ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

CENG 407

SOFTWARE REQUIREMENT SPECIFICATION REPORT

Aleyna DEDE
201711019

Ali BOZDOĞAN
201717009

Eylül ERDOĞAN
201711025

28.11.2021

Contents

I. 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.

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.

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.

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 threads, 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 threads
- Object Detection for spying
- Capability of autonomous and manual driving

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.

1.5. References

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

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.

2.   Overall Description

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

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 threads. 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

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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. Requirement Specification
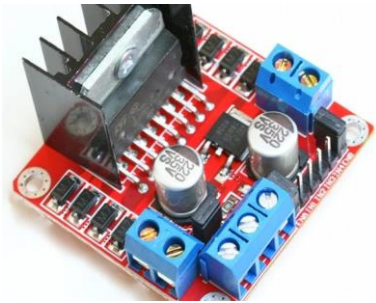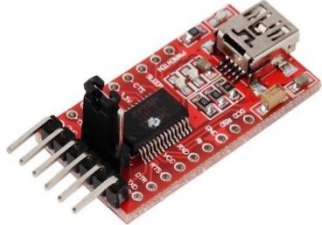
3.1. Interface Requirements

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 thread 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.

3.1.2. 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.

3.1.3. 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.

3.1.4. 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.

3.2. Detailed Description of Functional Requirements

In this part of the report, we stated functional requirements of the project.
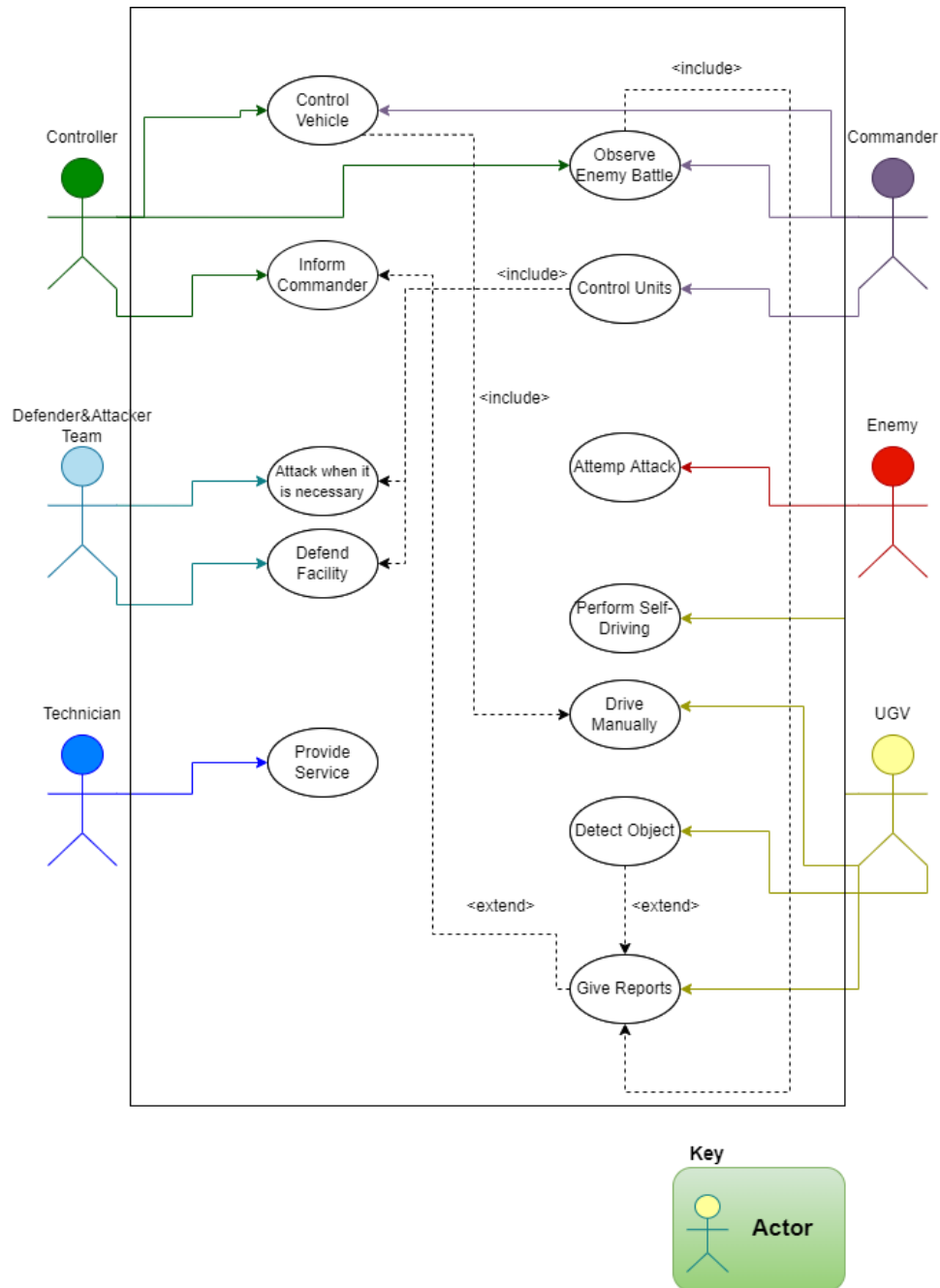
3.2.1. Use Case Diagram

Figure 1: Use Case Diagram

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 threads.
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 thread.
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.

3.2.2.1.   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.

3.2.2.2. Scenario 2: Inform Commander

Use Case Number	2
Use Case Name	Inform Commander
Summary	Commander gets notification or warning about detected threads.
Actor	UGV, Commander, Enemy
Trigger	Thread 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. Thread or enemy move around facility. 4. Thread 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.

3.2.2.3. 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.

3.2.2.4. 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.

3.2.2.5.   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.

3.2.2.6. 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. Thread or enemy move around facility. 4. Thread 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.

3.2.2.7.   Scenario 7: Control Units

Use Case Number	7
Use Case Name	Control Units
Summary	Commander controls units and give commands when there is a thread.
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.

3.2.2.8. 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.

3.2.2.9. 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.

3.2.2.10. 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.

3.2.2.11. 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.

3.2.2.12.   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, thread, 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. • Thread 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 thread 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. Non-Functional Requirements

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

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 thread 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.

3.3.2. 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. 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 threads. Therefore, apk file of application must be also protected.

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.

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].