TECHNOLOGICAL ENGINEERING
volume XV, number 1/2018
ISSN 1336 - 5967
DOI: 10.1515/teen-2018-0018

DESIGN OF DIGITAL CONTROL SYSTEM FOR LINE FOLLOWING ROBOT

[1]Mohanad Abdulhamid, [2]Muindi Mutheke

[1]AL-Hikma University,Iraq, [2]University of Nairobi, Kenya

Article history:
Received 27.12.2018
Accepted 30.12.2018
Available online 31.12.2018

Abstract
Design of mobile robots has become an increasingly growing trend in the technology of modern times. They are very attractive engineering systems, not only because of many interesting theoretical aspects concerning kinematics, intelligent behavior and autonomy, but also because of applicability in many human activities. A typical example is the line following robot (LFR). In order for a LFR to function effectively, it must demonstrate excellent line tracking control. This is achieved by having accurate and responsive control algorithms as well as high precision color sensor systems. This paper proposes a system to show that good line tracking performance can be achieved with moderately simple digital control algorithms. The platform used is a differentially driven wheeled robot constructed using the Lego Mindstorms components. The simulation models are presented and analyzed using MATLAB Simulink. The main programming environment is the EV3 Software.

Keywords
Design; digital control; line following robot

1 INTRODUCTION

Robotics is a branch of engineering that involves the conception, design, manufacture and operation of robots. The field overlaps with electronics, computer science, artificial intelligence, mechatronics, nanotechnology, bioengineering and control engineering. Robots are mechatronic engineering products, capable of acting autonomously while implementing assigned behaviors in various physical environments. The developed use of robots in many areas makes the fundamental understanding of them fundamental.

In recent years there has been a rapid increase in the use of digital controllers in control system. It has become routinely practicable to design very complicated digital controllers and to carry out the extensive calculations required for their design. The current adoption of digital rather than analog control in robotics is due to the genuine advantages found in working with digital signals rather than continuous time signals.

The use of analog controllers in control engineering poses problems such as limited accuracy, susceptibility to noise and drift of power supply, cost ineffectiveness and less flexibility. Digital control systems are more suitable for modern control systems because of reduced cost, noise immunity and speed.

Line following robots need to adapt accurately, faster, efficiently and cheaply to changing operating conditions. The drawbacks prominent in analog controllers reduce their suitability in robotics. Hence, the necessity for digital controllers which provide better performance capabilities. Some works on this topic can be found in literatures [1-5].

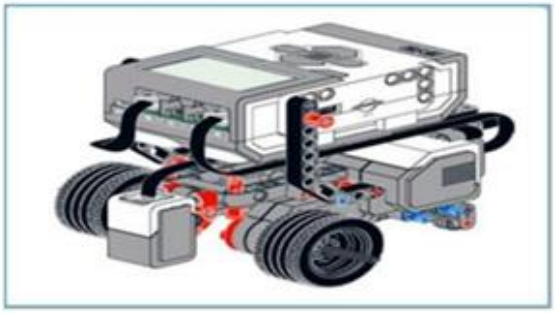

Figure 1. Line following robot

2 DESIGN METHODOLOGY

2.1 Lego mindstorms line follower robot design

A line follower shown in Fig. 1 is a mobile robot which is able to follow a visible line on a surface consisting of contrasting colours. To build and run the robot, the required hardware included; Lego EV3 brick, power supply, 2 large servo motors, a set of wheels, colour sensor, connector cables, beams, axles, bushes and pins. The EV3 brick formed part of the chassis, equipped with wheels. The servo motors are used to drive the two front wheels. Two rear small castor wheels supported the robot. The robot had a colour sensor mounted at the front end to identify the line. It is centered between the two front wheels, which are separated by a distance of 7.4 cm. It is designed to follow an oval track made of black electrical tape (18 mm wide) on a white surface.

2.2 Study of Lego Mindstorms EV3 motor

Lego Mindstorms has not published the EV3 motor's electromechanical characteristics. Table 1 shows the proposed parameters used in this paper, while, Table 2 shows the operational specifications.

