layers of the receiver: layer 7 of the sender communicates with layer 7 of the receiver. Layer 6 of the sender communicates with layer 6 of the receiver, etc. This is implemented by defining Peer-to-Peer protocols and by exclusive utilization of services of the layer beneath.

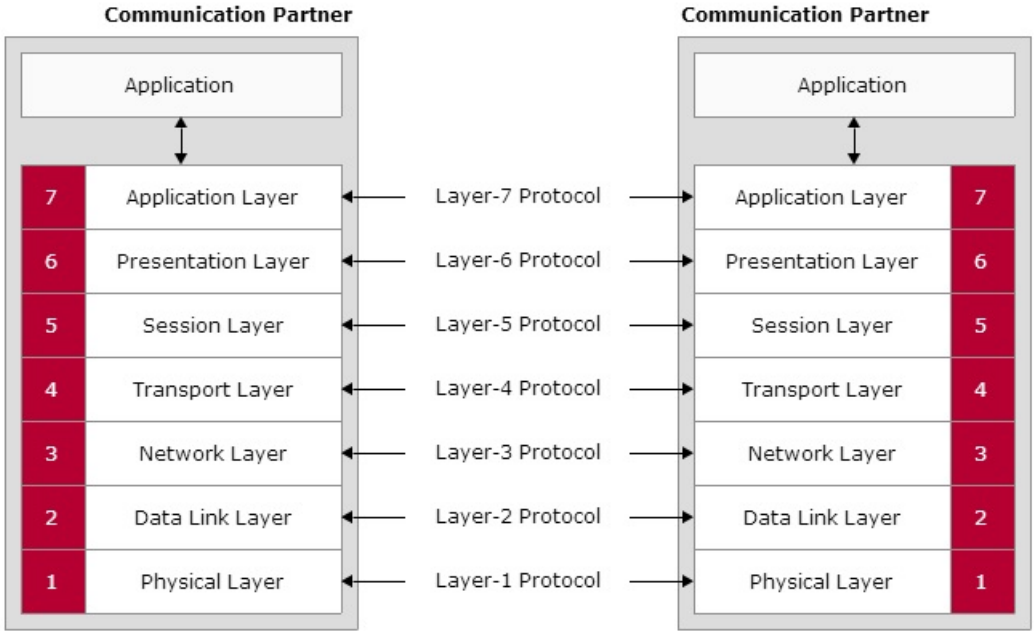


Fig. 2.5: Peer-to-Peer Communication [16]

2.1.4 Three Layer Model

For serial data transfer between ECUs in the automobile, not all of the communication functions, defined in the ISO/OSI model, are important. Essentially, only the two lowermost layers are relevant to serial data communication in the car. These layers are the Data Link Layer and the Physical Layer [16].

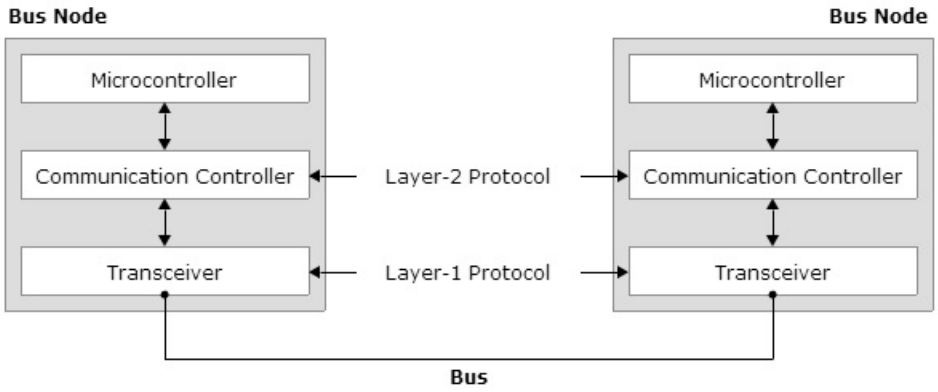


Fig. 2.6: Bus Node Architecture [16]

The Data Link Layer performs the following tasks: Addressing, message building (Framing), bus access, synchronization, error detection and error correction. The layer 2 protocol defines how these tasks are executed. Generally, a communication controller handles execution of layer 2 tasks.

The Physical Layer contains a description of the physical bus interface and conventions for physical signal transmission. Generally, the physical bus interface is implemented by a transceiver. In sending, the most important task of a transceiver is to convert the data received by the communication controller to the voltage levels defined in the layer 1 protocol. In the opposite direction, the transceiver converts voltage levels received from the bus into input signals that can be handled by the communication controller.

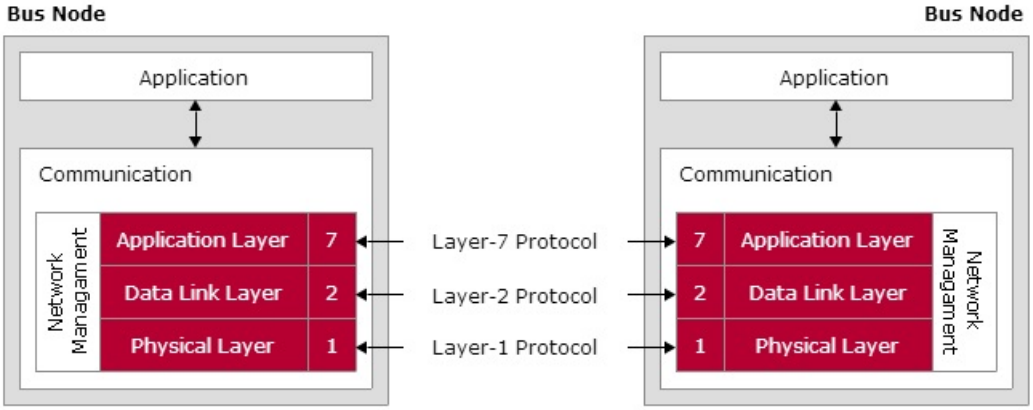


Fig. 2.7: Three Layer Model [16]

2.1.5 Architecture of Serial Bus Systems - Topologies

Topology or system topology is seen as the interconnection structure among the communication members in serial communication frameworks. In networking ECUs in vehicles, standard topologies are for the most part utilized: Star, Ring and Bus topology. Since the decision of a certain topology has colossal outcomes for the execution of a serial communication framework, this choice ought not to be made without considering the framework's obligation conditions [16].

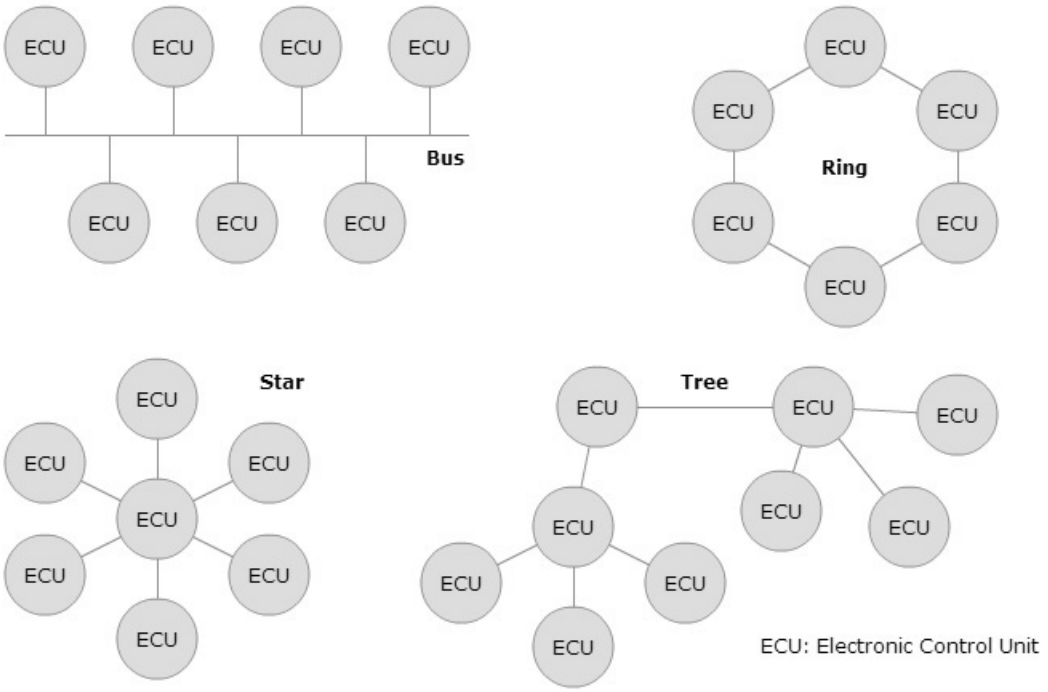


Fig. 2.8: Topologies [16]

Very widely used in the motor vehicle are serial bus systems, i.e. serial communication systems based on a bus topology. CAN, LIN and FlexRay are all based on the bus topology: all communication members (bus nodes) are connected passively to a common transmission medium (bus). Since by definition all information achieve all bus nodes in a serial bus system which is verifiably a dissemination system or telecast system.

An exceptionally critical preference of serial bus systems is that they allow any coveted logical interaction structures. Additionally worthwhile are the low wiring cost and simple extendibility. Regardless, a serial bus system does not simply show preferences. For instance, its bus length and number of bus nodes are restricted, in light of the fact that sign recovery is not regularly performed in serial bus systems. Besides, long electrical lines must be ended by a supposed "Characteristic Impedance" to forestall sign reflections at the line closes. Lastly, a break in the transmission medium prompts disappointment of the correspondence communication system.

2.1.6 Protocols

About two decades ago, the vehicles have experienced an authentic electronification process: Continually expanding amounts of electronic systems accommodate a developing level of security and comfort in automobiles driving. Serial communication systems assume a key part here. That is on account of numerous significant electronic systems would not be plausible without data exchange. Thus we have three main protocols used in serial bus communication, which are called CAN, FlexRay and LIN.

FlexRay, CAN Bus, and LIN bus are very different protocols and fit very different applications. Therefore, it is very unlikely that any of these protocols will replace the other [42].

Fig. 2.9: Protocols [1]

They are all automotive protocols. Like other protocols, they all have their pros and cons. CAN is the main protocol that is in use nowadays. All newer model cars are mandated by law to use this protocol for connecting engine control units (ECUs). LIN is a protocol that is used for interconnecting other components within a car; its main advantage is that is cheaper to implement than CAN. Lastly, FlexRay is the newest protocol of the bunch. It was designed to supersede CAN, being both more reliable and permitting faster data speeds; it was however, also more expensive. The consortium that was developing the protocol disbanded in 2009. However, the specifications are still available, and the protocol is currently being converted into a standard [42]. For more details and comparisons in different specifications for the 3 protocols, please refer to Appendix A.

2.2 Controller Area Network (CAN)

2.2.1 Introduction

The development of the CAN started by the early 1980's by Robert Bosch GmbH. This came after the realization of that normal wiring systems were not effective to connect all electronic control modules in the vehicle. It was based on the fact that a mid-size car over 600 different harness types with more than 2000 meters of cable length and had 100 kg. Outreach took this serial bus system for the first time in 1991 and today covers about 70 % of the total network in an automobile from. Nowadays, CAN is performing many helpful services in automotive industry through joining and networking ECUs in the powertrain, chassis and convenience areas in the car. Most importantly, CAN is portrayed by extremely dependable data exchange that fulfils real-time requirements [10].

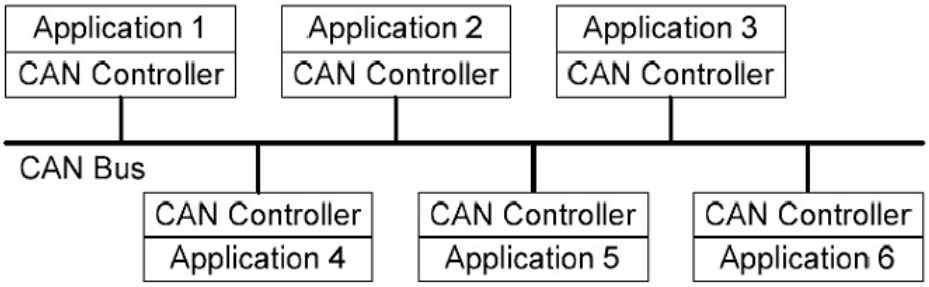


Fig. 2.10: Basic CAN Bus interface [10]

The CAN works with the "multi-master" principle; which means that when the bus is free, any unit from the ECUs can transmit a message, the unit with higher priority to be transmitted gain access to the bus. This is each control device capable of equal rights to send on CAN and to receive it. In addition, the linear bus structure CAN offers the advantage that in case of failure of a control device the operation of the system is not affected, and the communication of the other participants can continue undisturbed. Due to the considerable number of control units in a modern motor vehicle (in excess of 60 ECU), complex networks have prevailed with several bus systems. The introduction of several CAN systems results in a great relief of the individual networks, as well as a significantly better topological handling. By this means information can be transmitted faster and more secure, with the individual buses are connected via gateways.

Nodes can be added to the CAN network without requiring any change in the software or hardware of any node and application layer. This gain in flexibility compared to other systems (for example, point-to- point connections) plays in particular in the automotive industry, different equipment of vehicles, a considerable role. This results in easier ways through this network expansion by adding more nodes.

2.2.2 CAN advantages over conventional wiring

Bus systems allow a significant reduction of cables and connectors. Thus, leading to a decrease in the price and weight of the connection system. The result is an Advanced communication skills, which would not be possible by a simple wiring through diagnostic components constant control is available. A protocol detects transmission errors that may occur due to electromagnetic interference, for example, and corrects them automatically by retransmission. Security is thus given by redundancy. Modularization e.g. ECUs also lowers the price as control devices often need to be programmed accordingly [44].

To an increasing extent meanwhile sensors (e.g. steering angle sensor) and actuators (e.g. wiper motor VW) equipped with processors to process the data. If the data of such intelligent components go directly to the bus system, they do not pollute the ECU with the forwarding. In the diagnostics section of the CAN bus for the transmission of status and fault memory is used, and for flash programming the ECU.

The advantages of linear CAN bus topology at a glance:

- Smaller harness, the cabling is low
- Transmission medium is an inexpensive and easy-to-use twisted-pair cable.
- CAN stations can be retrofitted relatively easily inserted and removed in the existing CAN bus. It is merely the connection is established or disconnected to the bus. This aspect plays a significant role especially in troubleshooting and repair
- The failure of a CAN station has no direct impact on the CAN bus. All other stations can continue to communicate without restriction.

The disadvantages of this topology have the following effect on the CAN bus:

- The bus line can not be executed as long as the electrical properties (e.g., signal reflections) set physical limits in connection with the transmission speed.
- The same is true for the stubs to the control units in vehicles. Depending on the transmission speed must not exceed a certain length.
- In order to optimize the signal quality, the ends of the bus must be "terminated" with terminating resistors. Especially at high transfer speeds can make the entire bus inoperative an improperly terminated cable end.

For more information about CAN Standardization, please refer to Appendix B.

2.2.3 CAN Classes

Current motor vehicles cross link a large number of control units together, bring the different demands. Therefore, multiple CAN bus systems are installed in the vehicle. These differ primarily in the transmission rate and are divided into three classes.

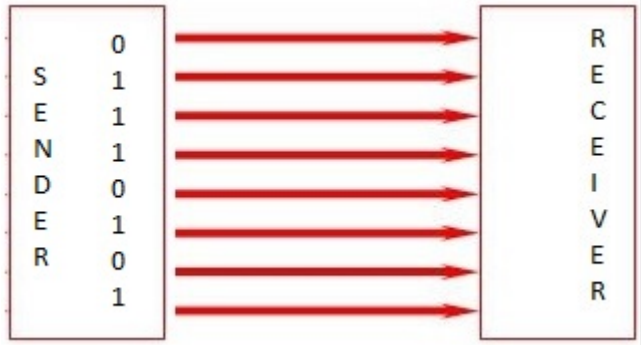
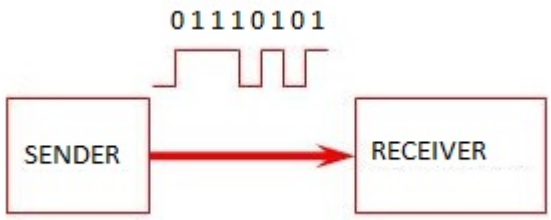
Tab. 2.1: Can Classes

CAN A $< 10\ KBit/s$ Diagnosis (Conventional)	Vehicles with CAN-bus equipped with a diagnostic system. Such systems read out fault memory and allow for actuator diagnosis. The data transfer speed is not as important as the data are only occasionally read in the workshop for maintenance and diagnostic purposes. The diagnostic port (also K-line and L-line called) but must be robust and fault tolerant. On newer vehicles, the diagnosis directly on the actual bus (CAN C) is carried out.
CAN B $<= 125\ KBit/s$ Comfort, Display, Body	About this (low-speed CAN) bus to communicate electronic control units for lighting, air conditioning, lock and fittings. Here is a transfer your important data when not as important high speed (e.g., K-CAN, body CAN, comfort-CAN). The bus must still be fail-safe and robust. Therefore, he works in the car mostly on the fault-tolerant standard ISO 11989-3.
CAN C $<= 1\ MBit/s$ Engine, Transmission, Diagnosis (BUS)	At this (high-speed CAN) bus, the control units for engine management, transmission, ESP, ASR and ABS are connected, for example. The bus must be real-time capable, i.e., the data transfer may be delayed only by the extremely short bus. Meanwhile, a real-time diagnostics on a separate diagnostic bus is possible. This bus must be fast, because large amounts of data must be transferred in a short time. In the car usually comes standard ISO 11898-2 for use.

2.2.4 Communication

For encryption of a message, usually 8-bit code is to be used. Depending on the manner in which the bytes of a message are to be transmitted from transmitter to receiver, a distinction is made between the parallel and serial transfer.

Tab. 2.2: Difference between Parallel and Serial Data Transmission

Parallel	**Series**
	
In the parallel data transfer eight bits are simultaneously (parallel) transmitted from the transmitter to the receiver. This, however, a cable with eight parallel routed cables is necessary.	The serial interface is mainly used in digital communication between control units.The data to be transmitted bit by bit in order (serially) transmitted on a single line.
Advantage: High Transfer Speed Disadvantage: High Cabling Application: Applied to Computers	Advantage: Reduced Wiring Disadvantage: Slower Data Transfer Application: Applied to the CAN bus in vehicles

Data Transmission Sequence

Each device (ECU) connected to the CAN Bus control device has something called Bus Connection. This consists of a bus controller and a transceiver.

Bus Controller coordinates the data transmission and receiving over the bus, so that the CPU free of these tasks and is thus not charged additionally. In order for the data exchange via a common bus line does not end in an uncontrollable chaos, the observance of certain rules of communication is essential.

These rules are laid down in a protocol. The bus controller ensures that this is adhered to. Similar to the signal converter and the output stage of the control unit, provides the transceiver that the signals on the bus in the bus controller readable signals are converted and vice versa. Only protocol compliant signal on the bus is to be sent. Bus controller and transceiver are usually combined to form one component.

Tab. 2.3: Data Transmission

Controller	The CAN controller receives the data to be sent from the microcomputer in the control unit. It prepares them and gives them to the CAN transceiver. The data in the controller in a high-frequency square-wave voltage (200 Hz) and converted to a low DC voltage (e.g., $5V$) modulated. The controller also receives data from the CAN transceiver, that also prepares and outputs it to the microcomputer in the control unit.
Transceiver	The link between the control unit and data bus. The CAN transceiver is a transmitter and a receiver. It transforms the data from the CAN controller and sends it to the data bus lines. Just as it receives the data and converts it to the CAN controller.
Data Bus Termination	The data bus termination is a resistor that prevents the data which is sent from the ends come back as "Echo" and corrupt the data. The diagnosis is made possible through these resistors. They prevent a so-called feedback. Frequently 120 Ω are used.
Data Bus	The data bus lines are bi-directional and are used to transmit the data. They are with CAN High (1) and CAN Low (0) respectively. To prevent interference to the data transmission, the two data bus lines are twisted together. At the same time this also prevents noise radiation from the data bus. Additional shielding (such as coaxial cables) is not required. On both lines, the respective voltage is opposite to that when the voltage on the one line is 5 V, it is on the other 0 V and vice versa. Result, the voltage sum is constant at all times, and the electromagnetic field effects of the two data buses are mutually cancelling on. For data transmission, only a single-core cable would be needed basically. However, such a single-wire bus allowing only a relatively small transfer rate. Moreover, the electric vehicle mass must be available to all stations connected to the single-wire. Therefore, a single-wire bus is more sensitive to external electrical glitches.

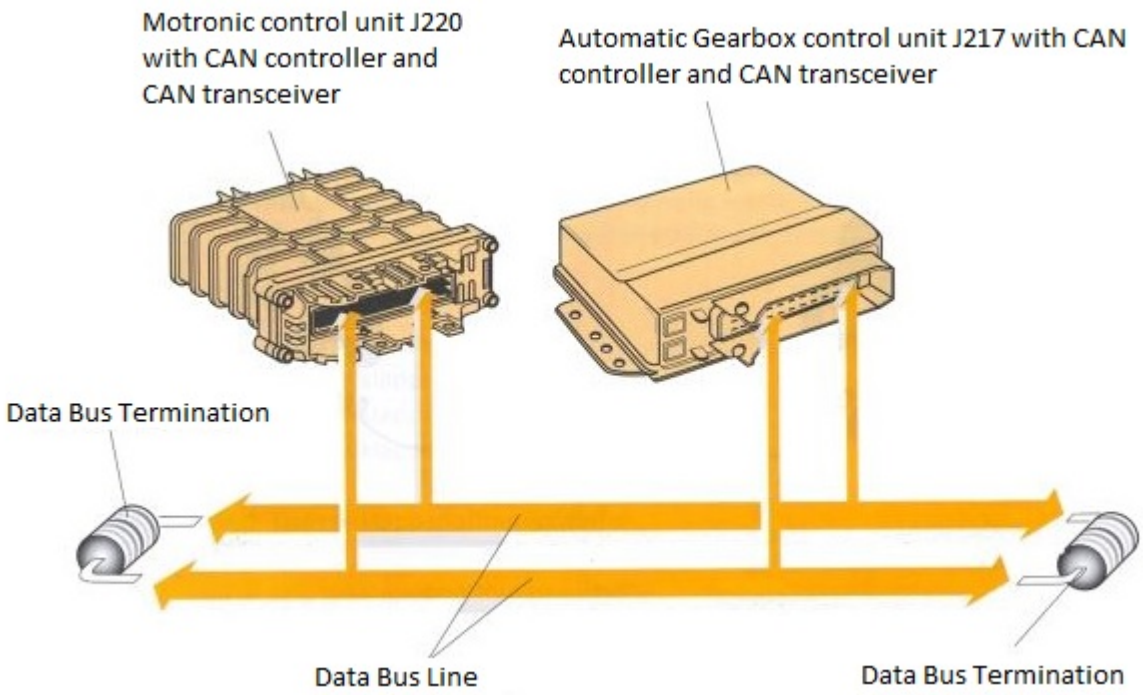


Fig. 2.11: Data Transmission [44]

Tab. 2.4: Data Transmission Sequence

Data	Description
provide	The data is the CAN controller provided by the control unit to send.
send	CAN transceiver receives the data from the CAN controller, converts them into electrical signals and transmits them.
transfer	In the CAN lines, the voltage signals are broadcast to all connected control devices.
receive	All other control devices that are networked with the CAN data bus, are to be receivers.
check	Check the control devices, whether they need the data received for their functions or not.
take	If the data is important, they will be accepted and processed, otherwise neglected.

- **CAN Network**
  A typical CAN network consists of a number of CAN nodes that are connected together by a physical transmission medium which is the CAN Bus. In practice, the CAN network is usually focused on using the line topology with a linear bus to which each

ECU is connected separately by a CAN interface [16]. An unshielded twisted two-wire line is the physical transmission medium utilized most often as a part of requisitions (Unshielded Twisted Pair UTP), over which symmetrical signal transmission happens. Normally, UTPs have cable cross sections between 0.34 mm^2 and 0.6 mm^2. Line safety ought to be short of what 60 $m\Omega$.

The most extreme data rate is 1 $Mbit/s$. A maximum network augmentation of about 40 meters is permitted. With increasing cable length at the same time decreases the data rate. At the closures of the CAN network, bus termination resistors help avoiding transient phenomena (reflections). According to ISO 11898, the greatest number of CAN nodes is 32 where they are connected in parallel with CAN. The branches of CAN to the individual nodes should not exceed a cable length of 0.3 m.

The CAN physical layer can be divided into two distinct systems. Such a distinction in High-Speed CAN (HS-CAN) and low-speed CAN (LS-CAN) is made in principle. CANH and CANL are at the line ends of a HS -CAN with a terminating resistor (108 $\Omega < R <$ 132 Ω; $\approx$ 120) connected according to Physical layer modelling of the bus systems CAN and FlexRay in Motor vehicle. This terminator is not required by the LS-CAN according to ISO 11898-3. The reason for this is the significantly lower transmission rate of this network [10]. The HS-CAN is mostly used for drive applications and the communication of the suspension components and has transfer rates 125-1000 $Kbit/s$. The transfer rates of the LS-CAN are at bit rates between 25 to 125 $Kbit/s$ significantly lower than those of the HS-CAN. Therefore, the LS-CAN is mainly used for applications that do not require high data rates (e.g., body electronics). It should be noted that when one line fails, the HS-CAN fails the entire bus system. As described above, here the CAN nodes are only connected to the CANH and CANL line, whereby a differential voltage measurement in a line defect is impossible. The LS -CAN also has a connection to the vehicle ground. In case of interruption of the two CAN wire so the LS -CAN will continue to work as a system. For this network a special logic is integrated, which guarantees an unqualified single wire. Due to the necessary signal processing, the transmission rate is compared with the HS-CAN significantly lower.

- **CAN Controller**
  An ECU that want to be a participant in any CAN communication requires a CAN interface. This includes a CAN controller and a CAN transceiver. The CAN transceiver connects the CAN controller to the physical transmission medium which is the CAN Bus. Normally, the two components are isolated electrically by optical or magnetic decoupling, so in case of any over-voltage on the CAN bus which may destroy the CAN transceiver, the CAN controller and the underlying host are preserved.

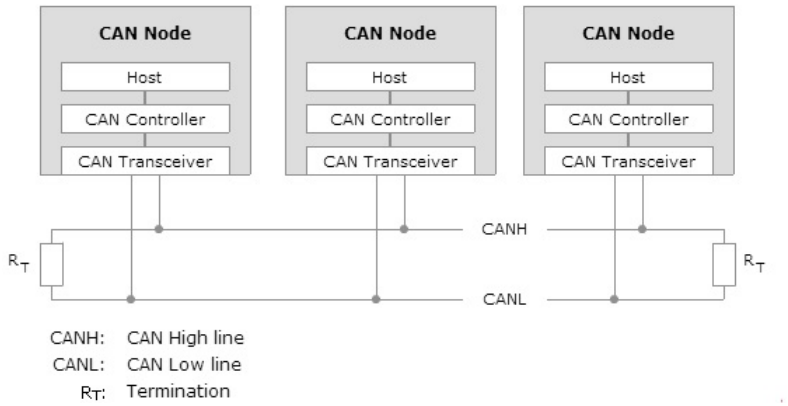


Fig. 2.12: Structure of a CAN Network

In a CAN network, the CAN nodes differ in the number of CAN messages they each send or receive. There are also great differences in the frequencies of sending and receiving. For example, one CAN node might receive five different CAN messages, each at a cycle of ten milliseconds, while another CAN node just needs to receive one CAN message every 100 milliseconds. These obvious differences have resulted in two fundamental CAN controller architectures: CAN controllers with and without object storage. (Full and Basic CAN controllers).

- **CAN Transceiver**
  Formerly, the CAN controller was often-times associated with the communication media (CAN bus) by a discrete circuit. Nowadays, CAN transceivers handle the bus connection. A CAN transceiver dependably has two bus pins: one for the CAN high line (CANH) and one for the CAN low line (CANL). This is on the grounds that physical signal transmission in a CAN system is symmetrical to accomplish electromagnetic similarity, and the physical communication medium in a CAN system comprises of two lines.

  Normally, a refinement is made between fast CAN transceivers and low-speed CAN transceivers. Rapid CAN transceivers help information rates up to 1 $Mbit/s$. Low-speed CAN transceivers just help information rates up to 125 $Kbit/s$. Then again, low-speed CAN transceivers guarantee an issue tolerant design of the bus interface (e.g. a disappointment of one of the two correspondence lines does not bring about aggregate correspondence disappointment).

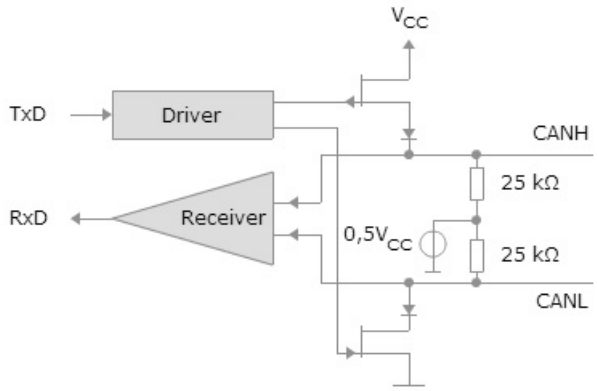


Fig. 2.13: CAN Transceiver [16]

Fig. 2.13 shows the basic layout of a high-speed CAN transceiver. When both output transistors are blocking, both CAN lines assume the same potential ($0.5*Vcc$), and the differential voltage is zero. As soon as both transistors conduct, this produces a differential voltage between the two lines that is a function of the load resistance. According to ISO 11898-2 this difference should be 2 Volt. Accordingly, a current of approx. 35 mA flows.

- **CAN BUS**
  Physical signal transmission in a CAN network is based on transmission of differential voltages (differential signal transmission). This effectively eliminates the negative effects of interference voltages induced by motors, ignition systems and switch contacts. Consequently, the transmission medium (CAN bus) consists of two lines: CAN high line (CANH) and CAN low line (CANL).

  Twisting of the two lines reduces the magnetic field considerably. Therefore, in practice twisted pair conductors are generally used as the physical transmission medium.

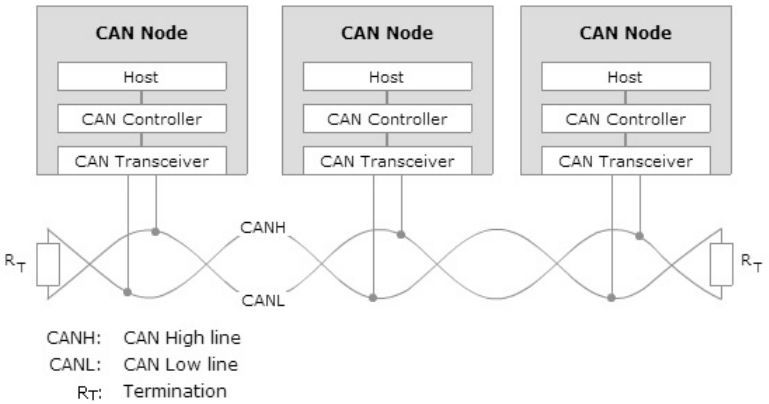


Fig. 2.14: Twisted CAN Network [16]

- **CAN Bus Levels**
  The actual transfer of information is based on a transfer of voltage differences (differential signal transmission). The voltage level of HS-CAN and LS-CAN is also distinguished here according to ISO 11898-2 and ISO 11898-3. In principle, two signal levels are possible. A dominant level sustains the value of logic 0; a recessive level is logic 1, the assignment of the two levels with respect to the differential voltage between CANH and CANL.

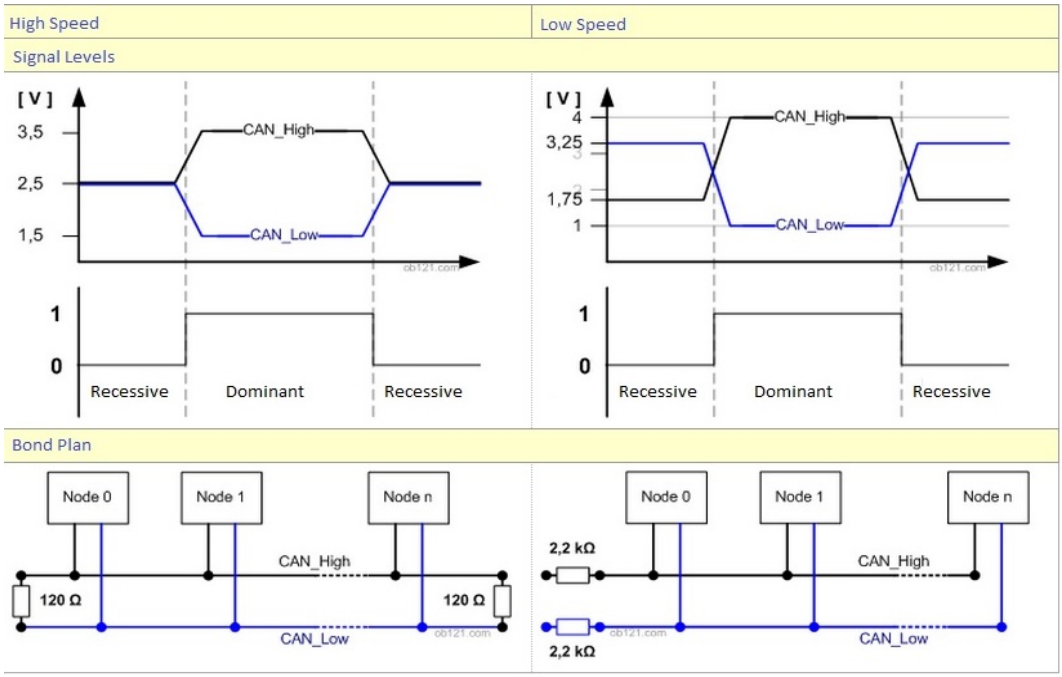


Fig. 2.15: HS and LS CAN Voltage Levels

Through all the mentioned differences between the two types of CAN is a direct coupling of the two systems is not possible. To connect the two networks is the use a gateway to the signals of a network to the changing layer properties of the other network adjusts unavoidable. The operation of these gateways to link different bus systems is based on the OSI model in Fig. 2.16. So is via an adaptation of the different layers the data to be transferred from the outgoing to the receiving bus structure.

