ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

CENG 408

USER'S MANUAL REPORT

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

Version 1.0

17.05.2021

About This Guide

This document is prepared to help users set up the basic software and hardware development environment for using KAŞİF UGV efficiently. Therefore, our primary goal is to explain how users can easily set up our environment by providing detailed steps and figures.

Release Notes

Date	Version	Release notes
05.06.2022	V1.0	First release.

Table of Contents

1. Introduction

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 an 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.

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

2. Get Started

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.

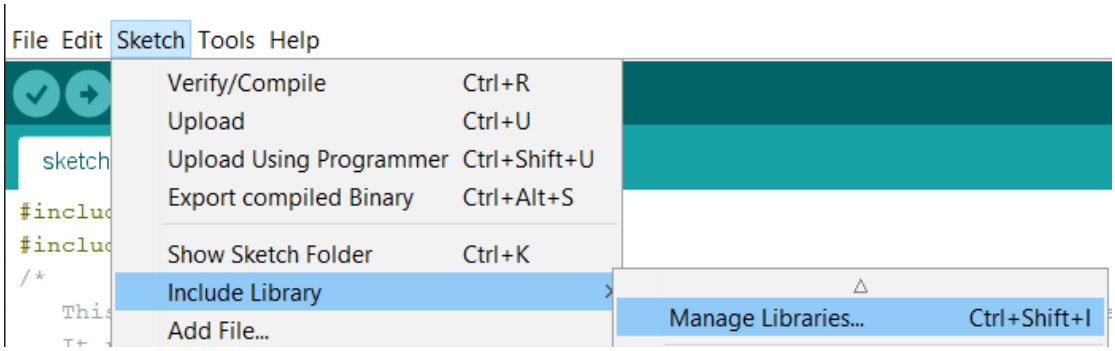

Figure 1

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

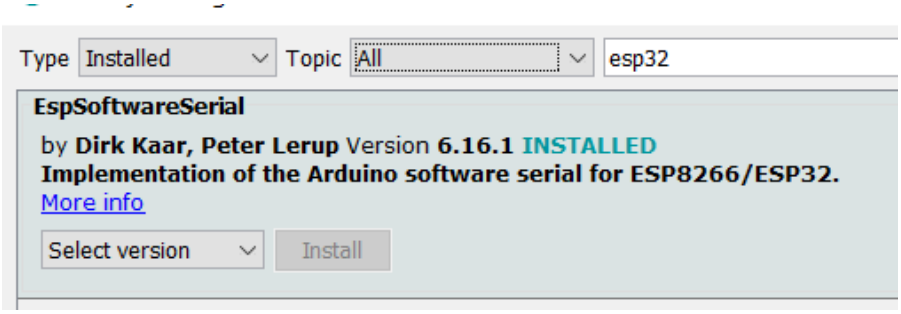

Figure 2

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.

