Line follower mobile robots, prototypes of robots and functional of mobile robots

D M Anton[1], S E Szabo[1], C A Mociar[1] and A F Iuhas[1]

[1] Faculty of Managerial and Technological Engineering, University of Oradea, 410087 Oradea, Romania

E-mail: danielanton0227@gmail.com

Abstract. This study presents the working principle of line follower robots. This study starts from the description of the components used in robots of this type to the explanation of the functions used to program robots in the C ++ programming language and the use of multiple cases and exceptions in the programming of robots of this type, because they can be used in different ways and domains and for these, the multiple working characteristics must be determined from the beginning. The study establishes the way these robots work, for a future project to create a family of disinfection robots with UV-C neons.

1. Introduction

Robots have begun to be introduced more and more frequently in most companies in the industry. From storage robots in warehouses, used to load, unload and move goods from the place of arrival, to storage, and then to the work area, to climbing robots on shelves, which can move the goods from heights of 10-20 meters. Robots in various forms become more and more useful in everyday life, such as kitchen robots, vacuum cleaners, lawn mowers, solar panel cleaners, each system becoming more accurate and efficient, using assemblies of systems with sensors, motors, artificial intelligence systems, that are interconnected with each other, such as centralized window closing systems, doors, intelligent control for thermal boilers, rolling panels for windows, curtains, air monitoring systems, temperature, air conditioning, radiant panels, photovoltaic panels, plant watering systems, sprinklers.

2. Components of a assembly set of mobile robots with sensors

These types of robots in the simplest models have sensor systems, control systems, motor control systems, power supply systems and can vary depending on the field of use and the accuracy that is needed for some works.

These mobile line follower robots use sensory systems that can be: distance, proximity, human presence sensors, barcode reading sensors. The type of robots used in this study is based on analog color sensors. They are based on QTR-1A (model sensor) type sensors, mounted on PCBs (printed circuit board) with 8 QTR type A sensors[1]. Types A are listed as some of the most accurate because they read values across a wide spectrum. The signals exported by them are from 0 to 1023. These values are read as signals by their comparators, and the microcontroller receives them in the form of varying voltage. These sensor models are small, have pairs of IR LEDs (ligh-emitting diode) or are also called

phototransistors. Each sensor is independent, with its own transmitter and receiver, which can generate a signal for each channel, thus substantially increasing the accuracy.

For line follower robots in most cases they are used as line sensors, but they can also have the function of proximity sensors. These sensors each use a traction resistor to form voltage dividers. They produce an analog voltage at the output, which can be measured between a minimum of 0.2 V and a maximum of 4.8 V. To determine the values, the track of line follower robots are made of white melamine chipboard, and on them the path to be followed by the robot is drawn with a black insulating tape. These conditions may vary, but the operating algorithm will also have to be adapted to the new conditions.

IR sensors are very sensitive to direct sunlight, because it disrupts the sensors and generates incorrect readings, so the robot risks losing the default track (line). When the sensors have a black surface in front of them, the transmitted voltage values are high - they tend to 5V due to the fact that these surfaces reflect a maximum of 10% of the emitted light, this decreases the internal resistance of the receiver. And when the surface is white these values decrease proportionally - they tend to 0V. In figure 1 is presented the schematic wiring diagram of sensors IR (infrared sensor).

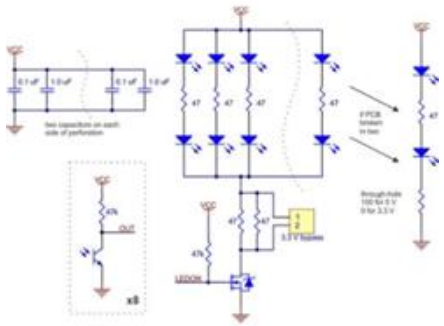


Figure 1. Schematic wiring diagram of sensors IR.

