

**A DEEP LEARNING BASED REAL-TIME
OBJECT DETECTION IMPELEMENTATION IN
A VIRTUAL INSTRUMENT CLUSTER**

**A THESIS SUBMITTED TO
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
KOCAELİ UNIVERSITY**

**BY
ABDELRAHMAN MAGDY IBRAHIM FAWZY**

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRONICS AND TELECOMMUNICATION ENGINEERING**

KOCAELİ 2019

A DEEP LEARNING BASED REAL-TIME
OBJECT DETECTION IMPLEMENTATION IN
A VIRTUAL INSTRUMENT CLUSTER




A THESIS SUBMITTED TO
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
KOCAELI UNIVERSITY

BY

ABDELRAHMAN MAGDY IBRAHIM FAWZY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRONICS AND TELECOMMUNICATION ENGINEERING

Prof. Dr. Oğuzhan URHAN
Supervisor, Kocaeli University
Assist. Prof. Dr. Orhan AKBULUT
Jury member, Kocaeli University
Assist. Prof. Dr. Ramazan DUVAR
Jury member, Doğuş University


.....

.....

.....

Thesis Defense Date: 29.07.2019

ACKNOWLEDGEMENT

I would like to express my profound gratitude to all the people helped me during my work on this thesis.

First, I would like to express my very great appreciation to my research supervisor Prof. Dr. Oğuzhan Urhan, for his valuable and constructive suggestions during the planning and development of this research work. His willingness to give his time so generously and being very patient has been very much appreciated.

I would also like to thank Res. Asst. Ayhan Küçükmanisa, for his advices and assistance in practical parts of this work.

My grateful thanks are also extended to Mathematician Büşra Gürbüz, for her help in doing the accuracy tests and analysis of some data read from the car. And to my brother Engineer Mohamed Fawzy, who helped me in some programming staff.

I would also like to extend my thanks to my managers and colleagues at work for keeping helping and being patient with me.

Not forgetting the Yurtdışı Türkler ve Akraba Topluluklar Başkanlığı (YTB) and the Kocaeli University administration for giving me this great opportunity to make my masters in Kocaeli University in Turkey.

Finally, I wish to thank my parents for their support and encouragement throughout all of my study, and being the first and last reason for me being able to reach this stage.

July- 2019

Abdelrahman FAWZY

CONTENTS

| | |
|---------------------------------------------------------------------------|-----|
| ACKNOWLEDGEMENT | i |
| CONTENTS | ii |
| LIST OF FIGURES | iv |
| LIST OF TABLES | vi |
| SYMBOLS AND ABBREVIATIONS | vii |
| ÖZET..... | ix |
| ABSTRACT | x |
| INTRODUCTION | 1 |
| 1. GENERAL INFORMATION | 3 |
| 1.1. Vehicle Safety Technologies (VST)..... | 3 |
| 1.2. Advanced Driver-Assistance Systems (ADAS) | 3 |
| 1.3. Virtual Instrument Clusters | 10 |
| 1.4. Proposed System | 13 |
| 2. BACKGROUND | 15 |
| 2.1. Control Area Network (CAN)..... | 15 |
| 2.1.1. CAN architecture | 16 |
| 2.1.2. CAN node components..... | 18 |
| 2.1.3. BUS arbitration & Messaging | 19 |
| 2.2. OBD-II..... | 19 |
| 2.3. Objects Classification & Detection | 20 |
| 2.3.1. Feature extraction | 22 |
| 2.3.2. Classification | 24 |
| 2.3.3. Viola-Jones object detection framework [41]..... | 25 |
| 2.3.4. Region-based Convolutional Network (R-CNN) | 26 |
| 2.3.5. Spatial Pyramid Pooling (SPP-net)..... | 27 |
| 2.3.6. Fast R-CNN | 28 |
| 2.3.7. Faster R-CNN | 28 |
| 2.3.8. You Only Look Once (YOLO)..... | 29 |
| 2.3.9. Single Shot MultiBox Detector (SSD) | 30 |
| 2.3.10. YOLOv2 (& YOLO9000) | 31 |
| 2.3.11. RetinaNet | 32 |
| 2.3.12. YOLOv3 | 33 |
| 2.3.13. Conclusion | 34 |
| 3. IMPLEMENTATION | 36 |
| 3.1. Tools Used in this Work..... | 36 |
| 3.1.1. NVIDIA Jetson TX2..... | 36 |
| 3.1.2. CAN-BUS PHY (Transceiver) | 37 |
| 3.1.3. Other components | 38 |
| 3.2. Stages Approach to the Final System..... | 39 |
| 3.2.1. Reading data from car's CAN-BUS | 39 |
| 3.2.2. Analyzing & understanding data | 39 |
| 3.2.3. Creating virtual gauges design & controlling it in Real-Time | 41 |
| 3.2.4. Integrating camera Real-Time stream | 41 |
| 3.2.5. Installing & training YOLO..... | 42 |

| | |
|--------------------------------------------------------------------------------|----|
| 3.2.6. Compiling YOLO with the virtual gauges integrated in a one system | 42 |
| 4. TESTING AND EVALUATING | 43 |
| 4.1. Confusion matrix | 43 |
| 4.2. Accuracy metrics | 44 |
| 4.3. Experimental Results..... | 44 |
| 5. CONCLUSION AND FUTURE WORKS | 48 |
| REFERENCES..... | 49 |
| PUBLICATION AND WORKS | 53 |
| RESUME | 54 |



LIST OF FIGURES

| | | |
|--------------|-----------------------------------------------------------------------------------|----|
| Figure 1.1. | Lane Keeping Assist System | 5 |
| Figure 1.2. | Blind Spot Monitoring System in the i-ACTIVSENSE Mazda Technology | 5 |
| Figure 1.3. | Adaptive Cruise Control System in Ford Co-Pilot 360 Technology | 6 |
| Figure 1.5. | Night Vision Enhancement System | 9 |
| Figure 1.6. | Speed Limit Violation Warning System | 10 |
| Figure 1.7. | Analog Instrument Cluster in the 1971 Chevrolet Monte Carlo | 11 |
| Figure 1.8. | Digital Instrument Cluster from the 1984 Dodge 600 | 11 |
| Figure 1.9. | “CHECK FILLER CAP” Warning on BMW E39 Character Display | 12 |
| Figure 1.10. | Virtual Instrument Cluster in Tesla Model S..... | 12 |
| Figure 1.11. | Head-UP Screen by BMW | 13 |
| Figure 1.12. | Block diagram of the system components | 14 |
| Figure 2.1. | Point to Point vs CAN protocol wiring | 16 |
| Figure 2.2. | Layers of OSI model used by CAN protocol..... | 16 |
| Figure 2.3. | CAN Protocol Physical Wires | 17 |
| Figure 2.4. | CAN Protocol Node Components | 18 |
| Figure 2.5. | Standard CAN 2.0 A Message Frame Format | 19 |
| Figure 2.6. | Extended CAN 2.0 B Message Frame Format | 19 |
| Figure 2.7. | Standard OBD-II Connector | 20 |
| Figure 2.8. | Classification of image vs Object Detection | 21 |
| Figure 2.9. | Haar Features | 22 |
| Figure 2.11. | Viola-Jones Object Detection Algorithm | 25 |
| Figure 2.12. | R-CNN Regions with CNN features..... | 26 |
| Figure 2.13. | A network structure with Spatial Pyramid Pooling Layer..... | 27 |
| Figure 2.14. | Comparison between R-CNN, SPP-net and Fast R-CNN | 28 |
| Figure 2.15. | Comparison between R-CNN, SPP-net, Fast R-CNN & Faster R-CNN | 29 |
| Figure 2.16. | YOLO Algorithm Detection Example..... | 30 |
| Figure 2.17. | SSD Algorithm Model Architecture | 31 |
| Figure 2.19. | YOLO vs RetinaNet vs SSD performance on COCO 50 Benchmark | 34 |
| Figure 3.1. | CAN Transceiver Circuit Schematic Design..... | 37 |
| Figure 3.2. | CAN Transceiver PCB | 37 |
| Figure 3.3. | OBD-II to DB9 Connector Pins..... | 38 |
| Figure 3.4. | Jetson TX2 J26 Pinouts | 39 |
| Figure 3.5. | SC while Monitoring Data of all Received IDs from Toyota Corolla CAN-Bus | 40 |

| | | |
|-------------|-----------------------------------------------------------------------------------|----|
| Figure 3.6. | SC while Monitoring Data Bytes of Breaks ID 0x224 From Corolla CAN-Bus | 40 |
| Figure 3.7. | SC while Logging & Resending Data of Speed ID 0x38D through Peugeot CAN-Bus | 41 |
| Figure 3.8. | A Screenshot of the Final System Working in Real-Time | 42 |
| Figure 4.1. | TP Example | 45 |
| Figure 4.2. | FP Example..... | 45 |
| Figure 4.3. | TN Example..... | 46 |
| Figure 4.4. | FN Example..... | 46 |



LIST OF TABLES

| | | |
|------------|----------------------------------------------------------------------------|----|
| Table 2.1. | RetinaNet outperforms YOLO at COCO 75 Benchmark | 35 |
| Table 3.1. | Jetson TX2 Development Board Tech Specs..... | 36 |
| Table 4.1. | Confusion Matrix | 43 |
| Table 4.2. | Detailed system testing results using YOLOv3 and Tiny YOLO models | 47 |
| Table 4.3. | Final system testing results using YOLOv3 and Tiny YOLO models | 47 |



SYMBOLS AND ABBREVIATIONS

| | |
|-------|---------------------------------------------|
| ACC | : Adaptive Cruise Control |
| ADAS | : Advanced Driver Assistance System |
| ALDL | : Assembly Line Diagnostic Link |
| APGS | : Advanced Parking Guidance System |
| ARM | : Advanced RISC Machines |
| CAN | : Control Area Network |
| CNN | : Convolutional Neural Network |
| CPU | : Central Processing Unit |
| CRC | : Cyclic Redundancy Code |
| DTC | : Diagnostic Trouble Code |
| ECU | : Electronic Control Unit |
| ESC | : Electronic Stability Control |
| FN | : False Negative |
| FP | : False Positive |
| GIC | : Graphical Instrument Cluster |
| GM | : General Motors |
| GND | : Ground |
| GPIO | : General Purpose Input Output |
| GPU | : Graphical Processing Unit |
| HDMI | : High-Definition Multimedia Interface |
| HOG | : Histogram of Oriented Gradient |
| IC | : Integrated Circuit |
| ID | : Identifier |
| IDE | : Integrated Development Environment |
| IPAS | : Intelligent Parking Assistant System |
| L4T | : Linux for Tegra |
| LCA | : Lane Change Assistance |
| LCD | : Liquid Crystal Display |
| LDW | : Lane Departure Warning |
| LIDAR | : Light Detection and Ranging |
| LKS | : Lane Keeping System |
| mAP | : Mean Average of Precession |
| Max | : Maximum |
| Mbps | : Megabits per second |
| MCU | : Microcontroller Unit |
| Min | : Minimum |
| OBD | : On-Board Diagnostics |
| OSI | : Open Systems Interconnection |
| PC | : Personal Computer |
| PCB | : Printed Circuit Board |
| PHY | : Physical |
| RADAR | : Radio Detection and Ranging |
| RAM | : Random Access Memory |
| R-CNN | : Region-based Convolutional Neural Network |

| | |
|-------|--------------------------------------------|
| R-FCN | : Region-based Fully Convolutional Network |
| RoI | : Region of Interest |
| RPM | : Revolutions Per Minute |
| Rx | : Received |
| SAE | : Society of Automotive Engineers |
| SIFT | : Scale-Invariant Feature Transform |
| SoM | : System on Module |
| SPP | : Spatial Pyramid Pooling |
| SSD | : Single Shot Detector |
| SUV | : Sport/Smart Utility Vehicle |
| TN | : True Negative |
| TP | : True Positive |
| TV | : Television |
| Tx | : Transmitted |
| US | : Unites States |
| USB | : Universal Serial Bus |
| VCC | : Voltage at the Common Collector |
| VST | : Vehicle Safety Technologies |
| WHO | : World Health Organization |
| YOLO | : You Only Look Once |

SANAL GÖSTERGE PANELİNDE DERİN ÖĞRENME TABANLI GERÇEK-ZAMANLI NESNE ALGILAMA UYGULAMASI

ÖZET

Lüks ve güvenlik otomotiv endüstrisinde iki vazgeçilmez bileşen hale gelmiş durumdadır. Lüks, araç yolcular ile ilgiliyken güvenlik sadece yolcularla ilgili değil, aynı zamanda aracın kendisi ve aracı çevreleyen nesnelere yani; insanlar, diğer araçlar veya herhangi bir yol kenarındaki nesnelere vb. ile de ilgilidir. Bu tanım genellikle yol trafik güvenliği olarak adlandırılır. Araç ve yol trafik güvenliği için kullanılabilecek birçok teknoloji vardır.

Bu çalışmada, sanal gösterge panelinde uygulanan Gelişmiş Sürücü Destek Sistemi (ADAS) oluşturmak için derin öğrenme tabanlı bir gerçek-zamanlı nesne algılama algoritması kullanılmıştır. Gerçek bir aracının CAN BUS'undan gelen veriler OBD-II portu üzerinden okunmuş ve tasarlanan gösterge panelinde gösterilmiştir. Sanal gösterge panelinin grafik arayüzü, alınan verileri gerçek-zamanlı olarak göstermek için OpenGL kullanılarak tasarlanmıştır. Arabaya monte edilmiş bir kamera tarafından görüntülenen canlı görüntüler, gerçek-zamanda nesne algılama ve sınıflandırma için nesne algılama algoritmasına iletilmiştir. Ardından, bu bilgilerin hepsi sanal gösterge panelinde tek bir entegre sistem olarak gösterilmiştir. Sistem 20 fps hızına ve %79 doğruluğuna kadar görüntü işleyebilmektedir.

Anahtar Kelimeler: ADAS, Gerçek-Zaman, Nesne Algılama, Nesne Sınıflandırma Sanal Gösterge Paneli.

A DEEP LEARNING BASED REAL-TIME OBJECT DETECTION IMPLEMENTATION IN A VIRTUAL INSTRUMENT CLUSTER

ABSTRACT

Luxury and safety have become two indispensable components in the automotive industry. Luxury is related to the passengers of the vehicle themselves. But safety is related not just to the passengers, but also to the vehicle itself and the objects surrounding the vehicle including human beings, other vehicles and any other roadside objects. This definition is generally called road traffic safety. There are many vehicle and road traffic safety technologies.

In this study, a deep learning based real-time object detection algorithm is used to create an Advanced Driver-Assistance System (ADAS) that is implemented in a virtual instrument cluster. The data coming from the CAN-BUS of a vehicle is read via OBD-II port and displayed in the cluster as well. A virtual cluster graphical interface is designed using OpenGL to show the received data in real-time. A live stream captured by a camera installed on the car is fed to the object detection algorithm for real-time object detection and classification. Next, all are shown in the virtual cluster as a one integrated system. The system is able to process by up to 20 fps and accuracy up to 79%.

Keywords: ADAS, Real-Time, Object Detection, Object Classification, Virtual Instrument Cluster.

INTRODUCTION

The US Department of Transportations' National Highway Traffic Safety Administration (NHTSA) declared in March 2014 that all vehicles less than 4500 kg (10000 pounds) will be required to contain a back camera starting from May 2018. The congress required the rule as a part of the 2007 Cameron Gulbransen Kids Transportations Safety Act that occurred after a two years old child go to the back of his father's SUV car without his father recognizing him. His father then derived the car back over him accidentally and reasoned in his death in the family's garage.

Also reports published by the WHO (World Health Organization) says that about 1 million people or more loses their life and about 50 million are being injured by road accidents per year. The real disaster is knowing that most of these people are youth or children between 10 and 19 years old.

Vehicle Safety technologies (VST) and Advanced Driver-Assistance Systems (ADAS) that are these systems developed to help the driver in his/her driving process and making vehicles smarter and safer have become a must in all new vehicles. Not for luxury or welfare as believed but first and foremost for the safety and security of drivers, passengers, passers-by and all other roadside objects.

This kind of systems uses a lot of technologies like radars, lidars, cameras, object detection and segmentations algorithms, position measuring sensors, distance measuring sensors (like ultrasonic) and more and more technologies that makes these systems aware of the threatening states warns the driver, suggest actions or may be takes decisions in critical situations. There is a wide variety of types of these systems like anti-collision systems, lane change and blind spot detection, pedestrian detection, night-vision enhancement, driver impairment monitoring and adaptive cruise control, etc.

Most of these systems uses the instrument panel or in another word the instrument cluster in front of the driver for displaying some information, messages or warnings. Drivers primarily relies on the vehicle's instrument cluster as the main source of

information. Specially in commercial, industrial and special purpose vehicles the instrument cluster is exposed to high demands, and therefore, it must be extremely resistant. The information delivered to driver must be easy structured, readable and commensurate with specific customer needs. carmakers these days tends to use digital panels other than analog panels aiming to reach these goals, cut costs, increase proficiency and satisfy the market. the accustomed panels were always made of plastic chaises which contains informatic lights and mechanical indicators controlled by stepper motors. Digital panels exchanged those indicators with graphical virtual representations shown on LCD screens controlled by microprocessors and Graphical Processing Units GPUs. These digital panels are now widely used in intermediate and low categories unlike before they were only used in high-class vehicle categories.

The "Valid" GIC for example is a panel that is similar to the old traditional analog panels, however it is built on new graphical representations. Valid for example can display a lot of information on a digital graphical LCD including; engine gauges, brake system, air system, tire pressure, fuel & trip information, OBD diagnostics information, battery voltages, camera video (IP) and more other information. Also, a lot of new systems like this are made by developers like Cypress, Continental, Visteon and others contains variety of features and technologies.

