Path planning of line follower robot

2 authors:

Mustafa Engin
Ege University
37 PUBLICATIONS   **218** CITATIONS

Dilsad Engin
Ege University
24 PUBLICATIONS   **105** CITATIONS

Some of the authors of this publication are also working on these related projects:

Project   Ultrasonik Transmiter ile SCADA Uyumlu PLC Kullanılarak Seviye Kontrolünü Sağlayan Bir Sistemin Tasarımı ve Uygulanması View project

PATH PLANNING OF LINE FOLLOWER ROBOT

B8

Mustafa Engin[1], Dilşad Engin[2]

[1]Ege Technical and Business College, Department Electronics Technology of Technology, Ege University
Bornova, 35100, Izmir, Turkey
phone: + (90) 533 435 08 59, fax: + (90) 232 388 75 99, email: mustafa.engin@ege.edu.tr
[2]Ege Technical and Business College, Department of Control and Automation, Ege University
Bornova, 35100, Izmir, Turkey
phone: + (90) 555 260 40 79, fax: + (90) 232 388 75 99, email: dilsad.engin@ege.edu.tr
web: www.ege.edu.tr

ABSTRACT

This paper presents the development of a line follower wheeled mobile robot. In this project, LM3S811 which is ARM cortex-3 based microcontroller is chosen as the main controller to react towards the data received from infra-red line sensors to give fast, smooth, accurate and safe movement in partially structured environment. A dynamic PID control algorithm has been proposed to improve the navigation reliability of the wheeled mobile robot which uses differential drive locomotion system. The experimental results show that the dynamic PID algorithm can be performed under the system real-time requirements.

Keywords - *embedded system, wheeled mobile robot, PID control algorithm.*

1 INTRODUCTION

Embedded system includes many areas of knowledge, microcontroller hardware and software, interfacing technologies, automatic control theory, and sensor technologies etc. To speed up the learning process and motivate students to learn actively, the project-based learning approach may be applied in the embedded system design laboratory [1-4]. The low-cost wheeled mobile robot construction which is proposed in this paper, serves as a good example on which students can learn embedded system design skills. It covers not only common embedded system peripherals, but also energy control and real-time control firmware implementation. The process of the construction of wheeled mobile robot can give students the idea that hardware circuits and software algorithms are both important for a successful embedded system design. The competition between student groups in the racing contest can also motivate them to explore in depth the skills acquired in this laboratory as well as give them lots of fun [5-7].

The remainder of this paper is organized as follows: The line follower robot structure and architecture issues and challenges along with their technical issues and problems are discussed in section 2. Programming details will be explained in section 3. Section 4 describes the integration of the complete system.

2 LINE FOLLOWER WHEELED MOBILE ROBOT STRUCTURE

Generally, the line follower robot is one of the self-operating mobile machines that follow a line drawn on the floor. The path can be a visible black line on a white surface or vice versa. The basic operations of the line follower are as follows:

- Capturing the line position with optical sensors mounted at the front end of the robot. Most are using several numbers of photo-reflectors. Therefore, the line sensing process requires high resolution and high robustness.

- Steering the robot to track the line with any steering mechanism. This is just a servo operation; actually, any phase compensation will be required to stabilize tracking motion by applying digital PID filter or any other servo algorithm.

- Controlling the speed according to the lane condition. The speed is limited during passing a curve due to the friction of the tire and the floor.

From physically building the robot platform, to setting up, programming, and hardware or software fine tuning, everything needs to be taken into account when building a differential wheeled mobile robot. A mobile robot can be regarded essentially as an ensemble of five main parts and subsystems.

- Chassis and body.

- Sensors and signal processing circuits.

- Microcontroller.

- Motor drivers

- Actuators (Motors and wheels)

2.1 The Chassis and Body

The Chassis would be the main part of a robot's body. It is designed to carry all of the other components, transmission mechanisms, electronics and battery. It needs to be sufficiently large and provide adequate fixtures to accommodate all necessary parts, as well as sturdy enough to cope with the weight of the parts along with additional

loads which can appear in dynamic conditions such as vibrations, shocks or chassis torsion and actuators torque.

