ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARMENT

Project Report

CENG 408

Innovative System Design and Development
II

202107

Advisor: ***Dr. Serdar ARSLAN***

Prepared By	**ID**
Aleyna DEDE	201711019
Ali BOZDOĞAN	201717009
Eylül ERDOĞAN	201711025

25.05.2022

Contents

Abstract

Autonomous vehicle technology and object detection algorithms include multiple engineering concepts. In this project, it is aimed to combine these techniques and generate a national unmanned ground vehicle. Therefore, real time object detection operation will be handled and sent to commanders via our application which offers both manual and autonomous operations. It can be also used for threat detections. It will be capable of making reconnaissance and surveillance around critical facilities which requires high security. The main motivation behind this project is keeping our soldiers safe while planning an attack or defending an area. By using our system, battlefield can be monitored from a distance and when there is a danger or threat, the necessary precautions can be taken since detection is handled and commanders will be informed about these suspicious detections. In this way, vehicle will support our army by spying enemy and warn soldiers when threats are detected.

Key words:

Autonomous Vehicle, Object Detection, Defense Industry, Artificial Intelligence and Robotics

Özet

Otonom araç teknolojisi ve nesne tespit algoritmaları birçok mühendislik konseptlerini içerir. Bu projede, bu teknikleri birleştirmek ve milli insansız kara aracı üretmek amaçlanmıştır. Bu sayede, gerçek zamanlı nesne tanımlama operasyonu gerçekleştirilecek ve komutanlara geliştirecek olduğumuz uygulama aracılığı ile gönderilecektir. Bu uygulama manuel ve otonom operasyonları sunacaktır. Aynı zamanda, tehditleri tespit etmek için de kullanılacaktır. Yüksek güvenlik gerektiren kritik tesislerin etrafında keşif ve gözlem yapma kabiliyetine sahip olacaktır. Bu projeyi geliştirme sürecinin arkasındaki ana motivasyon kaynağımız, saldırı ya da savunma için plan yapma sürecinde, askerimizi güvende tutmaktır. Sistemimiz kullanılarak muharebe alanı uzaktan izlenebilmekte ve herhangi bir tehlike veya tehdit olduğunda tespit işlemi yapıldığı için gerekli önlemler alınmakta ve bu şüpheli tespitler hakkında komutanlara bilgi verilmektedir. Bu sayede araç, düşmanı gözetleyerek ordumuza destek olacak ve tehdit tespit edildiğinde askerleri uyaracaktır.

Anahtar Kelimeler:

Otonom Araç, Nesne Tespiti, Savunma Sanayi, Yapay Zeka ve Robotik

1.Introduction

In this report, a comprehensive explanation of our vehicle is stated. For that purpose, we will state Literature Review about project, and clarify Software Requirement Specification Software Design Description of the project. We will also share our Work Plan to state how we developed our project.

1.1. Problem Statement

Soldiers are in danger while spying on enemy's facilities or battlefield. On the other hand, there are many facilities which require high security due to classified information. With our project, it is aimed to keep our soldiers safe and save their energy for another beneficial purposes. Even more, soldiers do not need to worry about upcoming threats, since our vehicle sends a warning when there is one. Therefore, we will increase the survival rate of our soldiers since they will be replaced with Kaşif UGVs.

1.2. Background or Related Work

In recent years, autonomous vehicle became very critical and popular technology. Especially, defense industry companies are aware of how they can use this technology for soldiers' benefits. Therefore, companies like ASELSAN, HAVELSAN, ROKETSAN, etc. developed samples of autonomous vehicle which are used for different purposes. Generally, they are used for surveillance and making reconnaissance. On the other hand, armed versions of these vehicles exist as well.

1.3. Solution Statement

To solve this problem, we used ESP32 Camera, Arduino Uno board, DC motors, sensors, and other hardware components. All these tools and techniques will be explained in SRS and SDD documents which are stated in this document. By using wireless communication, we will receive real time video which includes suspicious objects. These objects are detected by using YOLO algorithm and other AI concepts. By using DC motors, wheels, Arduino board and other hardware components, moving operation of car will be handled. In this way, our vehicle will be capable of implementing self-driving, manual-driving, and object detection operations.

1.4. Motivation

This project includes many engineering concepts such as Artificial Intelligence, Robotics, Machine learning and many more. Thanks to this project, we improve ourselves regarding these concepts. Additionally, we had an opportunity to implement hardware components to our project which provides satisfying results. The most important motivation behind this project idea is helping our army by replacing our soldiers when there is an upcoming or possible threat.

2.LITERATURE SEARCH

Abstract

In recent years, autonomous vehicles moved from an imagination to very realistic possibility. With the current technology, they can almost perform all driving tasks without any human activity. To accomplish these tasks, multiple disciplines and concepts need to be combined. This combination includes computer science, electrical engineering, artificial intelligence, mechanical engineering, etc. Since autonomous vehicles includes such variety concepts, it is developed by using various algorithms, techniques, and approaches.

Autonomous vehicles are generated using different equipments including sensors, radars, cameras, and communication systems. All these subsystems perform different tasks and with different implementations. In general, autonomous vehicles are used for self-parking, avoiding obstacles, go on a certain line, etc. However, it is not the first choice when it comes to providing safety.

On this project, we aimed to provide safety of an area using autonomous vehicle. There are many high security facilities which use soldiers, cameras, laser-beams, etc. However, if autonomous systems are used for security situations, many benefits would arise. Therefore, we will develop a system which takes tour around of the facility/area and detects are there any suspicious or dangerous objects. Therefore, it will make reconnaissance and surveillance around critical facilities. In this way, we believe that this system will prevent incoming military attacks and inform military when there is an emergency.

2.1. Introduction

Autonomous vehicles are actively developing and being used in the world day by day. In the Introduction part of our report, we will provide general information about autonomous vehicles. Autonomous vehicles are cars that can move without the need for any intervention, thanks to their unmanned driving and object detection systems. These vehicles can detect objects around them by using camera, radar and lidar technologies and help us to get clear images. Autonomous vehicles detect the positions of objects in the environment using various sensors. The vehicle is driven by examining the data from these various sensors in the computer system. Unmanned vehicles, which provide convenience and safer life to our lives, are becoming more and more common. Our aim in this project is developing a system which provides security using autonomous vehicle. While doing that, we will keep our soldiers safe and decrease the human power. It will detect dangerous objects which can be guns, remote control, suspicious people based on timing and other factors. We believe that it is an efficient approach for defense and military systems since it provides information in hazardous situations.

Autonomous Vehicles occurred of 5 diverse automation levels from zero to five.

Level 0: There is no driving automation at level 0. All driving is done by the driver.

Level 1: Autonomous system uses Driver Assistance when driver drives vehicle on the road at Level 1. For instance, if driver wants to park on the roadside, it helps to make easy parking.

Level 2: In level 2 which refers to Partial Automation, driver is generally not having the control of car, however, when there is an emergency, driver must take the control.

Level 3: In level 3 which known as Conditional Automation, there is a major change which car starts to use sensors such as LIDAR, RADAR, camera, etc. However, in critical situations, driver may need to take over the control.

Level 4: In level 4 which refers to High Automation, car starts to determine some critical decisions such as changing lanes, using signals, etc. Yet, if there is a traffic congestion, car may have trouble while deciding.

Level 5: Level 5 known as a Complete Automation which is the level that no human is needed. Therefore, pedals, brakes or steering wheels are discarded. Autonomous car is responsible for moving by itself.

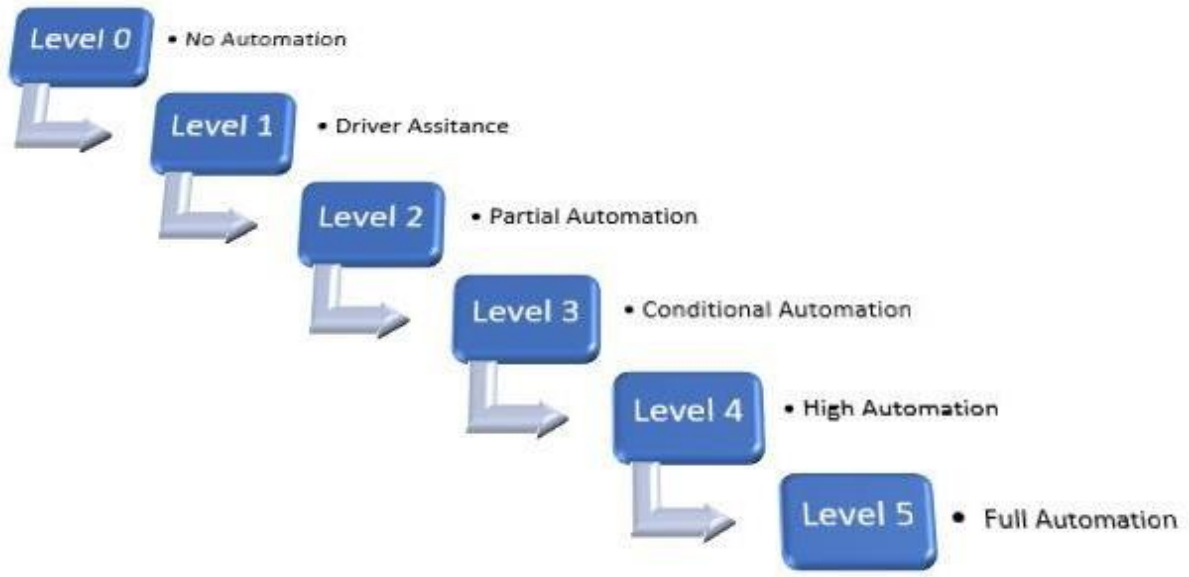


Figure 1: Levels of Autonomous Cars

In this Literature Review report, we have 9 sections which are given in order. In *1.Introduction* section, we described our project scope and what will be explained in the rest of the report. In *2.History of Autonomous Vehicles* section, we analyzed autonomous vehicles which are developed from 1925 to 2021. Afterwards, we provided samples from Defence Industry products in *3.Autonomous Vehicles in Defence Industry* section. Next, we described perception and provide summarized information related perception in *4.Object Detection with* Perception section. We also stated safety issues regarding autonomous vehicles in next section.

Afterwards, in *6. Solution and Algorithms* section, we described detailed solutions, techniques and algorithms related to technology that we will be using in this project. Next, in 7. Previous Works and Articles section, we stated articles that we used while writing this literature review. We also shared which comparisons and techniques that we possibly use during the development of project. Then, we summarized solutions and techniques which are explained in the report. Finally, in 9.Conclusion section, we also stated summarized version of our project scope and what will be implementing during the development process.

2.2. History of Autonomous Vehicles

In **1925**, Francis Houdina developed the first sample of autonomous vehicles in the history. This development was a radio-operated automobile which was a 1926 Chandler. The vehicle had transmitting antenna on its tonneau. Also, it was controlled by another car which is following it using a transmitter. In this way, this automobile could travel through streets without any human sitting at the steering wheel. Francis Houdina published this invention as American Wonder which changed the automobile industry. [1]

In **1939**, Norman Bel Geddes released the idea of the first self-driving vehicle which uses electricity to be able to operate. The vehicle was self-driving by using radio-controlled electro-magnetic area where magnetized metal spikes hidden and embedded. The idea is known as FUTURAMA.[2]

In **1953**, RCA Labs developed the mini car which was guided by wires positioned on the lab's floor.[3]

In **1954**, General Motors introduced Firebird I the first gas turbine automobile in history. This development was crucial for self-driving history since it leads to Firebird II which was aired couple of years later.[4]

In **1956**, Firebird II was introduced by General Motors as “Laboratory on Wheels”. It was designed to drive as an autonomous vehicle on electronic highways.[5] However, Firebird II was a concept car, therefore; it was never produced. General Motor aired a short musical which aims visualize self-driving cars and show people how can technology evolve in the future.[5]

In **1957**, RCA Labs developed self-driving car with an actual scale in Lincoln, Nebraska. Using embedded wires, signals were emitted, and they could expect to stay car within its line. [6]

In **1962**, Robert Fenton who is a professor at Electric-Electronic department of Ohio State University created a car which can control braking, steering and speed. The car used embedded roadway with wires for moving like previous developed systems.[6]

In **1969**, John McCarthy who is one of the founders of Artificial Intelligence concept published an article called “Computer Controlled Cars”[7]. He highlighted that a computer could control car using television camera input. In this way, the chauffeur will be automatic. In the article, features that car would have, issues, cost and other additional information is given.

In **1977**, Japanese engineers improved the idea of General Motors by using a camera system to be able to process images of the road. However, the vehicle could not travel as much as fast.[2]

In **1980**, Mercedes-Benz which was vision-guided robotic van developed by Ernst Dickmanns from University of Munich. Vehicle was controlled by computer commands using computer vision with image sequences that are cought by four cameras. It could speed up to 63 km/h on roads without traffic.[3]

In **1987**, Germans improved the solution of Japanese engineers who completed their system in 1977. The vehicle called VaMoRs and it could speed up 175km/h without any human interaction.[8]

Between **2004-2007**, Defense Advanced Research Projects Agency (DARPA) organized a competition for American autonomous vehicles. The competition is called DARPA Grand Challenge. The competition served for military use; therefore, it was one of the most important organizations of the United States Department of Defense. There were several DARPA Grand Challenges through years and each of them have huge impacts on autonomous vehicle industry and military products. The first competition of DARPA Grand Challenge organized in 2004. In this first competition, none of the vehicles were able to finish the parkour. Two vehicles travelled the most distance which was 11.78 km. However, that was not enough, therefore, there was no winner and prize given. Hence, second DARPA Grand Challenge was organized in 2005. This time five vehicles could complete the parkour which was 212 km. Also, in 2005 race, vehicles passed through 100 sharp turns and 3 narrow tunnels. The third and last competition of DARPA Grand Urban Challenge event occurred in 2007. In this race, vehicles had some strict limitations and rules such as parkour complete time and distance. In the race, vehicles must operate even in rainy and foggy weather with blocked GPS. Team Tartan Racing won the race and received the prize of 2 million dollars. [9]

In **2009**, Google started to develop Waymo self-driving car project which aims to have empty driver seat. For that purpose, company actually tought an autonomous vehicle how to understand and perceive the environment. After many years, Google contionued to improve project and released two version of Waymo, which are Waymo One and Waymo Via. Waymo One aims to be used for daily services like going to market, school or job. Waymo Via focuses to be used for transportation services.[10]

In **2015**, Tesla released the feature of autopilot which provides self-steering, self-changing lines and adjusting speed. Although the car was not fully autonomous, it was a critical step to moving on fully autonomous ones.[6]

In **2016**, a company called Ottomotto developed Otto Truck which achieved the longest continuous traveling without human intervention as a semi-truck. The total route was 132 miles without any leading vehicle, driver, or any other human interaction.[11]

Between **2016**-**2021**, Tesla company is being one of the leading companies in autonomous vehicle industry. They improved the Autopilot technology which they published in 2005. One of the reasons behind their success is powerful sets of vision processing. They use Deep Neural Network to operate visual processing techniques. In addition to Autopilot technology, they also developed Autosteer+ and Start Summon technologies to have full of self-driving capability. [12]

In this section, we explained which samples do autonomous-vehicle industry have and how far we have come in this area. It can be seen that the autonomous vehicle industry includes variety of technologies, and vehicles can be used for different purposes. Since in our project we are aiming to use autonomous vehicles for providing security, in *3. Autonomous Vehicles in Defence Industry* section, we examined military approaches, products and algorithms that has been developed in recent years. Therefore, we focused on unmanned ground vehicles (UGV).

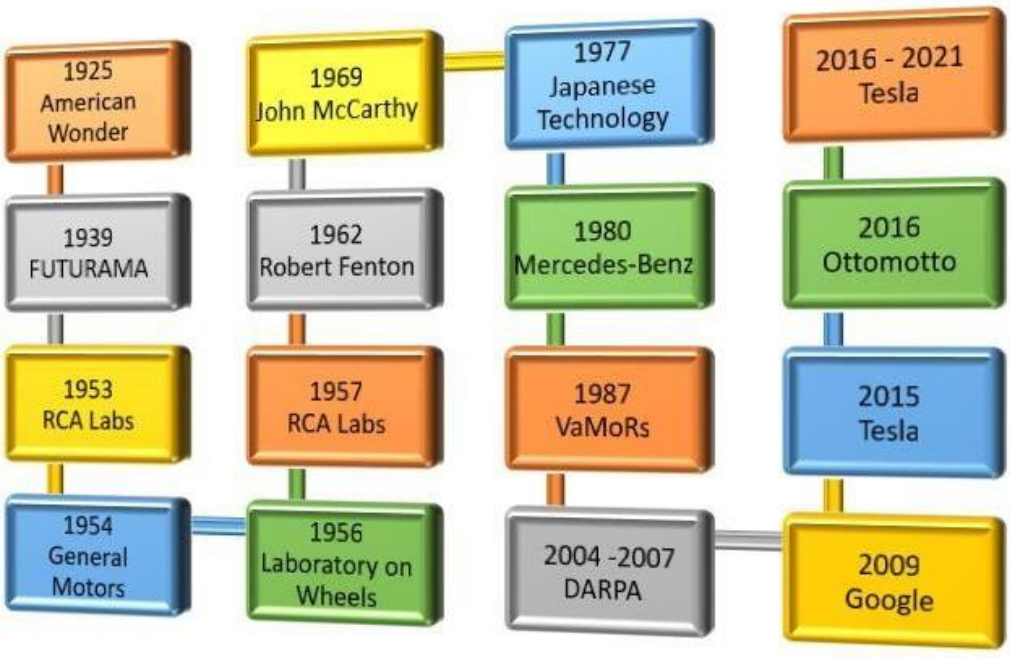


Figure 2: History of Autonomous Cars

2.3. Autonomous Vehicles in Defence Industry

We observed that Defence Industry companies has been using autonomous vehicles, drones, and many other autonomous systems recently. That is why, we aimed to develop a system which can be used for high security required facilities or protection of a certain area. In this section, we will provide examples of both Turkish and foreign countries' products that belong to autonomous-vehicle and UGV concepts. In below table, we summarized the vehicles that we explained in this section; hence, the summarize form can be observed in following figure.

VEHICLE NAME	YEAR	DIMENSIONS	WEIGHT	DETECTION TYPE	RANGE	ARMED
TARANTULA UGV	2018	500x222x210 cm	2,000 kg	Unknown	3 km	YES
ACROB UGV	2021	65x31x14 cm	Unknown	Day/Night Cameras	500 m	NO
O-İKA 2 UGV	2019	Unknown	1100 kg	Mast-mounted Camera	Unknown	YES
ALPAN UGV	2020	142x106x164 cm	500 kg	Day/Night Cameras, LIDAR, RADAR	3 km	YES
BARKAN UGV	2021	140x90x110 cm	500 kg	Day/Night Cameras, LIDAR, RADAR	Unknown	YES
URAN-9 UCGV	2015	512x253x250 cm	10,000 kg	Electro-optic and thermal cameras	3 km	YES
Gladiator TUGV	2005	178x112x135cm	725 kg	Day/Night and Thermal Cameras	Unknown	YES
Miloš UGV	2017	172x70x95 cm	680 kg	Day/Night and Thermal Cameras	3 km	YES

Figure 3: Summarize of UGVs

2.3.1. Turkish Defence Industry Products

2.3.1.1. Tarantula UGV

Tarantula is armed unmanned ground vehicle platform (UGV) which is created by ASELSAN and TEAS in 2018. It is actively used by Turkish Armed Foundation to operate many functions which are surveillance, patrol, defence, and other security services. Tarantula also has a controller which has up to 3 km range. By using the controller, it can shoot as well. It has electrical engine which provides 8 hours to operate. It has 2 tons weight and can carry 800 kg of load. [13] Additionally, Tarantula has length of 500 cm. Its width is 222 cm, and it has 210 cm height.[14]

2.3.1.2. ACROB UGV

In September 2021, Republic of Turkey Ministry of National Defense announced that ACROB UGV enter to the products which services for Turkish Armed Forces. ACROB was generated the company called Elektroland Defense. One of the most significant features of Acrob is being resistant to the different physical environments. Even more, it can climb stairs and other vertical obstacles on road. It is equipped with day/night colour cameras. It uses electricity to be powered and its controllers has maximum 500m range. It is also a small UGV with 31 cm width, 14 cm height and 65 cm length. [15]

2.3.1.3. O-İKA 2 UGV

In 2019, Turkish companies ASELSAN and Katmerciler developed a UGV named O-İKA 2. The vehicle is capable of detection, surveillance and using different weapon systems. As previous ones, O-İKA 2 has camera at both front and rear. Images taken with this camera can be transferred as well. Although the vehicle is autonomous, weapons can be used manually when there is an emergency. Therefore, vehicle can be remotely commanded or used autonomously based on the situation. O-İKA 2 has 1.1 tons weight. Currently, Land Forces Command has the user authority.[16]

2.3.1.4. ALPAN and BARKAN UGVs

In 2020, Turkish company HAVELSAN introduced ALPAN UGV which was their first prototype vehicle. Afterwards, they added some subsystems and introduced BARKAN UGV in 2021. Both UGVs capable of handling different geographical conditions. They also have environmental sensors like LIDAR and RADAR. In addition, they are equipped with day/night camera and other image processing systems.[17] ALPAN has a length of 142 cm, and its width is 106 cm. Also, when it is unarmed, it has 164 cm height. ALPAN has 6 hours operation run time with around 10-15km/h velocity. In addition, it can be controlled from 3km range distance.[18] On the other hand, BARKAN has 500 kg weight. It can operate for 8 hours, and its velocity is 12 km/h at most. BARKAN has a length of 140 cm, width of 90 cm and height of 110 cm.[19]

2.3.2. Foreign Defence Industry Products

2.3.2.1. Uran-9 UCGV

Uran-9 UCGV (Unmanned Combat Ground Vehicle) is produced in Russia, 2015. The producer was Russian military equipment manufacturer JSC 766 UPTK. Uran-9 can speed up to 35 km/h. It was generated to provide fire support and make reconnaissance and surveillance in urban areas. Its weight is approximately 10,000 kg with 512 cm length, 253 cm width and 250 cm height. Therefore, this vehicle is one of the hugest ones compared to its competitors. Its missiles have the range of 0.4km to 6.km. It has both camera types which are electro-optic and thermal imaging. It can detect the suspicious situations up to 6 km distance in the afternoon. It decreases to 3 km at nights. Uran-9 can be used either manual or autonomous mode. The path finding algorithm is used to make vehicle autonomous. For manual mode, Uran-9 has a controller which can operate from a distance of 3 km. [20]

2.3.2.2. The Gladiator TUGV

The Gladiator which was the first tactical unmanned ground vehicle was developed to support the United States Marine Corps. It was designed by Marine Corps on February7, 2005. Then it was produced several hundred in production in 2007. The Gladiator TUGV is an unmanned, compact, robust, tele-operated, multi-purpose ground exploration, observation, and target detection vehicle system possessing a discovering and direct engagement capability. It provides the armed forces with remote observation, exploration, and target detection, biological, nuclear, and chemical reconnaissance in different parts of the battlefield, obstacle breaching, and direct fire capability to deactivate threats and reduce risk to the war warrior. Moreover, it supplies remote imagery software to relay images, including day and night images, and thermal images. It's weight is 725,75 kg with 1,78 m length, 1,12 m width, and 1,35 m height. Since its an unmanned ground vehicle, each Gladiator is equipped with a remote-control unit capable of displaying mission data, operational status, and mission observation.[21]

2.3.2.3. Miloš UGV

Miloš developed by the Military Technical Institute Belgrade in Serbia, 2017. It uses a day and night camera, thermal camera, and a laser ranger. It has one or two guns attached to its turret. Maximum total weight is around 700 kg or 300 kg of cargo. It has two cameras which provide day and night vision are installed on front and back for driving. Moreover, it has thermal camera that enables recognition up to 450 meters. It can recognize solder using charge coupled device up to 1000 meters. In its equipment is laser rangefinder for range up to 2000 meters. Miloš unmanned ground vehicle can negotiate gradients of 57.7% and side slopes of up to 46.6%. It can cross a maximum vertical obstacle of 200mm and a maximum trench width of 250mm. The UGV measures 1.72 m-long, 0.7 m-wide, and 0.95 m-high and has a curb weight of 680 kg, while the weights of the driving base and the battle station are 430 kg and 250 kg. The battle station can revolve in 360° at speeds between 0.05° per second and 48° per second, while the elevation range of the station is between -15° and 50°.[22]

2.4. Object Detection with Perception

2.4.1. Requirements for Perception

2.4.1.1. Definition of Perception

In our project, we will develop a vehicle which will be able to make reconnaissance and surveillance around an urban environment. For that purpose, we must identify each object around facility. To do that, we will percept objects to understand what it is: a human, a riffle, a controller for a bomb, a suspicious car, etc. However, computers cannot handle this operation as fast as humans.

2.4.1.2. Goals for Perception

The objects can be separated in two main sections: Static and Dynamic. Static objects refer to objects which are stabilized and do not move. On the other hand, dynamic objects refer to continuously moving objects such as pedestrians, other cars, wheelers, etc. Object types can be observed from following figure.

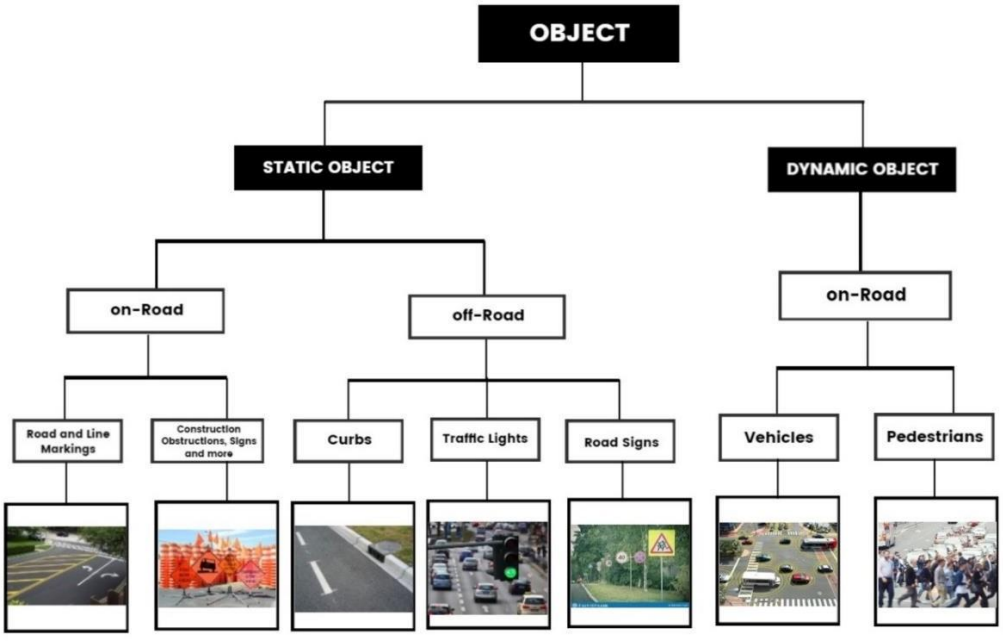


Figure 4: Static and Dynamic Objects

2.4.1.3. Challenges of Perception

While implementing perception, many issues could arise. One of the biggest problems is having limited amount of labeled training data.[23] Since our priority is detecting dangerous objects, trained data must include these kinds of objects for sure. Our vehicle must be able to separate an umbrella and a riffle to provide incorrect warnings. To achieve this separation appropriately, we will use detection algorithms with modern machine learning techniques. Researchers are still implementing studies to equalize detection level of machines with human level. There are many research that compare humans with computer vision algorithms which refers to deep neural networks(DNNs).[24]

2.4.1.4. Solutions to Challenges of Perception

We can reach the best possible solution by combining radar, lidar and camera systems while detecting vision. Another vital solution is to enable the autonomous vehicle to perform better segmentation and detection by using large datasets and more training data.