In this study an Advanced Driver-Assistance System is implemented in a virtual instrument cluster. A general information about the proposed system itself, the idea it is built on and the technologies used in achieving this work is discussed in the next two sections 1 and 2. Tools used and the stages approach to this work are then described in details in the section 3. Drives us to the testing methodologies that is discussed in section 4 beside showing the evaluated results according to more than one accuracy metric. Finally, inferences and future works related to this study are provided.

1. GENERAL INFORMATION

This section provides a brief over view about the thesis and related works.

1.1. Vehicle Safety Technologies (VST)

After the super-fat of the culture of acquisition of personal cars, the rate of car manufacturing increased by a huge way. Increasing with it not only comfort but also road risks and accidents.

So, Vehicle Safety Technologies (VST) in the automotive industry are those safety technologies and systems developed specially to safeguard passengers and passers-by lives beside insuring the general automobiles and road safety. Car-to-car and vehicle-to-vehicle communication devices, GPS tracking features and Electronic stability control (ESC) systems are examples of vehicle safety technologies.

1.2. Advanced Driver-Assistance Systems (ADAS)

Human mistakes are mainly the reason for almost all road accidents. So, Advanced driver-assistance systems are that type of vehicle safety technologies developed to aid vehicle's driver while he/she is driving aiming to decrease these human errors.

Advanced driver-assistance systems automate and enhance vehicle systems to make it more intelligent and increase by that car and road safety.

The GM vibrating seat warning system is a great example of an ADAS system. The Cadillac's vibrating seat starting from the Cadillac ATS 2013 warns the car driver if it began to drift out of the road lanes. So, if the driver starts to drift right or left from the road lanes the seat vibrates on the same side the vehicle is drifting towards, warning the driver of a danger, and may adjust the steering wheel by its self in some vehicles' ADAS systems like that in the 2019 Toyota Corolla. Also, the Cadillac's vibrating seat vibrates on both sides of the seat if the vehicle recognized a danger in front of the car. A lot of ADAS systems are being developed now days, and some of them are enforced by governments on manufacturers that to ensure secure roads.

Generally, systems inside a vehicle can be grouped into three categories:

- Information systems provides information about other vehicle subsystems like status of engine temperature, status of handbrakes, information coming from various vehicle's sensors like that coming from car parking sensor system etc., providing general information like the current time, the outside temperature, etc. and also provides information that are not related to the vehicle itself like navigation systems and traffic information receivers.
- Entertainment systems provides visual or auditory entertainment like car radio, USB readers and DVD players.
- Safety systems aims to increase safety of driver and passengers, either by directly supporting the driver in driving task or indirectly supporting the car itself. Like ABS (directly), ESP (indirectly), Cruise control system, Airbag system, Auto-lock system and etc.

ADAS can benefit from these three types of systems and may merge or make systems from different categories interact to reach the aim of enhancing the safety of vehicle and passengers. For example, using the radio which is an entertainment device in announcing some information or warnings to the driver in time of danger.

ADAS can be categorized into five categories [43]; lateral control systems, longitudinal control systems, reversing or parking aids, vision enhancements systems, and intelligent speed adaptation.

Lateral control systems:

They are systems that monitors the area surrounding the vehicle, and takes an action to prevent a possible collision. There are three types of these systems are currently being developed:

- The Lane Keeping Systems (LKS) and Lane Departure Warning Systems (LDW) like shown in Figure 1.1., helps the vehicle to stay in the road lane, preventing sleepy or oblivious drivers from crossing the lanes of the road and hit an obstacle. The lane warning systems alert the driver when starts drifting out of the lanes; The lane keeping systems takes real actions by correcting the vehicle's route towards inside of the lane again. They generally use image analysis to detect the lines on the road. These systems are found now days in a lot of modern vehicles like in the

2019 Toyota Corolla like mentioned before. And also, in most of new models of Cadillacs, Audis and other a lot of car brands for sure.

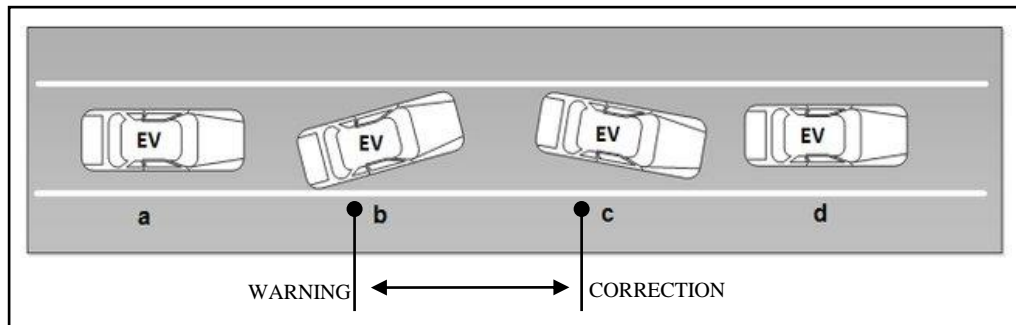


Figure 1.1. Lane Keeping Assist system [7]

- The Lane Change Assistance (LCA) and Blind Spot Monitoring Systems like that in Figure 1.2. detects the presence of an overtaking vehicles specially in the points of vision difficult for the driver to see, and alerts the driver if present. In general cameras and/or radars are used in systems like this. The Volvo S80 was the first to use this system in 2007 and then it is used in variant models of Volvo, Mazda, Ford and Mitsubishi.

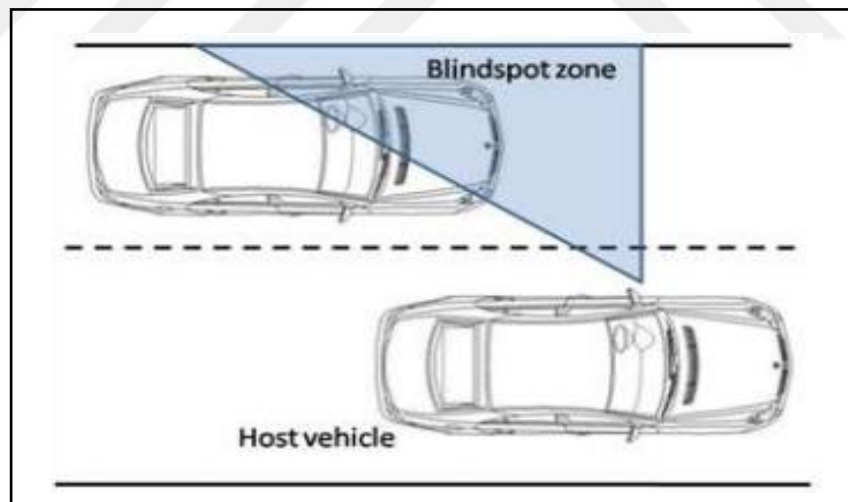


Figure 1.2. Blind Spot monitoring system in the i-ACTIVSENSE mazda technology [8]

- Side-obstacle warning systems also uses cameras and/or radar to sense the obstacles surrounding the car. Once a possible danger is recognized, these systems notify the driver of the danger that may happen. And may also take some correcting actions in emergency situations in some systems.

Longitudinal control systems:

These systems keep the situation in front and back of the vehicle under observation, and act upon the throttle and the brakes if necessary. There are five types of these systems are currently being developed:

- Adaptive Cruise Control (ACC) or Distance Keeping Systems like that in Figure 1.3. uses sensors like Radio Detection and Ranging (RADAR), Light Detection and Ranging (LIDAR), etc. to calculate the gap between the host vehicle and any obstacles ahead. Traditional cruise control systems maintain the speed set by the driver by acting upon the throttle, whereas ACCs also use the brake and other vehicle's dynamics parameters of the vehicle to decelerate if an obstacle is detected or to accelerate when traffic allows it to. Mitsubishi was the first to offer a lidar-based distance detection system in 1992 on the Japanese market called Debonair, and marketed as "Distance Warning"; The system was able to warn the driver in case of a vehicle is existing, without influencing throttle, brakes or gear shifting.

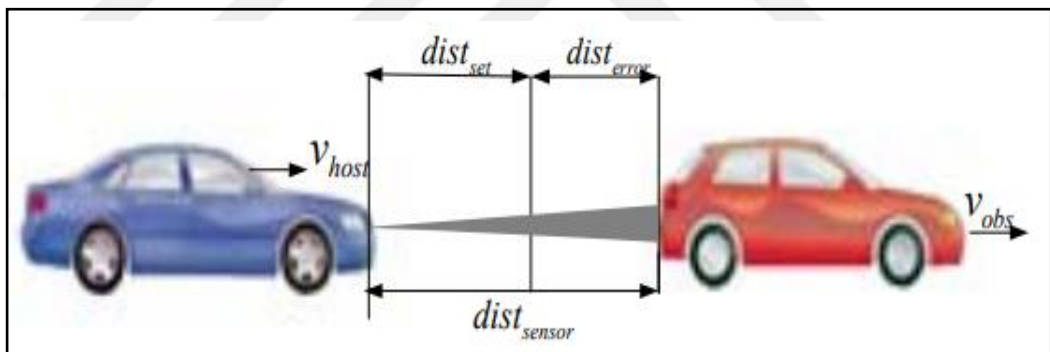


Figure 1.3. Adaptive Cruise Control System in Ford Co-Pilot 360 technology [9]

- The Forward collision warning and avoidance systems also uses cameras, radar or sometimes laser (LIDAR) to observe obstacles at the sides or in front of the vehicle. when the system senses a possibility of a crash, it warns the driver, and if there is no action taken by the driver it avoids the accident automatically by lowering the speed or/and changing the direction of the vehicle. At low speeds about 50 km/h or lower, it may use the brakes even to completely stop the car, but at high speeds it mostly steers the wheels to an appropriate position.

A lot of vehicle manufacturers started to introduce this system between years 2010 & 2014 starting from Volvo, Mercedes and Honda in 2010, VW and Ford in 2011, Audi in 2012, Skoda, Mitsubishi and FIAT in 2013 and BMW in 2014.

- Intersection collision warning systems allows vehicle to communicate with the road infrastructure to detect vehicles or pedestrians crossing an intersection and avoid collision with them. This type of systems started to appear in countries like the US where about 30% of accidents takes place at crossroads, which is a reason for more than half of the road injuries and around 10 thousand of deaths per year [18].
- ‘Stop and Go’ systems are designed for intense urban traffic. They allow the vehicle at low speed to keep track of the anterior vehicle when stopping and moving without the driver's intervene. They are built on the same premise as the ACC technology.
- Pedestrian detection and Human sensing systems alert the driver if a pedestrian or a vulnerable object enters the path of the vehicle. Different types of technologies are used in developing this kind of systems but the two main ones are laser and stereovision. Many manufacturers like Volvo, Ford, Nissan and GM started to offer these systems starting from 2017.

Reversing & Parking assistance systems:

The reversing and parking assistance systems aims to provide helping to the driver at low speed driving, for example in maneuvering operations.

- Reversing assistance systems are composed of a camera looking at the rear mounted on the back of the vehicle and a display mounted on a panel in front of the driver. They allow the driver to have a better view of what stands behind his vehicle.
- Parking assistance systems are like the Lane Change Assistance (LCA) Systems, however it is specified for low speed and short distance, like in car parking. These systems calculate the estimation gap between the vehicle’s bumpers and the obstacles near them and shows them to the driver. Generally, they use ultrasonic sensors.
- Intelligent Parking Assist System (IPAS), known as the Advanced Parking Guidance System (APGS) for Toyota models in the US, was produced by Toyota

in 1999 as an automatic parking system that helps driver in the parking process, it is introduced in the Hybrid Prius and Lexus models in the Japanese market. Vehicles containing this system can steer itself into a parking space without or with a very little driver's intervention, through an in-dash screen and some control buttons provided by the IPAS. An advanced version was revealed outside Japan for the first time in 2006 for the Lexus LS sedan.

Automatic parking systems in general are systems skillfully move a car automatically into a parking space achieving a parallel (like in Figure 1.4.), perpendicular, or angle parking. It is very useful specially in crowded and difficult places where drivers may have difficulty or take time in parking in a proper way, increasing safety and comfort. These systems calculate the coordinates of the vehicle and wheels' angle and achieves the parking process by steering the wheels in the proper angles in a proper speed.

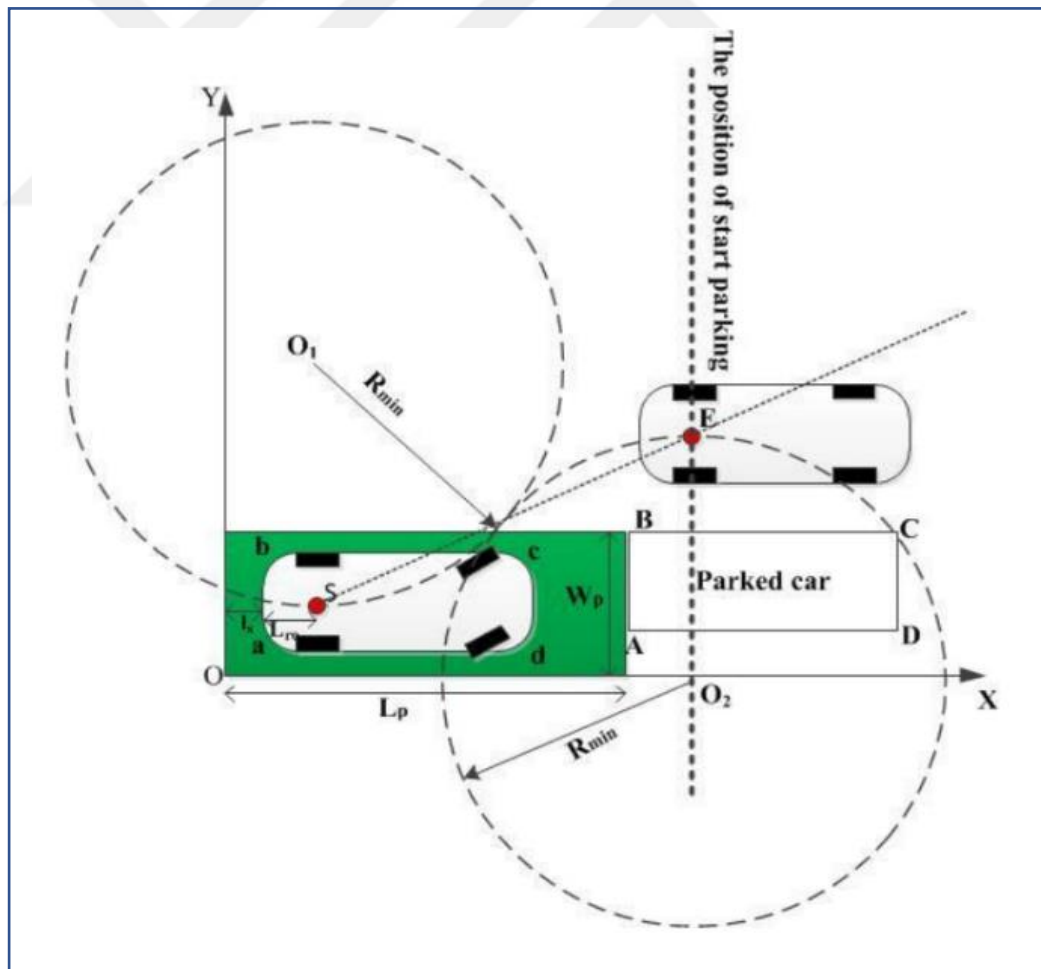


Figure 1.4. Path planning of parallel parking in Self-Driving cars [10]

Vision enhancement systems:

Light and weather conditions can badly impair the ability of the driver to detect potential hazards on the road. Automotive manufacturers and researchers started to develop Night Vision Systems, generally based on infrared images aims to aid the vehicle's driver during driving in a bad weather or night time. There are two techniques commonly used in such systems;

The first technique is based on near infrared images and requires illuminating the objects on the road with an infrared light beam; after processing, the resulting image shows enlightened objects.

The second technique is based on the thermal map of the environment provided by far infrared video images. No light source is required. Pedestrians, animals, and running vehicle are hotter than the normal environment and therefore are more visible in the image.

Both techniques can be combined, and the resulting image can be presented to the driver on a frontal screen or a head-up display like in Figure 1.5.



Figure 1.5. Night vision enhancement system [11]

Intelligent speed adaptation systems:

These systems like shown in figure 1.6. aims to maintain the speed of the vehicle below the authorized limitations. It depends on a navigation system and/or a communication system to

provide the local speed limit. This limit is notified to the driver through the frontal monitor, audible alert or acting with the brakes pedal when it is reached.



Figure 1.6. Speed limit violation warning system [12]

1.3. Virtual Instrument Clusters

In automotive industry an Instrument Cluster or Panel is a set of instrumentation dashboard displays values of gauges like speedometer, RPM, fuel gauge, etc. and indicators such as gearshift position, seat belt warning light, handbrake light, engine errors and other many warning and information indicators and lights.

At the invention of vehicles instrument clusters where all of analog indicators. These indicators started to be changed from its analog form to digital electronic indicators step by step until a fully digital clusters are being manufactured today. The first electronic instrument cluster was made in the Aston Martin Lagonda in 1976, then it is started to be included in some models of vehicles like the Cadillac Seville 1978 and other luxury vehicles. Instrument clusters can be divided into three types; Analog, Digital (or Virtual) and Heads UP.

Analog (Conventional) instrument clusters:

These clusters are also called the traditional or conventional instrument clusters, and an example of them is shown in the Figure 1.7. They use analog indicators and needles

for displaying values. They may contain speedometer, RPM (Tachometer), fuel gauge, temperature meter and some other important indicators like engine warning and low fuel level indicators.