There are some good materials for designing robots such as plastic, aluminum and carbon-composites. We must pay attention to the resistance, weight and mechanical ability for choosing one of them. In the designed robot, printed circuit board (PCB) has been used for chassis because of its lightweight and being strong enough for our project. All components can be installed on the PCB to decrease the weight. It is noted that the performance is much more important than other issues.

2.2   Sensors and signal processing circuit

Line follower robot uses Infrared Ray (IR) sensors to find the path and direction. IR sensors contain an infrared transmitter and infrared receiver pair. IR sensors are often used to detect white and black surfaces. White surfaces generally reflect well, but black surfaces reflect poorly. Hence, the distance between sensors and ground surface is important and it is more important that how we put sensors near to each other. The distance between sensors and ground surface must be 2 to 10 mm and the distance between each sensor is dependent on the line width. In the designed robot, we have used eight sensors and they have a suitable distance between each other. If the line width is narrow, the distance between sensors must be reduced; otherwise, while curving the line, the robot will not be turned on time.

Generally, the received signals from the sensors are analog and must be converted to the digital form. Therefore, the designed signal processing circuit can send the sensors' signals to the microcontroller directly.

2.3   Microcontroller

We have used the TI Stellaris microcontroller LM3S811 in our project. The LM3S811 microcontroller has a Reduced Instruction Set Coding (RISC) core. Internal oscillators, timers, UART, USB, SPI, pull-up resistors, pulse width modulation, ADC, analog comparator and watch-dog timers are some of the features [8]. With on-chip in-system programmable Flash and SRAM, the LM3S811 is a perfect choice in order to optimize cost.

2.4   Motor Drivers

A well-known and suitable motor driver is IC L298 which can be used to control two motors. It is a high voltage, high-current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as DC and stepping motors [9]. Two enable inputs are provided to enable or disable the device independently of the input signals. L298 has 2 amperes per channel current capacity and it can support up to 45 volts for outputting. Moreover, L298 works well up to 16 volts without any heat sink.

2.5   The Actuators (Motors and wheels)

There are many kinds of motors and wheels. Our choice depends on the robot function, power, speed, and precision.

Actually, it is better to use gearbox motors instead of common DC motors because it has gears and an axle and its speed does not change towards the top of a hill or downhill. Motors are rated to operate at 1700 rpm at 7 volt nominal voltage.

It is better to use wheels for line follower robots, instead of a tank system. We can use three wheels. Two of them are joined to the motors and installed at the rear of the robot and the other wheel is free and installed in front of the robot as a passive caster.

To get better maneuver, our robot uses two motors and two wheels on the rear and a free wheel on the front. The power supply is 7.6 V with a regulator. The designed robot has eight infrared sensors on the front bottom for detecting the line. Arm based microcontroller Stellaris and driver L298 were used to control direction and speed of motors. General view of the line follower robot that we built is shown in Fig. 1. The robot is controlled by the microcontroller. It performs the change in the motor direction by sending an appropriate signal to the driver IC according to the received signals from the sensors.

3 REAL TIME TASK SCHEDULING

We built a light-weighted and high-speed robot because points are awarded based upon the distance covered and the speed of the overall robot. Therefore, we used two high speed motors and a highly sensitive signal conditioning circuit. The body weight and wheels' radius have effects on the speed, too. The weight of the designed robot is around 300 gr. and it could be lighter. The photograph of the top and bottom views of the designed robot is shown in Fig. 1.

The microcontroller sends instructions to the driver after processing the data received from sensors. The driver powers the motors according to the inputs. Actually the driver supplies positive voltage to one of the motor pins and negative voltage to the other. There are five states of movement:

- To move forward; both of the motors are turned on and rotate forward simultaneously.

- To move left; the right motor is turned on and the left motor is turned off.

- To move right; the left motor is turned on and the right motor is turned off.

- To move left fast; the right motor rotates forward and the left motor rotates backward.

- To move right fast; the left motor rotates forward and the right motor rotates backward.

