Universidad Politécnica de Madrid

Athens Programme March 2017

UPM115 Physical Computing based on Open Software and Hardware Platforms

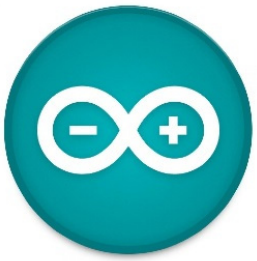
Professors :

Antonio Pérez-Serrano

Xabier Quintana Arregui

Morten Andreas Geday

Francisco J. López Hernández



By :

Majed Somai

Mohamed Mahdi

Overview

During the UPM115 Athens course, we had the opportunity to acquire knowledge in the different existing available possibilities to create projects according to our necessities : in the first days we studied the theory behind electrical system components. At each evening we had practical sessions where we built small circuits controlled by applications based on the Arduino. We also learnt to build interactive interfaces that control circuits based on Arduino.

At the end of the week, we had two days to make a project with the available materials : we choose to build a Wifi controlled robot that could detect obstacle and send back the information to the user. The project included two graphical interfaces : one to control the robot and another to visualise the data sent by the robot. The robot could also be controlled using the keyboard. The same component works at 3.3V and other 5V : so we had to make some kind of voltage regulator to get the needed voltage.

Presentation of the robot

To make our robot we mainly used 2DC motor and wheels, the ESP8266EX Wifi module, two capacitors, two 9V batteries and a motor controlling card. The frame of the robot, some kind a card, has been given us by our professors so we could concentrate on programming our components separately and integrate them to the robot. At the end, this is how our robot looked like :

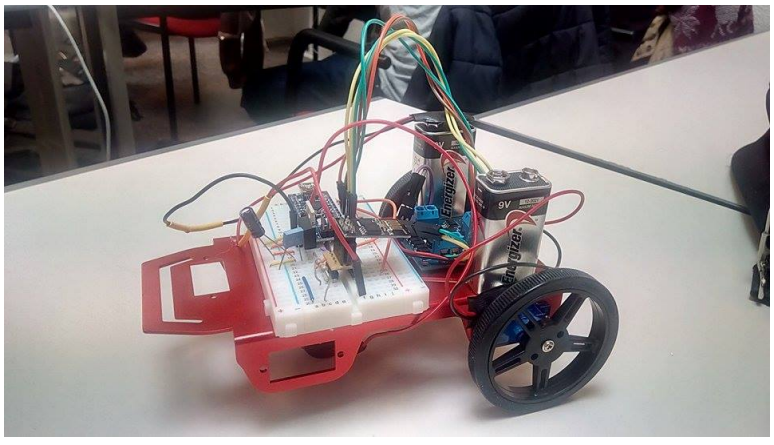


FIGURE 1 – Final version of the robot

This figure shows the schematic view of the circuit :

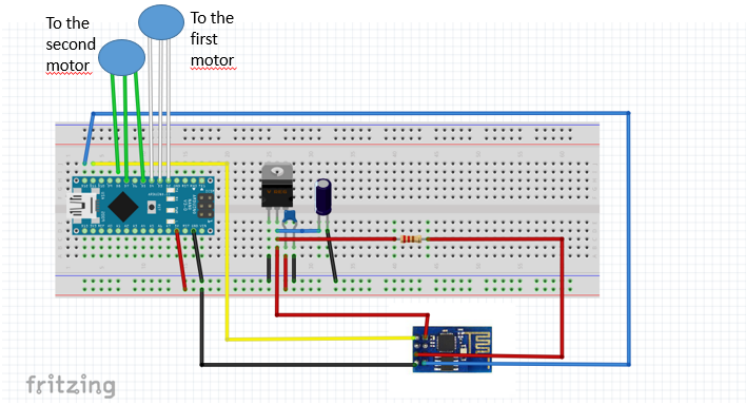


FIGURE 2 – Electronic circuit

Graphical interface controlling the robot

This interface controls the motors of the robot. The two sets speed buttons can be used to instantly change the speed of each motor : the speed can be controlled using the textboxes and the direction is controlled by the checkbox. These modules have mainly been developed to test and debugging purposes. The stop button instantly stops the robot and the slider adjusts the target speed of the robot.
As for the middle controller it can be used to steer the motor