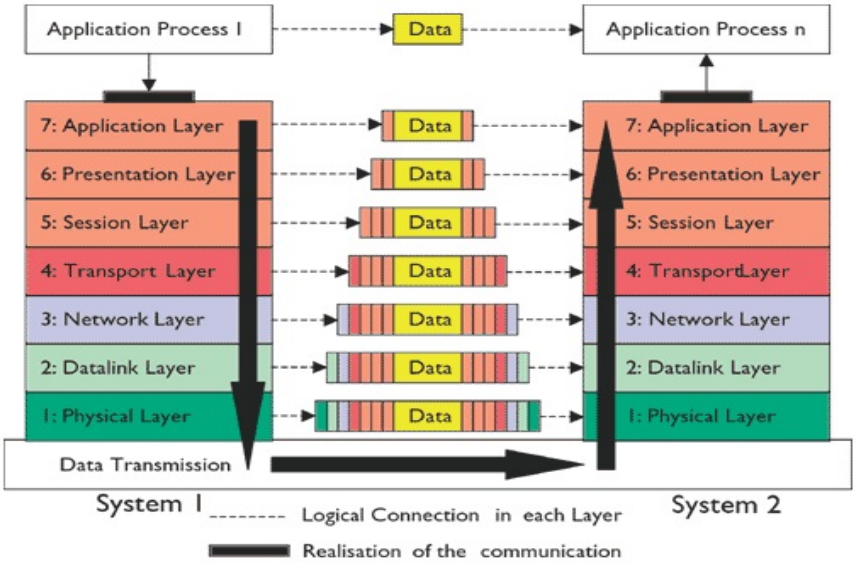


Fig. 2.16: Gateway in the OSI Model [28]

2.2.5 Data Protocol

For transferring messages in the CAN bus, ISO 11898-1 defined something called data frame. Message transfer is manifested and controlled by four different frame types [16]:

Data Frame which carries the data from the transmitter to the receiver

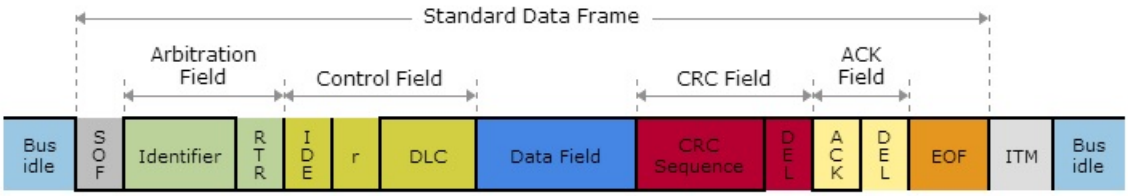


Fig. 2.17: Standard Data Frame [16]

Remote Frame is transmitted by a bus unit to request the transmission of the data frame with the same identifier

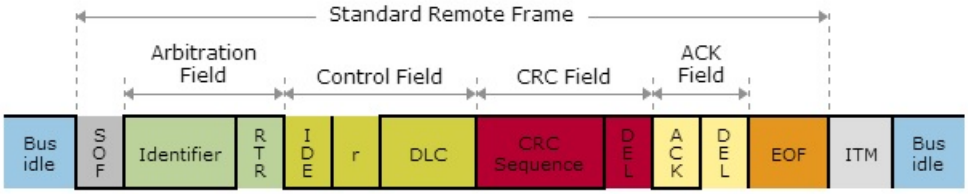


Fig. 2.18: Standard Remote Frame [16]

Error Frame is available to indicate errors detected during communication

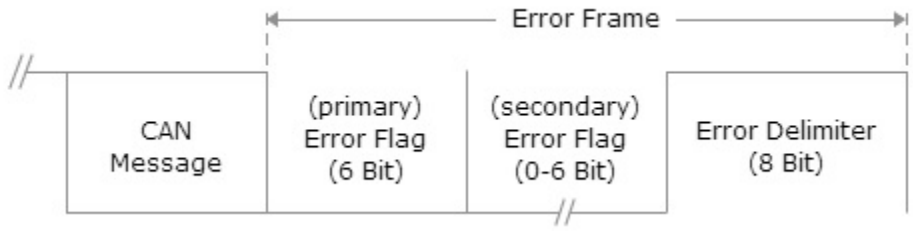


Fig. 2.19: Standard Error Frame [10]

Overload Frame is used to provide for an extra delay between the preceding and the succeeding data or remote frames

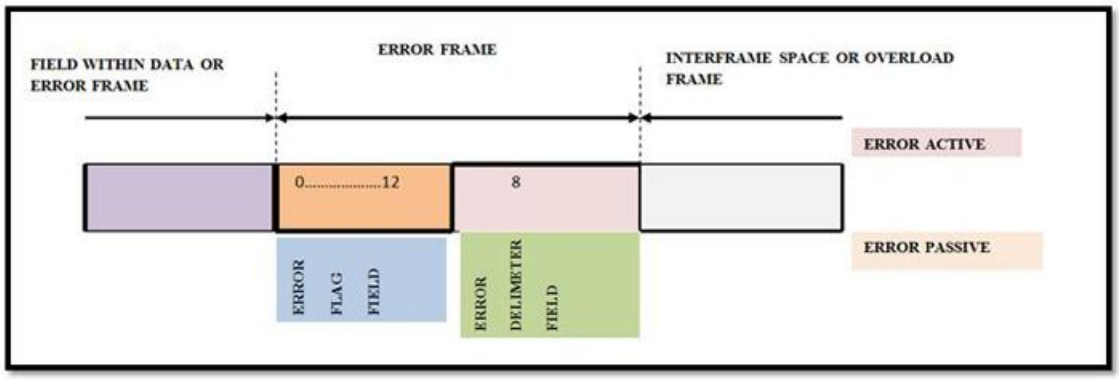


Fig. 2.20: Standard Overload Frame

For more information about the different data protocols of the CAN as well as the different frames, please check it in Appendix B.

2.2.6 Signal Analysis

- **Identifier and Arbitration**

In the CAN bus, all participants are equal, this means that in a random access to the bus, each participant as soon as the bus is free, can send. It sent a message-oriented protocol. A participant on the CAN bus has not, as usual, an address. For the identification of messages its identifier, which is Identifier sent. That is, each message can be received by each participant.

It may happen that several participants begin the transmission process at the same time; though it must be decided who is first to act. The data protocol with the highest priority is sent first. For example, the data log from the ABS control unit is ,for security, more important than from the control unit transmission control. Each data protocol is according to its priority in the status field code (identifier), consisting of 11 bits assigned. Here, each bit has a value which a value is assigned to the Identifier works on the basis of the dominance of the low level. Participants send the Identifier their message, differ to those in a bit. The participant who sends a recessive high level at this moment, noticed that the signal sent by him not correspond to what is now applied on the bus. It stops transmitting until the other party has completed the transmission of their message. The participant with the smaller identifier has the higher priority and sets us apart and is transmitted. Participants with lower priority messages must send their messages then again. It is this process arbitration .

The advantage resulting from this is that receiving stations can be added without having to make changes to the software and hardware. The 11 bits long Identifier when CAN 2048 allows different messages. Each Identifier may only be used simply, that is, only one participant sent are.

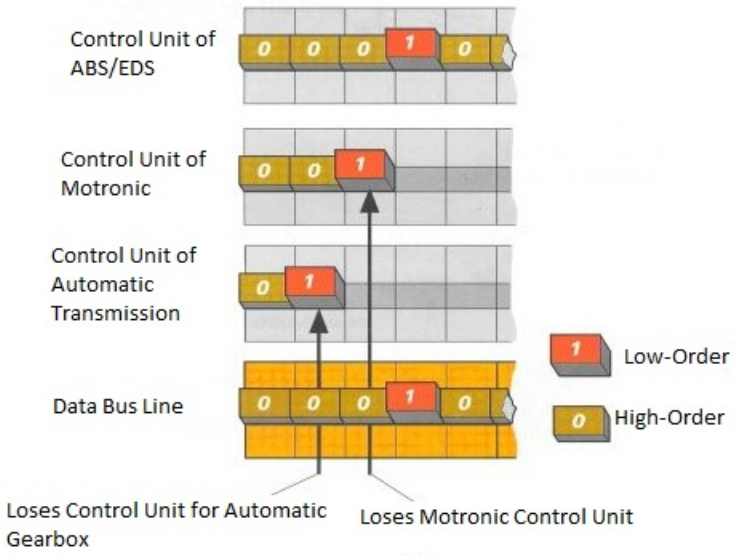


Fig. 2.21: Example for priority of messages [44]

In the above Figure, Fig. 2.21, is an example for the priority of sending the messages on the CAN. So here we have the identifier of the Automatic Transmission has the second bit first the lower value and loses. The next bit is the Motronic control unit sends a recessive high level and eliminated. However, the ABS control unit gains and sends the message.

- **Signal level in the CAN-Bus**

  As described before the high and low signal levels (voltages), below is an example for voltage level changing in a CAN bus for a BMW car.

Voltage Levels K-CAN (BMW)
When the voltage level of the CAN-Low, high (4.0 V) to low changes (1.0 V), that is a logic 0

Changes the voltage level back to high (4.0 V), this is a logical 1

The voltage level of CAN-High is always a mirror image to CAN-Low.

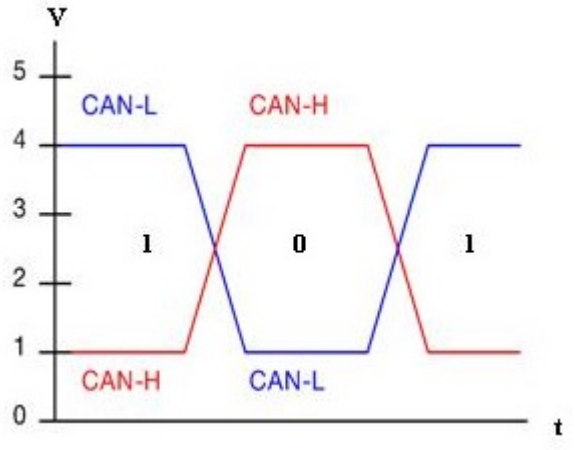


Fig. 2.22: K-CAN Voltage Levels

Voltage Levels PT-CAN (BMW)
In the non-active status of the bus, the bus level of low and high is at 2.5 V (logic 1).

The voltage levels of the CAN bus will be active, substituted - low to low (1.0 V).

The CAN high contrast goes high (4.0 V). This means a logic 0

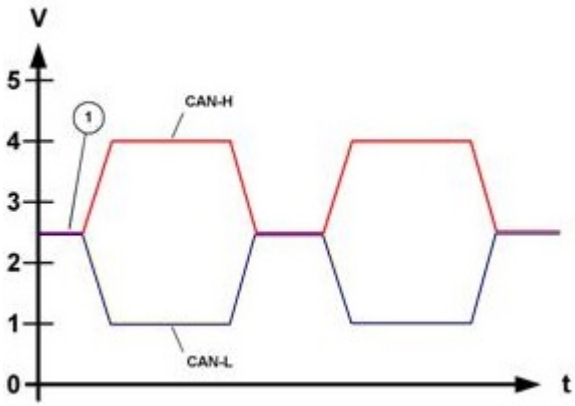


Fig. 2.23: PT-CAN Voltage Levels

2.3 Test Vehicle Citroën C6

For the pracitical implementation of this project, a Citroën C6 V6 HDI 205 car was used. This test vehicle was from HTW Dresden by FSD Company.

Fig. 2.24: Citroën C6 [2]

2.3.1 Overview

The system of this vehicle was studied well, as well as the electronic interface. It was preferable to try the project on this car rather than other cars which were also available in the laboratory, because this car has only 3 CAN Bus Lines, 1 High Speed CAN "CAN IS" ($500KBaud$) and 2 Low Speed CANs "CAN CONF & CAN CAR" ($125KBaud$). This was much easier to get the required CAN messages, thus having the ability to control them by sending and receiving messages via the bus line

More information and general specifications of the car can be found in Appendix C.

2.3.2 CAN connections

As stated above, the C6 Car has 3 CAN BUS Lines. Each of them are connected to different ECUs in the car which perform different functions. So from the below figure, the location of the different ECUs in the car can be shown.

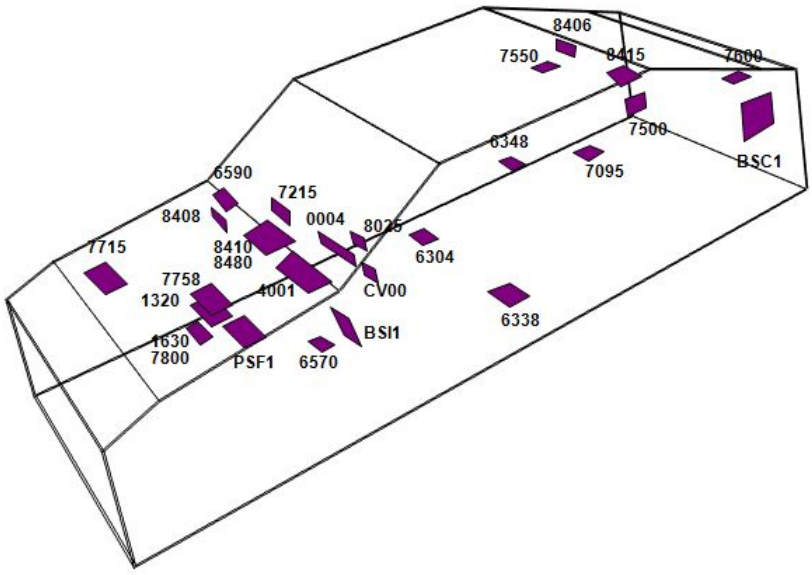


Fig. 2.25: Locations of different ECUs in Citroën C6

Each number in the above figure represent an ECU in the car. For example, 2610 represent the ECU which controls the left headlights, whereas, 6338 is the control unit of the seat storage. For more information and details about the different ECUs, please refer to Appendix C.

Therefore, according to these configurations and specifications, the required information of the car were studied well, and thus recognizing the needed ECUs for the project. For example, the horn and the lights of the car were needed to be used as an action which will be taken when an object is coming towards the light, and so it was realized that these ECUs are numbered: 2110, 2610, 2630, 2635 and CV00. SO that means that if we need to control these functions, we have to send messages on CAN IS and CAN CAR.

2.4 Radar Sensor

2.4.1 Introduction

Automotive radar facilitates various functions which increase the driver's safety and comfort. Exact measurement of distance and relative speed of objects in front, adjoining, or behind the car allows the realization of systems which enhance the driver's ability to perceive objects during bad optical visibility or objects hidden in the blind spot during parking or changing lanes. Radar technology has proved its ability for automotive applications for several years.

Compared to other technologies, radar is very robust technology performing in difficult surroundings. Especially at difficult conditions like darkness or condensation (rain or snow) radar is working reliable. Radar sensor enable the detection of other traffic participants or obstacles in driving situations independent from the surroundings. Radar is available day and night and is working nearly independent from weather conditions. When compared to

its optical counterpart video (with image processing) or radar, the advantages of radar are obvious [8]:

- Direct distance and speed measurement
- Robust against climate impacts and contamination
- Unaffected by light
- Measurement of stationary and moving objects on and in the region of the street
- Invisible integration behind electromagnetically transparent materials (e.g. bumpers).

However, some disadvantages could be:

- Objects detected by the radar sensor can have some reflections
- Small objects like bicycles or pedestrians cannot be detected in a high range, so maybe maximum detection distance could be 10-12 meters
- All objects are detected as points, in contrast with the laser sensor which detects objects with their dimensions; so one can differ if this object is a car or another thing
- At high distance, the range of error in the angle and the distance measured is high

2.4.2 Short Range Radar

24 GHz SRR (Short Range Radar) sensors are used for safety and convenience functions in the automotive industry. The SLR (Sequential Lobing Radar) technique enables the accurate location of objects regarding distance, velocity and bearing of objects in the detection range with a single sensor. By implementing short range Radar sensors into the vehicle, it is now possible to detect objects in front of a bumper in a distance of 10cm to 30m. The large horizontal detection angle (130°) of each sensor allows observing the full length and width of the vehicle with a relative low amount of sensors. The fast recognition situations in the longitudinal and lateral movement of the vehicle is possible with the main focus on the time span just before a collision could happen [12].

Fig. 2.26: Two Radar Sensors attached in the front bumper of a car

In resistance to all harsh environmental conditions as weather, rain and pollution, 24 GHz UWB radar sensors are able to provide object detection and tracking. These sensors are intended for the following applications:

- Parking assistance: Front- and rear-mounted sensors, with a range of 1.8-meters, can detect small objects in front of large objects and measure direction of arrival
- Pre-crash sensing: The sensors scan out up to 30 meters to provide an advanced warning of an imminent collision to arm airbag, pre-tension seat restraints or other injury mitigation strategies
- Blind spot detection: Short-range sensors detect objects in critical zones
- Stop & Go: Detection of objects in front of the vehicle thus enabling automatic acceleration and braking in urban traffic

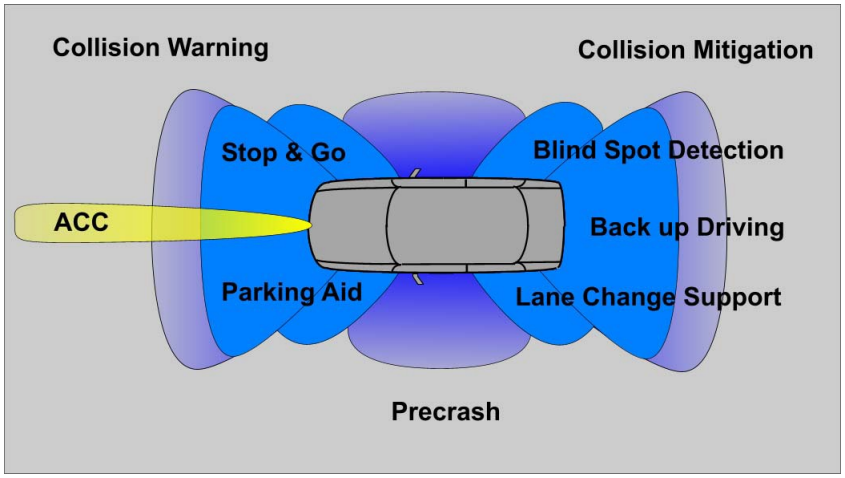


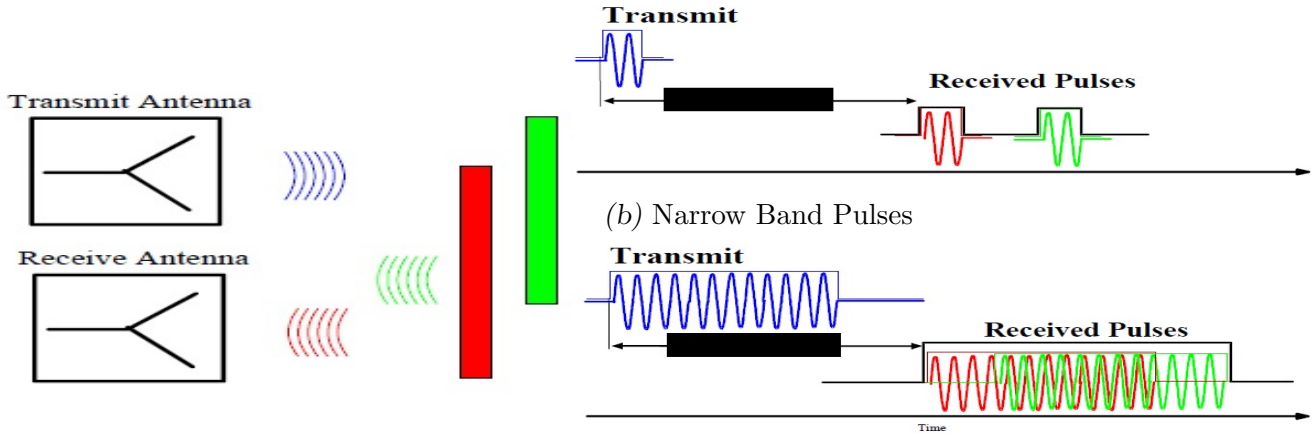
Fig. 2.27: Overview of Automotive Applications [12]

2.4.3 Ultra Wide Band UWB

The need and request for driver assistance and active safety applications is increasing significantly. The functions shown in Fig. 2.27 require varying sensor output performance. Various suppliers and research groups offer and report sensors based on various principles and with very individual performance. Either single sensors or sensor networks can accomplish environment perception around the car. One important parameter with severe influence on the functionality is the operating frequency bandwidth. The competing systems are narrow band and UWB radar sensors. Because of the bandwidth that is used, both systems will show different behaviour concerning range resolution [32].

So why Ultra Wide Band (UWB) system is more preferable?

UWB provides the ability to distinguish closely spaced objects.



(a) Antenna [32] *(c)* UWB Pulses

Fig. 2.28: Transmission and Receiving of Pulses

The need for a UWB bandwidth is given by the application. E.g. the prediction of a possible collision needs a reliable object tracking capability. Thus, SRR has to have sufficiently high range resolution in the centimetre range to detect smaller objects and vulnerable road users such as motor cyclists and children, so UWB systems with high resolution detect small objects in front of large objects.

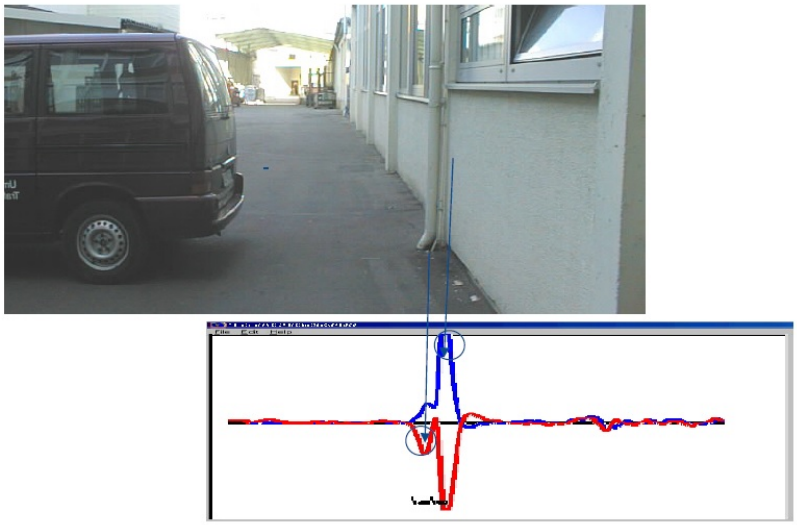


Fig. 2.29: UWB Radar Resolution [32]

2.4.4 Principle of Operation

SLR - Range Measurement

In principal, radar can work as a pulse radar system or CW (Continuous Wave). Radar system whereas the exactness of radar is dependent on the bandwidth. Pulse radar is using

periodic sequences of short pulses as transmit signal. The measured distance to objects is calculated from the time difference between transmitted and received signals [35].

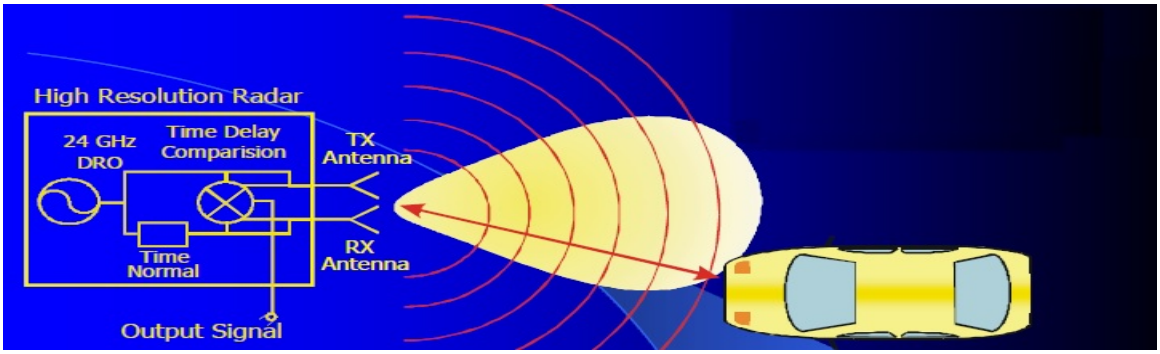


Fig. 2.30: Range Measurement [35]

Pulse Radar Correlation
Minimum range: 15cm
Maximum range: 20m (10sqm)
Object Resolution: 15cm

Correlation
Correlation occurs when the pulse travelling time is equal to the internal delay. The resulting IF signal is related to the range of the detected object.

Time Delay Comparison
Pulse travelling time is compared with internal time delay.

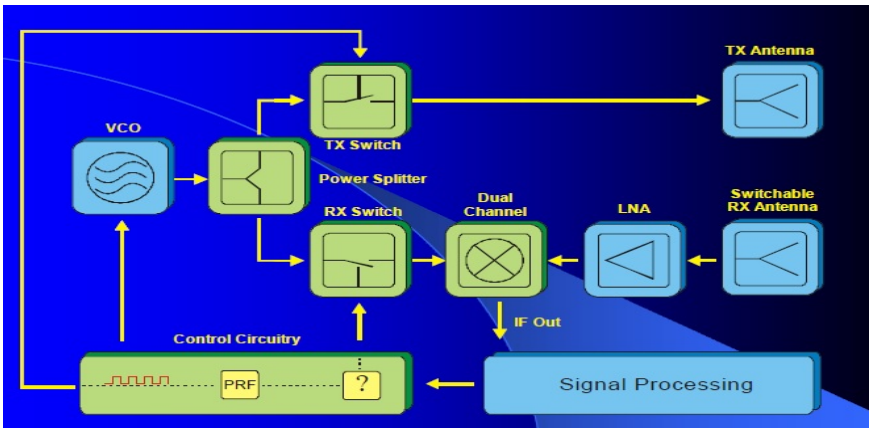


Fig. 2.31: Block Diagram - SLR Range Measurement [35]

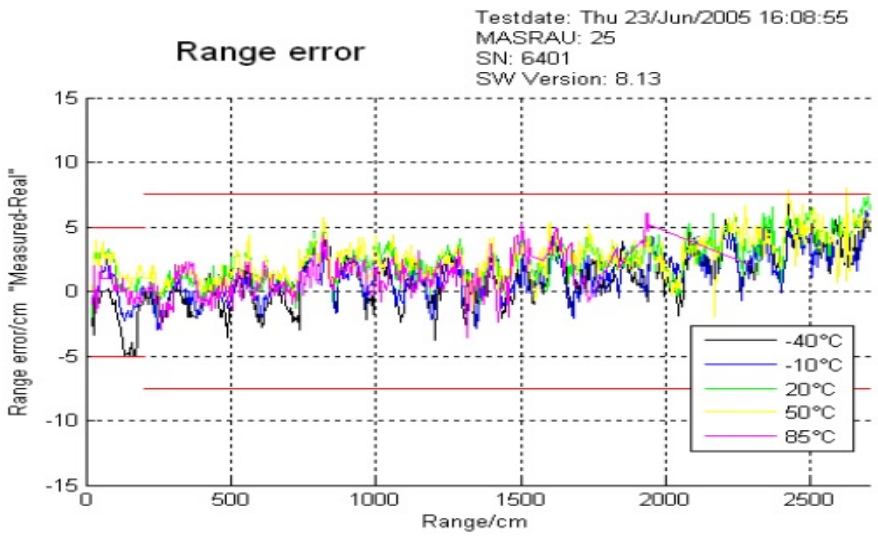


Fig. 2.32: Range Accuracy Performance [32]

SLR - Angle Measurement

Key component of any angle reporting radar is the antenna. Conventional approaches for electrical beam scanning concepts require relatively large antenna apertures which is due to space constraints behind a car bumper disadvantageous. Mechanical scanning approaches are due to cost reason not a competitive solution for short range radar sensors.
Tyco Electronics therefore developed the radar sensor based on the Sequential Lobing concept. By switching between two different antenna patterns, it is possible to detect the bearing angle of objects accurately. The sensor switches back and forth between the two antenna characteristics [35].

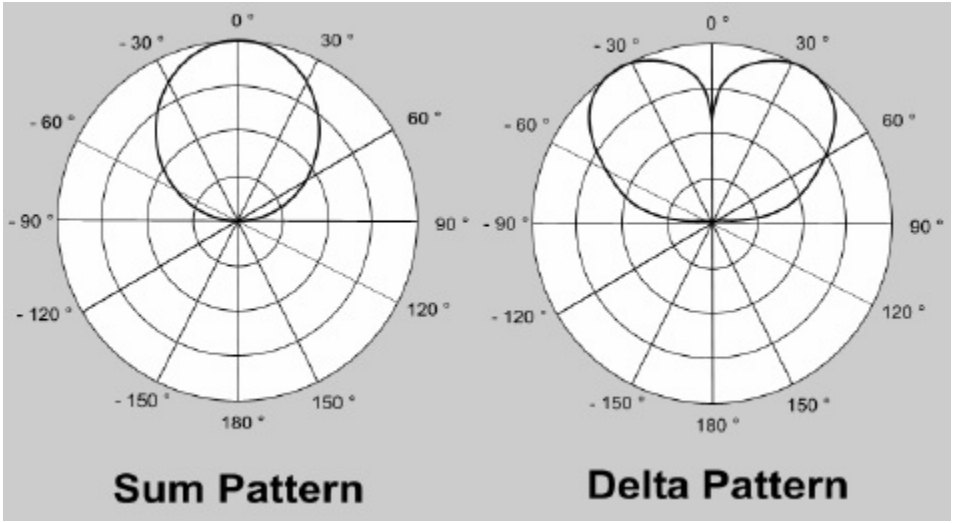


Fig. 2.33: Antenna Patterns [35]

Through the mono-pulse theory it is possible to extract bearing information by processing

the return signals of both sensor patterns.