2.4.1.5. Sensors of Perception

In Perception, there are 3 well-known sensors which are LIDAR, RADAR and Camera. In this section we will be focusing on these sensors. Each of them has their own pros and cons.[25]

- **CAMERA**

  Camera is the oldest sensor technology used in vehicles. Also, it is the most similar sensor type to our eyes. [26] Cameras are great choice while detecting RGB information. Therefore, they have extremely high resolution which comes as an important benefit. However, if the weather is extreme like sunny or foggy, the results are not that great. In addition, contrast is another issue which is an important limitation of camera technology.

- **LIDAR**

LIDAR technology is known as laser imaging detection (light detection) and ranging. It is used for determining distance (range) by targeting object with lasers. For that purpose, it measures the difference of time between sending light and receiving back the reflected light.[27]
When we compare LIDAR with camera and RADAR, we observed that LIDAR is the only sensor that gives resolution at range which means the ability to receive appropriate and accurate detection of objects in space. Additionally, using LIDAR, we can receive great results day and night without any loss of performance unlike Cameras.[28]. In addition, LIDAR implement image analysis much simpler and faster than cameras. However, it has weaknesses, too. It does not have high amount of resolution; hence, it cannot detect objects through bad weather as well as RADAR does. Additionally, LIDAR cannot detect colors so it cannot replace cameras in this point. Therefore, it is very difficult to LIDAR detect any road signs, traffic lights or etc. since it cannot recognize colors.

- **RADAR**

RADAR technology is known as laser imaging detection (light detection) and ranging. It can be basically broken down into receiver and transmitter. Radio waves are reflected when they see an object.[27] Additionally, RADARS can work in bad weathers as well. In this point, we can say that they are better than cameras and LIDAR. However, they do not have range information and resolution at range. Additionally, it cannot detect objects as well as camera does. Therefore, it might have trouble while identifying what the object really is.

In below table, we summarize advantages and disadvantages of each sensor we described.

	Name	Advantage	Disadvantage
 CAMERA	CAMERA	- Can detect colorful objects - High resolution - Low cost - Can recognize 2D	- Bad results when in different weather conditions
	LIDAR	- High-definition of 3D modeling - Gives great results day and night	- Affacted from bad weather - High cost - Cannot recognize colors
	RADAR	- Unaffected by weather conditions - Low cost	- Low-definition modeling

Figure 5: Pros and Cons of Sensors

It can be observed that, three techniques have their own weak and powerful sides. Therefore, it would be better if we can combine three of them. That is why, we can conclude that vehicles are just like humans. Humans do not have only a sensor. They see, touch, taste, smell and hear. When these abilities are working simultaneously, we say that this person is healthy.

2.5. Safety of Autonomous-Vehicles

Today's autonomous vehicles make their movements using sensors. They solve difficult tasks with machine learning. Although high-accuracy algorithms in Machine Learning have emerged recently, automotive software is still insufficient to solve problems such as security standards, interpretability, validation, and performance rankings.

The development of safety-critical systems for autonomous vehicles is based on strict safety methodologies, design, and analysis to prevent hazards in the event of a malfunction. There are 2 main safety standards used worldwide to address the safety of electrical and electronic components, these are ISO26262 and ISO/PAS 21448. Through these standards, they must apply software development, system, and hardware methodologies.

With the use of machine learning in such systems, it is an inevitable fact that these models reveal gaps in security standards in current engineering applications. Many examples of these gaps are discussed on the security of our machine learning, considering factors such as official verification and traceability of design code. (Salay, Queiroz and Czarnecki 2017).

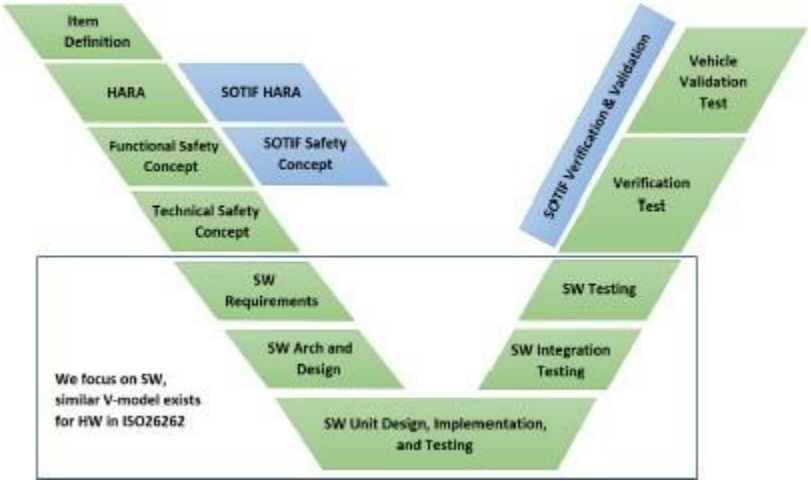


Figure 6: Standarts for Safety of Autonomous Vehicles

2.5.1. ISO 26262 Standard

The other name of ISO26262 is the E/E functional safety standard. This standard is the absence of irrational risks arising from defective E/E components to the safety of the vehicle. Requires a Hazard Analysis and Risk Assessment (HARA) to identify hazards at vehicle level. Potential dangers that may occur in the future direct the security engineers working in this field to the security targets used to create functional security requirements. These requirements lead to the system development process, which is divided into hardware and software development processes in the later process.

2.5.2. ISO/PAS 21448 Standard

It is ISO/PAS 21448 or SOTIF (Security Functionality) standard. The features of this standard are design features, development, and verification stages. This standard defines the performance limitations of the software (including machine learning components) and allows it according to the scenarios and inputs it belongs to. In such cases, this standard reduces the risks of applications that are known to be unsafe. Other risks of error become acceptable in this way. As a result, we are aiming to perform system which provides standards that we explained above.

2.6. Methods and Algorithms

2.6.1. Methods for Self-Driving

Self-driving cars may be the future of transportation, but we do not really know whether it is safer than nonautonomous driving or not. Automakers are spending billions each year to develop self-propelled cars. But it turned out from different studies that people are more concerned than enthusiastic about the appearance of this new technology

When autonomous vehicles are operated without automation and partially automated or highly automated, security becomes a major challenge between the interaction between human and autonomous vehicle. In the full automation mode, the reliability of the software and hardware come to the fore as a very critical issue.

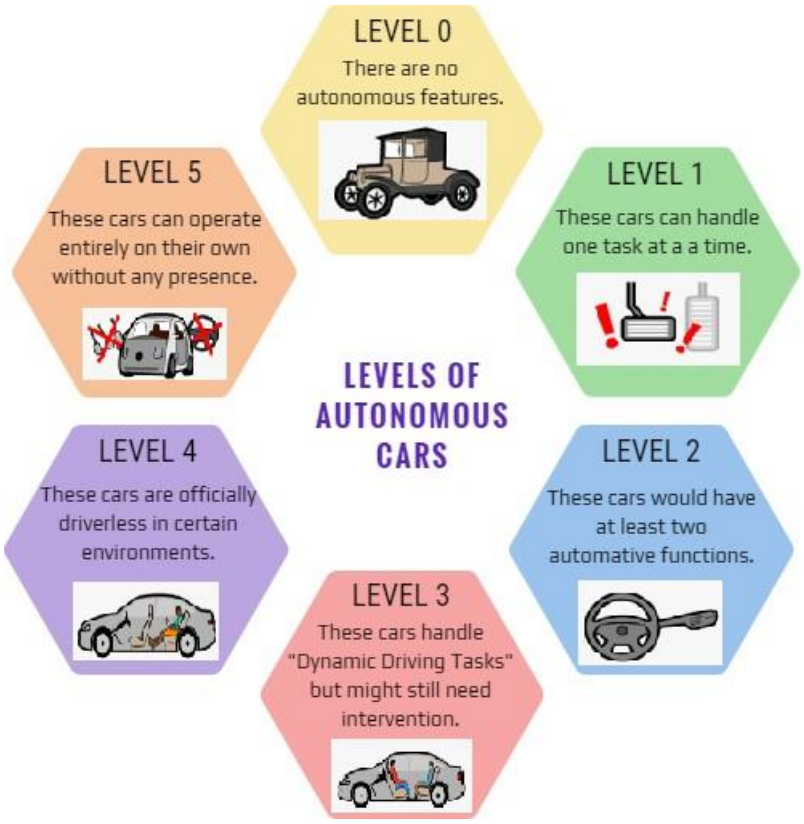


Figure 7: Autonomous Levels

These levels are very useful as with these we can keep track of what happens when we move from human-driven cars to fully automated ones. This transition will have enormous consequences for our lives, our work, and our future travels. As autonomous driving options are widespread, the most advanced detection, vision, and control technologies allow cars to detect and monitor all objects around the car, relying on real-time object measurements.

In the 5th level, it is used in the form of full autonomous. It has possibilities that can provide serious convenience to our lives in this transition. Autonomous driving options are common, so cars are the most advanced sensing, vision, and control technologies to detect and track all objects around the car based on real time object measurements. Vehicles perform autonomous driving with a multi-disciplinary engineering approach.

If a computer decides instead of us, we do not control the processes. Every computer and program may have a back door, and the question arising is what can be done if someone enters the computer that can save our lives.

2.6.2. Methods for Detecting Objects

Object Detection can be defined as identifying the object with boundaries. To be able to detect objects, there are some well-known algorithms and techniques. In this subsection, we analyzed these solutions and aimed to decide algorithm that we will use in our project. For all algorithms, strengths and weaknesses table is stated. The algorithms that we analyzed are stated in following figure.[29]

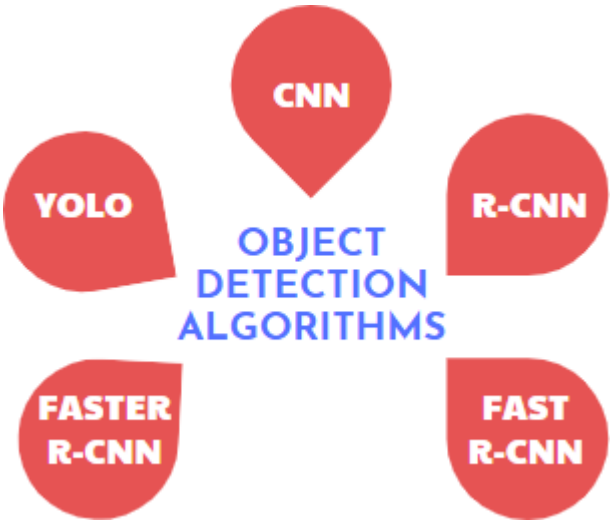


Figure 8: Object Detection Algorithms

2.6.2.1. CNN

CNN which is known as Convolutional Neural Network object detection algorithm is used for image recognition. In this method, we select different regions from different parts of an image and try to classify the presence of an object in that specific region. However, CNN algorithm can choose limited number of regions.

Therefore, since number of regions can be huge, this algorithm fails. That is why, alternative solutions are developed. [30] In following figure, strengths and weaknesses of CNN algorithm is given.

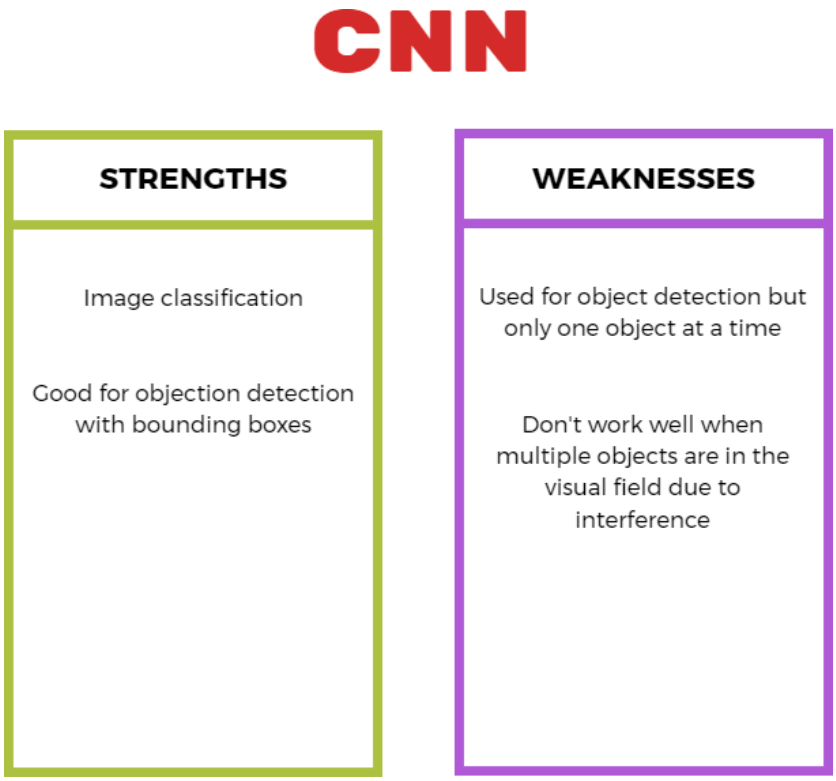


Figure 9: Strengths and Weaknesses of CNN

2.6.2.2. R-CNN

To discard the problem of choosing many regions, R-CNN which refers to Region Based Convolutional Neural Network uses selective search to indicate candidate regions. Therefore, it searches for possible target areas. After 2000 regions are indicated, each of them uses CNN separately and their bounding boxes and classes will be predicted. Since in selective search the number of 2000 is a fixed number, it makes algorithm slower. In selective search, algorithm first receives images as an input, then, it creates sub-segmentations which means we have multiple

regions in image. Finally, it gathers similar regions together. These regions refer to final locations of objects.

In following figure, strengths and other weaknesses of CNN algorithm is given.[31]

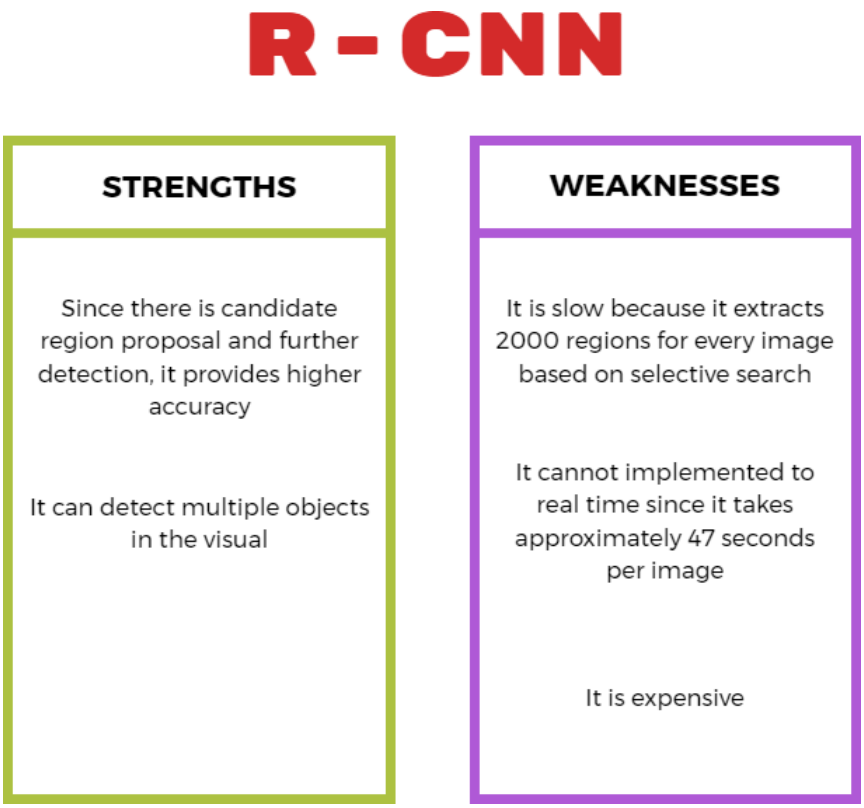


Figure 10: Strengths and Weaknesses of R-CNN

2.6.2.3. FAST R-CNN

As it is mentioned in its name, Fast R-CNN is the faster algorithm than R-CNN. It is because we do not have to feed 2000 regions each time. Instead, a CNN (convolution operation) is applied to an entire image and feature map is generated from it. After feature map is generated, ROI (Region of Interest) is created using selective search. For each object detection, ROI extracts a fixed-length feature vector from feature map.[32] Using this ROI feature vector, algorithm uses

softmax function to predict the class of region and identifying bounding box's values.[30] In following figure, strengths and other weaknesses of CNN algorithm is given.

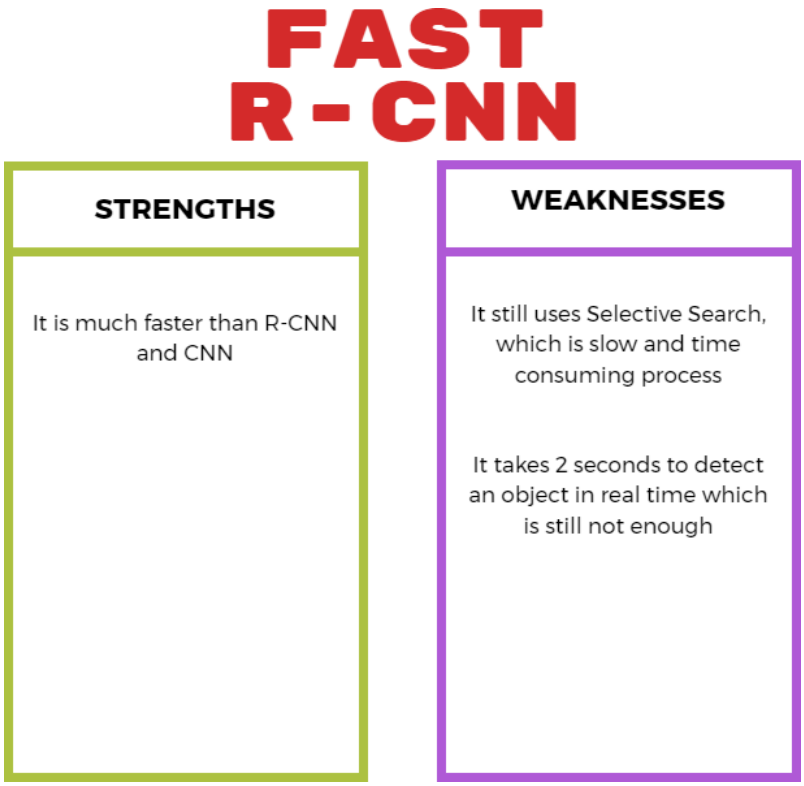


Figure 11: Strengths and Weaknesses of Fast R-CNN

2.6.2.4. FASTER R-CNN

Faster R-CNN is faster than Fast R-CNN, because it does not use selective search anymore. Instead of doing that, it uses Region Proposal Network which provides feature map directly. Therefore, we do not have to search for feature map any longer. In this way, algorithm works much faster.[30] In following figure, both strengths and weaknesses of Faster R-CNN algorithm are stated.

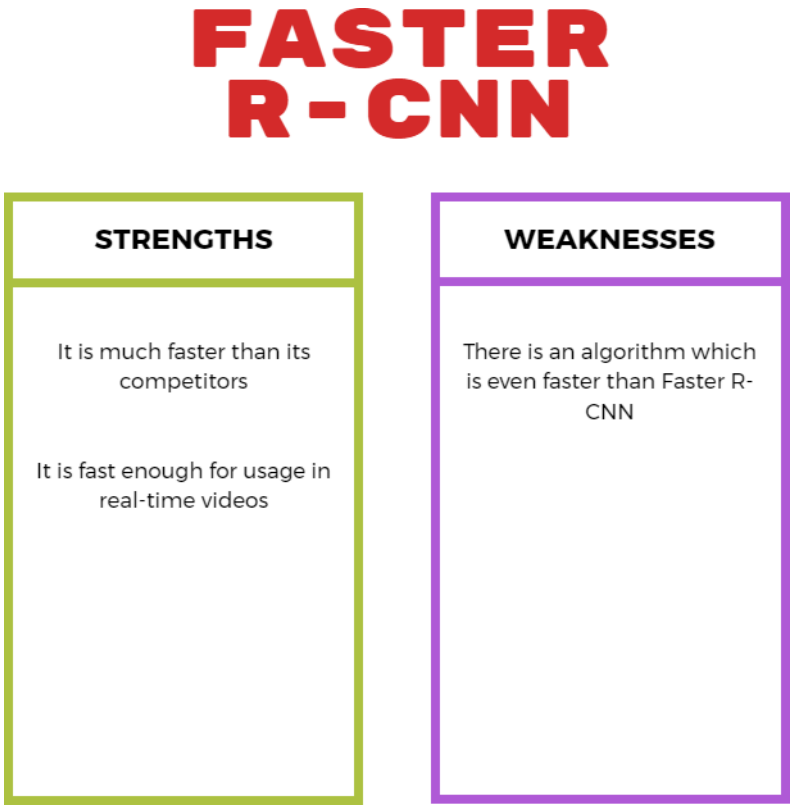


Figure 12: Strengths and Weaknesses of Faster R-CNN

2.6.2.5. YOLO

YOLO (You Look Only Once) algorithm is very different from previous techniques. It is trained to implement bounding box prediction and classification at the same time. That is why, YOLO algorithm is even faster than Faster R-CNN algorithm. However, YOLO has similarity with previous algorithms as well. They all use bounding box regression. We stated that YOLO is fastest algorithm, yet, it has some drawbacks as well. If objects are close to each other YOLO may fail because each grid cannot have more than 2 bounding boxes. In addition, it has trouble with detecting small objects due to spatial limitations.[31]

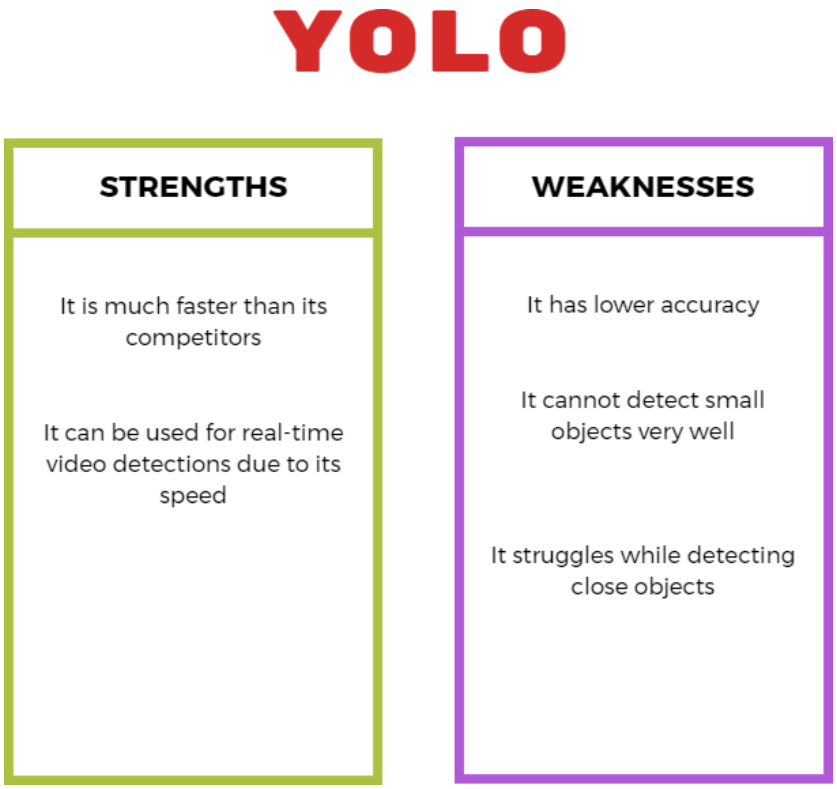


Figure 13: Strengths and Weaknesses of YOLO

When we examined all these algorithms, we had to decide which one is more important for us, speed, or accuracy. If we choose YOLO, we may have trouble with wrong or missing detections, and it would cause fatal problems. On the other hand, algorithm must work as much as fast since commanders must be warned as soon as possible. Therefore, we will look for best decision considering these issues. [33]

2.6.3. Comparison of Algorithms and Techniques

2.6.3.1. Comparison of Model Prediction Speeds

MODEL	TIME TO PREDICT SINGLE IMAGE
VGG16	47s + region proposal time
Fast R-CNN	.3s + region proposal time
SimpleNet	.09s
Yolo	**.0222s**

Figure 14: Prediction Speeds for Deep Object Detectors

In above table, Models are compared in terms of prediction speed. It can be observed that YOLO architecture predicts objects as the fastest.[34]

In above table, the VGG stands for the Visual Geometry Group of the VGG16 algorithm, and the value 16 in it means that there are 16 layers in total in the algorithm. There are 138 million parameters in this network. There is also SimpleNet which is an algorithm consisting of 13 layers using the CNN algorithm. In addition to above algorithms, there are other alternatives as well like MobileNets, SSD, etc. MobileNets are used to present an efficient class of vision models in embedded and mobile applications. It creates efficient parameters between latency and accuracy. They can be built lots of types such as embedding, recognizing, and classification. There is also SSD (Single-shot Detector) algorithm which predicts objects and find locations of them. At the end of the algorithm, we will receive probability of being object, location of object with coordinates, and width & height of bounding box as outputs. This algorithm is like YOLO since both find objects in single shot as it can be seen from how its named. Therefore, it can be stated that region proposal algorithms have better accuracy but slower run time. On the other hand, SSD and YOLO algorithms have good accuracy with faster run time.