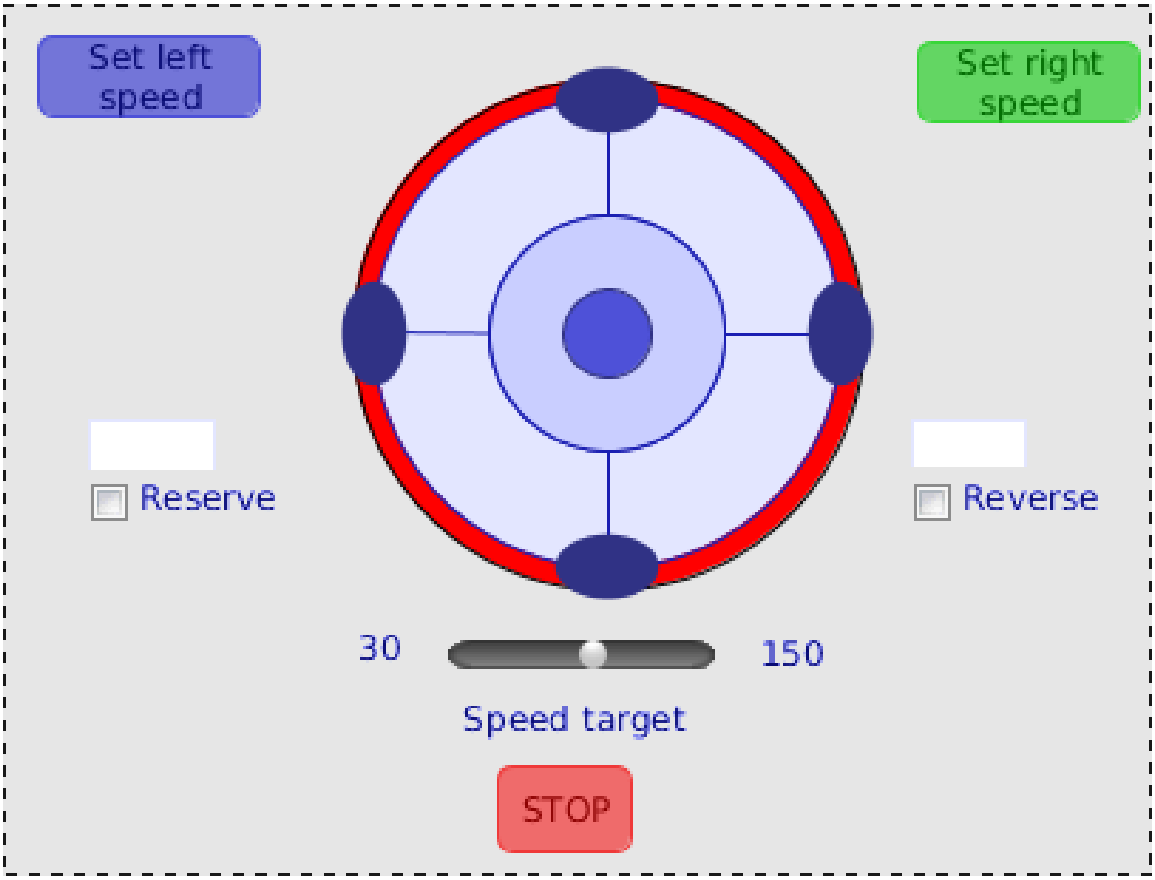


FIGURE 3 – Motor controlling GUI

When we move the controller, the speed of each motor adjusts in order to make it turn in the pointed direction. For now we have only implemented the four basic directions : front, rear, right, left but we could easily add more by getting the position of the controller more precise.

Source code of the GUI

This application has been made using the G4P library :

```
public void leftButton_click(GButton source, GEvent
   event) {

 // int a =Integer.valueOf(speedL.getText());
 int Lspeed = Integer.valueOf(speedL.getText());
 char Lvalue=(char)Lspeed;
    //println("leftButton - " +" :"+Lvalue );
```