So the technology approach works as follows:

- Evolved from existing HRR implementation
- Sequential Lobing Antenna
  - Reference Beam
  - Error Beam
- Both beams are sequentially processed in the same front-end hardware channel
- Target angle is determined by the amplitude and phase relationship between the Ref and Err signals

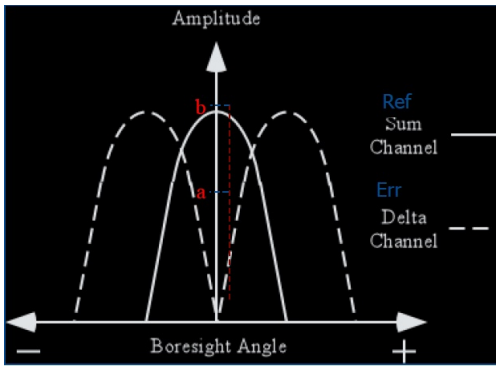


Fig. 2.34: Angle Measurement [32]

$$Ratio = \frac{a}{b} \tag{2.1}$$

Left/Right side determined by phase difference

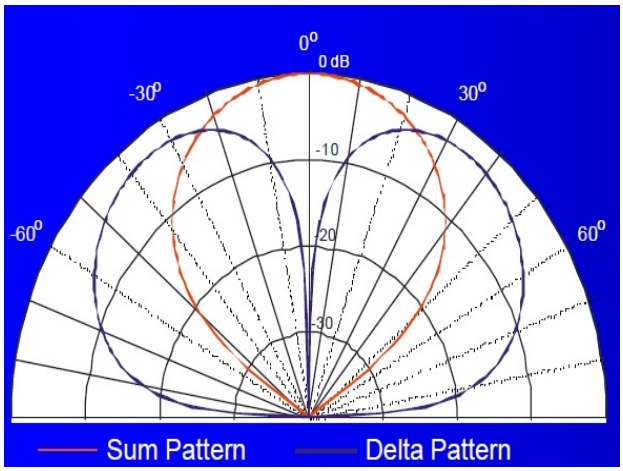


Fig. 2.35: Patterns [35]

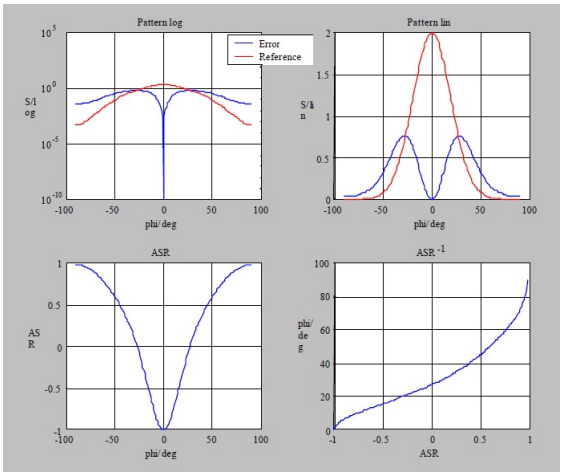


Fig. 2.36: Sensor Bearing Capability

So the angle can be:

$$ASR = \frac{S_{err} - S_{ref}}{S_{err} + S_{ref}} \tag{2.2}$$

- S_{err}: Magnitude of delta pattern
- S_{ref}: Magnitude of sum pattern

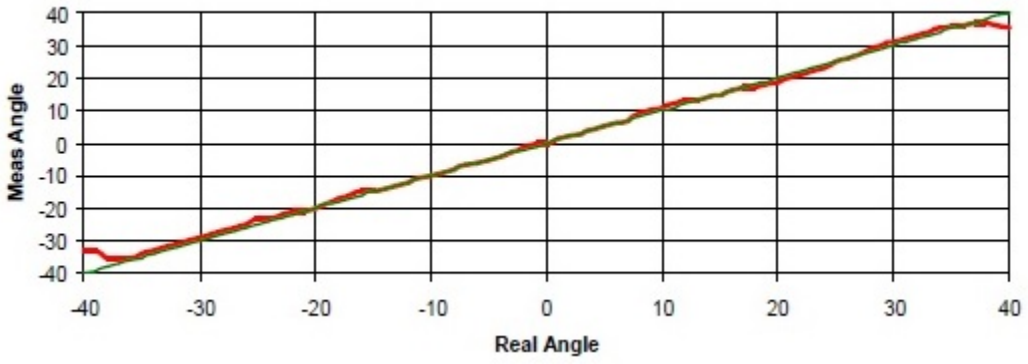


Fig. 2.37: Angle Accuracy [35]

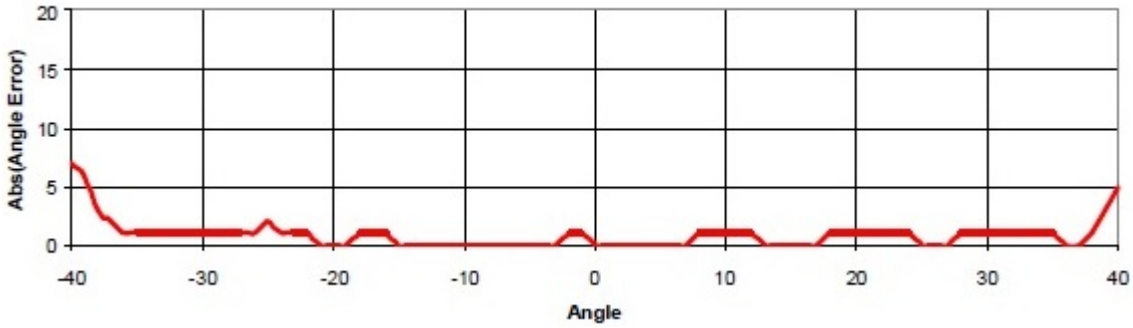


Fig. 2.38: Absolute Angle Error [35]

SLR - Velocity Measurement

In order to measure the object's velocity, Pulse-Doppler radar is used. This frequency modulated CW radar uses a transmit signal which is linear changed in frequency.
Besides the object detection intelligent radar sensors - as the SLR sensor - enable object tracking. Appropriate filter techniques suppress ghost objects. The detected objects are transferred cyclic via a bus system as object data for the subsequent processing in a supervisory control unit [35].

For more technical information about the Radar Sensor, please refer to Appendix D.

2.5 Global Positioning System (GPS)

The Global Positioning System (GPS) is a space-based satellite navigation system composed from a network of 24 satellites located in the orbit by the U.S. Department of Defense (DOD). This system provides location and time information in all weather conditions, anyplace on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. There are no subscription fees or setup charges to use GPS, so it is freely accessible to anyone with a GPS receiver [13].

2.5.1 Basic Structure and System Overview

For calculating GPS data, the system uses 3 main segments [39]:

1. Space Segment

   The space segment (SS) is composed of the orbiting GPS satellites or Space Vehicles (SV) in GPS parlance. The GPS space segment consists of a group of satellites sending radio signals to users. The United States is devoted to maintaining the availability of at least 24 operational GPS satellites, 95% of the time.
   GPS satellites fly in medium Earth orbit (MEO) at an altitude of approximately 20,200 km (12,550 miles). Each satellite circles the Earth twice per day. These satellites are travelling at speeds of approximately 7,000 miles an hour. This was very helpful during the development because the orbits are arranged and were correctly aligned so that at anytime, anywhere on Earth, there are at least four satellites "visible" in the sky.
   GPS satellites are power-driven by solar energy. They have backup batteries onboard to maintain them running in the event of a solar eclipse, in the case of no solar power. Small rocket boosters on each satellite keep them flying in the correct path.
   The satellites in the GPS constellation are arranged into six equally-spaced orbital planes surrounding the Earth. Each plane contains four "slots" engaged by baseline satellites. This 24-slot arrangement guarantees users can view no less than four satellites from almost any point on the planet.

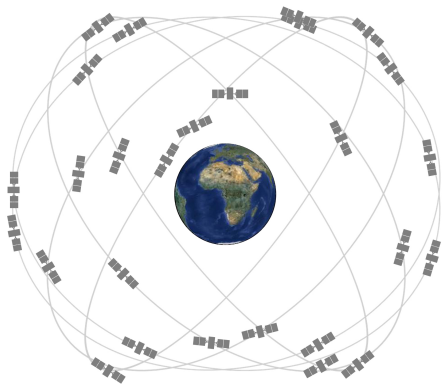


Fig. 2.39: Space Segments around the Earth [13]

2. Control Segment

   The GPS control segment consists of a global network of ground facilities that follow the GPS satellites, observe their transmissions, execute some analyses, and send commands and data to the constellation.

   This segment consists of 3 main stations:

   (a) **Master Control Station(MCS)**
   It receives navigation information from the monitor stations, utilizes this information to calculate the exact locations of the GPS satellites in space, and then uploads this data to the satellites.

   (b) **Monitor Stations**
   They track the GPS satellites as they bypass in the clouds and channel their observations back to the master control station.

   (c) **Ground Antennas**
   They are used to communicate with the GPS satellites for command and control purposes.

   The present operational control segment includes a master control station, an alternate master control station, 12 command and control antennas, and 16 monitoring sites.

3. User Segment

   The user segment consists of the GPS receiver equipment, which receives the signals from the GPS satellites and uses the transmitted information to compute the user's three-dimensional position and time.

2.5.2 Theory of Operation

A GPS receiver calculates its position by accurately timing the signals sent by GPS satellites high above the Earth. Each satellite repeatedly transmits messages that include [11]:

- The time the message was transmitted
- Satellite position at time of message transmission

GPS satellites circle the earth twice a day in a very precise orbit and send out signal information to earth. GPS receivers attain this information and use triangulation to compute the user's exact location. Basically, the GPS receiver compares the time a signal was transmitted by a satellite with the time it was received. The time difference informs the GPS receiver how far away the satellite is.

So it works as follows:

1. GPS satellites broadcast radio signals providing their locations, status, and precise time (t1) from on-board atomic clocks.

2. The GPS radio signals travel through space at the speed of light (c), more than 299,792 km/second.

3. A GPS device receives the radio signals, noting their exact time of arrival (t2), and uses these to calculate its distance from each satellite in view.

4. Once a GPS device knows its distance from at least four satellites, it can use geometry to determine its location on Earth in three dimensions.

To calculate its distance from a satellite, a GPS device applies this formula to the satellite's signal:

$$distance = rate * time \tag{2.3}$$

where rate is (c) and time is how long the signal travelled through space.
The signal's travel time is the difference between the time broadcast by the satellite (t1) and the time the signal is received (t2).

A GPS receiver must be locked on to the signal of at least three satellites to calculate a 2D position (latitude and longitude) and track movement. With four or more satellites in view, the receiver can determine the user's 3D position (latitude, longitude and altitude). Once the user's position has been determined, the GPS unit can calculate other information, such as speed, bearing, track, trip distance, distance to destination, sunrise and sunset time and more.

So if we have 3D position from the GPS, we can transform the longitude, latitude and altitude as distances in meters from the center of the earth. For example, if we have this point as shown in the following figure

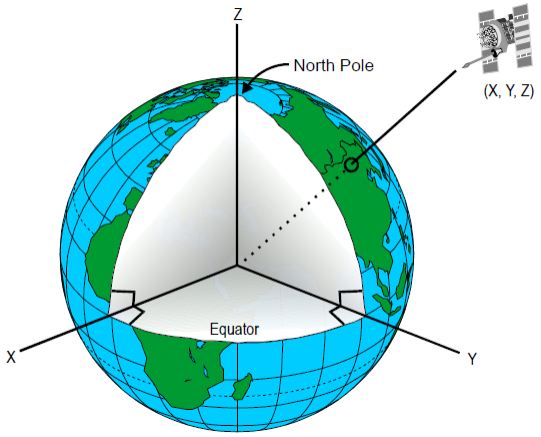


Fig. 2.40: Object located somewhere on the Earth using satellites [29]

According to the definition of longitude, latitude and altitude and by using trigonometry,

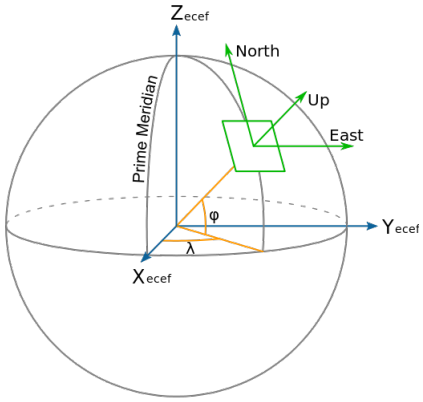


Fig. 2.41: Longitude and Latitude of an object [31]

the following formula can be used to calculate the distance:

$$\begin{aligned} x &= R * cos(\phi) * cos(\lambda) \\ y &= R * cos(\phi) * sin(\lambda) \\ z &= R * sin(\phi) \end{aligned} \tag{2.4}$$

where

R is the radius of earth $\approx$ 6,371 kilometres

λ is the Longitude

ϕ is the Latitude

So from these 3 equations, one can get the x, y and z coordinates which represent the exact distance of the object from the center of the earth

2.6 iBeacon

2.6.1 Introduction

iBeacon is a new Apple's technology providing location-based information and services. Using Bluetooth Low Energy (BLE), a device with iBeacon technology can be used to establish a region around an object. This allows an iOS device to determine when it has entered or left the region, along with an estimation of proximity to a beacon.

2.6.2 Theory of Operation

The iBeacons are small, cheap Bluetooth transmitters. Apps installed on iPhone or iPad can listen out for the signals transmitted by these iBeacons and respond accordingly when the device comes into the range.

The Core Location framework provides two ways to sense a user's access and exit into precise regions: geographical region monitoring (iOS 4.0 and later and OS X v10.8 and later) and beacon region monitoring (iOS 7.0 and later). A geographical region is an area defined by a circle of a specified radius around a known point on the Earth's surface. On the contrary, a beacon region is an area defined by the device's proximity to Bluetooth low-energy beacons [25].

- **Geographical region monitoring** uses location services to detect entry and exit into known geographical locations. You can use this capability to generate alerts when the user gets close to a specific location or to provide other relevant information.
- **Beacon region monitoring** uses an iOS device's onboard radio to detect when the user is in the surrounding area of Bluetooth low-energy (iBeacon). As with geographical region monitoring, you can use this ability to generate alerts or to supply other significant information when the user enters or exits a beacon region. Rather than being recognized by fixed geographical coordinates, conversely, a beacon region is identified by the device's proximity to Bluetooth low-energy beacons that broadcast a blend of the following values:
  - **UUID** a value that uniquely identifies one or more beacons
  - **Major** an integer that can be used to group related beacons that have the same proximity UUID
  - **Minor** an integer that differentiates beacons with the same proximity UUID and major value

2.6.3 Signals

The iBeacon keeps sending out radio signals, until it is detected by an iOS device which is identifying this iBeacon. Once detected by the device, using an app installed on this smart device, one can know the exact location of the detected iBeacon in the surrounding region as a GPS coordinate point and the distance between it and the device as well as proximity.

For this option, iOS uses a process called "ranging". Based on common usage scenarios, the device applies some filters to the accuracy estimate to determine the estimated proximity to a detected beacon. This is indicated by four proximity states:

Tab. 2.5: Proximity States of an iBeacon

Proximity State	Description
Immediate	This represents a high level of confidence that the device is physically very close to the beacon. Very likely being held directly up to the beacon. "Within a few centimetres"
Near	With a clear line of sight from the device to the beacon, this would indicate a proximity of approximately 1-3 meters
Far	This state indicates that a beacon device can be detected but the confidence in the accuracy is too low to determine either Near or Immediate. "Greater than 10 metres away"
Unknown	The proximity of the beacon cannot be determined. This may indicate that ranging has just begun, or that there are insufficient measurements to determine the state

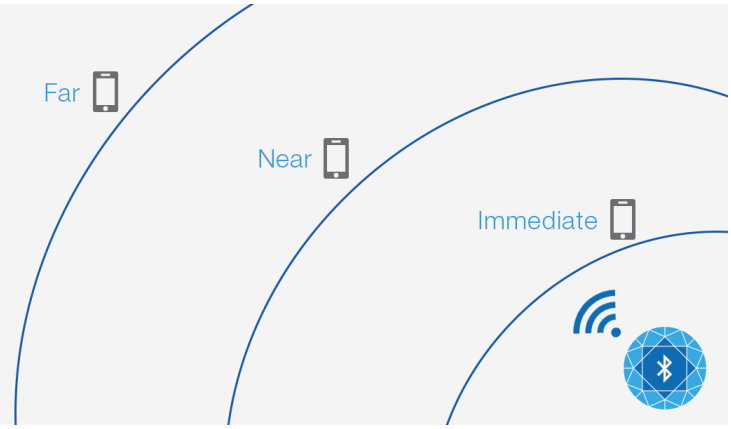


Fig. 2.42: Different levels of interactions of iBeacon at each range [25]

2.7 FlexDevel Controller

In order to control the functions of the car used and thus give actions like the horn or light or even an artificial alarm signal, a controller must be used for this purpose. Therefore, my decision was based on choosing a controller that does not only have different outputs as digital, analog and PWM, but only has an interface which can connect with CAN Bus systems and be able to send and receive messages from the CAN. Hence, the choice came to use FlexDevel Controller.

2.7.1 Introduction

FlexDevel is an evaluation board which acts as a universal micro controller development platform to test and use the most common car networks for automotive like FlexRay, CAN and LIN bus systems. This platform makes it simple to build FlexRay, CAN and LIN systems. A large number of interfaces allow solutions for a wide range of applications. The host

processor of FlexDevel is programmed via an easy-to-use function library (C-code). The FlexDevel platform can be used wherever there is a demand for quick, simple introduction of bus systems like FlexRay, CAN or LIN. It is thus predestined for universities that deal with automotive bus systems in more detail. In development labs, the FlexDevel hardware and software package serves as an introduction or base for a number of solutions in the area of automotive bus communications.

For more technical information about the FlexDevel Controller, please refer to Appendix E.

CHAPTER 3

Experimental Work

3.1 CAN Messages

The first step in the project was to significantly understand how someone can be able to send and receive messages. After the understanding of the CAN network and communication and the signal analysis, it was important to clearly recognize the way of the message transmission.

3.1.1 CANalyzer

CANalyzer is a worldwide development tool for CAN bus systems, which can help in observing, analyzing and supplementing data traffic on the bus line. Due to its powerful functions and user-programmability, all requirements are covered - from initial trial runs with CAN to selective error localization in complex problems [30], [15].

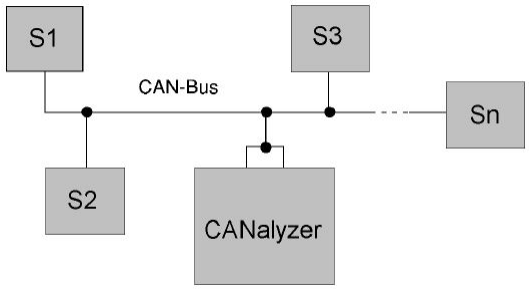


Fig. 3.1: CANalyzer linking between different stations on the CAN-Bus [15]

Even the fundamental built-in functions - which can be used without any programming information - supply for a plenty of possible applications. These functions such as listing bus data traffic (Tracing), displaying data segments of specific messages, transmitting predefined messages and replaying recorded messages, statistically evaluating messages, and acquiring statistics on bus loading and bus disturbances, in addition to recording messages for replay or offline evaluation.

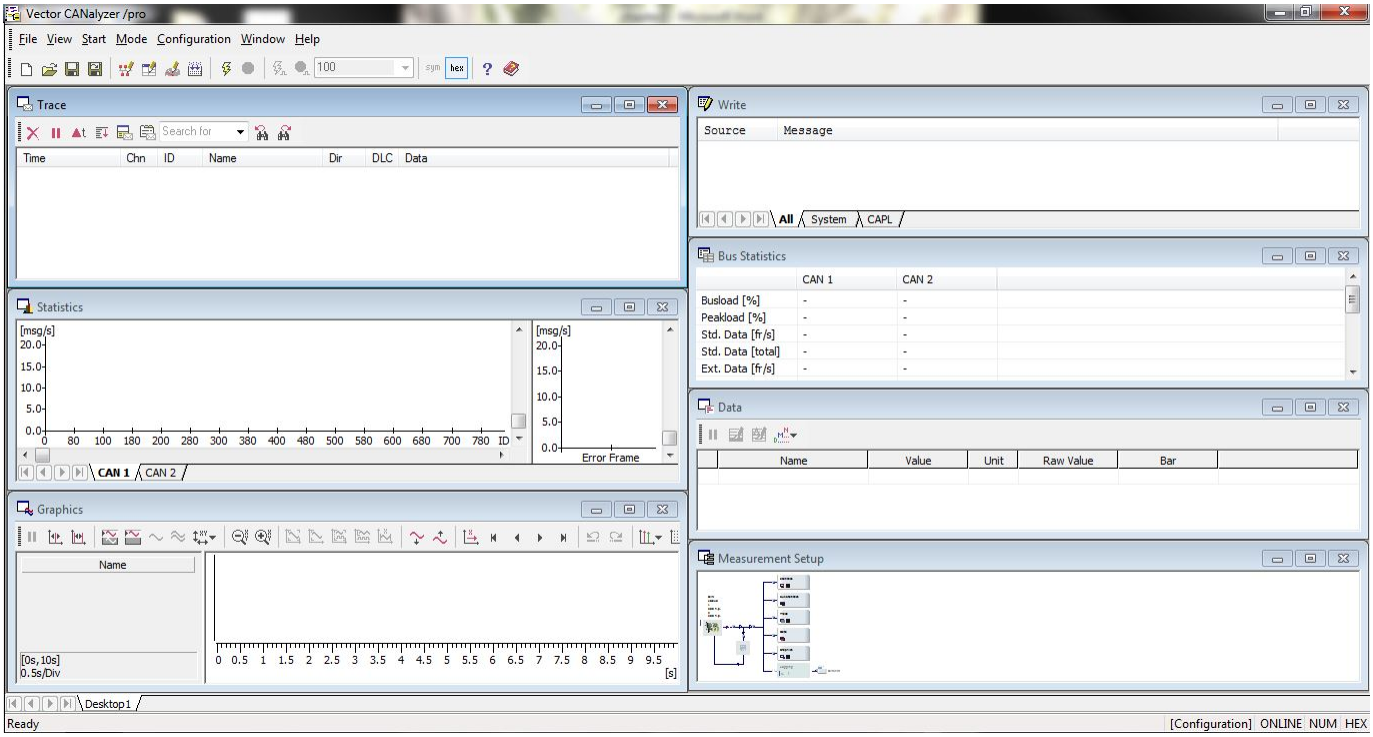


Fig. 3.2: CANalyzer different functions

From the above figure, it is well shown that CANalyzer has diverse functions as stated above. So through this software, one can be able to trace the CAN messages and visualize them, and even draw some graph to see the statistics of these signals.

Furthermore, there is a user-friendly database management program called CANdb++ which is included with the CANalyzer; it can be used to store the signals from the motor vehicle in a database which can be accessed later.

3.1.2 Setting up a connection

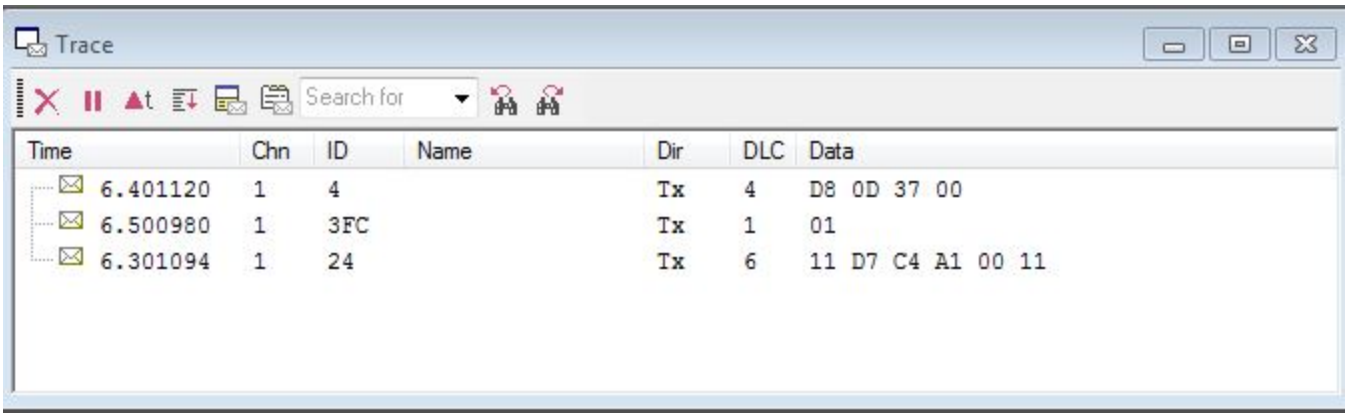
To start up with CANalyzer, it is necessary to set up a connection between the pc and the CAN Bus using a special USB interface called CANcaseXL. This is an interface with a powerful 32 bit 64MHz microcontroller which can process CAN messages with either 11-bit or 29-bit identifiers. It can also be possible to receive and analyze Remote Frames without any limitations. This interface can be connected to two different channels at the same time, which means that we can connect two CAN Lines in parallel.

Fig. 3.3: CANcaseXL

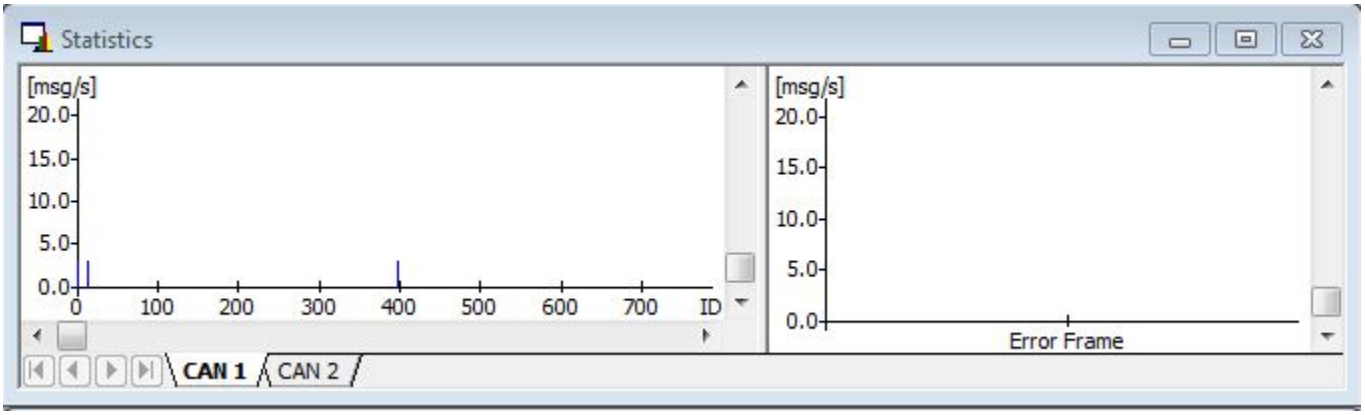
In order to set up this connection, we need a CANcaseXL, CAN cable (male-female or female-female), and a USB A/B. The CAN cable is connected from the case to the BUS Line while the USB cable is used to transfer the signals from and to the PC to be analysed on the CANalyzer.

3.1.3 Testing CAN-Transmission

After getting to know how to use the CANalyzer and how to set up the connection, initial simple tests with transmitting signals were done. 3 messages with IDs 4, 3FC, and 24 were generated to be transmitted from Channel 1 to Channel 2 every 100 ms, then they have been traced to see the transmission.



(a) Trace Window



(b) Statistics Window

(c) CAN Case LEDs

Fig. 3.4: Testing Signals

As shown in Fig. 3.4c, that channel 1 Tx LED is lighting which means that it is the channel

which is sending the signal whereas channel 2 Rx LED is lighting which informs us that this is the receiving channel.

3.2 Test Vehicle

It was very important after the understanding of the CAN messages and the car electronic interface to try to control some features in the car using this concept.

Actually, in the very beginning, I started working on another test vehicle in the laboratory which was S500 Mercedes-Benz W221.

Fig. 3.5: Mercedes-Benz S500

This car is different from the C6 car, as it has 8 CAN Bus Lines (2 Low-Speed and 6 High-Speed), which was considered to be a little bit difficult to find the signals required. I worked on this car and tried to detect the signals of the lights, horn and the doors, and then I created a database for each CAN. After then, I worked on sending my own messages on the CAN to be able to control these features by myself. However, this was not working, and there was a problem in manipulating the CAN by myself. I used another approaches but also I did not get any response. For further details about the CAN Bus Lines in the Mercedes S500, please refer to Appendix F

Therefore, I had to switch to the other car which was the Citroën C6 as it was much easier to manipulate the CAN with my own signals.
The same tests started on the car to try to get the messages of the required features to be controlled. And this was successfully done, then I tried to send my own messages on the BUS to turn on the horn or the lights when an occurrence happens, and it also worked. So this was the first step of the practical framework of the project, which was to have the ability to control the required output.

3.3 Radar Readings

3.3.1 Connection to PC

Now, the concept of analysing the radar signals is known, as well as how to receive these signals on the CAN. So the next step was starting to work with the radar sensor and try to receive readings from it and analysing it to be able to detect the distance and the velocity of a detected object.

The radar was connected with a cable directly to the CAN case then to the pc. By using CANalyzer, I was able to log these readings and creating a logged file from them, then analysing them.

3.3.2 Logging Data

In the below figure 3.6, is some signals from the logged data which have been taken from the radar.

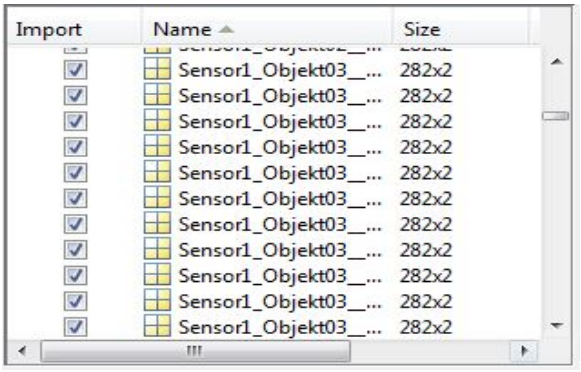


Fig. 3.6: Radar Logged Data

For the sensor, 10 objects are detected. For each object, there are some signals which are: Timestamp, Object ID, Distance, Angle, Velocity, Yawrate, SNR and counter.

3.3.3 Analyzing Radar Readings

By using a created GUI on MATLAB, the objects were represented as dots on x and y axis with their exact coordinates from the 0-position which is the position of the radar sensor, and a line showing the direction of movement of the object. Therefore, I made some tests and measurements with the sensor and plotted the readings to realize how the objects can be detected by the sensor and to know the accuracy of detection too.

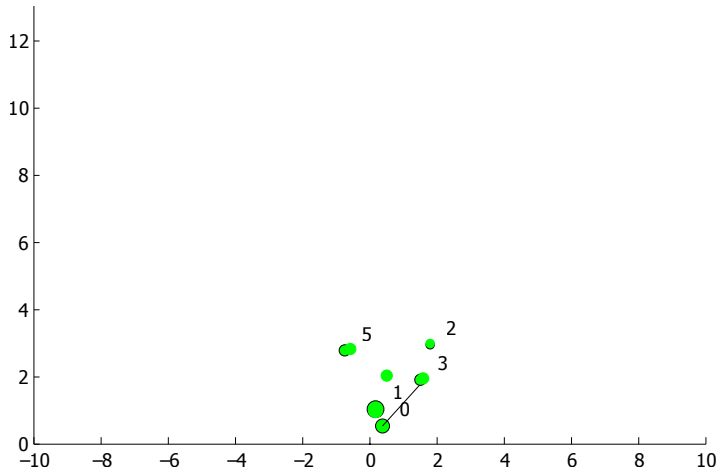


Fig. 3.7: Plot of Radar Objects (1 Sensor)

However, in this project, 2 sensors were used. One is to be put on the left side of the rear bumper, and the other one is to be put on the right side. So below is the GUI model showing the detected objects of the 2 sensors.

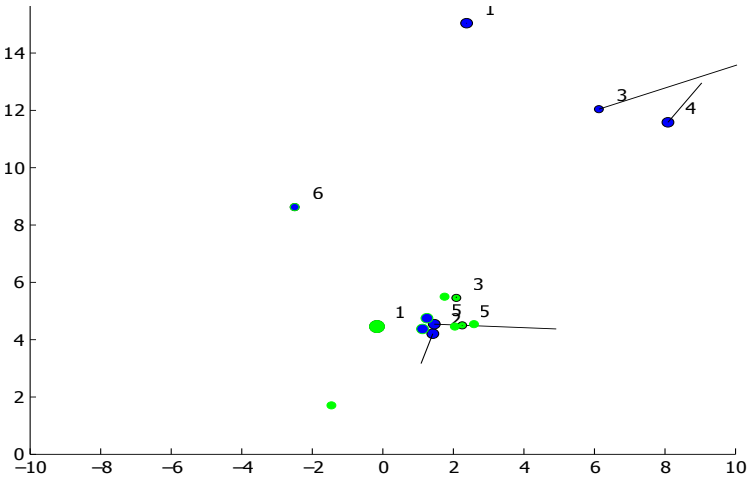
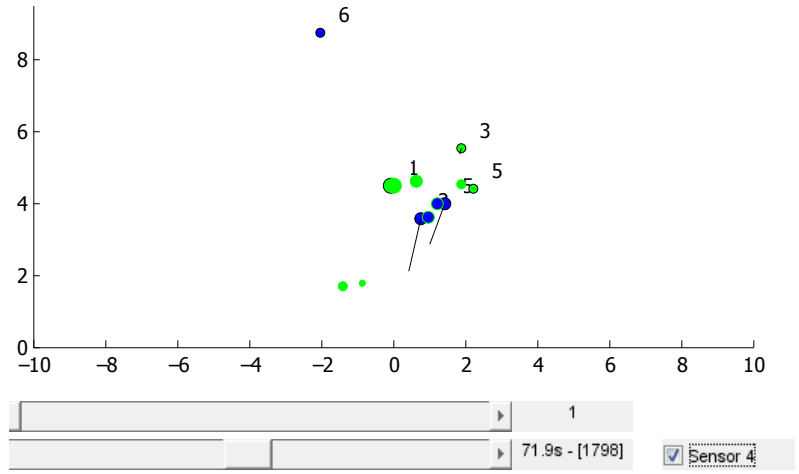
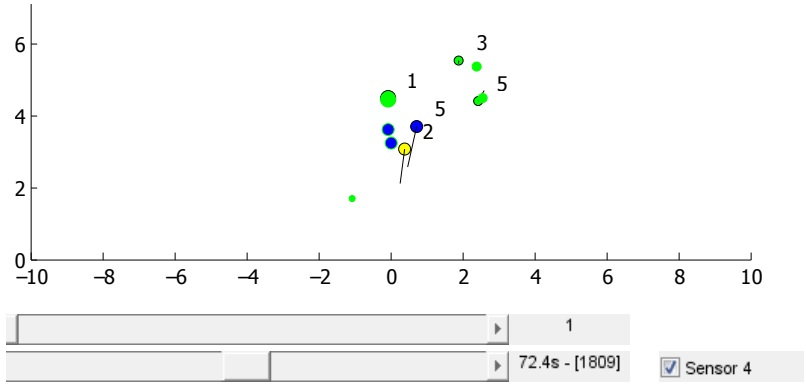


Fig. 3.8: Plot of Radar Objects (2 Sensors)

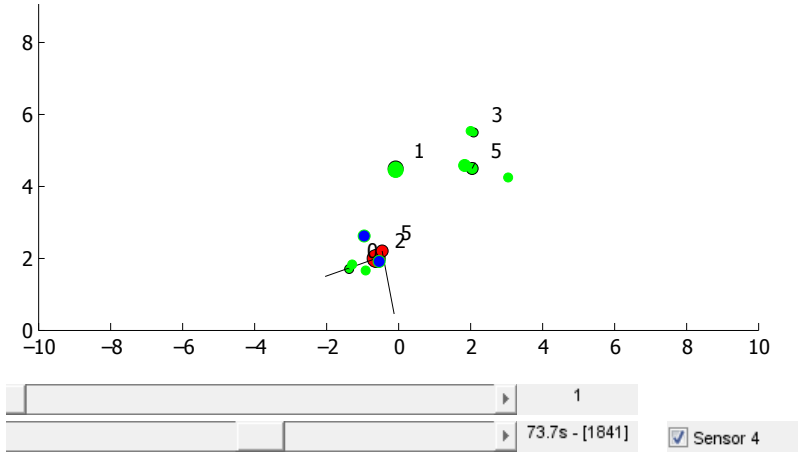
Based on the main idea of the project which was creating alarm signals for the bicycles that are coming towards the car and are in critical situation, some amendments to the GUI were implemented to generate warnings for the critical objects. So according to the distance of the object, the velocity and the time needed for it to collide with the car, different warning signals were given out, which were represented in the GUI as changes in the color of the objects. So in figure 3.9, there is an example for object coming towards the sensor, and how its status is changing from not critical to critical.



