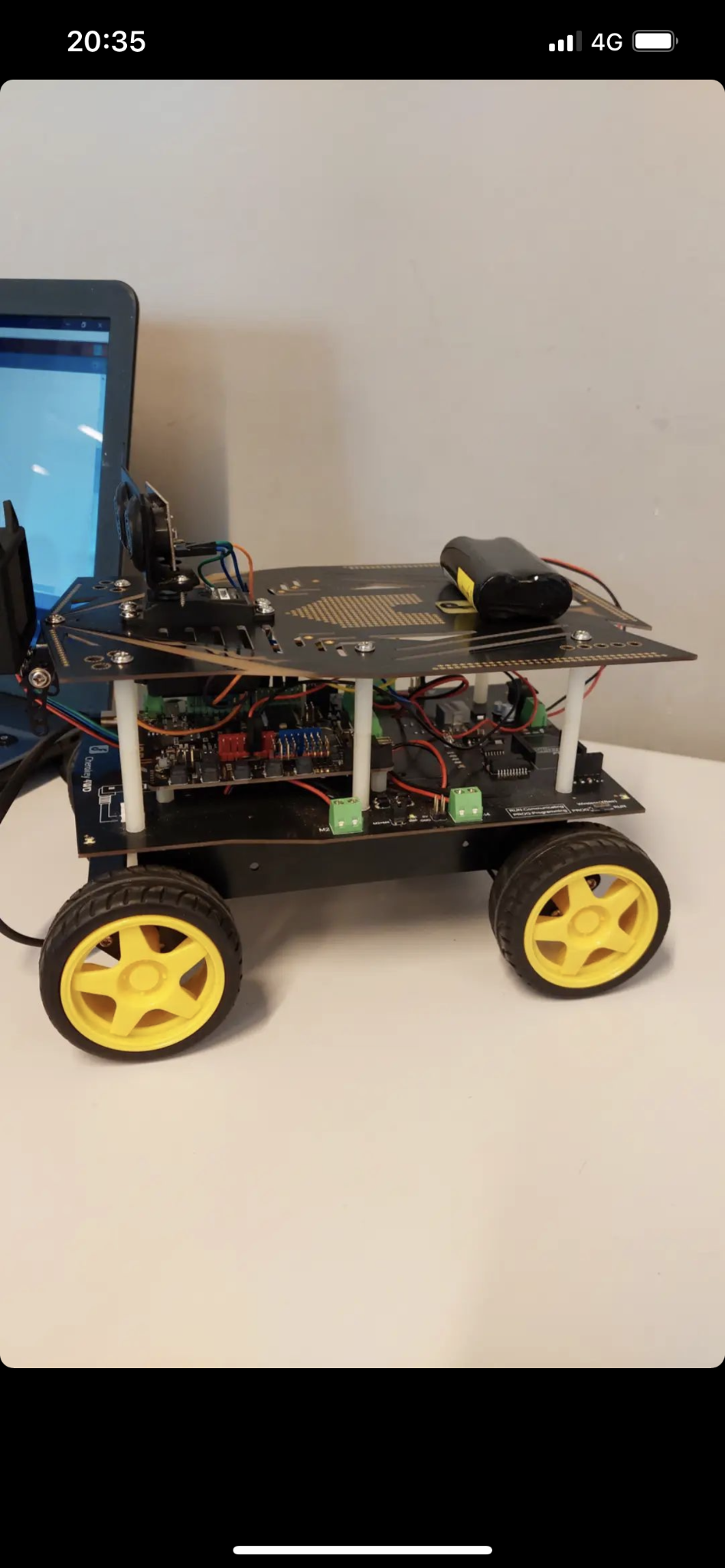
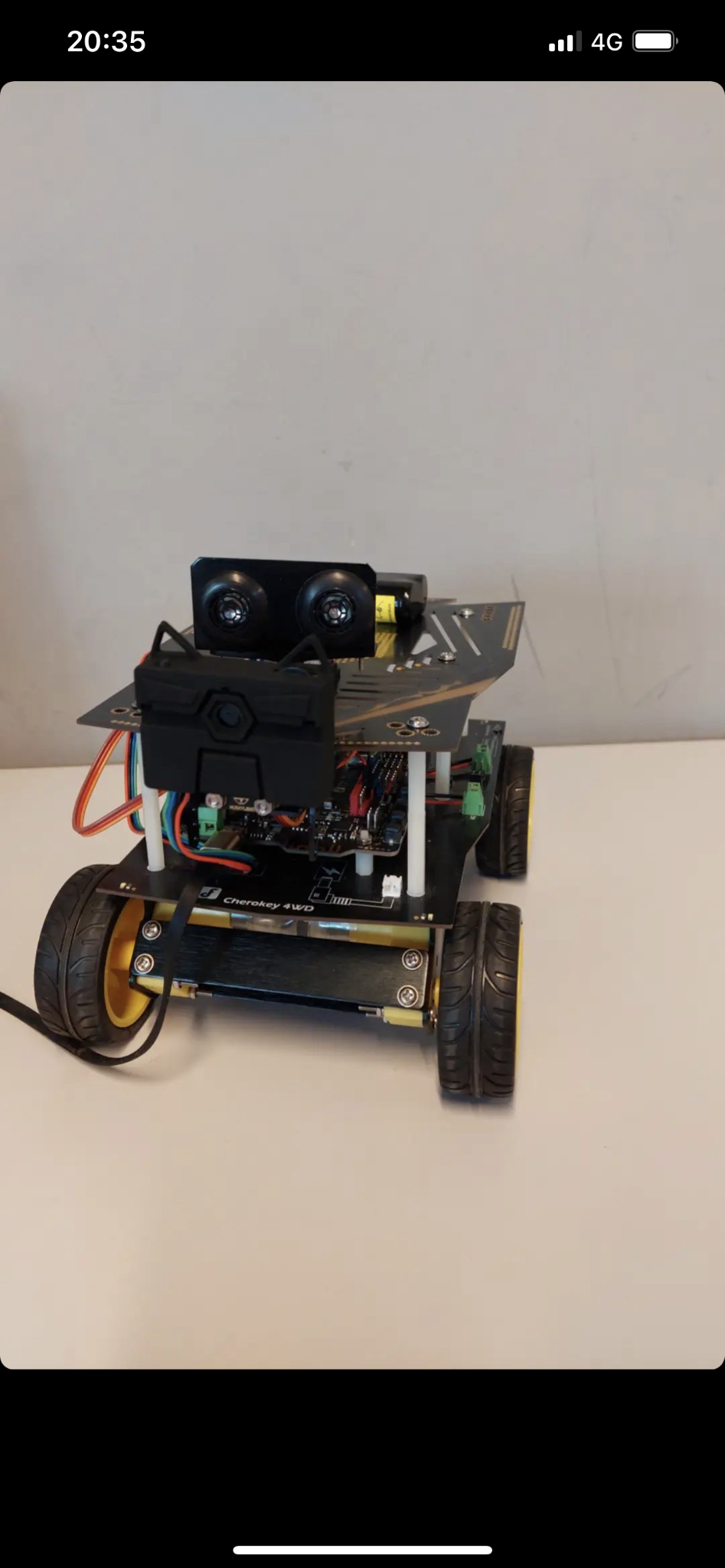
**Sensor and Network**