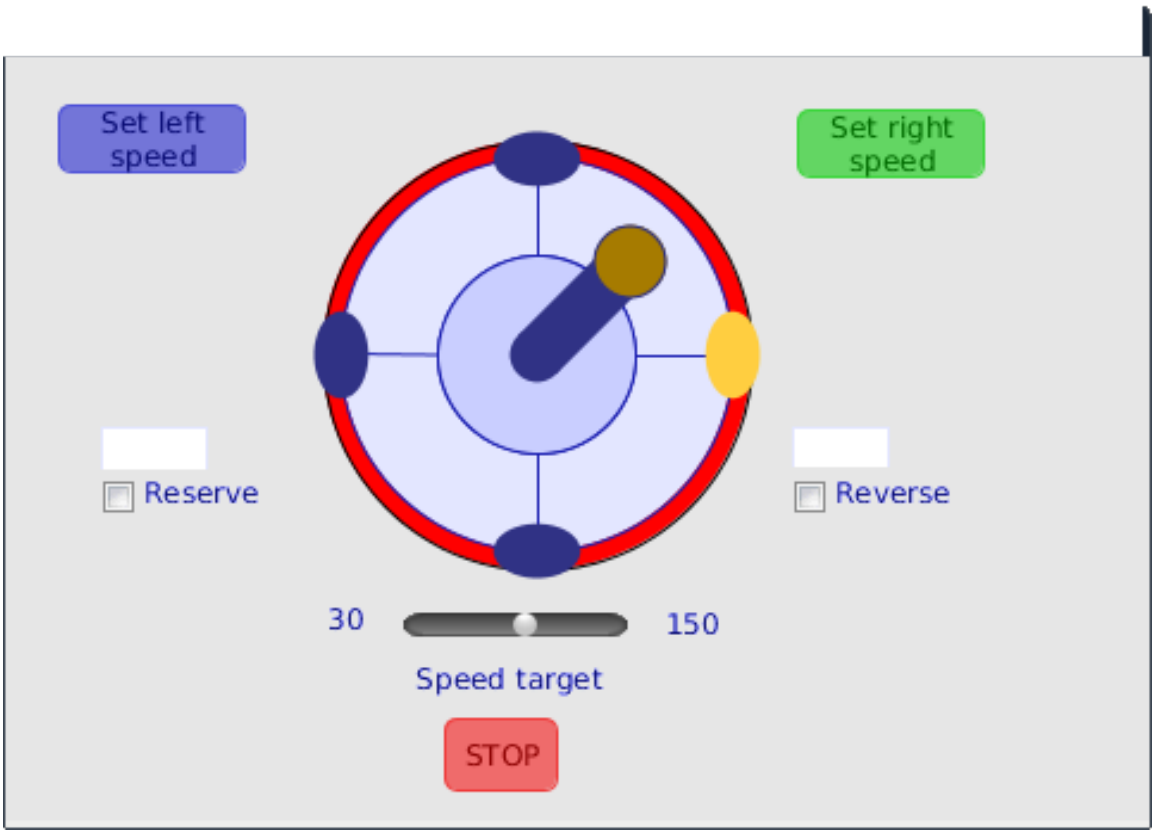


FIGURE 4 – Exemple of robot steering

```
    if(reverseL.isSelected ())
     left_dir='r';
    else left_dir='f';

   sendMotorSpeed('l',Lvalue,left_dir,(char)0,(char)0);
 /* if(reverseL.isSelected ())
    println("Leftchecked");
    else  println("LeftNotchecked");*/
}

public void rightButton_click(GButton source, GEvent
   event) {

 int Rspeed = Integer.valueOf(speedR.getText());
 char Rvalue=(char)Rspeed;
```

```
 if(reverseR.isSelected ())
     right_dir='r';
    else right_dir='f';

 sendMotorSpeed('r',Rvalue,right_dir,(char)0,(char)0);
}



public void controller_change1(GStick source, GEvent
   event) {

    previousPos = currentPos;
    currentPos =controller.getPosition();

    char dir='f';
    println("pos = "+currentPos);

   switch (currentPos){
      case 0: if(previousPos==6) dir ='f';
              else if(previousPos==2) dir='r';
                  sendMotorSpeed('l',(char)maxSpeed,dir,(char)1,(char)(ma
              break;
      case 6: dir ='f';
          sendMotorSpeed('a',(char)maxSpeed,dir,(char)1,(char)0);
            break;
      case 2: dir='r';
          sendMotorSpeed('a',(char)maxSpeed,dir,(char)1,(char)(maxSpeed/2
              break;
      case 4: if(previousPos==6) dir ='f';
              else if(previousPos==2) dir='r';
                 sendMotorSpeed('r',(char)maxSpeed,dir,(char)1,(char)(max
               break;
 }

}
public void stop_click(GButton source, GEvent event) {
  sendStopMotor('a');
}


public void speedSlider_change1(GCustomSlider source,
   GEvent event) {
```

```
   int Rspeed = (int)speedSlider.getValueI();
  maxSpeed=(char)Rspeed;

}

public void createGUI(){
  G4P.messagesEnabled(false);
  G4P.setGlobalColorScheme(GCScheme.BLUE_SCHEME);
  G4P.setCursor(ARROW);
  surface.setTitle("Sketch Window");
  leftButton = new GButton(this, 22, 20, 80, 30);
  leftButton.setText("Set left speed");
  leftButton.addEventHandler(this,
     "leftButton_click");
  rightButton = new GButton(this, 332, 22, 80, 30);
  rightButton.setText("Set right speed");
  rightButton.setLocalColorScheme(GCScheme.GREEN_SCHEME);
  rightButton.addEventHandler(this,
     "rightButton_click");
  speedL = new GTextField(this, 40, 155, 45, 18,
     G4P.SCROLLBARS_NONE);
  speedL.setOpaque(true);
  speedL.addEventHandler(this, "textfield1_change1");
  speedR = new GTextField(this, 330, 155, 41, 17,
     G4P.SCROLLBARS_NONE);
  speedR.setOpaque(true);
  label1 = new GLabel(this, 25, 83, 80, 20);