To read the signals from the sensors, the speed is adjusted in 3 steps: 50%, 75%, 100%. For straight line both engines have maximum speed when turning left, the right engine has 100% and the left engine has 50% speed. For turning right it is vice versa. Of course, depending on the radius of the curve, the speed can be adjusted algorithmically.

The robot's microcontroller uses Fuzzy [2] logic to interpret the position on the line. The source code is programmed for sensors 4 and 5 to be fixed on the black line. Those sensors must constantly see the line. If a left curve follows, the line moves to sensors 1 and 2, then the right motor will maintain the speed at 100%, and the left motor will reduce the speed to 50% to regain its position. In figure 2 is presented the reading signals sensor on channel 1 values between 0.2 - 4.8V, real-time reading channel 2 speed.

The robot also has two Pololu micro motors[3] with a 50: 1 reduction, the value of the output speed after the reducer being 600 rpm, for the supply voltage from the 12V driver. This driver is controlled by programming the PWM (pulse-witdh modulation)[4] pins (3, 5 for the left motor and the right motor 6, 9 of the robot) on the Arduino Nano microcontroller[5].

The motors are supplied by time interval pulses (PWM). This allows you to digitally control the speed that can be adapted to the opening of the curve and the accuracy required for the operation to be performed. Figures 3, 4, 5 show the PWM speeds for different duty cycles, at 50%, 75%, 100%. These intervals represent the supply voltage on the vertical axis, and the time in milliseconds for the motor supply on the horizontal axis.

3. Robot construction

In the construction of the robot, the response time of the microcontroller was taken into consideration, in order to determine the accuracy. Its size is 170mm long, the motors are arranged equidistant from the middle of the robot at a distance of 60mm.

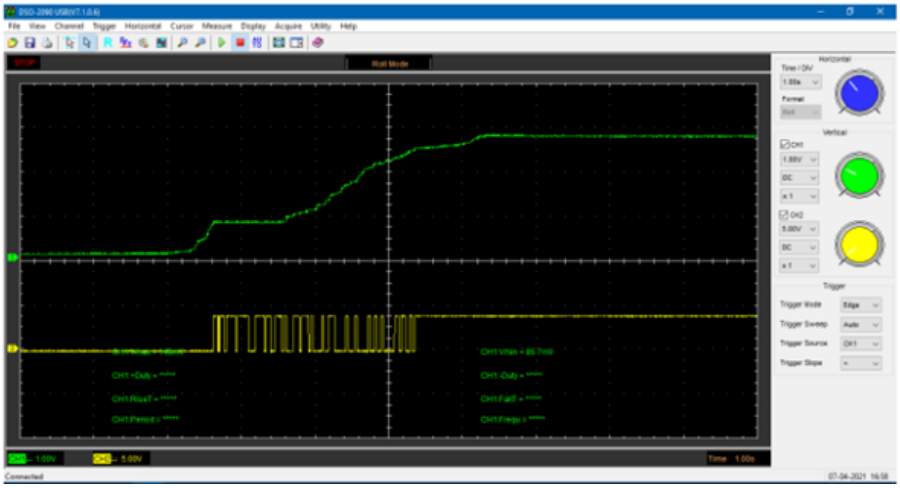


Figure 2. Reading signals sensor.

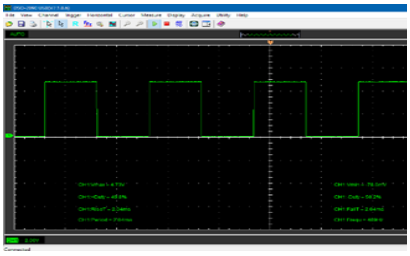


Figure 3. Signal PWM,duty cycle 50%.

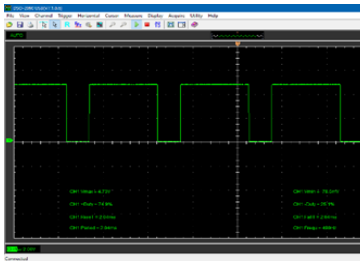


Figure 4. Signal PWM, duty cycle 75%.