(a) No Warning



(b) Medium Warning



(c) High Warning

Fig. 3.9: Different Levels of Warnings

Because the main idea of the project was to detect the critical situations for the car and thus giving alarm signals, it was important to try this as a virtual output, not just in simulation. So as stated before that I have the signals of the lights and horn in the car, thus, I started implementing what I have reached in simulation to real life. I used these signals as an output

for the car in case of a bike near from it. So when it is critical, the lights of the car keep lighting at a certain frequency, and when the situation is very critical, the frequency increases as well as the horn also turns on.

Fig. 3.10: Rear lights turn on when the bike is coming

Moreover, we used some artificial lights (LEDs) attached to the mirrors in case of very critical situations. So these lights turn on when the bike is about to crash the car, thus it makes an alarm to the driver to take an action; for example, not to open the door.

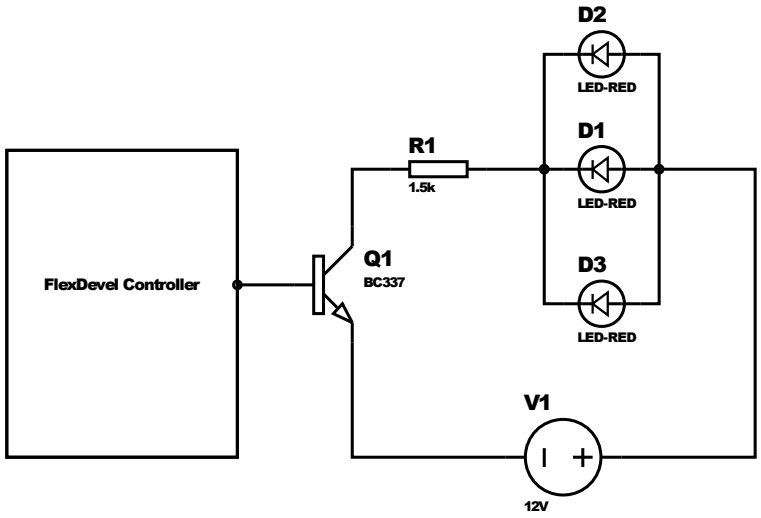


Fig. 3.11: Proteus Design for the artificial LEDs

Fig. 3.12: Artificial LEDs in Mirror

3.4 GPS Readings

The next approach which was used in the project to detect the critical objects was the GPS. But it was very essential to clearly understand how to understand the GPS readings. Due to what was stated above in Chapter 2, that the GPS gives us only the location as longitude and latitude in degrees, however in comparing these data with the Radar data which outputs distance in meters, won't be practical unless the significant analysis of the GPS readings.

Accordingly, I have started to work on reading the GPS data and analysing them.

3.4.1 GPS readings from Car

In the beginning, I started with taking GPS readings from the car. And for this step, GPS Receiver (GR-213) was used. GR-213 Smart GPS Receiver is a total solution GPS receiver. It has the ability to communicate with other electronic utilities via compatible dual-channel through RS-232 or TTL and saves critical satellite data by built-in backup memory which can be accessed later. The GR-213 tracks up to 20 satellites at a time, re-acquires satellite signals in 100 ms and updates position data every second.

The GR-213 sends valid navigation information over output channels. These data include:

1. Latitude/longitude/altitude
2. Velocity
3. Date/time

4. Error estimates

5. Satellite and receiver status

So by using this receiver, I was able to log some readings from the Car while it is moving. So I did the initial steps, which was taking the readings from the GPS receiver and analyzing them.

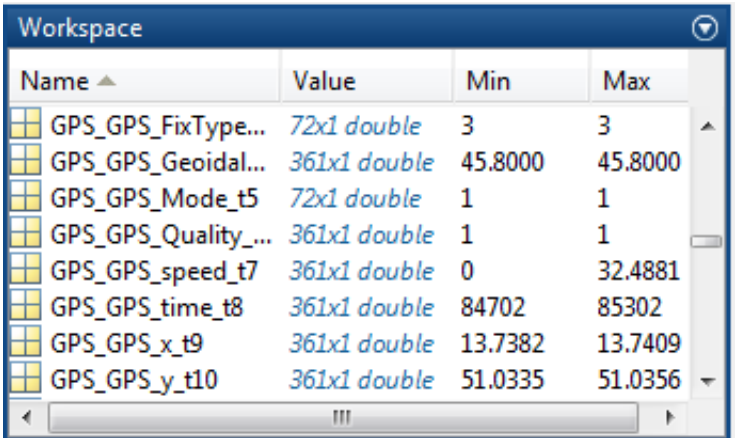


Fig. 3.13: GPS Car Logged Data

As seen from Figure 3.13 that from these readings, we can be able to know the x,y and z position of the car, which are the longitude, latitude and altitude, thus be able to transform them into meters to get the distance. As well as, knowing the speed and the course of the car, which can be used further in the project.

3.4.2 GPS readings from Bike

The next procedure was to do the same with the bike. So I needed also to get the GPS data from the bike during its path, however, in this step, I did not use a GPS receiver. Simply, there is an application which was developed here in the laboratory which can be installed on iOS devices and can be able to give readings of GPS, accuracy, velocity and course. Therefore, this application was installed on an ipad, and the driver of the bicycle started logging and drove the bicycle with this ipad in his bag.

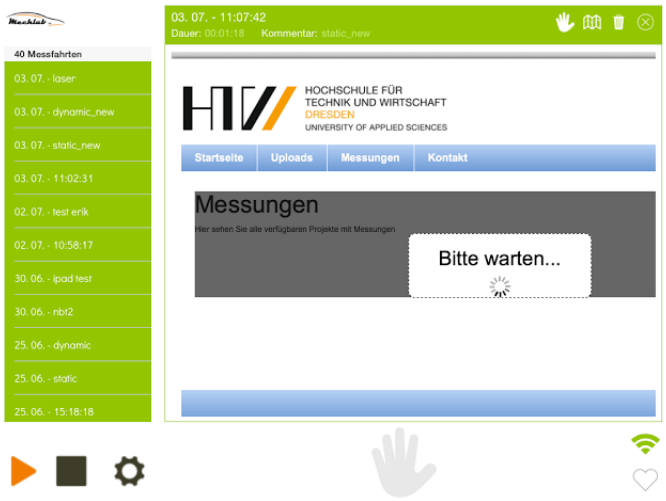


Fig. 3.14: Screenshot from the Application

After then, the logged data were uploaded on the server of the laboratory, and I was able to analyze them also. Below in Figure 3.15, is an example from the logged data from this application.

	A	B	C	D	E	F	G	H	I
	timestamp	**Latitude**	**Longitude**	**Altitude**	**VerticalAccuracy**	**HorizontalAccuracy**	**Speed**	**Course**	**LocationTimestamp**
	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER
1	timestamp	Latitude	Longitude	Altitude	VerticalAccuracy	HorizontalAccuracy	Speed	Course	LocationTimestamp
2									
3	1396428423	51.033959	13.738854	117.011562	4	5	0	149.414062	104702
4	1396428424	51.033959	13.738854	116.96487	4	5	0	149.414062	104703
5	1396428425	51.033959	13.738854	116.96487	4	5	0	149.414062	104704
6	1396428426	51.033959	13.738854	116.807034	3	5	0	149.414062	104705
7	1396428427	51.033959	13.738854	116.785366	3	10	0	149.414062	104706
8	1396428428	51.033976	13.738859	116.838772	3	5	0.31	149.414062	104707
9	1396428429	51.03398	13.738863	117.480679	4	5	0.4	149.414062	104708
10	1396428430	51.033983	13.738861	117.052273	4	5	0.37	149.414062	104709
11	1396428431	51.033984	13.738859	117.777493	4	5	0.36	149.414062	104710
12	1396428432	51.033982	13.738856	118.198208	4	5	0.31	149.414062	104711

Fig. 3.15: GPS Bike Logged Data

3.4.3 Merging the readings

The final step in the GPS measurements, was to merge both readings together. Hence, we can be able to get exact locations of both the bike and the car at the same timestamp. So I started taking readings from both receivers simultaneously. A path was defined which was some streets around the laboratory, and then the bike and the car started moving in this path whilst the GPS data from both were logged.

The following figures are showing the exact route that was taken from the bike and the car to get the readings.

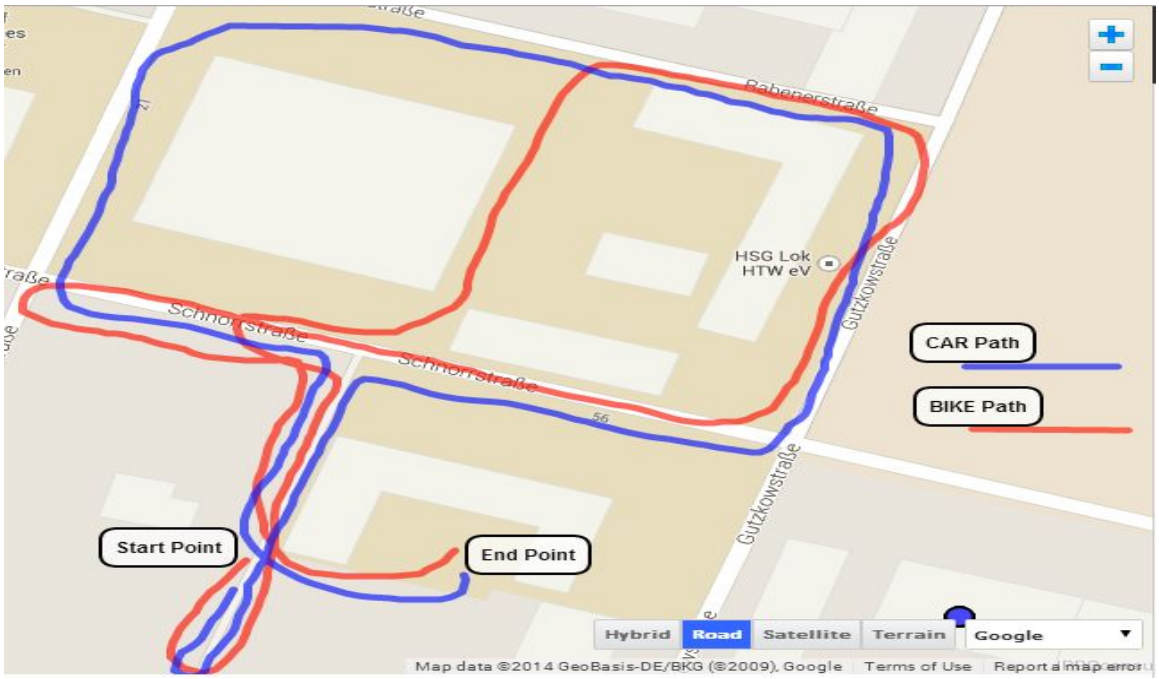


Fig. 3.16: The route on Google Maps

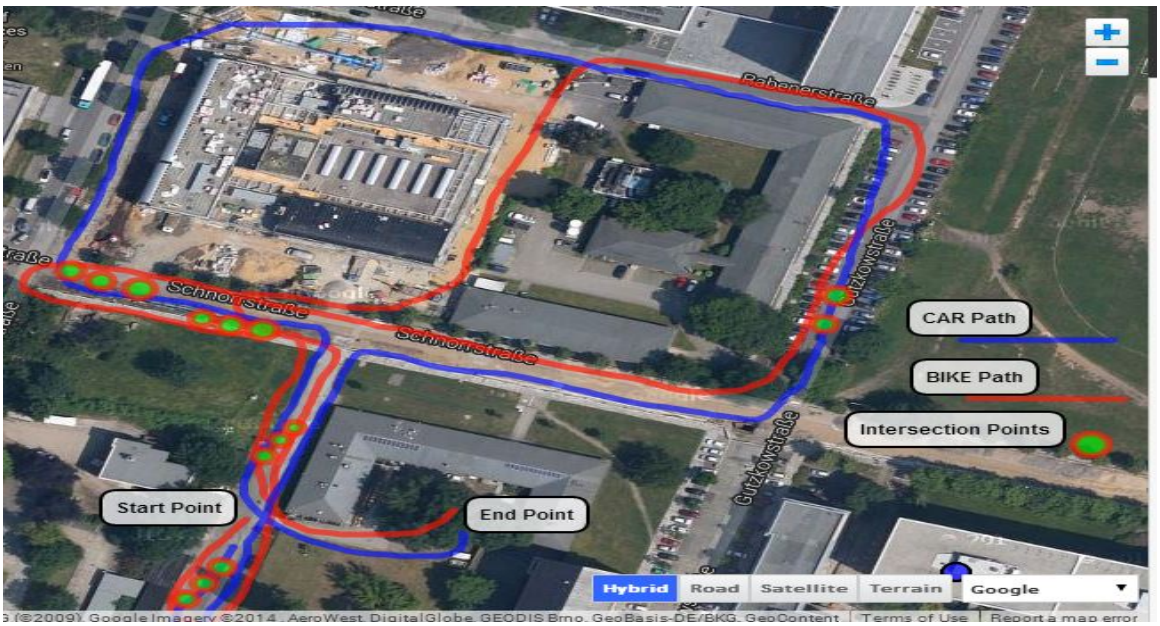


Fig. 3.17: The route on Google Earth

And in Figure 3.18 is a screenshot from the video taken in the car during taking the measurements which is showing the intersection of the bike and the car at some points during their route.

Fig. 3.18: Car and Bike intersection point

After taking these readings, some ammendemts were made to ensure that both readings are logged at the same time which means that they start and end at the same timestamp. Then using MATLAB, I tried to plot these data together.

The GPS data from both logged files, were converted to distance using equation 2.4, then they were plotted and compared together to find the intersection points between the bike and the car, and afterwards, detecting the critical situations where the bike is near to the car and there is a possibility to crash together. Also the same concept that was applied in the Radar readings was applied to the GPS readings, which is generating of 3 warnings according to the seriousness of the situation. Horizontal and vertical accuracy were considered in the detection, because as stated in Appendix G that GPS does not give us accurate data. So according to this, the intersection points were plotted with an accuracy region.

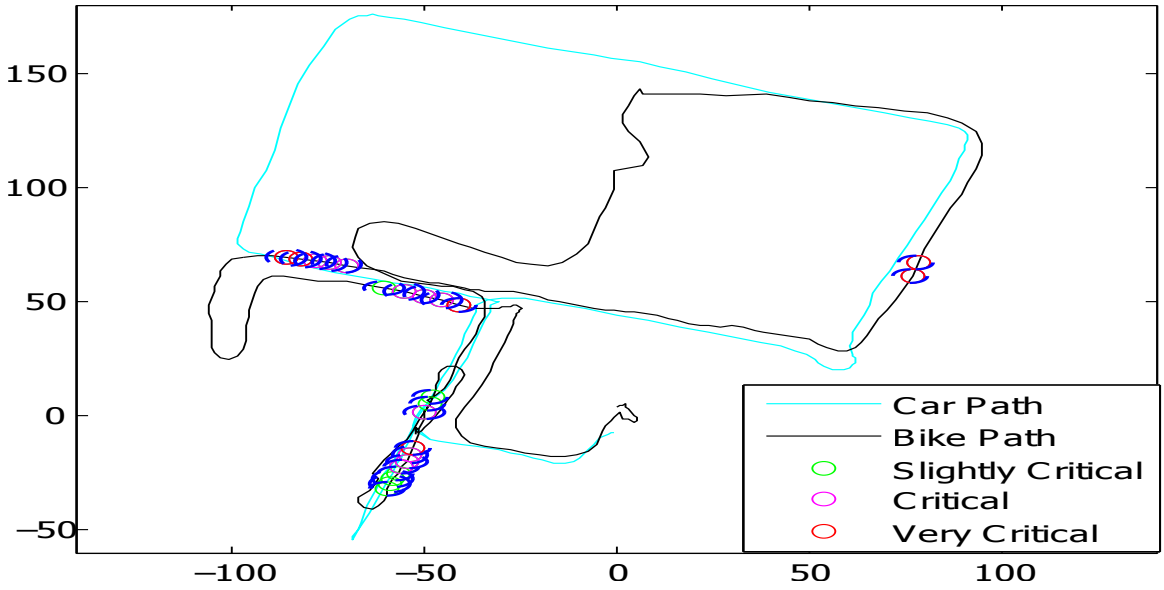


Fig. 3.19: MATLAB plot of the GPS readings

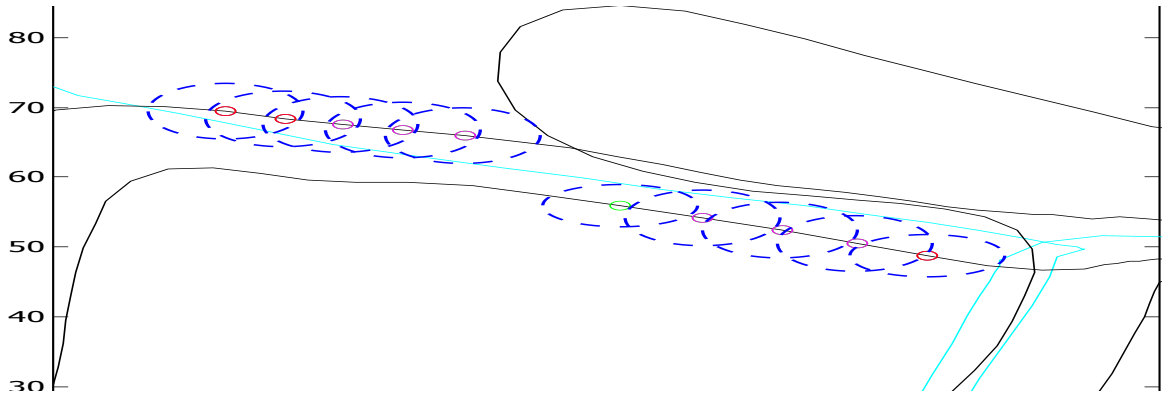


Fig. 3.20: GPS Accuracy

The warnings were defined on the plot with three different colors; the green means that it is low warning, magenta means medium warning while red detects that there is a critical situation, thus gives high warning.

3.5 iBeacon Readings

3.5.1 Identifying on the application

Some measurements were taken from the iBeacon to check how does it actually work. iBeacon returns a log file with the UTC, longitude and latitude of the iOS device and the distance between the device and the iBeacon. In fig 3.21 are some screenshots of how the iBeacon is detected on the app installed on the iOS device.

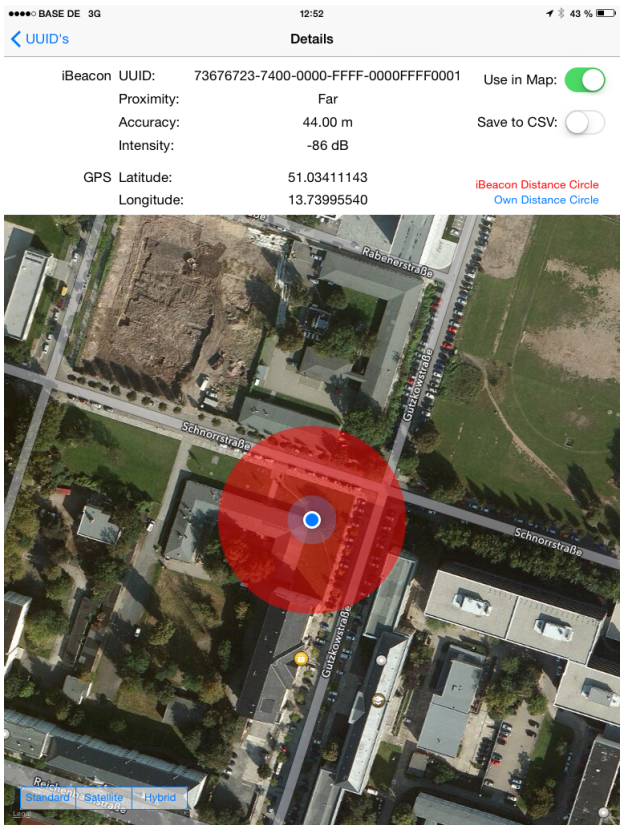


Fig. 3.21: iBeacon Detection Application

3.6 Simulation

Before going into the procedure of applying all of the above steps into real life (car), it was very important to ensure that of these measurements that were taken separately can be merged together to give out the desired output. So I wanted to make sure that the GPS with the Radar together, can detect the critical object and so giving alarm signals when required.

For this step, I had to use Simulink to work on performing a simulation with all the approaches merged together and then generating the warning generations at critical situations, and if this worked as required, so the next step will be applying this to the car.

A Simulink model was created with a predefined route and position of the car and the 3 bikes.

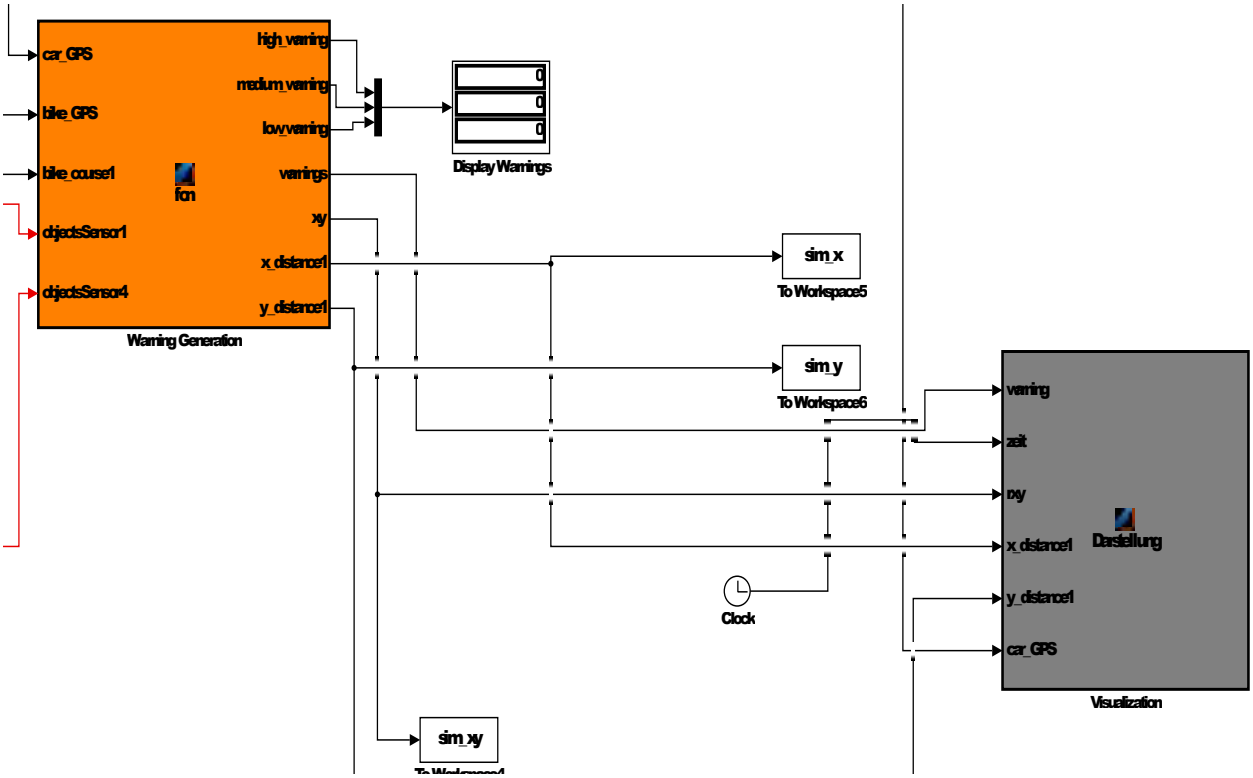


Fig. 3.22: Part of the Simulink Model used

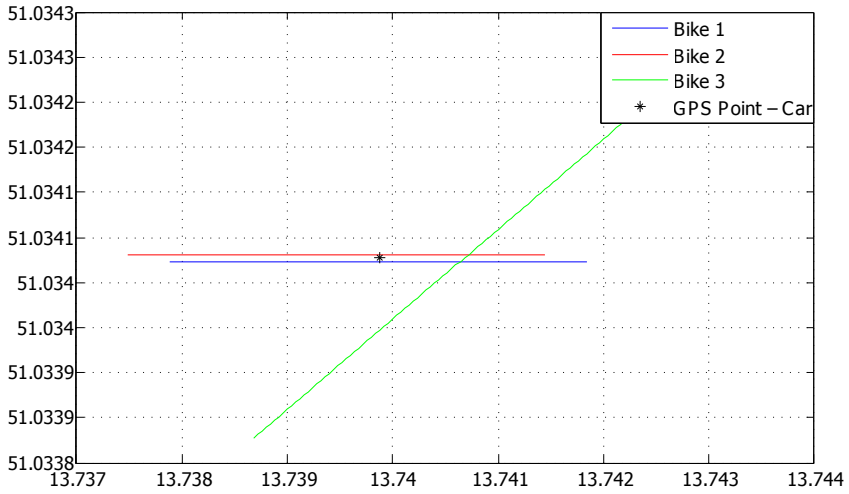
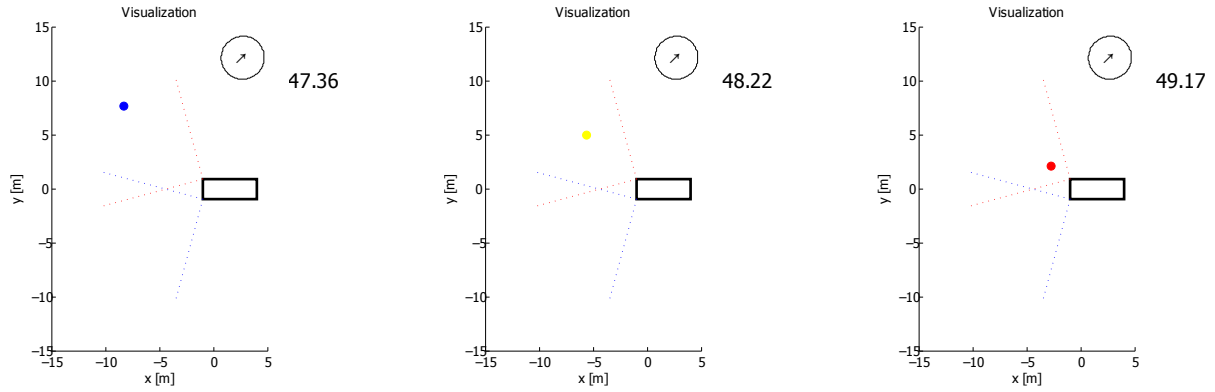


Fig. 3.23: Simulink Model - Car and Bikes positions

Then, the Radar data were generated and analyzed as well as the GPS data for the 4 vehicles. Critical situations were also defined, and then starting to detect if one of the bikes gets in a critical situation with the car or not, and so generating some warning generations according to different situations.



(a) Object detected - No Warning

(b) Object detected - Medium Warning

(c) Object detected - High Warning

Fig. 3.24: Simulink Model Results

Figure 3.24 can show the results obtained from the object detection system generated in the model. So in the 3 above figures, it is shown that a bike which was moving towards the car was detected by sensor 1. At the beginning, no warning was generated because the bike was a little bit faraway from the car and it has no danger to crash, however, after some seconds, the bike got nearer, so a medium warning was generated, and then the bike became in a critical situation that it may crash with the car, therefore the high alarm was given out. These alarms are shown in the model as yellow and red colors.

3.7 Real Measurements

After making sure that each device is working good separately and giving out the required output and results, as well as, they work together when merged in simulation; so now it is time to take real measurements and then try the model to see if it will give the same output or not.

At the beginning, I started taking measurements of the radar and GPS together, so GPS receiver was started in the car as well as the Radar, and the application was started to log also for the bicycle's driver.

To start with, I decided to make static measurements first then the dynamic ones; because in the static, the car will be standing still with one non-changeable course which will be easier to detect any errors that can occur. However, I faced some problems while taking these real measurements. The first problem that there was a great difference in the time stamp between the GPS and the Radar. Which means that at a certain time, the bike should be at a certain position near the car, but this position was detected by the Radar and identified by the GPS coordinates but at different times.

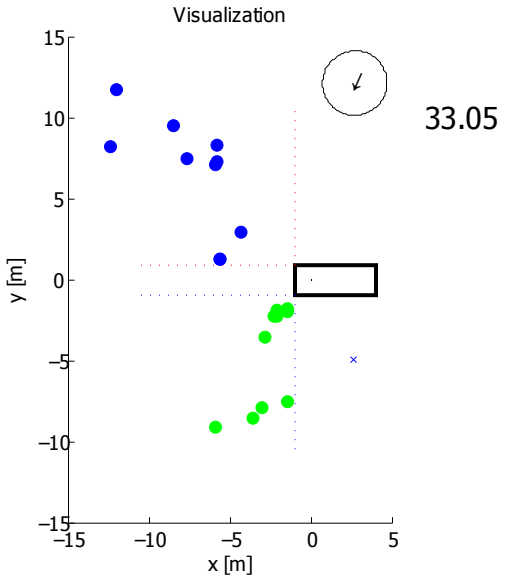


Fig. 3.25: Error - Difference in timestamps

As shown in Fig. 3.25, that at second 33.05, a bike is detected by the radar sensor coming towards the car, however, at the same second the GPS coordinates of the bike show that the it is beside the car moving in another direction, but some seconds later it was shown moving towards the car. So this shows that there was some differences in seconds between the GPS readings and the Radar readings.

I tried to find out the problem in the shown simulation, and by checking all the readings that was taken from both logged data, it was significantly shown that there is a problem in the logged data from the bike. The UTC shown had some problems, that there are some repeated seconds as well as some missed seconds, so by this accumulation, it will lead to difference in timestamps on the simulation which leads to errors in detection. The graph below is showing how there are some problems in the UTC logged from the bike

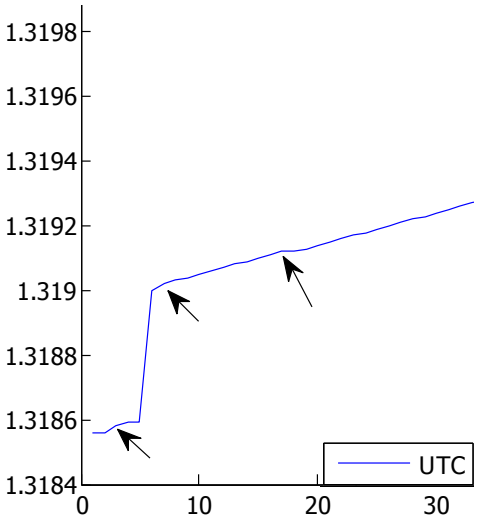


Fig. 3.26: UTC of GPS reading with Errors (Arrows)

As a result, I decided to stop using the application installed on the iOS device, because maybe there are some problems in the logging process. So the solution was to use a GPS receiver as the one installed in the car, however, this receiver gives out outputs in NMEA Format. It was the same in the car, but in the car PC there was a program which can automatically change the NMEA Format to longitude and latitude but there must be a CAN Case connected. So it was easier to log the data and change them manually to get the required outputs.
More information about the NMEA format and the analysis of its signals can be found in Appendix H.

So now the step was taking the measurements using the new procedures as well as the old ones to compare and see the difference in errors or logged data. We defined a route (the same route taken before), and the GPS receiver for both car and bike was used, as well as the application (to plot the differences), the radar and the iBeacon.

3.7.1 Static Measurements

I started analysing the results. First, I began with the static one. because it was easier as the car is standing standstill and only the bike was moving around it. The car was standing at 6° to the north and the bike started moving around it, coming from the left and the right sides of the car. Measurements were taken using GPS receiver and iPad for the bike, Radar and GPS receiver for the car. The GPS receiver for the bike was just used for comparison between the measurements of the receiver and the iPad and seeing the differences; however, it cannot be used in an Advanced Driver Assistance Safety System.

Readings were taken, plotted and analyzed. Below in Fig. 3.27 is shown the path logged from the receiver and the iPad. The red path is from the iPad, and the blue is from the receiver.

