Robian19 Line Following Robots

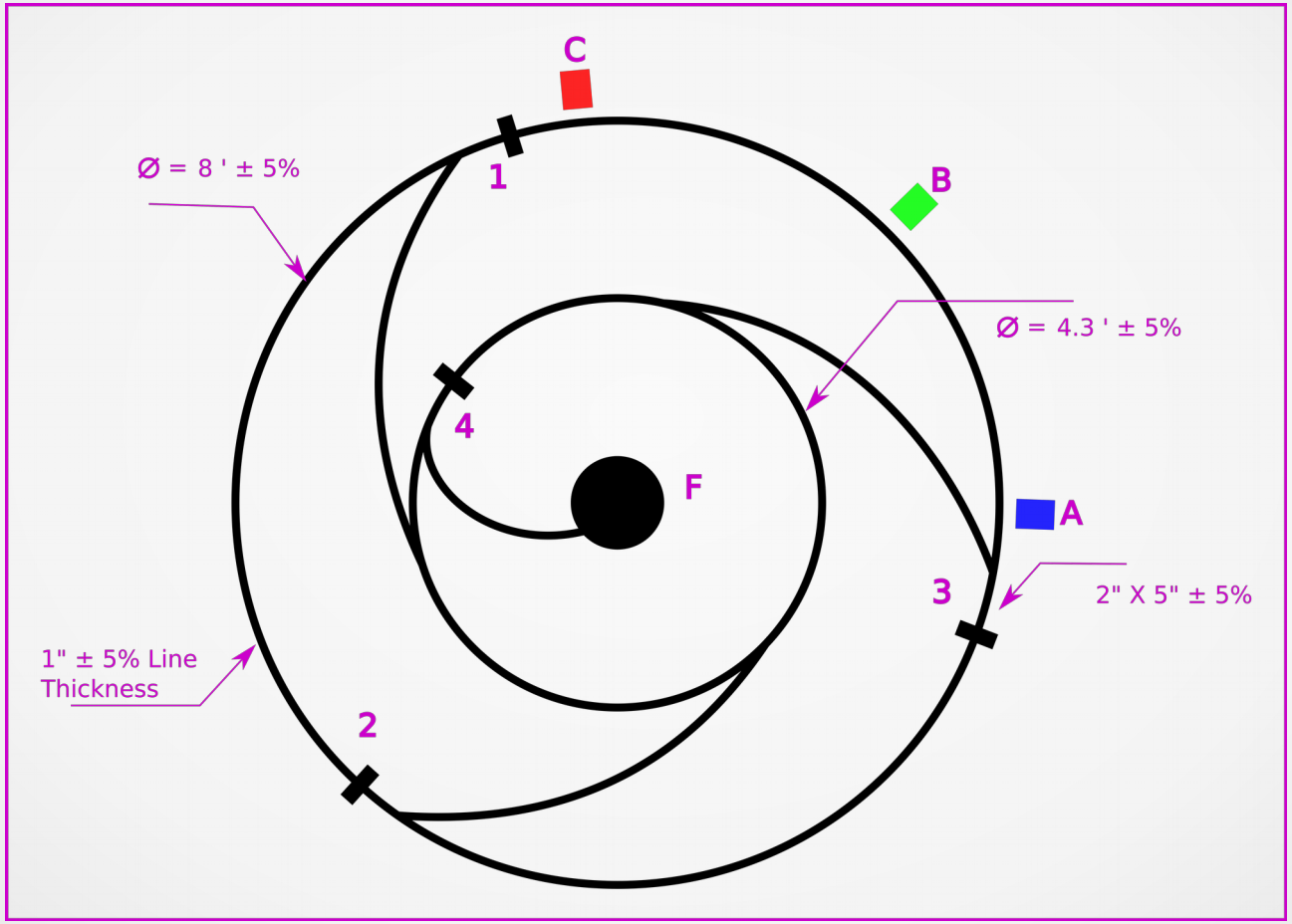
Presented by: Dr. Omar Ahmad

Overview

- **Part 1: Robian19 Introduction**
  - Robian19 Theme and Arena
  - Scoring System
  - Dates and Venue
  - Registration for Robian19
- **Part 2: Robot Design**
  - Introduction to design of main PCB
  - Making the Robot chassis
  - Choosing Motors
- **Part 3: Sensing requirements**
  - Sensor board for line sensing
  - Calibrating sensors
  - Getting deviation error from line sensors
  - Color sensor module TCS3200

Part 1: Robian19 Introduction

Robian19 Arena & Theme



Scoring System

No	Task Description	Points
1.	Robot moving and sensing	10
2.	Successfully marking a check point (4 checkpoints)	10 x 4
3.	Successfully complete 1st round of outer circle.	10
4.	Successfully detect and enter the correct turning point to the inner circle.	15
5.	Stop at the end point.	5
6.	Completion time bonus = (30 – 0.25 * seconds to complete track), Zero if robot takes more than 120 seconds	30
	Total	**110**

Dates and Venue for Robian19

Dates:

- Saturday 7th December 2019 (Qualification Rounds)
- Monday 9th December 2019 (Finals)

Venue:

- Seminar Hall, CUI Islamabad

Registration for Robian19

Register as teams:

- No more than 3 members per team.