  label1.setTextAlign(GAlign.CENTER, GAlign.MIDDLE);
  label1.setOpaque(false);
  label2 = new GLabel(this, 323, 93, 80, 20);
  label2.setTextAlign(GAlign.CENTER, GAlign.MIDDLE);
  label2.setOpaque(false);
  label3 = new GLabel(this, -24, 134, 80, 20);
  label3.setTextAlign(GAlign.CENTER, GAlign.MIDDLE);
  label3.setOpaque(false);
  label4 = new GLabel(this, 71, 133, 80, 20);
  label4.setTextAlign(GAlign.CENTER, GAlign.MIDDLE);
  label4.setOpaque(false);
  label5 = new GLabel(this, 103, 226, 80, 20);
  label5.setTextAlign(GAlign.CENTER, GAlign.MIDDLE);
  label5.setText("30");
  label5.setOpaque(false);
  label6 = new GLabel(this, 248, 227, 80, 22);
  label6.setTextAlign(GAlign.CENTER, GAlign.MIDDLE);
```

```
  label6.setText("150");
  label6.setOpaque(false);
  controller = new GStick(this, 108, 30, 230, 190);
  controller.setMode(G4P.X4);
  controller.setOpaque(false);
  controller.addEventHandler(this,
     "controller_change1");
  stopButton = new GButton(this, 184, 277, 49, 32);
  stopButton.setText("STOP");
  stopButton.setLocalColorScheme(GCScheme.RED_SCHEME);
  stopButton.addEventHandler(this, "stop_click");
  reverseL = new GCheckbox(this, 40, 175, 120, 20);
  reverseL.setIconAlign(GAlign.LEFT, GAlign.MIDDLE);
  reverseL.setTextAlign(GAlign.LEFT, GAlign.TOP);
  reverseL.setText("Reserve");
  reverseL.setOpaque(false);
  reverseL.addEventHandler(this,
     "reverseCheckL_clicked1");
  reverseR = new GCheckbox(this, 330, 175, 120, 20);
  reverseR.setIconAlign(GAlign.LEFT, GAlign.MIDDLE);
  reverseR.setTextAlign(GAlign.LEFT, GAlign.TOP);
  reverseR.setText("Reverse");
  reverseR.setOpaque(false);
  reverseR.addEventHandler(this, "reverseL_clicked1");
  label7 = new GLabel(this, 155, 251, 113, 20);
  label7.setTextAlign(GAlign.CENTER, GAlign.MIDDLE);
  label7.setText("Speed target");
  label7.setOpaque(false);
  speedSlider = new GCustomSlider(this, 164, 218,
     100, 40, "grey_blue");
  speedSlider.setLimits(60, 30, 150);
  speedSlider.setNumberFormat(G4P.INTEGER, 0);
  speedSlider.setOpaque(false);
  speedSlider.addEventHandler(this,
     "speedSlider_change1");
}

// Variable declarations
// autogenerated do not edit
GButton leftButton;
GButton rightButton;
GTextField speedL;
GTextField speedR;
GLabel label1;
```

```
GLabel label2;
GLabel label3;
GLabel label4;
GLabel label5;
GLabel label6;
GStick controller;
GButton stopButton;
GCheckbox reverseL;
GCheckbox reverseR;
GLabel label7;
GCustomSlider speedSlider;
```