Table 1.
Lego Mindstorms EV3 large motor characteristics

Motor Parameter	Unit	Value
Torque constant,	N.m/A	0.2
Back e.m.f constant,	V.s/rad	0.5
Armature resistance,	Ω	5
Armature inductance,	H	0.005
Viscous damping coefficient, B	N.m/rad.s	0.0006
Rotor inertia coefficient, J	N.m.	0.001

Table 2.

Operational specifications

Nominal Voltage	7.2V or 9V
Rotation Speed at no load	160 – 170 rpm
Running Torque	0.20 N-m
Stall Torque	0.40 N-m

Fig.2 shows the motor model simulated using MATLAB Simulink.

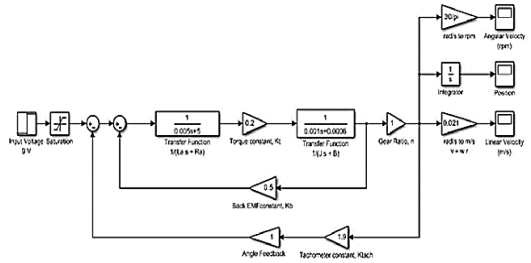

Figure 2. Motor implementation on Simulink

2.3 Line following robot algorithms

Line following works by using the colour sensor (in reflected light intensity mode) to read the changes in the reflected light levels along the edge of a dark and light surface. The reflected light intensity is measured as a percentage from 0% (very low reflectivity) to 100% (very high reflectivity). More light is reflected from a white surface compared to the black surface. Depending on the light sensor value, the motors are directed to vary the speed.

In a program, white and black values are defined using a threshold value. Threshold is the average of the sensor value with the sensor on the black line and one found on the white area. Different measurements for black and white depend on factors such as the light level in the room, the robot's battery level, and the type of surface used.

The light sensor will read the light value. Then the robot can be programmed such that if the sensor sees black, which is when the sensor value is less than the threshold, the robot should turn right, else it should turn left. The basic line following approach is shown in Fig.3, and can be summarized as follows:

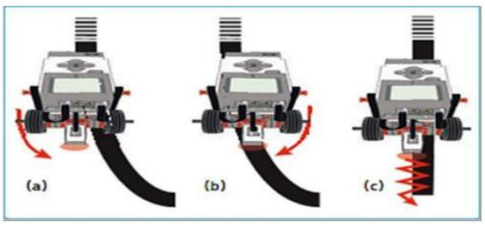

Figure 3. Basic line following approach

1. The robot will be started. It will then be set to move forward. It will be made to steer right until it detects the line edge.
2. Once the sensor sees black, the robot will continue to go forward while turning left gradually.
3. Whenever the sensor will see white (i.e. the robot leaving the line), the robot will turn to the right until the sensor finds black again.
4. The sequence then will be repeated in a loop, unless the robot is stopped.

2.4 Digital controller design

A robot without a controller will oscillate a lot about the line, leading to more consumption of battery power, less speed and following the line less efficiently.

When designing a line following robot, the transient response specifications are defined as:

1. Rise time: It is how fast the robot will try to get back to the line after it has drifted off.
2. Overshoot: The distance past the line edge the robot will tend to go as it is responding to an error. 3-The amount of overshoot indicates the relative stability of the system.
3. Steady-state error: The offset from the line as the robot follows a long straight line.
4. Settling time: The time the LFR will take to settle down when it encounters a turn.

The performance criteria are stipulated as follows:

1. Constant speed of 0.1m/s to be maintained despite the presence of turns.
2. Steady-state error: Less than 2%
3. Settling time of less than 0.1 seconds
4. Overshoot (%) of less than 1.0
5. Finite phase margin

The robot controller to be designed is to be modified until the transient response met is satisfactory.