Speedometer and RPM are connected direct to the gearbox and starts rotation at specific speed. The level of fuel is received from a floating device in the fuel tank as a voltage level. Also, engine temperature is measured by a thermostat providing analog measurements.



Figure 1.7. Analog instrument cluster in the 1971 Chevrolet Monte Carlo [13]

Digital (Virtual) instrument clusters:



Figure 1.8. Digital instrument cluster from the 1984 Dodge 600 [14]

Digital instrument clusters passed through different stages. The first stage like in Figure 1.8. was a combination between analog and digital approaches where the tachometer is replaced with LED bulbs, the speedometer is replaced with a seven-segment display and other indicators are used as same as the earlier analog ones.

In the second stage still, the traditional analog needles are used for speedometer, RPM, temperature and fuel level indicators beside using sensors and ECU signals to gather required data to the system, a character LCD is also added as a trip computer that can provide information such as speed of the trip and time taken etc. But the most important part in this display and the main revolution is that it was able to indicate warnings to the driver in written format, as shown in Figure 1.9.



Figure 1.9. “CHECK FILLER CAP” warning on BMW E39 character display [15]

In the third stage a full-size LCD TFT monitor is used to display values and indicators instead of the needle gauges driven by stepper motors. And this is mainly the type of systems called Virtual Instrument Clusters. An example is shown in Figure 1.10. In this type of systems, the driver may be able to choose from several designs and select the design he/she prefers.



Figure 1.10. Virtual Instrument Cluster in Tesla Model S [16]

In this system also data is gathered using ECU and other related sensors. But due to the real-time processing of the system it is able to provide more exact values. In these systems users can customize a lot of things and features according to his need like themes and designs or the time of driving for example (Wither in the morning for example or in the afternoon etc.).

Heads Up displays:

These type of displays like that in Figure 1.11. are introduced now days in luxury models. It is a fully digital system projected to the driver on side windscreen. It is widely used in navigation systems beside displaying speed for example and other important parameters, messages or warnings. It is useful in helping keeping focused in both road and map without the need to look at the onboard computer itself. And in some systems also it is introduced with voice instructions too.



Figure 1.11. Head-UP Screen by BMW [17]

1.4. Proposed System

In this study an Advanced Driver-Assistance System is implemented on the NVIDIA Jetson TX2 development kit using the YOLOv3 object detection algorithm integrated in a Virtual Instrument Cluster. A block diagram of the system is depicted in Figure 1.12. As seen from this figure a forward-looking camera capture the scene and transfer raw image frames to processing unit where a GPU included Jetson TX-2 processing

unit is employed. The processing unit is connected to vehicle CAN-BUS network as well in order to read some signal such as vehicle speed, motor rpm etc. for showing them on the virtual cluster.

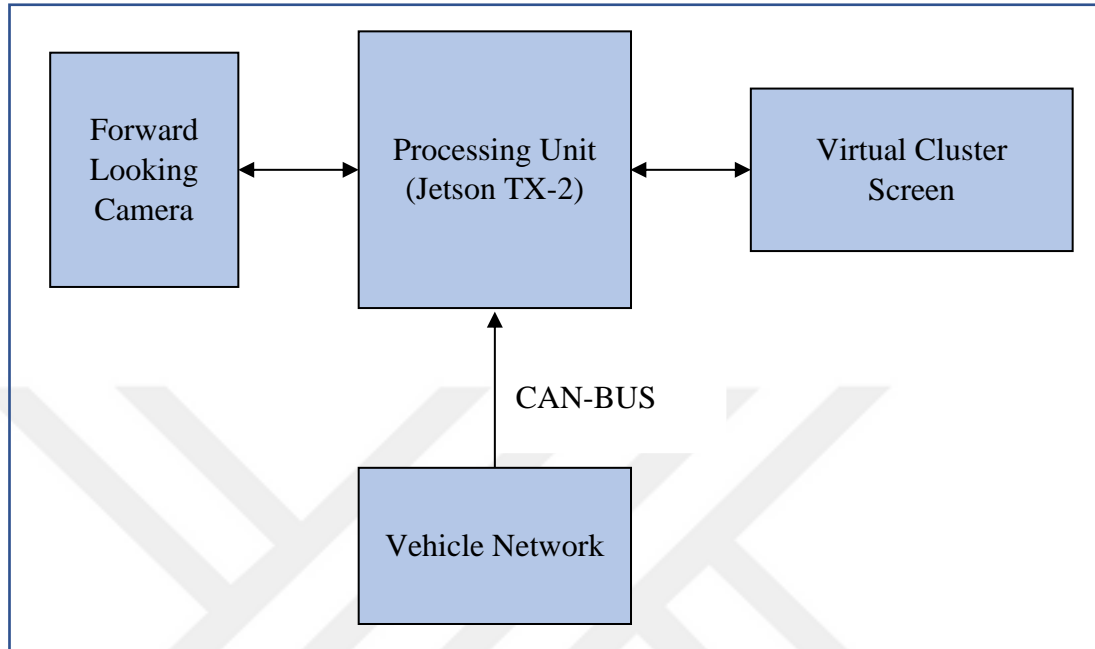


Figure 1.12. Block diagram of the system components

The object detection algorithm is used to detect objects surrounding the car, showing them to the driver on the front panel. Once correct object detection is performed; the driver can be warned or the vehicle itself may carry out some corrective actions in emergency conditions. This can help driver avoiding accidents, passing traffic lights or recognizing traffic speed limitations for certain roads and so on.

Components of the proposed system and the stages of implementation are being discussed in later sub-sections.

2. BACKGROUND

In this section some basic knowledge related to this study will be discussed.

2.1. Control Area Network (CAN)

CAN is a serial communication protocol designed especially for automobiles but it is now used also for many other fields and projects. CAN is a multi-master protocol that allows microcontrollers and devices to communicate with each other without the need to a host computer with speed up to 1Mbits/s [40]. So, it allows vehicle subsystems like engine, ABS, gear control, airbags, breaks, central locking etc. to communicate with each other inside the car without the need to a host management computer.

Robert Bosch was the first to introduce the protocol in 1983, and then it was published in 1986 officially in Detroit at the SAE (Society of Automotive Eng.) conference. Intel and Philips manufactured in 1987 the first CAN controller IC introduce to markets. And it was the Mercedes W140 that first manufactured based on the CAN bus wiring system.

Inside the vehicle there is a lot of microcontrollers or MCUs that controls a lot of devices, sensors etc., each microcontroller and each device have its own features and rules for sharing data, but on the other hand they need to communicate with each other, and so they need a common HW and SW. In the past before CAN, point to point wiring method was used to connect each device to all devices it needs to communicate with, which was acceptable in the earlier limited systems. But by the great increasing of number of devices inside the vehicle now days and the need to use more complex and huge number of signals in communication and the need to deliver information in real-time the point to point wiring became a problem. The CAN protocol was designed specially to solve such a problem. Instead of the great number of wiring connecting all the ECUs CAN protocol uses a serial bus to link them all reducing the complexity of the system. The difference between the point to point wiring and the wiring after introducing CAN protocol is illustrated in the following Figure 2.1.

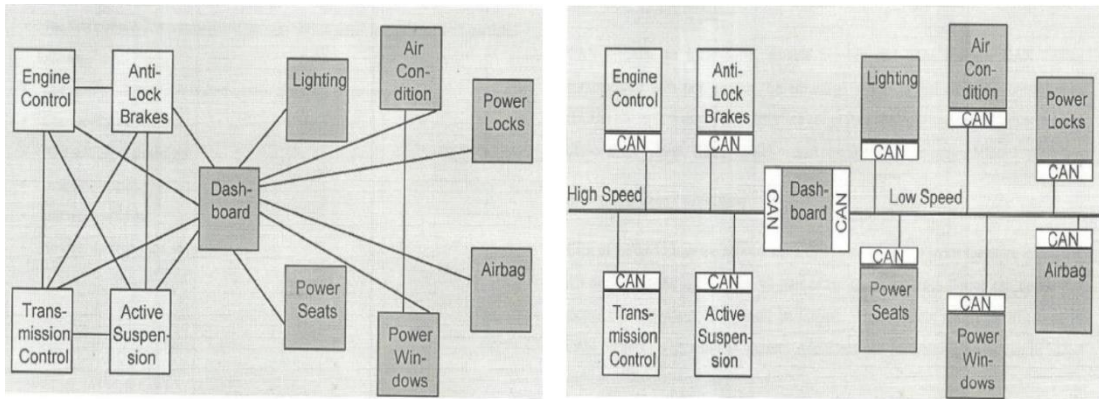


Figure 2.1. Point to Point vs CAN protocol wiring [19]

2.1.1. CAN architecture

CAN uses the known OSI model architecture illustrated in Figure 2.2. for communication. Devices in an OSI-based network can transfer data between each other through 7 layers defined by the OSI model. And they are widely used as the base architecture for many communication protocols.

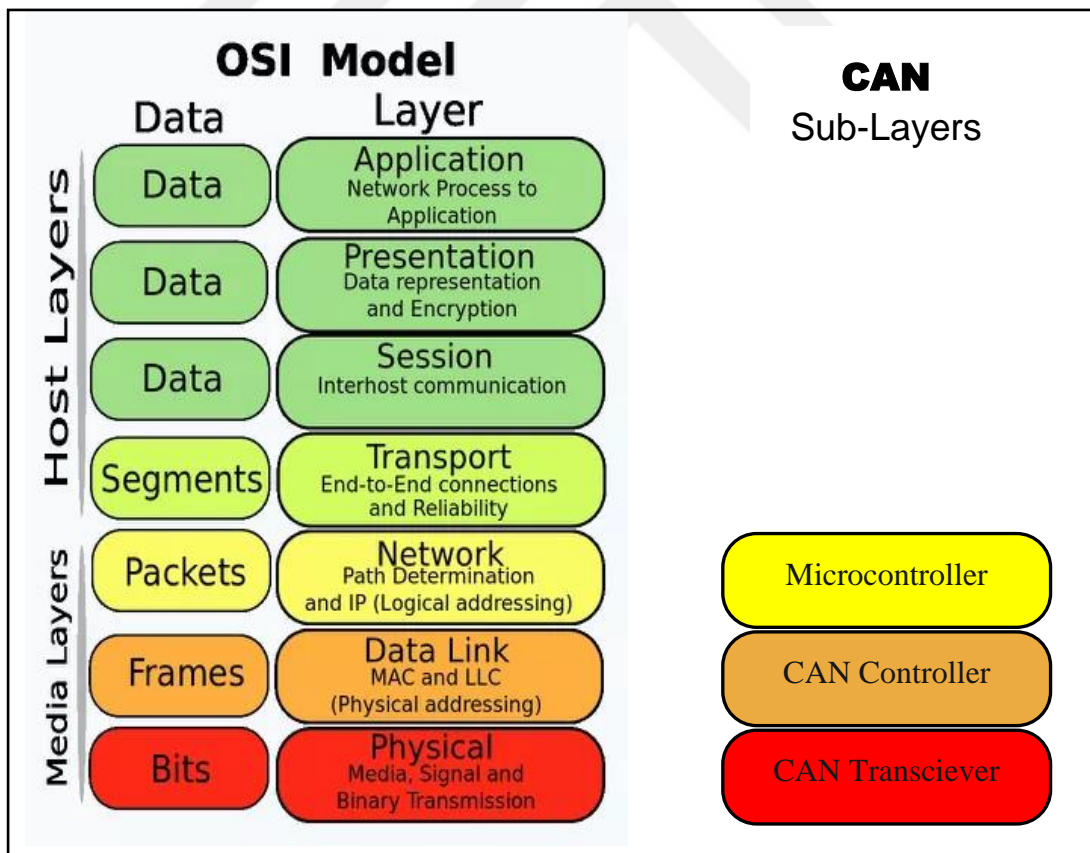


Figure 2.2. Layers of OSI model used by CAN protocol [20]

Just the bottom 2 layers of the OSI model are used by CAN; Physical & Data-link layers. The BOSCH CAN didn't define the rest of the layers in order to allow the system developers to manipulate them according to what every system need.

- Physical layer is the real hardware means of communication in the network including the cables, boards and all other physical means. In this layer type of cables, voltage levels and the regulation process of transmission of bits through between devices are defined.
- Data Link layer is the layer accountable for receiving data from physical layer bit by bit and package them in one frame. Also, in case of transmitting it is accountable for sending frames through the physical layer to other devices, and waiting for the acknowledgement from receiving devices after sending.

A device in CAN protocol network is also called a node. All nodes of a CAN-BUS network are connected through two twisted pair wires called CAN High and CAN Low. When the bus is in idle mode, both of these lines carry 2.5V, but when data bits are being transmitted, the CAN high line goes to 3.5V and the CAN low drops to 1.5V which generates a 2.5V differential voltage between the two lines; That allows CAN-BUS to resist magnetic and electrical field noises, that makes CAN very reliable in electrically noisy environments like that in vehicles. This is illustrated in Figure 2.3.

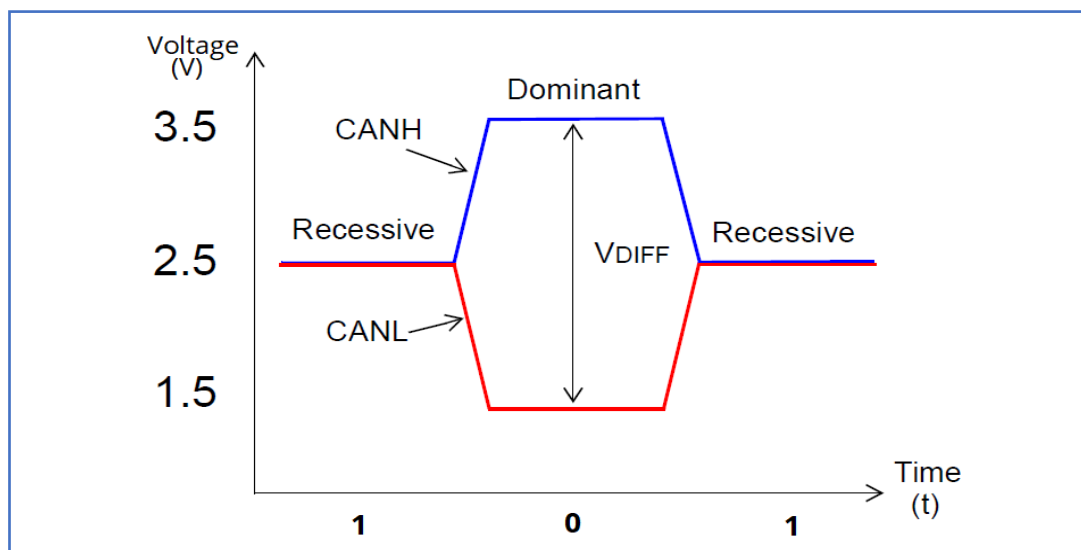


Figure 2.3. CAN Protocol Physical Wires [21]

2.1.2. CAN node components

Each node of CAN protocol consists of Microcontroller, CAN Controller and CAN transceiver like illustrated in Figure 2.4.

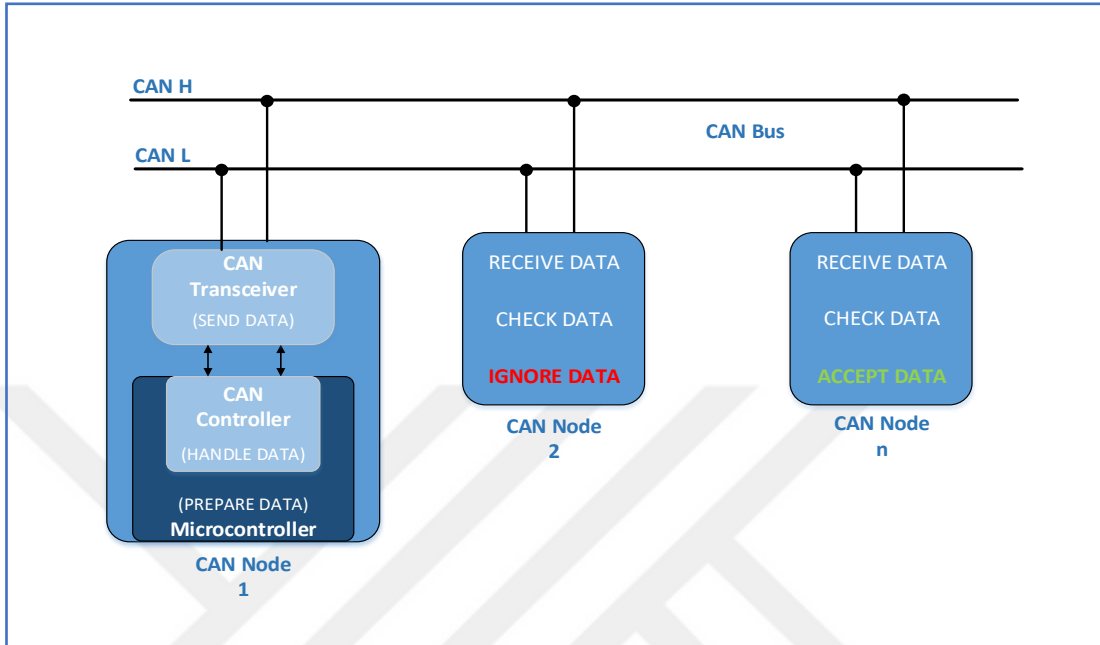


Figure 2.4. CAN Protocol Node Components

- **Microcontroller (MCU/ECU):** Decides the meaning of coming messages and what messages It wants to transmit to other nodes. Also, it controls devices, actuators and sensors connected to the node. Microcontroller may be a controller or an ECU for engine, brakes, air condition, auto lock, power seats etc. or any other vehicle's subsystem connected to the vehicle's CAN bus.
- **CAN controller:** In data receiving mode CAN controller packages serially received bits from the bus through transceiver after a full message is received completely, operates error checking algorithms (like: Cycle Redundancy Check) to determine the existing of errors due to transferring or not, and then Decides whether the node is interested in the message or not. Also, in sending mode it takes messages from microcontroller and transmits them to the bus serially when it is free through the transceiver.
- **CAN transceiver:** It is the physical layer for a CAN node which directly send the bits through CAN wires serially. And it acts as the focal point between the CAN controller and the physical wires of the CAN bus lines.