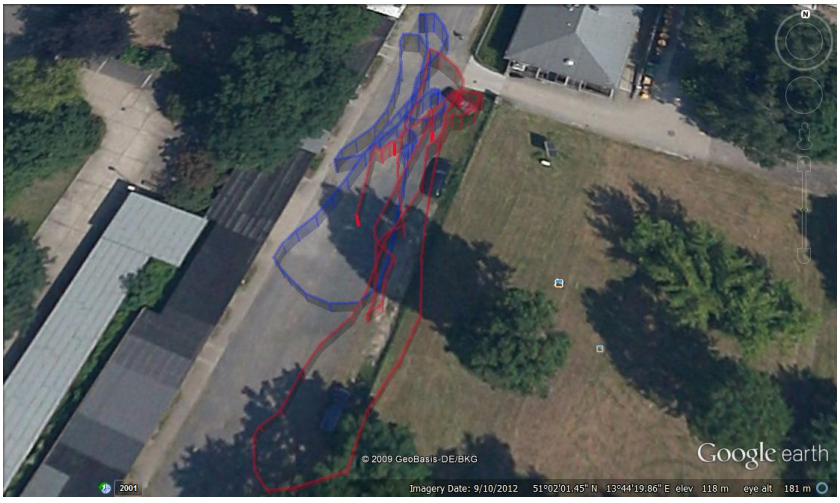


Fig. 3.27: Path on Google Earth for Static Measurements

As shown in the above figure, that the two paths are considered wrong and have lots of

errors. It is shown as the same route taken for the bike to move around the car, but not the same GPS coordinates that were already taken by the bike in the movement; which shows that the GPS measurements have a great error-accuracy which leads to wrong readings.

The next step was to make sure of the difference in timestamps between the receiver and the iPad to check which is better in giving the data.

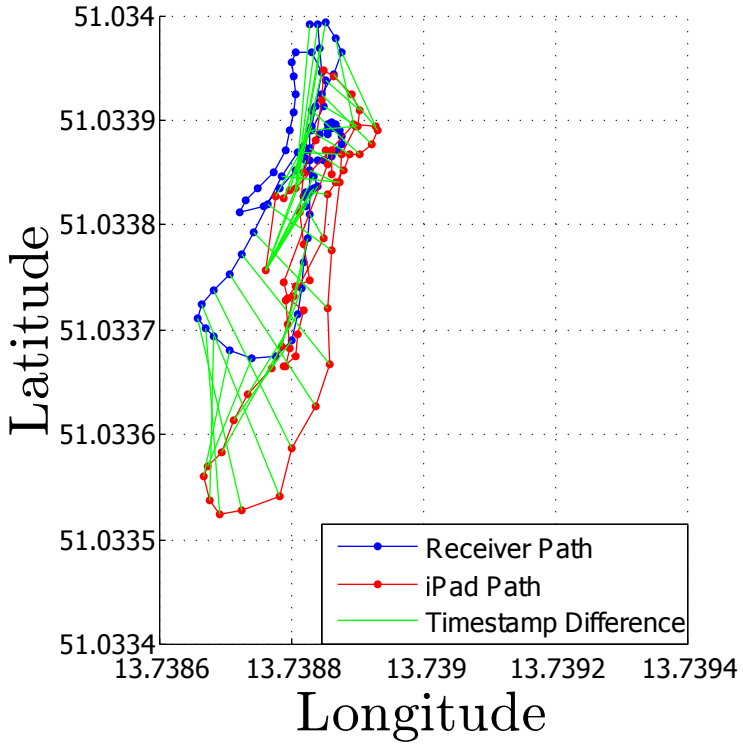


Fig. 3.28: Difference in timestamps between iPad and Receiver - Static

From Fig. 3.28, difference in timestamps between both tools was not significantly shown or realized. It seems to be good; straight lines were plotted between each point which shows that there was no significant difference in seconds between both tools.

And then, these measurements have to be loaded on the simulink model to check the simulation of how objects (bike) are detected from both radar and GPS, and check if this is the required output or not. So the logged data from the the tools were imported in Matlab and run by the simulink model, and the results were analyzed.

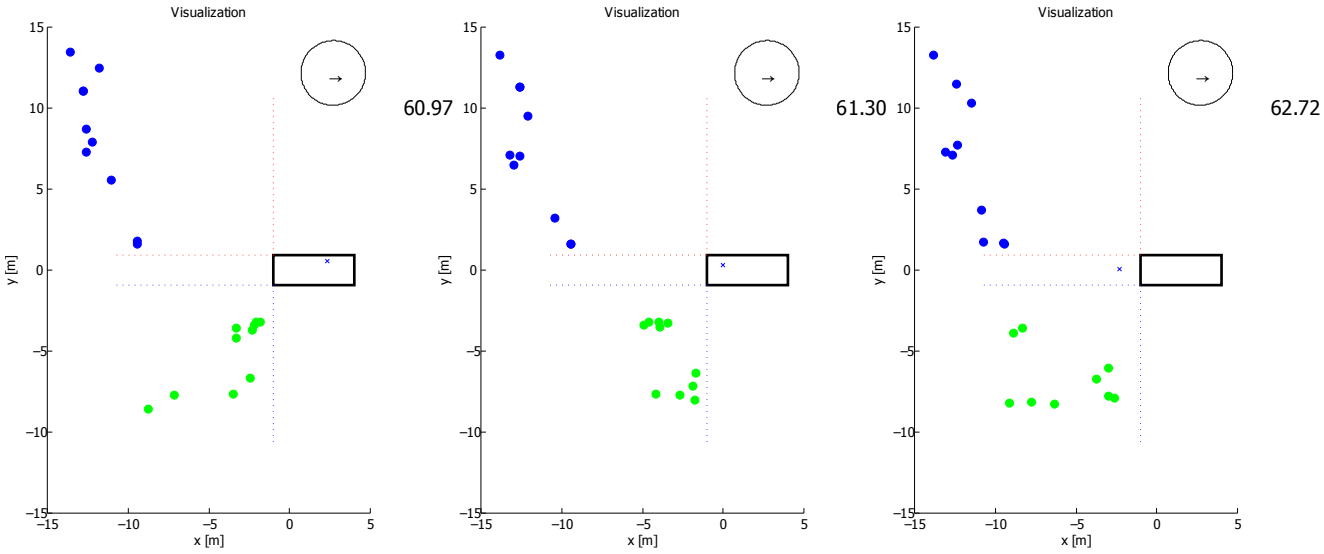


Fig. 3.29: Simulink Model Results - Static Measurement

As seen in the above group of figures, that the error accuracy of the GPS is significantly shown in the results. So it is clearly shown that starting from second 60, the bike started to move to the opposite side of the car from its left. And this is shown in the simulation and detected in the right position from the radar sensor, however, it is detected at a different position by the GPS, but moving in the same path; which means that there is an error accuracy for around 3 meters in the x, and 4 meters in the y. And for more elaboration, it is shown in the video captured for the static measurement, Fig. 3.30 that the bike was moving in the direction detected by the radar at this second, which is detected in a different position by the GPS.

Fig. 3.30: Screenshot from the video - Static

3.7.2 Dynamic Measurements

These measurements were more complicated because, the car is moving in streets with a lot of detected objects as well as the bike is moving in a different paths but meeting with the car at some points. So same procedures for the static measurements analysis were taken.

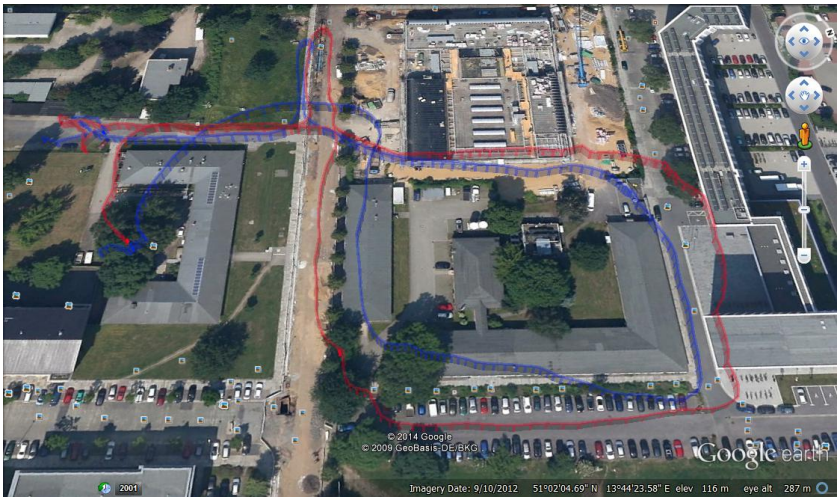


Fig. 3.31: Path on Google Earth for Dynamic Measurements

As shown also in Fig. 3.31 that the same problem occurs. The two paths of the receiver and the iPad are not giving right positions of the bike where it moved. They give the same path, but not the exact GPS coordinates points.

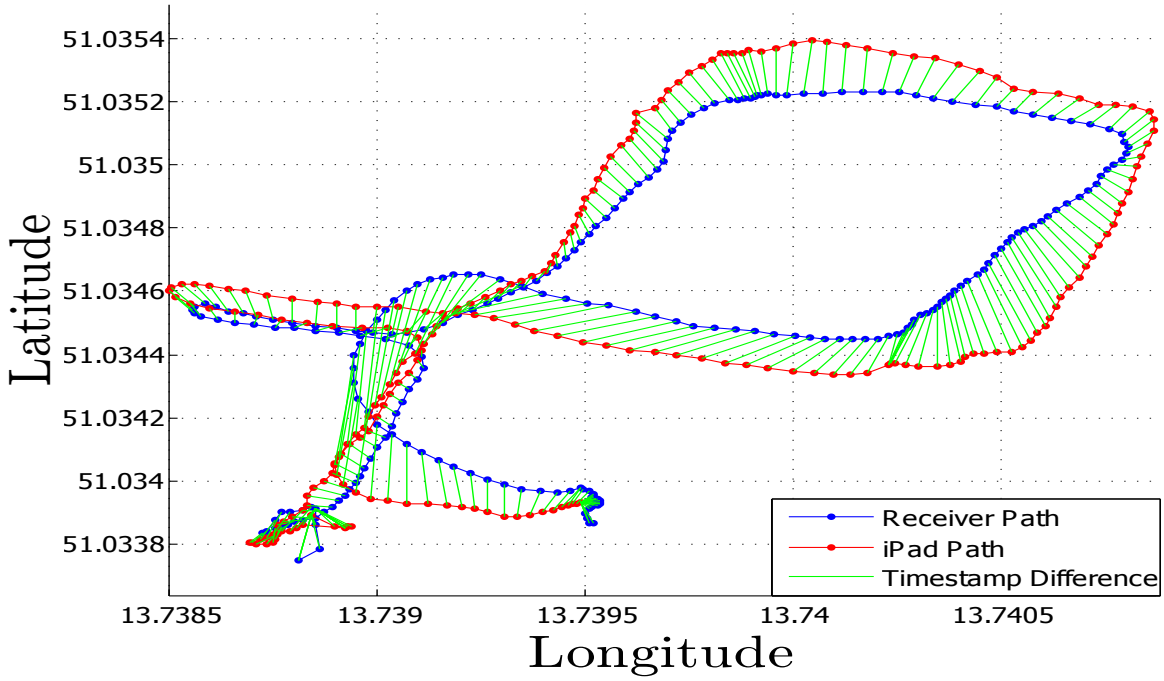


Fig. 3.32: Difference in timestamps between iPad and Receiver - Dynamic

And above is the plot for the timestamps difference between the two tools (receiver and iPad). Here we can see that also there are no significant differences between the timestamps for both tools. Furthermore, by analyzing the UTC data from the GPS, it was found out that the great problem still exists in logging the timestamp. This is clearly shown in Fig. 3.33.

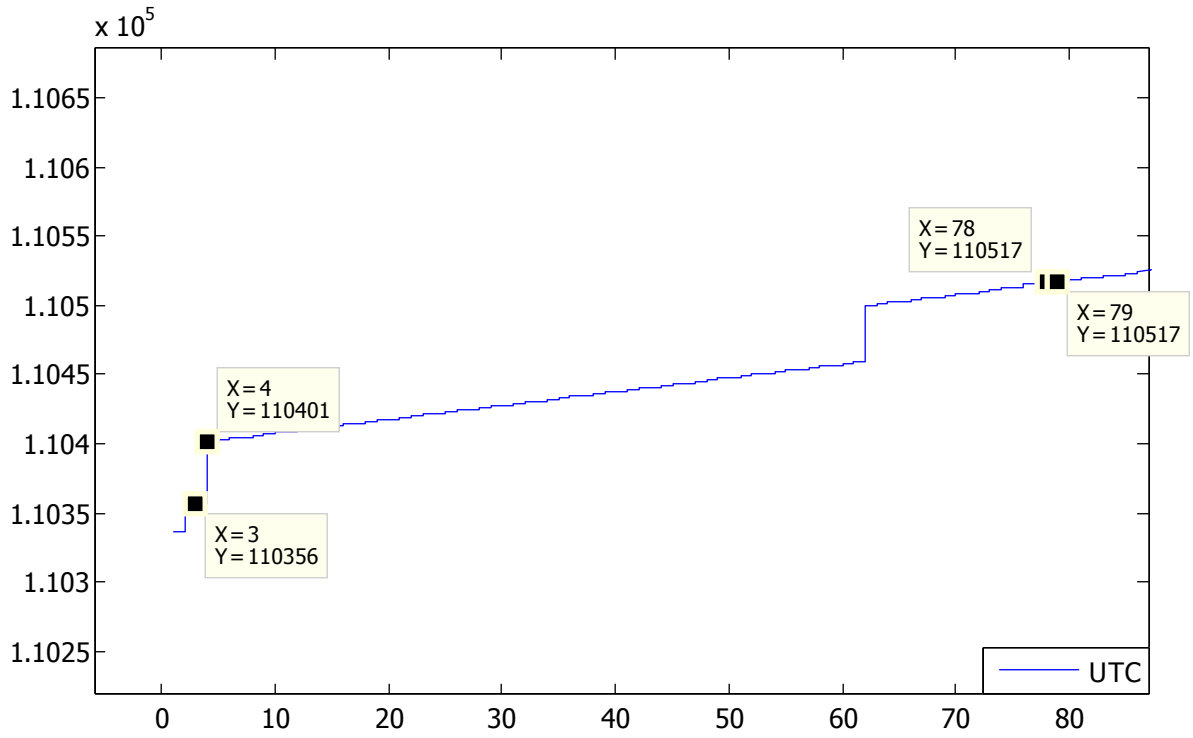


Fig. 3.33: Error in the UTC

So it is significantly shown that we have some repeated and some missed seconds. For example, at point x=3, the UTX was 11:03:56, but in the following point x=4, the UTC is 11:04:41 which means that we have 5 missing seconds. This happens also later; so as we see at point x=78, the UTC was 11:05:17, and at x=79, the UTC was also 11:05:17, which shows that we have 1 repeated second in the data. This also happened several times along the readings. So the problem now with the GPS is not only the accuracy, but also the readings of the UTC.

3.8 Finalizing Systems

After finding out these errors during the measurements, it was decided to create two simulated systems. The first one will be ideal system which has no errors in any measurements and this was going to be used as a control system; which means that it will be used as a comparison between a real system with its errors and the system that should be. The second system was a real simulated system; which means that this system will contain simulated data but with errors defined. The errors will be the normal errors that were found out

during the measurements; and thus I can be able to find solutions to overcome these errors and reach the best required output.

3.8.1 Simulation of the Ideal System

The ideal system required to be implemented is to achieve the best output and create the warnings at the real-time measured data. So it should detect the actual position of the bikes and the car as well as the actual time at this moment to give out the right warning.

So 3 bikes were defined with their initial positions, paths and courses, as well as a car with its standing course and position. GPS and radar data were created and then simulated. In ideal case, the radar and the GPS should be synchorized together at the same time and be detected together from the car; because supposedly, this is the same object detected.

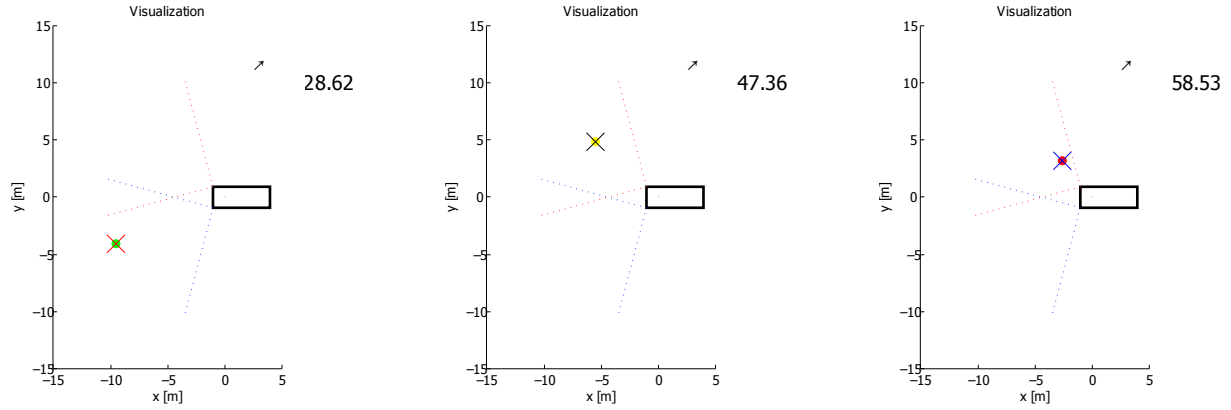


Fig. 3.34: Ideal System - Simulation

So as clearly shown from the above figures, that this should be the ideal system that works. The dots represent the objects detected by the radar sensor, while the crosses represent the position of the bikes detected from the GPS. Normally, the 2 data should be synchorized together at the same time and position because they are the same object. So here in the simulation, they were detected at the same time and giving the same distance from the car, thus giving the right warnings at the critical time.

3.8.2 Simulation of the Real System

However, in real life, this is not the correct measurements. We have a lot of errors that can be found during taking real data. So it was very important to create another system which contains these errors and then we can be able to analyze and try to overcome them.

- **Radar**
  The first error was in the radar data. The problem in the radar measurements that they don't give out accurate angle or distance, however they give out the data with a specified resolution and an error. So it was necessary to convert the radar measurements to real ones. As predefined in the radar data sheet attached in Appendix D, it was

shown that the distance measured by the sensor has a resolution of 0.01 m/bit and the angle has a resolution of 1 deg/bit. Also, as shown before in figures 2.32 and 2.38 that the distance of the detected objects has an accuracy error of ± 0.05 meters and an angle error of $\pm 5°$. Thus, the calculations were made on the measured data to be converted into real measurements.

- **GPS**
  The GPS has two main problems that need to be taken into consideration while implementing this system.
  - **Update Rate**
    The first problem that we have in the GPS was the update rate. As stated in Appendix G, that the GPS gives out reading (new position) every 1 second. So that means, that the radar object can be detected 1 second earlier by the car.

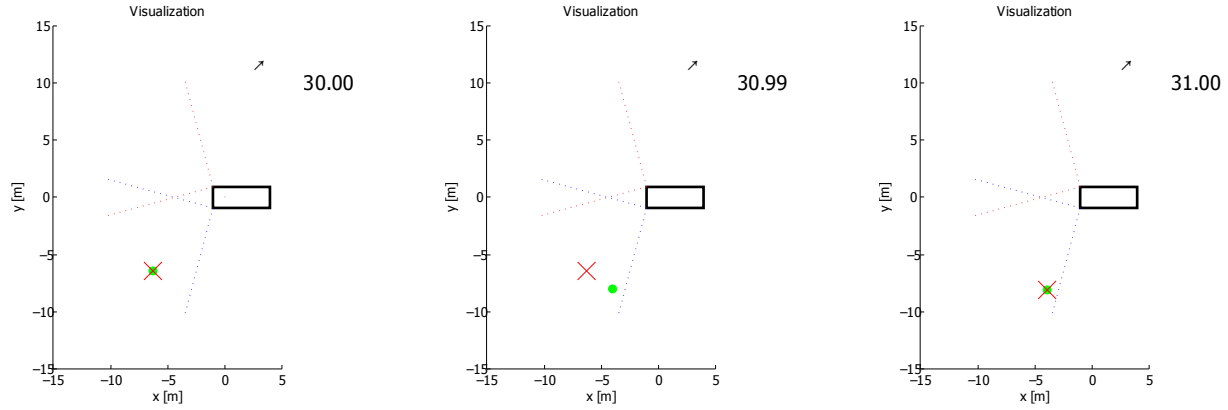


Fig. 3.35: Real System - GPS Update Rate Error

The above figure is showing the simulation of this problem of the GPS, so the cross here is representing the GPS object and the dot is representing the object detected by the radar sensor. If this was left as it is, so that means that the warnings that are given out when a fusion happens between the GPS measurements and the radar measurements can be delayed; because the fusion in the critical area can be detected 1 second later. Therefore, the solution with this problem is to always generate the warning in case of a fusion happens in the previous cycle in the simulation. This means that if already a fusion happens between the radar and the GPS just in the cycle before entering the critical region, so for sure the object detected in the critical region in the next cycle by the radar will be the same object detected by the GPS 1 second later. Thus a warning is given if this condition was true. In the figure below, it is significantly shown that a warning is given while only the radar object detected in the critical region, because already in the previous cycle, a fusion happened between the radar and the GPS.

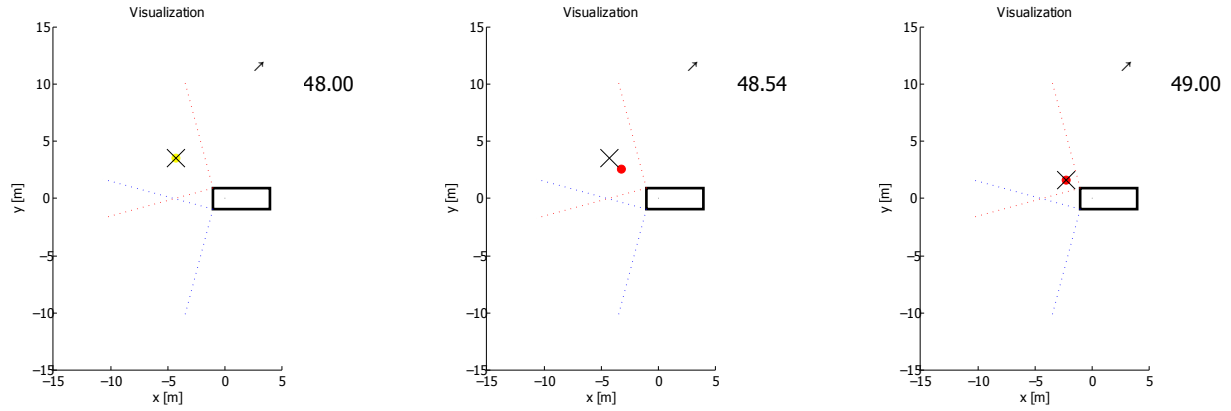


Fig. 3.36: Real System - GPS Update Rate Solution

As shown in Fig. 3.36, that in second 48, the radar and the GPS were fused together. So this let us make sure that during the next cycle, the object detected by the radar will be the same object given by the GPS. It is then clearly shown that the warning did not wait for another fusion to be generated, however, it is generated automatically when the radar object is detected in the critical region.

– **Position Error**
The second main problem with the GPS is the error given out in the position. It is known and shown in the previous measurements that the GPS gives out an error in the position between 3-5 meters. Therefore, it was a must to consider this error in the simulation of the real system and then try to overcome it. So, a random number between -3 and 3 was created in each cycle and added to the GPS distance, so we make sure that the error of the GPS in taken into consideration in the simulation. So as a whole, the GPS coordinates of the bikes are moving in their same path, but not taking the actual route due to some accuracy errors in the horizontal and vertical distance. This was clearly defined before and proven in Fig. 3.31 and Fig. 3.27. This problem was solved using the previous step which is the fusion, but added to it a fusion between the GPS and radar with a circular area of 2.5 meters. So if the GPS and the radar are detected in a circular region between -2.5 & 2.5 meters, so that means it is probably the same object detected by the radar, is the bike detected by the GPS, thus warnings are generated.

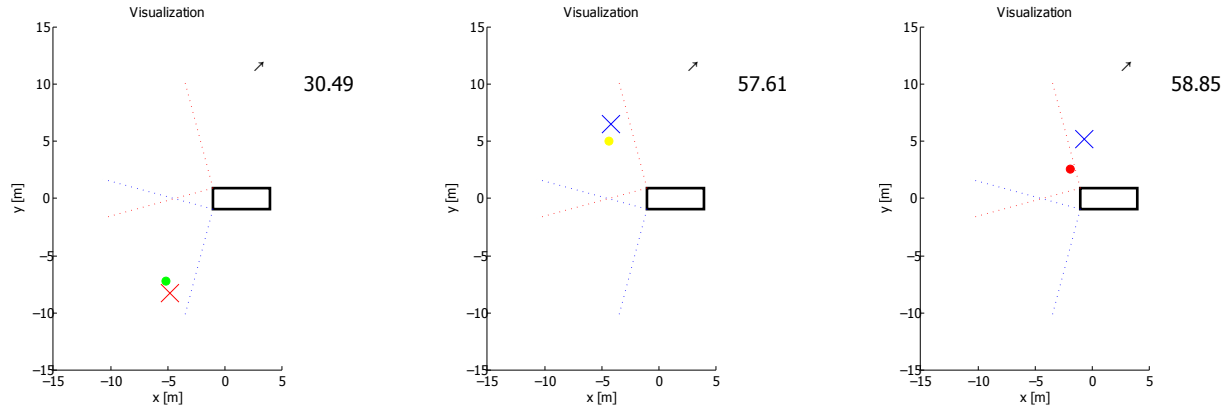


Fig. 3.37: Real System - GPS Position Error

Another approach used to make sure of the detected object and to overcome the error was using the velocity. The velocity of the bike with its course can be used to assume the next position from the previous one for the bike. So in a cycle, we have the longitude and latitude of a bike; so by using the velocity and the course, we can be able to calculate the next position that the bike can be in, thus providing an assumption if the bike is same object detected by the radar or not as well as if it is in the critical position or not.

Afterwards, I decided to try giving out the best output from the simulated real system. So I created different systems with different range of errors varying between 3-5 meters as an accuracy, and trying to figure out the best way to overcome the varying errors of systems from low and high accuracy error.

CHAPTER 4

Results

Many steps and procedures have been done throughout the work in this project including simulation or real measurements. However, finally, it was decided to finish the project as a simulated system, then to be edited in the future and applied to real life. Thus, the most important results to be shown and discussed were the results given out from creating ideal and real systems from simulated data. However, all the results of my work in this project were shown before in the previous chapters.

4.1 Ideal System

As seen before in the simulation the path of the bikes in the ideal case, below is a plot of the whole path taken by the 3 bikes either by GPS or radar sensor.

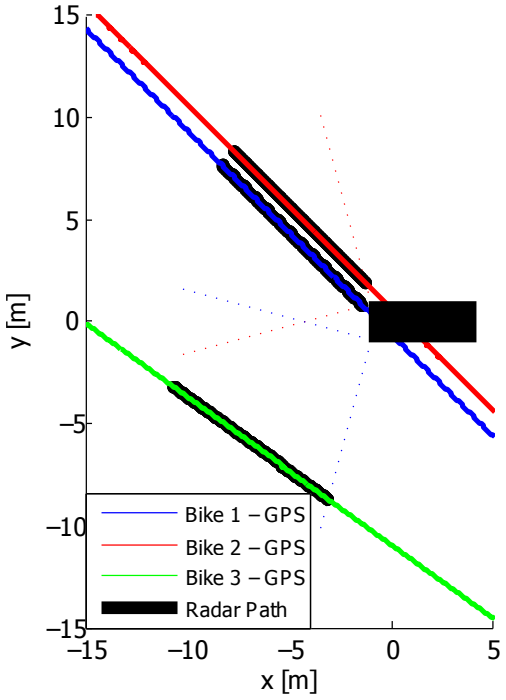


Fig. 4.1: Routes Plot - Ideal System

It was very important for me to create the best output of an ideal system and how it should be working. Therefore, I decided to make a 3D simulation for a car and moving bikes to be able to see how the output should be in real life with such conditions. So from the following figures, we can notice how the system should be actually working in real life and how the warnings are generated according to distance, speed and course of the bike. The coloured circle represents the warning which changes to different colours according to the current case.

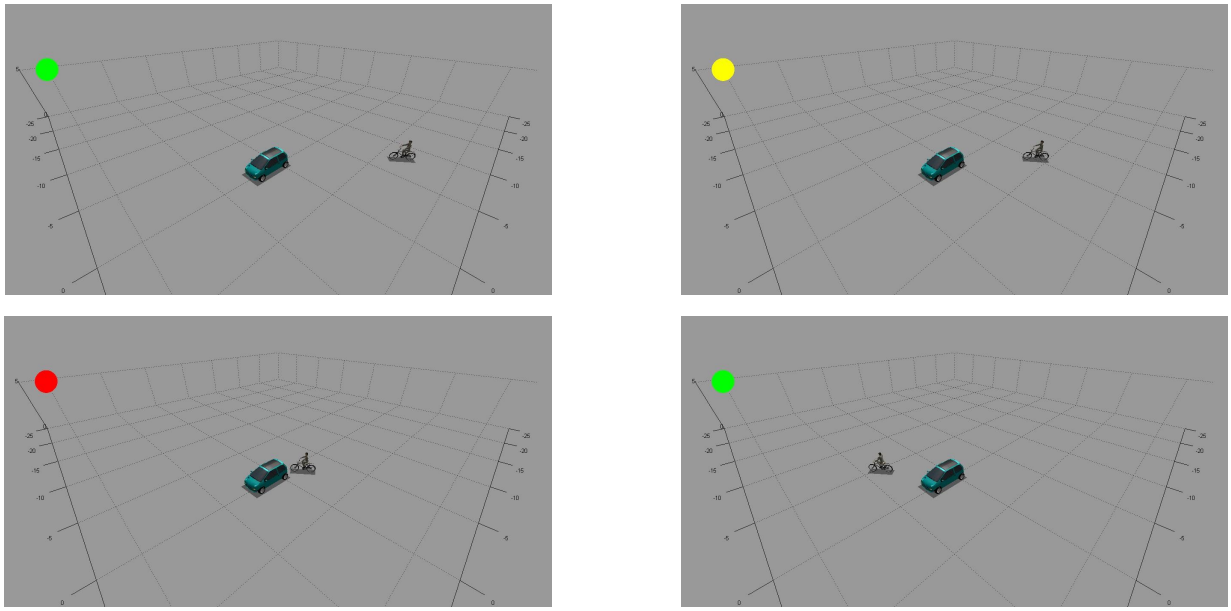


Fig. 4.2: Ideal System - 3D Simulation

4.2 Real System

After creating a system which is mostly related to what happens in real life; including errors for both radar and GPS, so I decided to find the best region of detection and fusion for different types of systems. Briefly, the GPS gives an accuracy error within a circle region of 3-5 meters, so this means that we either have a high possibility of error (which is the 5 meters), and a low possibility of error which is the 3 meters. So it was very important to find out if the detection region of fusion of the 2.5 meters will be possible to work with the different types of errors, or I will have to change. Thus, I decided to make 3 different systems; a system which gives an error of a circular region of 3 meters, 1 with 4 meters and the last one with 5 meters. Afterwards, the detection region of 2.5 meters was tried on these different systems.

1. **Circular Region of 3 meters**
   With creating a circular region of 3 meters and trying the fusion area of 2.5 meter; it was working. So i tried to decrease the fusion area to make it as low as possible. The lowest was 1 meters, however in some cases, the 1 meter area can generate a late warning due to low probability of fusion. So I tried then to decrease this a little bit, and for different cases, the 1.5 meters circular region was working perfectly.

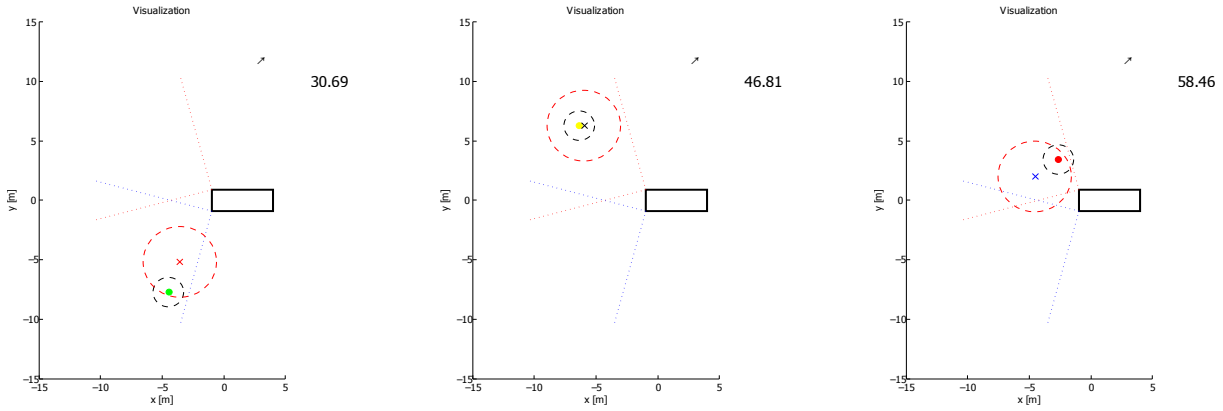


Fig. 4.3: Real System - 3 meters error

2. **Circular Region of 4 meters**
   In the circular error region of 4 meters, the object detected by the GPS point was not always fused with the radar object within an area of 1.25 meters. The probability of fusion was low; as it was tried several times with different errors. Therefore, it was required to change the fusion area to a bigger one. Different regions were tried, but a circular region of 2.25 meters was the minimum possible region which was working in different cases including bike or car course and bikes' speed.