2.4.1 Proposed controller design

The proposed controller is a Proportional-plus-Integral -plus-Derivative (PID) digital controller. The PID controller would control the position of the robot with quick respon-se time and minimize the overshoot. The proportional part would determine the magnitude of turn required to correct the error sensed. The integral part would improve the steady state error (proportional offset) which increa–sees while the robot is not on the line. The derivative part would measure the deviation from the path and mini-mize overshoot. It would reduce the oscillating effect about the line. The derivative control is used to provide anticipative action.

2.5 Implementation of line following control algorithm for lego mindstorms EV3 hardware

Fig.4 shows the Simulink line tracking program with PID controller, while Fig.5 shows EV3 software line following program with PID controller. Sensors and motors contain blocks that interface with the EV3 hardware. Actual speed values block uses the values from each motor encoder to calculate the position and velocity of the robot. Desired velocity takes the user-provided velocity (m/s) and converts it into the desired state values for the velocity controller. Desired light takes the color sensor's white and black values to choose an appropriate reference value for the light. Velocity control has the PID controller implementation to control the forward velocity. Line tracking controller has the PID controller implementation to control the turning.

However, to download and run a line tracking Simulink model on the Lego Minfstorms EV3 robot, EV3 Wi-Fi Dongle or USB Ethernet Adaptor, and Wi-Fi Router are required to set up a network connection between EV3 brick and host computer.

The line following program is then written in EV3 software programming language. The black and white light intensity values are calibrated accordingly for the

robot and the track. Using the provided USB cable, the program is downloaded and run on the robot. PID parameters(K_p, K_i, and K_d) tuning is done experimentally to achieve smoother line tracking.

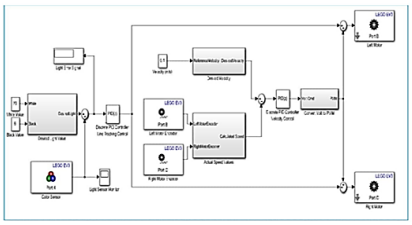

Figure 4. Simulink line tracking program with PID controller

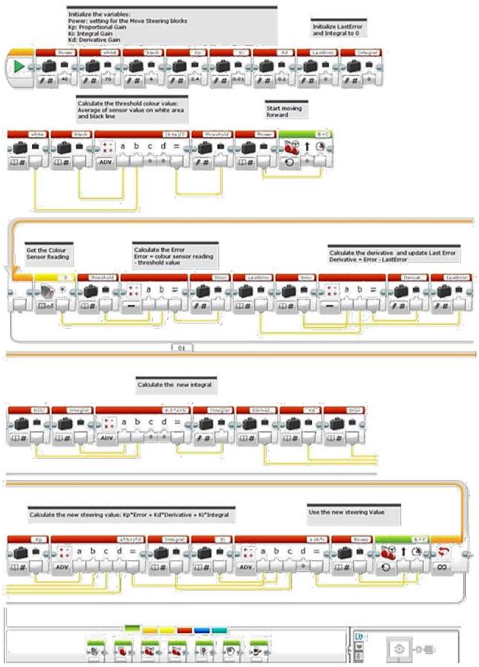

Figure 5. EV3 Software line following program with PID controller

3 SIMULATION RESULTS

3.1 EV3 large motor characteristics

Table 3 shows the EV3 motor load characteristics, from which the linear relationship between power level and EV3 large motor speed noticeable as shown in Fig.6. Also, from table 3, the rotation speed of the EV3 large motor is proportional to the input voltage.

Table 3.

EV3 motor load characteristics

Input Voltage	Torque	Rotation speed	Current	Mechanical power	Electrical power	Efficiency
4.5 V	17.3 N.cm	24 rpm	0.69 A	0.43 W	3.10 W	14 %
6.0 V	17.3 N.cm	51 rpm	0.69 A	0.92 W	4.14 W	22 %
7.5 V	17.3 N.cm	78rpm	0.69 A	1.41 W	5.17 W	27 %
9.0 V	17.3 N.cm	105 rpm	0.69 A	1.90 W	6.21 W	31 %
10.5 V	17.3 N.cm	132 rpm	0.69 A	2.39 W	7.24 W	33 %
12.0 V	17.3 N.cm	153 rpm	0.69 A	2.77 W	8.28 W	33 %