2.6.3.2. Comparison of Sensors Under Different Weather Conditions

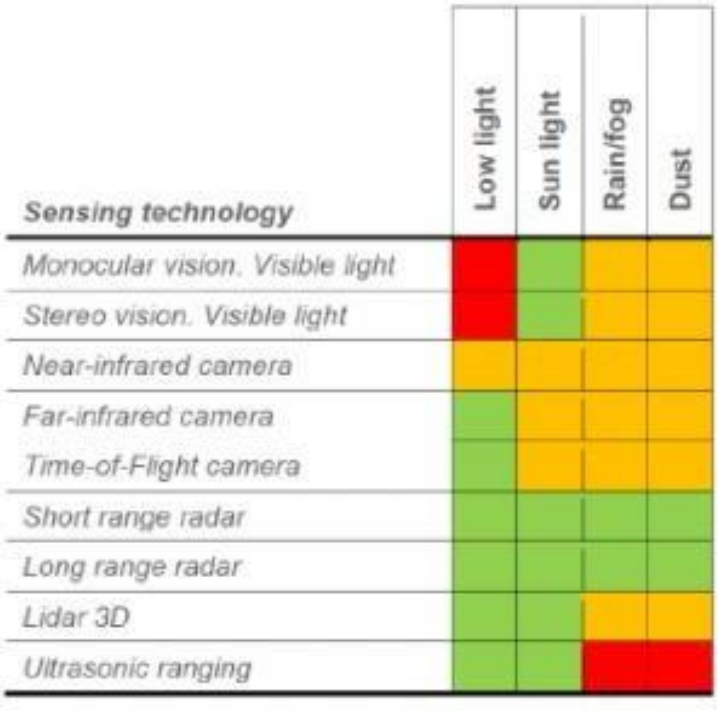


Figure 15: Sensors with different weather conditions

In above table, it is stated that each sensor acts differently based on weather conditions. Since in our project, we are aiming to use our vehicle in all seasons, we will be careful while choosing our sensor. In the table, red color refers to Bad, yellow color refers to Average, and green color refers to good results.[35]

2.6.3.3. Comparison between YOLO and other Object Detection Algorithms

In a review, researchers shared the results of object detection algorithms performance. The first related table is given at following figure.[36]

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

Figure 16: Comparison between Object Detection Algorithms

As it can be observed from above table, YOLO provides less accuracy compared to Fast R-CNN or Faster R-CNN algorithms. In addition, according to FPS values, we can state that YOLO algorithm works much faster than Faster R-CNN algorithm. There is another chart that supports above table:

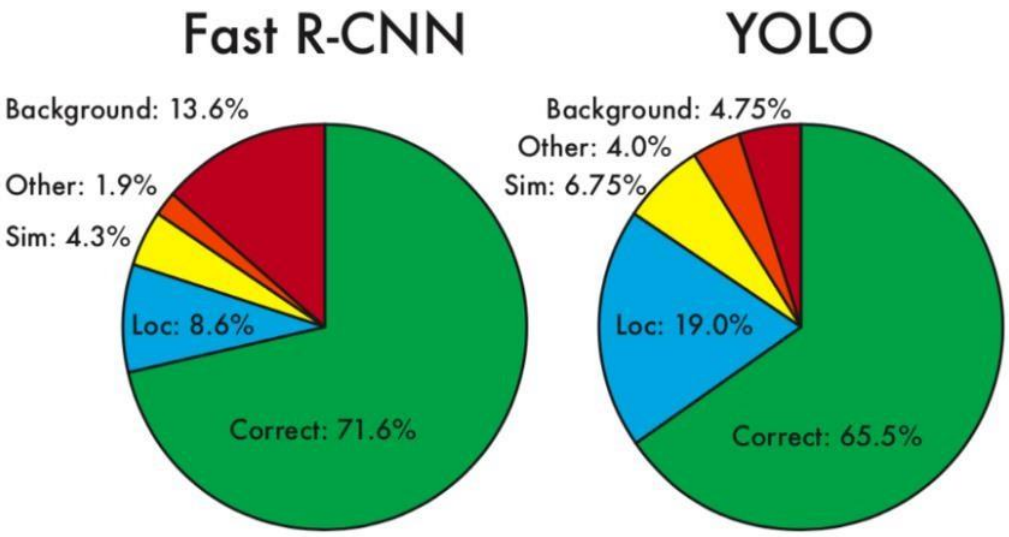


Figure 17: Accuracy Comparison between Fast R-CNN and YOLO

In above chart, it can be observed that Fast R-CNN gives more accurate results compared to YOLO algorithm. Therefore, when it comes to speed, YOLO algorithm is better. On the other hand, when it comes to accuracy, CNN methods work better. Hence, their combination gives good results as is shown in below figure.

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	**79.8**	87.7	49.6	74.9	52.1	86.0	81.7	83.3	**81.8**	**48.6**	**73.5**	59.4	79.9	65.7
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [28]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEP_ENS_COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [29]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [33]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

Figure 18: Combination of YOLO and Fast R-CNN

From above table, it is stated that YOLO itself is less successful than combination of Fast R-CNN and YOLO. We will choose the most appropriate technique by considering these tables and values.

2.7. Previous Works and Articles

[34] Gene Lewis. *Object Detection for Autonomous Vehicles,* pages 4-5, Stanford, CA, 2016.
This paper focuses on accuracy of object detection algorithms and comparison between them. It overviews which algorithm gives the best results while making predictions. It also reviews architecture, training algorithms, results of time prediction between models. As the article stated, YOLO model gives the fastest predict time among most popular models. We examined the time prediction results between models using this article and it will guide us while choosing our model.

[23] Bunyo Okumura, Michael R. James, Yusuke Kanzawa, Matthew Derry, Katsuhiro Sakai, Tomoki Nishi, Danil Prokhorov. *Challenges in Perception and Decision Making for Intelligent Automotive vehicles,* pages 1-12, Japan, 2016.
This paper overviews the challenges of perception for autonomous vehicles. It focuses on the difficulty while detecting objects and explained possible solutions. For that purpose, article explained that we can use classifiers for high-level decision making. It also examines classification successes under different road circumstances.

[24] Robert Geirhos, David H.J. Janssen, Heiko H. Schütt, Jonas Rauber, Matthias Bethge, and Felix A. Wichmann. *Comparing Deep Neural Network Against Humans Object Recognition When the Signal gets Weaker,* pages 15-17, Germany, 2018.
This paper focuses on a comparison between human-level classification performance with computer vision algorithms that refers to DNN. Additionally, in the paper, there are some experiments that compares well-known DNN algorithms like GoogLeNet, VGG-16 and AlexNet in each other. As a result of these comparisons writers decided that these 3 algorithms could be successful models for human feedforward visual detections. We observed that reaction of most popular models when different experiments are implemented. Therefore, we will be careful about the environment when choosing our model and algorithms.

[34] Enrique Marti, Miguel Angel de Miguel, Fernando Garcia, Member, IEEE, Joshue Perez. *A Review of Sensor Technologies for Perception in Automated Driving,* Pages 4, Spania, 2019.
This paper focuses on sensors which are used in automanous vehicles. In the paper, each sensor is explained in detail. Article presents their sensing setup as well. Additionally, article higlights that each sensor has its own weakness and strenght. Therefore, there is no winner among these sensors. Thanks to this research, we are informed about most popular sensor technologies, and it guided us regarding behaviour of sensors under different weather conditions.

[29] RuiJingXiaoQu, SongMen, WenLing, ZheJiang. *Comparison between Convolutional Neural Network and YOLO in image idendification,* Pages 1-3, China, 2019.
This article summarizes what are the most preferable object detection algorithm and it highlights their strengths and weaknesses. It seperates CNN and YOLO family and overviews each algorithm. Therefore, it also focuses on what are the difference between CNN and YOLO methods. We used this article while searching for algorithm to apply while detecting objects. Therefore, it guided us regarding our system's detection techniques.

[31] Danyang Cao, Zhixin Chen and Lei Gao. *An improved object detection algorithm based on multi-scaled and deformable convolutional neural networks,* Pages 1-4, China, 2020.
This article overviews what are the most popular object detection algorithms, what are their strengths and weaknesses and how they differ in each other. Therefore, it analyses the methods and compare them based on many features. This article guided us about what are advantages of R-CNN object detection algorithm other than enabled for multiple object detection simultaneously. We learnt that R-CNN also provides more accurate and reliable detections when compared to the other popular algorithms like YOLO and SSD. Therefore, we will be considering about this feature while developing detection part.

[36] Allen Wu. *Note for YOLOv1,* All Pages, China, 2018.

In this research, comparison of most popular object detection techniques is given. In paper, YOLO algorithm is summarized, and experimental comparisons are stated. According to paper, YOLO is appropriate choice in general cases. Comparisons are mainly based on accuracy speed, which are the most important factors for us. Therefore, thanks to this research, we observed different algorithms' performance and decide regarding our future solutions. Since both speed and accuracy factors are very significant for us, we will consider the comparison charts in our development process.

2.8. Summary

2.8.1. Technology Used

In Solutions for Self-Driving section, we explained there are 6 different levels of autonomous vehicles. In this section, we analyzed how exactly level 6 differs from remaining other levels.

In Solutions for Detecting Objects, we examined 5 types of algorithms. We examined the CNN object detection algorithm for image recognition. The CNN algorithm examines a limited number of specified regions. Using selective search, R-CNN selects candidate regions from 2000 regions at a time and collects similar regions together. With these regions it refers to the final positions of objects. Fast R-CNN applies a CNN to the entire image and generates a feature map from it. Then, it creates an ROI using selective search. Algorithm uses softmax function to predict the class of region and identifying bounding box's values using this ROI feature vector. Faster R-CNN uses Region Proposal Network instead of selective search. This makes the algorithm run faster. YOLO is the fastest running algorithm among these algorithms. This algorithm is trained to implement bounding box prediction and classification at the same time. They all use bounding box regression. These algorithms have advantages as well as disadvantages. We explained them in detail in the Solutions for Detecting Object section and compared them with each other.

2.9. Conclusion

In this literature review, we made research about Autonomous Vehicles as UGVs and summarize what we learnt and observed. This review does not contain the list of best algorithms or techniques, yet, it has comparisons between algorithms, definition of techniques and advantages, disadvantages tables, etc. While comparing these techniques and solutions, we observed that there are weaknesses and strengths of each algorithm since performance may depend on weather conditions, speed, computing power, sensor type, etc. Additionally, we looked deeper into the samples of autonomous vehicles, therefore, we analysed what other developers used and what kind of products come as a result. According to all of this information, we will choose techniques to develop our own system.

We'd like to state that, we are not going to develop a system which only autonomously move, we also add subsystems to our vehicle which enables our vehicle to make reconnaissance and surveillance around critical facilities. For that purpose, we are aiming to add a camera system which can continuously send real-time video to receiver. We are going to implement detection algorithms to that video and identify risky events. Even more, this vehicle can be also used for spying the enemy in battlefield. In this way, performance in battlefield or around a critical facility can be monitored or plans may be updated by commanders without risking any soldier's life. Additionally, when there is an emergency or necessary situation, commanders will be able to control the vehicle using controllers. Therefore, our vehicle can be operated either manual or autonomous mode. In conclusion, our system will be responsible of two main parts:

- Vehicle will be autonomously and manually operated.
- Suspicious events will be detected, and commander will be informed about these events immediately.

With this project, we are determined and excited to develop a national unmanned ground vehicle. With the ability to patrol autonomously, our soldiers will be safer.

References

[1] First Autonomous Vehicle Sample, “The History of Self-driving Cars” [Online].

Available:

https://www.digitaltrends.com/cars/history-of-self-driving-cars-milestones/

[Accessed 25 October 2021].

[2] Idea of First Autonomous Car, “The History of Self-driving Cars” [Online].

Available:

https://www.titlemax.com/resources/history-of-the-autonomous-car/

[Accessed 25 October 2021].

[3] Process of Autonomous Vehicles, “The History of Self-driving Cars” [Online].

https://en.wikipedia.org/wiki/History_of_self-driving_cars#1920s

[Accessed 25 October 2021].

[4] Firebird I, “Development Process of Firebird Series” [Online].

https://www.gmheritagecenter.com/gm-vehicle-collection/1954_Firebird_I.html

[Accessed 25 October 2021].

[5] Firebird II Concept Car, “Future of Technology” [Online].
https://www.businessinsider.com/gm-1956-self-driving-car-video-2015-10

[Accessed 25 October 2021].

[6] Study about Autonomous Vehicles in 1958, “Driving the Future” [Online].
Available:
https://www.sfchronicle.com/drivingthefuture/timeline/
[Accessed 25 October 2021]

[7] Article “Computer Controlled Cars” [Online]. Available:
http://jmc.stanford.edu/commentary/progress/cars.pdf
[Accessed 28 October 2021].

[8] Self-Driving Car VaMoRs “Speed Increasement of Vehicle” [Online]. Available:
https://www.lifehacker.com.au/2018/12/today-i-discovered-the-self-driving-car-trials-of-the-1980s/
[Accessed 28 October 2021].

[9] DARPA Grand Challenge “Autonomous-Vehicle Race” [Online]. Available:
https://en.wikipedia.org/wiki/DARPA_Grand_Challenge
[Accessed 29 October 2021].

[10] Google’s Self-Driving Car “Waymo Series” [Online]. Available:
https://waymo.com/waymo-driver/
[Accessed 29 October 2021].

[11] Otto Self-Driving Vehicle “Otto Autonomous Semi-Truck” [Online]. Available:
https://en.wikipedia.org/wiki/Ottomotto
[Accessed 29 October 2021].

[12] Tesla Autonomous Vehicles “Fully Autonomous Cars” [Online]. Available:
https://www.tesla.com/autopilot
[Accessed 29 October 2021].

[13] Tarantula UGV “Tarantula UGV Qualities” [Online]. Available:
https://polygonjournal.com/2018/04/06/turkish-company-presents-new-armed-ugv-tarantula/
[Accessed 30 October 2021].

[14] Tarantula UGV Features “Dimensions of Tarantula” [Online]. Available:
https://www.army-technology.com/projects/cobra/
[Accessed 30 October 2021].

[15] ACROB UGV “ACROB Mini UGV Qualities” [Online]. Available:
https://www.armyrecognition.com/weapons_defence_industry_military_technology_uk/acrob_small_ugv_to_enter_into_service_with_turkish_armed_forces.html
[Accessed 30 October 2021].

[16] O-İKA 2 Project Scope “O-İKA 2 UGV Qualities” [Online]. Available:
https://www.raillynews.com/2020/11/time-to-start-middle-class-unmanned-ground-vehicle-o-ika-2-project/
[Accessed 30 October 2021].

[17] BARKAN UGV “Unmanned land vehicle BARKAN” [Online]. Available:
https://www.dailysabah.com/business/defense/havelsans-upcoming-focus-to-be-digital-troops-general-manager-says
[Accessed 30 October 2021].

[18] ALPAN UGV “Unmanned land vehicle ALPAN” [Online]. Available:
https://www.oguzkagansavunma.com/en/alpan.php
[Accessed 30 October 2021].

[19] BARKAN UGV “Physical Features of BARKAN” [Online]. Available:
https://www.havelsan.com.tr/sektorler/egitim-ve-simulasyon/robotik-ve-otonom-sistemler/havelsan-barkan
[Accessed 30 October 2021].

[20] URAN-9 UCGV “Uran-9 Unmanned Ground Combat Vehicle ” [Online]. Available:
https://www.army-technology.com/projects/uran-9-unmanned-ground-combat-vehicle/
[Accessed 30 October 2021].

[21] The Gladiator TUGV “Gladiator Tactical Unmanned Ground Vehicle” [Online]. Available:
https://military.wikia.org/wiki/Gladiator_Tactical_Unmanned_Ground_Vehicle
[Accessed 31 October 2021].

[22] Miloš UGV “Miloš Unmanned Ground Vehicle” [Online]. Available:
https://www.army-technology.com/projects/milos-unmanned-ground-vehicle/
[Accessed 31 October 2021].

[23] Article about Challenges in Perception “Challenges in perception and decision making for intelligent automotive vehicles” [Online]. Available:
https://www.researchgate.net/publication/299997309_Challenges_in_Perception_and_Decision_Making_for_Intelligent_Automotive_Vehicles_A_Case_Study
[Accessed 31 October 2021].

[24] Article Humans vs DNN “Comparison of Computer vision and human vision” [Online]. Available:
https://arxiv.org/pdf/1706.06969.pdf
[Accessed 31 October 2021].

[25] Pros and Cons of Sensors “RADAR vs. LIDAR vs. CAMERA” [Online]. Available: https://www.techbriefs.com/component/content/article/tb/stories/blog/37699 [Accessed 31 October 2021].

[26] Camera Sensor “Advantages and disadvantages of sensors” [Online]. Available: https://en.wikipedia.org/wiki/Lidar [Accessed 31 October 2021].

[27] Definition of LIDAR “What is LIDAR Technology” [Online]. Available: https://autocrypt.io/camera-radar-lidar-comparison-three-types-of-sensors/ [Accessed 31 October 2021].

[28] LIDAR vs Camera “Advantages and disadvantages of LIDAR over Camera” [Online]. Available: https://leddartech.com/lidar-radar-camera-demystifying-adas-ad-technology-mix/ [Accessed 31 October 2021].

[29] Article about CNN and YOLO Family “Comparison between Convolutional Neural Network and YOLO in image identification” [Online]. Available: https://www.researchgate.net/publication/339685696_A_survey_Comparison_between_Convolutional_Neural_Network_and_YOLO_in_image_identification [Accessed 1 November 2021].

[30] Object Detection Algorithms “CNN, R-CNN, YOLO Techniques” [Online]. Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e [Accessed 1 November 2021].

[31] Literature about R-CNN, CNN and YOLO "Object Detection with Neural Networks" [Online]. Available: https://hcis-journal.springeropen.com/articles/10.1186/s13673-020-00219-9 [Accessed 1 November 2021].

[32] Fast R-CNN Algorithm "How Fast R-CNN and R-CNN differ?" [Online]. Available: https://towardsdatascience.com/computer-vision-a-journey-from-cnn-to-mask-r-cnn-and-yolo-1d141eba6e04 [Accessed 1 November 2021].

[33] YOLO Algorithm "Pros and Cons of YOLO Algorithm" [Online]. Available: https://www.geeksforgeeks.org/yolo-you-only-look-once-real-time-object-detection/ [Accessed 1 November 2021].

[34] Literature about Autonomous Vehicles "Object Detection for Autonomous Vehicles" [Online]. Available: https://web.stanford.edu/class/cs231a/prev_projects_2016/object-detection-autonomous.pdf [Accessed 25 October 2021].

[35] Article about comparison of sensors "A Review of Sensor Technologies for Perception in Automated Driving" [Online]. Available: https://www.researchgate.net/publication/336010399_A_Review_of_Sensor_Technologies_for_Perception_in_Automated_Driving [Accessed 31 October 2021].

[36] Article about comparison between Object Detection Methods "Note for YOLOV1" [Online]. Available: https://medium.com/@arashilen/note-for-yolo-v1-98e00c4915e5 [Accessed 1 November 2021].

3.SOFTWARE REQUIREMENT SPECIFICATION

3.1. Introduction

Autonomous vehicles are automobiles that are capable of self-driving which means there is no need for any human intervention since vehicle detects road and surrounding objects with the help of control systems. These systems include sensors such as ultrasonic, LIDAR, RADAR, etc. Hence, it uses artificial intelligence augmented systems to be able to give the low-level decisions. Additionally, it uses object detection algorithms to be able to avoid them and operate tasks using its camera which can continuously send a video. Even more, these systems are started to be used in defence industries frequently. We believe that such systems will be even more common in the future since it does not require any human power and protects humans from risky situations. There are multiple samples of autonomous cars which have different usage purposes. Our project is going to be used for surveillance and to make reconnaissance. Additionally, it can be used as a spy since it has a capability to transmit real time stream to commander's monitor. In this section of the report, we explained our purpose, scope of our project, intended audience features, glossary and finally overview of our SRS document.

3.1.1. Purpose of This Document

In this project, our goal is to develop a system that supplies security by using unmanned ground vehicle. This unmanned ground vehicle will keep our soldiers safe and reduce soldiers' strength. Moreover, it detects dangers such as guns, suspicious persons, and unexpected situations using timing, sensors, and other factors. It provides instant information when it observes hazards. In this way, we plan to prevent dangers faster, and to reduce the loss of life. While doing the project, we design to provide diverse software requirements such as Artificial Intelligence, Deep Learning, Machine Learning, and Object Detection for unmanned ground vehicle. In the continuation of the project, we aim to test this unmanned ground vehicle by simultaneously completing the hardware of the vehicle. We aim to determine the vision and detection with the data coming from the distance, sound, and camera systems on the autonomous vehicle, and to bring the received data to the best possible solution by combining these sensors. While the vehicle is driving in a certain area, it will detect objects by object detection and report situations that it deems dangerous. In this area, it will drive thanks to artificial intelligence. This SRS document contains the project requirements, and Software Requirements Specification for autonomous vehicle task.

3.1.2. Intended Audience and Reading Suggestions

This document is Software Requirement Specification report, which is intended for domain experts, software/computer engineers, developers, testers, and project managers. Before reading this document, it is recommended to read our Literature Review Report to understand which concepts our project includes, possible algorithms and techniques. It will provide an overview of our product. This report focuses on requirement specification of the project and overall description of the system.

3.1.3. Scope of the Project

In our project, it is aimed to develop national unmanned ground vehicle which acts as a spy or guardian of a field. Therefore, our project will not be only self-driving car, it will also be able to detect objects which can be threats, enemies, suspicious objects, or events. As soon as vehicle detects suspicious event, it will warn commanders via its application. As a warning method, vehicle will inform commander with time and distance data as well. In this way, upcoming attacks can be prevented. Also, our vehicle is going to be operated in either manual or autonomous mode. Therefore, commanders can easily direct it to battlefield or critical facility for observation without risking any human being's life. Since it can be used via remote control, one of its usage purposes is spying the enemy field. To handle these implementations, we will combine multiple engineering disciplines. Since our car should be moving both autonomous and manual, we will get to know which board or sensor to use. At this part of development process, hardware tools are used frequently. For object detection and self-driving part, we will use Machine Learning and Artificial Intelligence concepts to implement detecting and self-driving algorithms. In both part of our project, we will be using a combination of hardware and software.

Our system will include:

- Object Detection for upcoming threats
- Object Detection for spying
- Capability of autonomous and manual driving

3.1.4. Glossary (Definitions, Acronyms, and Abbreviations)

TERM	DEFINITION
Actor	An actor can be a user, or another software system that interacts with the system.
Android	A name of the mobile operating system made by American company; Google.
C/C++	C and C++ high-level and general-purpose programming language. C is a structured programming language while C++ is object-oriented programming language.
Python	Python is another high-level programming language which is preferable when Artificial Intelligence algorithms is used.
IOS	A mobile operating system created and developed by Apple.
Software Requirement Specification (SRS)	A document that provides comprehensive description of system's functions, requirements, constraints, and conditions to be able to perform. This document is an SRS document.
User	Users will be students who are participated to competition using our online platform and they will find solutions for given problem as teams.
Autonomous	Autonomous means independent. In our project, we call our system autonomous vehicle since it does not need any human intervention to be able to move.
Object Detection	Object Detection is a popular computer science technology. By using vision techniques, computer decides and identifies what the object is.

UGV	UGV refers to Unmanned Ground Vehicle. It is a vehicle that can move on the ground by itself. There are various of UGVs used for different purposes. In this project, we designed our own UGV which will for surveillance purposes.
GUI application	GUI refers to Graphical User Interface. It provides an interface which user can interact with existing software. In our project, we are planning to implement a GUI application which visualizes operations of our system.
Arduino	Arduino is an open-source micro-controller board. We use Arduino board to upload UGV's movement-related code.
Arduino IDE	Arduino board must be programmed via its IDE which called Arduino IDE. Therefore, Arduino IDE is an Integrated Development Environment for Arduino boards.
Spyder IDE	We use Python programming language for many operations in our system. Spyder IDE is an open-source development platform for Python programming language.
Local Area Network (LAN)	Local Area Network is a computer network which can be used within a limited area.

3.1.5. References

We used IEEE Std 830™-1998(R2009) Recommended Practice for Software Requirements Specifications.

3.1.6. Overview of Document

The document consists of 3 main titles.

- The first title gives information about the document, such as the articles used as references and the terms used in the article.
- The second title introduces the system. Function properties used, user classes Characteristics and requirements for creating the system.
- The last title gives a detailed introduction to the requirements, such as the interface requirements used and the explanation of the requirements in detail.

3.2. Overall Description

This part will clarify of the principal aspect of Autonomous Vehicles system and necessities.

3.2.1. Product Perspective

Our system includes multiple hardware and software subsystems. Since we are aiming to introduce our system as a product, we used multiple hardware tools as well. On the other hand, our project includes two main computer science concepts: detection algorithms and autonomous systems. For these techniques, we will be using different programming languages such as Python, Java, and C/C++. In addition, there different tools and IDEs that is used for development process. In this section, we summarized that what features are included as hardware and software. Since our project is a vehicle, we used 4 wheels, 4 motors for each wheel, AA battery enclosure and screws.[1] We also used L298N Motor module for motors and wires to handle connections. As a software development IDE, we use Arduino and Spyder. We use Arduino Uno board to be able to run our code on the vehicle. Vehicle's moving parts mostly implemented in Arduino board. For object detection, we use ESP-32 CAM board with FTDI programmer to access our local network to be able to send video continuously. The detailed explanation of these boards and other hardware products will be given at *3.1.2. Hardware Interfaces* section. Vehicle can operate in both manual and autonomous modes.

For both modes, we used HC-SR04 Ultrasonic Sensor to avoid objects on the road. We assembled a HC-06 Bluetooth module which provides wireless serial communication. When this module assembled with our vehicle, it can be operated with controllers if our vehicle is in manual mode. For autonomous mode, we use servo motor as well. To monitor detections and real time video, we create an application that can be accessed using the same Wi-fi or Internet connection. Commanders or Controller soldiers are also able to control vehicle through this app with buttons.

In below table, we stated what are the features of our vehicle, summarized explanation, and which tools we will use as software and hardware.

Task	Operation Definition	Software Tools	Hardware tools
Manuel Driving	Vehicle shall operate by a controller.	• Python • C/C++ • Arduino IDE • Spyder IDE	• Servo Motor • 4 wheels, motors, and chassis • L298N Motor Module • HC-SR04 Ultrasonic Sensor • Arduino Uno board
Autonomous Driving	Vehicle shall operate by itself. There will be no human intervention.	• Python • C/C++ • Arduino IDE • Spyder IDE	• 4 wheels, motors, and chassis • L298N Motor Module • HC-SR04 Ultrasonic Sensor • Arduino Uno board
Guardian mode detection	Vehicle shall detect objects that can be threats. If there is any, commander will be informed.	• Python • C/C++ • Arduino IDE • Spyder IDE	• ESP32 Camera Module • OV2660 Camera • FTDI Programmer

Spy mode detection	Vehicle shall detect and count enemies in an enemy battlefield for surveillance.	• Python • C/C++ • Arduino IDE • Spyder IDE	• ESP32 Camera Module • OV2660 Camera • FTDI Programmer

3.2.2. Product Functions

Detection of Moving and Still Objects: The system detects the silhouette and the person using its own algorithm. Thanks to this algorithm, it can distinguish moving and still objects. The vehicle gives the moving objects as feedback to the system.

Environmental monitoring: Thanks to the hardware and software on the vehicle, it will be able to autonomously monitor and control the environment 24/7. When an obstacle or dangerous situation is detected, the monitoring is decided accordingly.

Object Tracking: The system has hardware and software to track all objects. The system has an algorithm that can tag harmful and harmless objects. It will reflect harmful or potentially harmful objects on the screen in red labels.

Warning System: The system sends harmful or potentially harmful objects to the host computer as feedback. It also notifies it with sound and light warning systems to draw attention to its surroundings.