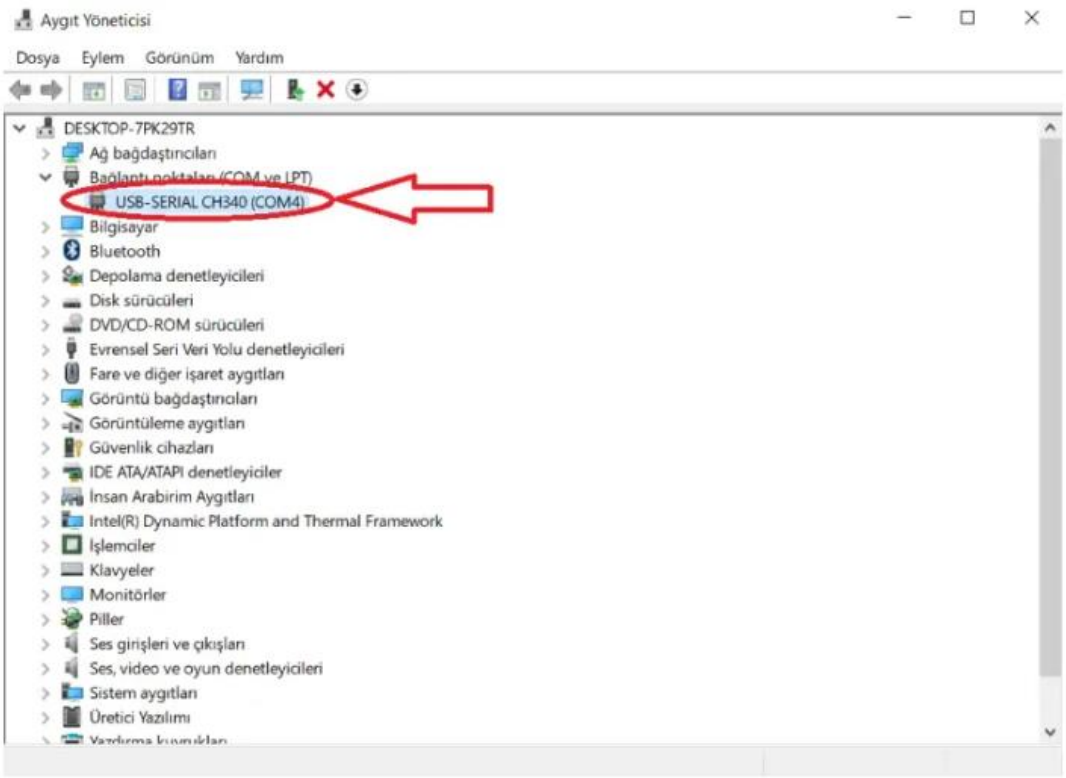

Figure 3

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.

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.