2.1.3. BUS arbitration & Messaging

CAN-BUS doesn't use addressing for identifying message receiver nodes, but every node in the network receives all messages sent from all nodes, then each node defines whether to accept or ignore it according to the interest through the message Identifier.

The identifier is the CAN message's field that defines both the priority of the message for arbitration and the data content. The message with the lower identifier is that with higher priority in the bus arbitration.

The last versions of CAN protocol are the Standard CAN 2.0A that uses 11-bit identifier and the Extended CAN 2.0B that uses 29-bit identifier. And both of them can be used to transmit data up to max 8 bytes per message. Their message data frame formats are shown in the following Figure 2.5 and Figure 2.6 respectively.

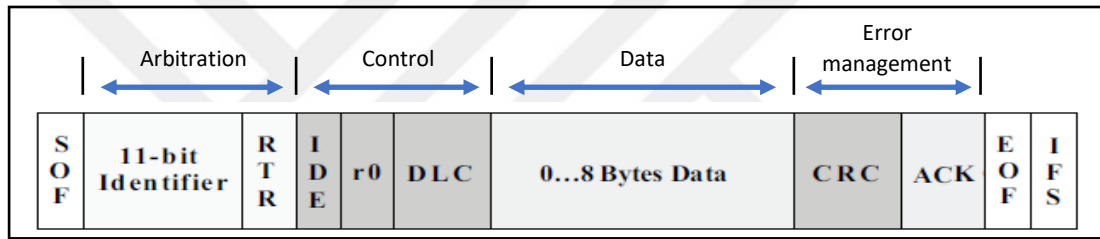


Figure 2.5. Standard CAN 2.0 A Message Frame Format [22]

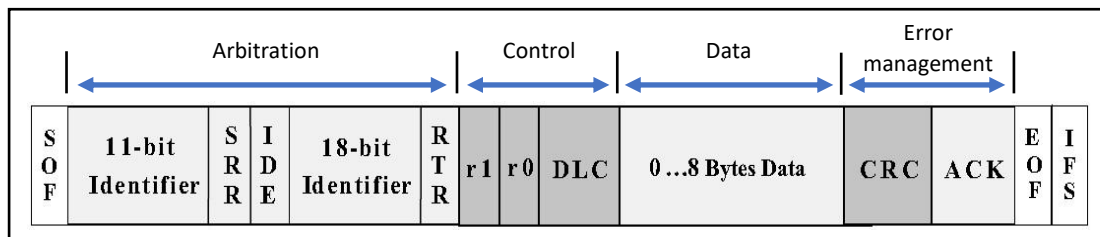


Figure 2.6. Extended CAN 2.0 B Message Frame Format [22]

2.2. OBD-II

OBD is an automotive terminology stands for On-Board Diagnostics, which refers to the ability of a vehicle to perform self-checking and reporting. OBD system allows driver & technicians to reach several subsystems of the vehicle and check out their status through data provided from the ECU.

Modern OBD implementations use communications ports allow providing various real-time info beside the common diagnostic trouble codes (DTCs). The methods for reaching this diagnostic info and the details about DTCs provided by ECUs are defined by the SAE J1979 standard.

There are several OBD standard interfaces like ALDL, M-OBD, EOBD, OBD-II and others. OBD-II is introduced in 1991 as an enhancement of OBD-I in terms of standardization and potentiality. It provides diagnostic tools with DTCs, actuator tests and sensor data. An OBD-II interface port is shown in Figure 2.7.

OBD-II interface supports five main protocols, at least one of them is found in a vehicle contains OBD-II. They are SAE J1850 PWM, SAE J1850 VPW, ISO 9141-2, ISO 14230 KWP2000 and ISO 15765 CAN.

In this work the ISO 15765 CAN is used through the OBD-II port to access the car's CAN-BUS.



Figure 2.7. Standard OBD-II Connector [23]

2.3. Objects Classification & Detection

Classifying whether an image of a cat or a dog is one problem. But Detecting them in an image and determining their locations in this image is a different problem. Image classification is the process of predicting the class of a single object in an image. But if the image contains more than one object then which one of them the classifier will predict?! Or what if the location(s) of this/these object(s) in the image is needed to be determined?! In this case classifiers can't solve the problem and another approach is needed. The process of predicting the location of an object along with the class it belongs to is called Object Detection. Both Classification and Object Detection examples are illustrated in the following Figure 2.8.

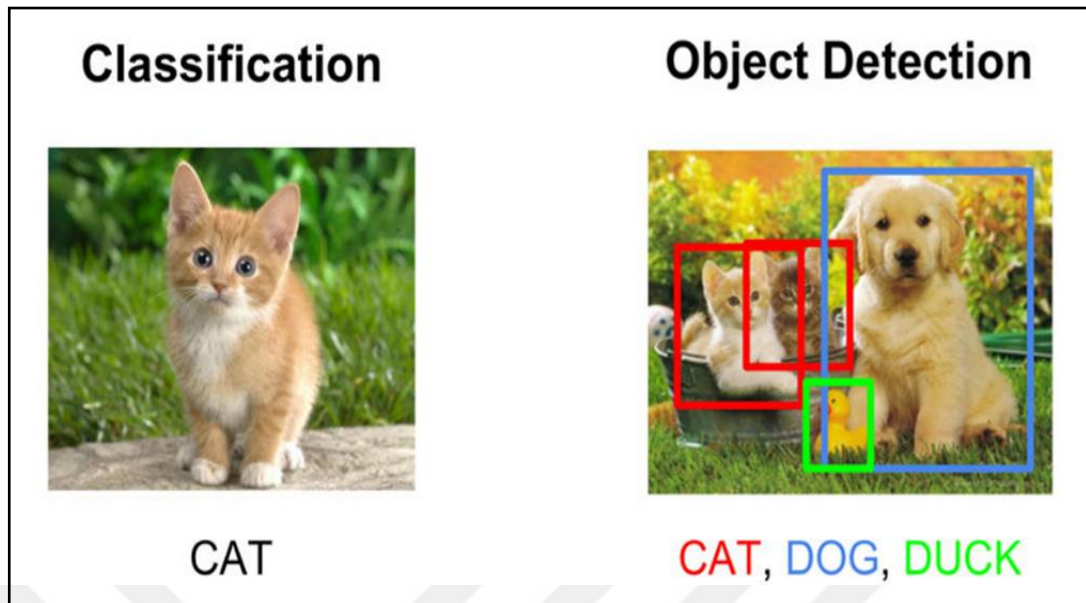


Figure 2.8. Classification of image vs Object Detection [24,25]

Classifiers made by neural networks are used to solve the first problem. But the second problem is quite different and demands different methodologies. There is a variety of algorithms that can solve the detection task, each with its own features and performances.

In this study YOLO algorithm is used for object detection purposes. In this section various methods for object detection will be introduced, discussing their features, the differences between them and the reason for choosing YOLO in this work. Also, YOLO algorithm itself will be discussed passing through its characteristics, how it works and the different versions of it.

Generally, object detection methods are classified into one of two approaches; Machine Learning-based and Deep Learning-based approaches. In Machine Learning methods, features are define first using a feature extraction method, then an algorithm like SVM (Support Vector Machine) is used to perform classification. However, Deep Learning methods are based on CNN (Convolutional Neural Networks) and they have the ability to perform detection without defining features specifically.

- Machine Learning approaches:
Conventional Machine Learning methods usually applies two stages in the object detection process; Feature Extraction and Classification. While the Deep Learning approaches processes them in one step.

2.3.1. Feature extraction

Every class defining an object has some characteristic features that help in identifying it. For instance: all squares are of four equal straight sides. Object detection approaches uses feature extraction methods to extract these special features. For example, objects that forms perpendicular corners with equal sides are considered if a square is predicted. The same thing is applied in face identification approaches. As any face contains some characteristic features defining it like nose, eyes and mouth. Also features like color of skin and distance between these face objects can be needed to detect a face.

There is a lot of feature extraction methods like Haar, SIFT, HOG, etc.

1. Haar feature selection

Like discussed above faces of all persons have the same characteristic features defines them. Haar defines these characteristics.

Some characterizations of human faces:

- The eye region is darker than the upper-cheeks.
- The nose bridge region is brighter than the eyes.

Combination of characteristics that figures face features:

- Location and size of eyes, mouth, bridge of nose
- Value: oriented gradients of pixel intensities

The commonly used four Haar features illustrated on an image of a face are shown in Figure 2.9.

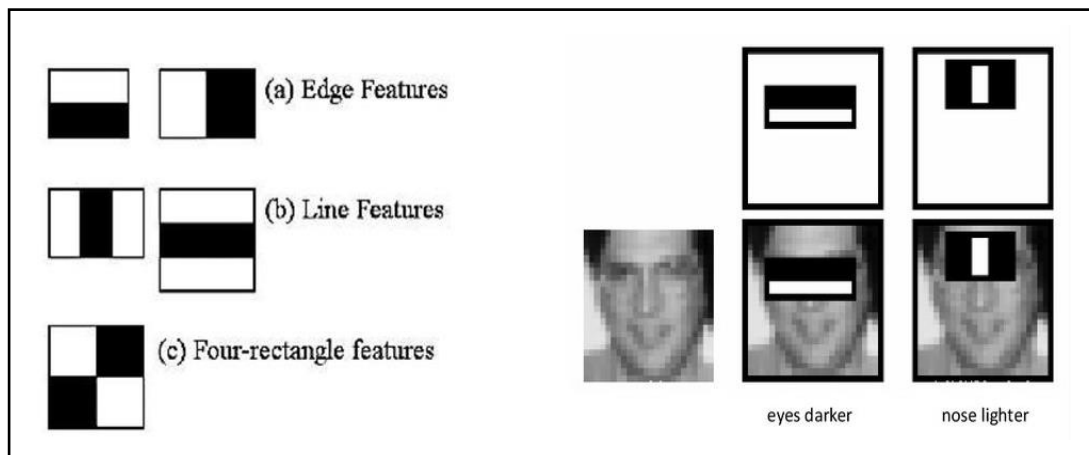


Figure 2.9. Haar Features [27,41]

2. Scale-Invariant Feature Transform (SIFT)

It is an algorithm used in detection and description of features in an image. It is patented by the University of British in Canada and published in 1999 by David Lowe. It is used in applications like object detection & recognition, robotic mapping and navigation, 3D modeling, photo stitching, identification of human gesture, etc.

SIFT extracts features from collection of reference images and stores them in a database and they are called “keypoints”. Euclidean distance of feature vectors is then used in recognizing objects in other images by comparing features in database and those in the new image to find equivalent features. Out of the whole set of matches applied on the object, subsets of keypoints and its scale, location, and orientation in the new image are specified to separate out perfect equivalents. Set of features that agree to an object are exposed to more comprehensive verification models and probability of presence of a specific object is calculated.

Disadvantage: The hurdle of SIFT is that it is very complex mathematically and computationally. SIFT is based on the Histogram of Gradients. So, it is not very efficient for low-latency devices, as the calculations of gradients of all the pixels in the patch is costly and time consuming.

3. Histogram of Oriented Gradients (HOG) features

It is a feature signifier algorithm used in object detection approaches in image processing and computer vision. The algorithm calculates number of prominences of gradient orientation in specific location of an image.

In this algorithm the pixels that are directly surrounding every single pixel are looked out, with the goal of determining how dark is this pixel compared to the neighboring pixels. Next, a pointer is drawn showing in which orientation the image is getting darker. This procedure is applied on all the pixels in the image and an oriented pointer replaces each single pixel. The created pointers are called gradients. Gradients show the influx from light to dark across the entire image. Image is then break up into small squares of 16x16 pixels. In each square, number of gradients point in each major direction is counted, and direction of all pointers in this square is replaced with the directions of the majority pointer directions. As a result, for that, the original image converted into simple representation that captures basic structure of a face.

So, in brief like illustrated in Figure 2.10. detecting faces in HOG algorithm is the process of extracting a part of an image that matches with a known HOG pattern that was extracted from a bundle of training faces.

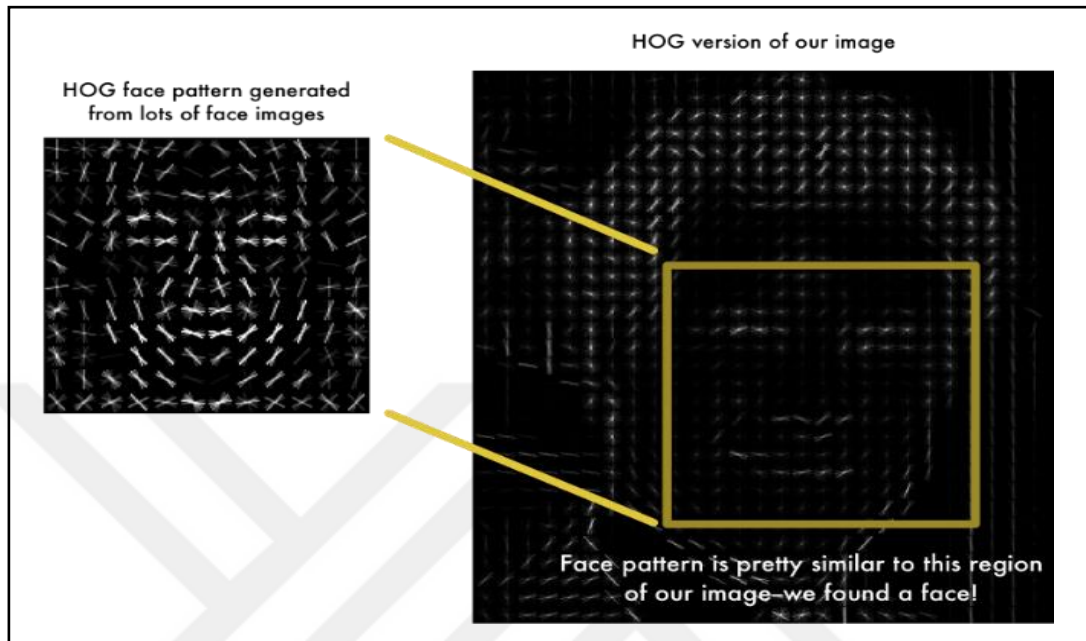


Figure 2.10. Face Detection using HOG Algorithm [29]

Disadvantage: Despite being good in many applications, it still used hand coded features which failed in a more generalized setting with much noise and distractions in the background.

2.3.2. Classification

Classifiers are algorithms used to separate amount of data or objects according to their specific group (ex. Person, Car, Animal, etc.). One of the most known classifiers is the SVM.

Support Vector Machines (SVM) is a classifier algorithm that builds hyperplanes in a multidimensional space, that separates cases of distinct class types, to carry out classification task. In SVM objects are reorganized using some kind of mathematical functions called Kernels and the process of arrangement is called mapping.

Viola-Jones framework one of the known object detection algorithm based on machine learning is briefly explained in the next page.

2.3.3. Viola-Jones object detection framework [41]

Viola-Jones is the first competent object detection algorithm to provide a real-time object detection efficient rate. It is introduced in 2001 by Paul Viola and Michael Jones. It is made specially for face detection but it can although be trained to detect a variety of other object classes.

It is a machine learning based approach used to train cascade function by a set of positive and negative images, then use this function for detecting objects in different images. A boosting algorithm called Adaboost is implemented to build strong bound on the generalization performance. In the learning process, Adaboost algorithm is used for selecting features which are capable of detecting the object of interest. The features used are based on the Haar feature selection method functions which are called Haar-like features.

The algorithm has four stages like illustrated in Figure 2.11.

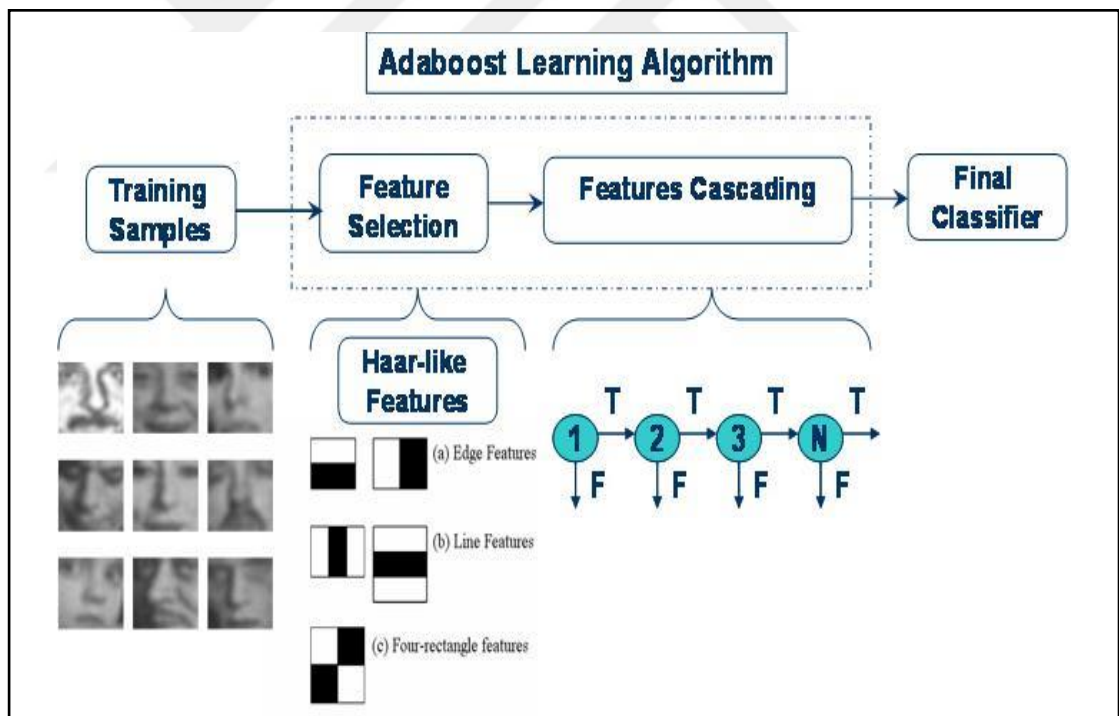


Figure 2.11. Viola-Jones Object Detection Algorithm [26]

Disadvantage: Viola-Jones requires the face being detected in an image to be clear not wearing masks or something like that. And to be straightly looking towards the camera. Not looking up, down, left or right. That to have a full-face view.