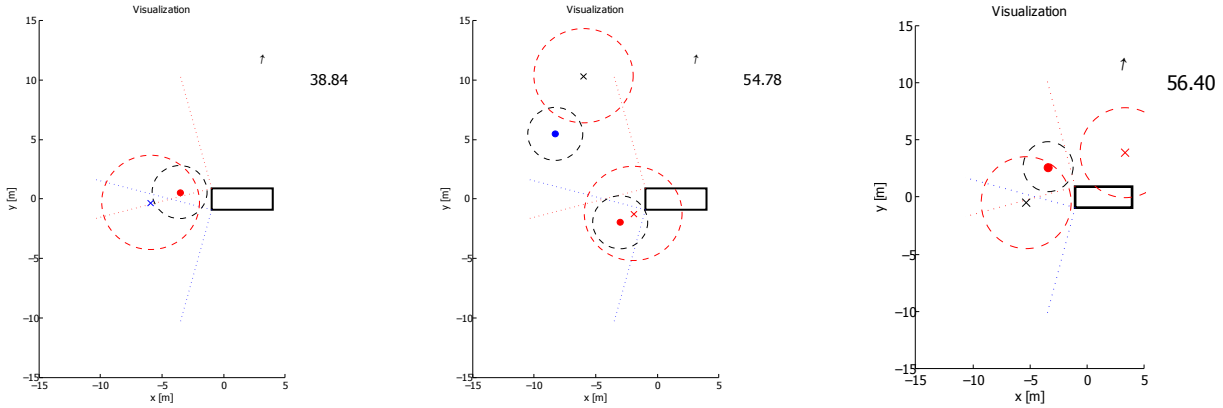


Fig. 4.4: Real System - 4 meters error

3. **Circular Region of 5 meters**
   The highest possible error that a GPS receiver can give out is a circular region of 5 meters; that's why it was important also to try this error. So we can have a an estimate of all possible accuracy errors that can occur in real life measurements. As have seen before from the last 2 possibilities, that whenever the accuracy error is increasing, the fusion region also needs to be increasing. So it is assumed that a circular region of 2.25 meters will not work perfectly for this error. But first, I tried the 2.25 meters error to see if the probability of fusion will be high or low as expected. After different trials, it was found out that a circular region of 3.75 is quite enough for the fusion area which can cover most of the cases tried including difference in speeds and/or courses.

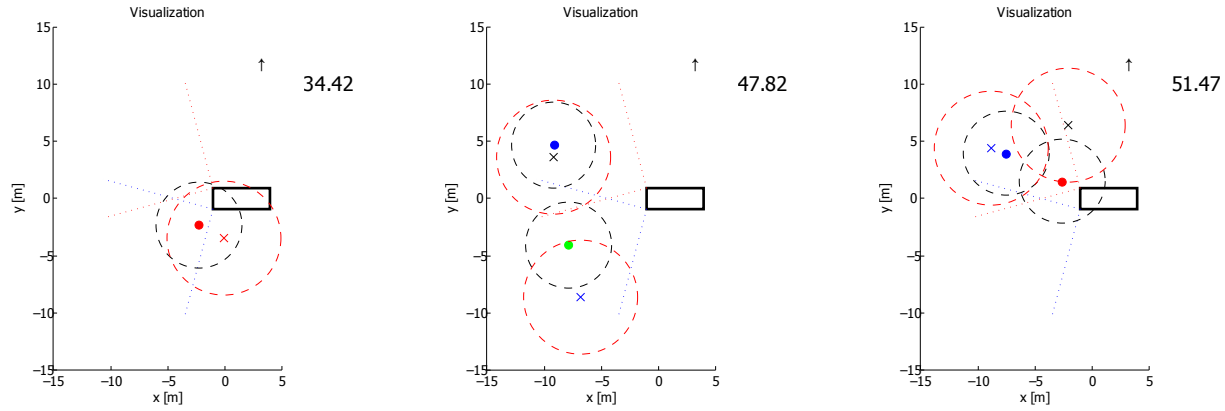
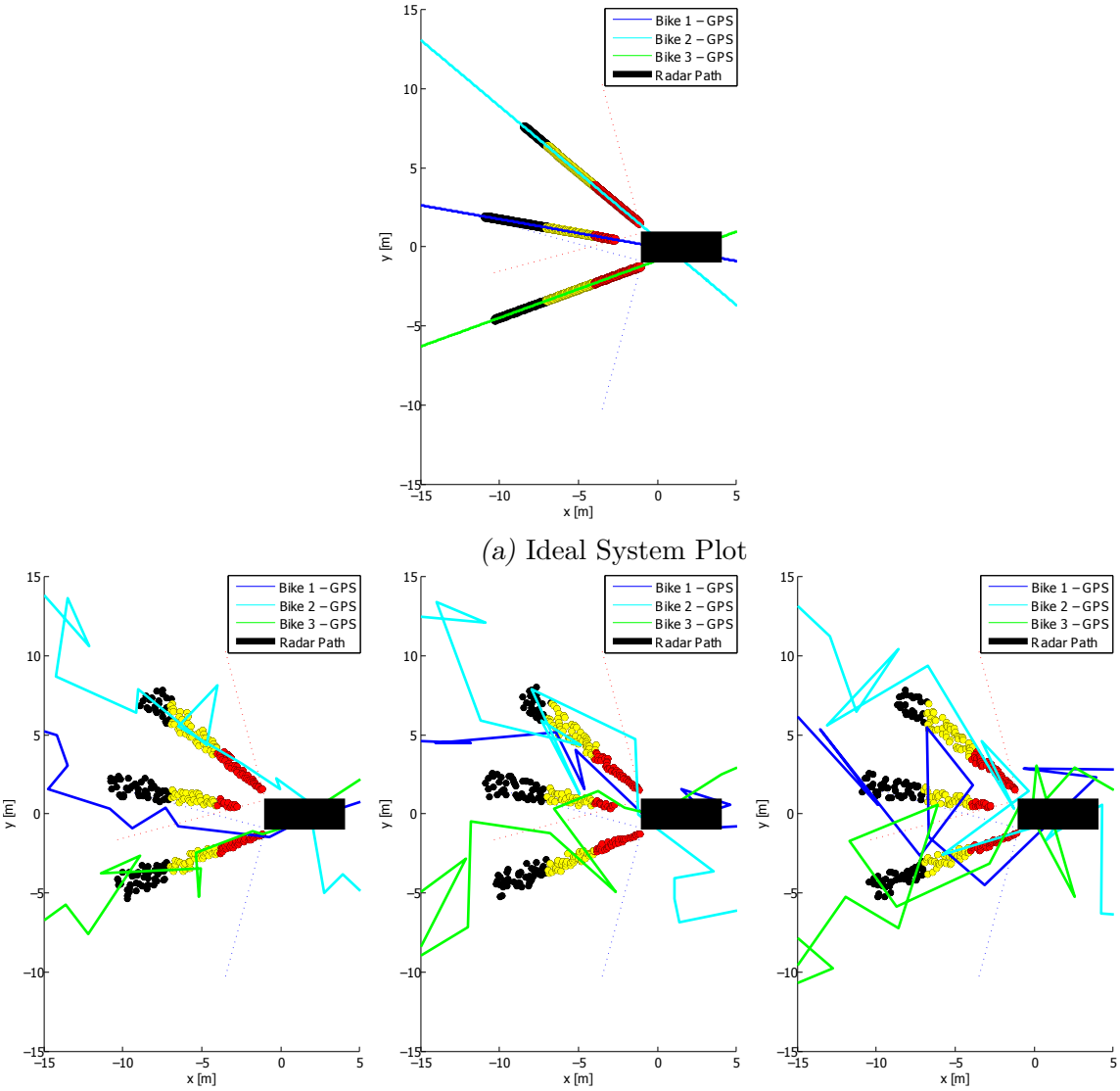


Fig. 4.5: Real System - 5 meters error



(a) Ideal System Plot

(b) 3 meters error *(c)* 4 meters error *(d)* 5 meters error

Fig. 4.6: Comparison between Ideal and Real Systems

From the above results, it is clearly shown that I was able to reach nearly the same output from the ideal system. In case of 3, 4 or 5 meters error, each one has its own technical amendments in the system to be able to give the required output which is the warning generations according to different cases no matter what was the error from the GPS. And here the 3 bicycles in the 4 cases were moving at different speeds but they were able to reach the desired output according to the distance between the bike and the car. So that means that fusion occurs between radar and GPS points at different regions according to the error. Low accuracy error requires small fusion region, while whenever the error increases, the fusion area also has to increase in order to cope with the new error.

CHAPTER 5

Conclusion

From these results, it is finally concluded that such Advanced Driver Assistance System cannot be implemented with these direct readings that we have from the GPS. From the static and dynamic measurements that were done in real life, we figured out a lot of errors either in the timestamp, UTC and accuracy error. Such a system is mainly working on detecting the positions of the bikes and the cars and thus preventing an accident to occur. But as we can see, even if there is a small error in the UTC or the position by 1 or 2 meters, it will not perfectly work in such a system. For avoiding accidents using this concept, we need an exact positions and times for the bike and car moving in the streets because if there is a difference in some seconds or in a position, it may cause a negative impact on the driver. For example, at a certain second there is an object detected by the radar which is coming towards the car. This object is not yet known if it is another car, pedestrian or a bike; though we need the GPS point to surely clarify this. However, at this point, the GPS is giving that the bike is still 5 meters away from the car, and this is because the accuracy error, thus it will lead to a negative action to be taken by the driver. Or in another way, the bike is coming towards the car, however it is not yet detected by the GPS because there are some missing seconds in the detection system, which is giving that the bike is not yet in the range of the car position. Also, at this point, the driver is not able to take an action, because the system is not yet giving an output which is showing that there is a critical situation about to occur. Therefore, it was required ,in case of the GPS receiver to be used, to skip at this period the real life measurements and try to develop a complete system which is working with simulated data and has the same errors that can occur in real life, then trying to implement the best technical data that can give out the most required output and results. Accordingly, I started working on developing two systems, one is the ideal system which is considered to be a control one, and the other is a real system which is edited and implemented to be compared finally with the output and results from the ideal system.

All in all, advanced driver assistance safety system is a great important issue that is highly needed these days after seeing a lot of accidents that occur between bikes and cars. Such a system has to be developed and implemented in all cars. This paper was mainly talking about developing of a new system that uses 3 approaches of reaching the required output and the warning generation in case of critical situations. The idea was to use radar sensors attached in the rear bumper of the car, GPS data taken from the car and the bike, and

iBeacon which is considered to be an additional approach to make sure of the critical situation. In the previous chapters, everything was described in details. Chapter 1 was giving an overview and introduction about such systems and how my system is going to work and what is its exact function. Followed by chapter 2, which was mainly describing everything I have used in doing my project including hardware or software. Afterwards, chapter 3 was describing my exact experimental work starting from the early beginning as initializing and understanding till the end and getting out the results. And finally chapter 4, was a summary of the main results that were revealed from my work.

After 6 months working on this system, and starting from the fine details to reach the required output through simulation or real life, it was proven that the GPS data with its errors is not considered to be a reliable approach for an effective system to avoid accidents. Tiny errors in time or position may cause another thing that is not expected to happen leading to negative effects either for the driver or another people in the streets, and by this, accidents cannot be prevented, however, they may occur, if GPS is used with its errors in the system. So different techniques were implemented in a simulated system to overcome such GPS errors and reach the required output.

CHAPTER 6

Future Recommendations

After all the results and the work, it was proven that GPS is considered to be the main problem of the system with its range of errors. So for future work, I recommend using Garmin GPS 19x. The highly accurate GPS 19x HVS position receiver/antenna provides up to 10 Hz update rates for position, velocity and time data. It offers high-sensitivity reception and enhanced position acquisition. With its enhanced position, heading and speed accuracy delivered up to 10 times more often than other receivers/antennas, it provides smoother drawing of your position on the plotter at higher speeds, and provides a more accurate heading at slow speeds. Compared to devices with slower update rates, GPS 19x offers dramatic improvement when used at low speeds. Position and course information from slower devices can jitter and swing considerably at speeds below 1 mph. The GPS 19x provides a consistent and smooth heading which is especially important when trying to hit a specific waypoint or mark [14].

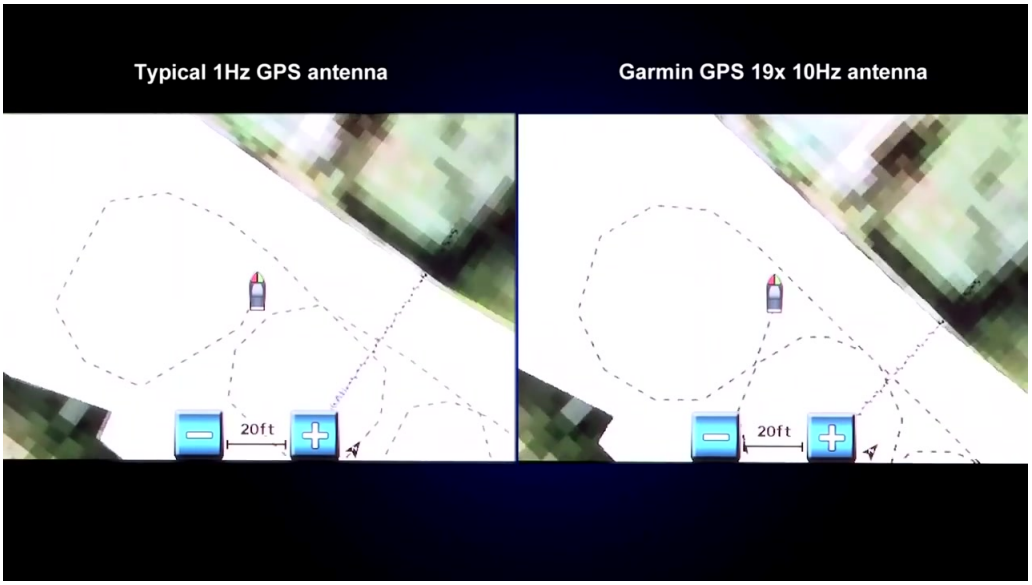


Fig. 6.1: Garmin 19x HVS .vs. GPS Receiver - Track Log Accuracy

However, this will only improve the GPS readings from the car which will replace the 1 Hz

receiver with this one. But the idea of the project was developing such a system by using smartphones, so it will not be possible to improve the readings from the GPS data taken from a smartphone of the cyclist. Though, I recommend that we can merge the GPS data from the smartphone with other built-in sensors like the acceleration for example to try to improve the readings.

Concerning the radar errors, I used in this project the tyco radar sensor which is considered to be old (approx. 10 years old), so it is recommended to try new radar sensors which can give out better measurements. Moreover, we only have two sensors attached in the rear bumper of the car with 90 degrees angle for each one. Therefore, we can use more sensors to give us a wide range of detection.

For example, Fujitsu developed recently the world's first Super-Wide-Angle 3D Laser Radar with a Horizontal and Vertical Range of 140 Degrees. This new technology gives out more than doubles area of visibility to detect hazards around vehicles, enabling sophisticated reverse driving support systems.The 3D laser radar that is able to measure distances to objects over a wide detection angle range. It does this by using a newly developed scanning angle expansion lens to beam the laser over a wide-angle range, and a high speed, multipoint laser scanning system to detect a wide range at high speed. As a result, objects can be detected in three dimensions over a wide range with fewer sensors and a sophisticated vehicle backing support system can be created. Moreover, in contrast to vehicle-mounted cameras, which simply display a vehicle's surroundings, the technology enables systems that detect when objects are abnormally close.

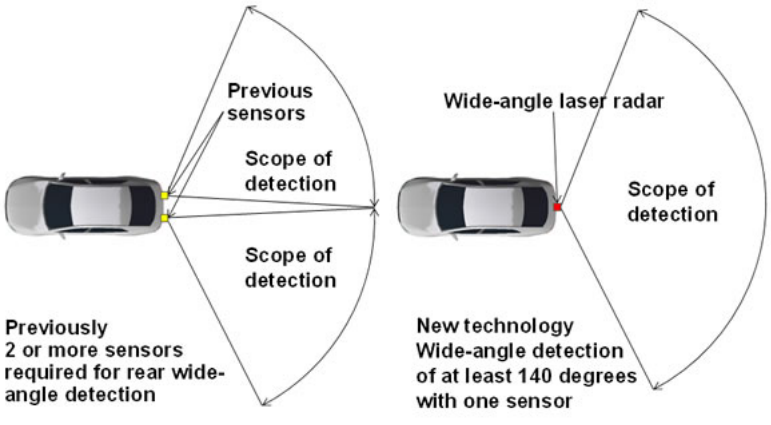


Fig. 6.2: Comparison of conventional sensor and wide-angle laser radar

Moreover, another recommendation would be using the velocity of the bicycle in assuming the new position. So for example, we have a longitude and latitude of a bicycle, as well as we have the course and the speed of it, therefore, we can calculate the new longitude and latitude of the bicycle by using the old data, the course and the speed. By using this new distance, we can assume the new position of the bicycle, thus knowing if it is moving towards the car or in the area of fusion and synchronization of the radar point or not.

Finally, I faced a problem during my project which is developing an application for the usage of iBeacon. The application that was developed was not working correctly, that's why my colleagues were trying to implement a new one that can give out readings from the iBeacon. So in the end, I did not use the iBeacon in the project. Thus, I recommend in the future work to merge the iBeacon data with the GPS and the radar in order to achieve better results and output.

Bibliography

[1] Automotive bus converter.

[2] Citroën c6.

[3] New safety kit from volvo, 2005.

[4] Radar sensors are designed for driver-assistance systems, 2006.

[5] Can physical layer and termination guide, 2014.

[6] Continental AG. Advanced driver assistance systems.

[7] Dr.-Ing. Jean Emmanuel Bakaba. More safety for cyclists at junctions in built-up areas. 2013.

[8] H-L Bloecher, A Sailer, G Rollmann, and J Dickmann. 79 ghz uwb automotive short range radar–spectrum allocation and technology trends. *Advances in Radio Science*, 7(4):61–65, 2009.

[9] Bosch. Driver assistance systems.

[10] Robert Bosch. Can specification version 2.0. *Rober Bousch GmbH, Postfach*, 300240, 1991.

[11] Inc. Corvallis Microtechnology. Introduction to the global positioning system for gis and traverse.

[12] Hans Dominik. Short range radar-status of uwb sensors and their applications. In *Radar Conference, 2007. EuRAD 2007. European*, pages 251–254. IEEE, 2007.

[13] Garmin. What is gps?

[14] Inc. Garmin International. Garmin 19x hvs technical specifications. Garmin Ltd., 2012.

[15] Vector Informatik GmbH. Canalyzer user guide version 3.1. Vector.

[16] Vector Informatik GmbH. virtual vectoracademy.

[17] Dean A. Goetz. San diego bicycle accident attorney - bike safety tips - car accident right hook.

[18] Ian Gresham, Alan Jenkins, Robert Egri, Channabasappa Eswarappa, Noyan Kinayman, Nitin Jain, Richard Anderson, Frank Kolak, Ratana Wohlert, Shawn P Bawell, et al. Ultra-wideband radar sensors for short-range vehicular applications. *Microwave Theory and Techniques, IEEE Transactions on*, 52(9):2105–2122, 2004.

[19] Richard Hammond. Blind spots.

[20] Richard Hammond. Beware the blind spot. *IN RICHARD HAMMOND'S FRIDAY COLUMN*, 2009.

[21] Mike Hanlon. Volvo launches blind spot information system (blis).

[22] AUDI HK. Assistance systems - adaptive cruise control.

[23] Holux. Gps receiver gr-213. 2005.

[24] Anh T. Huynh. V2013 infiniti jx35: Getting us one step closer to a driverless car.

[25] Apple Inc. Location and maps programming guide. 2014.

[26] Infiniti. Enhanced confidence with guiding technology.

[27] Admiral Infiniti. Infiniti blind spot intervention system explained for nj drivers.

[28] Alice Julia. How osi model and its layers works, 2012.

[29] Dallas Kasaboski. Trip to the algonquin radio observatory to track gps satellites!, 2012.

[30] H Kashif, G Bahig, and S Hammad. Can bus analyzer and emulator. In *Design and Test Workshop (IDT), 2009 4th International*, pages 1–4. IEEE, 2009.

[31] David Kohanbash. Coordinate systems (where in the world is your robot?).

[32] Brad Kruse. Short range radar for vehicular applications. 2006.

[33] Christopher Lampton. Blind spot monitoring technologies.

[34] Jeremy Laukkonen. Advanced driver assistance systems - increasing situational awareness to decrease danger.

[35] Tyco Electronics M/A-COM. Switched monopulse radar for automotive applications slr.

[36] Chris McGovern. About accidents.

[37] Michael. Blind spot intervention and your infiniti, 2013.

[38] NITIN. Affix the circle blind spot mirror in your vehicle for wellbeing journey, 2012.

[39] US Official. Government information about the global positioning system (gps) and related topics. *Tietoja GPS-järjestelmästä. Osoitteessa: http://www. gps. gov/systems/gps/space/# III*, 2012.

[40] Ralf Richter. Radar more safety on heavy vehicles. *ATZ worldwide*, 108(9):8–10, 2006.

[41] Simona. How to avoid blind spots.

[42] Steve C Talbot and Shangping Ren. Comparision of fieldbus systems can, ttcan, flexray and lin in passenger vehicles. In *Distributed Computing Systems Workshops, 2009. ICDCS Workshops' 09. 29th IEEE International Conference on*, pages 26–31. IEEE, 2009.

[43] The International Road Transport Union. Scientific study "etac" european truck accident causation. 2, 2007.

[44] John Wiesinger. Der can-bus - grundlagen von automobil bussystemen.

APPENDIX

APPENDIX A

Comparison between LIN, CAN and FlexRay Protocols

Tab. A.1: Comparison between different protocols

Bus	**LIN**	**CAN**	**FlexRay**
Cost/Node [$] (ABIreport: Y08)	1.50	3.00	6.00
Used in	Subnets	Soft real-time	Hard real-time
Application Domains	Body	Powertrain, Chassis, ...	Chassis, Powertrain
Message Transmission	Synchronous	Asynchronous	Synchronous & Asynchronous
Message Identification	Identifier	Identifier	Time Slot
Architecture	Single Master typ. 2....10 slaves	Multi-Master typ. 10....30 nodes	Multi-Master up to 64 nodes
Access Control	Polling	CSMA/CA	TDMA
Data Rate	20 *kbps*	1 *Mbps*	10 *Mbps*
Physical Layer	Single Wire	Dual-Wire	Dual-Wire (Optical Fiber)
Latency Jitter	Constant	Load Dependant	Constant
Babbling Idiot	N/A	Not Provided	Provided
Extensibility	High	High	Low

APPENDIX B

Controller Area Network

B.1 Standardization

Similar to other networks, CAN can be assigned to the OSI layer model. A special standardization of CAN technology exists since 1994, and is described by some ISO standards:

- ISO 11898-1 Description of the CAN Protocol.

  In connection to the reference model of data communication, the CAN protocol simply blankets the Data Link Layer (MAC Medium Access Control, LLC Logical Link Control) and the Physical Layer (PLS Physical Signalling).

  The two ISO documents ISO 11898-2 and ISO 11898-3 cover the two sub-layers of the reference model for data communication: PMA (Physical Medium Attachment) and PMS (Physical Medium Specification). They describe two different CAN physical layers:

- ISO 11898-2 Description of the High-Speed CAN Physical Layer (Transfer rates up to 1 $Mbit/s$).

- ISO 11898-3 Description of the Low-Speed CAN Physical Layer (Transfer rates up to 125 $Kbit/s$).

Fig. B.1 shows the relationships between the ISO/OSI reference model of data communication, the CAN standard and its implementation.

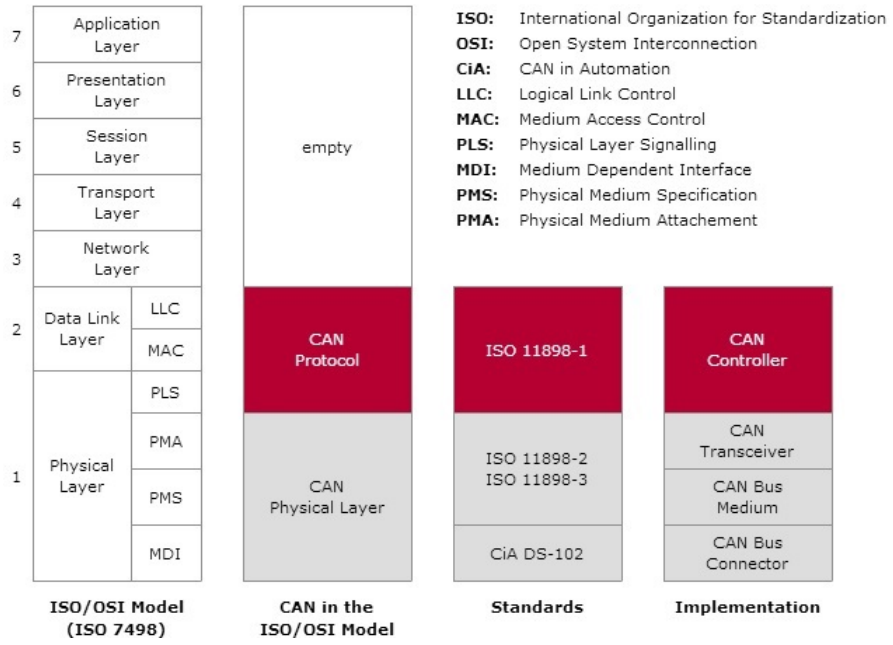


Fig. B.1: Standard and Implementation

So according to the ISO/OSI Reference Model, each standard has its own function.

B.2 Layered Architecture of CAN

Layered Architecture of CAN according to the ISO/OSI Model

- The Physical Layer defines how signals are transmitted and though, it deals with the description of Bit Timing, Bit Encoding and Synchronization. Within this specification, the Driver/Receiver characteristics of the physical layer are not defined so as to allow transmission medium and signal level implementations to be optimized for their application.
- The MAC sublayer represents the kernel of the CAN protocol. It presents messages received by LLC sublayer and accepts messages to be transmitted to the LLC sublayer. The MAC sublayer is responsible for Message Framing, Arbitration, Acknowledgement, Error Detection and Signalling. The MAC sublayer is supervised by a management entity called Fault Confinement which is self-checking mechanism for distinguishing short disturbances from permanent failures.
- The LLC sublayer is concerned with Message Filtering, Overload Notification and Recovery Management.

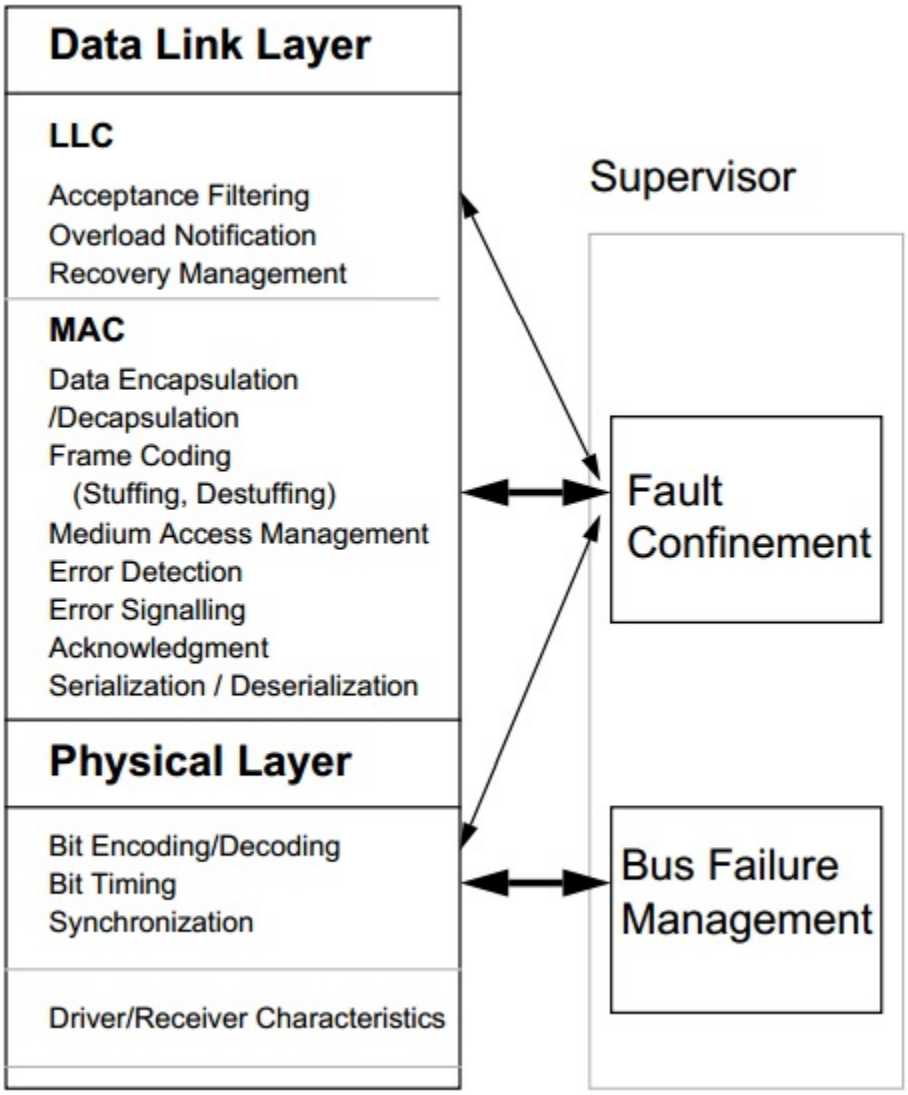


Fig. B.2: Layered Architecture of CAN

The scope of this specification is to define the Data Link Layer and the consequences of the CAN protocol on the surrounding layers.

B.3 Data Protocols

1. **Data Frame**

   A data frame can transport a maximum payload of eight bytes. For that there is the so-called data field, which is framed by many other fields that are required to execute the CAN communication protocol. They include the message address (identifier or ID), data length code (DLC), checksum (cyclic redundancy check sequence CRC sequence) and RX acknowledgement located in the acknowledgement field.

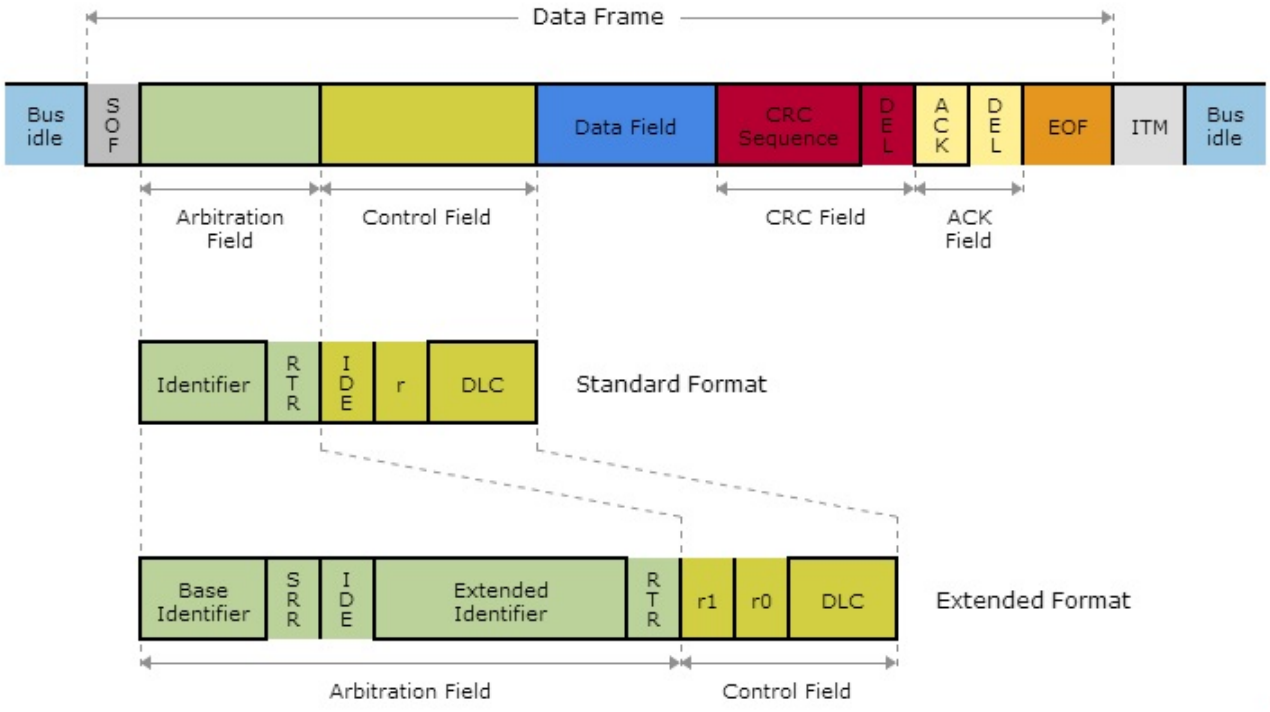


Fig. B.3: Data Frame in Standard and Extend Format

A data frame is made up of numerous distinctive segments. Every individual segment does a critical errand throughout transmission. Assignments to be performed are: Initiate and keep up synchronization between communication partners, secure the communication connections characterized in the communication matrix and transmit and ensure the user data.