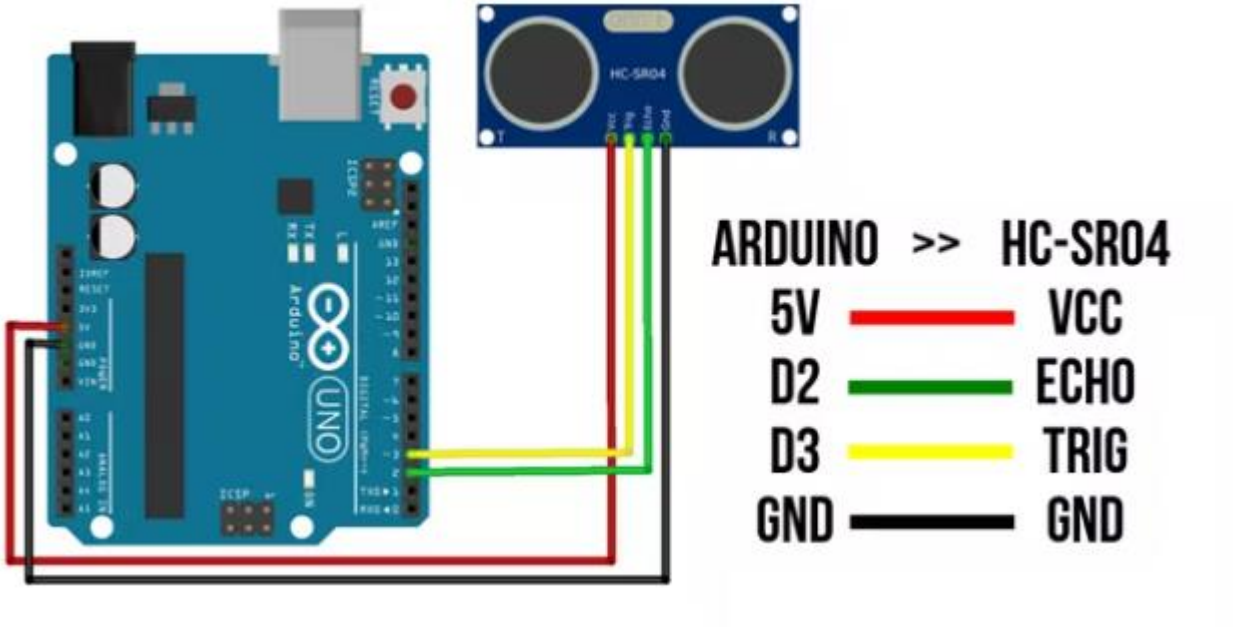

Figure 4 - HC-SR-04

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

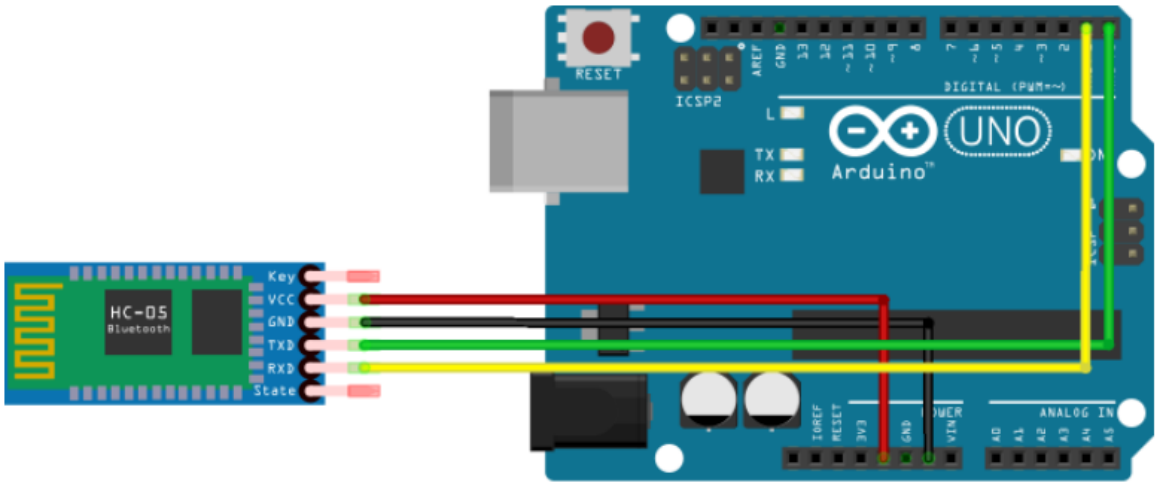

Figure 5 - HC-05

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.

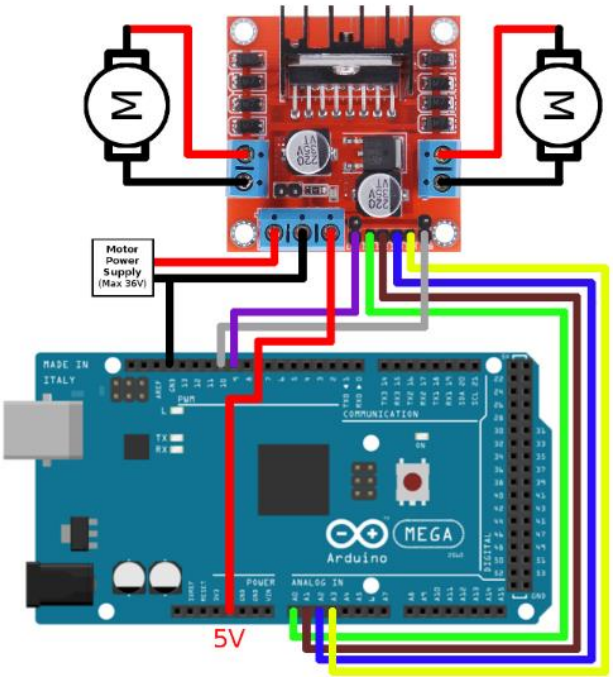

Figure 6 - Motor Driver

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

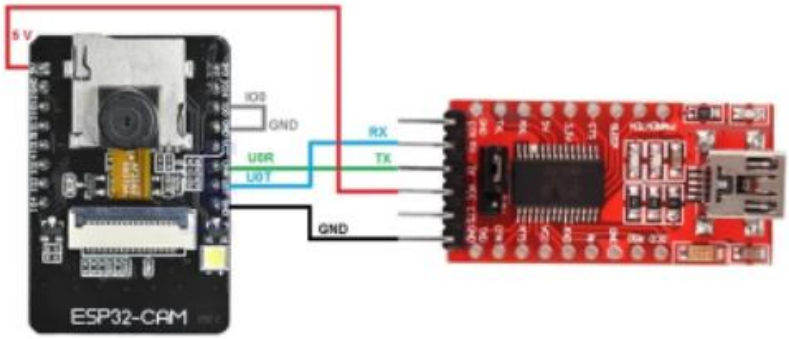

Figure 7 - ESP32 with FTDI

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.