Registration Fee:

- PKR 1500/- per team

What you will get?

- Free training sessions (seminars/workshops)
- Robot Main Board (PCB) to get you started.
- Certificates of participation
- Refreshments

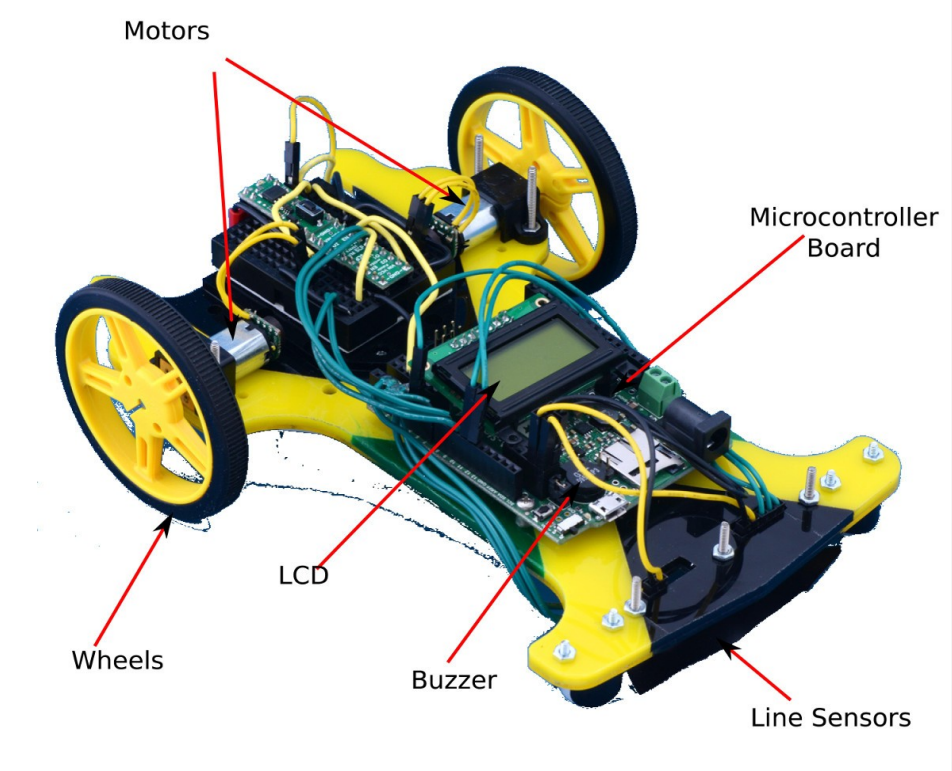
Contact:

- **Sir Imran Lodhi** (Head Robian19)
  - Room G02. ECE Department
- **Dr. Omar Ahmad** (Head Technical Committee)
  - Room 330. ECE Department