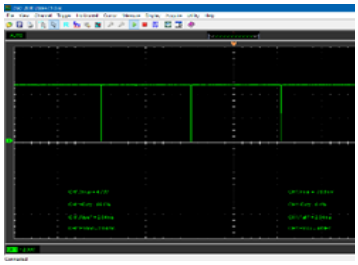


Figure 5. Signal PWM, duty cycle 100%.

The sensors were positioned as far forward as possible and at a distance of 4 mm from the chipboard, in order to receive values as accurate as possible, identical to those of previous measurements.

A set of wheels is mounted on the robot's motors, designed according to the weight and complexity of the route and then made on a 3D printer. These wheels are printed with settings of 0.2 mm per layer and 18% filling, to be as light as possible, with PLA (Polylactic Acid) filament. The printing is done at 210^0C at the extruder, and the bed at 70^0C, the printing speed is 80 mm/s.

The wheels are initially provided with an outer wall, so that after printing a two-component layer of silicone can be poured into these wheels with the syringe. Silicone, when pouring into the wheel must be added to only one part of the wheel in order to eliminate any air bubbles and at the end, the wheel must be either vibrated or lightly beaten by the table so that the air remaining through the silicone rises on top. After the silicone has hardened, the outer wall for the silicone is heated and gently peeled off by the wheel, after which it is left to dry for another two hours. The shorter the outer wall from the wheel, the thinner the silicone layer which is useful if the robot weighs more.

The ADC (Analog to Digital Converter) [6] function of the ATMEGA328 microcontroller converts the analog signal from the sensors into a PWM digital signal so that it can be processed by the microcontroller and sent on to the motor drivers. The working frequency of this ADC converter is 400Hz, being limited by the internal functions of the program.

Figure 7 show the first types of robots built.

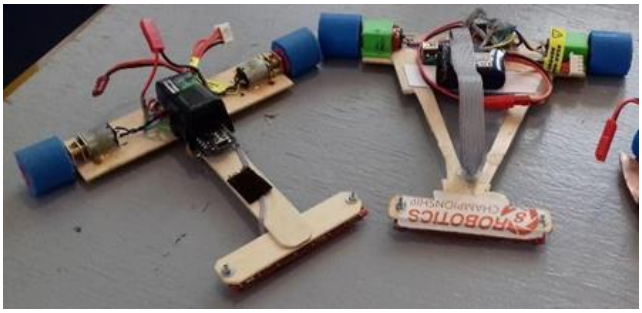


Figure 6. The first versions of robot.

Figure 7. Award second place Robotics Championship Oradea.