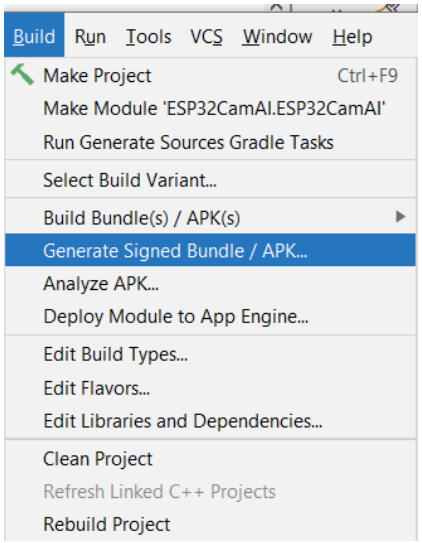

Figure 8

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

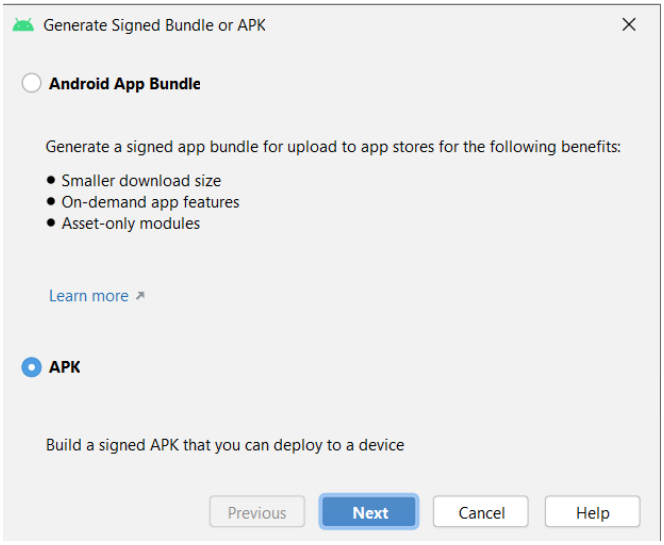

Figure 9

Next, user needs to select the build path.

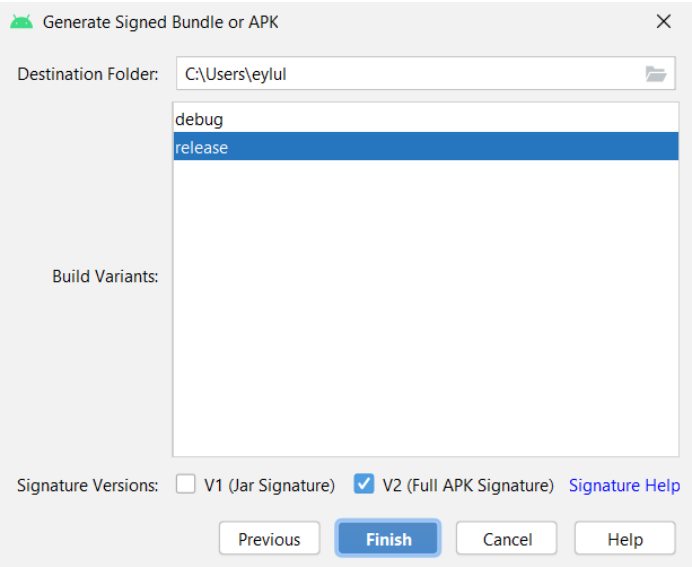

Figure 10

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.