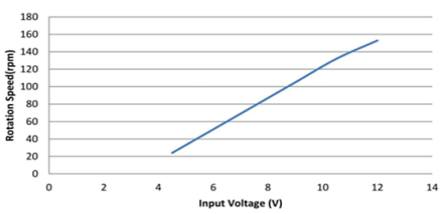

Figure 6. Graph of rotation speed against applied voltage

3.2 PID parameters tuning

Different values of PID parameters(Kp, K_i, K_d) are chosen in order to get the step response.

1- For K_p=1, K_i=0, K_d=0, the result is shown in Fig.7

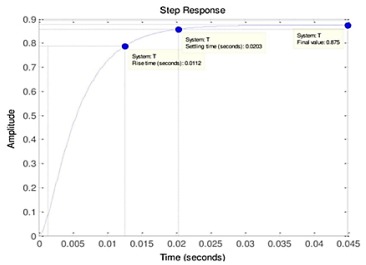

Figure 7. Step response: K_p = 1

Observations:
Rise time = 0.0112 seconds
Settling time = 0.0203 seconds
Final value = 0.875

2- For K_p=5, K_i=0, K_d=0, the result is shown in Fig.8

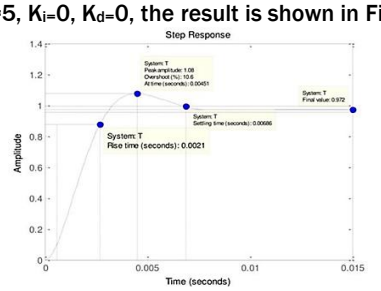

Figure 8. Step response: K_p = 5

Observations:
Rise time = 0.0021 seconds
Settling time = 0.00686 seconds
Final value = 0.972
Overshoot (%) = 10.6
Peak amplitude = 1.08

3- For K_p=2.4, K_i=0, K_d=0, the result is shown in Fig.9

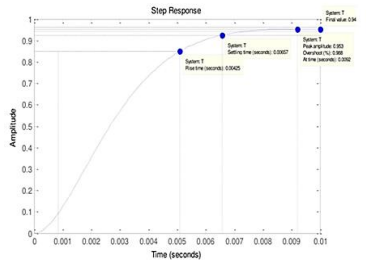

Figure 9. Step response: K_p = 2.4

Observations:
Rise time = 0.00425 seconds
Settling time = 0.00657
Final value = 0.94
Overshoot (%) = 0.988
Peak amplitude = 0.953

4- For K_p=2.4, K_i=0.01, K_d=0, the result is shown in Fig.10

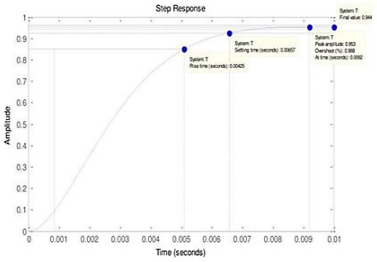

Figure 10. Step response: K_p = 2.4, K_i = 0.01, K_d = 0

Observations:
Rise time = 0.00425 seconds
Settling time = 0.00657 seconds
Final value = 0.944
Overshoot (%) = 0.988
Peak amplitude = 0.988

5- For K_p=2.4, K_i=0.01, K_d=0.1, the result is shown in Fig.11

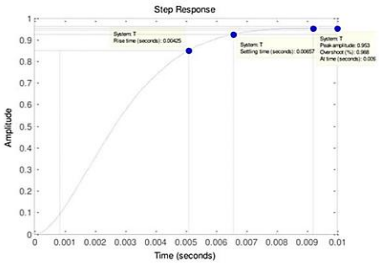

Figure 11. Step response: K_p = 2.4, K_i = 0.01, K_d = 0.1

Observations:
Rise time = 0.00425 seconds
Settling time = 0.00657 seconds
Final value = 0.944
Overshoot (%) = 0.988
Peak amplitude = 0.953

3.3 Frequency response

Frequency responses are obtained for different values of PID parameters.

