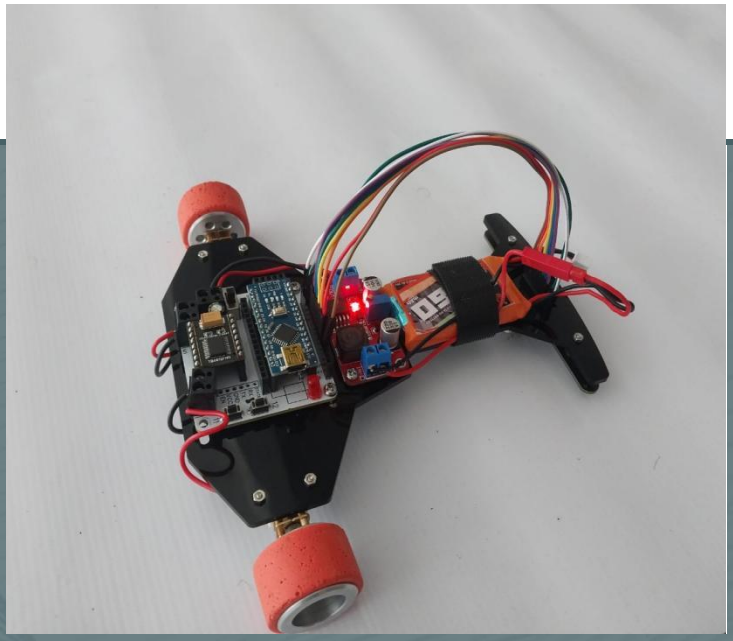
Line Follower Robot

Team Abhiyantra
JYOTHY INSTITUTE OF TECHNOLOGY

19-7-2023

Abstract

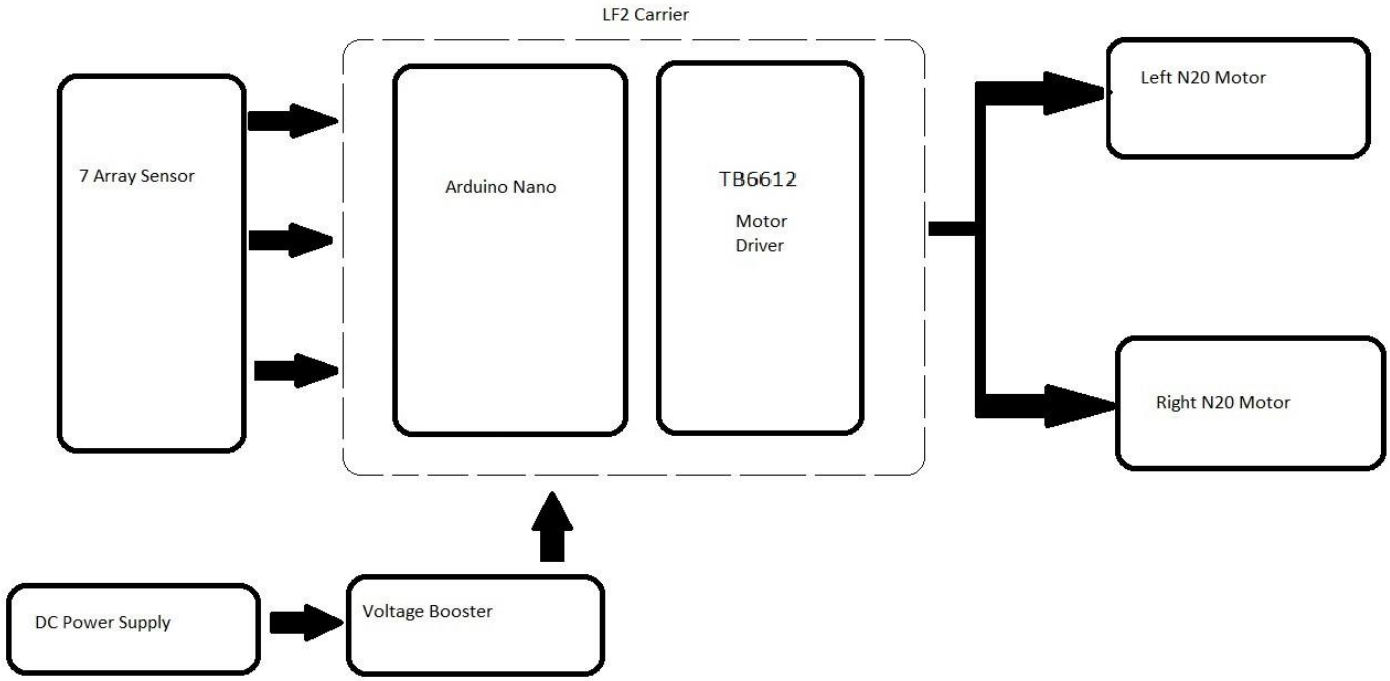
Line following robot is an autonomous vehicle which detect black line to move over the white surface or bright surface. In this project, the line following robot is constructed using Arduino nano microcontroller and consists of seven QRE1113 sensors along with two brushed DC Motors and a castor ball. The chassis is entirely made up of acrylic material. The infrared sensors are used to sense the black line on a white surface. When the infrared signal falls on the white surface, it gets reflected without absorbtion and when it falls on the black surface, it does not reflect back. In this system, a brushed dc motor with silicon wheels and a castor ball are used to control the robot car's direction that is left, right and forward. The Arduino nano controller is used to control the speed of DC motors from the TB6612fng driver circuit.