- Deep Learning approaches:

A classifier like VGGNet or Inception can be used and turned into an object detector by sliding a small window across the image. At each step the classifier runs to predict what kind of object is inside the current window. The sliding window gives several hundred or thousand predictions for that image, but only the ones the classifier is the most certain about is kept. This approach works but it's obviously going to be very slow, since the classifier is need to run many times. Also, it is computationally expensive.

For this reason and for the fact that number of occurrences of objects of interest in an image is not fixed and might have different spatial locations within the image and different aspect ratios, which compels the need to select a huge number of regions that could computationally blow up; the use of standard convolutional networks followed by a fully connected layer is not the perfect solution. And therefore, algorithms like R-CNN, R-FCN, SSD, YOLO, etc. is developed to find these occurrences and localize them faster.

2.3.4. Region-based Convolutional Network (R-CNN)

To bypass the problem of selecting a huge number of regions, Ross Girshick proposed a method uses selective search to extract just 2000 regions from the image and he called them region proposals. Therefore, now, instead of trying to classify a huge number of regions, just 2000 regions are used. These 2000 region proposals are generated using a selective search algorithm that Generates initial sub-segmentation, then uses greedy algorithm to reclusively combine similar regions into larger ones, and finally uses the generated regions to produce the final candidate region proposals. This is illustrated in the following Figure 2.12.

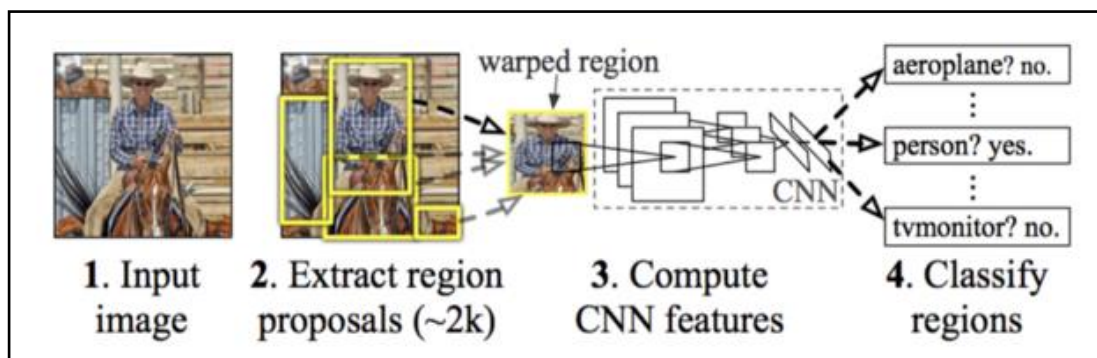


Figure 2.12. R-CNN Regions with CNN features [30]

Disadvantages:

It cannot be implemented for real-time as it still takes a huge amount of time to train the network as it would have to classify 2000 region proposals per image, and it takes around 47 seconds for each test image.

2.3.5. Spatial Pyramid Pooling (SPP-net)

Running CNN to extract only 2000 regions using selective search still very slow and takes a lot of time. So, SPP-Net tried to fix this with calculating the CNN representation for entire image only once, and use that to calculate the CNN representation for each patch generated by selective search.

Also, it used spatial pooling instead of traditionally used max-pooling after the last convolutional layer to provide a fixed size representation of input for the fully-connected layers of the CNN. This is illustrated in Figure 2.13.

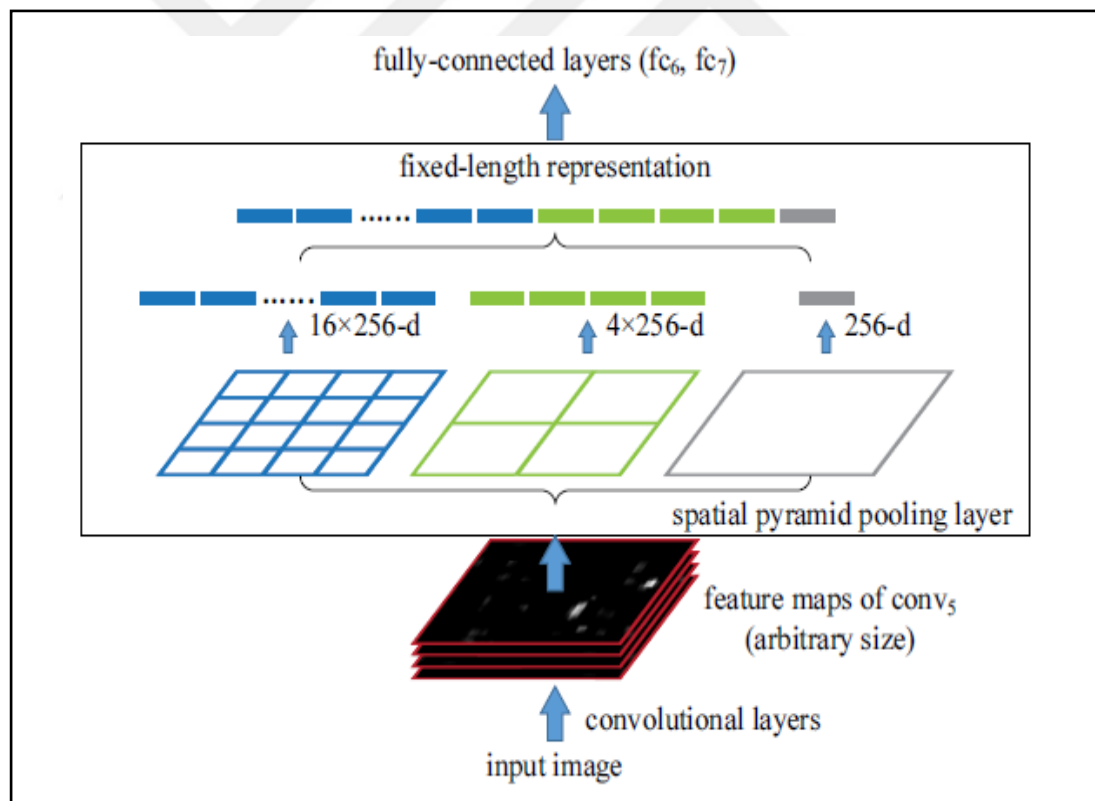


Figure 2.13. A network structure with Spatial Pyramid Pooling Layer [32]

Disadvantage: “There was one big drawback with SPP net is that it was not trivial to perform back-propagation through spatial pooling layer. Hence, the network only fine-tuned the fully connected part of the network.” [25]

2.3.6. Fast R-CNN

The same author of R-CNN developed a faster algorithm called Fast R-CNN based on the ideas of R-CNN and SPP-net by fixing some of their problems. Fast R-CNN used a back-propagation calculation like that of max-pooling, except that pooling regions overlaps. Thus, a cell can have gradients pumping in from multiple regions.

Also, in Fast R-CNN bounding box regression is added to the classification neural network to provide multitasking instead of training the network separately for both. These two fixes increased the accuracy and reduced the overall training time compared to SPP-net.

The reason “Fast R-CNN” is faster than “R-CNN” is the no need for feeding 2000 region proposals to the convolutional neural network every time. Instead, the convolution operation is done only once per image and a feature map is generated from it. A comparison between R-CNN, SPP-net and Fast R-CNN is provided in the following graph in Figure 2.14.

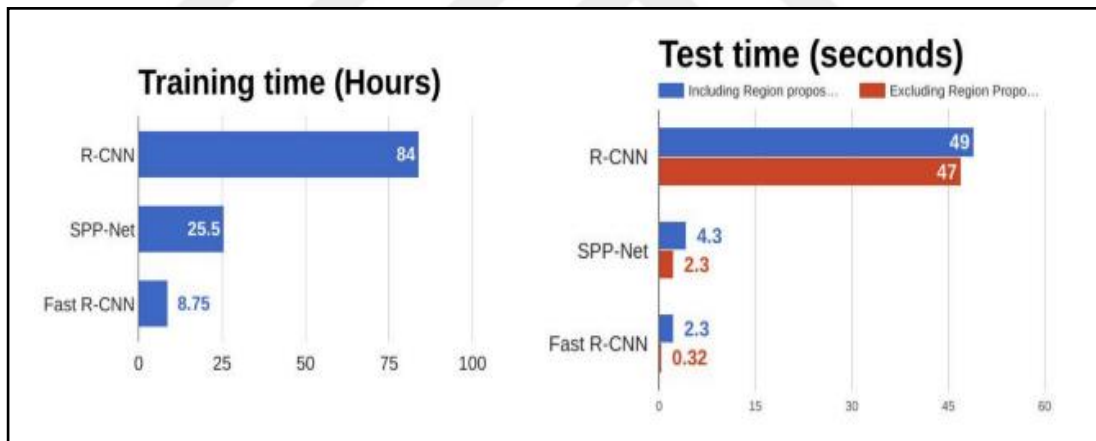


Figure 2.14. Comparison between R-CNN, SPP-net and Fast R-CNN [33]

2.3.7. Faster R-CNN

Selective search used by R-CNN and Fast R-CNN to find out region proposals is slow and time-consuming process that affects the performance of the network. Therefore, Shaoqing Ren came up with an object detection algorithm that eliminates the selective search algorithm and lets the network learn the region proposals.

“Similar to Fast R-CNN, the image is provided as an input to a convolutional network which provides a convolutional feature map. Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is

used for that. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.” [28]

In the following Figure 2.15. a comparison between test-time of R-CNN, SPP-net, Fast R-CNN and Faster R-CNN illustrates the improvements in speed.

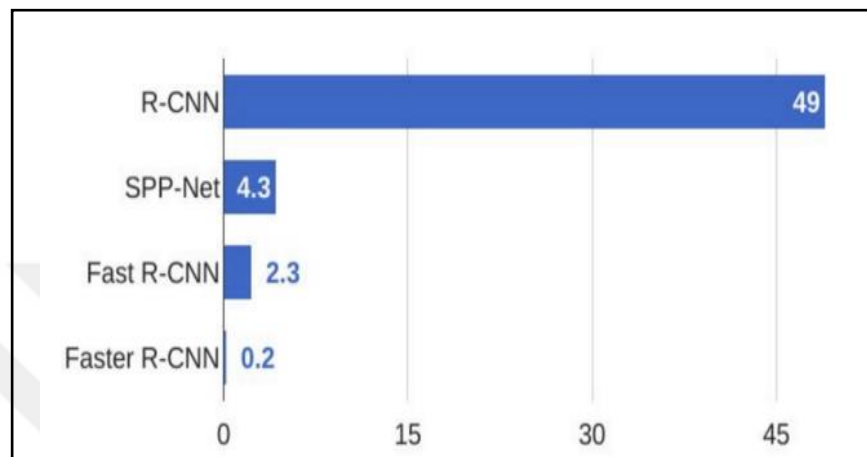


Figure 2.15. Comparison between R-CNN, SPP-net, Fast R-CNN & Faster R-CNN [33]

2.3.8. You Only Look Once (YOLO)

YOLO is a neural network introduced by J. Redmon et al. in 2016 proficient in detecting objects and their locations in an image, just in one image pass. It draws bounding boxes around the detected objects, and can detect more than one object at the same time. Other algorithms usually perform multilayer tasks on the image to be able to classify and detect objects like illustrated in the previous sections. But what makes YOLO very fast and performant is that it does detections in just a single network pass. and this is the major breakthrough yolo made.

Darknet is a neural networks training framework. And it serves as the basic framework for training YOLO. Means it sets the architecture of the network. It is open source, written in C/CUDA and supports both CPU and GPU computation.

YOLO is able to predict objects class probabilities and to draw bounding boxes in a single network in a single evaluation. The simplicity of the YOLO model allows it to perform predictions in real-time. And this is the main reason for choosing YOLO for the purposes of this study.

YOLO algorithm like illustrated in Figure 2.16. takes an image as an input and divides it into an $S \times S$ grid. Each cell of the grid predicts N bounding boxes with a confidence score. This confidence is simply the probability that the box contains an object multiplied by the IoU between the predicted and the ground truth boxes.

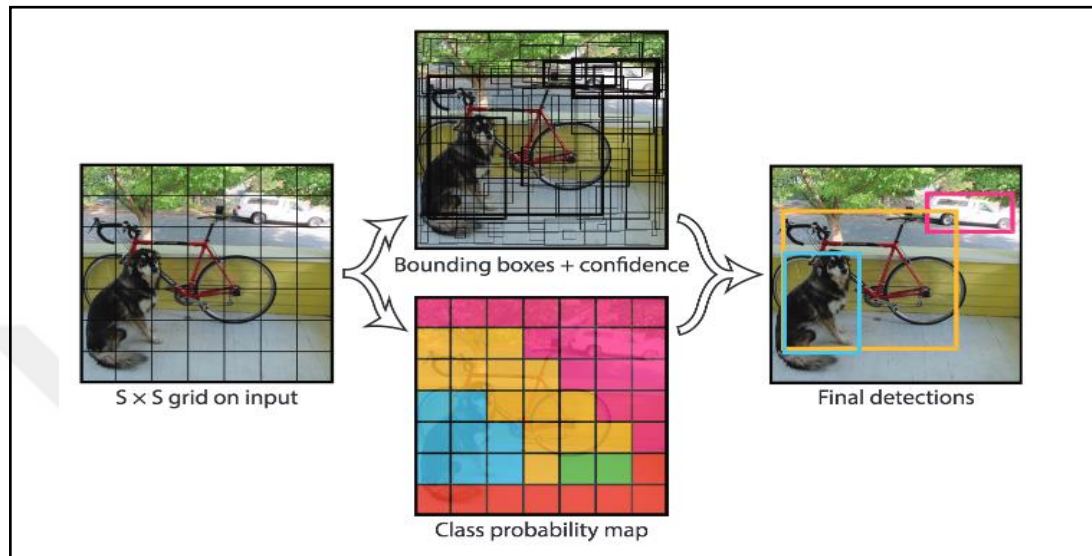


Figure 2.16. YOLO Algorithm Detection Example [34]

YOLO network consists of 24 convolutional layers followed by 2 fully-connected layers. Reduction layers with 1×1 filters followed by 3×3 convolutional layers replace the initial inception modules. The Tiny YOLO model is a lighter version with only 9 convolutional layers and fewer number of filters. Most of the convolutional layers are pretrained using the ImageNet dataset with classification.

2.3.9. Single Shot MultiBox Detector (SSD)

SSD is an algorithm developed by W. liu in 2016 with the same idea of YOLO. It is able to detect objects, calculate objects' class probabilities and their bounding boxes in an image all at the same time. SSD provided a good accuracy with a high rated speed too. Like shown in Figure 2.17. SSD uses a feature extraction model called VGG-16 pretrained on ImageNet, then adding a number of convolution feature layers of decreasing sizes making a shape of pyramid represents images at different scales. Large fine-grained feature maps at earlier levels are good at capturing small objects and small coarse-grained feature maps can detect large objects well. In SSD, the detection happens in every pyramidal layer, targeting at objects of various sizes.

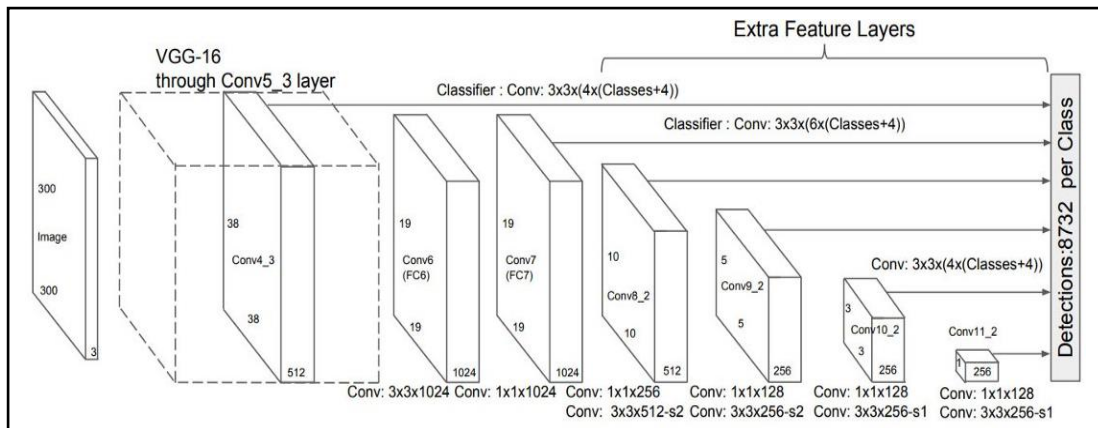


Figure 2.17. SSD Algorithm Model Architecture [35]

2.3.10. YOLOv2 (& YOLO9000)

YOLOv2 produced by Redmon & Farhadi in 2017 as an enhancement version of YOLO. YOLO9000 is built on top of YOLOv2 but trained with joint dataset combining the COCO detection dataset and the top 9000 classes from ImageNet.