Route Determination: The system determines its route to scan its surroundings in 360°. When it detects harmful or potentially harmful objects during the route, it decides its route to follow these objects.

Detecting Harmful and Harmless Objects: The system can distinguish harmful or harmless objects within itself. For example, if a weapon-like object is detected in the hand of a person, it will label it as harmful and give a warning.

Vehicle Identification: After the system detects the vehicles, it detects how long the vehicles have stopped at that location thanks to the timer in it. It directly informs the system of foreign and large vehicles. It also gives a warning to the system on approaching vehicles.

3.2.3. User Classes and Characteristics

In this section, actors of our project are listed. There are 6 different actors in our project. Each actor's tasks are described.

3.2.3.1. Controller as a Soldier

Controller is a soldier or commander who is responsible of moving the vehicle manually.

Tasks of controller are given below:

- Controls the vehicle via remote control.
- Informs commander when detection warning arises.
- Observes enemy battlefield.

3.2.3.2. Defender&Attacker Team

Defender&Attacker Team refers to group of soldiers which shows up when there is an emergency.

Tasks of defender&attacker team are given below:

- When the detection warning received, team shall come to area and prevent overcoming attacks.
- When attack arises, team shall defend the facility.
- Team may attack enemy battlefield based on intelligence received by our vehicle.

3.2.3.3. Technician

Technician refers to an employee who is responsible of technical issues of the UGV.

Tasks of defender&attacker team are given below:

- Checks UGV's battery status.
- Updates batteries regularly.
- Provides service when there is a damage on UGV.

3.2.3.4. Commander

The commander is the authority person who makes the necessary plans on the battlefield and takes the responsibility of the soldiers.

Tasks of commander is given below:

- Updates plans or creates new plans based on performance monitoring on the battlefield or around a critical facility.
- Controlling the vehicle using controllers in case of an emergency or necessary situation.
- Commands military units.

3.2.3.5. Enemy as Human & Object

It refers to all kinds of dangerous, offensive, harmful objects or enemies on the battlefield.

Tasks of enemy as human and object are given below:

- Bring dangerous objects such as weapons, tanks, explosives to the battlefield.
- Harming vehicles, people, battlefield using hazardous materials.
- Being on the battlefield as enemy troops.
- Engaging in suspicious behavior.

3.2.3.6. UGV

Unmanned Ground Vehicle ensures the security of the battlefield.

Tasks of unmanned ground vehicle is given below:

- Drives unmanned on the battlefield.
- Detects objects, guns, and suspicious persons.
- Reports suspicious situations that have been there for longer than normal conditions.
- Being on the battlefield as enemy troops.
- Notifies when it sees a dangerous situation.

3.2.4. Operating Environment

The features of the vehicle must be as follows in order to fulfill the relevant functions:

- Power supply continuously so that it can be 24/7
- Camera, warning system, network connection and processor capacity to process images and make decisions
- Having powertrain and wheels to move

Related features will be tested in real time field and can be played on.

3.2.5. Constraints

This system moves by itself and detects objects in a certain area. It decides for itself whether there is a suspicious situation or not. If it decides on the suspicious situation, it will report it. This system is designed to ensure the safety of soldiers. For this reason, it is very important that it always works correctly. There are some cases where this system requires constraints.

- If the system cannot identify the objects in milliseconds, the system should give a message about the identification.
- The system may not be able to select objects clearly in a foggy environment, so it should give a warning.
- The system can give notifications for non-dangerous objects even though they appear dangerous. For example, it can detect a toy gun as if it were a real gun. In this case, the controller needs scrutiny.
- Since our autonomous-vehicle is operates a source-code, it can be hacked, and additional security issues can arise. To eliminate this situation, the control will be carried out using controllers. Thus, cyber-attacks will be prevented.
- Hardware components of the system may be damaged. In such a restriction, the system will give a warning. The system will act according to the commander's assessment.

- Camera must connect 2.4 GHz Wi-Fi instead of 5 GHz.

3.2.6. Dependencies and Assumptions

In this section, we listed what are the dependencies and assumptions our system has, and we stated brief explanation for each of them.

Environment Conditions: Our system will operate when environmental factors are regular. System may fail when there is a bad weather, holes, scope, etc. We also assume that path of UGV is separated with obstacles.

Operating System: The system will be supported by Windows 10 and GNU Linux. For mobile application where real time video will be monitored, system must be in Android 4.0.3. and upper versions' platforms. For IOS, the version must be IOS 7.0 as a minimum version. IOS and Android devices cover 99% of the smart phone market, therefore, other operating systems are ignored. Smart phone must have at least Bluetooth 2.0 protocol to perform control manually. Additionally, system must have minimum 2.4GHz communication frequency. Also, Wi-fi direct should be supported since camera sends video using LAN.

Hardware: There are several hardware modules which are required for operation of UGV. Bluetooth-Serial module board can be both HC06 and HC05. Arduino board type must be Uno(R3) or Nano. There must be exactly 4 wheels and motors which are wired to each wheel. For autonomous mode, L298N motor module must be used. Ultrasonic Sensor HC-SR04 must be assembled for avoiding objects in a road. In addition, any small servo motor will be enough for autonomous operation of vehicle. ESP32-CAM module must be also provided to send videos via Wi-fi. To be able to upload code, there must be either FTDI programmer or Arduino board. All these systems must be assembled appropriately to perform tasks.

Software: To be able to build our programs, there must be Arduino IDE and compiler for other programming languages as well. It could be Visual Studio Code which supports most of the popular languages or Spyder IDE which is a preferable choice when it comes to Python. We assume compiler for necessary language is exists.

Range of Remote Control and Camera: To be able to control vehicle the distance between vehicle and controller must be at most 10 meters. On the other hand, ESP32 Camera can send videos from approximately 20 meters away.

Battery: We assume that system has 4 x AA batteries to supply Arduino IDE. There must be additional power supply for Camera which can be both power bank and battery. For FTDI Programmer, supplying voltage via 5V or 3.3V is a major necessity.

System Security: While continuously video sent from the ESP32 Camera, both smart phone and camera module will be connected to same IP address. Therefore, anyone who knows Wi-fi address and password can access to system and watch the video. That is why, password must not be easily brute forced or guessed.

3.3. Requirement Specification

3.3.1. Interface Requirements

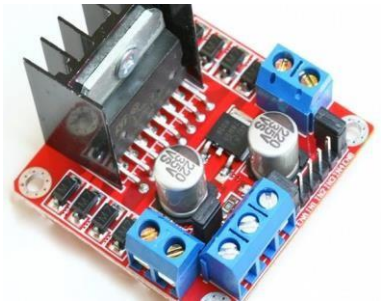
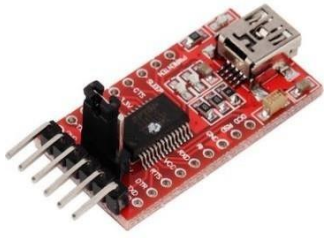
3.3.1.1. User Interfaces

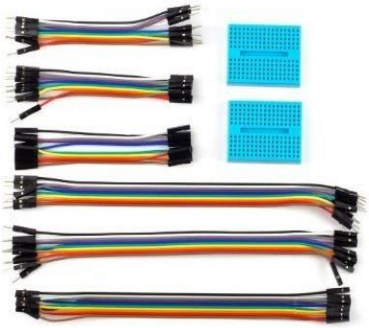
User interface of our application and system will be in English. Additionally, it will be understandable and easy to use. In the application, we will state a part which is separated for real time sent video. Also, both manual and self-driving tasks can be operated via our application. For manual driving, there will be arrows which represents movement directions of UGV. When a threat or enemy is detected, commander will be informed with a warning message that is displayed on application. In conclusion, user will be capable of operating vehicle and observe the outputs that is received from UGV.

3.3.1.2. Hardware Interfaces

In below table, we stated all hardware products that we used in our system. Their explanations and visualizations are also given.[2]

Figure	Name	Definition & Usage Purpose
	Arduino Uno Board	Arduino Uno is an open-source micro-controller board. We use Arduino Uno board to upload UGV's movement-related code. In board, self-driving and manual-driving code is embedded.
	Robotic Car Cit (4 Wheels, 4 motors, chassis, screws, and AA battery enclosure)	Robotic Car Cit is used to create and assemble our vehicle. It provides platform to add various sensors and controllers like Arduino Uno board, Bluetooth module, ultrasonic sensor, etc.
	HC-SR04 Ultrasonic Sensor	HC-SR04 is a module that can measure distance. Using HC-SR04 Ultrasonic sensor, our UGV will be capable of avoiding obstacles. Its range is between 2 cm and 400 cm. It has transmitter and receiver, and only 4 pins which makes it easy to assemble.[3]

	HC05/HC06 Bluetooth Module	HC06 Bluetooth module is used for wireless serial communication.[4] Device is capable of send serial data through pc or smart phones. Therefore, we use this module to control our vehicle manually via smartphone. Its range is up to 9 meters.
	L298N Motor driver	L298N is a motor driver which controls speed and direction using DC motors. Therefore, we use this driver to be able to move our car appropriately. This module can also control up to 4 dc motors, which is adequate for our system.
	FTDI Programmer	To be able to program ESP32-CAM, we use FTDI Programmer module which is USB to TTL Serial Converter. We also give power to our ESP32-CAM module using this FTDI programmer.

	ESP32-CAM module	ESP32-CAM is a Wi-Fi enabled module which can be used in IoT projects. Its range is between 15 and 20 meters. We used this module to get real time video via Wi-Fi. Therefore, this module will send video continuously and object detection algorithms will run on received video.
	Additional wires, breadboard, and connector cables	Since there are multiple boards, sensors and modules, there must be wires and cables to handle connections and give power to our UGV. Therefore, we used connector wires, cables, and breadboards.
	Power Supply	To make system work, power supplies must be provided. Therefore, we used 4x1.5V batteries and powerbanks to give power.

3.3.1.3. Software Interfaces

We will be developing the autonomous vehicle as a prototype. Therefore, we use Arduino Uno board on the vehicle. For this board, we use Arduino IDE version 8.1.12 in its software interface. Moreover, we use diverse software tools such as Python, Spyder IDE, and C/C++ programming languages. We downloaded Bluetooth RC Controller software from the Google Play Store for the Bluetooth system. Depending on the software we will develop, we can evolve this application in our own application.

3.3.1.4. Communication Interfaces

There are two methods that we used while communicating between our UGV and application.

1) Bluetooth
2) Wi-Fi

Using these two techniques, we control our vehicle manually and inform commanders when there is an emergency.

3.3.2. Detailed Description of Functional Requirements

In this part of the report, we stated functional requirements of the project.

3.3.2.1. Use Case Diagram

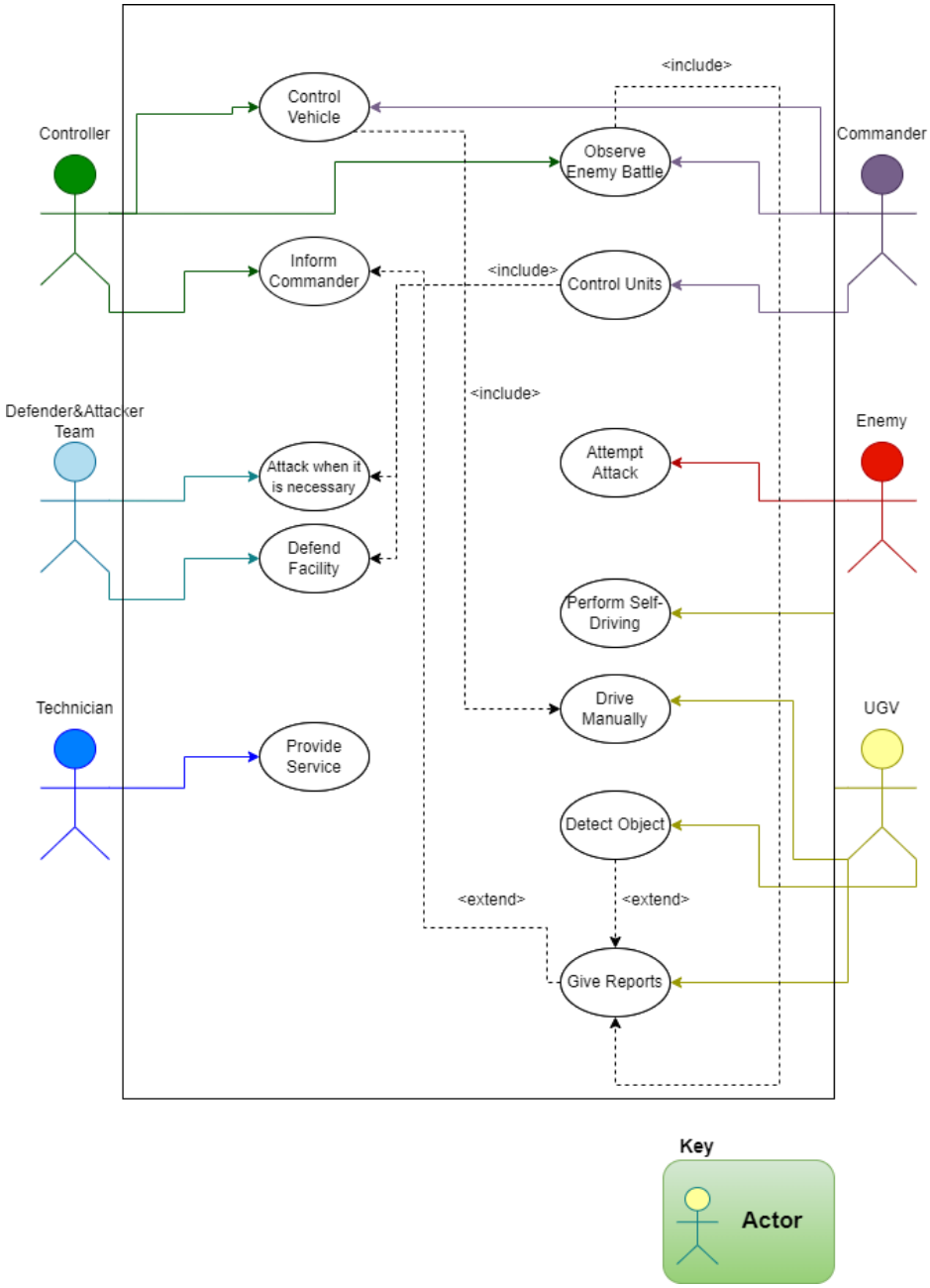


Figure 1: Use Case Diagram

3.3.2.2. Use Cases

In below table, all use cases are explained briefly. Detailed explanation is given as subsections for each use case.

Use Case Title	Description
Control Vehicle	Vehicle is controlled via Bluetooth by controller.
Inform Commander	Commander gets notification or warning about detected threats.
Attack when it is necessary	If facility is under attack, then team may attack back to enemy battlefield.
Defend Facility	Team defend facility when facility is under attack.
Provide Service	Technician is responsible for taking care of UGVs. S/He provides service when needed.
Observe Enemy & Enemy Battle	Commander or soldier observes enemy or enemy battlefield via sent video.
Control Units	Commander controls units and give commands when there is a threat.
Attempt Attack	Enemy may attempt to attack via gun, remote controller, or a bomb.
Perform Self-Driving	UGV moves by itself. There will be no human intervention.
Drive Manually	UGV is controlled by soldiers.
Detect Objects	UGV detects all objects around.
Give Reports	UGV reports when hazardous materials or events are detected among all objects.

3.3.2.2.1. Scenario 1: Control Vehicle

Use Case Number	1
Use Case Name	Control Vehicle
Summary	Vehicle is controlled via Bluetooth by controller.
Actor	Commander, Soldier, UGV
Trigger	Spy mode is activated, or commander intends to make reconnaissance.
Precondition	• Spy mode must be selected. • Remote Controller must be used by commander or soldier. • Power supplies of UGV must be provided appropriately. • HC06 Bluetooth module must be assembled appropriately. • Source code must be embedded in Arduino Uno board.
Scenario	1. Power supply must be provided to UGV. 2. Soldier or commander selects Spy mode. 3. Soldier or commander clicks on direction buttons on application.
Exceptional Situations & Alternative Flows	1. Application may not work. • Restart the application and report situation to technician. 2. Control buttons may do not work. • Check the HC05 Bluetooth connection. 3. Bluetooth connection may fail. • Check the HC05 Bluetooth connection and restart the module.
Postcondition	1. UGV starts moving.

3.3.2.2.2. Scenario 2: Inform Commander

Use Case Number	2
Use Case Name	Inform Commander
Summary	Commander gets notification or warning about detected threats.
Actor	UGV, Commander, Enemy
Trigger	Threat or enemy is detected.
Precondition	• Spy mode or guardian must be selected. • Power supplies of UGV must be provided appropriately. • UGV must have enemy or object around of it. • There must be Wi-fi connection. • Source code must be embedded in Arduino Uno board.
Scenario	1. Power supply must be provided to UGV. 2. UGV starts to make reconnaissance. 3. Threat or enemy move around facility. 4. Threat or enemy object is detected. 5. Warning message is sent to commander.
Exceptional Situations & Alternative Flows	1. Message may not transmit to commander. • Check Wi-fi connection and detected object.
Postcondition	1. Warning message is received by commander.

3.3.2.2.3. Scenario 3: Attack when it is necessary

Use Case Number	3
Use Case Name	Attack when it is necessary
Summary	If facility is under attack, then team may attack back to enemy battlefield.
Actor	Commander, UGV, Attacker&Defender Team, Enemy
Trigger	Facility is under attack, and commander gives the order of attack back.
Precondition	• Commander must give the order of attack. • UGV is ready to use for both spy and guardian modes.
Scenario	1. Power supply must be provided to UGV. 2. Attack command is received by commander. 3. Defender&Attacker team arrived to field and ready to attack. 4. UGV waits for command to be able to spy on enemy battlefield.
Exceptional Situations & Alternative Flows	1. Attack command may not be received by UGV and soldiers. • Check Wi-Fi connection of UGV and application. As a second way, commander may give order to soldiers directly.
Postcondition	1. Team attacks to enemy battlefield.

3.3.2.2.4. Scenario 4: Defend Facility

Use Case Number	4
Use Case Name	Defend Facility
Summary	Team defend facility when facility is under attack.
Actor	Commander, UGV, Attacker&Defender Team, Enemy
Trigger	Facility is under attack, and commander gives the order of defending facility.
Precondition	• Commander must give the order of defend. • UGV is ready to use for both spy and guardian modes.
Scenario	1. Power supply must be provided to UGV. 2. Facility is under attack. 3. Defend command is received by commander. 4. Defender&Attacker team arrived to field and ready to defend.
Exceptional Situations & Alternative Flows	1. Defend command may not be received by UGV and soldiers. • Check Wi-Fi connection of UGV and application. As a second way, commander may give order to soldiers directly.
Postcondition	1. Team defends the facility which is under attack.

3.3.2.2.5. Scenario 5: Provide Service

Use Case Number	5
Use Case Name	Provide Service
Summary	Technician is responsible for taking care of UGVs. S/He provides service when needed.
Actor	UGV, Technician
Trigger	UGV is broken or check time is arrived.
Precondition	• UGV has not enough battery. • UGV is broken or damaged.
Scenario	1. UGV runs out of battery. 2. Technician provides proper battery or power supply for UGV. 3. If UGV is damaged, then technician fix it.
Exceptional Situations & Alternative Flows	1. Tools of UGV may not be available at that moment. • Technician provides necessary tools and fix the UGV as soon as possible. 2. Technician may not be able to solve the problem. • Technician shall contact with developers of UGV.
Postcondition	1. UGV is ready to use again.

3.3.2.2.6. Scenario 6: Observe Enemy & Enemy Battle

Use Case Number	6
Use Case Name	Observe Enemy & Enemy Battle
Summary	Commander or soldier observes enemy or enemy battlefield via sent video.
Actor	UGV, Enemy, Commander
Trigger	Spy mode is activated.
Precondition	• Spy mode must be selected. • Power supplies of UGV must be provided appropriately. • UGV must have enemy or object around of it. • There must be Wi-fi connection. • Source code must be embedded in Arduino Uno board. • ESP32-CAM must be ready to use.
Scenario	1. Power supply must be provided to UGV. 2. Commander or soldier selects spy mode. 3. Threat or enemy move around facility. 4. Threat or enemy object is detected. 5. Warning message is sent to commander. 6. Commander starts examining the status of enemy.
Exceptional Situations & Alternative Flows	1. Real time video may not be sent to receiver which is our application. • First, check the Wi-fi password and SSID is correct. • Second, check smart phone is connected to same Wi-fi with ESP-32 Camera. • Third, check Wi-fi connection has 2.4Ghz instead of 5Ghz. 2. Detection algorithm may not work appropriately. • Technician is informed about situation to fix the problem.
Postcondition	1. Commander understands the condition of enemy.

3.3.2.2.7. Scenario 7: Control Units

Use Case Number	7
Use Case Name	Control Units
Summary	Commander controls units and give commands when there is a threat.
Actor	Commander, Soldier
Trigger	Facility is under attack or when team attacks to enemy.
Precondition	• UGV must have enemy or object around of it. • The team must attack the enemy.
Scenario	1. Facility is under attack. 2. Commander is informed about attack via UGV. 3. Commander gives defend or attack order. 4. Units start to operation.
Exceptional Situations & Alternative Flows	1. Command may not be received by UGV and soldiers. • Check Wi-Fi connection of UGV and application. As a second way, commander may give order to soldiers directly.
Postcondition	1. The commander gives commands to the team according to the condition.

3.3.2.2.8. Scenario 8: Attempt Attack

Use Case Number	8
Use Case Name	Attempt Attack
Summary	Enemy may attempt to attack via gun, remote controller, or a bomb.
Actor	Enemy
Trigger	Enemy attacks to facility.
Precondition	• Enemy decided to attack facility.
Scenario	1. Enemy attacks to facility.
Exceptional Situations & Alternative Flows	1. Enemy may not be detected by our UGV. • Detection algorithm will be updated with detailed data.
Postcondition	1. UGV observes enemies.

3.3.2.2.9. Scenario 9: Perform Self-Driving

Use Case Number	9
Use Case Name	Perform Self-Driving
Summary	UGV moves by itself. There will be no human intervention.
Actor	UGV, Commander
Trigger	Commander selects Guardian mode.
Precondition	• Power supplies of UGV must be provided appropriately. • Source code must be embedded in Arduino Uno board. • There must be Wi-fi connection. • Our application must be downloaded in commander's or soldier's device. • Guardian task is assigned to UGV.
Scenario	1. Controller (Soldier) or commander opens our application in his/her smartphone. 2. S/he chooses spy mode. 3. UGV starts to move itself through certain path.
Exceptional Situations & Alternative Flows	1. Ultrasonic sensor may not be worked as it is expected. • Technician will be informed. Hardware connections and source code will be checked, if problem cannot be detected, ultrasonic sensor may be updated with a new one. 2. UGV may not move on road. • Ultrasonic sensor will be checked by Technician. Also, additional hardware objects may act as obstacle.
Postcondition	1. UGV starts to move without human intervention.

3.3.2.2.10. Scenario 10: Drive Manually

Use Case Number	10
Use Case Name	Drive Manually
Summary	UGV is controlled by soldiers.
Actor	Controller (Soldier), Commander, UGV
Trigger	User selects Spy mode.
Precondition	• Power supplies of UGV must be provided appropriately. • Source code must be embedded in Arduino Uno board. • There must be Wi-fi connection. • Distance between controller and UGV must be up to 9 meters. • Our application must be downloaded in commander's or soldier's device. • Controller opens our application in his/her device.
Scenario	1. Controller (Soldier) or commander opens our application in his/her smartphone. 2. S/he chooses spy mode. 3. S/he press the direction buttons.
Exceptional Situations & Alternative Flows	1. Application may not work. • Restart the application and report situation to technician. 2. Control buttons may do not work. • Check the HC05 Bluetooth connection. 3. Bluetooth connection may fail. • Check the HC05 Bluetooth connection and restart the module. 4. In smart phone, Bluetooth may be off. • Open the Bluetooth in smart phone.
Postcondition	1. UGV moves based on input comes from controller or commander.

3.3.2.2.11. Scenario 11: Detect Objects

Use Case Number	11
Use Case Name	Detect Objects
Summary	UGV detects all objects around.
Actor	UGV, Enemy
Trigger	Objects moves around UGV.
Precondition	• Power supplies of UGV must be provided appropriately. • UGV must have enemy or object around of it. • There must be Wi-fi connection. • Source code must be embedded in Arduino Uno board.
Scenario	1. UGV starts moving around facility. 2. Objects starts moving around facility. 3. UGV observes objects. 4. UGV detects objects.
Exceptional Situations & Alternative Flows	1. Enemy objects may not be detected. • Dataset can be trained with more military objects. 2. Camera may fail during detection. • Check necessary internet connections are satisfied. • Check camera is available for usage. • Check source code is run as it is expected.
Postcondition	1. Object is detected and identified.

3.3.2.2.12. Scenario 12: Give Reports

Use Case Number	12
Use Case Name	Give Reports
Summary	UGV reports when hazardous materials or events are detected among all objects.
Actor	UGV, Enemy, Commander
Trigger	Enemy, threat, or hazardous object is detected.
Precondition	• Power supplies of UGV must be provided appropriately. • UGV must have enemy or object around of it. • There must be Wi-fi connection. • Source code must be embedded in Arduino Uno board. • Threat is detected. • Our application must be downloaded in commander's or soldier's device. • Controller opens application in his/her device.
Scenario	1. UGV starts moving around facility. 2. UGV starts detecting objects around facility. 3. UGV detects threat or enemy. 4. UGV sends message about detected object via application. 5. Controller (Soldier) or commander opens our application in his/her smartphone. 6. Controller sees the message in his/her device.
Exceptional Situations & Alternative Flows	1. Report can be transmitted to commander when there is no emergency. • That means detection cannot be performed appropriately. Dataset and source code must be updated as soon as possible. 2. UGV cannot send reports to commander. • Check Wi-Fi connection and if detection is really performed.
Postcondition	1. Message is received by commander.

3.3.3. Non-Functional Requirements

In this part of the report, we stated non-functional requirements of the project.

3.3.3.1. Performance Requirements

Performance Requirements	Description
Response Time	Real time video must be displayed on commander's monitor with at most 3 seconds delay. When detection algorithm runs, delay of video will increase by 3 seconds. We Therefore, when threat is detected, warning message will be delivered to commander less than 7 seconds.
Error Handling	When unpredictable failure occurs, UGV informs commander and technician its status. Technician recovers UGV as soon as possible.
Workload	System will handle many sub-systems simultaneously. System is going to be able to handle object detection, sending video via LAN, and moving autonomously/manually at the same time.
Scalability	Each UGV will be controlled by single smartphone via our application. Therefore, multiple users will not use our system simultaneously, and system will not crash because of low-scalability issue.
Application requirements	In smart phone, there must be 10MB free space for our application. CPU speed or RAM of device is not a crucial concern.