It is composed of seven different bit fields:
Start Of Frame, Arbitration Field, Control Field, Data Field, CRC Field, ACK field, End Of Frame

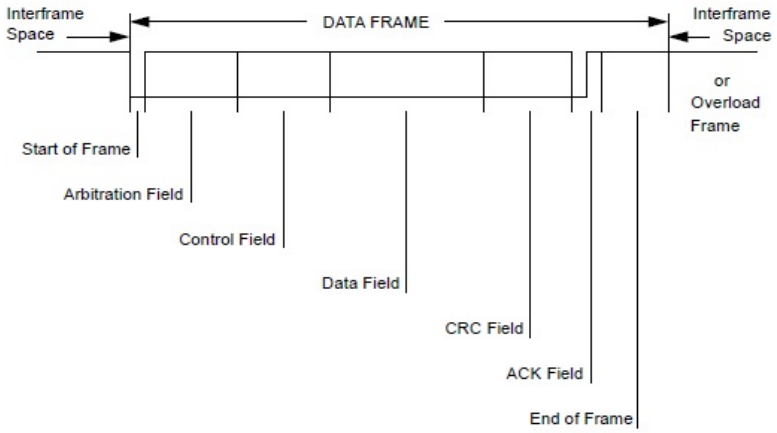


Fig. B.4: Data Frame

Start Of Frame:
Transmission of a data frame begins with the start bit SOF. It marks the beginning of Data Frames and Remote Frames. It consists of a single dominant bit. A station is only allowed to start transmission when the bus is idle (previous recessive level). All stations have to synchronize to the leading edge caused by the Start of Frame of the station starting transmission first.

Arbitration Field:
This field consists of the Identifier and the RTR-Bit.

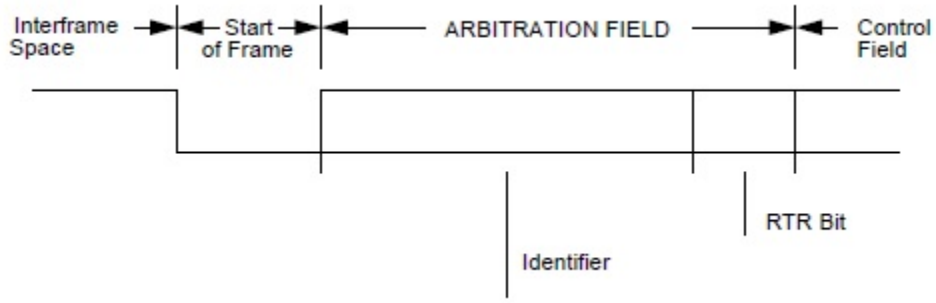


Fig. B.5: Arbitration Field

- **Identifier (ID)**
  This sets the priority of the data frame, and together with the acceptance filtering it provides for sender-receiver relations in the CAN network that are defined in the communication matrix. The Identifier's length is 11 bits. These bits are transmitted in the order from ID-10 to ID-0. The least significant bit is ID-0. The 7 most significant bits (ID-10 - ID-4) must not be all recessive.
- **RTR Bit**
  It is used by the sender to inform receivers of the frame type. In the data frames, the RTR bit has to be dominant, however, it has to be recessive in the Remote Frame

Control Field:
This field consists of sex bits. It includes Data Length Code and two bits reserved for future expansion. The reserved bits have to be sent dominant. Receivers accept dominant and recessive bits in all combinations.

- **Data Length Code (DLC)**
  The DLC communicates the number of payload bytes to the receivers. The payload bytes are transported in the data field. A maximum of eight bytes can be transported in one data frame. The number of bytes in the data field is indicated by the data length code. This data length code is 4 bits wide and is transmitted within the Control Field.

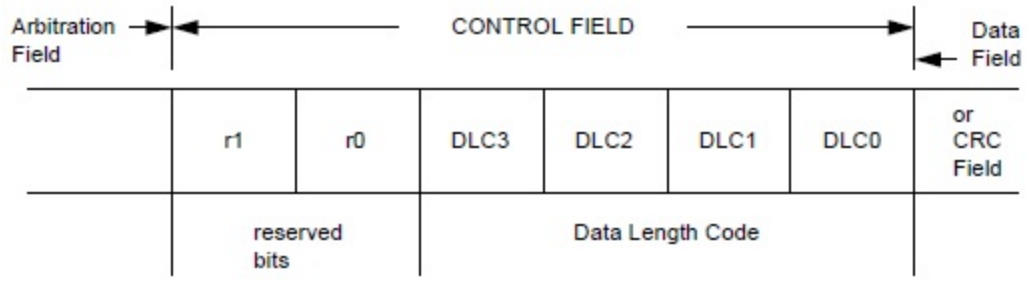


Fig. B.6: Data Length Code

Coding of the number of data bytes by the Data Length Code:

- **d** – Dominant
- **r** – Recessive

Tab. B.1: Coding of the number of data bytes by the Data Length Code

Number of Data Bytes	Data Length Code			
	DLC3	DLC2	DLC1	DLC0
0	d	d	d	d
1	d	d	d	r
2	d	d	r	d
3	d	d	r	r
4	d	r	d	d
5	d	r	d	r
6	d	r	r	d
7	d	r	r	r
8	r	d	d	d

Data Field:

The Data Field consists of the data to be transferred within a Data Frame. It can contain from 0 to 8 bytes, with each contain 8 bits which are transmitted in MSB first.

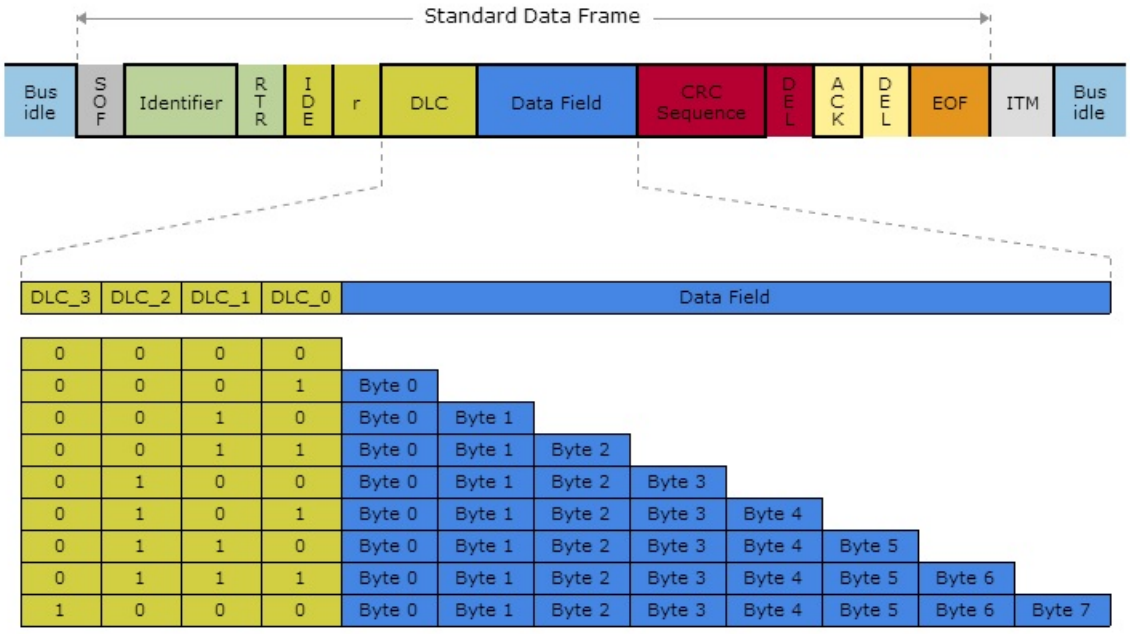


Fig. B.7: DLC and Data Field

CRC Field:

The payload is protected by a checksum using a cyclic redundancy check (CRC) which is ended by a delimiter bit.

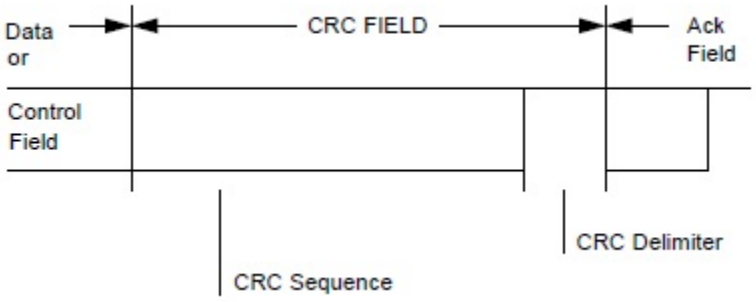


Fig. B.8: CRC Field

ACK Field:

Based on the results of the CRC, the receivers acknowledge positively or negatively in the ACK slot (acknowledgement) which also is followed by a delimiter bit. The ACK Field is two bits long and contains the slot and the delimiter. In the ACK Field, the transmitting station sends two recessive bits. A receiver which has received a valid message correctly, reports this to the transmitter by sending a dominant bit during the ACK slot.

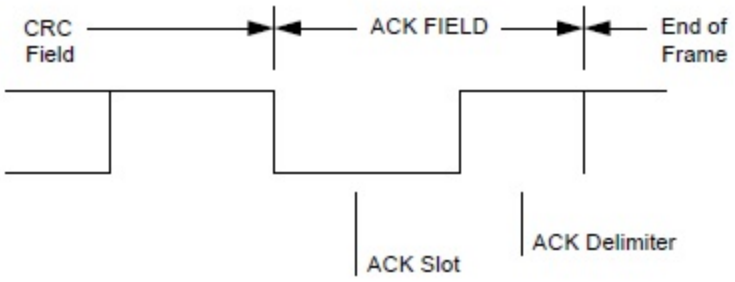


Fig. B.9: ACK Field

End Of Frame:
Each Data Frame and Remote Frame is delimited by a flag sequence consisting of seven recessive bits.

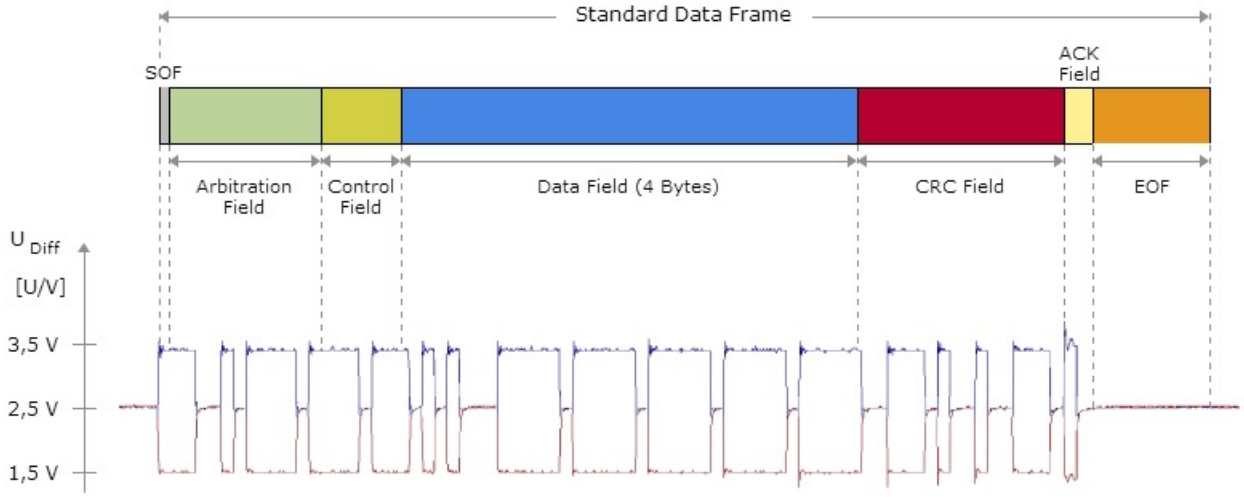


Fig. B.10: Physical Transfer of a Data Frame in Standard Format

2. **Remote Frame**

It is composed of six different bit fields:
Start of frame, Arbitration Field, Control Field, CRC Field, ACK field, End of Frame

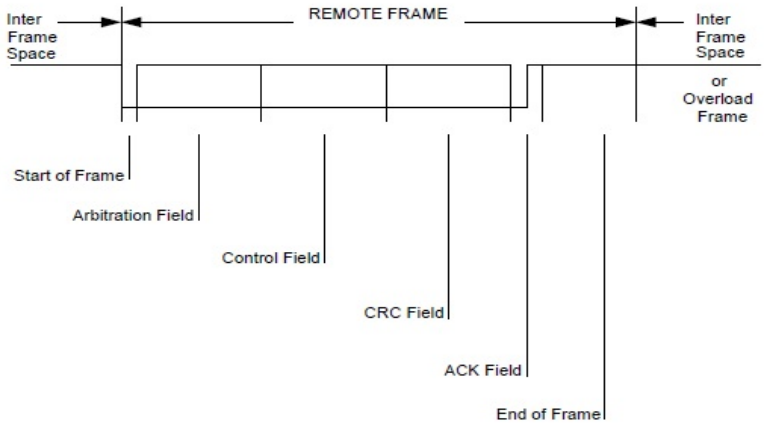


Fig. B.11: Remote Frame

Except for the lack of a data field, the layout of a remote frame identical to that of a data frame. Data and remote frames are differentiated by the RTR bit. In the case of a data frame, the RTR bit is sent as dominant, while it is identified as recessive in the remote frame.
The polarity of the RTR bit indicates whether the transmitted frame is a data frame or a remote frame.

3. **Error Frame**

The error frame consists of two different fields. The first one is given by the superposition of Error Flags contributed from different stations. The next field is the Error Delimiter.

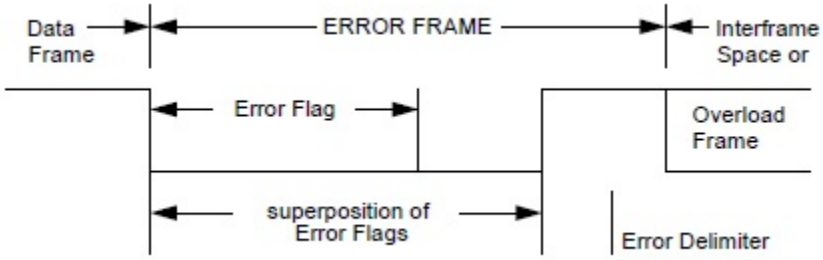


Fig. B.12: Error Frame

In order to terminate an error frame correctly, an 'error passive' node may need the bus to be 'bus idle' for at least 3 bit times (if there is a local error at an 'error passive' receiver). Though, the bus must not be loaded to 100%.

Error Flag

There are 2 forms of an Error Flag the Active and the Passive.

(a) **ACTIVE ERROR FLAG:** It consists of six sequential dominant bits.

(b) **PASSIVE ERROR FLAG:** It consists of six sequential recessive bits unless it is overwritten by a dominant bit from another node.

Error Delimiter

It consists of eight recessive bits. After the transmission of the error flag, each station sends recessive bits and monitors the bus until it detects a recessive bit. Then, it sends seven more recessive bits.

4. **Overload Frame**

The overload frame consists of two bit fields; overload flag and overload delimiter.

There are two kinds of overload conditions which both leads to the transmission of the flag:

(a) The internal conditions of the receiver, which requires delay in Data or Remote Frames.

(b) Detection of a dominant bit during intermission.

Due to overload condition 1, the start of the overload frame is only allowed at the first bit time of an expected intermission, however, due to overload condition 2, the overload frame starts one bit after detecting the dominant bit.

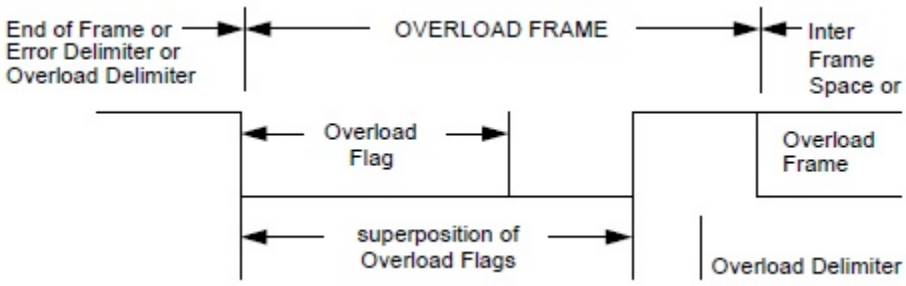


Fig. B.13: Overload Frame

At most two overload frames may be generated to delay the next Data or Remote Frames.

B.4 Signal Analysis

- **Signals**

  Signals are detectable quantities used to convey information about time-varying physical phenomena. A signal is classified into several categories depending upon the criteria used for its classification [44].

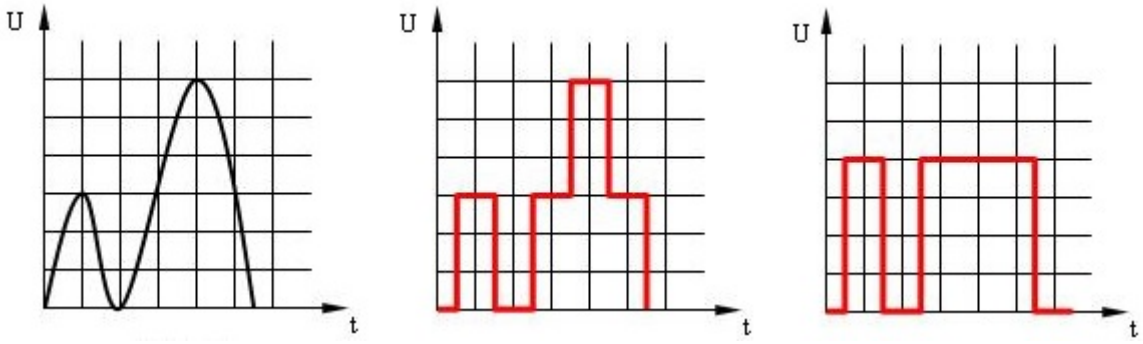
  We have different types of signals:

  i Analog

  ii Digital

  iii Binary

  An analog signal can have any value between 0% and 100%. The signal is thus continuously. However, the development of a digital signal is stepped, it was prepared from the analog signal. A binary signal only knows the states 0 and 1, or High and Low.



(a) Analog Signal *(b)* Digital Signal *(c)* Binary Signal

Fig. B.14: Different Types of Signals [44]

- **Number Systems**

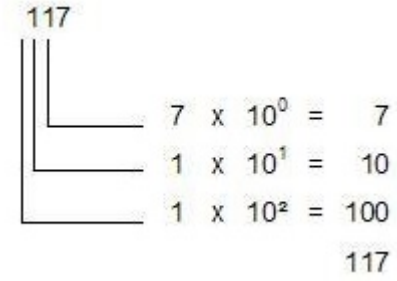
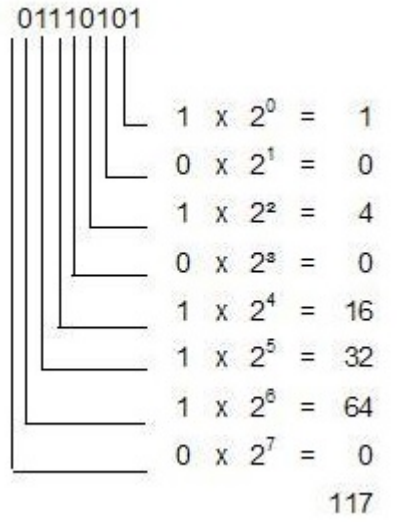
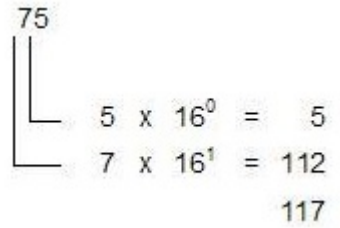
  In computer technology, there are three important number systems:

  i Decimal

  ii Binary

  iii Hexadecimal

Tab. B.2: Different Number Systems

	The decimal (base 10) is the well-known Arabic number system.
	The binary (base 2) is one of the most suitable payment systems in data processing since it has only two states: 0 and 1, or on or off, or high voltage or low voltage. Each character consists of a specific sequence of binary digits (e.g. 10010110). This binary code, the computer or controller process information.
	As a shorthand notation for the hexadecimal system is binary (base 16). With two hexadecimal numbers can be summarized eight-digit binary numbers. The advantage is a shorter notation.

- **Bit**

  A bit (binary digit) is the smallest unit of information (a switching state per unit time) in a data processing system. In electronics, this information can only have the value 0 or 1 or ”yes” or ”no” have.

  To exchange data via CAN information is structured in a specified manner: The basic unit of information is the ”signal”. It may include various ranges of values: at least one bit, for example, ”signal = 1” means ”brake on” or ”signal = 0” means ”brake not activated”, and also 16 or more bits, such as for ”engine torque” or ”vehicle speed”.

  With two bits there are four different variants. Each variant can be assigned to an

information, which is binding on all control units.

As an example of the window is in the door module. If the signal first bit 0 V and second bit is from 0 V door module, this is the Information "window is currently in motion."

Tab. B.3: Window's Signal Example

Possible Variants	**2 bit**	**1 bit**	**Window Information**
one	0 volts	0 volts	in motion
two	0 volts	5 volts	standing still
three	5 volts	0 volts	the area covered
four	5 volts	5 volts	in blocking tendency

- 1 bit = 2 possibilities
- 2 bits = 4 possibilities
- 3 bit = 8 possibilities
- 4 bit = 16 possibilities
- 5 bit = 32 possibilities
- 8 bit = 256 possibilities
- 12 bit = 4096 possibilities

..... and so on

So the formula that is used calculate the number of possibilities from the bits is:

$$\textbf{Number of variants} = 2^n \, (where \, n = number \, of \, bits) \tag{B.1}$$

Examples of Signal Structuring:

1. An engine torque is represented on the CAN with 12 bits, which means that we can have possible integers from 0 to 4095. The conversion into the real torque is carried out according to the rule: $\textbf{Torque} = ((CAN - value * 0.25)200) \, Nm$ For torques of 200 Nm to 823.75 Nm at a resolution of 0.25 Nm can be represented.
2. Signal "Current Gear" of the AT-Transmissions (320-4 Bit):
   - "0" means gear is being changed 0 0 0 0 0 0 0 0 0 0 0 0
   - "1" means 1st gear 0 0 0 0 0 0 0 0 0 0 0 1
   - "2" means 2nd gear 0 0 0 0 0 0 0 0 0 0 1 0
   - "3" means 3rd gear 0 0 0 0 0 0 0 0 0 0 1 1
   - "4" means 4th gear 0 0 0 0 0 0 0 0 0 1 0 0

  - "8" means reverse 0 0 0 0 0 0 0 0 1 0 0 0
  - "9" means Neutral

  All other representable numbers from "0" to "2047" may are not used.

  Attention! This is not the same conversion for all vehicles; it is just an example.

- **Converting Binary to Decimal**

Tab. B.4: Binary-Decimal Conversion

Code							
7	6	5	4	3	2	1	0
Power of 2							
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Results							
128	64	32	16	8	4	2	1
Example Code: 01001110							
0	1	0	0	1	1	1	0
For each position, a 1(set) or 0(not set) will be assigned							
0	64	0	0	8	4	2	0
Only the positions which has 1(set) are filled with the values entered above							
The decimal value is the total of all items here, which is: 64 + 8 + 4 + 2 = 78							

- **Binary Coded Decimal**

  For the conversion of larger numbers in figures of another number system, a larger computational effort is required. By an encoded representation of the number of parts can reduce the cost of the conversion. The number of digits required for the coded representation is larger than that of a direct conversion, but clearer. The decimal numbers can have in each location 10 different digits. 4 points are required for the binary representation of 10 elements. A binary-coded decimal number has therefore each point of a binary decimal foursome.

  For conversion of the decimal number 316 in binary coded decimal

Tab. B.5: Large number conversion example

3	1	6
0011	0001	0110

  The binary coded decimal number 316 in this case is composed of three groups of numbers of four digits.

All bit combinations in the binary coded number system can not be exploited because the decimal number system so has only 10 digit characters. Thus, all four groups can be exploited further characters were for those prohibited in binary coded decimal code, imported from the alphabet.

The basis for the hexadecimal system is 16 in a byte, a two-digit hexadecimal number to be encoded.

Example:
Hexadecimal: $C4_{16}$
Binary Coded: 1100 0001

where: A=10, B=11, C-12, D=13, E=14, F=15

So the decimal value will be: $C * 16^1 + 4 * 16^0 = C * 16 + 4 * 1 = 192 + 4 = 196$

If there is a character in a hexadecimal number, it is easy to see that number as a hex number. At a number of digits detecting a hexadecimal number is not possible. For an unambiguous assignment of the spelling of a hex number is provided with an h as Postfix or $ as a prefix.

Examples:
$FE = 254 (15*16 + 14*1); $12 = 18 (1 *16 + 2*1); 22h = 34 (2*16 + 2*1); ABh = 171 (10*16 + 11*1)

Tab. B.6: Conversions of Number Systems

bin	hex	dec
0000	0	00
0001	1	01
0010	2	02
0011	3	03
0100	4	04
0101	5	05
0110	6	06
0111	7	07
1000	8	08
1001	9	09
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

- **A Byte**

  A byte is composed of eight bits. It can be personalized with a byte 256 represent different characters.

  - 1 kilobyte (KB) = 2^{10} bytes = 1024 bytes
  - 1 megabyte (MB) = 2^{20} bytes, so 1024 KB (1,048,576 bytes)
  - 1 gigabyte (GB) = 2^{30} bytes, i.e. 1024 MB (1,073,741,824 bytes)

  The conversion is not exactly a factor of 1000, but a factor of 1024!

  CAN uses a non-return-to-zero coding (NRZ). This one bit a state of the logic level is assigned to exactly, low = 0 and high = 1. Several identical bits in succession appear on the bus so as a correspondingly longer constant level. Since the CAN clock is not transmitted, basically after five equal bits a so-called stuffing (Stuffbit) is inserted. The low level is dominant on the CAN bus. When recessive high level and low-dominant level is transmitted on the bus simultaneously appears on the bus, the low level.

  It should be noted that the signals in every vehicle differs in the byte format. We have two byte formats; the Byte Array Format "Motorola" and the Byte Request Format "Intel". In the "Motorola", we have the rearmost byte in the order is the LSB (Least Significant Byte), whereas the first byte in the order is the MSB (Most Significant Byte). However, in the "Intel" format, it is arranged oppositely.

(a) Motorola Processor *(b)* Intel Processor

Fig. B.15: Byte Order for different processors

However, the significance of the bits within a byte is equal in both formats:

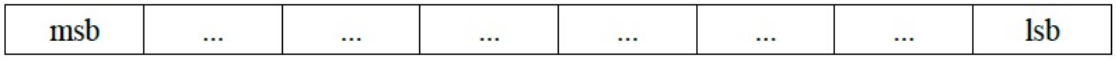


Fig. B.16: Significance of bits

- **Bit time and transfer rate**

  The transmission speed is in the form of the transfer rate specified. This specifies the maximum number of transmittable bits in one second. Thus, the unit of transmission rate "bits per second", or "$bits/s$" .

For the calculation of transfer rate, the time it takes for a single bit for the transmission passes (the so-called " bit time ") a significant role. Since such bit times are only fractions of a second, the bit time is in "microseconds". A microsecond is one millionth of a second known. If you know the bit time, the transmission rate can be easily calculated: Transfer rate = 1 bit/bit time.

APPENDIX C

Citroën Car

C.1 Technical Specifications

Citroën - C6 - 2.7 V6 HDI (208 Hp)

Tab. C.1: Technical characteristics of Citroën C6

Brand	Citroen
Model	C6
Generation	C6
Engine	2.7 V6 HDI (208 Hp)
Doors	4
Power	208 Hp
Maximum speed	214 km/h
Acceleration from standstill to 100 km/h	11.2 sec
Fuel tank volume	72 l
Year of putting into production	2004 year
Coupe type	Sedan
Seats	5
Length	4908 $mm.$
Width	1860 $mm.$
Height	1464 $mm.$
Wheelbase	2900 $mm.$
Front track	1558 $mm.$
Rear (Back) track	1586 $mm.$
Minimum volume of Luggage (trunk)	421 l
Position of engine	Front, transversely
Volume of engine	2721 cm^3
Max power in	4000 $rpm.$
Torque	440/1900 Nm
Fuel System	Diesel Commonrail

Turbine	Turbocharging (Supercharging)
Position of cylinders	V engine
Number of cylinders	6
Number of valves per cylinder	4
Fuel Type	diesel
Wheel Drive	Front
Number of Gear (automatic Gearbox)	6
Front suspension	Hydro-pneumatic element
Rear suspension	Hydro-pneumatic element
Front brakes	Disc
Rear brakes	Disc
ABS	yes
Steering type	Steering rack
Power steering	Hydraulic Steering
Fuel consumption (economy) - urban	10.6 $l/100km$.
Fuel consumption (economy) - extra urban	5.9 $l/100km$.
Fuel consumption (economy) - combined	7.6 $l/100km$.
Emission standard	EURO IV
Weight	1871 kg.
Max weight	2335 kg.
Tire size	225 / 55 / R17

C.2 Electronic Control Units (ECUs)

This table is showing the different ECUs inside the car

Tab. C.2: Identification of the components of the test vehicle

Number	Component Name
0004	Instrument Cluster
1320	Injection Controller
1630	Automatic Transmission Control Unit
2110	Additional Brake Light
2120	Two-Function Brake Switch
2610	Left Headlight
2630	Left Rear Light
2635	Right Rear Light
4001	Control Panel, The Upper Field of Vision
5008	Rain / Brightness / Tunnel sensor
6031	Motor + Control Unit Sequential Windows Front Passenger Side
6032	Motor + Control Unit Sequential Windows Front Driver's Side
6304	Controller Position Memory Front Passenger Seat
6338	Control Unit Seat Storage

6348	Control Unit Location back-seat
6393	Programming Board Driver's Seat
6394	Programming Board Passenger Seat
6570	Control Unit Airbag and Belt Tensioners
6590	Control Unit Detection Impact Pedestrian
6910	Portable Wind Deflector
7092	Control Electric Parking Brake
7095	Assembly Electric Parking Brake
7215	Multi-Function Screen
7500	Parking Aid Control Unit
7550	Control Unit for controlling the Lateral Stability
7600	Control Unit - Tire Pressure Detection
7715	Suspension Control Device
7758	Control Unit Variable Damping
7800	Controller Stability Control
7804	Rotational Speed Sensor, Accelerometer Stability Control
8025	Control Unit for Heating / Ventilation / Air Conditioning
8026	Front Panel Auxiliary Air Conditioning
8410	Car Radio
8415	CD Changer
8480	Transmitter / Receiver Telematics
8602	Control Unit Interior Protection
BB00	Battery
BB10	Controller Power Supply
BFH5	Box 5 Fuse(s) Inside
BSC1	Switching Unit Trunk
BSI1	Central Switch Unit BSI
C001	Diagnostic Connector
CA00	Ignition
CV00	Switching Module (COM 2000)
PSF1	Programming Board - Engine Compartment Fuse Box

So each of the 3 CAN Lines is connected to some of these ECUs, and here is a simple chart showing the connections of 3 CANs in the car with the corresponding ECUs.

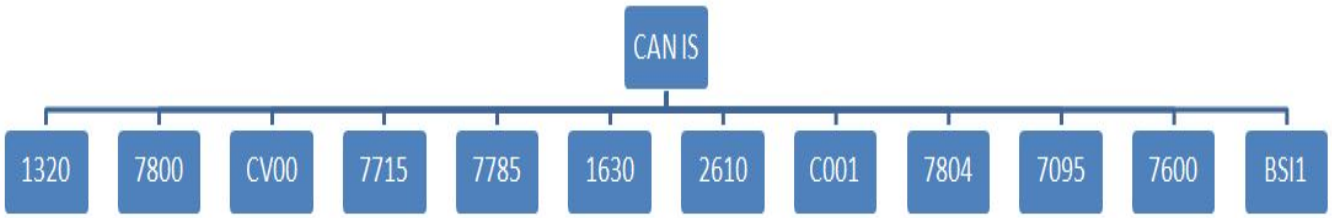


Fig. C.1: ECUs on CAN IS

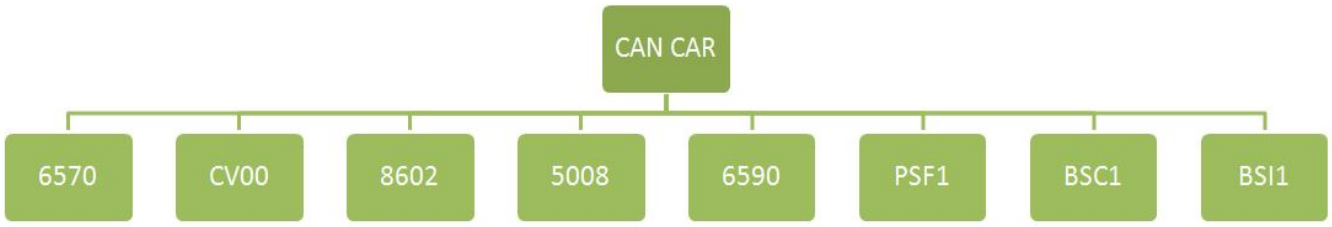


Fig. C.2: ECUs on CAN CAR

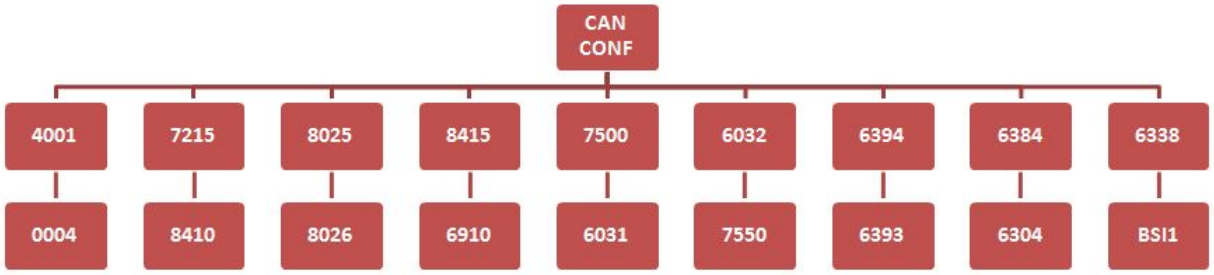


Fig. C.3: ECUs on CAN CONF

APPENDIX D

Radar Sensor

Fig. D.1: Short Range Radar Sensor [4]

D.1 Sensor Objectives

Here are all data information related to the parameters of the sensor.

