**Project documentation**

**-Wi-fi controlled Arduino robot-**

Ardelean Eugen Richard

Ciubuc Bogdan Alexandru

30433

1. **Description**

Our project presents the following functionalities: through a wi-fi module connected to the robot, a webpage is created having several buttons that represent a certain action that the robot will perform.

* The **FORWARD** button makes the robot go forward, ignoring any possible obstacles.
* The **BACKWARD** button tells the robot to go backward, ignoring any possible obstacles.
* The **LEFT** button tells the robot to spin to its left side.
* The **RIGHT** button makes the robot spin to its right side.
* The **ACTIVATE** button tells the robot to activate its gyroscope and travel around the room without bumping into objects. It constantly checks if there are any obstacles in front of it, and if it detects something, it will go into a measurements states where it checks all its surroundings and decides which path to take, then spins accordingly.

In order to use the robot, a person needs to be connected to the wi-fi network activated by the wi-fi module and access the following site through any browser:

192.168.4.0.

1. **Code explanation**

The first part of the code (up to line 12) includes the needed libraries and defines pins through which the two motors of the robot will be tied. We then define a Servo type object (for the Servo Motor), along with a NewPing one (to use the sensor that measures the distance).

The **setup** method, containing the code that will be executed at the beginning of the upload, does the following things: sets the motors’ pins signals to 0 and declares their mode as **OUTPUT**, initializes the Serial monitor with a baud rate of 115200 then through a series of AT commands, initializes the Wi-fi module with the following connection details: name – ArduinoBwoi, password – 11111111.

The **StartMotor** method, as its name suggests, will be used to start a motor from the robot. It takes as parameters the two pins of a motor tied to the robot, as well as the direction it should spin in, with what speed. What it contains is simple deductive logic that chooses the direction and sends the speed parameter to the pin needed for the motor to spin in said direction. For this, the **digitalWrite** and **analogWrite** methods are used.

The **delayStopped** method is used to stop the spinning of the robot for a time given as parameter. For this stop, the speed and direction of the motors are set to 0, then the delay method is called with the parameter of the method.

A volatile int vector is declared after, and it is used to get the distances that the sensor reads, by doing multiple readings and turning the servo motor as it does this. This functionality is done through the **readDirection** method which gets as parameter the pin the sensor is connected to. We have decided to split this 180 degrees radius that the servo can spin on in 7 parts, delayed with half a second. The servo is then returned to its original position which is at 90 degrees and detached.

The next method, **sendData** is the one that ties the functionality of the robot to the wi-fi module. It gets as parameters the AT command sent to the webpage, the timeout of it and whether we need to debug the said command. It then reads the response from the serial, after printing the command. After that, it checks which webpage we are currently on (each button generates a webpage) and for each of them does the functionality described by that button. For the more interesting functionality, the **ACTIVATE,** we have decided to tell the robot that while it is on that page, it should constantly do that functionality through a while loop. So, the robot constantly reads in front of it when the distance becomes lower than 10, the robot will stop and read its surroundings and select the longest distance it has read. Afterwards the index of the said distance in the volatile int vector described at the previous paragraph is saved and a spin is done based on the said index through a formula, to get to the accurate direction. The choosing of the index is very tricky, as the distance read would often be miscalculated and given the value 0 by the sensor, which we considered to be highest one (through debugging). Then, as we had the value 3 as the middle index of the vector, we compute the rotation with the seen calculus so it does not rotate with a negative angle. If an obstacle is not detected, the robot will simply move forward.

The final method we have included is the necessary **loop** method. Here we test the availability of the Serial and whether the Wi-fi module is active, then compute the commands gotten from the site described in the first paragraph to move the robot according to the buttons. The commands are sent through the **sendData** method described above. The final command of each iteration will be the close command.