In order to make the application communicate with the robot we established this client-server architecture :

```
String msg;
import processing.net.*;
Server myServer;

void setupSerial(){

myServer = new Server(this, 5204);
}

void sendMotorSpeed(char left_right,char speed_,char
   advance_,char motorNbr,char speed_2){//

  msg="m"+left_right+advance_+speed_+motorNbr+speed_2;
     //motorNbr ==0 control 1 motor 1 control 2

  myServer.write(msg);

}
void sendStopMotor(char lef_right){//left right or all
  msg="s"+lef_right;
  myServer.write(msg);
}
```

The message sent to the Arduino is decoded in the programming running on the robot. Finally, this is the code of the main application that enables the control of the robot via the Keyboard

```
// Need G4P library
import g4p_controls.*;
```

```
char left_dir,right_dir;
int previousPos ,currentPos;
char maxSpeed=120;
int controllerX = 108+230/2, controllerY=30+ 190/2;
int controllerR = 180;
 void setup(){
  size(480, 320, JAVA2D);
  setupSerial();
  createGUI();
  customGUI();


}


public void draw(){

}


public void customGUI(){

}
void keyPressed() {
  boolean b=true;
  previousPos = currentPos;
  if (key == CODED) {
    if (keyCode == UP) {
      currentPos=6;
    } else if (keyCode == DOWN) {
      currentPos=2;
    }
    else if (keyCode == LEFT) {
    currentPos=4;
    }
    else if (keyCode == RIGHT) {
    currentPos=0;
    }
  } else if (keyCode == ENTER){
    sendStopMotor('a');
    b=false;
  }

    char dir='f';
```

```
 println("pos = "+currentPos);
 if(b){
 switch (currentPos){
  case 0: if(previousPos==6) dir ='f';
              else if(previousPos==2) dir='r';
//println("speed = "+speedT +" pos?="+currentPos);
  sendMotorSpeed('l',(char)maxSpeed,dir,(char)1,(char)(maxSpeed/3));
  //sendStopMotor('l');
  break;

  case 6: dir ='f';
    sendMotorSpeed('a',(char)maxSpeed,dir,(char)1,(char)0);
    break;

  case 2: dir='r';
      sendMotorSpeed('a',(char)maxSpeed,dir,(char)1,(char)(maxSpeed/2));

      break;
  case 4: if(previousPos==6) dir ='f';
              else if(previousPos==2) dir='r';
 sendMotorSpeed('r',(char)maxSpeed,dir,(char)1,(char)(maxSpeed/3));
//sendStopMotor('r');
  break;
 }
 }
}
```

Embedded code

The embedded code is composed of two main elements : the Wifi and a motor module. The main code is very basic : it setups the Arduino and waits for messages coming from the Wifi.

```
/*#########################################################################
 Author:
 * Majed SOMAI (04-2017)



 Connections:
 * BOARD -> ARDUINO
```

```
 * 1A    -> 2
 * 1B    -> 4
 * E1    -> 3
 * 2A    -> 7
 * 2B    -> 8
 * E2    -> 5
##############################################################################

#include "MotorControl.h"
#include "Serie.h"

// Initialization
void setup()
{
   setUpIOPins();
   setupSerial();
}


// main loop
void loop()
{
ReadMessage();

}
```