YOLOv2 applied a variety of modifications to make YOLO more accurate and faster in prediction process:

- **Batch Normalization:** Added batch normalization on all the convolutional layers, that leads to more accurate and faster training speed.
- **Image resolution matters:** Fine-tuning the base model with high resolution images 448x448 with the old 224x224 improved the detection performance.
- **Convolutions with anchor boxes:** Like in faster R-CNN, YOLOv2 used anchor boxes to predict bounding boxes instead of using the fully-connected layers. Using anchor boxes resulted in a very small drop in mAP but a large increase in the recall.
- **K-mean clustering of box dimensions:** YOLOv2 used pre-defined sizes and scales of anchor boxes like in Faster R-CNN, but using standard Euclidean distance-based k-means clustering is not perfect as larger boxes gives more error than smaller ones. So YOLOv2 uses k-means clustering that leads to better IoU average.
- **Direct location prediction:** In YOLOv1 there was no restrictions on prediction locations, so predicted bounding box could be far from the original grid location.

That makes the model unstable. So, YOLOv2 constrained the location of bounding boxes in a way that it would not diverge from the centre location too much.

- Add fine-grained features: The 13x13 feature map output is suitable for detecting large objects. But to detect small objects 26x26x512 feature maps from earlier layer is mapped into 13x13x2048 feature map, then merged with the original 13x13 feature maps.
- Multi-scale training: For training the model to be durable with input images with different sizes, new image dimensions are randomly chosen for the input every 10 batches, network is resized and then continue training. The newly sampled size is a multiple of 32.

2.3.11. RetinaNet

RetinaNet is an object detection algorithm introduced in 2018 to deal with dense detection problems in background and foreground. It is based on two building blocks; Focal Loss and Feature Pyramid Network.

- Focal loss is a method designed to prevent negatives from clouding the detector by trying to solve the imbalance between background and foreground that contains the objects of interest.
- Feature Pyramid Network is the mainstay for RetinaNet. The same as the pyramid representation of images in SSD, Feature Pyramid Network provides a basic vision methodology for object detection at different scales.

The main idea of Feature Pyramid Network is illustrated in Figure 2.18. Each network stage is correspondent to a sequence of pyramid levels, and contains more than one convolutional layer of the same size scaled down by a factor of 2 every stage.

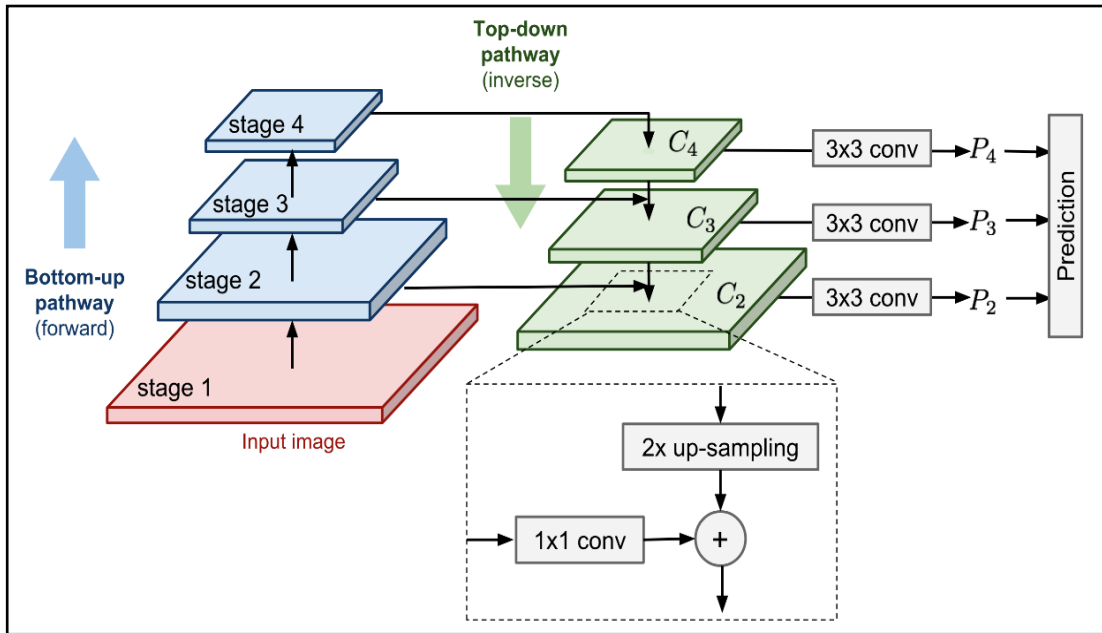


Figure 2.18. The illustration of the Feature Pyramid Network [36]

2.3.12. YOLOv3

YOLOv3 is developed by applying a set of “incremental improvements” (like the author of YOLO named it) on YOLOv2. YOLOv3 introduced some needed refinements for a real-time accurate object detection process as following:

- Logistic regression for confidence scores:
YOLO and YOLOv2 uses summation of squared errors for both localization loss for bounding box offset prediction and classification loss for conditional class probabilities. But YOLOv3 calculates confidence score for bounding boxes using logistic regression that increased mAP.
- No more SoftMax for class prediction:
Instead of the single SoftMax layer used in YOLOv2, YOLOv3 used multiple independent logistic classifiers for each class, for predicting class confidence. That is very useful specially in images contains more than one label, that even may be mutually implicated.
- Darknet & ResNet as the base model:
Instead of the Darknet-19 used for feature extraction in YOLOv2, YOLOv3 uses Darknet-53 classification network which is much deeper and contains 53 convolutional layers. It has better performance and 1.5x faster than ResNet-101,

and it has similar performance and 2x faster than ResNet-152 (like the author mentioned in the paper).

- Multi-scale prediction:

YOLOv3 added more than one convolutional layer after the base feature extractor model to make predictions at three various image scales something like Pyramid Network used in SSD and RetinaNet.

- Skip-layer concatenation:

YOLOv3 concatenates the output of previous layer with that of prediction layers (except for output layer). This process doubles the input size but it made the detecting process better specially for small objects.

2.3.13. Conclusion

YOLOv3 performs at par with well-known other detectors like RetinaNet, while being considerably faster (about 4x faster) [42]. It is also better and faster than SSD. Here is a comparison of performances shown in Figure 2.19 and Table 2.1.

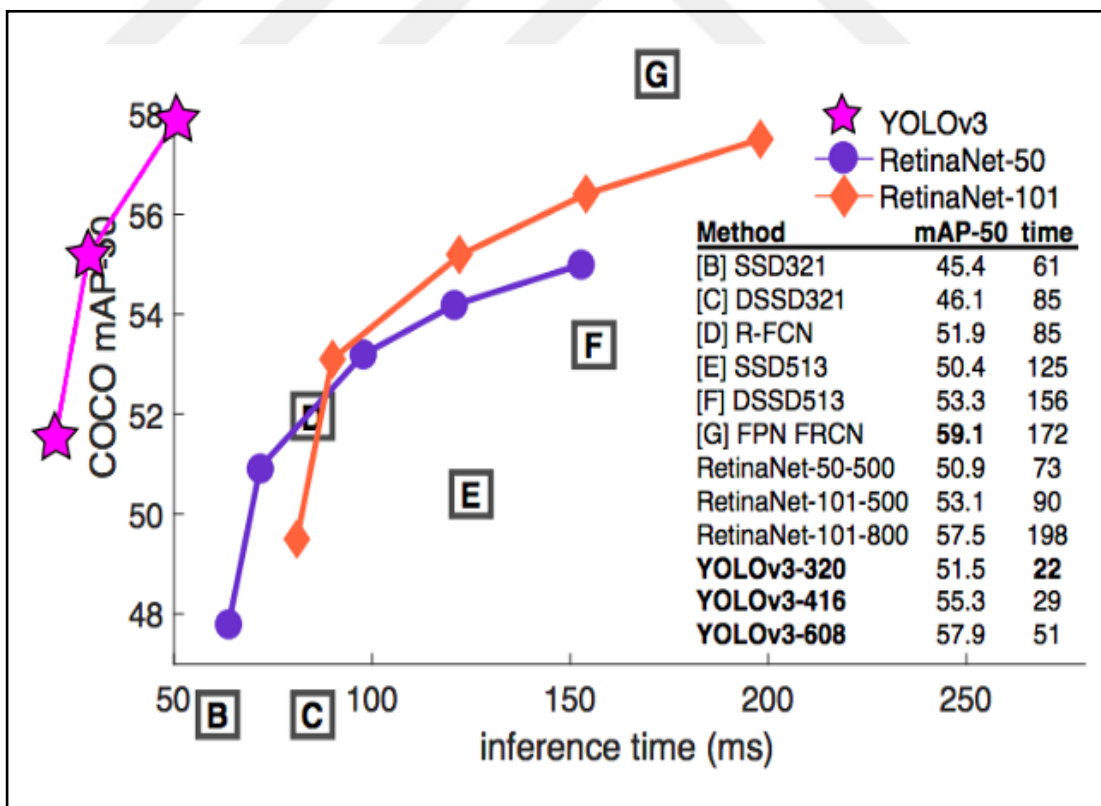


Figure 2.19. YOLO vs RetinaNet vs SSD performance on COCO 50 Benchmark [37]

Table 2.1. RetinaNet outperforms YOLO at COCO 75 Benchmark [37]

| | backbone | AP | AP ₅₀ | AP ₇₅ | AP _S | AP _M | AP _L |
|---------------------------|--------------------------|-------------|------------------|------------------|-----------------|-----------------|-----------------|
| <i>Two-stage methods</i> | | | | | | | |
| Faster R-CNN+++ [5] | ResNet-101-C4 | 34.9 | 55.7 | 37.4 | 15.6 | 38.7 | 50.9 |
| Faster R-CNN w FPN [8] | ResNet-101-FPN | 36.2 | 59.1 | 39.0 | 18.2 | 39.0 | 48.2 |
| Faster R-CNN by G-RMI [6] | Inception-ResNet-v2 [21] | 34.7 | 55.5 | 36.7 | 13.5 | 38.1 | 52.0 |
| Faster R-CNN w TDM [20] | Inception-ResNet-v2-TDM | 36.8 | 57.7 | 39.2 | 16.2 | 39.8 | 52.1 |
| <i>One-stage methods</i> | | | | | | | |
| YOLOv2 [15] | DarkNet-19 [15] | 21.6 | 44.0 | 19.2 | 5.0 | 22.4 | 35.5 |
| SSD513 [11, 3] | ResNet-101-SSD | 31.2 | 50.4 | 33.3 | 10.2 | 34.5 | 49.8 |
| DSSD513 [3] | ResNet-101-DSSD | 33.2 | 53.3 | 35.2 | 13.0 | 35.4 | 51.1 |
| RetinaNet [9] | ResNet-101-FPN | 39.1 | 59.1 | 42.3 | 21.8 | 42.7 | 50.2 |
| RetinaNet [9] | ResNeXt-101-FPN | 40.8 | 61.1 | 44.1 | 24.1 | 44.2 | 51.2 |
| YOLOv3 608 × 608 | Darknet-53 | 33.0 | 57.9 | 34.4 | 18.3 | 35.4 | 41.9 |

From these comparisons YOLO is very accurate compared to other object detections approaches. Just RetinaNet is considered more accurate. But because of the high speed of YOLO against RetinaNet, YOLO is chosen in this study. This is because the study is working on automotive field that needs both accuracy and speed together.

3. IMPLEMENTATION

In this section tools used in this work & steps of implementation will be discussed.

3.1. Tools Used in this Work


3.1.1. NVIDIA Jetson TX2

In this work the system is implemented on the Nvidia Jetson TX2 development board. TX2 is an embedded system-on-module (SoM) with dual core NVIDIA Denver 2 + quad core ARM Cortex-A57, 8GB 128-bit LPDDR4 and integrated 256-core Pascal GPU supports CUDA. It contains an HDMI output port, a USB 3.0 interface and 2 CAN-BUS interfaces. The Jetson TX2 technical specs are shown in Table 3.1.

TX2 runs Linux and it supports the NVIDIA Jetson Development Pack for Linux for Tegra (Jetpack L4T), which is an installer that automates installing and setting up the development environment required to develop for the Nvidia Jetson Embedded Platforms, including flashing the Jetson kit with the latest OS image, libraries, APIs, development tools, samples, and documentation. Thus, it is useful for deploying computer vision and deep learning.

Table 3.1. Jetson TX2 Development Board Tech Specs [38]

| Jetson TX2 | |
|--------------|----------------------------------------------------------------------|
| GPU | NVIDIA Pascal™, 256 CUDA cores |
| CPU | HMP Dual Denver 2/2 MB L2 + Quad ARM® A57/2 MB L2 |
| Video | 4K x 2K 60 Hz Encode (HEVC) 4K x 2K 60 Hz Decode (12-Bit Support) |
| Memory | 8 GB 128 bit LPDDR4 58.3 GB/s |
| Display | 2x DSI, 2x DP 1.2 / HDMI 2.0 / eDP 1.4 |
| CSI | Up to 6 Cameras (2 Lane) CSI2 D-PHY 1.2 (2.5 Gbps/Lane) |
| PCIe | Gen 2 1x4 + 1x1 OR 2x1 + 1x2 |
| Data Storage | 32 GB eMMC, SDIO, SATA |
| Other | CAN, UART, SPI, I2C, I2S, GPIOs |
| USB | USB 3.0 + USB 2.0 |
| Connectivity | 1 Gigabit Ethernet, 802.11ac WLAN, Bluetooth |
| Mechanical | 50 mm x 87 mm (400-Pin Compatible Board-to-Board Connector) |



3.1.3. Other components

- OBD-II to DB9 connector

OBD-II port of the car contains CAN High & CAN Low on the pins 6 and 14 respectively like shown in Figure 3.3.

For connecting the transceiver to the car's CAN-BUS these pins are reached by using an OBD-II to DB9 connector, that is connected from one side to the CAN Transceiver and from the other side to the OBD-II port of the car.

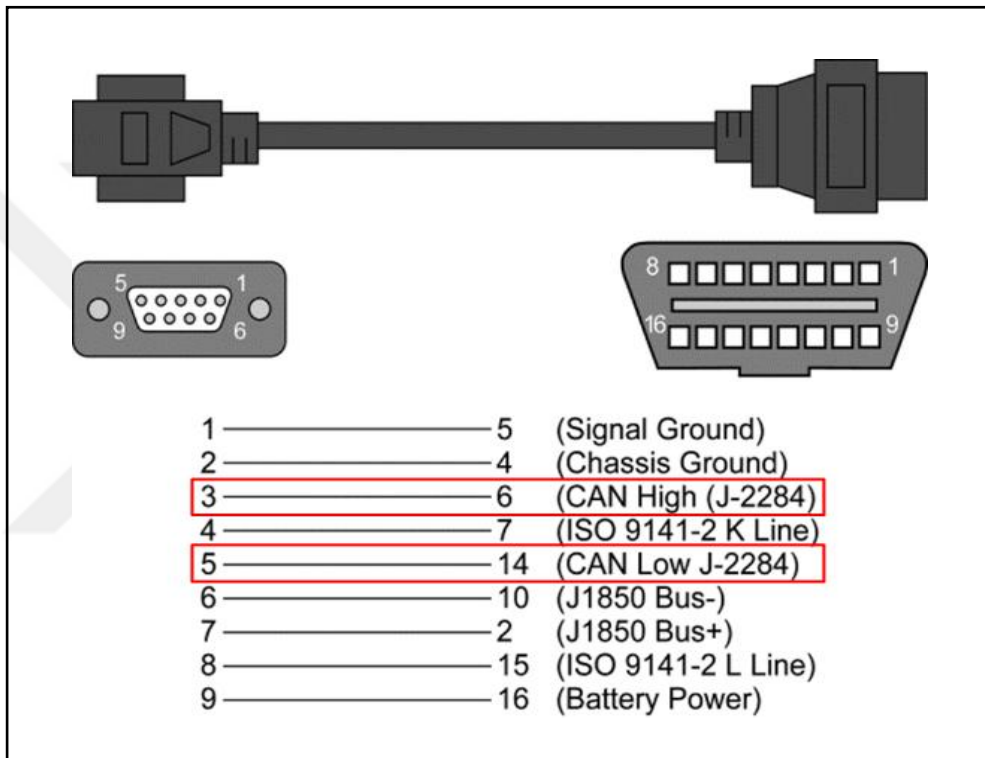


Figure 3.3. OBD-II to DB9 Connector Pins [39]

- Vehicles
System is tested on Peugeot 5008 2011 and Toyota Corolla 2014 cars contains OBD-II ports.
- Monitor
MustHD 7" 4k LCD monitor connected to the board through normal HDMI cable is used as the virtual cluster screen.
- Camera
A4Tech USB webcam with acceptable performance is used as the forward-looking camera.

3.2. Stages Approach to the Final System

The steps of creating the system can be arranged into 6 steps or stages.