METHADOLOGY: DESIGN AND IMPLEMENTATION OF THE SYSTEM

This system is based on Arduino Nano microcontroller board. The system operation is forward, left and right direction of robot car by rotating two N20 motor attached with two wheels and a castor ball. The motion of the robot car depends on sensing of Infrared (IR) sensors. The overall system block diagram is as shown below.

BLOCK DIAGRAM



CIRCUIT DESING OF LINE FOLLOWER ROBOT :

The circuit connections for the line follower robot is given as follows,

i. Motor driver to arduino nano

MOTOR DRIVER PINS	ARDUINO NANO PINS
ENA	D9
ENB	D10
AIN 1	D4
AIN 2	D3
BIN 1	D6
BIN 2	D7

STAND BY	D5

ii. Arduino nano to sensor

ARDUINO NANO PINS	**SENSOR PINS**
VCC[5V]]	VCC
GND	GND
A0	0
A1	1
A2	2
A3	3
A4	4
A5	5
A6	6

iii. Motor to Motor driver

MOTOR PINS	**MOTOR DRIVER PINS**
Left motor pin 1	AO1
Left motor pin 2	AO2
Right motor pin 1	BO1
Right motor pin 2	BO2

iv. Other peripherals

DEVICE	**ARDUINO NANO PINS**
Switch 1	D11
Switch 2	D12
LED red	D13

WORKING PRINCIPLE :

The line follower robot is a popular type of autonomous robot that can navigate along a path or track, following a line marked on the ground. The robot uses sensors to detect the line's position and adjusts its movements accordingly to stay on track. A common configuration for line follower robots is using an array of sensors, such as 7 infrared sensors, to detect the line's position. Here's a general overview of the working principle of a line follower robot using a 7 array sensor setup:

Sensor array placement: The 7 infrared sensors are typically placed in a line across the front of the robot. The sensors are evenly spaced to cover the width of the line being followed.

Line detection: The infrared sensors are capable of detecting the difference in reflectivity between the line and the surface around it. The line will generally reflect less infrared light compared to the surrounding area. The robot scans this sensor array repeatedly to determine which sensors are over the line and which ones are not.

Sensor readings interpretation: The robot's microcontroller or onboard computer processes the readings from the sensor array. It calculates the position of the line relative to the center of the array. For instance, if the line is in the center of the array, all sensors should have similar readings. If the line is closer to the left, the sensors on the left side of the array will show less reflection (higher readings), and vice versa for the right side.

Line following algorithm: The robot uses a control algorithm to decide how to adjust its movements based on the sensor readings. One common algorithm is the proportional-integral-derivative (PID) controller. The PID controller calculates an error value based on the difference between the desired position (center of the sensor array) and the actual position of the line.

Motor control: The output of the PID controller determines how the robot's motors should adjust their speed or direction. If the robot detects that it's deviating from the centerline, it will adjust the motor speeds to correct its course and move back to the line's center.

Continuous operation: The robot continues to scan the sensor array, process the data, and make corrections in real-time. This feedback loop allows the robot to stay on the line and follow its path accurately.

COMPONENTS USED :

1. N20 Motors - 1000rpm 12V
2. Seven array QRE 11 sensor
3. TB6612fng motor driver module
4. Arduino nano
5. XL 6010 Boost converter
6. LiPo 2s 360 mAh 30C Battery
7. Wheels and tyres
8. Chasis

PID USAGE :

The PID (Proportional-Integral-Derivative) algorithm is commonly used in line follower robots to help them accurately follow a given path or track, such as a line on the ground. The goal of the line follower robot is to maintain its position over the line by adjusting its motion based on the line's position relative to the robot's sensors. The algorithm includes the following:-

Sensing: The robot is equipped with one or more line sensors, such as infrared sensors or reflective sensors, which detect the presence or absence of the line beneath them. These sensors provide feedback about the robot's position relative to the line.

Error Calculation: The PID algorithm works by continuously calculating the error, which is the difference between the desired position (the center of the line) and the actual position (detected by the sensors). The error is typically computed as follows:

error = desired_position - actual_position

Proportional Control (P): The proportional component of the PID algorithm is responsible for producing an output based on the current error. It is proportional to the magnitude of the error and helps the robot steer in proportion to how far it is from the line's center. The proportional control output is calculated as follows:

P_output = Kp * error

Here, Kp is the proportional gain, a constant that determines how aggressive the robot responds to the error. Higher values of Kp result in stronger corrections but may lead to overshooting or oscillations.