The module controlling the motors :

```
/*############################################################################
 Author:
 * Majed SOMAI (04-2017)

##############################################################################


#ifndef TWOMOTORCONTROL_H
#define TWOMOTORCONTROL_H
#include "MotorControl.h"
// Define constants and variables

const int advancePinRight = 7;//4

const int revervePinRight = 8;//2

const int RightMotorPin = 5;//3
```

```
const int advancePinLeft = 4;//7

const int reversePinLeft = 2;//8

const int leftMotorPin = 3;//5


unsigned char pwmRight = 0;
unsigned char pwmLeft = 0;

void setUpIOPins(){
 pinMode(revervePinRight, OUTPUT);
  pinMode(advancePinRight, OUTPUT);
  pinMode(RightMotorPin, OUTPUT);
  pinMode(reversePinLeft, OUTPUT);
  pinMode(advancePinLeft, OUTPUT);
  pinMode(leftMotorPin, OUTPUT);

}

void stopMotor(char left_right){//left right or all
        if(left_right=='r'||left_right=='R'){
      digitalWrite(revervePinRight, false);
      digitalWrite(advancePinRight, false);
      pwmRight = 1;
      analogWrite(RightMotorPin,pwmRight);

  }
  else if(left_right=='l'||left_right=='L'){
      digitalWrite(reversePinLeft, false);
      digitalWrite(advancePinLeft, false);
      pwmLeft = 1;
      analogWrite(leftMotorPin,pwmLeft);
  }
  else if(left_right=='a'||left_right=='A'){
      digitalWrite(reversePinLeft, false);
      digitalWrite(advancePinLeft, false);
      digitalWrite(revervePinRight, false);
      digitalWrite(advancePinRight, false);
      pwmLeft = 1;
      pwmRight = 1;
      analogWrite(leftMotorPin,pwmLeft);
```

```
      analogWrite(RightMotorPin,pwmRight);
  }


}



void setMotorSpeed(char left_right,unsigned char
   speed_,char direction_,char motorNbr,char
   speed_2){//left or rigth
  bool advance_=direction_=='f'?true:false;
     //f==forward

  if(left_right=='r'||left_right=='R'){
      digitalWrite(revervePinRight, !advance_);
      digitalWrite(advancePinRight, advance_);
      pwmRight = speed_;
      analogWrite(RightMotorPin,pwmRight);
      if(motorNbr){
         digitalWrite(reversePinLeft, !advance_);
         digitalWrite(advancePinLeft, advance_);
         analogWrite(leftMotorPin,speed_2);
      }

  }
  else if(left_right=='l'||left_right=='L'){
      digitalWrite(reversePinLeft, !advance_);
      digitalWrite(advancePinLeft, advance_);
      pwmLeft = speed_;
      analogWrite(leftMotorPin,pwmLeft);

       if(motorNbr){
         digitalWrite(revervePinRight, !advance_);
          digitalWrite(advancePinRight, advance_);
          analogWrite(RightMotorPin,speed_2);
      }

  }
  else if(left_right=='a'||left_right=='A'){
      digitalWrite(revervePinRight, !advance_);
      digitalWrite(advancePinRight, advance_);
      pwmRight = speed_;
      digitalWrite(reversePinLeft, !advance_);
```

```
      digitalWrite(advancePinLeft, advance_);
      pwmLeft = speed_;
      analogWrite(RightMotorPin,pwmRight);
      analogWrite(leftMotorPin,pwmLeft);
  }



}
#endif //
```

And finally , the Wifi module : it has been named serial because we first debugged the code using serial communication before switching to the Wifi mode.

```
#ifndef SERIECOMM_H
#define SERIECOMM_H

// Define constants and variables

#include "ESP8266.h"
#define SSID        "Athens2016"
#define PASSWORD    "Arduino2016"
#define HOST_NAME   "192.168.0.101"
#define HOST_PORT   (5204)
SoftwareSerial Serial1(11,12); // RX, TX
ESP8266 wifi(Serial1);
//Serial myPort;

String MsgReceived, msg;

char targetSpeed;

char commandeType;
char left_right;
char direction_;
 char speed_;
char speed2;
uint8_t buffer[128] = {0};
void setupSerial(){
    pinMode(LED_BUILTIN, OUTPUT);
      digitalWrite(LED_BUILTIN, LOW);
//  printArray(Serial.list());
  Serial.begin(9600);
  Serial.print("setup begin\r\n");
```

```
    Serial.print("FW Version:");
    Serial.println(wifi.getVersion().c_str());

    if (wifi.setOprToStationSoftAP()) {
        Serial.print("to station + softap ok\r\n");
    } else {
        Serial.print("to station + softap err\r\n");
    }

    if (wifi.joinAP(SSID, PASSWORD)) {
        Serial.print("Join AP success\r\n");
        Serial.print("IP:");
        Serial.println( wifi.getLocalIP().c_str());
    } else {
        Serial.print("Join AP failure\r\n");
    }

    if (wifi.disableMUX()) {
        Serial.print("single ok\r\n");
    } else {
        Serial.print("single err\r\n");
    }
        if (wifi.createTCP(HOST_NAME, HOST_PORT)) {
        Serial.print("create tcp ok\r\n");
    } else {
        Serial.print("create tcp err\r\n");
    }

    Serial.print("setup end\r\n");
      digitalWrite(LED_BUILTIN, HIGH);

}


char number;

void ReadMessage(){
        //MsgReceived = Serial.readString();

  uint32_t len = wifi.recv(buffer, sizeof(buffer),
     10000);
    if (len > 0) {
        for(uint32_t i = 0; i < len; i++)
```

```
            Serial.print((char)buffer[i]);
  commandeType =(char) buffer[0];
  Serial.println("commandeType = "+commandeType);
  left_right= (char)buffer[1];
  Serial.println("left_right = "+left_right);
 // left_right=Serial.read();//motor to control
  if(commandeType=='m')
     {
          // direction_= Serial.read();
           //speed_=Serial.read();
      direction_= (char)buffer[2];
        Serial.println("direction = "+direction_);
      speed_=(char)buffer[3];
        Serial.println("speed = "+speed_);
      number=(char)buffer[4];
      speed2=(char)buffer[5];
     setMotorSpeed( left_right, speed_,
        direction_,number,speed2);
    }
   else if(commandeType=='s'){

    stopMotor(left_right);
   }
}


}

#endif //
```