Arduino Uno

ligne horizontale

# 

# Goals

1. Assemble the car, the servo and the ultrasonic sensor
2. Use it to avoid obstacles
3. Use the app goble and control the car with bluetooth
4. Add a camera to the car and make it follow an object or a color automatically

## 

## I - Assembling

The car is composed of an Arduino Uno motherboard which also contains bluetooth connectivity, the ultrasonic sensor, the Servo motor, the base containing the wheels, the camera, the battery. The servo and the camera are mounted on a further upper base placed above the arduino board and the ultrasonic sensor is mounted above the servo. Subsequently, all the pins necessary for the functioning of the components are connected via cables. The entire system, including the wheels, is powered by the battery, so if the battery is low, the power supply via the cable will not be enough to make the machine move.

### Difficulties

Difficulties during assembly were related to non-functioning or malfunctioning hardware components. In fact, initially the first ultrasound sensor was not working despite being well connected. Subsequently there were problems with the Arduino board because it did not correctly load the code inside the board. Once this problem was solved there was a new problem with a new ultrasound sensor as sometimes it did not detect the distance correctly, so it was replaced again. The last problem we encountered was related to the room as the first room that was mounted was not working. There were minor mounting issues with regards to understanding the guide on how to properly connect the cables to their pins but once you understand the Arduino uno board and its structure all the minor issues were easily solved.