3.2.1. Reading data from car's CAN-BUS

After connecting the transceiver board to the car's OBD-II port, it is connected to the Jetson TX2 using the Jetson J26 GPIO (General Purpose Input Output) Expansion Header on the board. The first CAN interface's Rx and Tx signals are reached through the J26 GPIO pins 5 & 7 respectively. Like shown in the J26 pinouts in Figure 3.4.

| J26 | | | | | |
|--------------|----|--|--|----|----------------|
| CAN_WAKE | 1 | | | 2 | VDD_3V3_SYS |
| CANO_STBY | 3 | | | 4 | VDD_1V8 |
| CANO_RX | 5 | | | 6 | AP2MDM_READY |
| CANO_TX | 7 | | | 8 | VDD_5V0_IO_SYS |
| CANO_ERR | 9 | | | 10 | GND |
| GND | 11 | | | 12 | I2C_GP2_CLK |
| CAN1_STBY | 13 | | | 14 | I2C_GP2_DAT |
| CAN1_RX | 15 | | | 16 | WDT_TIME_OUT_L |
| CAN1_TX | 17 | | | 18 | I2C_GP3_CLK |
| CAN1_ERR | 19 | | | 20 | I2C_GP3_DAT |
| GND | 21 | | | 22 | SLEEP |
| I2S1_CLK | 23 | | | 24 | I2S1_SDOUT |
| I2S1_SDIN | 25 | | | 26 | I2S1_LRCLK |
| DSPK_OUT_CLK | 27 | | | 28 | GND |
| DSPK_OUT_DAT | 29 | | | 30 | GNSS_PSS |

Figure 3.4. Jetson TX2 J26 Pinouts

The “can dump” that is one of the “can-util” Linux command line tools, is used to read the data coming from the CAN-BUS.

3.2.2. Analyzing & understanding data

For being able to analyze & understand data, 3 small programs are made using C and GTK.

- A. The first program is used to Read all data received from the car's CAN-BUS and list them by ID, then keep showing the data values of last received 10 messages for each ID. that's to keep monitoring the changes in received data values in real-time affected by car actions, that's to select IDs thought to be of interest and use them in the second program. A screenshot of this program is shown in Figure 3.5.

| Counter | Msg IDs | Msg..10 | Msg..09 | Msg..08 | Msg..07 | Msg..06 | Msg..05 |
|---------|---------|----------------------|----------------------|----------------------|----------------------|----------------------|---------------------|
| 1 | 170 | 8005833446928314138 | 8005833446928314138 | 8005833446928314138 | 8005833446928314138 | 8005833446928314138 | 8005833446928314138 |
| 2 | 32 | 721879040 | 721879040 | 721879040 | 721879040 | 671350784 | 721879040 |
| 3 | 452 | 13276048975010087939 | 13276048975010087939 | 13564279351161800707 | 13564279351161800707 | 13564279351161800707 | 130598761928963 |
| 4 | 610 | 592705486881 | 592705486881 | 592705486881 | 592705486881 | 592705486881 | 592705486881 |
| 5 | 705 | 13763227312840179976 | 13763227312840179976 | 13763227312840179976 | 13907342505211068680 | 13763227312840179976 | 13619112124764 |
| 6 | 865 | 6656 | 6656 | 6656 | 6656 | 6656 | 6656 |
| 7 | 36 | 16681605982115135490 | 16681605982115135490 | 16681605982115135490 | 16681605982115135490 | 16537490794005725186 | 165374907940057 |
| 8 | 611 | 15059717988520966 | 15059717988520966 | 15341192965231878 | 15622667941942790 | 15341192965231878 | 153411929652318 |
| 9 | 37 | 15492383405382830080 | 15492383405382830080 | 15492383405382830080 | 15492383405382830080 | 15492383405382830080 | 154923834053828 |
| 10 | 451 | 36 | 36 | 36 | 36 | 36 | 36 |
| 11 | 707 | 2053402836057718784 | 2053402836057718784 | 2053684311034429440 | 2053684311034429440 | 2053684311034429440 | 205368431103442 |
| 12 | 1088 | 4580180546 | 4580180546 | 4580180546 | 4580180546 | 4580180546 | 4580180546 |
| 13 | 547 | 3242591731706757120 | 3242591731706757120 | 3242591731706757120 | 3242591731706757120 | 3242591731706757120 | 324259173170675 |
| 14 | 180 | 13546827679130451968 | 13546827679130451968 | 13546827679130451968 | 13546827679130451968 | 13546827679130451968 | 135468276791304 |
| 15 | 456 | 128 | 128 | 128 | 128 | 128 | 128 |
| 16 | 186 | 3187671040 | 3187671040 | 3187671040 | 3187671040 | 3187671040 | 3187671040 |
| 17 | 548 | 576461851815051264 | 576461851815051264 | 576461851815051264 | 576462951326679040 | 576462951326679040 | 576462951326679 |
| 18 | 608 | 6336845051191623432 | 6336845051191623432 | 6336845051191623432 | 6264224507200339720 | 6264224507200339720 | 633684505119162 |
| 19 | 963 | 179220395327488 | 179220395327488 | 179220395327488 | 181419418583040 | 182518930210816 | 182518930210816 |
| 20 | 1217 | 50331649 | 50331649 | 50331649 | 50331649 | 50331649 | 50331649 |
| 21 | 896 | 16044073672515584 | 16044073672515584 | 16044073672515584 | 16044073672515584 | 16044073672515584 | 160440736725155 |
| 22 | 1597 | 21185691853783193 | 21185691853783193 | 21185691853783193 | 21185691853783193 | 21185691853783193 | 9 |
| 23 | 563 | 956301312 | 956301312 | 956301312 | 956301312 | 956301312 | 956301312 |

Figure 3.5. SC while Monitoring Data of all Received IDs from Toyota Corolla CAN-Bus

- B. The second program is used to Read and monitor just one specific ID at a time by listing all data received from it Byte-by-Byte. That's to be sure whether an ID picked from the first program was really interesting or not. And if so, then to be able to determine exactly which of the ID's bytes is of important values to the system and what is the meaning of these values. In Figure 3.6. bytes read from the engine temperature sensor id (0x488) 1st byte is shown while its value is 0x112 which indicates a real 74 or 75 °c.

| Counter | Msg IDs | Byte..1 | Byte..2 | Byte..3 | Byte..4 | Byte..5 | Byte..6 | Byte..7 | Byte..8 |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 916 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 915 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 914 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 913 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 912 | 548 | 32 | 0 | 0 | 0 | 2 | 97 | 0 | 8 |
| 911 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 910 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 909 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 908 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 907 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 906 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 905 | 548 | 32 | 0 | 0 | 0 | 2 | 97 | 0 | 8 |
| 904 | 548 | 32 | 0 | 0 | 0 | 2 | 98 | 0 | 8 |
| 903 | 548 | 32 | 0 | 0 | 0 | 2 | 96 | 0 | 8 |
| 902 | 548 | 32 | 0 | 0 | 0 | 2 | 98 | 0 | 8 |
| 901 | 548 | 32 | 0 | 0 | 0 | 2 | 97 | 0 | 8 |
| 900 | 548 | 32 | 0 | 0 | 0 | 2 | 98 | 0 | 8 |
| 899 | 548 | 32 | 0 | 0 | 0 | 2 | 98 | 0 | 8 |

Figure 3.6. SC while Monitoring Data Bytes of Breaks ID 0x224 From Corolla CAN-Bus

C. The third program is used to Separate data collected according to Its IDs into separate log files. Then sending these data again through the CAN-BUS to the car but each ID separately. That's to be sure more and more of the IDs and their bytes' value meanings. In Figure 3.7. the log file of the speedometer ID 0x38D is shown while its value is $0x1A=26_{10}$ that means a real speed 70 KM/h.

```

38D
1 (1517235415.475594) can0 38D#1AA74C8DB30800
2 (1517235415.515628) can0 38D#1AA74C95B11000
3 (1517235415.555647) can0 38D#1AA74C9CB12800
4 (1517235415.595724) can0 38D#1AA74CA4B23D00
5 (1517235415.635680) can0 38D#1AA74CABAF4900
6 (1517235415.675711) can0 38D#1AA14CB3B25100
7 (1517235415.715721) can0 38D#1AAC4CBAB06000
8 (1517235415.755742) can0 38D#1AA14CC2AF7300
9 (1517235415.795541) can0 38D#1AA74CC9B08300
10 (1517235415.835926) can0 38D#1AA74CD1AD9D00
11 (1517235415.875736) can0 38D#1AA74CD9B0A000
12 (1517235415.915741) can0 38D#1AB24CE0AFBD00
13 (1517235415.955752) can0 38D#1AB24CE8AEC500
14 (1517235415.995749) can0 38D#1AC34CEFB1D700
15 (1517235416.035737) can0 38D#1AC34CF7B0EE00
16 (1517235416.075681) can0 38D#1AC34CFFB0F500
17 (1517235416.115737) can0 38D#1ACE4D06B10F00

```

Figure 3.7. SC while Logging & Resending Data of Speed ID 0x38D through Peugeot CAN-Bus

Understanding values of bytes related to each ID was approached specifically using the second program by monitoring specific ID and its bytes in real-time.

3.2.3. Creating virtual gauges design & controlling it in Real-Time

After data coming from the CANBUS is captured, a design for gauges is carried out and controlled using C++ & OpenGL, using an already existing design for an Audi A4 car's gauges design. In this stage the real-time data read from the vehicle's CANBUS is used to control the virtual design in real-time.

3.2.4. Integrating camera Real-Time stream

Next, the camera stream is introduced to the system and displayed in real-time in the middle of the virtual cluster.

3.2.5. Installing & training YOLO

In this stage the Object detection algorithm YOLOv3 is installed and get to work on the Jetson TX2 board using the detailed descriptions on the YOLO website. Then it is trained using a dataset of 1000 images to detect some road objects like road lanes. But just for now for more accurate results the pre-trained YOLOv3 and Tiny YOLO models are used to detect just these 7 objects (“person”, “car”, “truck”, “bus”, “motorcycle”, “bicycle” and “traffic light”).

3.2.6. Compiling YOLO with the virtual gauges integrated in a one system

At the last stage of integration all the parts are of the system is put together worked as a one system. The system controls the indicators and gauges in the virtual cluster using real-time data coming from the CAN BUS of the vehicle. Also, at the same time, the real-time stream of the camera is passed to the object detection algorithm YOLOv3 and the final YOLO output is displayed in the middle of the virtual cluster too. In Figure 3.8., the integrated system’s screenshot is shown while system is detecting objects in real-time.



Figure 3.8. A Screenshot of the Final System Working in Real-Time

4. TESTING AND EVALUATING

Evaluating the accuracy of a system is the most important task in any study. The India Driving Dataset IDD that contains 4,794 testing images from total number of 46,588 images is used to test this system using 3,414 images from it.

A C++ program is written to convert the Ground truth annotation files of the IDD from XML format to the same format as YOLO txt output files. Then system outputs are logged into txt files. And another C++ program is written and used to compare the system's output files with the ground truth files. Finally, Accuracy, Precision, Recall & F1 Score values are calculated for both YOLOv3 & Tiny YOLO models.

Before looking at the results of testing lets understand some important accuracy metrics used in evaluation.

4.1. Confusion matrix

The confusion matrix is a table reflects the performance of a system through comparing 4 values. Table 4.1. illustrates these 4 values.

Table 4.1. Confusion Matrix

| | | Predicted Class | |
|--------------|-----|-----------------|-----|
| | | No | Yes |
| Actual Class | No | TN | FP |
| | Yes | FN | TP |

TP & TN are the correctly predicted positive or negative values compared to the actual class.

- True Positives (TP) - When actual class is YES and predicted class is also YES.
- True Negatives (TN) - When actual class is NO and predicted class is also NO.

FP & FN are the values occurs when the actual class is different from the predicted class.

- False Positives (FP) – When actual class is NO and predicted class is YES.
- False Negatives (FN) – When actual class is YES but predicted class in NO.

4.2. Accuracy metrics

Values of confusion matrix are used in calculating 4 important accuracy metrics.

- Accuracy

It is the ratio of classes predicted correctly to total number of predictions. Despite being an important metric, accuracy is not the perfect indicator in some kinds of models, like those having a large imbalance in classes existence, achieving high accuracy rate but with wrong real indication. So, beside accuracy, further performance metrics are needed to be able to evaluate the performance of a model better. such as the F1 score.

- Precision

It is the ratio of correctly predicted observations as positives, to the total number of observations predicted as positives. which means that precision is inversely proportional with the rate of false positives.

- Recall (Sensitivity)

It is the ratio of correctly predicted observations as positives, to the total number of observations must has been predicted as positives (means to all positives in actual class). So, it answers the question: How many predicted positive values from the all actual positive values.

- F1 score

“F1 score is the harmonic mean of Precision and Recall.” [6]

4.3. Experimental Results

The part of reading real-time data from car’s CAN-BUS and using these data to control the virtual instrument cluster in real-time is working great.

The pre-trained YOLOv3 and Tiny YOLO models are used for testing the system. The output of the system is logged. Then it is compared to the truth table of the IDD dataset

by calculating the IoU between the recognized objects and the original objects in the truth table. Comparison is applied in this way:

- All objects recognized by the system that originally exist in the truth table and with an $\text{IoU} \geq 0.5$ are considered as True Positive (TP).

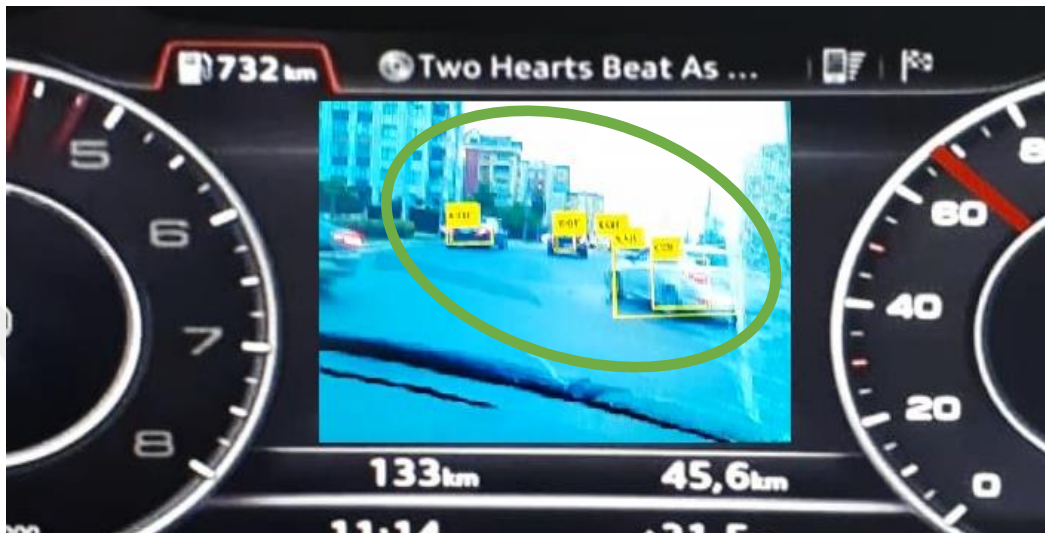


Figure 4.1. Cars are detected as cars correctly (cars exist in truth table) => TP

- All objects detected by the system but originally doesn't exist in the truth table or with an $\text{IoU} < 0.5$ are considered as False Positive (FP).

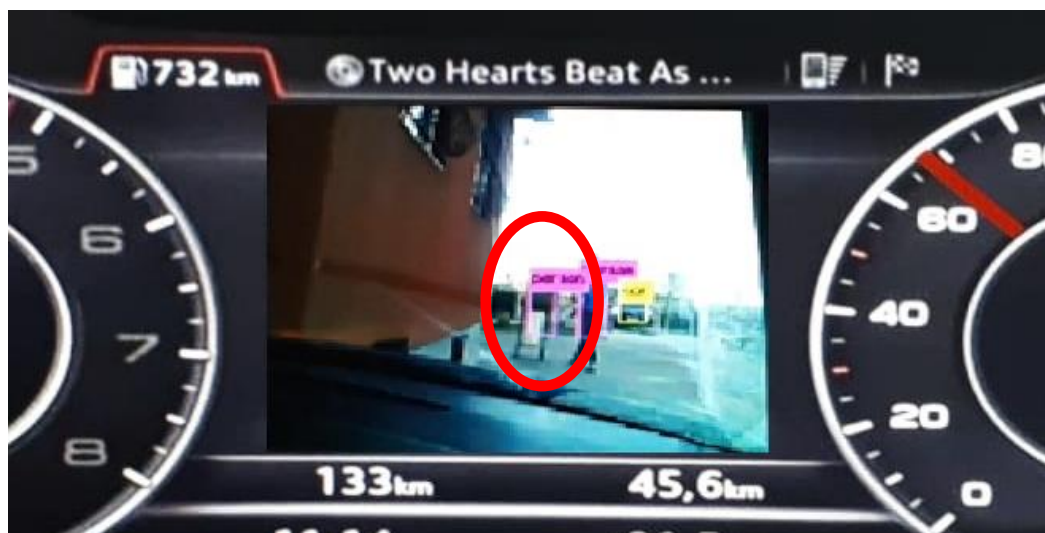


Figure 4.2. Trash Can is detected as a person incorrectly (no person in truth table) => FP

- All objects not recognized by the system and originally doesn't exist in the truth table are considered as True Negative (TN).