3.3.3.2. Safety Requirements

Safety Requirements	Description
Prevent Damage	The vehicle must take care to avoid other objects and must not harm the environment.
Instantly Reporting	The vehicle should instantly report system failures.
Accurately and Safely Reporting	The vehicle must report system faults accurately and safely.
Safe Intervention	The vehicle should allow the controller to take control of the vehicle when the system is not working.
Reaction and Detection	The vehicle needs to detect objects in its environment and react to them if it detects a dangerous situation.

3.3.3.3. Security Requirements

Security Requirements	Description
Wi-fi Connection	ESP-32 CAM module send video continuously via Wi-Fi. Therefore, Wi-Fi password must be protected to prevent leaks. Since we set password in our source code, it must be carefully saved in computer or external hard drive.
Bluetooth Connection	Our vehicle can move manually. Therefore, anyone who has a smartphone can easily connect to our Bluetooth module. To prevent this, our module asks pin before connection is completed.

Application Access	We will develop an application which displays video sent from ESP-32 CAM. Therefore, if apk file of our application is leaked, it could cause threats. Therefore, apk file of application must be also protected.

3.3.3.4. Software Quality Attributes

Software Quality Attributes	Definition
Reliability	Every functionality on the code should be able to work without error in any condition where condition is normal.
Robustness	Our UGV shall work properly under difficult environment conditions.
Portability	The system should work on Android and Windows.
Correctness	System must predict objects with high accuracy.
Learnability	System shall be easy to understand and simple to learn for usage.
Maintainability	When a failure arises, system must recover it without causing fatal changes on the code structure.
Extensibility	System can be improved with additional features. Therefore, new system is extendable.
Testability	System should work without any errors; any it must be testable with different scenarios.

Efficiency	The system should operate with maximum performance with minimum used battery or power supply.
Usability	By using our application which is easy to use, our vehicle can be operated.
Administrability	Our UGV can also operate manually by controller.
Autonomy	Our UGV can move without any human intervention.
Modifiability	In the future, the car may become available for attack if the project is improved. But we will not attack using vehicle.

3.4. References

[1] Components of our UGV, “How to assemble a car?” [Online]. Available: https://www.codemahal.com/video/building-a-4wd-autonomous-car-with-arduino/ [Accessed 28 November 2021].

[2] 4WD Car Cit Supplies, “Hardware Components of Car” [Online]. Available: https://www.codemahal.com/video/building-a-4wd-autonomous-car-with-arduino/ [Accessed 28 November 2021].

[3] HC-SR04 Ultrasonic Sensor “Features of HC-SR04” [Online]. Available: https://www.codemahal.com/video/building-a-4wd-autonomous-car-with-arduino/ [Accessed 28 November 2021].

[4] HC06 Bluetooth Module “HC06 Bluetooth Module Features” [Online]. Available: https://classes.engineering.wustl.edu/ese205/core/index.php?title=Bluetooth_Module_(HC-06)_%2B_Arduino#:~:text=The%20HC%2D06%20bluetooth%20module,%3A%20smart%20phones%2C%20PC). [Accessed 28 November 2021].

4.SOFTWARE DESIGN DESCRIPTION

4.1. Introduction

This Software Design Description Report provides comprehensive information on what kind of system software Autonomous Vehicle should include. This document includes the working principles of the methods that we will develop. Moreover, it mentions knowledge regarding designs of the android application. In our project, we will build our system using both hardware and software techniques. For software part, we will be using C/C++ to move our vehicle in both manual and self-driving modes. For that purpose, we use Arduino IDE. To be able to operate our car in manual mode, we aim to develop our own software which receives real-time video and capable of moving car from remote control. This software application may be written in Java, C# or HTML/CSS. For real-time object detection, we decided to use YOLO algorithm which provides fast results. In addition, accuracy of this algorithm satisfies the requirements, hence, it is preferred for our project. For hardware part, we will use variety of modules and tools which are detailed in 2.4.Hardware Design section.
In the future process of our development, these techniques can be changed based on test results we received. Therefore, we will decide step by step as we receive outputs of our system.

4.1.1. Purpose of This Document

This Software Design Document provides a description of the technical design for Kaşif UGV (Unmanned Ground Vehicle). The main purpose of this document is to explain the technical vision of how requirements in SRS document will be fulfilled. Additionally, it will provide an overview of the system's design arhitecture. This document explains what is to be built and how can we built the system. Therefore, document contains all the interfaces, diagrams and interactions that we used during the development process of the project.

4.1.2. Scope of the Project

This project is aimed to design a fully autonomous vehicle and to develop it in order to protect the security of life of law enforcement officers and soldiers from external dangers. Our project has both software and hardware. It is aimed to produce a final product containing hardware and software by making simulations in appropriate environments. It needs to be equipped with a powerful processor and a good camera to detect hazards.

Our project has the following features:

- Object detection and tagging
- Notify the host when a malicious object is detected
- Ability to work with 24/7 patrol method
- Fully autonomous
- Ability to work synchronously with the main system
- Emergency response

In order to be successful in the project, the software and hardware parts must work in sync with each other. The final product needs to be tested in a real environment after the simulation phase.

4.1.3. Glossary (Definitions, Acronyms, and Abbreviations)

TERM	DEFINITION
Actor	An actor can be a user, or another software system that interacts with the system.
Android	A name of the mobile operating system made by an American company; Google.
C/C++	C and C++ high-level and general-purpose programming language. C is a structured programming language while C++ is an object-oriented programming language.
Python	Python is another high-level programming language that is preferable when Artificial Intelligence algorithms are used.
IOS	A mobile operating system created and developed by Apple.
Software Design Description (SDD)	A document that provides a comprehensive description of the system's design, requirements, and conditions to be able to perform methods. This document is an SDD document.
User	Users will be students who are participated to competition using our online platform and they will find solutions for the given problem as teams.
Autonomous	Autonomous means independent. In our project, we call our system autonomous vehicle since it does not need any human intervention to be able to move.
Object Detection	Object Detection is a popular computer science technology. By using vision techniques, computer decides and identifies what the object is.

UGV	UGV refers to Unmanned Ground Vehicle. It is a vehicle that can move on the ground by itself. There are various of UGVs used for different purposes. In this project, we designed our UGV which will be for surveillance purposes.
GUI application	GUI refers to Graphical User Interface. It provides an interface in which users can interact with existing software. In our project, we are planning to implement a GUI application that visualizes the operations of our system.
Arduino	Arduino is an open-source microcontroller board. We use an Arduino board to upload UGV's movement-related code.
Arduino IDE	Arduino board must be programmed via its IDE which is called Arduino IDE. Therefore, Arduino IDE is an Integrated Development Environment for Arduino boards.
Spyder IDE	We use the Python programming language for many operations in our system. Spyder IDE is an open-source development platform for the Python programming language.
Local Area Network (LAN)	A Local Area Network is a computer network that can be used within a limited area.
JAVA	We will use Java to develop application which provides to control moving and display detecting objects.
YOLO	YOLO algorithm has good accuracy with faster run time. That's why we used YOLO to detect objects.
Android Studio	We developed an application on Android using Android Studio. This application shows the image of detected objects and provides a movement vehicle by using the arrow keys.

4.1.4. Overview of This Document

In this software design document, we aimed to provide a comprehensive design description of Kaşif UGV. Therefore, we stated design statements of our system. In 1. Introduction section, we provide a summarized explanation regarding our system and shared the basic features of our system. We categorized 1. Introduction section as 1.1. Purpose of This Document, 1.2. Scope of the Project, 1.3. Glossary, and 1.5. Motivation subsections. In 1.1. Purpose of This Document section, we stated what this document aims to explain. In 1.2. Scope of the Project, we stated what must be achieved to create our system, and shared features of our system. In 1.3. Glossary, we provide a definition of all special terms that we used in this report. In 1.5. Motivation section, we shared why we implement such system, and reason behind our methodologies. Afterwards, in 2. System Design section, we shared comprehensive design architecture of our system. It includes subsections which are 2.1. Architectural Design, 2.2 Decomposition Description, 2.3. User Interface Design and 2.4. Hardware Design. In 2.1. Architectural Design, we stated different diagrams of our system and explain what is the context of the design idea behind the system. In 2.2. Decomposition Description explains general structure of our system. We provided a hierarchy between among the modules, and the structure. In 2.3. User Interface Design section, we stated designs for our mobile application of the project. These designs are given as figures with brief explanations. We stated which features can be controlled from the application and stated how the GUI will be designed. In 2.4. Hardware Design, we shared the design arhitecture between hardware components, and state how we actually assemble the system. In 3. Requirement Matrix section, we provided a table format which indicates the connection between components and requirements which are stated in SRS document. For that purpose, we generated a link between system components and requirements by explaining which requirement is satisfied by which component. We checked each requirement is attached to the necessary component. Therefore, this document will serve as a guideline for soldiers, technician, commander and development team for understanding the general design arhitecture of Kaşif UGV.

4.1.5. Motivation

Today, technology is advancing very quickly. One of the technologies of our age is autonomous vehicles that can make decisions on their own, supported by artificial intelligence. The usage areas of autonomous vehicles have started to increase day by day because they provide excellent convenience to our lives. Most importantly, when used in military areas, it reduces the loss of life to 0 and provides the security of the area it is located 24/7. As a group that closely follows technological developments, we determined that the future is in this field and decided to develop autonomous vehicles. We aware of the fact that we will be competent engineer candidates experienced in this field when we receive the final product, and we continue our work and move forward with the goal of doing the best. The contribution we will make to our country and ourselves is our greatest source of motivation.

4.2. System Design

This system design section includes the Architectural Design of the system, the definition of the problem, which technologies are used, User Interface Design, and Hardware Design. Also, it contains various diagrams such as Sequence Diagram, Activity Diagram, Data Flow Diagram, Class Diagram.

4.2.1. Architectural Design

In this section, we described the problem description, technologies used and different diagrams to provide better explanation of our architectural design. As we explained in previous sections, our system will include 3 main tasks. Hence, we will generate an architecture which contains these tasks. In Figure 1, Kaşif UGV's main structure is given.

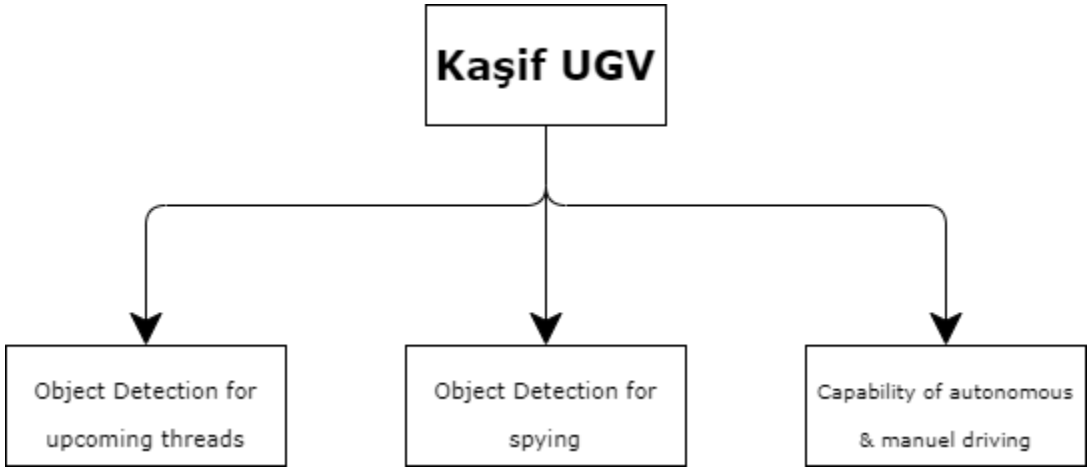


Figure 1: System's general tasks

4.2.1.1. Problem Description

Due to the fact that our country is in the middle east geography, it presents a dangerous situation. Our soldiers and security forces are responsible for protecting our country on a 24/7 basis, but when the human factor is involved, there is of course no personnel who can work fully efficiently on a long scale. Thanks to the autonomous vehicles that can make decisions with the high processor and artificial intelligence support developed, it is possible to provide security in a smooth and reliable way and to reduce the loss of life to 0. The project is being developed with the aim of both ensuring the security of our country and preventing possible loss of life.

4.2.1.2. Technologies Used

There are all the hardware products we use in our system below:

- Arduino Uno Board
- Robotic Car Cit (4 Wheels, 4 motors, chassis, screws, and AA battery enclosure)
- HC-SR04 Ultrasonic Sensor
- HC05/HC06 Bluetooth Module
- L298N Motor driver
- FTDI Programmer
- ESP32-CAM module
- Additional wires, breadboard, and connector cables
- Power Supply

There are all the software tools we use in our system below:

- YOLO

  We used the YOLO algorithm because it is faster and more advantageous than Faster R-CNN and other object detection algorithms.

- Arduino IDE version 8.1.12

  We use Arduino Uno board to be able to run our code on the vehicle.

- Python

  We use Python programming language for many operations in our system.

- Spyder IDE

  Spyder IDE is an opensource development platform for Python programming language.

- C/C++

  We used C/C++ on Arduino to enable the vehicle to do manual driving, autonomous driving, guard mode detection and spy mode detection.

The application which will be responsible for operating vehicle will be developed by a programming language which can combine above solutions. It can be Java, HTML/CSS, or Python. In future process of development, it will be decided.

4.2.1.3. Data Flow Diagram

In figure 2, data flow diagram is stated.

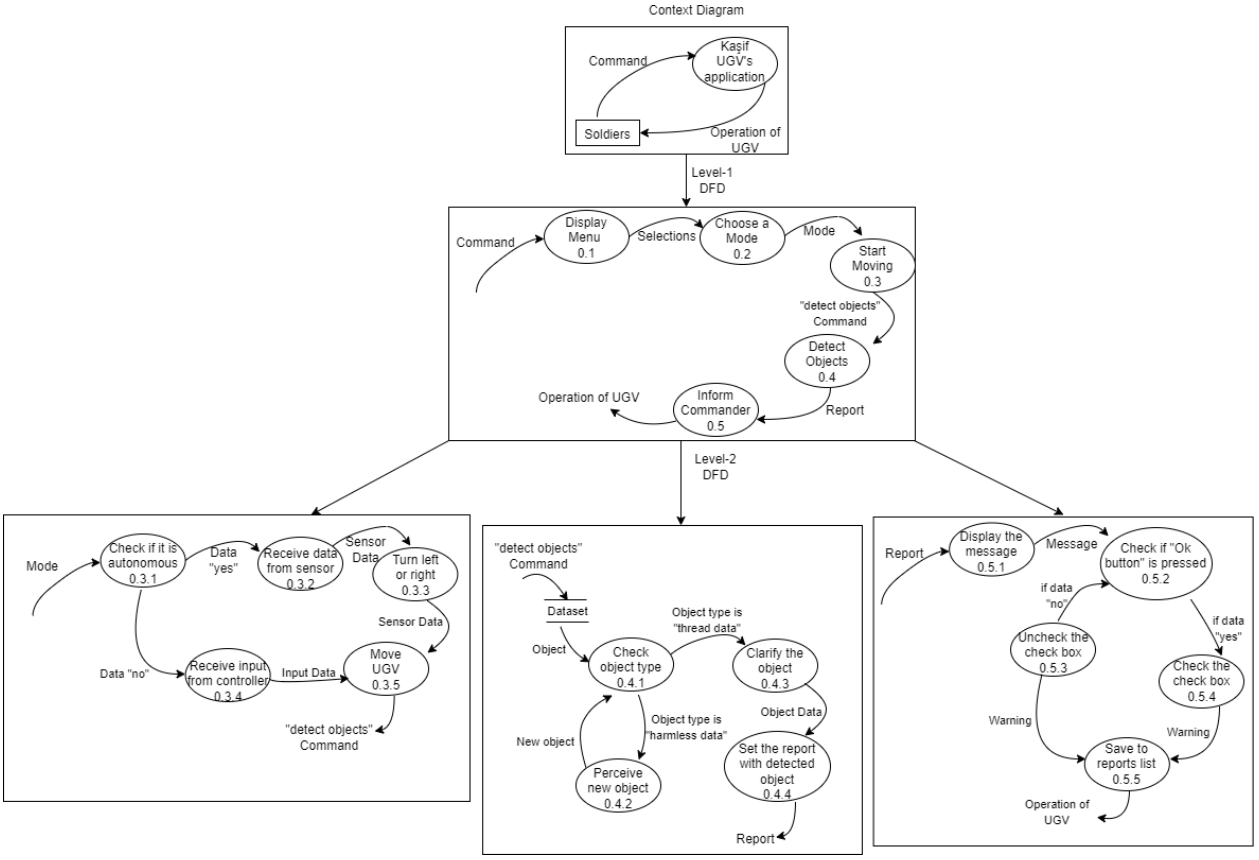


Figure 2: Data Flow Diagram

4.2.1.4. Activity Diagram

In figure 3, activity diagram is stated.

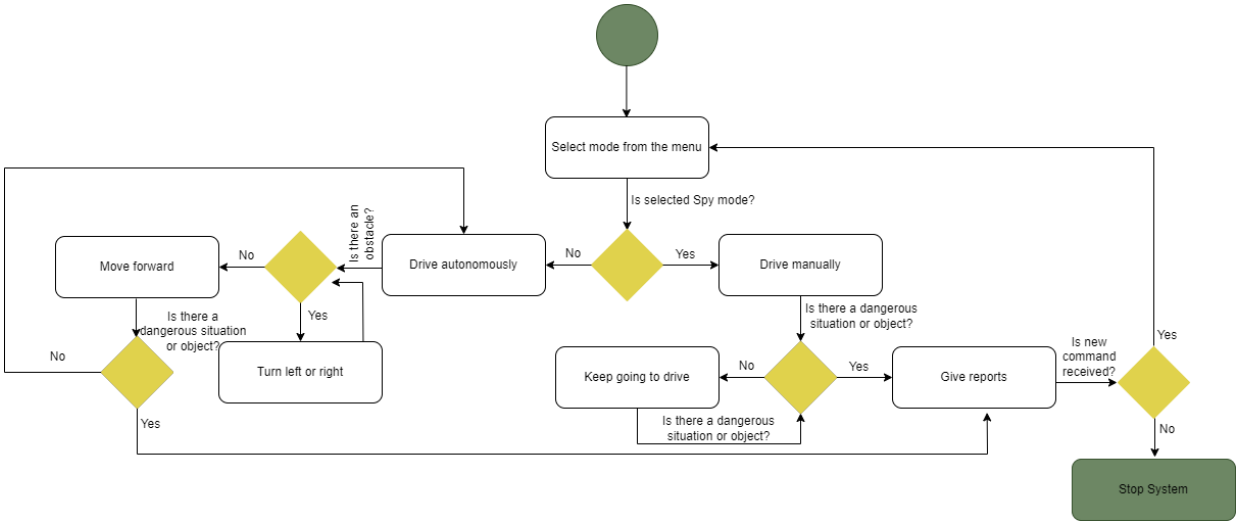


Figure 3: Activity Diagram

4.2.1.5. Class Diagram

In figure 4, class diagram is stated.

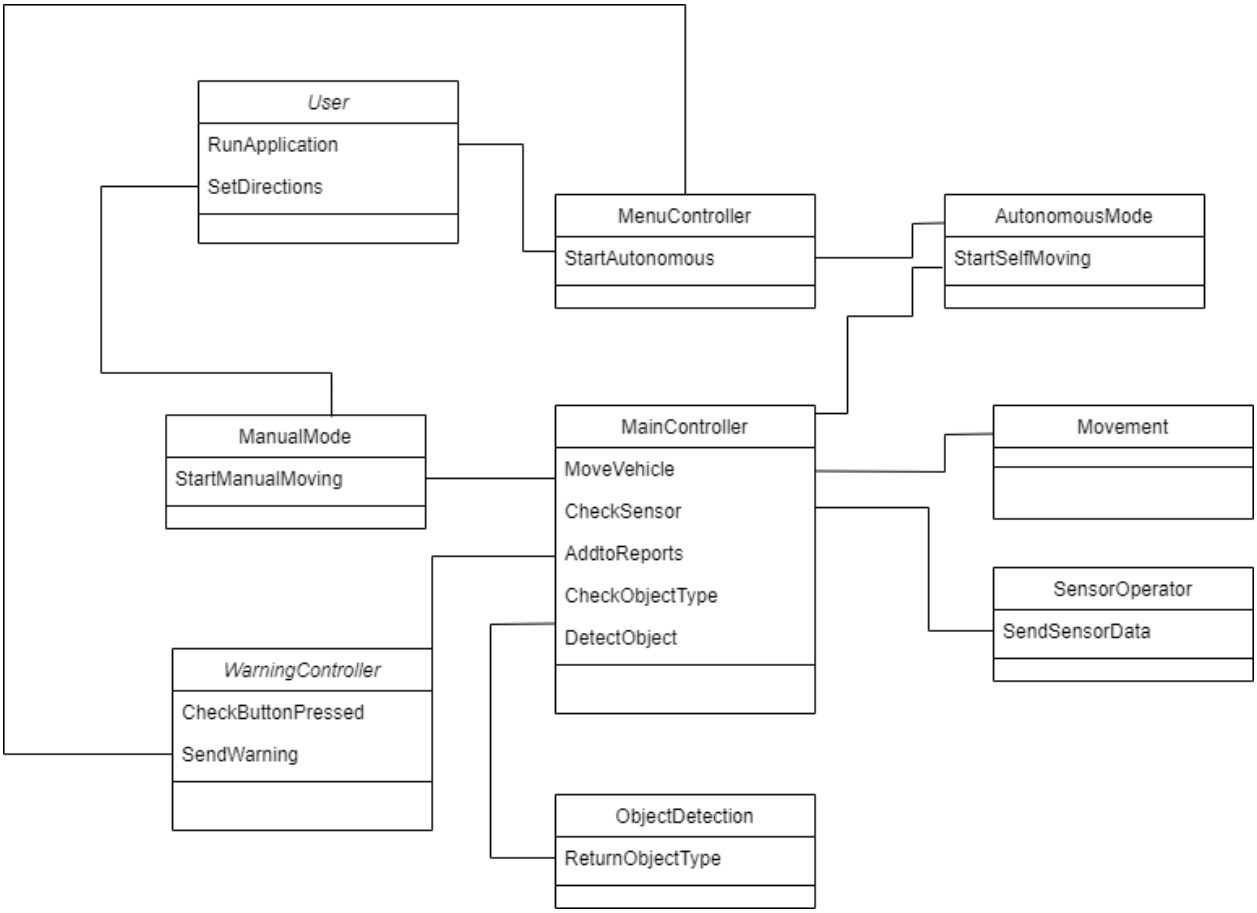


Figure 4: Class Diagram

4.2.1.6. Sequence Diagram

In figure 5, sequence diagram is stated.

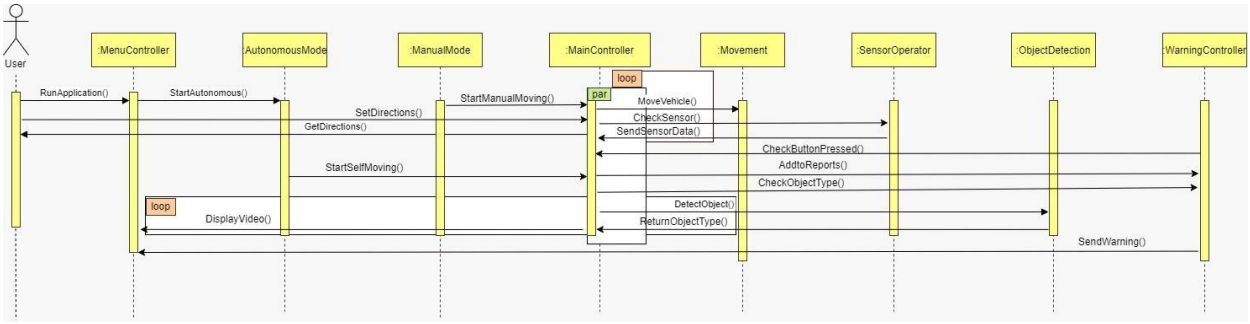


Figure 5: Sequence Diagram

4.2.2. User Interface Design

In this section, we will state our designs for possible use cases and explain what features are included for each UI screen.

4.2.2.1. Home Page

Figure 6: Home Page Screen

To be able to operate any feature of our project, user must need to choose which mode will Kaşif operate. Therefore, in home page, we prompt user to choose a mode which can be guardian&autonomous or spy&manual. When one of these options is selected, screen will change to the one the following screens.

4.2.2.2. Spy Mode Screen

Figure 7: Spy Mode Screen

This interface is the main interface of our mobile application. It includes UC-01 Control Vehicle and UC-11 Detect Objects use cases. To be able access this page, “Spy Mode” should be selected in home page. Commanders and soldiers can control the vehicle from here and observe the stream that is coming from vehicle’s camera. In the screen there are 7 buttons and two menu bars in total. The left-side menu bar is responsible for changing screens to display other features of the application. From right-side menu bar user can exit from the application, and reach general information related the application. By pressing “LED” button, user can open the led light of ESP32 camera which provides light. This feature can be used for night operations or surveillances. By pressing “STREAM” button, user will observe the real time video which is coming from the ESP32 camera. Therefore, user can watch the environment of Kaşif UGV. When user presses “DETECT OBJECT” button, our vehicle starts to detect objects around it. User will be able to observe that is detected at that moment. Sample visual is given in above figure.

4.2.2.3. Threat Captured in Spy Mode

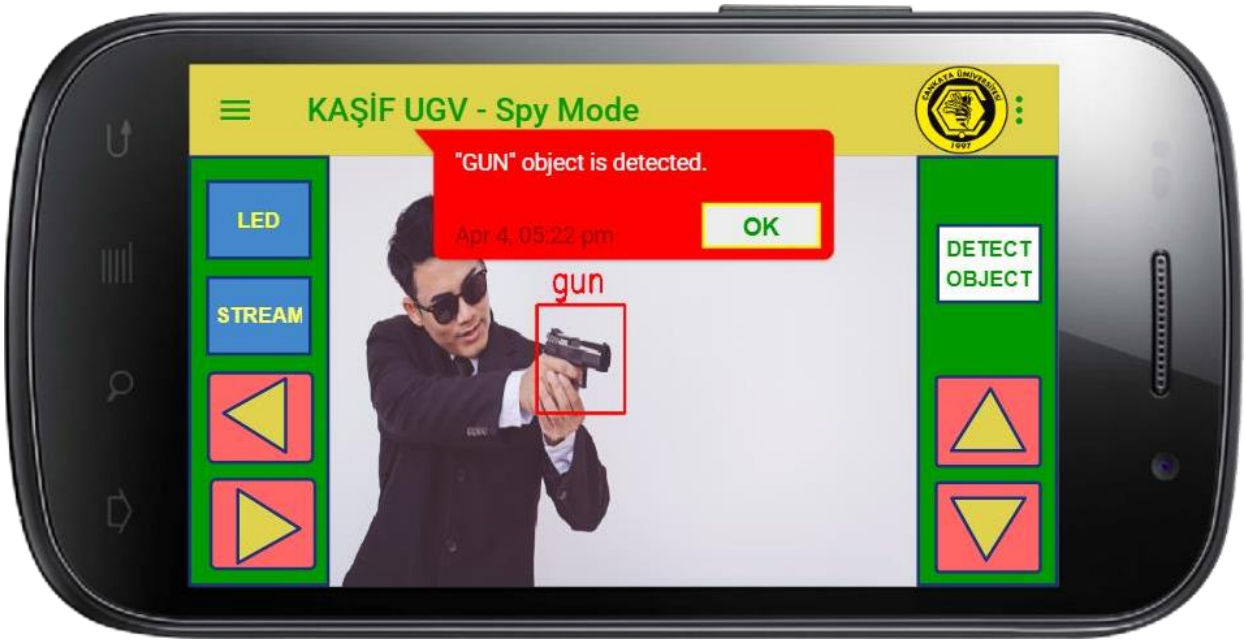


Figure 8: Inform Commander UI

This interface mainly includes UC-02 Inform Commander, UC-10 Drive manually use cases. To be able access this page, “Spy Mode” should be selected in home page. When the camera detects suspicious objects or threats, commander must receive a warning. In above figure, a warning can be observed. It indicates which and when the suspicious object is detected. There is also a “OK” button which will be used to check if commander received the message. If “OK” button is not clicked, in the Reports page, it will be seen as unchecked. Therefore, to get commanders’ and soldiers’ attention, warning box is colored as red.