We also tried to put an ultrasound detector on the robot in order to scan the area and send back the data to the user via a user interface, but there were conflicts with the Wifi module so we couldn't integrate it to the robot. The ultrasound detector is mounted on a servo motor so that it can scan the area.

and this is how the data are displayed to the user when the detector finds obstacles :

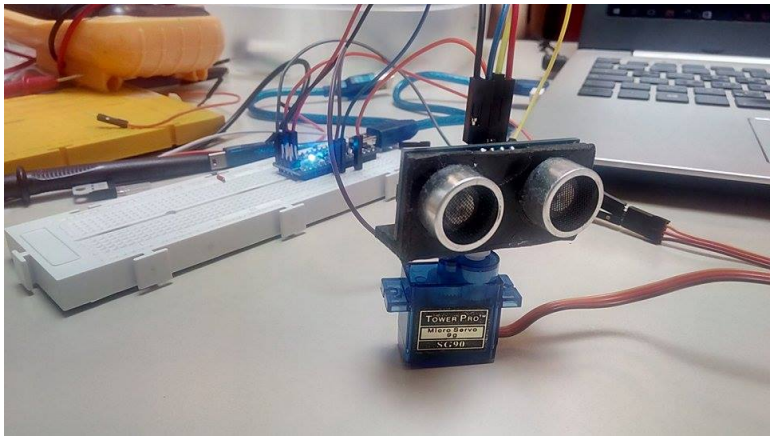


FIGURE 5 – Ultrason detector mounted on a servo

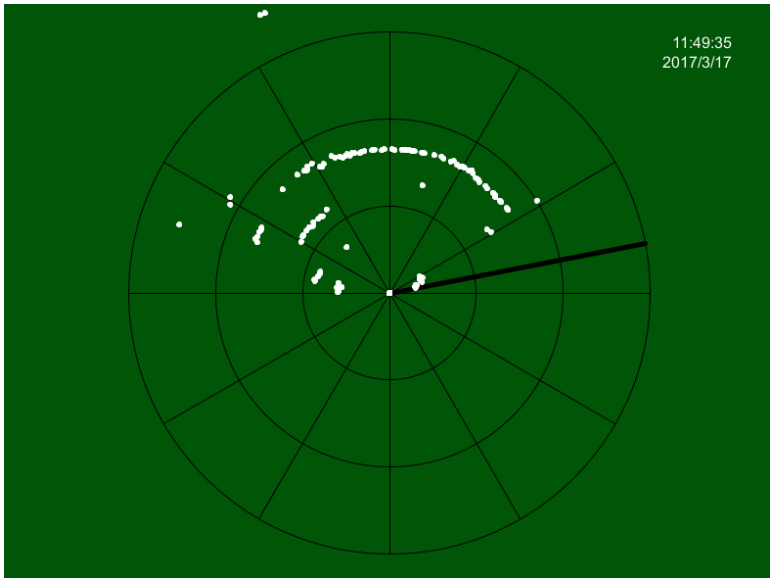


FIGURE 6 – Scan visualisation