Integral Control (I): The integral component is used to address any steady-state error that might persist even when the proportional control is applied. It sums up the past errors and applies a correction to eliminate any bias. The integral control output is calculated as follows:

I_output = Ki * ∑(error)

Here, Ki is the integral gain, a constant that determines how much accumulated error affects the robot's motion. The integral gain helps in dealing with any small errors that the proportional control alone cannot eliminate.

Derivative Control (D): The derivative component anticipates future error by considering the rate of change of the error. It helps to dampen the robot's response, reducing overshooting and oscillations. The derivative control output is calculated as follows:

D_output = Kd * (error - previous_error)

Here, Kd is the derivative gain, a constant that determines the contribution of the rate of change of error. The derivative gain helps the robot respond more smoothly to the line's curves.

Summing the Outputs: The PID controller's final output is the sum of the proportional, integral, and derivative control outputs:

PID_output = P_output + I_output + D_output

Actuation: The PID output is then used to control the motors or actuators of the line follower robot. For example, if the robot's sensors detect that it is drifting to the right of the line, the PID output will instruct the robot to adjust its motors to turn left, bringing it back to the center of the line. The robot continuously performs this feedback loop, adjusting its motion in real-time to maintain its position over the line.

Tuning the PID gains (Kp, Ki, and Kd) is an essential step in getting the line follower robot to behave optimally. The PID values might vary from one robot to another. So the PID values should be fine tuned in every robot to achieve stable and smooth line following operation.

TECHNICAL ISSUES FACED :

- Silicon tyres were used which is far softer and has got a slick surface. It had no threads or grooves on it so there is as much contact between the tires and the track as possible. The arena that was provided to us was untidy. Silicon attracts too much of dust and hence, our robot was not able to complete the track. it got deviated before reaching the finish line for which we would suggest you to provide additional vertical force onto the wheels to make sure it won't be deviated from it's stationary position.

- We also faced some wiring issues from time to time for which we would suggest you to design a custom PCB according to your requirements, this will definitely solve the issue of lose connections which might as well lead to short circuit sometimes.

SCOPE FOR IMPROVEMENTS :

- Torque and Speed of the motors are inversely proportional:

In standard brushed DC motors, as the rpm of the motor starts to increase, the torque of the motor decreases, which can be one of the downside of higher rpm motors. To overcome this challenge one can opt for a motor that provides high torque even at higher speeds such as core less magnetic motors or industrial grade brush less DC motors.

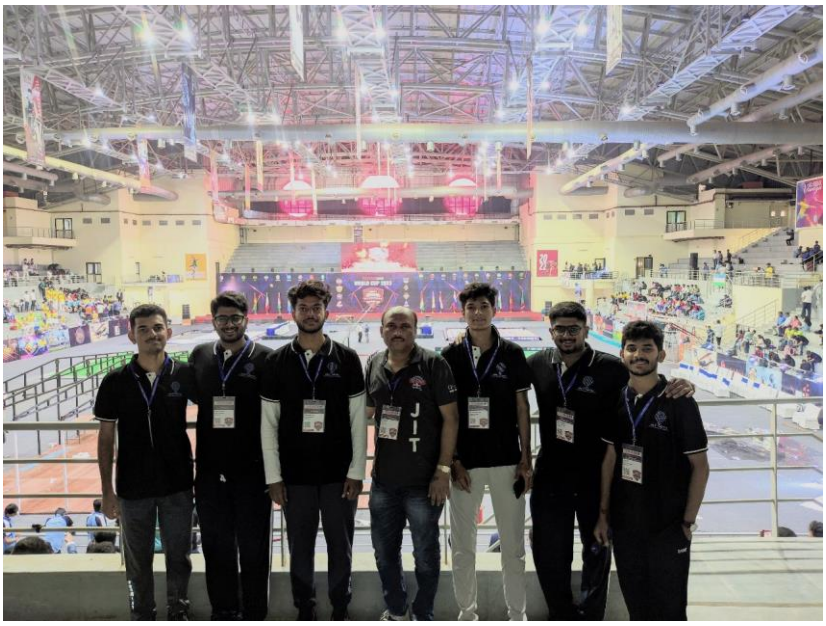
- Sensor Refresh rate :

Usually, at higher speeds, the Infrared sensors wont be able to detect the line accurately as the refresh rate of the sensor will we low for the desired speed. In order to over come this we can replace the sensors having lower refresh rates with the ones with comparatively higher refresh rate.

- Micro controllers with High clock frequency :

When considering sensors with higher refresh rate as well as motors with higher speeds, for the robot to follow the line more efficiently the refresh rate of the sensor, the speed of the motor and the clock cycle of the micro controller should go hand in hand.

- Downforce : flipping a propeller upside down will apply pressure onto the wheels by keeping the robot pushed towards the ground. The faster the propellers turn, the more vertical pressure gets applied onto the wheels which optimizes the traction and allow the robot to take curves extremely fast at higher speeds without getting deviated.



TEAM ABHIYANTRA