Tab. D.1: Sensor Parameters

Radar Parameters	
Radar Principle	Pulse Radar
Frequency	24.125 GHz, UWB Compliant
Tx Power	<17 dBm EIRP
Bandwidth	4 GHz @ 20 dB points
Pulse Width	1.0 .. 1.5 ns
Pulse Repetition Frequency	≈ 3 MHz
Detection Cycle Time	40 ms
Antenna Detection Characteristics	±8° Elevation 3 dB limit ±65° Azimuth 3 dB limit

Sensor Network Local Interface	Private CAN 2.0B 500kB
Power Supply (Sensors)	Vehicle Power
Power Consumption (Sensors)	< 5 W Each
Operating Temperature	-40 to +85 deg C
Sensor Size (cm)	9.5 x 6.3 x 2.1
Object Parameters	
Number of Objects	10
Parameters Types	Range, Bearing, Velocity, Yaw Rate, Quality, Track ID
Range Parameters	
Detection Range	30 m
Range Performance	20 m (10 sqrm target)
Range (Object) Resolution	15 cm
Range Accuracy	± 3 cm
Bearing Parameters	
Bearing Detection Principle	Switched Mono-pulse
Bearing Detection Range	$\pm 40°$
Bearing Accuracy	$\pm 2°$ for Bearing Range $\pm 5°$ $\pm 5°$ for Bearing Range $\pm 5°$ .. $25°$ $\pm 10°$ for Bearing Range $\pm 25°$ .. $40°$
Velocity Parameters	
Velocity Detection Principle	Range Rate Velocity Measurement
Velocity Range	$\pm$ 100 km/h
Velocity Accuracy	$\pm$ 1 km/h
Velocity Resolution	$\pm$ 0.05 m/s
Tracking Parameters	
Tracking Principle	Extended Kalman Filter Approximation
Track Initiation	Retrospective Detector (4 in a row)
Track Kill	4 consecutive cycles with no detection

D.2 Sensor Connectors

For building a connector to mate with the sensor, the following parts are required:

Tab. D.2: Connector Parts

Required Quantity	Part Number	Description
1	872-665 Keying A	Hirschmann Connector

Tab. D.3: Sensor Connector Pin Usage

Pin Number	C2 SLR
1	Batt - (GND)
2	CANH
3	CANL
4	Batt+
5	MPIO_0
6	MPIO_1

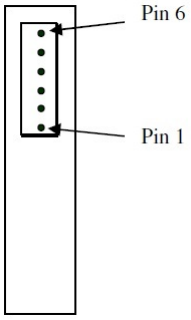
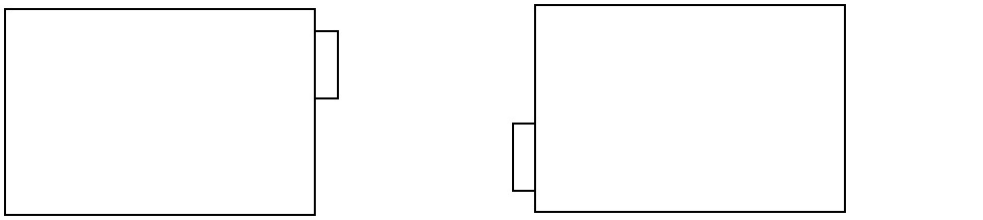


Fig. D.2: Sensor Side View looking into connector mating face



(a) Normal sensor orientation *(b)* Upside down sensor orientation

Fig. D.3: Sensor Orientation

D.3 Power Supply Requirements

Tab. D.4: Power Supply Requirements

Power Supply Parameter	Value
Current	400 mA nominal @ 12V
Voltage	9 - 16 V DC

D.4 Sensor Address

An SLR sensor must determine its own address in order to appropriately transmit some messages on the CAN bus. Sensor address is determined by interacting with the cabling harness through the address input pins (MPIO_0, MPIO_1)on its connector. The mapping of the sensor address (sensor identifier) and the address input lines are as follows:

Tab. D.5: Sensor Address

Sensor Address	MPIO_1	MPIO_0
	0 = Pin connected to $0V$ (GND)	
	1 = No connection	
1	0	0
2	0	1
3	1	0
4	1	1

D.5 CAN Interface

The CAN standard is CAN V2.0 part A. The physical interface is compatible with ISO/DIS 11898. Table 2.16 presents bit-timing details for the CAN bus.

Tab. D.6: Bit-Timing Parameters for CAN Bus

CAN Bus Parameter	Value
Sampling Mode (Intel 82527 register)	0
Sync Segment	200 nSec
Time Segment 1	1200 nSec
Time Segment 2	600 nSec
Data Rate	500 kbps
Synchronization Jump Width	200 nSec

The following table shows the different messages that can be transmitted and received by the sensors and RDU.

Tab. D.7: RDU CAN Communication Messages (n: Sensor Number)

Messege	ID	DLC	RDU	Sensor n
Sync-Message	0x100	8	Tx	Rx
Command-Message	0x11n	7	Tx	Rx
Target-Messages	0..9 0x3n0-0x3n9	8	Rx	Tx
Measurement-Data Message	0x2n0-0x2nF	8	Rx	Tx
Error Management Message	0x4Fn	8	Rx	Tx
Status Message	0x40n	8	Rx	Tx

D.6 Message Content

The contents of the messages transferred in Measure_0-Mode are described by the following tables:

1. ***Sync Message (RDU to Sensor)***

   CAN ID: 0x100

Tab. D.8: Content of Sync Message

Data- Byte No.	Bit No.	Function	Resolution Offset	Range
0	7-4	Mode for sensor 1	N/A	Front Bumper Mode = 0x2
	3-0	Mode for sensor 2	N/A	Front Bumper Mode = 0x2
1	7-4	Mode for sensor 3	N/A	Front Bumper Mode = 0x2
	3-0	Mode for sensor 4	N/A	Front Bumper Mode = 0x2
2	7-0	Vehicle Velocity	0.5 m/s	-127 - +127, -128 = unknown velocity Vehicle movement: Forwards Front CAN velocity sign: + Rear CAN velocity sign: - Vehicle movement: Rearwards Front CAN velocity sign: - Rear CAN velocity sign: +
3	7-4	Message Counter	N/A	0-15
	3-0	Bumper Type	-	Bumper Shape Type 0: Default type 1-15: specific types of bumper
4	7-0	PRF Control	N/A	0/255: default mode, 1-254: TBD the value is toggled cyclic between 0x00 and 0xFF
5	7-0	Reserved	-	Reserved Bytes
6	7-0	Reserved		
7	7-0	Reserved		

2. ***Command Message (RDU to Sensors)***

CAN ID: 0x11n (n: Sensor Address)

Tab. D.9: Content of Command Message

Data- Byte No.	Bit No.	Function	Resolution	Range
0	7-4	Message Counter	N/A	0-15

	3-0	Measurement Control	N/A	0: Send Data measurement messages 2: Sensor command transmitted 15: Do not send data measurement messages
1	7-0	Sensor Rotation	1 deg/bit	-127 - +127, -128 = unknown
2	7	Sensor Orientation (Geometry)	Boolean	0: Normal Orientation 1: Upside-Down Orientation
	6-0	Reserved	-	Spare, used when an additional bits are necessary
3	7-0	Reserved	-	Spare, used when an additional command parameters are necessary
4	7-0	Reserved	-	
5	7-0	Reserved	-	
6	7-0	Reserved	-	

3. ***Target Messages (Sensor to RDU)***

CAN ID: 0x3n0 - 0x3n9 (n: Sensor Address)

Tab. D.10: Content of Target Messages

Data- Byte No.	Bit No.	Function	Resolution	Range
0	7-0	Range Filtered (11:4)	0.01 m/bit	0 - +4094, +4095 = unknown distance: this value indicates that the following targets are not valid
1	7-4	Range Filtered (3:0)		
	3-0	Velocity (11:8) Filtered	0.05 $m/s/bit$	-2047 - +2047, -2048 = unknown velocity
2	7-0	Velocity (7:0) Filtered		
3	7-4	SNR (3:0)	2 db/bit	$10dB + 2dB$*SNR(3:0) $0 \rightarrow 10dB$, $14 \rightarrow 38dB$, $15 \rightarrow$ unknown SNR
	3-0	Track ID	N/A	0 - 14, 15 = unknown ID

4	7-0	Bearing Filtered (7:0)	1 deg/bit	-126 - +126 = bearing value +127 = left hand side (bearing value unknown) -127 = right hand side (bearing value unknown) -128 = bearing value and side unknown
5	7-4	Mesg-Cntr (3:0)	N/A	0 - 15
	3-0	Sensor-No (3:0)	N/A	1-4; 0/5 - 15 = undefined
6	7-5	Yaw-Rate Filtered	50 $deg/sbit$	-3 - +3 -4 unknown yaw rate
	4	ObservedThisCycle	boolean	0: not observed, 1: observed this cycle
	3-0	Bearing Observed (5:2)	1 deg/bit	-31 - +31 deviation from filtering -32 unknown bearing deviation
7	7-6	Bearing Observed (1:0)		
	5-0	Range Observed (5:0)	0.01 m/bit	-31 - +31 deviation from filtering -32 unknown range deviation

Calculation of the unfiltered values conforms to following rules:

Tab. D.11: Calculation Rules for unfiltered values

Function	Filtered Value	Observed Value (Deviation)	Unfiltered Value
Range	UNKNOWN	DON'T CARE	UNKNOWN (Whole target information is not valid
	KNOWN	UNKNOWN	UNKNOWN
	KNOWN	KNOWN	RangeUnfiltered = RangeFiltered + RangeObserved *if the sum is not in the allowed range, RangeUnfiltered value will be UNKNOWN*
Bearing	UNKNOWN	DON'T CARE	UNKNOWN
	KNOWN	UNKNOWN	UNKNOWN

	KNOWN	KNOWN	BearingUnfiltered = BearingFiltered + BearingObserved *if the sum is not in the allowed range, RangeUnfiltered value will be UNKNOWN*

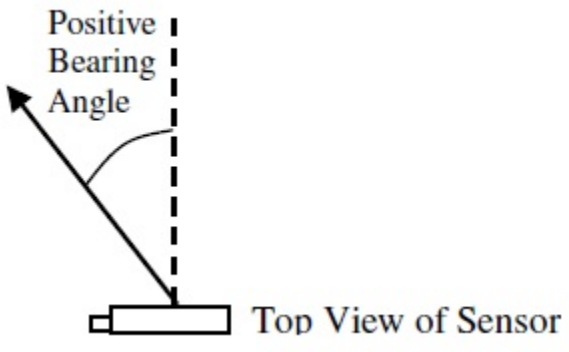


Fig. D.4: Definition for Sign of Target Bearing

4. ***Error Management Status Message (Sensor to RDU)***

   CAN ID: 0x4Fn (n: Sensor Address)

Tab. D.12: Content of Error management-Message

Data- Byte No.	Bit No.	Function	Range
0	7-0	Error State [3:0]	Error State = Condition + Readiness Condition: 0 error inactive, 1: error active Readiness: 0 path is executed, 1: path is not executed Upper-Bit = Condition, Lower Bit + Readiness
1	7-0	Error State [7:4]	
2	7-0	Error State [11:8]	
3	7-0	Error State [15:12]	
4	7-4	Error State [19:16]	
5	7-4	Message Counter	0 - +15
	3-0	Sensor Number	1 4; 0 / 5-15 = undefined
6	7-0	Error State [23:20]	Error States
7	7-0	Error State [27:24]	

5. ***Status Message (Sensor to RDU)***

   CAN ID: 0x40n (n: Sensor Address)

Tab. D.13: Content of Status-Message

Data- Byte No.	Bit No.	Function	Range
0	7-0	VCO Voltage (15-8), high byte	DAC setting for VCO voltage. Range = 0-255 VCO voltage = 5.0V * DAC setting / 256
1	7-0	VCO Voltage (7-0), low byte	
2	7-0	Diagnostic Code (15-8), high byte	0 = everything OK 1 65535 = diagnostic codes
3	7-0	Diagnostic Code (7-0), low byte	
4	7-4	Sensor Mode	Current sensor mode for generated targets
	3-0	Reserve	Actual not used
5	7-4	Message Counter	0 - +15
	3-0	Sensor Number	1- 4; 0/ 5-15 = undefined
6	7-0	MD Byte 15	
7	7-0	MD Byte 16	

APPENDIX E

FlexDevel Technical Data

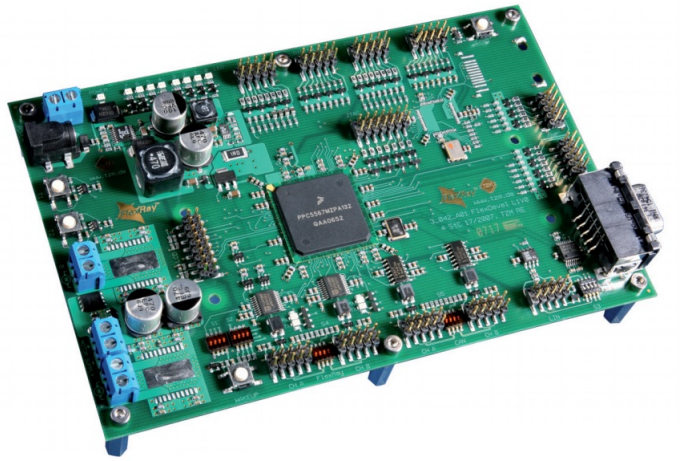


Fig. E.1: FlexDevel Conroller - Article 3-042-P01

FlexDevel Technical Data

1. Electrical Characteristics

Tab. E.1: Electrical Characteristics

Supply Voltage				
		Min.	Typ.	Max.
Operating		9.0V	12.0V	16.0V
Absolute Maximum		-18.0V	-	18.0V
Quiescent Supply Current	with DC-Motor Supply	Typ.: 10mA @ 12V		
	withoud DC-Motor Supply	Typ.: 0.5mA @ 12V		
Supply Current in Operating Mode		Typ.: 150mA @ 12V		

2. Physical Characteristics

Tab. E.2: Physical Characteristics

Connectors	
1 x Power supply	Round DC jack 2pol.
1 x Power supply	Screw block 2pol.
1 x Ethernet	RJ45 8pol.
1 x RS232	Sub-D 9pol.
1 x Motor supply	Screw block 2pol.
2 x Motor terminal	Screw block 2pol.
2 x FlexRay	2 x 5 pin 2,54mm 10pol.
2 x CAN	2 x 5 pin 2,54mm 10pol.
1 x LIN	2 x 5 pin 2,54mm 10pol.
2 x SPI	2 x 5 pin 2,54mm 10pol.
1 x Analog in	2 x 5 pin 2,54mm 10pol.
1 x Digital out	2 x 5 pin 2,54mm 10pol.
1 x Digital in	2 x 5 pin 2,54mm 10pol.
1 x PWM out	2 x 5 pin 2,54mm 10pol.
Weight approx.	150 g
Dimensions approx.	L x W x H 177 x 110 x 20 mm^3

3. Environmental Conditions

Tab. E.3: Environmental Conditions

Temperature	Operating / Non-operating: 0 - 55°C Storage: 0 - 70°C
Relative Humidity	0% to 90% r.H., non-condensing

4. Block Diagram

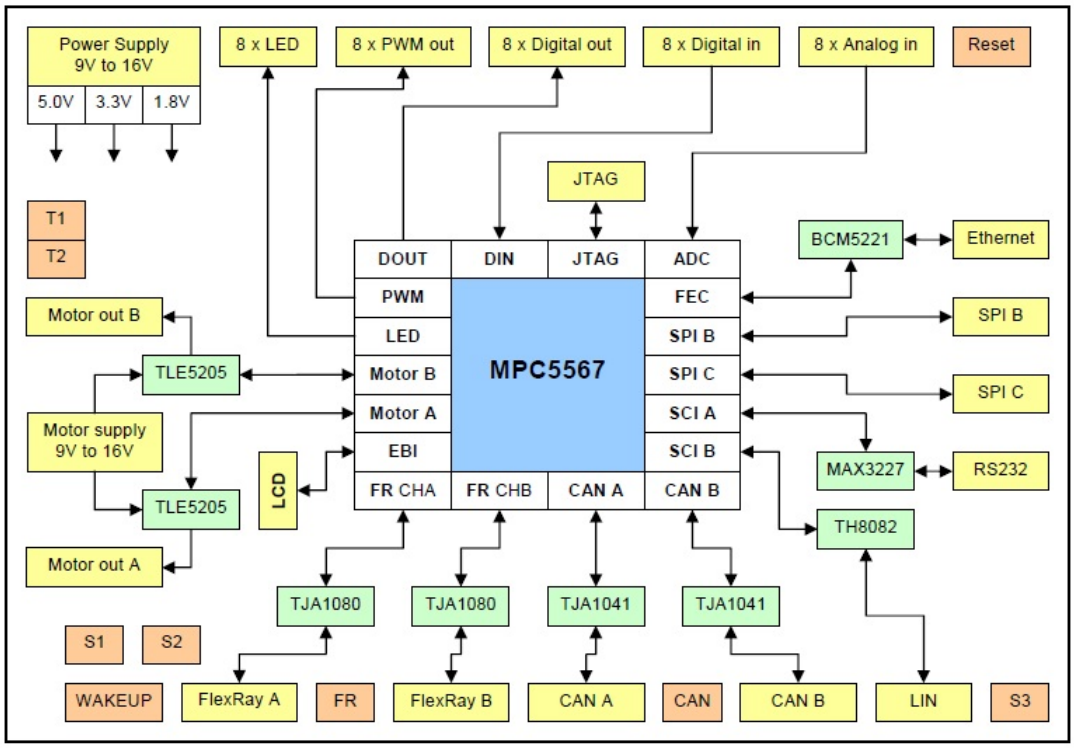


Fig. E.2: FlexDevel Block Diagram

5. Layout Diagram

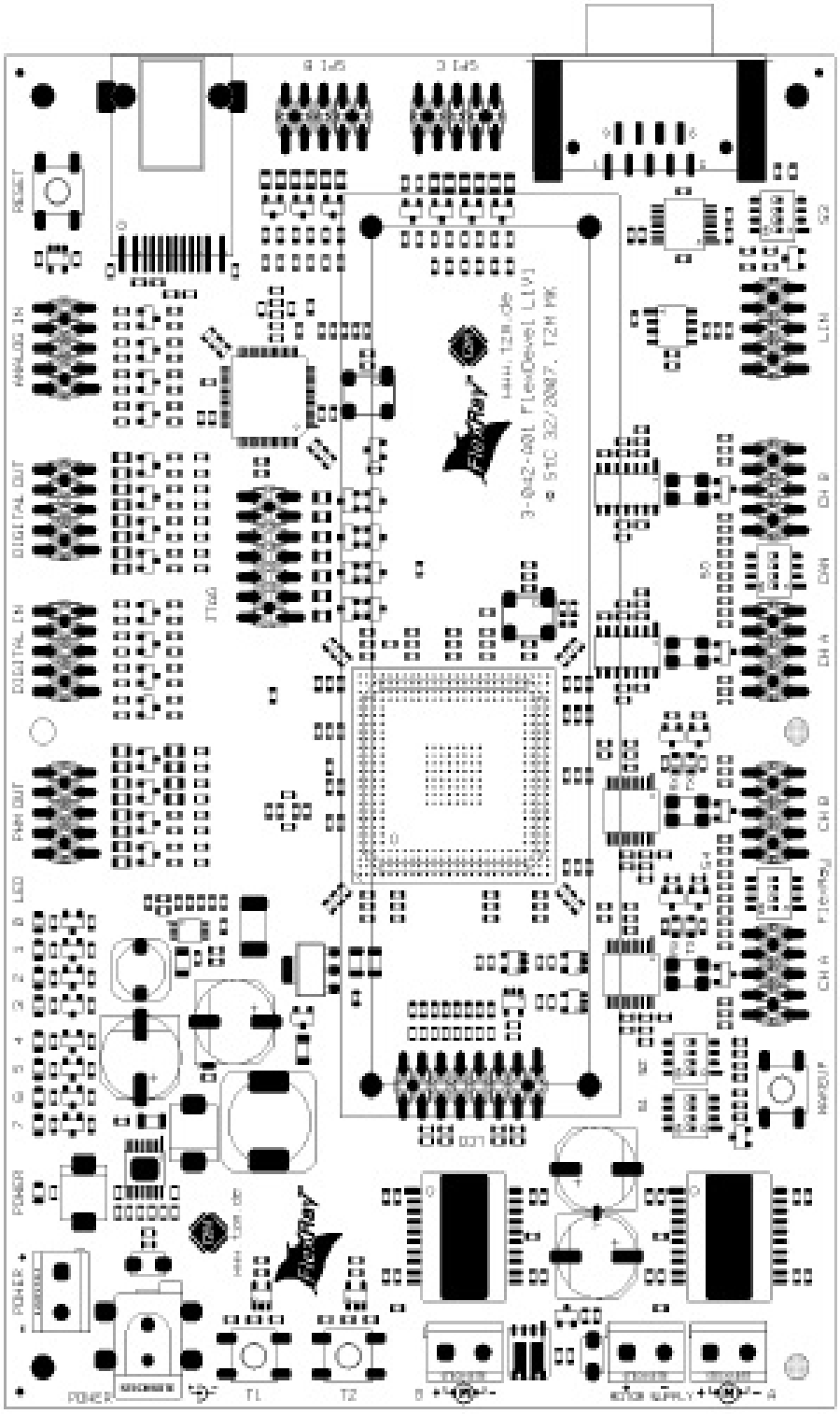


Fig. E.3: FlexDevel Layout Diagram

APPENDIX F

S500 Mercedes-Benz W221 Car

Tab. F.1: Mercedes-Benz S500 CAN BUS

Identifier Letter	Designation	Baud Rate	Coloring	Level
A	Telematics-CAN	125 $KBit/s$	Black (BK) Black/White (BK/WH)	Low High
B	Interior-CAN	125 $KBit/s$	Brown (BN) Brown/Red (BN/RD)	Low High
C	Drive-CAN	500 $KBit/s$	Blue (BU) Blue/White (BU/WH)	Low High
D	Diagnose-CAN	500 $KBit/s$	Gray (GY) Gray/White (GY/WH)	Low High
E	Landing Gear-CAN	500 $KBit/s$	Green (GN) Green/White (GN/WH)	Low High
F	Central-CAN	500 $KBit/s$	Yeloow (YE) Yellow/White (YE/WH)	Low High
G	Front Area-CAN	500 $KBit/s$	Violet (VT) Violet/White (VT/WH)	Low High
H	Driving Dynamics-CAN	500 $KBit/s$	Violet (VT) Violet/White (VT/WH)	Low High

APPENDIX G

GPS Receiver

Fig. G.1: GR-213 GPS Receiver

G.1 Technology Specifications

G.1.1 Physical Dimension

Single construction integrated antenna/receiver. Size: 64.5 x 42 x 17.8 (mm) 2.54" x 1.65 x 0.7 (Inch).

G.1.2 Environmental Characteristics

1. Operating temperature: $-40°C$ to $+80°C$(internal temperature).
2. Storage temperature: $-45°C$ to $+100°C$.

G.1.3 Electrical Characteristics

1. Input voltage: $+4.5 \approx 5.5$ VDC without accessories.
2. Backup power: 3V Rechargeable Lithium cell battery, up to 500 hours discharge.

G.1.4 Performance

1. Tracks up to 20 satellites.
2. Update rate: 1 second.
3. Acquisition time
   - Reacquisition 0.1 sec., averaged
   - Hot start 1 sec., averaged
   - Warm start 38 sec., averaged
   - Cold start 42 sec., averaged
4. Position accuracy:
   (a) Non DGPS (Differential GPS)
      - Position 5-25 meter CEP without SA
      - Velocity 0.1 meters/second, without SA
      - Time 1 microsecond synchronized GPS time

   (b) DGPS (Differential GPS)
      - Position 1 to 5 meter, typical
      - Velocity 0.05 meters/second, typical

   (c) EGNOS/WAAS/Beacon
      - Position $<$ 2.2 meters, horizontal 95% of time , $<$ 5 meters, vertical 95% of time

   - Dynamic Conditions:
   - Altitude 18,000 meters (60,000 feet) max
   - Velocity 515 meters / second (700 knots) max
   - Acceleration 4 G, max
   - Jerk 20 meters/second, max

G.1.5 Protocal&Interface

- NMEA Protocol Output : V 2.2
- Standard
  - Baud Rate : 4800 *bps*
  - Data Bit : 8
  - Parity : N
  - Stop Bit : 1

  - Format : GGA, GSA, GSV, RMC

- Optional :
  - Baud Rate : 9600,19200,38400 *bps*
  - Format : GLL, VTG, ZDA, SiRF Binary

- Interface:
  - RS-232 + COMS TTL Level, or
  - RS-232 + DGPS

G.1.6 LED Function

- Power On/Off and Navigation
- Update Indication

[23]

APPENDIX H

NMEA Transmitted Messages

The default communication parameters for NMEA output are 4800 baud, 8 data bits, stop bit, and no parity.

Tab. H.1: NMEA-0183 Output Messages

NMEA Record	Description
GPGGA	Global positioning system fixed data
GPGLL	Geographic position- latitude/longitude
GPGSA	GNSS DOP and active satellites
GPGSV	GNSS satellites in view
GPRMC	Recommended minimum specific GNSS data
GPVTG	Course over ground and ground speed

H.1 Global Positioning System Fix Data (GGA)

Table H.2 contains the values for the following example:

$$\$GPGGA, 161229.487, 3723.2475, N, 12158.3416, W, 1, 07, 1.0, 9.0, M, , , , 0000^*18$$

Tab. H.2: GGA Data Format

Name	Example	Units	Description
Message ID	$GPGGA		GGA protocol header
UTC Time	161229.487		hhmmss.sss
Latitude	3723.2475		ddmm.mmmm
N/S Indicator	N		N=north or S=south
Longitude	12158.3416		dddmm.mmmm
E/W Indicator	W		E=east or W=west
Position Fix Indicator	1		See Table H.3

Satellites Used	07		Range 0 to 20
HDOP	1.0		Horizontal Dilution of Precision
MSL Altitude	9.0	Meters	
Units	M	Meters	
Geoid Separation		Meters	
Units	M	Meters	
Age of Diff. Corr.		second	Null fields when DGPS is not used
Diff. Ref. Station ID	0000		
Checksum	$*18$		
$< CR >< LF >$			End of message termination

Tab. H.3: Position Fix Indicator

Value	Description
0	0 Fix not available or invalid
1	GPS SPS Mode, fix valid
2	Differential GPS, SPS Mode, fix valid
3	GPS PPS Mode, fix valid

H.2 Geographic Position with Latitude/Longitude(GLL)

Table H.4 contains the values for the following example:

$$\$GPGLL, 3723.2475, N, 12158.3416, W, 161229.487, A^*2C$$

Tab. H.4: GLL Data Format

Name	Example	Units	Description
Message ID	$GPGLL		GLL protocol header
Latitude	3723.2475		ddmm.mmmm
N/S Indicator	N		N/S Indicator N N=north or S=south
Longitude	12158.3416		dddmm.mmmm
E/W Indicator	W		E=east or W=west
UTC Position	161229.487		hhmmss.sss
Status	A		A=data valid or V=data not valid
Checksum	*2C		
$< CR >< LF >$			End of message termination

H.3 GNSS DOP and Active Satellites (GSA)

Table H.5 contains the values for the following example:

$$\$GPGSA, A, 3, 07, 02, 26, 27, 09, 04, 15, , , , , , 1.8, 1.0, 1.5^*33$$

Tab. H.5: GSA Data Format

Name	Example	Units	Description
Message ID	$GPGSA		GSA protocol header
Mode 1	A		See Table H.6
Mode 2	3		See Table H.7
Satellite Used(1)	07		Sv on Channel 1
Satellite Used(1)	02		Sv on Channel 2
.....			.....
Satellite Used			Sv on Channel 20
PDOP	1.8		Position Dilution of Precision
HDOP	1.0		Horizontal Dilution of Precision
VDOP	1.5		Vertical Dilution of Precision
Checksum	*33		
$< CR >< LF >$			End of message termination

Tab. H.6: Mode 1

Value	Description
M	Manual–forced to operate in 2D or 3D mode
A	2DAutomatic–allowed to automatically switch 2D/3D

Tab. H.7: Mode 2

Value	Description
1	Fix Not Available
2	2D
3	3D

H.4 GNSS Satellites in View (GSV)

Table H.8 contains the values for the following example:

$$\$GPGSV, 2, 1, 07, 07, 79, 048, 42, 02, 51, 062, 43, 26, 36, 256, 42, 27, 27, 138, 42^*71$$

$$\$GPGSV, 2, 2, 07, 09, 23, 313, 42, 04, 19, 159, 41, 15, 12, 041, 42^*41$$

Tab. H.8: GSV Data Format

Name	Example	Units	Description
Message ID	$GPGSV		GSV protocol header
Number of Messages	2		Range 1 to 3
Message Number	1		Range 1 to 3
Satellites in View	07		Range 1 to 12
Satellite ID	07		Channel 1 (Range 1 to 32)
Elevation	79	degrees	Channel 1 (Maximum 90)
Azimuth	048	degrees	Channel 1 (True, Range 0 to 359)
SNR (C/No)	42	dBHz	Range 0 to 99, null when not tracking
.....	.....		
Satellite ID	27		Channel 4 (Range 1 to 32)
Elevation	27	degrees	Channel 4 (Maximum 90)
Azimuth	138	degrees	Channel 4 (True, Range 0 to 359)
SNR (C/No)	42	dBHz	Range 0 to 99, null when not tracking
Checksum	*71		
$< CR >< LF >$			End of message termination

NOTE: Items $< 4 >, < 5 >, < 6 >$ *and* $< 7 >$ repeat for each satellite in view to a maximum of four (4) satellites per sentence. Additional satellites in view information must be sent in subsequent sentences. These fields will be null if unused.

H.5 Recommended Minimum Specific GNSS Data (RMC)

Table H.9 contains the values for the following example:

$$\$GPRMC, 161229.487, A, 3723.2475, N, 12158.3416, W, 0.13, 309.62, 120598, ,^{*}10$$

Tab. H.9: RMC Data Format

Name	Example	Units	Description
Message ID	$GPRMC		RMC protocol header
UTC Time	161229.487		hhmmss.sss
Status	A		A=data valid or V=data not valid
Latitude	3723.2475		ddmm.mmmm
N/S Indicator	N		N=north or S=south
Longitude	12158.3416		dddmm.mmmm
E/W Indicator	W		E=east or W=west
Speed Over Ground	0.13	knots	
Course Over Ground	309.62	degrees	True
Date	120598		ddmmyy

Magnetic Variation(1)		degrees	E=east or W=west
Checksum	*10		
$< CR >< LF >$			End of message termination

H.6 Course Over Ground and Ground Speed (VTG)

Table H.10 contains the values for the following example:

$$\$GPVTG, 309.62, T, , M, 0.13, N, 0.2, K^*6E$$

Tab. H.10: VTG Data Format

Name	Example	Units	Description
Message ID	$GPVTG		VTG protocol header
Course	309.62	degrees	Measured heading
Reference	T		True
Course		degrees	Measured heading
Reference	M		Magnetic(1)
Speed	0.13	knots	Measured horizontal speed
Units	N		Knots
Speed	0.2	km/hr	Measured horizontal speed
Units	K		Kilometers per hour
Checksum	*6E		
$< CR >< LF >$			End of message termination

H.7 ZDASiRF Timing Message

Outputs the time associated with the current 1 PPS pulse. Each message will be output within a few hundred ms after the 1 PPS pulse is output and will tell the time of the pulse that just occurred.
Table H.11 contains the values for the following example:

$$\$GPZDA, 181813, 14, 10, 2003, 00, 00 * 4F$$

Tab. H.11: ZDA Data Format

Name	Example	Units	Description
Message ID	$GPZDA		ZDA protocol header
UTC Time	181813		Either using valid IONO/UTC or estimated from default leap seconds
Day	14		01 TO 31
Month	10		01 TO 12

Year	2003		1980 to 2079
Local zone hour	00	knots	Offset from UTC (set to 00)
Local zone hour	00		Offset from UTC (set to 00)
Checksum	4F		
$< CR >< LF >$			End of message termination