4.2.2.4. Guardian Mode Screen

Figure 9: Guardian Mode Screen

This interface includes UC-09 Perform Self-Driving use case. To be able to enter this screen, Guardian&Autonomous mode must be selected from the home page. Therefore, once above screen open, our vehicle will start to move itself. As it can be seen from above figure, there are no direction arrows anymore. There are only “STREAM”, “DETECT OBJECT” and “LED” buttons to control ESP32 camera. In here, there will be a warning pop-up as well. The reports will be written into Reports page as checked or unchecked. In short, by using this page, without human invervention, we can observe objects in facility’s environment and receive a warning when there a threat is detected.

4.2.2.5. Display Reports

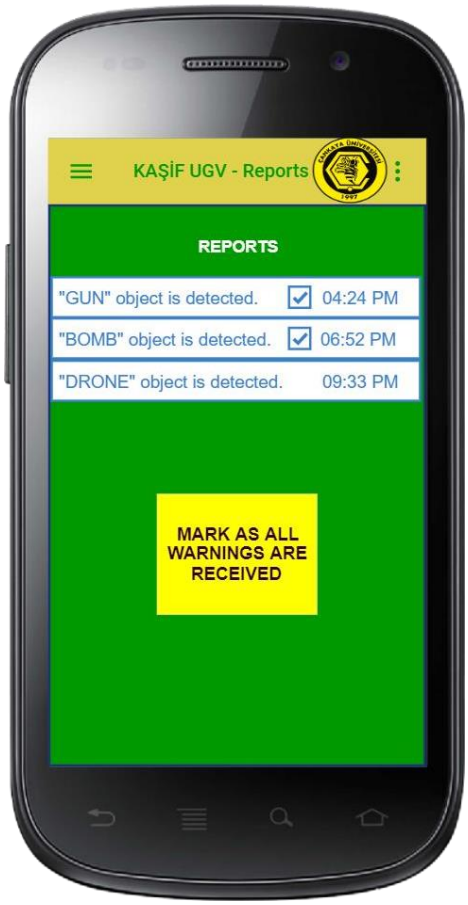


Figure 10: Reports Screen

This interface is used to display reports received from our vehicle. It includes UC-012 Give Report use case. To be able access this page, “Reports” page should be selected from menu page. In this page, warning messages will be listed with time information. From above figure, it can be observed that there is a checkbox at the near of each message. It refers to the “OK” button that we explained in *2.3.3. Threat Captured in Spy Mode* section. If commander or soldier presses OK button when the warning is received, then there will be a checkbox at the near of the message. This feature can be used to understand if any threat is detected when the commander or soldier was not available to see the warning. They can check the list and observe all warnings. After that, by pressing “MARK AS ALL WARNINGS ARE RECEIVED” button, they can check every warning to refer all of them are received.

4.2.3. Hardware Design

Since in our project, we will also generate the physical product of the vehicle, we need variety of hardware components. That is why, we needed regular and comprehensive design to make all components work properly. In this section, we will state our hardware design by categorizing it into five subsections which are ESP32 Camera – FTDI Module Design, Motors – L298N Motor Driver - Arduino Uno Board Design, HC05 Bluetooth – Arduino Uno Board Design, HC-SR04 Sensor – Arduino Design, and Power Supply Design. All the components which we explain the design architecture below are explained in SRS document in *3.1.2. Hardware Interfaces*.

4.2.3.1. ESP32 Camera – FTDI Module Design

We stated that we used ESP32 Camera to get real-time data via Wi-Fi with FTDI programmer module. In ESP32 Camera module we have a development board, and OV2640 camera. There is no USB programming board on ESP32 development module, that is why, we used external programmer FTDI. We assembled these two components as it can be seen from below figure[1].

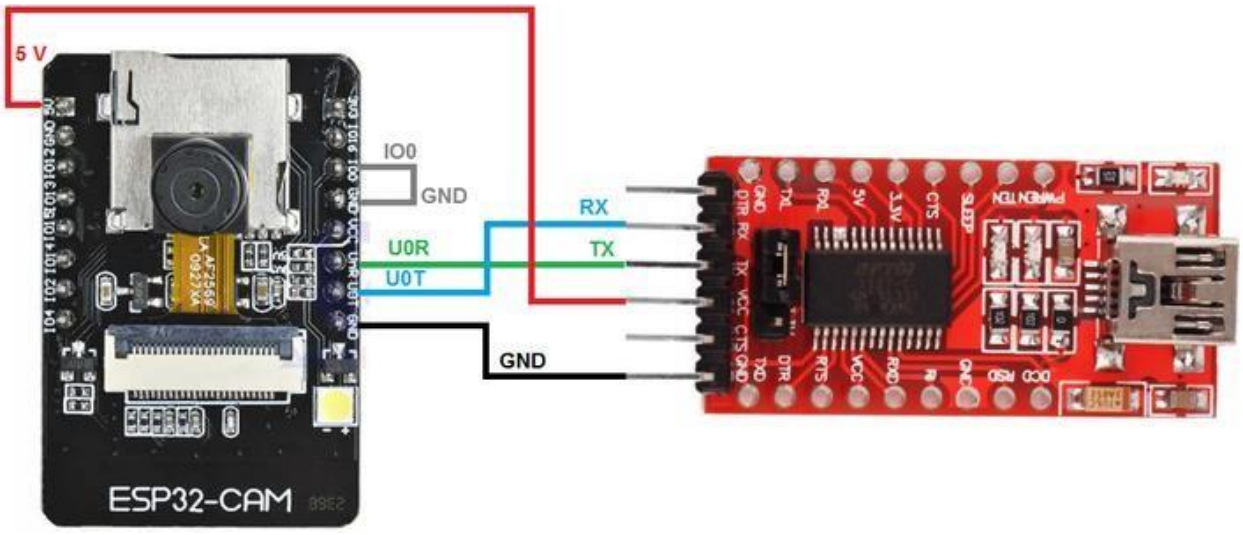


Figure 11: Connections between ESP32 Camera and FTDI module

As it can be seen from above figure, we supply board with exactly 5V power to use it with maximum performance. All the wire connections are stated in Figure 11. Additionally, there is one crucial step which we must perform during uploading process of our code. After upload is done and serial port is opened, we must press RESET button on ESP32 development board. After that, we opened serial monitor and disconnect GPIO0 from ground since code is already uploaded. After implementing these steps, our camera is ready to be find from external devices via Bluetooth and recording videos.

4.2.3.2. Motors – L298N Motor Driver - Arduino Uno Board Design

For each wheel, we used a DC motor. Therefore, there are 4 DC motors in our circuit. In below figure, only half of the connections to provide better understanding of connections.[2] On L298N motor driver, we plugged pins to set wheels' speed and direction of motors. By using the direction control pins, we can control motor spins either forward or backward.

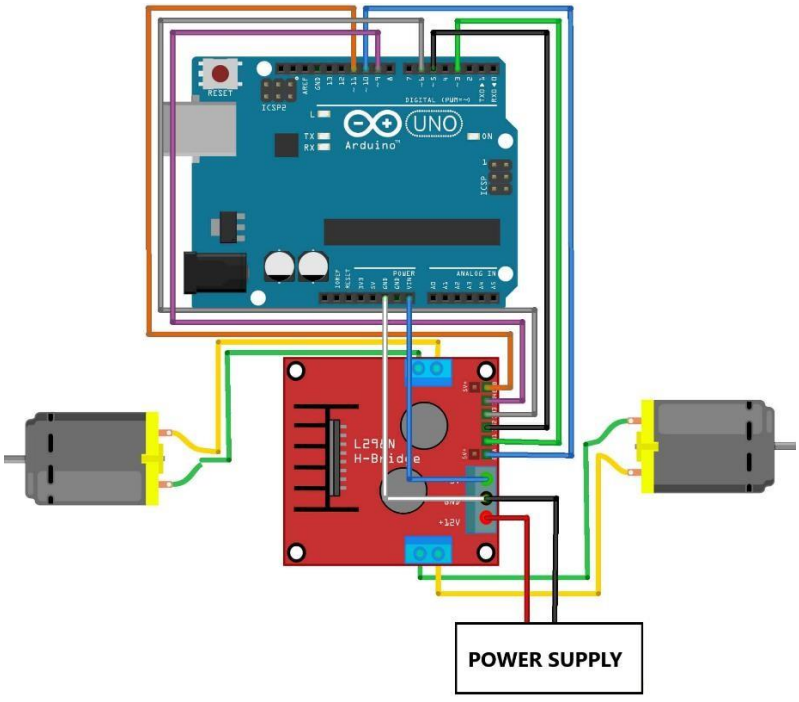


Figure 12: Connections between Motor, motor driver and Arduino

4.2.3.3. HC05 Bluetooth - Arduino Uno Board Design

To able to operate wheels from remote control, we used HC05 Bluetooth module which ıs connected to our Android smartphone. Using this module, we can control our vehicle manually via our smartphone using our application. HC05 will be blinking when it is in configuration mode. To connect HC05 with Android smartphone, Bluetooth settings must be switched to ON. Once the connection is satisfied, it stops blinking. The circuit that we performed is given at below figure[3].

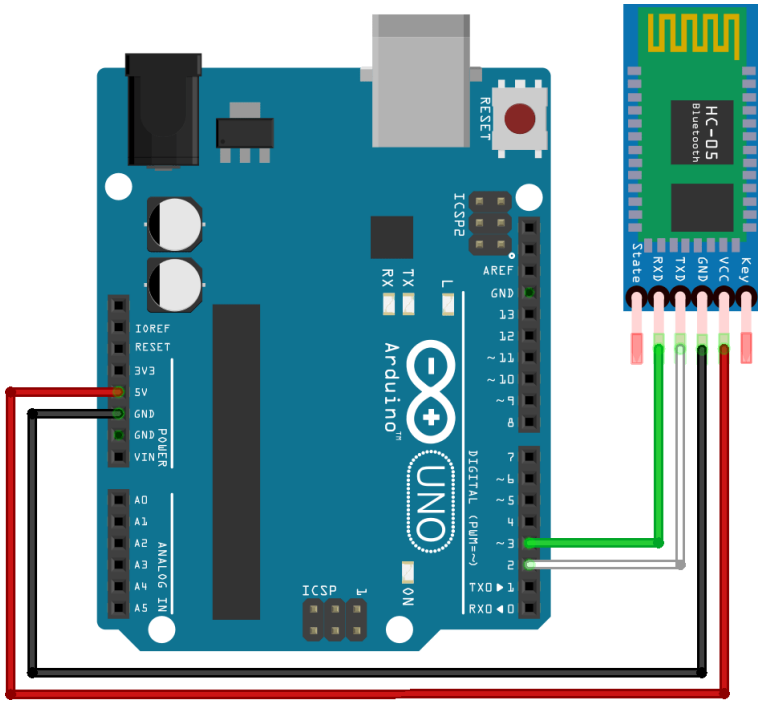


Figure 13: Connections between Arduino and HC05

4.2.3.4. HC-SR04 Sensor - Arduino Uno Board Design

Since our vehicle must be operated in autonomous mode as well, we used HC-SR04 sensor which measures distance. It emits an ultrasound at 40kHz to detect obstacles around it. Therefore, it calculates distance based on speed of sound[4]. To make HC-SR04 sensor work properly, we assemble the circuit as follows:

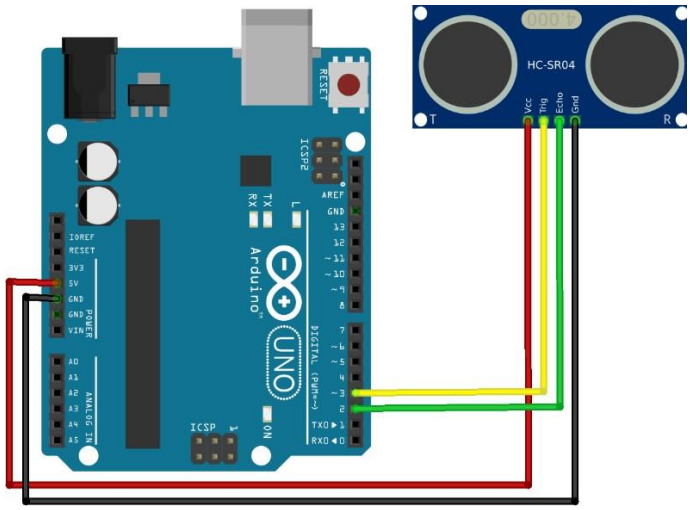


Figure 14: Connections between Arduino and HC-SR04

In below figure, how can distance be calculated using HC-SR04 is explained.

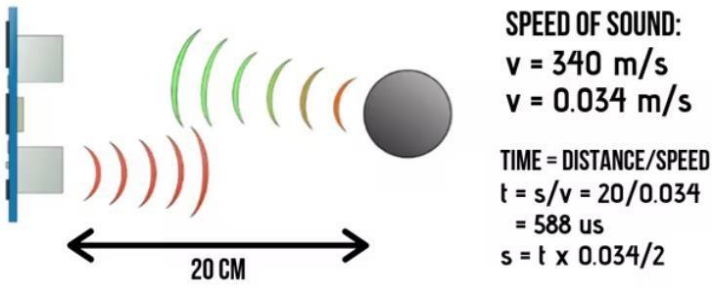


Figure 15: Distance Calculation using HC-SR04

Since sound wave needs to travel forward and bounce backward, we divided speed by two when calculating the time.

4.2.3.5. Power Supply Design

In this section, power requirements of crucial hardware components are given.

HC-SR04 Ultrasonic Sensor:

- **Voltage:** 5V
- **Current:** 15mA

L298N Motor Driver:

- **Voltage:** 5-35V
- **Current:** 0-36mA

ESP32 Camera Module:

- **Voltage:** 5V
- **Current:** 2A

DC Motor:

- **Voltage:** 3-12V
- **Current:** 95mA

HC-05 Bluetooth Module:

- **Voltage:** 1.8-3.6V
- **Current:** 50mA

4.3. Requirements Matrix

In Figure 16, requirement matrix of our system is stated.

	Components	COMP-01	COMP-02	COMP-03
Requirements				
FREQ - 1				X
FREQ - 2		X	X	
FREQ - 3		X	X	
FREQ - 4		X	X	X
FREQ - 5				X
FREQ - 6		X	X	
FREQ - 7				X
FREQ - 8				X
FREQ - 9				X
FREQ - 10				X
FREQ - 11		X	X	
FREQ - 12				X

COMP - 01 = Object Detection for upcoming threads
COMP - 02 = Object Detection for Spying
COMP - 03 = Capability of Autonomous & manuel driving

FREQ - 1 = Control Vehicle
FREQ - 2 = Inform Commander
FREQ - 3 = Attack when it is necessary
FREQ - 4 = Defend Facility
FREQ - 5 = Provide Service
FREQ - 6 = Observe Enemy & Enemy Battle
FREQ - 7 = Control Units
FREQ - 8 = Attempt Attack
FREQ - 9 = Perform Self-Driving
FREQ - 10 = Drive Manually
FREQ - 11 = Detect Objects
FREQ - 12 = Give Reports

Figure 16: Requirement Matrix

References

[1] Assembly of ESP32 Camera, “Connections between FTDI and ESP32 Development Module” [Online]. Available:
https://www.instructables.com/ESP32-Cam-Programmer/
[Accessed 25 December 2021].

[2] How to use L298N Motor Driver, “Connections between motors and L298N” [Online]. Available:
https://create.arduino.cc/projecthub/ryanchan/how-to-use-the-l298n-motor-driver-b124c5
[Accessed 25 December 2021].

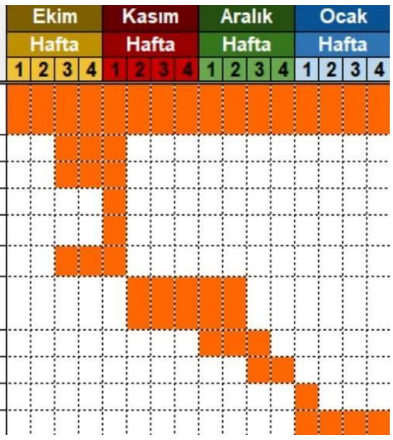
[3] How to use HC05 with Arduino “Circuit Diagram of HC05 and Arduino” [Online]. Available:
https://www.aranacorp.com/en/arduino-and-bluetooth-module-hc-05/
[Accessed 26 December 2021].

[4] How to use HC-SR04 with Arduino “Circuit Diagram of HC-SR04 and Arduino” [Online]. Available:
https://create.arduino.cc/projecthub/abdularbi17/ultrasonic-sensor-hc-sr04-with-arduino-tutorial-327ff6
[Accessed 26 December 2021].

5.WORK PLAN

In below figure, we stated our work plan which clarifies our tasks to implement the project. We divided them into rows and assign due dates for each individual task. As a team we shared each task equally. Therefore, each team member worked for each task on the Work Plan table. After each task, a feedback from advisor is received.

TASK DEFINITION	Status	START DATE	FINISH DATE
DOCUMENTS	In progress	26.Eki.21	25.Eki.22
Project Proposal Form	Complete	23.Eki.21	5.Kas.21
Project Selection Form	Complete	24.Eki.21	5.Kas.21
GitHub Repository	Complete	28.Eki.21	5.Kas.21
Project Work Plan	Complete	7.Kas.21	12.Kas.21
Literature Review	Complete	25.Eki.21	12.Kas.21
Software Requirements Specification (SRS)	Complete	12.Kas.21	10.Ara.21
Project Webpage	Complete	11.Ara.21	17.Ara.21
Software Design Description (SDD)	Complete	18.Ara.21	31.Ara.21
Project Report / Project Tracking Form	Complete	1.Oca.22	7.Oca.22
Presentation	Not started	5.Oca.22	23.Oca.22



6.Test Plan, Test Design Specifications and Test Cases

6.1. INTRODUCTION

6.1.1. Version Control

Version No	Description of Changes	Date
1.0	First Version	March 25, 2022

6.1.2. Overview

KAŞİF is an unmanned ground vehicle. The use case of the system had been explained in the SRS document. The system will be tested according to these features in different environments.

6.1.3. Scope

This document includes the test plan of the use cases, test cases according to the test plan, and test design specifications. Therefore, the following sections will explain how we will implement test cases and what will be the test criteria.

6.1.4. Terminology

Acronym	Definition
SRS	Software System Requirements
SDD	Software Design Document
UGV	Unmanned Ground Vehicle
GUI	Graphical User Interface
Autonomous Driving	A control mode which has no need to human interaction when driving.
Manual Driving	A control mode in which vehicle requires a controller to be able to move.
Android Application	A software which is coded by team members, and it includes different modes and features of the system.

6.2. FEATURES TO BE TESTED

In this section, we will explain our test plan and provide general information regarding the features to be tested. Additionally, for each feature, there will be a Test Design Specification given at the end of this document.

6.2.1. Detecting Objects

KAŞİF UGV has a camera which can detect objects and send the real time video to our Android application. Therefore, users are capable of observing external threats around that environment.

6.2.2. Autonomous Driving

KAŞİF UGV can move without any human interaction. For that purpose, the system uses the outputs that are received from its sensors. In this way, it can act as a guard instead of soldiers.

6.2.3. Manual Driving

KAŞİF UGV can be operated via our Android Application. For that purpose, the system has a Bluetooth module to transmit data. In this way, KAŞİF can be used for surveillance to make a reconnaissance.

6.2.4. Warning Users

KAŞİF UGV can detect harmful objects specifically. When it detects such an object, a warning is sent to the Android Application, and a message is displayed on the screen.

6.3. FEATURES NOT TO BE TESTED

In this section, we described features which we will not be tested.

6.3.1. Performance on different environments

Our hardware components can be used when the environmental factors are regular. Therefore, we will not test KAŞİF UGV under such circumstances.

6.3.2. Autonomous mode when the path is not regular

KAŞİF UGV can operate in autonomous mode when the path is separated by obstacles. Because of the sensor's performance, it cannot operate in any other road autonomously. Therefore, we will not test KAŞİF UGV when the path is not regular.

6.4. ITEM PASS/FAIL CRITERIA

To be able to get success on this project, our vehicle should move properly without a controller, or move according to the controller's inputs, detect objects with high accuracy, and warn the user when the detected object is classified as a threat. If any of the features are described in Section 2, we will consider the test as failed.

6.4.1. Exit Criteria

- 100% of the test cases are executed
- All High and Medium Priority test cases passed

6.5. REFERENCES

[1] CENG408_Group7_SRS, March 25, 2022. Available: https://github.com/CankayaUniversity/ceng-407-408-2021-2022-Autonomous-Vehicle/wiki/Software-Requirement-Specification-(SRS)

[2] CENG408_Group7_SDD, March 25, 2022. Available: https://github.com/CankayaUniversity/ceng-407-408-2021-2022-Autonomous-Vehicle/wiki/Software-Design-Description-(SDD)

6.6. TEST DESIGN SPECIFICATIONS

6.6.1. Detecting Objects (DO)

6.6.1.1. Subfeatures to be tested

6.6.1.1.1. ESP-32 Camera and Android Application Connection (DO.CAAC)

To be able to display detected objects on the screen, the ESP-32 Camera and Android Application must be connected via Wi-Fi. After the connection is handled, we receive a real time video on the screen. At this step, system is ready for detection procedure.

6.6.1.1.2. Clarify Object Identity (DO.COI)

Since we are detecting objects, the identity of each object is crucial because we want to know what the external objects in the environment are. Therefore, we require identity of each object with high accuracy.

6.6.1.1.3. Clarify Object Type (DO.COT)

Once we observe objects and detect their identity, we should know the object type as it is threat or not. For that purpose, we clarify each object's type based on their safety.

6.6.1.2. Test Cases

In this section, we listed related test cases for feature Detecting Objects (DO).

TC ID	Requirements	Priority	Scenario Description
DO.CAAC.01	3.2.2.11	High	Wi-fi connection's configuration is set to 2.4 Ghz.
DO.CAAC.02	3.2.2.11	High	ESP-32 Camera is detected by Wi-fi.
DO.CAAC.03	3.2.2.11	High	Press "STREAM" button on Android Application.
DO.CAAC.04	3.2.2.11	High	Receive real time video from camera.

TC ID	Requirements	Priority	Scenario Description
DO.COI.01	3.2.2.11	High	Press “DETECT” button on Android Application.
DO.COI.02	3.2.2.11	High	Every possible object which can be safe or dangereous is detected.

TC ID	Requirements	Priority	Scenario Description
DO.COT.01	3.2.2.11	High	Clarify object identity as it is dangereous or not.

6.6.2. Autonomous Driving (AD)

6.6.2.1. Subfeatures to be tested

6.6.2.1.1. Autonomous Mode Choice (AD.AMC)

User should make a choice between autonomous and manual modes. When Autonomous Mode button is pressed vehicle needs to be configured as autonomous automatically.

6.6.2.1.2. Checking Obstacles (AD.CO)

KAŞİF UGV has a sensor which is responsible for checking obstacles. This sensor should calculate the distance between vehicle and obstacle and return the relative data.

6.6.2.1.3. Vehicle Movement (AD.VM)

KAŞİF UGV should start to move according to the sensor’s data. The distance should be calculated carefully to provide accidents and crashes. According to that information, vehicle should move from its current position.

6.6.2.2. Test Cases

In this section, we listed related test cases for feature Autonomous Driving (AD).

TC ID	Requirements	Priority	Scenario Description
AD.AMC.01	3.2.2.9	High	Connection between Android Application and HC-05 Bluetooth Module satisfied.
AD.AMC.02	3.2.2.9	High	Autonomous Mode choice must be sent to vehicle’s bluetooth module.

TC ID	Requirements	Priority	Scenario Description
AD.CO.01	3.2.2.9	High	Choice data is received by HC-05. HC-SR04 sensor module is activated.
AD.CO.02	3.2.2.9	High	Path is created regularly with obstacles and other solid materials.
AD.CO.03	3.2.2.9	High	HC-SR04 calculate the distance between obstacle and itself.
AD.CO.04	3.2.2.9	High	According to calculated distance, vehicle starts to move.

TC ID	Requirements	Priority	Scenario Description
AD.VM.01	3.2.2.9	High	When vehicle escaping from one obstacle, it should not hit to another one.
AD.VM.02	3.2.2.9	High	Sensor's peformance should be observed to understand how much distance it can calculate.
AD.VM.03	3.2.2.9	High	According to calculated distance path width will be updated.

6.6.3. Manual Driving (MD)

6.6.3.1. Subfeatures to be tested

6.6.3.1.1. Manual Mode Choice (MD.MMC)

User should make a choice between autonomous and manual modes. When Manual Mode button is pressed vehicle needs to be configured as manual automatically.

6.6.3.1.2. Check Cables and Power (MD.CCP)

To be able to drive in manual mode, the Bluetooth sensor must be turned on and Bluetooth module should exist. This relationship should be checked before making this move.

6.6.3.1.3. Bluetooth and Android Application Connection (MD.BAAC)

For the vehicle to be driven manually, the Bluetooth and Android application must be connected via Wi-Fi. After connecting, the vehicle will be able to be driven with the motion screen.

6.6.3.2. Test Cases

In this section, we listed related test cases for feature Manual Driving (MD).

TC ID	Requirements	Priority	Scenario Description
MD.MMC.01	3.2.2.10	High	The user has to choose between autonomous or manual driving.
MD.MMC.02	3.2.2.10	High	The driver should direct KAŞİF using the direction keys in the android application.
MD.MMC.03	3.2.2.10	High	It should check the data from sensors and camera while driving.

TC ID	Requirements	Priority	Scenario Description
MD.CCP.01	3.2.2.10	High	Check Bluetooth module and all power cables.
MD.CCP.02	3.2.2.10	High	Check if the wires are in the right place using the pin-to-pin diagram.

TC ID	Requirements	Priority	Scenario Description
MD.BAAC.01	3.2.2.10	High	Connection between Android application and HC-05 Bluetooth module satisfied.
MD.BAAC.02	3.2.2.10	High	Manual Mode choice must be sent to vehicle's Bluetooth module.