Most embedded system applications need to react to the inputs or environment changes in real time, which means that the accuracy of computations is as important as their timelines. Furthermore, digital control algorithms need a fixed sampling time interval for measuring inputs and delivering output commands. Therefore, the idea of

applying interrupts for task scheduling is introduced in this work.
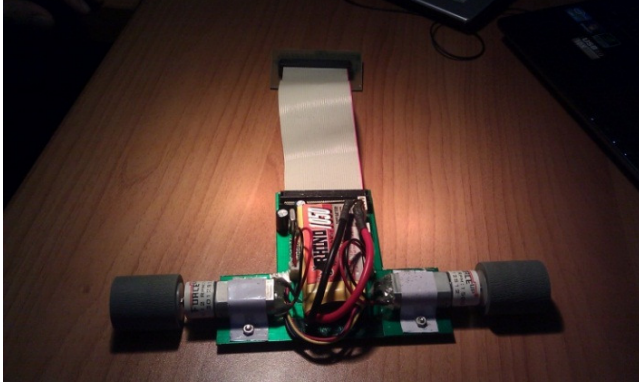
(a)
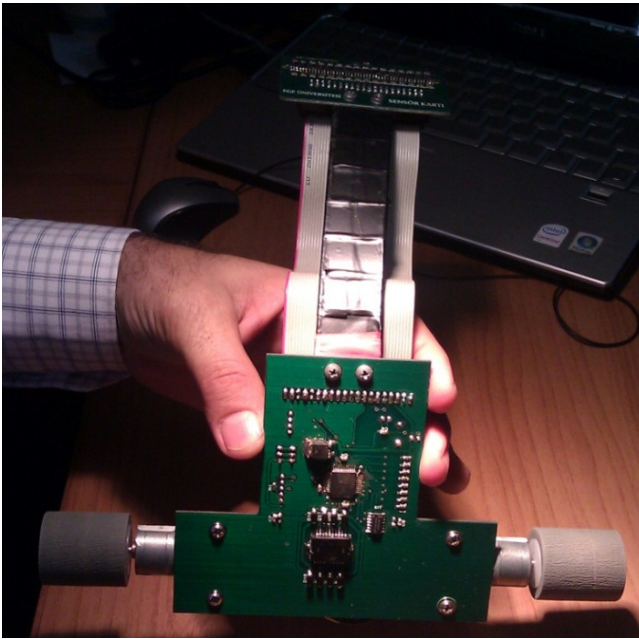
(b)

Figure 1 – Images show (a) top, (b) bottom views of the built line follower robot.

3.1   The Quadratic Line-Detection Algorithm

A better way of detecting the line position, compared to the other simple line-following robots, by using a quadratic interpolation technique is introduced. Eight reflective optical sensors were used, and the coordinate of the leftmost sensor was 0. To find out the correct position of the black line, we had to locate three consecutive sensors with higher output readings than the other five sensors as shown in Fig. 2. Assume that the coordinates of these 3 sensors are x_1, x_1+1, and x_1+2, and the true shape of the sensor output values are in the range of $[x_1, x_1+2]$ which can be approximated by a quadratic curve. One can then find the following relationships between the coordinates of the sensors and the output values:

$$y_1 = ax_1^2 + bx_1 + c \qquad (1)$$

$$y_2 = a(x_1 + 1)^2 + b(x_1 + 1) + c \qquad (2)$$

$$y_3 = a(x_1 + 2)^2 + b(x_1 + 2) + c \qquad (3)$$

The coordinate value, at which the output value of the quadratic curve is the maximum, is considered as the true position of the line. By using the basic calculus, one would know that the coordinate value is:

$$x = -\frac{b}{2a} \qquad (4)$$

$$a = \frac{y_1 + y_2 - 2y_3}{2} \qquad (5)$$

$$x = y_2 - y_1 - 2ax_1 - a \qquad (6)$$

It is assumed that the coordinate for the center position of the line-following robot is 0. Therefore, the error e between the line position and the center position of the robot is
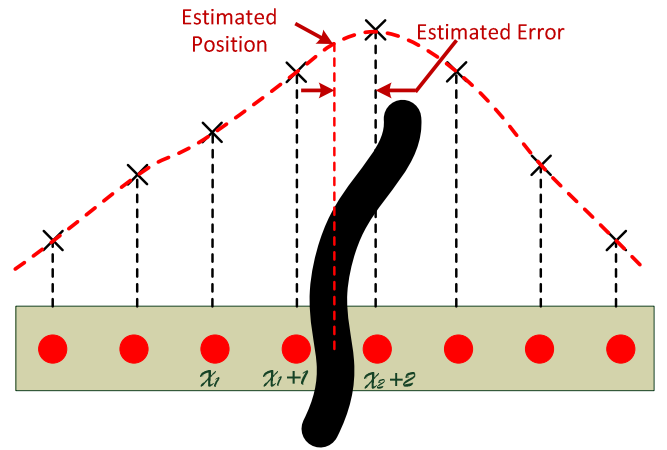
$$e = 0 - x = -x \qquad (7)$$

Figure 2 – The line detection algorithm via quadratic interpolation.

3.2   PID Tracking Control Algorithm

The popular proportional-integral-derivative (PID) controller was introduced in this project to make the robot follow the racing track. The error between the center of the sensors and the track to be followed was then processed by the PID controller to generate velocity commands for the right and left wheels.

First, the controller calculates the current position, and then calculates the error based on the current position. It will then command the motors to take a hard turn, if the error is high or a small turn, if the error is low. Basically, the magnitude of the turn taken will be proportional to the error. This is a result of the proportional control. Even after this, if the error does not decrease approximately to zero, the controller will then increase the magnitude of the turn further and further over time till the robot centers over the line. This is the result of the integral control. In the process of centering over the line, the robot may overshoot the target position and move to the other side of the line where the above process is followed again. Thus the robot may

keep oscillating about the line in order to center over the line. To reduce the oscillating effect over time, the derivative control is used. The proportional term is only a gain amplifier, and the derivative term is applied in order to improve the response to disturbance, and also to compensate for phase lag at the controlled object.

Pseudo Code for the PID Controller;

```
Kp = 10
Ki = 1
Kd = 100
offset = 45    ! Initialize the variables Tp = 50
integral = 0   ! the place where integral value will be stored
lastError = 0  ! place where last error value will be stored
derivative = 0 ! place where derivative value will be stored
Loop forever
    LightValue = read sensors      ! read sensors.
    error = -x      ! calculate the error using equation (7).
    integral = integral + error     ! calculate the integral
    derivative = error - lastError   ! calculate the derivative
    Turn = Kp*error + Ki*integral + Kd*derivative
    powerA = Tp + Turn         ! power level for motor A
    powerB = Tp - Turn         ! power level for motor B
    MOTOR A direction=forward power=PowerA
    MOTOR B direction=forward power=PowerB
    lastError = error      ! save the current error
    end loop forever       ! do it again.
```

PID controller requires the K_p, K_i and K_d factors to be set to match wheeled line follower robot's characteristics and these values depends on robot structures, actuators, sensors and other electronic circuits. There is no easy way to calculate K_p, K_i and K_d factors. It requires manual trial and error method until you get the desired behavior. We defined these factors following these guidelines;

- Start with low speed and setting values of K_p, K_i and K_d to 0.

- Then, try setting K_p to a value of 1 and observe the robot. The goal is to get the robot to follow the line even if it is very wobbly. If the robot overshoots and loses the line, reduce the value of K_p. If the robot cannot navigate a turn or seems sluggish, increase the K_p value.

- Once the robot is able to follow the line, set K_d value to 1 and then try increasing this value until you see less wobble.

- Once the robot is fairly stable at following the line, assign a value of .5 to 1.0 to K_i. If the K_i value is too high, the robot will jerk left and right quickly. If it is too low, you won't see any perceivable difference. Since integral is cumulative, the K_i value has a significant impact. You may end up adjusting it by .01 increments.

- Once the robot is following the line with good accuracy, you can increase the speed and see if it is still able to follow the line. Speed affects the PID controller and will require retuning as the speed changes.