## II - Obstacle avoidance

### Our method

This task consisted in creating a technique that made sure that, through the use of the ultrasonic sensor, the machine was able to detect obstacles and avoid them. To do this we mounted the sensor on a Servo motor in such a way that the sensor could rotate and then we managed the behavior of the car by code. In the setup function the serial is initialized and position 60 is set to the Servo, this position indicates that the servo points exactly in front of it. In the loop function we use the MeasureDistance () function which allows to obtain the distance between the sensor and the obstacle and we check if the distance is less than 40, if yes, the machine stops moving and the sensor begins to scan the surrounding environment to if to understand where to go to dodge the obstacle. The servoSweep () function is used to do this. This function makes the Servo move 10 degrees at a time, first from 60 to 120, then from 60 to 0. The servo stops moving when it finds a point where the calculated distance between the sensor and the obstacle is greater than 40. At this point, check if the position of the Servo is less than 60, if so the car curves to the right, otherwise it curves to the left. Then the Servo is reset to the starting position, that is 60 and the machine is made to go forward. As for the choice of the distance to identify an obstacle as "close" we decided to set the value 40 because through tests we have seen that using a delay of 200 inside the loop, 40 is the best value that allows the machine to dodge obstacles as best you can.

### Difficulties

The difficulties encountered regarding the development of the algorithm used to avoid obstacles are related to the fact of understanding what is the best way to make the car quickly realize that there is an obstacle in its path and that he has to change direction to avoid it. In fact, since we used a single sensor that initially points to 60 degrees, i.e. in front of you you have to find the correct value in order not to generate false positives, i.e. the case in which the car thinks there is an obstacle when in reality there is not, or false negatives, i.e. the case in which the car does not notice the obstacle. For this reason it is much more convenient to make the car walk at a speed that is not too high so that the sensor has time to understand what it has in front of it and to set a correct distance value which in our case is 40. Another difficulty concerns the technique to avoid obstacles. In our case we decided to scan the surrounding environment and go in the first direction which has a distance greater than 40. In our opinion this is the best technique because even if the machine initially goes in the wrong direction it immediately realizes that in that direction there is another obstacle and through trial and error finds the best direction to dodge the obstacles.

## III - Car with remote control

### Our method

For this project we used the Goble Library and his app to control the car. The functioning is really simple, we used the coordinates of the joystick to do the rotations of the wheels depending on that.

### Difficulties

The main difficulty we encountered came from the initial way of moving, when you want to go on the left for example, the car will stop, rotate left and move on, but this technique is a waste of time if we want to make small adjustments.

### Bonus feature

About the difficulties, we go over it, by implementing the possibilities to take curves. To do that we detect when the user is pressing the joystick forward, and at the same time moving it a bit on left or right, and with that info we will create a little difference between the two wheels to rotate while we are moving forward.

We also did a switching mod to be able to change between the remote control mod and the avoid obstacles one.

## III - Object and color detection

### Our method

To work with the camera functionalities we use the “HUSKYLENS” library. As long as we have a learned object, the camera will try to detect it, and if the object is in the center of the camera the car will go forward, if the object is on the left of the camera it will go left and the same for right. We succeed in doing this by getting the coordinates of the object detected by the camera.

### Difficulties

We didn't have so many difficulties for this part, except when fixing the camera to the car. Because it was making the car turn off, but this issue was fixed by changing the camera.

## III - Conclusion

In this report the methodologies used for the development of different algorithms applied on an Arduino Uno board mounted on a robot machine have been discussed. We have seen an automatic obstacles avoid algorithm that using an ultrasonic sensor is able to recognize obstacles in front of its path and avoid them, a remote control that through the goble library allows you to drive the car using the smartphone by connecting the machine to it via bluetooth and one that uses a camera with an artificial intelligence inside capable of recognizing colors, objects, faces which, through code adjustments, allows the machine to follow the object it has learned. The results of this project are very good as all the algorithms work, it would certainly be possible to make adjustments in such a way that, for example for remote control, the machine bends faster but in general the algorithms are efficient and well developed.