6.6.4. Warning Users (WU)

6.6.4.1. Subfeatures to be tested

6.6.4.1.1. Checking Wi-fi and Bluetooth Connection (WU.CWBC)

Detected dangerous objects must be transmitted by KAŞİF UGV. Additionally, detected objects must be able to be displayed on the warning screen. Therefore, ESP-32 Camera and Android app must be connected via Wi-Fi. The autonomous vehicle must be within Bluetooth range for transmitting the warning to the user. In this step, the system is ready for transmission and detection.

6.6.4.1.2. Detecting Threat (WU.DT)

The object is detected when dangerous objects or enemies move or are found in the facility. The vehicle warns the user depending on the correct identification of the detected hazardous object and its type.

6.6.4.1.3. Reporting Message (WU.RM)

KAŞİF UGV detects all objects while driving. It transmits a report to commander or controller (soldier) when hazardous materials or incidents are detected. This reporting message is displayed on the Android application screen with time and type knowledge.

6.6.4.2. Test Cases

In this section, we listed related test cases for feature Warning Users (WU).

TC ID	Requirements	Priority	Scenario Description
WU.CWBC.01	3.2.2.2 - 3.2.2.12	High	Wi-fi for ESP-32 Camera and HC-05 Bluetooth connections for Android app are provided.

TC ID	Requirements	Priority	Scenario Description
WU.DT.01	3.2.2.2	High	Detect threat or enemy object.
WU.DT.02	3.2.2.2	High	Inform commander for warning about dangerous objects or enemies using the Android app screen.

TC ID	Requirements	Priority	Scenario Description
WU.RM.01	3.2.2.2	High	A warning message is sent to the commander.
WU.RM.02	3.2.2.12	High	The controller or commander displays warning messages.
WU.RM.03	3.2.2.12	High	A detailed report was given for dangerous objects or enemies with the time and type information.

6.7. Detailed Test Cases

6.7.1. DO.CAAC.01

TC_ID	DO.CAAC.01
Purpose	Make ESP-32 visible by Wi-fi for connection process.
Requirements	3.2.2.11
Priority	High.
Estimated Time Needed	20 seconds
Dependency	There must be Internet connection.
Setup	A computer/router and Android smartphone must be provided.
Procedure	[A01] Connect computer and smartphone to the Internet.
	[A02] Open Network & Internet settings from computer.
	[A03] Press Mobile Hotspot section.
	[A04] Select network band as 2.4Ghz.
	[V01] Observe connection is ready with 2.4Ghz band.
Cleanup	Close Wi-fi connection.

6.7.2. DO.CAAC.02

TC_ID	DO.CAAC.02
Purpose	Make ESP-32 Camera ready to stream.
Requirements	3.2.2.11
Priority	High.
Estimated Time Needed	10 seconds
Dependency	DO.CAAC.01 needs to work correctly.
Setup	Computer must be working.
Procedure	[A01] Power up the ESP-32 Camera.
	[A02] Power on the Mobile Hotspot with configured settings.
	[A03] Open devices connected section.
	[V01] Observe ESP-32 Camera in the device list.
Cleanup	Disconnect power of ESP-32 Camera.

6.7.3. DO.CAAC.03

TC_ID	DO.CAAC.03
Purpose	Make Android application ready for receiving video data.
Requirements	3.2.2.11
Priority	High.
Estimated Time Needed	30 seconds
Dependency	DO.CAAC.01 and DO.CAAC.02 need to work correctly.
Setup	Android smartphone must be provided.
Procedure	[A01] Connect that smartphone to internet.
	[A02] Install Android application to smartphone.
	[A03] Open Android application.
	[A04] Choose one of the modes.
	[V01] Observe STREAM button on the screen and press it.
Cleanup	Close the application.

6.7.4. DO.CAAC.04

TC_ID	DO.CAAC.04
Purpose	Observing real time video stream from vehicle's camera.
Requirements	3.2.2.11
Priority	High.
Estimated Time Needed	10 seconds
Dependency	DO.CAAC.01, DO.CAAC.02, and DO.CAAC.03 need to work correctly.
Setup	Android smartphone must be provided.
Procedure	[A01] A connection message is observed after DO.CAAC.03 test case.
	[A02] Video is transmitted by Wi-fi.
	[V01] Observe the stream on the smartphone's screen.
Cleanup	Press STREAM button again to block display of stream.

6.7.5. DO.COI.01

TC_ID	DO.COI.01
Purpose	Make Android application ready for detecting objects from video data.
Requirements	3.2.2.11
Priority	High.
Estimated Time Needed	10 seconds
Dependency	DO.CAAC.03 needs to work correctly.
Setup	All the set up in DO.CAAC must be provided.
Procedure	[A01] Stream is receiving from ESP-32 camera.
	[V01] DETECT button is pressed.
Cleanup	Close the application.

6.7.6. DO.COI.02

TC_ID	DO.COI.02
Purpose	Detect every possible object around vehicle.
Requirements	3.2.2.11
Priority	High.
Estimated Time Needed	Not estimated.
Dependency	DO.COI.01 needs to work correctly.
Setup	All the set up in DO.CAAC must be provided.
Procedure	[A01] DETECT button must be observed.
	[A02] Press DETECT button.
	[A03] Detection algorithm is run.
	[V01] Observe detected objects.
Cleanup	Press DETECT button again to display only stream.

6.7.7. DO.COT.01

TC_ID	DO.COT.01
Purpose	Define if object is dangereous or not.
Requirements	3.2.2.11
Priority	High.
Estimated Time Needed	30 seconds
Dependency	DO.COI test cases need to work correctly.
Setup	All the set up in DO.CAAC and DO.COT must be provided.
Procedure	[A01] Object is detected.
	[A02] Check that object is the list of dangerous materials.
	[A03] If it is, identify it as a threat, otherwise identify it as a safe one.
	[V01] Observe the identifications.
Cleanup	Press DETECT button again to display only stream or close the application.

6.7.8. AD.AMC.01

TC_ID	AD.AMC.01
Purpose	Handle connection between bluetooth module with Android application.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	20 seconds
Dependency	Bluetooth module should be powered up. Android application is ready to use.
Setup	HC-05 Bluetooth module is assembled, and smartphone has an internet connection.
Procedure	[A01] Open the application.
	[A02] Open device's Bluetooth via application.
	[A03] List devices and find HC-05.
	[A04] Observe HC-05's led is blinking.
	[A04] Press HC-05 on the list.
	[V01] Observe blinking is stopped and connection is satisfied.
Cleanup	Disconnect the power supply of Bluetooth module or close the application.

6.7.9. AD.AMC.02

TC_ID	AD.AMC.02
Purpose	Handle mode choice of user.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	10 seconds
Dependency	AD.AMC.01 needs to work correctly.
Setup	HC-05 Bluetooth module is assembled, and smartphone has an internet connection.
Procedure	[A01] After observing connection is satisfied, observe the next page.
	[A02] Observe two buttons as options manual or autonomous.
	[A03] Press on AUTONOMOUS button.
	[V01] Choice should be sent to the vehicle.
Cleanup	Disconnect the power supply of Bluetooth module or close the application.

6.7.10. AD.CO.01

TC_ID	AD.CO.01
Purpose	Activate HC-SR04 sensor module.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	10 Seconds
Dependency	AD.AMC test cases should work correctly.
Setup	An admin user should be created.
Procedure	[A01] Choice is received by Bluetooth module.
	[A02] According to the choice, function calls are set for autonomous.
	[V01] Observe HC-SR04 sensor module is activated.
Cleanup	Disconnect the power supply of Bluetooth module or close the application.

6.7.11. AD.CO.02

TC_ID	AD.CO.02
Purpose	Create a proper path or parkour for efficient movement of the KAŞİF.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	5 Minutes
Dependency	Width of the road should be configured according to HC-SR04.
Setup	Materials and components for road is provided.
Procedure	[A01] Set up a dummy parkour to understand HC-SR04 performance.
	[A02] According to [A01] test, put the obstacles for road.
	[V01] Observe that road is ready for self-movement of the vehicle.
Cleanup	Clean the road.

6.7.12. AD.CO.03

TC_ID	AD.CO.03
Purpose	Calculate distance between obstacles and KAŞİF UGV.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	30 Seconds
Dependency	AD.CO.01 and AD.CO.02 needs to work correctly.
Setup	Sensor, arduino must be powered up. Application must be opened.
Procedure	[A01] Vehicle comes closer to obstacle.
	[A02] Signal is sent from sensor and comes back.
	[V01] Observe distance according to time during the [A02].
Cleanup	Power off the modules and close the application.

6.7.13. AD.CO.04

TC_ID	AD.CO.04
Purpose	Vehicle starts to move according to sensor's data.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	20 Seconds
Dependency	AD.CO.01, AD.CO.02 and AD.CO.03 needs to work correctly.
Setup	Sensor, arduino must be powered up. Application must be opened.
Procedure	[A01] Distance data is received.
	[A02] According to data, vehicle decides on direction to move.
	[V01] Observe vehicle moves itself.
Cleanup	Power off the modules and close the application.

6.7.14. AD.VM.01

TC_ID	AD.VM.01
Purpose	Handling vehicle movement without any crash or accident.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	5 Minutes
Dependency	All the AD.CO test cases should be working correctly.
Setup	Sensors and arduino board must be powered up. Path and obstacles must be ready.
Procedure	[A01] Observe an obstacle.
	[A02] Calculate the distance between obstacle and vehicle.
	[A03] According to distance, escape from obstacle.
	[V01] Observe while escaping from obstacle, vehicle do not crash.
Cleanup	Power off the modules and clean the path.

6.7.15. AD.VM.02

TC_ID	AD.VM.02
Purpose	Understanding sensor's performance for future paths and parkours.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	5 Minutes
Dependency	All the AD.CO test cases should be working correctly.
Setup	Sensors and arduino board must be powered up. Path and obstacles must be ready. For distance calculation, metric components must be provided.
Procedure	[A01] Put multiple obstacles to path with different features.
	[A02] Run the vehicle with autonomous-mode.
	[A03] Check distance calculation of sensor.
	[V01] Observe the performance of sensor with different circumstances.
Cleanup	Power off the sensor module and clean the path.

6.7.16. AD.VM.03

TC_ID	AD.VM.03
Purpose	To obtain better results, update path features according to observations.
Requirements	3.2.2.9
Priority	High.
Estimated Time Needed	5 Minutes
Dependency	AD.VM.02 should be working correctly.
Setup	Path must be organized according to AD.VM.02 test.
Procedure	[A01] According to observations at AD.VM.02, calculate distances.
	[V01] Using new distance data, update the path.
Cleanup	Clean the path and power off the HC-SR04 sensor.

6.7.17. MD.CCP.01

TC_ID	MD.CCP.01
Purpose	Check the Bluetooth module and cables
Requirements	3.2.2.10.
Priority	High.
Estimated Time Needed	60 seconds
Dependency	Bluetooth and Arduino datasheets
Setup	Datasheet must be provided.
Procedure	[A01] Open datasheet.
	[A02] Check cables of Bluetooth and pins.
	[A03] Check cables of Arduino.
	[A04] Select HC-05 Bluetooth module.
	[V01] Observe connection is ready with HC-05.
Cleanup	Close datasheets.

6.7.18. MD.CCP.02

TC_ID	MD.CCP.02
Purpose	Check wires in right place.
Requirements	3.2.2.10.
Priority	High.
Estimated Time Needed	30 seconds
Dependency	Pin-to-pin diagram.
Setup	Pin-to-pin diagram must be provided.
Procedure	[A01] Open pin-to-pin diagram.
	[A02] Check wires in right place by using diagram.
	[A03] Check cables of Arduino.
	[V01] Observe action is ready.
Cleanup	Close diagram.

6.7.19. MD.BAAC.01

TC_ID	MD.BAAC.01
Purpose	Connect application and HC-05
Requirements	3.2.2.10.
Priority	High.
Estimated Time Needed	15 seconds
Dependency	Application and Bluetooth connection
Setup	Application and Bluetooth must be provided.
Procedure	[A01] Open application and HC-05.
	[A02] Check the connection via application or module.
	[A03] Give direction via application.
	[V01] Observe movement.
Cleanup	Close connections.

6.7.20. MD.BAAC.02

TC_ID	MD.BAAC.02
Purpose	Check the Manual Mode
Requirements	3.2.2.10.
Priority	High.
Estimated Time Needed	30 seconds
Dependency	There must be Manual mode in a KAŞİF.
Setup	Bluetooth and Manual mode must be provided.
Procedure	[A01] Select the Manual mode via Bluetooth.
	[A02] Check connections in a KAŞİF.
	[A03] Give direction via application.
	[V01] Observe moving in a manual mode.
Cleanup	Close manual mode.

6.7.21. MD.MMC.01

TC_ID	MD.MMC.01
Purpose	Select Mode
Requirements	3.2.2.10.
Priority	High.
Estimated Time Needed	10 seconds
Dependency	There must be permission to give selection.
Setup	Select mode via Bluetooth.
Procedure	[A01] Connect Bluetooth by using phone.
	[A02] Select the manual mode via Bluetooth.
	[A03] Give direction via application.
	[V01] Observe moving in a manual mode.
Cleanup	Close Bluetooth module and application.

6.7.22. MD.MMC.02

TC_ID	MD.MMC.02
Purpose	Control directions
Requirements	3.2.2.10.
Priority	High.
Estimated Time Needed	A minute
Dependency	There must be connection with KAŞİF and selected manual mode.
Setup	Application and connection.
Procedure	[A01] Check connection with vehicle and application.
	[A02] Decision directions.
	[A03] Give directions via application.
	[A04] Check if it's correct.
	[V01] Observe true directions in a manual mode.
Cleanup	Close manual mode and application.

6.7.23. MD.MMC.03

TC_ID	MD.MMC.03
Purpose	Control data comes from sensors and camera.
Requirements	3.2.2.10.
Priority	High.
Estimated Time Needed	3 minutes
Dependency	There must be connection with sensors and camera.
Setup	Ready KAŞİF for everything
Procedure	[A01] Check connections sensors and camera.
	[A02] Check Wi-Fi connection with ESP-32.
	[A03] Check images comes from camera.
	[A04] Check data comes from sensors.
	[V01] Observe true and real data.
Cleanup	Close the power.

6.7.24. WU.CWBC.01

TC_ID	WU.CWBC.01
Purpose	Handle Wi-fi and HC-05 Bluetooth connections with Android app and computer.
Requirements	3.2.2.2 – 3.2.2.12
Priority	High.
Estimated Time Needed	2 Minutes
Dependency	All the DO.CAAC and AD.AMC.01 test cases should be working correctly.
Setup	A computer/router, Bluetooth module connection, and Android smartphone must be provided.
Procedure	[A01] Check all connections are done, using DO.CAAC procedures.
	[A02] Check KAŞİF UGV is within Bluetooth range.
	[V01] Observe the ESP-32 Camera and HC-05 Bluetooth connections.
Cleanup	Close Wi-fi connection and disconnect power of ESP-32 Camera and Bluetooth.

6.7.25. WU.DT.01

TC_ID	WU.DT.01
Purpose	Warn the user if the object is dangerous.
Requirements	3.2.2.2
Priority	High.
Estimated Time Needed	10 seconds
Dependency	All the DO.COT test cases should be working correctly.
Setup	All the set up in DO.CAAC and DO.COT must be provided.
Procedure	[A01] Check all connections are done, using DO.COT.01 procedures.
	[A02] According to observations at DO.COT.01, list information.
	[V01] Observe type and identification knowledge in the list.
Cleanup	Press DETECT button again to display only stream or close the app.

6.7.26. WU.DT.02

TC_ID	WU.DT.02
Purpose	Give information to the commander regarding hazardous objects or enemies.
Requirements	3.2.2.2
Priority	High.
Estimated Time Needed	10 seconds
Dependency	WU.DT.01 should be working correctly.
Setup	The path must be organized according to WU.DT.01 test.
Procedure	[A01] According to observations at WU.DT.01, collect information.
	[V01] Using listed information, give knowledge to the commander.
Cleanup	Clean the path or close the application.

6.7.27. WU.RM.01

TC_ID	WU.RM.01
Purpose	Give a report message to the commander for detecting objects or enemies.
Requirements	3.2.2.2
Priority	High.
Estimated Time Needed	2 Minutes
Dependency	WU.DT.02 should be working correctly.
Setup	Path must be organized according to WU.DT.02 test.
Procedure	[A01] According to observations at WU.DT.02, notify information.
	[V01] Using reporting message, display object type, and time knowledge on the Android app screen.
Cleanup	Clean the path and close the application.

6.7.28. WU.RM.02

TC_ID	WU.RM.02
Purpose	Display and observe warning messages by the commander or controller.
Requirements	3.2.2.12
Priority	High.
Estimated Time Needed	Not estimated.
Dependency	WU.RM.01 should be working correctly.
Setup	Path must be organized according to WU.RM.01 test.
Procedure	[A01] According to observations at WU.RM.01, examine information.
	[V01] Using reporting message, display knowledge on the Android app screen by the commander or controller.
Cleanup	Clean the path and close the application.

6.7.29. WU.RM.03

TC_ID	WU.RM.03
Purpose	Observe a detailed report message by the commander or controller for hazardous objects or enemies' time and type information.
Requirements	3.2.2.12
Priority	High.
Estimated Time Needed	Not estimated.
Dependency	WU.RM.02 should be working correctly.
Setup	Path must be organized according to WU.RM.02 test.
Procedure	[A01] According to observations at WU.RM.02, examine information.
	[V01] Using reporting message, display type, and time knowledge on the Android app screen by the commander or controller.
Cleanup	Clean the path and close the application.

6.8.Test Results

6.8.1. Individual Test Results

Individual Test Results are given in the below.

TC ID	Priority	Date Run	Run By	Result	Explanation
DO.CAAC.01	High	23.03.2022	Aleyna DEDE	Pass	Make ESP-32 visible by Wi-fi for connection process.
DO.CAAC.02	High	23.03.2022	Eylül ERDOĞAN	Pass	Make ESP-32 Camera ready to stream.
DO.CAAC.03	High	23.03.2022	Ali BOZDOĞAN	Pass	Make Android application ready for receiving video data.
DO.CAAC.04	High	23.03.2022	Eylül ERDOĞAN	Pass	Observing real time video stream from vehicle's camera.
DO.COI.01	High	23.03.2022	Ali BOZDOĞAN	Pass	Make Android application ready for detecting objects from video data.
DO.COI.02	High	23.03.2022	Aleyna DEDE	Pass	Detect every possible object around vehicle.
DO.COT.01	High	23.03.2022	Aleyna DEDE	Pass	Define if object is dangereous or not.
AD.AMC.01	High	20.04.2022	Eylül ERDOĞAN	Pass	Handle connection between bluetooth module with Android application.
AD.AMC.02	High	20.04.2022	Ali BOZDOĞAN	Pass	Handle mode choice of user.
AD.CO.01	High	20.04.2022	Ali BOZDOĞAN	Pass	Activate HC-SR04 sensor module.
AD.CO.02	High	20.04.2022	Eylül ERDOĞAN	Pass	Create a proper path or parkour for efficient movement of the KAŞİF.
AD.CO.03	High	20.04.2022	Eylül ERDOĞAN	Pass	Calculate distance between obstacles and KAŞİF UGV.
AD.CO.04	High	20.04.2022	Eylül ERDOĞAN	Pass	Vehicle starts to move according to sensor's data.
AD.VM.01	High	20.04.2022	Aleyna DEDE	Pass	Handling vehicle movement without any crash or accident.
AD.VM.02	High	20.04.2022	Ali BOZDOĞAN	Pass	Understanding sensor's performance for future paths and parkours.
AD.VM.03	High	20.04.2022	Aleyna DEDE	Pass	To obtain better results, update path features according to observations.
MD.CCP.01	High	30.03.2022	Ali BOZDOĞAN	Pass	Check the Bluetooth module and cables.
MD.CCP.02	High	30.03.2022	Aleyna DEDE	Pass	Check wires in right place.
MD.BAAC.01	High	30.03.2022	Ali BOZDOĞAN	Pass	Connect application and HC-05.
MD.BAAC.02	High	30.03.2022	Eylül ERDOĞAN	Pass	Check the Manual Mode.
MD.MMC.01	High	30.03.2022	Eylül ERDOĞAN	Pass	Select Mode.
MD.MMC.02	High	30.03.2022	Aleyna DEDE	Pass	Control directions.
MD.MMC.03	High	30.03.2022	Aleyna DEDE	Pass	Control data comes from sensors and camera.
WU.CWBC.01	High	4.05.2022	Ali BOZDOĞAN	Pass	Handle Wi-fi and HC-05 Bluetooth connections with Android app and computer.
WU.DT.01	High	4.05.2022	Aleyna DEDE	Pass	Warn the user if the object is dangerous.
WU.DT.02	High	4.05.2022	Eylül ERDOĞAN	Pass	Give information to the commander regarding hazardous objects or enemies.
WU.RM.01	High	4.05.2022	Eylül ERDOĞAN	Pass	Give a report message to the commander for detecting objects or enemies.
WU.RM.02	High	4.05.2022	Aleyna DEDE	Pass	Display and observe warning messages by the commander or controller.
WU.RM.03	High	4.05.2022	Ali BOZDOĞAN	Pass	Observe a detailed report message by the commander or controller for hazardous objects or enemies' time and type information.

6.8.2. Summary of Test Results

Priority	Number of TCs	Executed	Passed
High	29	29	29
Medium	0	0	0
Low	0	0	0
Total	29	29	29

We have operated 29 test cases and all test cases are passed. Therefore, exit criteria is provided.

6.8.3. Exit Criteria

We executed test cases which are given in the above table. All of test cases are passed. Software development process is completed within the estimated timeline successfully. Therefore, exit criteria is provided.

Criteria	Met or Not
100% of the test cases are executed	Met
All High Priority test cases passed	Met

7. USER MANUAL

7.1. INTRODUCTION

7.1.1. ESP32 Cam

While using ESP32 in our project, we provided connection on 2.4 GHz band Wi-Fi network. The ESP32 which is in the RF frequency range of 2.412 GHz to 2.484 GHz has Maximum RF transmission power is 21.5 dBm.

ESP32 that is powered by 40 nm provides Bluetooth 4.2 solutions on a unique chip, moreover, dual quality performance core is located also on this chip. ESP32 is the platform that meets the requests with its convenient design, security performance, power productive usage, durability and highly integrated structure. ESP32 hardware provides a useful software and hardware structure for developers. The software development system inside ESP32 was provided by Espressif. Espressif offers the development of IoT applications, Bluetooth, Wi-Fi and other useful features thanks to the system it has created.

7.1.2. Preparation

To create KAŞİF UGV, following materials must be satisfied:

- PC loaded with Windows, Linux or Mac operating system
- USB cables for both Arduino and FTDI modules
- Power bank and batteries with 3200 mAh and 3.7V
- Arduino IDE and Android Studio
- ESP-32 CAM, 4WD Car Kit, Arduino UNO, HC-SR04 and FTDI modules
- Jumpers to handle connections between boards
- Android device to operate our software application

7.2. Get Started

7.2.1. Get Libraries and Packages

Once the user has the toolchain explained in section 2.1, user also needs to install some specific libraries.

In Arduino IDE, from File -> Preferences -> Additional Boards Manager URLs, add https://dl.espressif.com/dl/package_esp32_index.json. Additionally, from Sketch -> Include Library -> Manage Libraries, install ESP32 package. The related path is given in following figure.

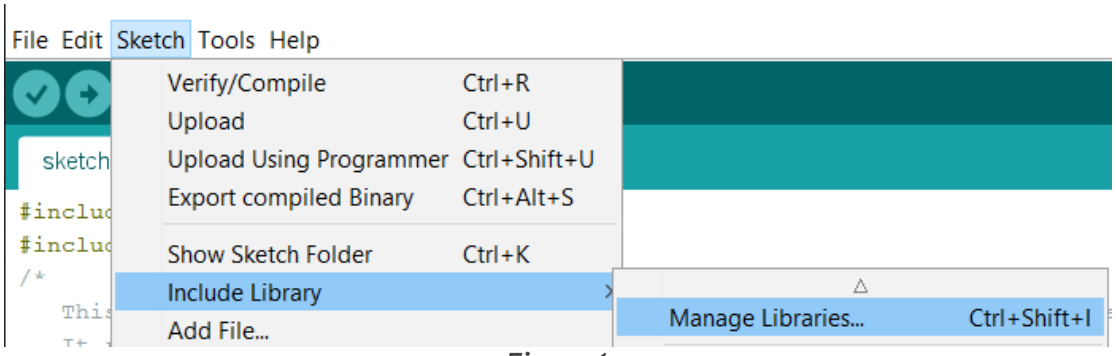


Figure 1

After the installation of related package, it should be looking as following figure.

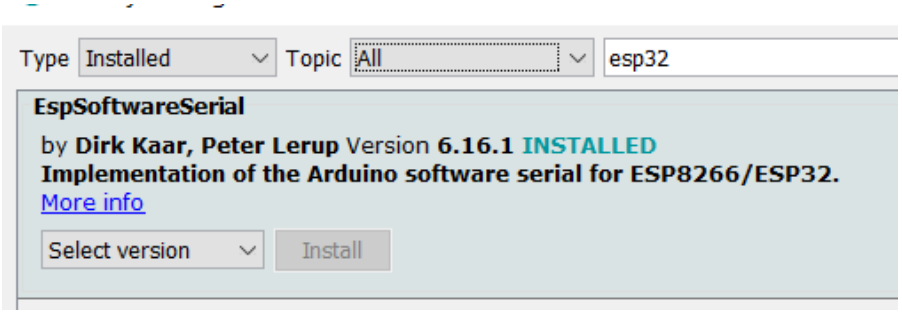


Figure 2

Once above packages are installed, Arduino IDE will be ready for usage of ESP-32 CAM. For Arduino UNO, there is no need to install any packages.

We use FTDI to upload code to ESP-32 CAM module. This FTDI module needs additional driver, so it must be installed to get successful run. Therefore, CH340 driver must be satisfied.

After the installation is handled, user should open the Device Manager and observe the following picture.