4 RESULTS AND DISCUSION

A line following robot is programed with simple (on/off) control as a comparison purpose in evaluating the performance of the dynamic algorithm controlled robot. The results of the experiment are summarized in Table-1. From the data in the table, it can be observed that dynamic PID algorithm controlled robot has better performance in every criteria listed in the table compared to simple (on/off) control robot. The dynamic algorithm controlled robot has higher velocity, consumes less time to complete one whole circuit, tracks the line smoother and has lower tendency to astray from line compared to uncontrolled robot. Therefore this system can be used in training undergraduate students on dynamic PID algorithm control system, its application and implementation in the real world and the advantages that it offers. Fig. 3 shows the designed robot during race pits test.
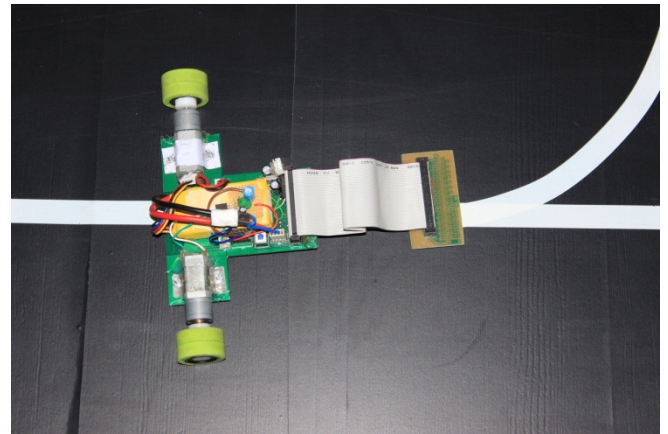

Figure 3 – The designed robot on the race pits.

Table 1– Experimental result for Line Following Robot.

Criteria	Dynamic PID algorithm	Simple (on/off)
Time to complete one whole circuit	47.6s	71.4s
Line tracking	Smooth	Not so smooth
Velocity	0.2m/s	0.14m/s
Tendency to astray from line	Low	High

5 CONCLUSION

The designed wheeled line follower mobile robot has eight infrared sensors on the bottom for detecting the line. The controller board includes Stellaris LM3S811 micro-

controller and the motor driver L298 which were used to control the direction and the speed of motors. The proposed dynamic PID algorithm derives the line follower locomotion by adequately combining the information from sensor module. Experimental results show that the proposed algorithm can successfully achieve target following in various scenarios, including straight line and circular motion, sharp-turn motion and S-shape line tracking. We are working currently to develop a more sophisticated algorithm which can perform faster line tracking with less energy consumption.

REFERENCES

[1] T. Braunl, *Embedded Robotics: Mobile Robot Design and Applications With Embedded Systems*. Springer-Verlag, 2nd edition edition, 2006.

[2] J. Dupuis, and M. Parizeau, "Evolving a Vision-Based Line-Following Robot Controller" *In Proceedings of the The 3rd Canadian Conference on Computer and Robot Vision*, *IEEE Computer Society,* pp. 75-79. Washington, DC, USA 2006.

[3] A. Kettler, M. Szymanski, J. Liedke, H. Worn, "Introducing Wada A new robot for research, Education and Arts", *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* Taipei, 18-22 Oct. 2010.

[4] Lee, C.-S., Su, J.-H., Lin, K.-E., Chang, J.-H., Chiu, M.-H. and Lin, G.-H., A hands-on laboratory for autonomous mobile robot design courses. *Proc. 17th Inter. Federation of Automatic Control World Congress*, Korea, pp.473-9,478 Sept. 2008.

[5] Konaka, E., Suzuki, T., Okuma, S, "Line-following control of two wheeled vehicle by a symbolic control," *Proc. of the 40th IEEE Conference on Decision and Control*, Orlando, USA, Dec. 4 - 7, 2001.

[6] C. Cardeira, J.S. Da Costa, "A low cost mobile robot for engineering education," *Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE* , 6-10 Nov. 2005

[7] V. Papadimitriou, and E. Papadopoulos, "Development of an Educational Mechatronics/ Robotics Platform Using LEGO Components," IEEE *Robotics and Automation Magazine*, Vol. 14, No. 3, pp. 99-110, September, 2007.

[8] http://www.ti.com/lit/ds/spms150g/spms150g.pdf

[9] http://www.st.com/internet/analog/product/63147.jsp