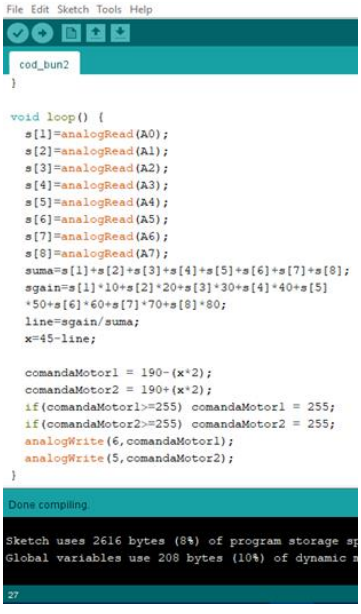
4. Code source robot

The first part of the program, presented in figure 8, initializes the global variables used in the code, followed by the setup section (void setup). In this section, the input or output modes of the pins on the Arduino Nano board are assigned and pins 3 and 9 are assigned to ground (GND). The next part, shown in figure 9, presents the loop function (void loop), in which the code is repeated endlessly until the power supply is interrupted. In the first part of this function, a vector is built, taking the values read from the sensors, and then with the help of this vector the position of the line is calculated using the Fuzzy algorithm, obtaining a value between 10 and 80 units.

```
File Edit Sketch Tools Help
cod_bun2
unsigned int s[10],suma,line;
long int sgain;
int x,afara;
int comandaMotor1, comandaMotor2;
void setup() {
  pinMode(3,OUTPUT);
  pinMode(A0,INPUT);
  pinMode(A1,INPUT);
  pinMode(A2,INPUT);
  pinMode(A3,INPUT);
  pinMode(A4,INPUT);
  pinMode(A5,INPUT);
  pinMode(A6,INPUT);
  pinMode(A7,INPUT);
  Serial.begin(9600);
  digitalWrite(3,0);
  digitalWrite(9,0);
}

void loop() {
  s[1]=analogRead(A0);
  s[2]=analogRead(A1);
  s[3]=analogRead(A2);
  s[4]=analogRead(A3);
Done compiling.
Sketch uses 2616 bytes (8%) of program storage space. Max
Global variables use 208 bytes (10%) of dynamic memory, l
27
```

Figure 8. Robot source code first part.



```
File Edit Sketch Tools Help
cod_bun2
}

void loop() {
  s[1]=analogRead(A0);
  s[2]=analogRead(A1);
  s[3]=analogRead(A2);
  s[4]=analogRead(A3);
  s[5]=analogRead(A4);
  s[6]=analogRead(A5);
  s[7]=analogRead(A6);
  s[8]=analogRead(A7);
  suma=s[1]+s[2]+s[3]+s[4]+s[5]+s[6]+s[7]+s[8];
  sgain=s[1]*10+s[2]*20+s[3]*30+s[4]*40+s[5]
  *50+s[6]*60+s[7]*70+s[8]*80;
  line=sgain/suma;
  x=45-line;

  comandaMotor1 = 190-(x*2);
  comandaMotor2 = 190+(x*2);
  if(comandaMotor1>=255) comandaMotor1 = 255;
  if(comandaMotor2>=255) comandaMotor2 = 255;
  analogWrite(6,comandaMotor1);
  analogWrite(5,comandaMotor2);
}
Done compiling.
Sketch uses 2616 bytes (8%) of program storage sp
Global variables use 208 bytes (10%) of dynamic
27
```

Figure 9. Robot source code another part.

Further, in the variable "x" it decreases from the middle value of 45 units, the position of the line calculated above and we obtain a value that varies between -35 and 35. The middle position has the value 0. This variable may or may not influence engine speed. In the variables "comdandaMotor1" and "comandaMotor2" the speed for each motor is calculated, and the PWM signal transmitted to the motor drivers is generated.

5. Results and conclusion

In this form, the robot used the engines at full capacity, the problems that occurred after running the route were overheating of the engines, because for the line follower test the robot must be built to have a low weight and be fast to travel the route in the shortest time. Because a very low weight was desired,

the 3D model was designed to add as little silicone as possible to the wheels, as this greatly increases the weight of the robot. The designed robot uses 8 IR sensors, so reading the line is easy and at the maximum speed of this robot.

These robots participated in the international competitions Robotics Championship Oradea and RoboManiacsTimișoara, and in these competitions the following prizes were obtained:
-Robotics Championship Oradea: Second place in the Line Follower Junior category, as presented in figure (7)
-Robo Maniacs Timișoara: 1st and 2nd place in the Line Follower Baby category

To participate in competitions, it is also necessary to test and simulate the robot in different working functions because some routes are more complicated, for example in the case of 90 degree curves, and the robot loses the line completely, without recovering.

A future study will present the integration of the reading functions of the last sensors that observed the line and will try to implement the PID (Proportional Integrated Derivatives) function, use the bring back function or search for it according to the last reading. It will move to hardware upgrades for robot components, better engine controllers, faster sensor reading and software part optimization with more predefined functions and PID functions.

References

[1] https://www.pololu.com/product/960
[2] https://en.wikipedia.org/wiki/Fuzzy_logic
[3] https://www.pololu.com/product/1093
[4] https://en.wikipedia.org/wiki/Pulse-width_modulation
[5] https://www.arduino.cc/en/pmwiki.php?n=Main/ArduinoBoardNano
[6] https://en.wikipedia.org/wiki/Analog-to-digital_converter