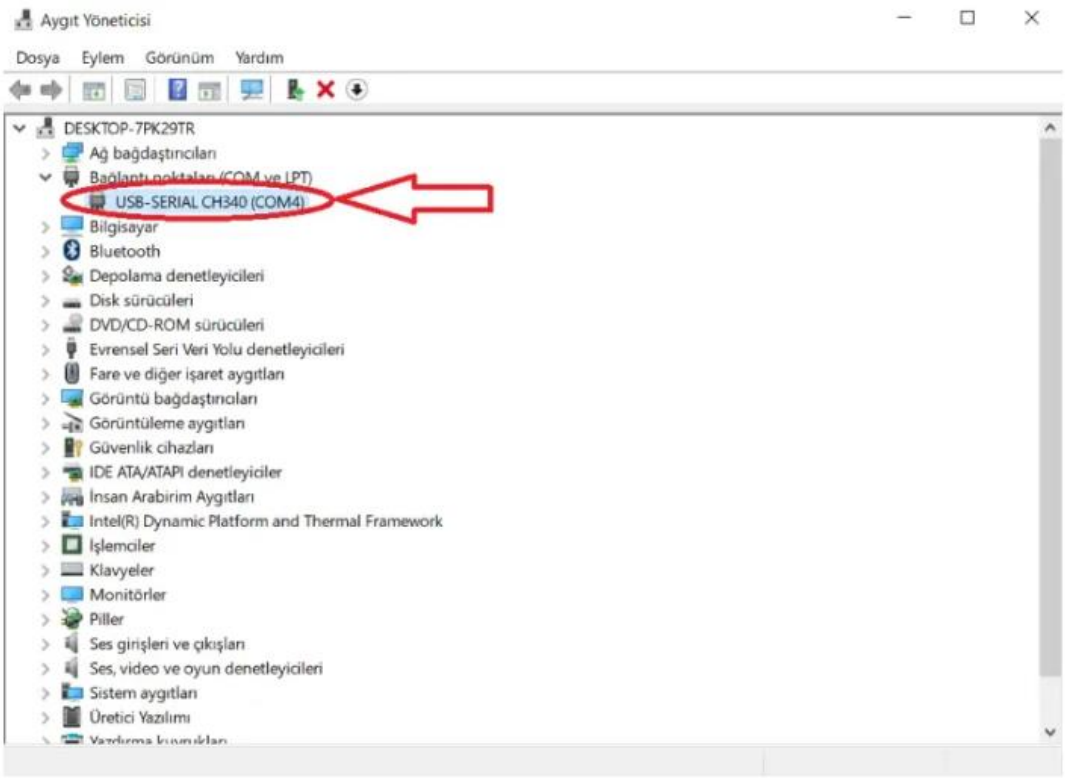


Figure 3

For Arduino UNO board, there is no need to install any additional library, Arduino IDE has already packages for Arduino UNO board. The board should be selected as following before uploading the code. The detailed information will be given in 5. Upload the Code section.

To be able to run our android application, Android Studio environment must be also set up.
We use Arduino IDE with 1.8.12 version and Android Studio with 4.1.2 version.

7.3. Hardware

Since our project is an autonomous unmanned land vehicle, it includes a lot of equipment. By supporting with pictures, it will be shown which electronic and mechanical parts can be connected and how. The sensor used in the picture below is the distance sensor. It is used for the vehicle to operate autonomously from obstacles. Its working logic, on the other hand, can measure the distance by sending beams at a constant speed to the obstacles in front of it.

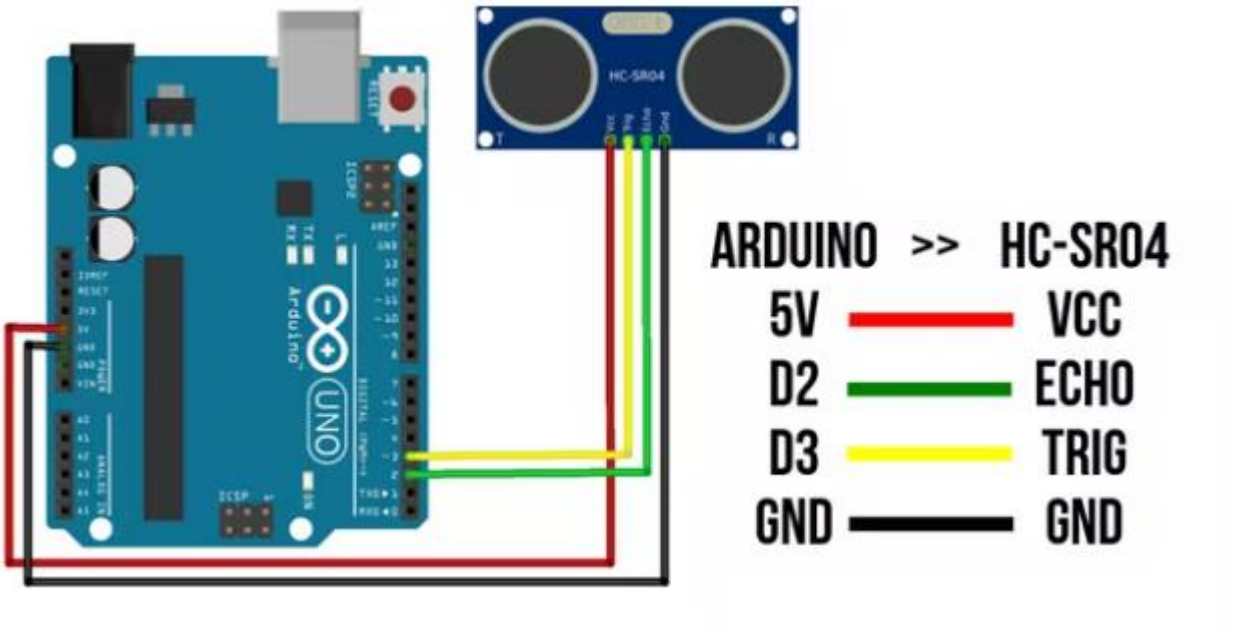


Figure 4 - HC-SR-04

In this section, the necessary Bluetooth module for using our vehicle remotely manually is shown.

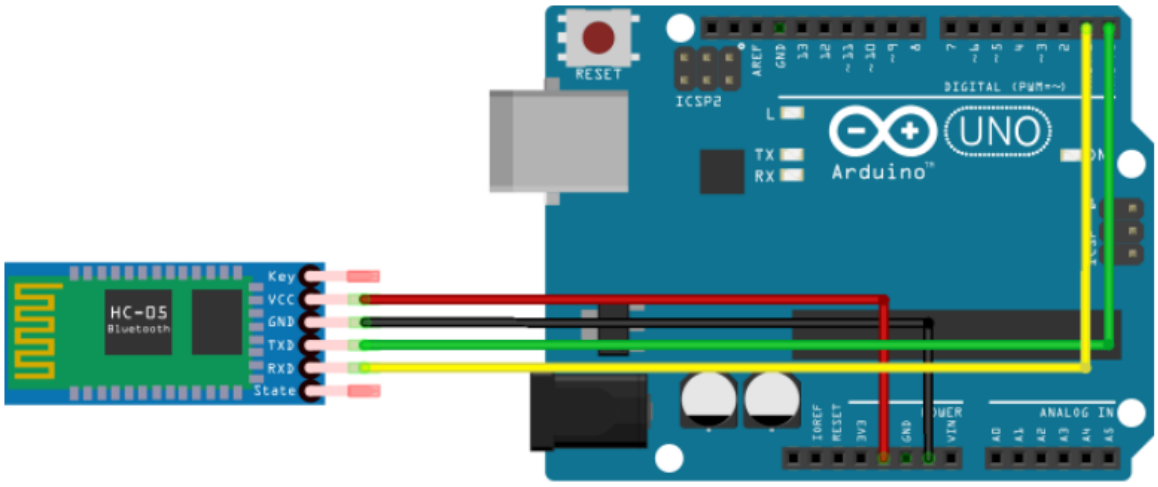


Figure 5 - HC-05

The connections of the motor driver are shown for our vehicle to move. Thanks to the motor driver, we can control the direction and speed of our vehicle thanks to the autonomous mode on the Arduino and the manual driving we provide via Bluetooth.

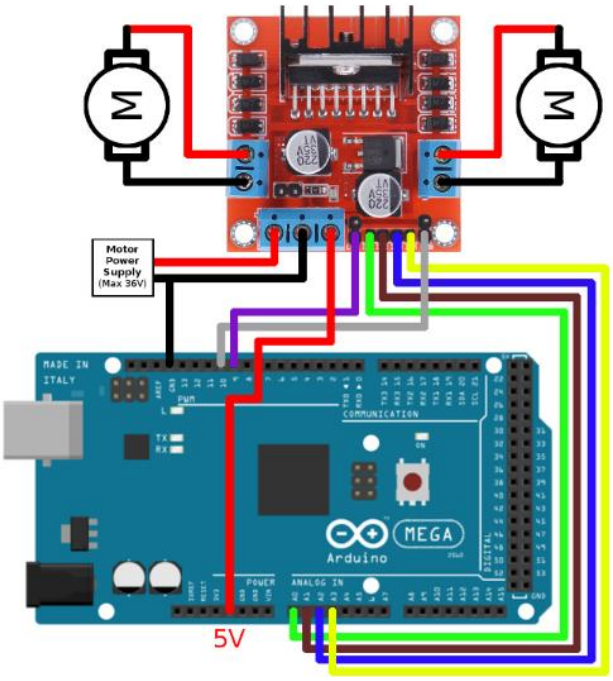


Figure 6 - Motor Driver

The connection of the tool module we use for the connection of the ESP32 camera we use is shown.

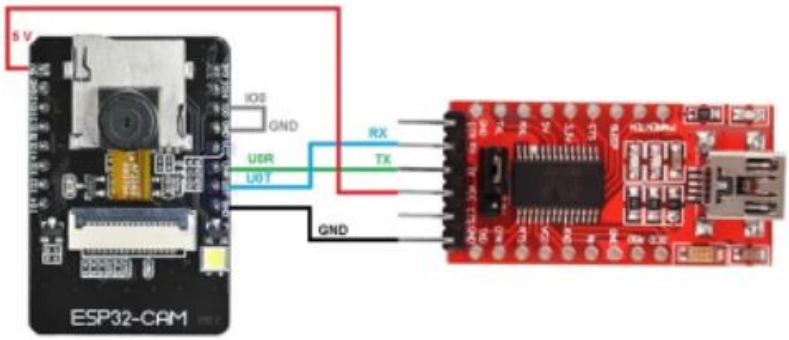


Figure 7 - ESP32 with FTDI

7.4. Software

Our project includes several programming languages and different programming environments. Part of our software includes C, and this part is responsible for movement of the vehicle and receiving & transmitting the data from user. Another important part of our project is written with Java with Android Studio IDE. In this project, we have .xml documents, tflite document and java files. After the source-code is provided, user should get the .apk file of the application to be installed by android device. For that purpose, in Android Studio, we select Build and after that we select Generate Signed Bundle / APK. The related part is given in following figure.

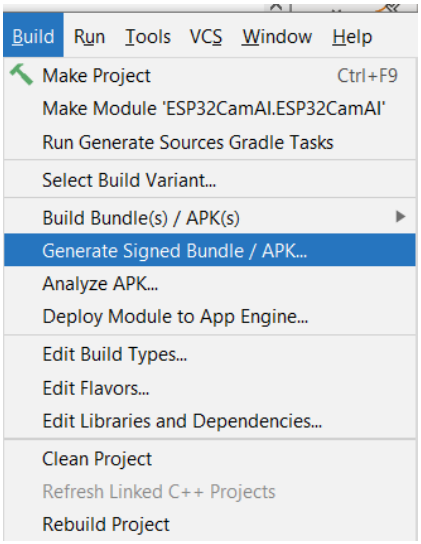


Figure 8

After that, user should observe following frame and from there, s/he should select the APK option.

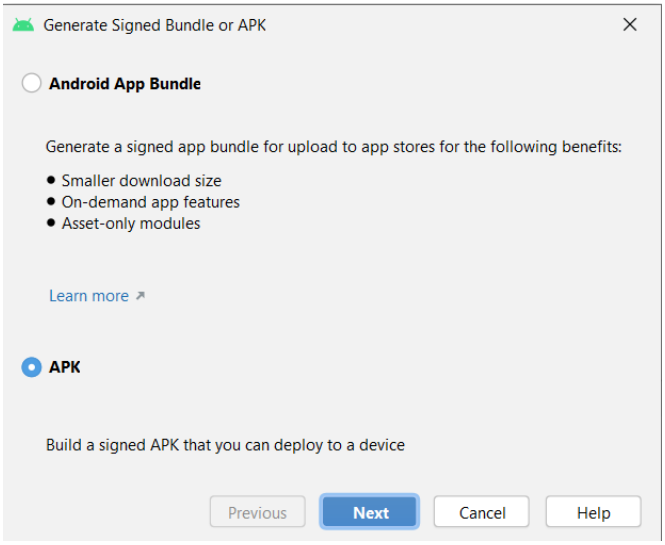


Figure 9

Next, user needs to select the build path.

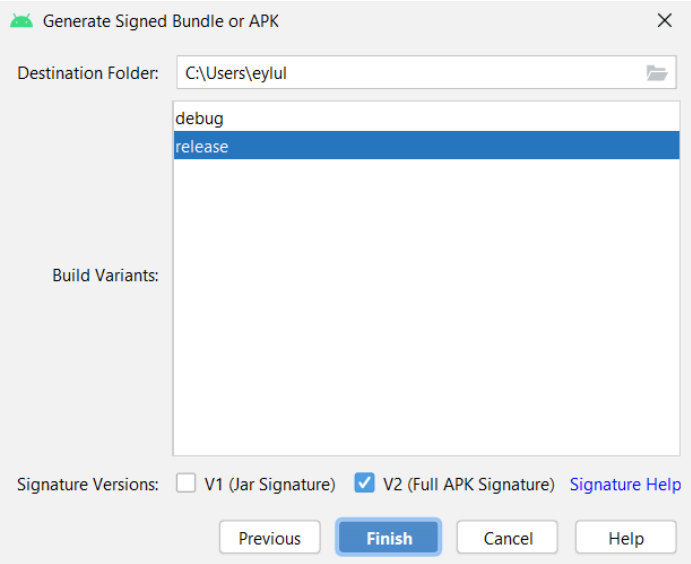


Figure 10

When the .apk file is generated, user should install it to android device. In the application, user will prompt to open Bluetooth of the device. After that, s/he chooses the related HC-05 module. If these steps are handled successfully, user should be observing following figure.

After Bluetooth configuration is handled and the above figure is observed, the ESP-32 CAM is the next module that needs to be configured. To have successful connection, 2.4 Ghz configuration must be provided since ESP-32 CAM is able to communicate with this baud-rate. This configuration can be set up as following steps. First, user should open “Open Network & Network Settings”. In Mobile hotspot section, turn on mobile hotspot. Network band should be configured as 2.4 Ghz. User should observe following figure.

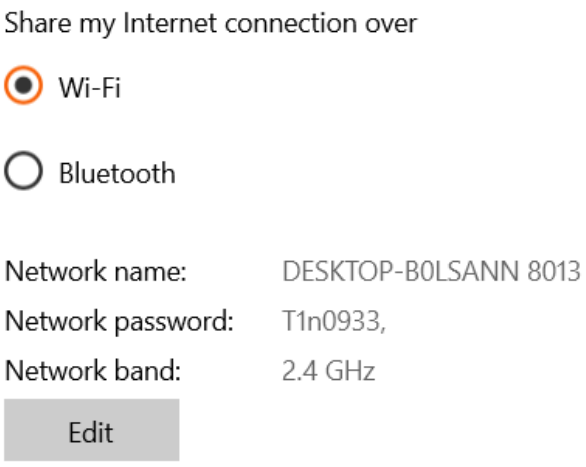


Figure 11

7.5. Upload the Code

For uploading ESP32:

- To be able to upload to ESP32, 4 materials must be provided. These are ESP32, mini usb cable and jumper and FTDI adapter.
- Install ESP32 in Arduino Ide with following these steps:
  File > Preferences

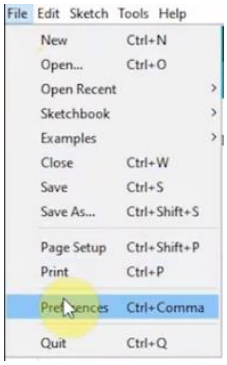


Figure 12

Figure 13

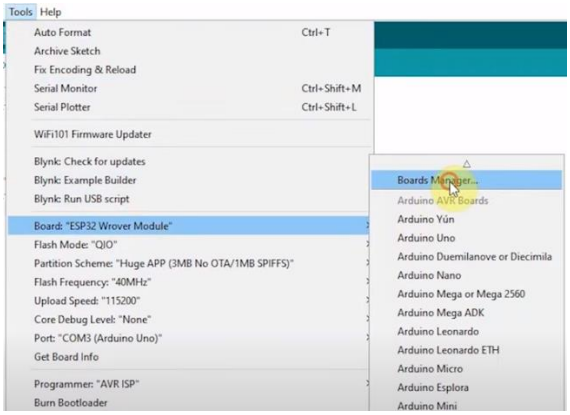


Figure 14

Figure 15

- ESP32 module is selected from the Tools section of Arduino, then the baud rate is set as given in the code.

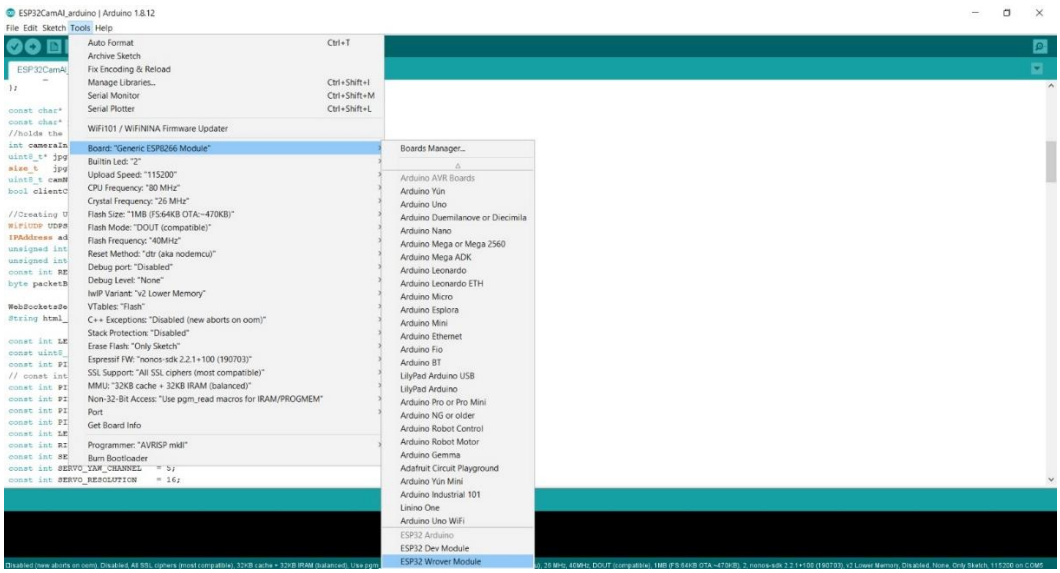


Figure 16

- The port number that the FTDI module is connected to is checked from the device manager and next the port number is selected.
- After the connections are supplied, enter the serial port and press reset once. After that it is checked to see if the ESP32 is powered. If there is no power, it can be tried again by changing the baud rate.

Figure 17

- It should be observed that the code is verified in Arduino Ide.
- In EsP32's hardware, UOT must be connected to RX, UOR must be connected to TX, GND must be connected GND, 5V connected with 5V. It should also be connected GPIO0 to GND, when programming the ESP32. After the program is transferred, unplug USB, disconnect GPIO0 from GND.

- GPO and IO0 must be connected in EsP32's hardware. Check code is uploading or not.

Figure 18

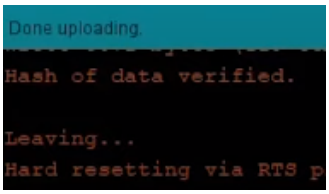


Figure 19

- The serial port opens. Disconnect GPIO0 from GND.

Figure 20

- In serial monitor, baud rate should be selected whichever you set.

Figure 21

- If the real time video is not observed in the application, the above steps should be checked and tried again.

For uploading Arduino:

- The Arduino program opens.
- Click on the Tools menu.
- Used the Arduino board type is selected from the Board section in the Tools menu. Next the port which the card is connected is selected from the Port section in the same menu.

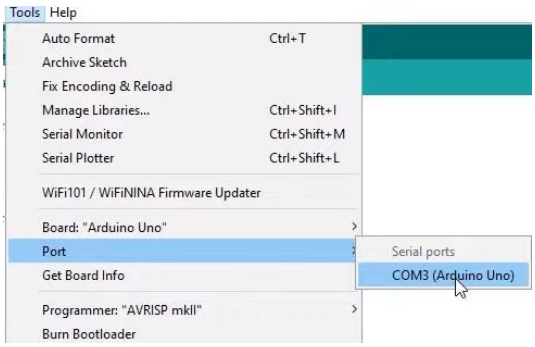


Figure 22

- The RX and TX cables on the Arduino are removed.
- Arduino board is ready to upload code.

- The code can be checked by clicking the Blink sign.

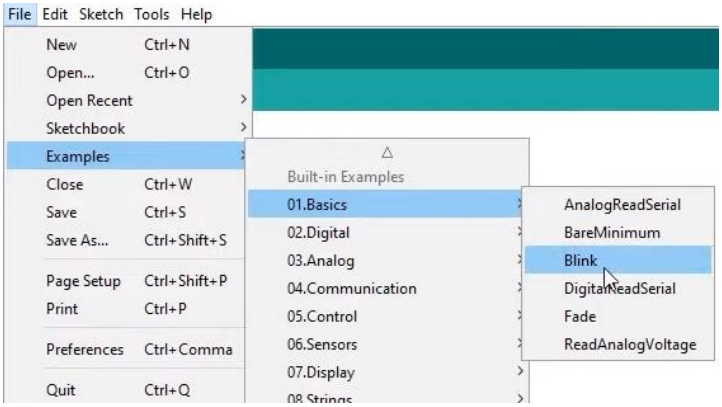


Figure 23

- Thus, it can be observed if there are any errors.
- Accordingly, the code can be loaded by clicking the Ok sign.

Figure 24

- After clicking Ok sign, the message "Done uploading" is observed in the submenu. In this way, the code will be uploaded to the Arduino board.

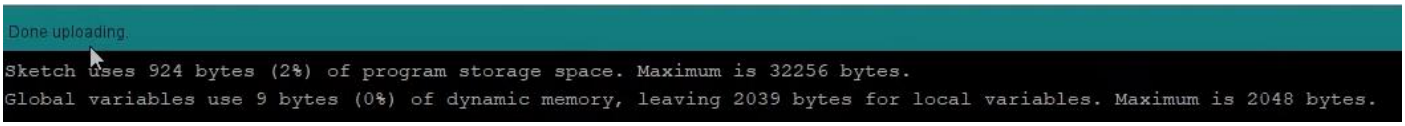


Figure 25

7.6. Start the Project

7.6.1. Set the Application

When the .apk file is generated, user should install it to android device. In the application, user will prompt to open Bluetooth of the device. After that, s/he chooses the related HC-05 module. If these steps are handled successfully, user should be observing following figure.If this figure can be observed, manual and autonomous modes should be performed when the related buttons are used.

Figure 26

If above figure is observed with no error, Bluetooth connection is satisfied successfully. Next, user should check the Camera connection. ESP-32 CAM needs to have 2.4 Ghz. If this configuration is already satisfied, when pushing STREAM button, user should observe the real-time video on the screen. After that, by pressing DETECT button, a box around objects should be appear and user can observe these detections from the device's screen.

7.6.2. Set the Vehicle

It is observed that the power is given by pressing the "Set the vehicle button" on the vehicle.
If the LEDs are not lit, the batteries must be charged.

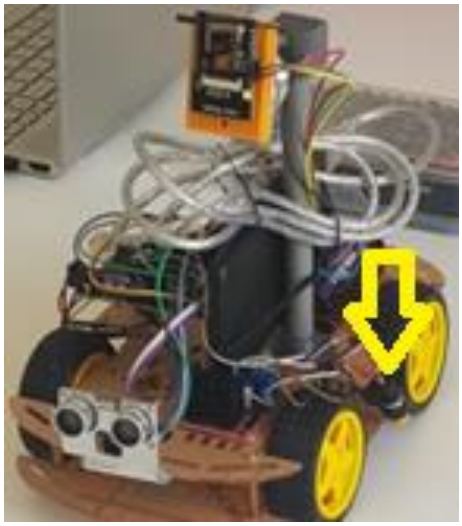


Figure 27

7.6.3. Run the Project

Our vehicle can be controlled with manual and autonomous options thanks to the mobile application we developed over Android Studio.

You can choose which feature to choose by selecting the buttons on the mobile application.

- You can do a manual drive by selecting the buttons here.

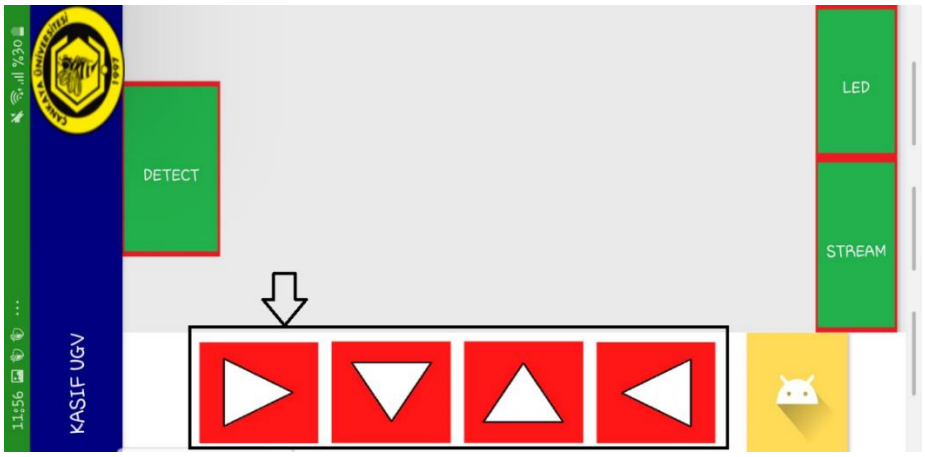


Figure 28

- With this button, you can switch the vehicle to autonomous driving.

Figure 29

- From here you can turn on the led for night driving.

Figure 30

- You can see the image live with this button.

Figure 31

- From here, you can see the vehicle's real-time object detection.

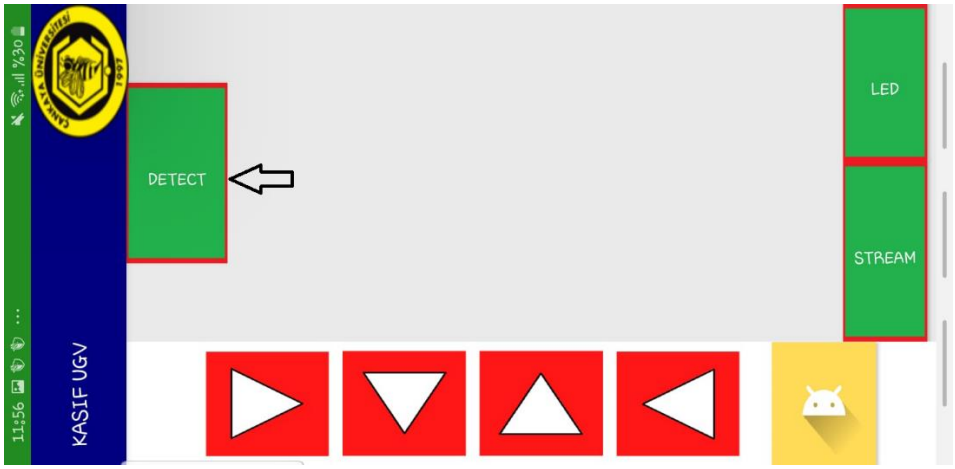


Figure 32