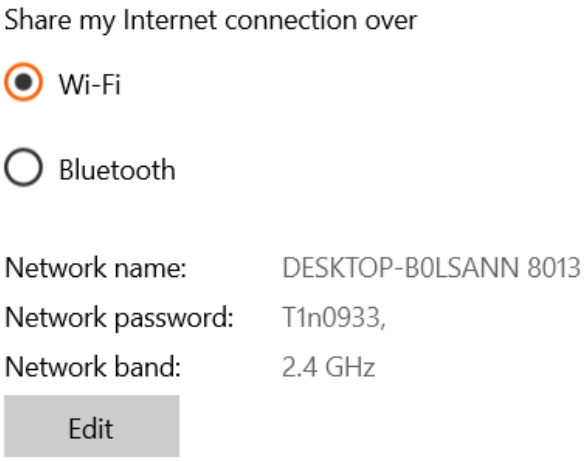

Figure 11

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

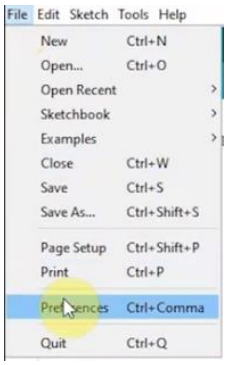

Figure 12

Figure 13

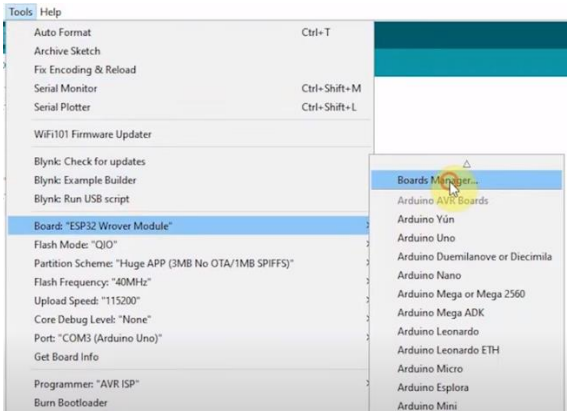

Figure 14

Figure 15

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

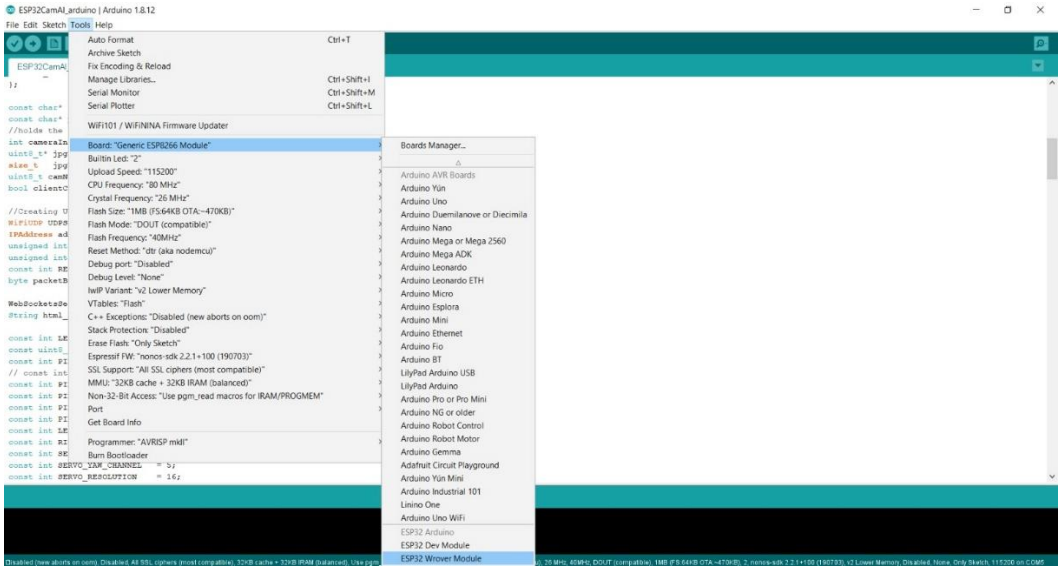

Figure 16

- 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

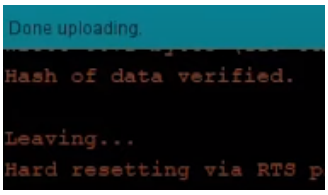

Figure 19

- 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.

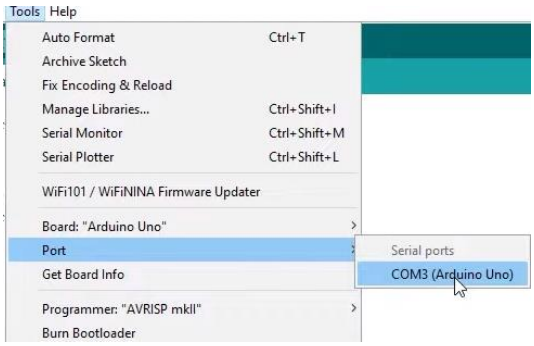

Figure 22

- 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.

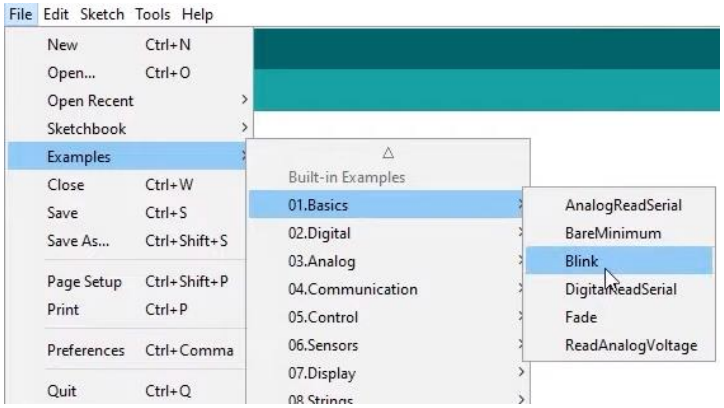

Figure 23

- 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.