1- For K_p=5, K_i=0, K_d=0, the result is shown in Fig.12

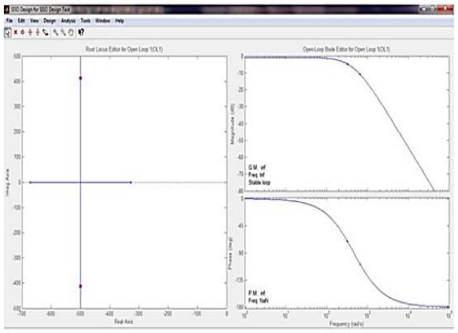

Figure 12. Various frequency plots for the compensated system during tuning (K_p = 5)

Observations
The root locus exhibited complex closed loop poles.
Both the phase and gain margin are infinite.
The system is stable.

2- For K_p=2.4, K_i=0.01, K_d=0.1, the result is shown in Fig.13

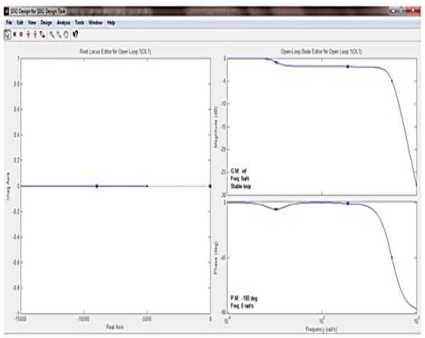

Figure 12. Various frequency plots for the compensated system during tuning (K_p = 2.4, K_i = 0.01, K_d = 0.1)

Observations
The root locus closed-loop poles changed from complex to real.
The system is still stable.
The infinite gain margin showed inherent stability.

4 GENERAL ANALTYIS

The effects of each of controller parameters, K_p, K_i, and K_d on the line following robot are summarized in the table 4.

Table 4.

Effects of increasing PID parameters

Parameter	Rise Time	Overshoot	Settling Time	Steady-state Error
K_p	Decrease	Increase	Small change	Decrease
K_i	Decrease	Increase	Increase	Eliminate
K_d	Small change	Decrease	Increase	No change

The difficulty of tuning increased with the number of parameters that are to be adjusted. To observe the response that resulted from the tuning adjustments, it is necessary to wait for several minutes. This made the tun-ing by trial-and-error a tedious and time-consuming task.

In practice, the stability of a mathematical model is not sufficient to guarantee acceptable system performance or even to guarantee the stability of the physical system that the model represented. This is because of the approximate nature of mathematical models.

The main problems associated with the implementtation of digital control are related to the effects of quantization and sampling. The advantages of digital control outweigh its implementation problems for most of the applications.

5 CONCLUSION

A line following robot was designed and built using Lego Mindstorms EV3 components. Digital control algorithms were developed. The advantages and limitations of implementing the digital control on different software were studied. The effectiveness of using PID controller for optimum line tracking was demonstrated by inspecting the movement pattern of the robot while following the track. To obtain the desired control response, K_p, K_i and K_d were successfully determined by tuning.

Used sources

[1] M. Mutheke: Digital control of a line following robot, Graduation Project, University of Nairobi, Kenya, 2015.

[2] A. Pathak: Line follower robot for industrial manufacturing process. International Journal of Engineering Inventions, Vol.6, Issue 10, PP. 10-17, 2017.
[3] N. Chowdhury: Algorithm for line follower robots to follow critical paths with minimum number of sensors. International Journal of Computer, Vol.24, No.1, PP. 13-22, 2017.
[4] P. Kumaresan: Case study: A line following robot for hospital management. International Journal of Pure and Applied Mathematics, Vol.116, No.24, PP.529-537, 2017.
[5] A. Attar: Line follower and obstacle avoidance bot using Arduino. International Journal of Advanced Computational Engineering and Networking, Vol.5, Issue 4, PP. 18-21, 2017.