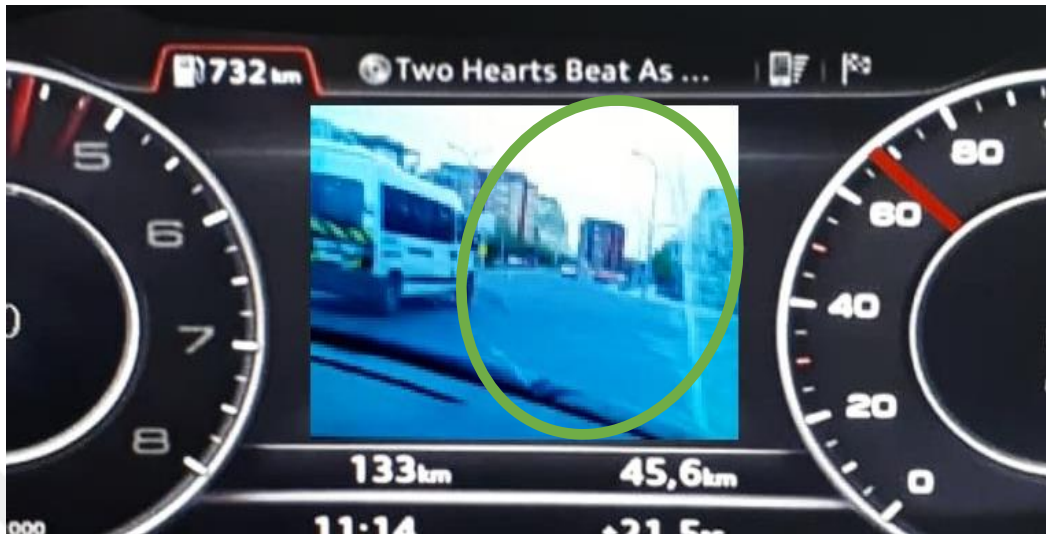


Figure 4.3. No person is detected and no person in the truth table => TN

- All objects not recognized by the system, but originally exists in the truth table and must has been detected are considered as False Negative (FN).

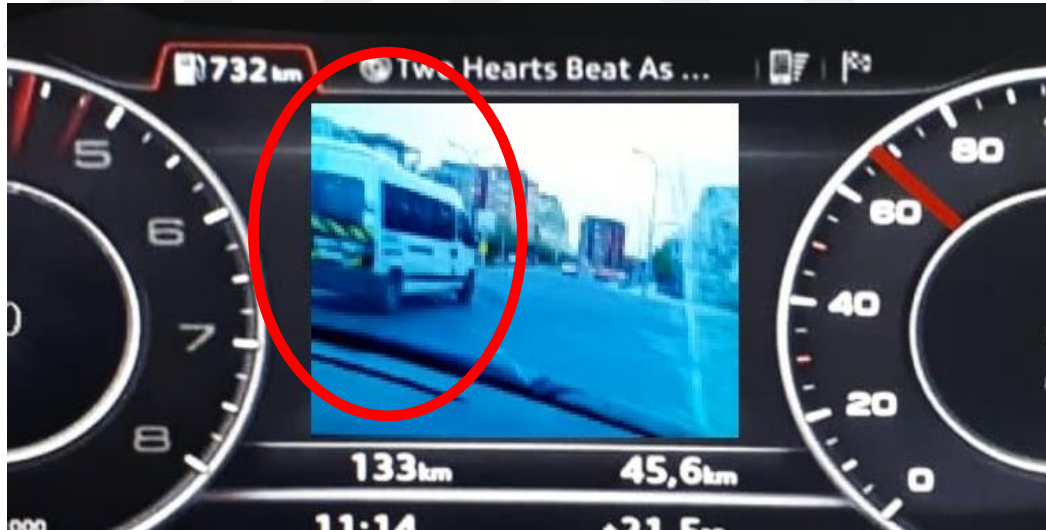


Figure 4.4. Car exists in truth table but is not detected by the system => FN

The detailed values of TP, TN, FP and FN are shown in Table 4.2. And the total values of them, beside the evaluated Accuracy, Precision, Recall and F1-Score for each of the YOLO V3 and V3-Tiny are shown together with the processing speed of the system in Table 4.3.

Table 4.2. Detailed system testing results using YOLOv3 and Tiny YOLO models

| | YOLO V3 | | | | | YOLO V3 Tiny | | | | |
|---------------|---------|-------|------|------|----------|--------------|-------|------|------|----------|
| | TP | TN | FP | FN | Accuracy | TP | TN | FP | FN | Accuracy |
| Bicycle | 59 | 3273 | 39 | 63 | 0.97 | 18 | 3299 | 17 | 104 | 0.96 |
| Bus | 683 | 2535 | 170 | 372 | 0.86 | 426 | 2587 | 204 | 629 | 0.78 |
| Car | 2820 | 1442 | 332 | 815 | 0.79 | 1763 | 1477 | 1149 | 1872 | 0.52 |
| Motorcycle | 1256 | 1865 | 166 | 1051 | 0.72 | 411 | 1874 | 236 | 1896 | 0.52 |
| Person | 2799 | 1195 | 727 | 1756 | 0.62 | 1187 | 1281 | 888 | 3368 | 0.37 |
| Traffic Light | 4 | 3370 | 19 | 27 | 0.99 | 0 | 3386 | 1 | 31 | 0.99 |
| Truck | 527 | 2245 | 603 | 399 | 0.73 | 259 | 2600 | 249 | 667 | 0.76 |
| Total | 8148 | 15924 | 2056 | 4483 | 0.79 | 4064 | 16504 | 2744 | 8567 | 0.65 |

Table 4.3. Final system testing results using YOLOv3 and Tiny YOLO models

| | TP | TN | FP | FN | Accuracy | Precision | Recall | F1 Score | Speed |
|--------------|------|-------|------|------|----------|-----------|--------|----------|-----------|
| YOLO V3 | 8148 | 15924 | 2056 | 4483 | 0.79 | 0.8 | 0.65 | 0.71 | 3-6 FPS |
| YOLO V3 Tiny | 4064 | 16504 | 2744 | 8567 | 0.65 | 0.6 | 0.32 | 0.42 | 19-22 FPS |

From Table 4.3 it seems that the speed of object detection is inversely proportional to the accuracy of the model. Means, for the YOLOv3 model it works very slow about 3-6 FPS (Frames Per Second) but with a great accuracy and F1 scores of about 79% or more and 0.71 respectively. On the other hand, Tiny YOLO model works relatively very fast about 19-22 FPS. Unfortunately, the accuracy and F1 score are noticeably low with values 65% and 0.42 respectively in this case.

The big values of False Negatives specially in YOLOv3-Tiny affects accuracy and F1-score considerably. As a False negative means that the system failed to recognize an existing object, FN can be reduced by training the model with more data varies in size, complexity, the time images are taken in (like in day light and in night), etc. which will probably increase the accuracy and F1-score.

5. CONCLUSION AND FUTURE WORKS

In this study, a deep learning based real-time object detection algorithm is used to create an Advanced Driver-Assistance System (ADAS) implemented in a virtual instrument cluster. A virtual cluster taking an Audi A4 gauges design as its basic design is created, then it is controlled by a real-time data read from CAN-BUS of the vehicle through the OBD-II port, and finally an object detection algorithm is integrated to the system to show surrounding objects detected on the middle of the cluster's screen.

System is tested using two models, the YOLOv3 which was accurate but very slow and the Tiny YOLO which was not that accurate but very sufficient in terms of speed. In future work the system is going to be trained with a bigger dataset or use other models for getting better results in terms of accuracy. Also, it will start reacting with driver like warning him of a danger with visual & auditory alerts. Or even taking the decision itself in emergency conditions.

REFERENCES

- [1] Pugliese M. 2018. *A very shallow overview of YOLO and Darknet. Clearly Erroneous.* Weblog [Online] Available from: <https://martinapugliese.github.io/recognise-objects-yolo/> [Accessed 2018].
- [2] Jinadasa L. 2016. *Study & Design of Instrument Cluster System for a ElectricTrack-Day Car.* Bachelor Thesis. University of Wolverhampton, United Kingdom. 2016.
- [3] Krig S., *Computer Vision Metrics Survey, Taxonomy and Analysis.* 1st ed., Apress, Berkeley, CA, 2014.
- [4] Redmon J., Divvala S., Girshick R., Farhadi A., *You Only Look Once: Unified, Real-Time Object Detection*, Facebook AI Research. University of Washington, Allen Institute for AI, 2016, arXiv: 1506.02640v5.
- [5] Redmon J. *Darknet: Open Source Neural Networks in C.* (Darknet13) [Framework]. Available from: <https://pjreddie.com/darknet/> . 2013-2016.
- [6] Peter A. Flach, Kull M., *Precision-Recall-Gain Curves: PR Analysis Done Right.* Intelligent Systems Laboratory University of Bristol, UK; 2015.
- [7] Basjaruddin^{1,2} N. C., Kuspriyanto¹, Suhendar², Saefudin² D., Aisyah² S. A., *Lane Keeping Assist System Based on Fuzzy Logic.* ¹School of Electrical Engineering and Informatics Bandung Institute of Technology, ²Department of Electrical Engineering Politeknik Negeri Bandung, Bandung, Indonesia; 2015 International Electronics Symposium (IES); 2015.
- [8] Prof. Tigadi A., Prof. Rudrappa B., Patil R., *Survey on Blind Spot Detection and Lane Departure Warning Systems.* International Journal of Advanced Research in Engineering, Science & Technology (IJAREST), 2015, Volume 7, Issue 5, ISSN(O):2393-9877, ISSN(P):2394-2444.
- [9] Pananurak W., Thanok S., Parnichkun M., *Adaptive Cruise Control for an Intelligent Vehicle*, International Conference on Robotics and Biomimetics, Bangkok, Thailand; 2009.
- [10] Lee B., Wei Y., Guo Y., *Automatic Parking of Self-Driving Car Based on Lidar*, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2017, Volume XLII-2/W7.
- [11] Tsimhoni O., Green P., *Night Vision Enhancement Systems for Ground Vehicles: The Human Factors Literature*, Technical Report UMTRI-2002-05, University of Michigan, US Army Research Office; 2002.

- [12] Pablo Carrasco J., de la Escalera A. and Maria Armingol J J. P., *Driving Warning System Based on Visual Perception of Road Signs*, Intelligent System Lab, The Third International Conference on Computer Vision Theory and Applications, Universidad Carlos III de Madrid, Spain; DOI: 10.5220/0001076800540060.
- [13] MCfan. The First-Generation Monte Carlo Club. *Full gauge dash upgrade*. [Online] 18th Dec 2012. Available from: <http://www.firstgenmc.com/forums/index.php?/topic/8396-full-gauge-dash-upgrade/> (Date of Access: 21 Jun 2019).
- [14] Wikipedia the free encyclopedia. *Electronic Instrument Cluster*. [Online] Available from: https://www.wikiwand.com/en/Electronic_instrument_cluster (Date of Access: 21 Jun 2019).
- [15] <https://www.bimmerfest.com/forums/showthread.php?t=536513> (Date of Access: 21 Jun 2019).
- [16] Xue Y. and Chen Y., *EV Instrument Panel Design Based on User Research and E-prime Experiment*, 19th Triennial Congress of the IEA, School of Art and Design, Xi'an University of Technology, Xi'an, China; 2015.
- [17] Cano E., Gonzalez P., Maroto M. and Villegas D., *Head-up Displays (HUD) in driving*; 2018. arXiv:1803.08383v1.
- [18] Fedral Highway Administrator, University of Minnesota, *Intersection collision avoidance system trial*. ITS International, 2010 [Online] Available from: <https://www.itsinternational.com/sections/nafta/features/intersection-collision-avoidance-system-trial/> (Date of Access: 21 Jun 2019).
- [19] Chandrashekar C., Dr. Ramesh B. and Vijay R., *Automatic car AC control using CAN Protocol*. International Journal of Innovative Science, Engineering & Technology (IJSET), Vol. 1 Issue 6; 2014. ISSN 2348-7968.
- [20] Mitchel B., *OSI Model Reference Guide*. Life Wire, 2019 [Online] Available from: <https://www.lifewire.com/osi-model-reference-guide-816289> (Date of Access: 21 Jun 2019).
- [21] Richards P., *A CAN Physical Layer Discussion*, Microship Technology Inc., AN228; 2002.
- [22] Ng W. L., Ng C. K., Ali B. M., Noordin N. K., *Performance Evaluation of Wireless Controller Area Network (WCAN) Using Token Frame Scheme*, International Conference on Intelligent Systems, Modelling and Simulation, DOI: 10.1107/s11277-013-1099-7.
- [23] Moniaga J. V., Manalu S. R., Hadipurnawan D. A. and Sahidi F., *Diagnostics Vehicle's Condition Using OBD-II and Raspberry Pi Technology: Study*

Literature. International Conference on Computer Science and Computational Intelligence (ICCSCI); 2017.

- [24] Vahab A., Naik M. S., Raikar P. G., Prasad S. R., International Research Journal of Engineering and Technology (IRJET), *Applications of Object Detection System*, 2019, Volume 06, ISSN:2395-0056/2395-0072.
- [25] Sachan A., *Zero to Hero: Guide to Object Detection using Deep Learning: Faster R-CNN, YOLO, SSD.*, CV-Tricks, 2017 [Online] Available from: <https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/> (Date of Access: 21 Jun 2019).
- [26] CVIS P. Viola and M. Jones. *Robust real-time face detection*. International Journal of Computer Vision. Springer, 2(57):137–154, 2004
- [27] Mita T., Kaneko T. and Hori O., *Joint Haar-like features for face detection*. IEEE International Conference on Computer vision. Vol.2, 2005; DIO:10.1109/ICCV.2005.129
- [28] Gandhi R., *R-CNN, Fast R-CNN, Faster R-CNN, YOLO-Object Detection Algorithms*, Medium Weblog. [Online] Available from: <https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e> (Date of Access: 21 Jun 2019).
- [29] Geitgey A., *Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning*, Artificial Intelligence, 2016 Weblog. [Online] Available from: <https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffe121d78> (Date of Access: 21 Jun 2019).
- [30] Girshick R., Donahue j., Darrrell T., Malik J., *Rich feature hierarchies for accurate object detection and semantic segmentation Tech report (v5)*. UC Berkeley, 2014, arXiv:1311.2524v5.
- [31] Ouaknine A., *Review of Deep Learning Algorithms for Object Detection*, Zyl Story, 2018 Weblog. [Online] Available from: <https://medium.com/zylapp/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852> (Date of Access: 21 Jun 2019).
- [32] Kaiming He, Zhang X., Ren S. and Sun J., *Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2015, arXiv:1406.4729v4.
- [33] Prof. John J., Balpande S., Jain P., Chatterjee A., Gupta R., Raut S., *Review Paper on Object Detection using Deep Learning- Understanding different Algorithms and Models to Design Effective Object Detection Network*. International Journal for Research in Applied Science & Engineering Technology (IJRASET), 2019, Volume 7, ISSN:2321-9653.

- [34] Dai J., Li Y., He K. and Sun J., R-FCN: *Object Detection via Region-based Fully Convolutional Networks*. Technical Report; 2016. arXiv:1605.06409v2.
- [35] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu¹, Alexander C. Berg¹, *SSD: Single Shot Multibox detector*.
- [36] Weng L., *Object Detection Part 4: Fast Detection Models*, Lil'Log, 2018 Weblog. [Online] Available from: <https://lilianweng.github.io/lil-log/2018/12/27/object-detection-part-4.html> (Date of Access: 21 Jun 2019).
- [37] Kathuria A., *What's new in YOLO v3?* Medium, 2018 Weblog. [Online] Available from: <https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b> (Date of Access: 21 Jun 2019).
- [38] Hussaini U., *NVIDIA's Jetson TX2 is a powerful board for local on-board Machine Learning applications*. Technobyte, 2017 Weblog. [Online] Available from: <https://www.technobyte.org/nvidias-jetson-tx2-powerful-board-local-onboard-machine-learning-AI/> (Date of Access: 21 Jun 2019).
- [39] <https://electronics.stackexchange.com/questions/281715/connecting-usb-powered-beaglebone-to-cars-can-bus-through-obd-connector-using-c> (Date of Access: 21 Jun 2019).
- [40] Wei Lun Ng, Chee Kyun Ng, Ali B. M., Noordin N. K. and Rokhani F. Z., *Review of Researches in Controller Area Networks Evolution and Applications*, Department of Computer and Communication Systems, Faculty of Engineering, University Putra Malaysia; 2010. DOI:10.7125/APAN.30.3.
- [41] Viola¹ P., Jones² M., *Rapid Object Detection using a Boosted Cascade of Simple Features*. Master thesis. ¹Mitsubishi Electric Research Labs, ²Compaq CRL One Cambridge Center; 2001.
- [42] Redmon J., Farhadi A., *YOLO9000: Better, Faster, Stronger*, Facebook AI Research, University of Washington, Allen Institute for AI, 2016, arXiv:1612.08242v1.
- [43] Dugarry A. and Dr Shan Fu. *Advanced Driver Assistance Systems Information Management and Presentation*. PhD thesis. Cranfield University, School of Engineering, Applied Mathematics and Computing Group, Britain. 2004.
- [44] Redmon J., Farhadi A., *YOLOv3: An Incremental Improvement*, Facebook AI Research, University of Washington, Allen Institute for AI, 2018, arXiv:1804.02767v1.

PUBLICATION AND WORKS

Fawzy A., Prof. Dr. Urhan O., A deep learning based Real-time object detection implementation in a Virtual instrument cluster, *IMASCON International Marmara Science and Social Sciences Congress*, Kocaeli, Turkey, 26-28 Apr (2019)



RESUME

Abdelrahman Magdy Fawzy (Abdurrahman Hasanoglu)

Born in Alexandria/Egypt in Feb 1991.

Finished primary, preparatory and secondary educations in Sidi-Gaber Language School (S.L.S) and received the General Certificate of Secondary Education in 2008.

Graduated from Faculty of Science, Alexandria University. And received the Bachelor's degree from the department of Computer Science & Statistics in 2012. A "Smart Home" is the graduation project in the Bachelor's degree.

Joined Kocaeli university in 2014 as a part of "Türkiye Bursları" scholarship program to make a Masters in Electronics and Communication Engineering. And this study is the research thesis for this master's degree.

Worked as an IT Engineer in more than one TV channels. And now working as an IT and broadcast engineer in a British company that provides technical support for TV channels and media companies.