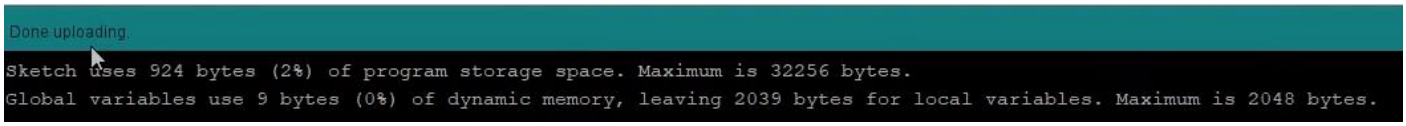

Figure 25

6. Start the Project

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.

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

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.

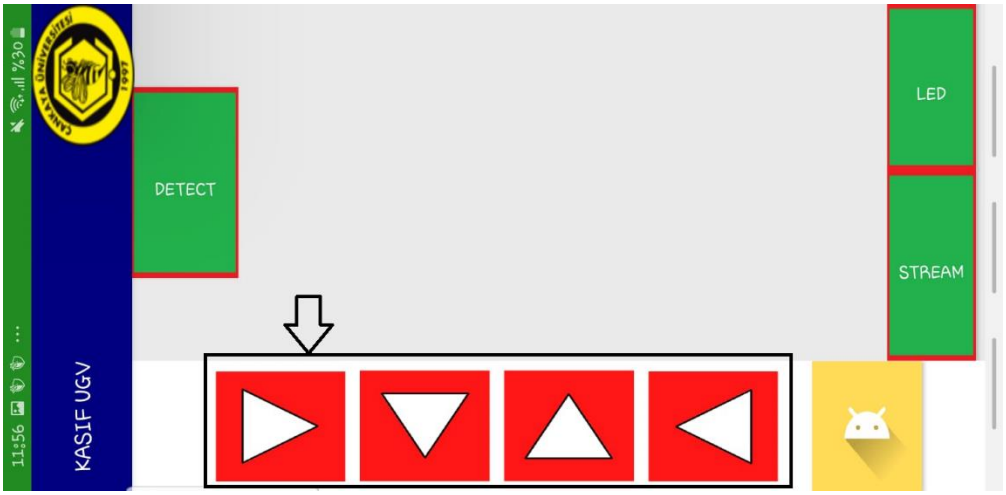

Figure 28

- 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.

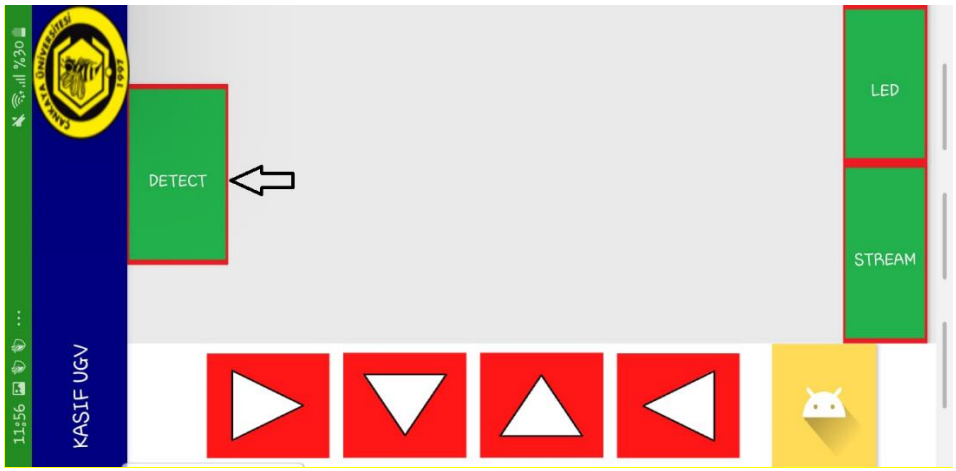

Figure 32