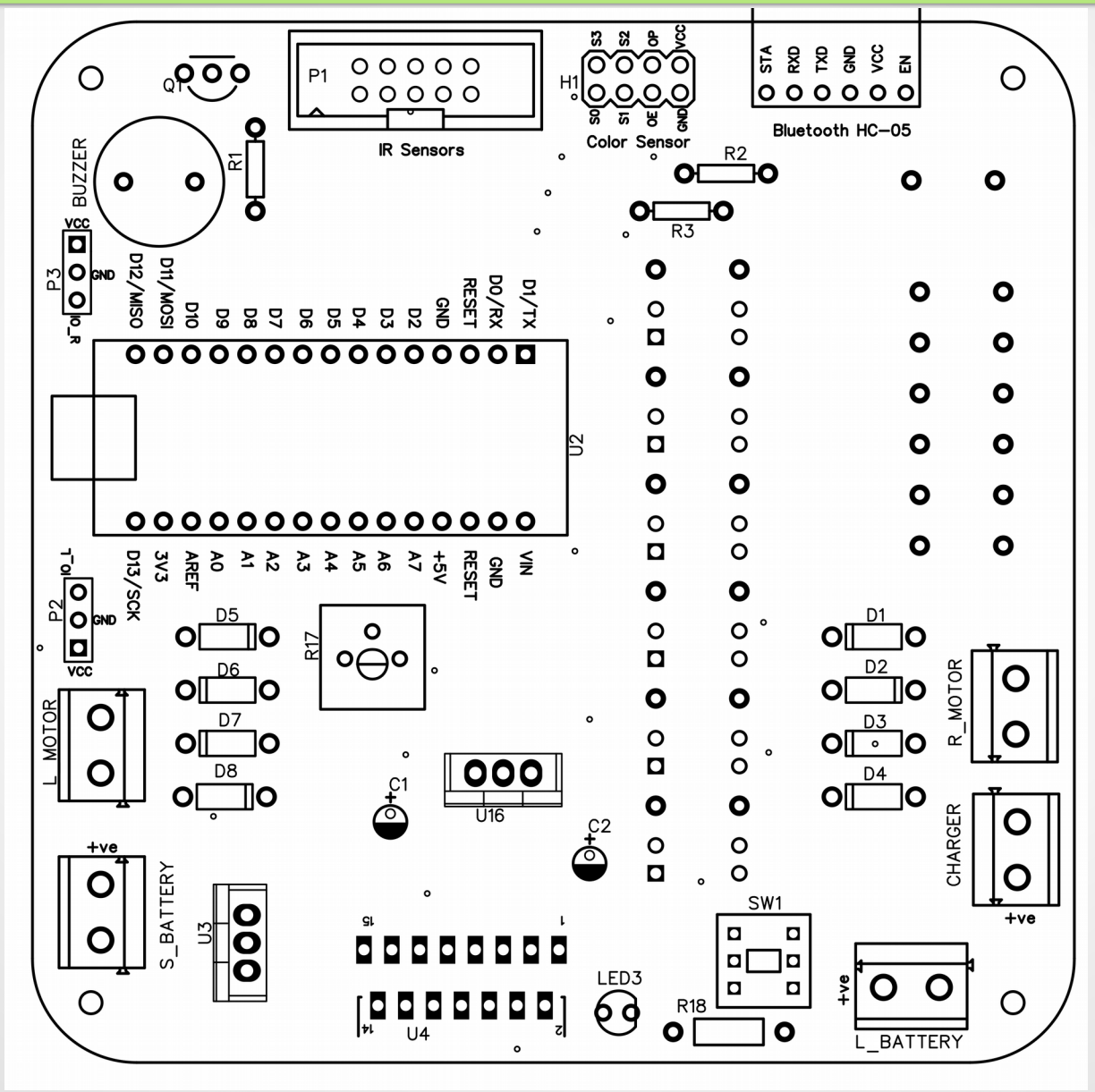
Email us at
robian@comsats.edu.pk

FOR ONLINE REGISTRATIONS

Part 2: Robot Design



Design of The Main PCB



Part List for Main Board

BOM_Robian Main Board_20191012105248

ID	Name	Designator	Quantity	Unit Cost	Total Cost	
1	Diodes 1N4448	D4,D3,D2,D1,D5,D6,D7,D8	8	0	0	
2	640 Ohm	R18	1	0	0	
3	Regulator LM7805	U3,U16	2	20	40	
4	Bluetooth Module HC-05	U1	1	450	450	Optional
5	Header-Male-2.54_1x3	P2,P3	2	20	40	
6	Color Sensor	H1	1	350	350	
7	L298N Motor Driver IC	U4	1	150	150	
8	0.1u Capacitor	C1,C2	2	0	0	
9	Switch On/Off 6 Pin	SW1	1	10	10	
10	Arduino Nano	U2	1	500	500	Optional
11	Opto Couplers CYPC817C	U5,U6,U7,U8,U9,U10	6	10	60	
12	Headers	R_MOTOR,L_MOTOR,CHARGER,L_BATTERY,S_BATTERY	5	10	50	
13	Transistor 2N3904/2N222	Q1	1	0	0	
14	IR Sesnors	P1	1	40	40	
15	100 Ohm	R1	1	0	0	
16	4.7 kOhm	R2,R3,R16	3	0	0	
17	220 Ohm	R4,R5,R6,R7,R8,R9	6	0	0	
18	1.8 kOhm	R10,R11,R12,R13,R14,R15	6	0	0	
19	BUZZER 5V	BUZZER	1	40	40	
20	1 kOhm	R17	1	0	0	
21	Power	LED3	1	5	5	
				Total →	**1735**	

The Robot Chassis

- Use a DIY type off the shelf kit
  - Easy and ready-made
  - Restricted in choice of motors and wheels etc
- Modify a toy car?
  - Cost effective if you have one already
  - Ackerman Drive
- Make one your self (highly recommended)
  - Cost effective
  - Spend on good motors
  - Good motors are generally difficult to get and then need coupling with wheels

http://www.maerivoet.org/website/software/arduino/manuals/electronics/robot-builders-bonanza.pdf

The Robot Chassis: Material Choice

- Wood ?
  - Saw, Drill, Glue nails etc will be needed
- Plastic?
  - Cutter, Drill, Screws, Magic Depoxy etc will be needed
- Modify a Toy?
  - Screw Driver, Cutter, etc

Choosing Motors

Important motor parameters:

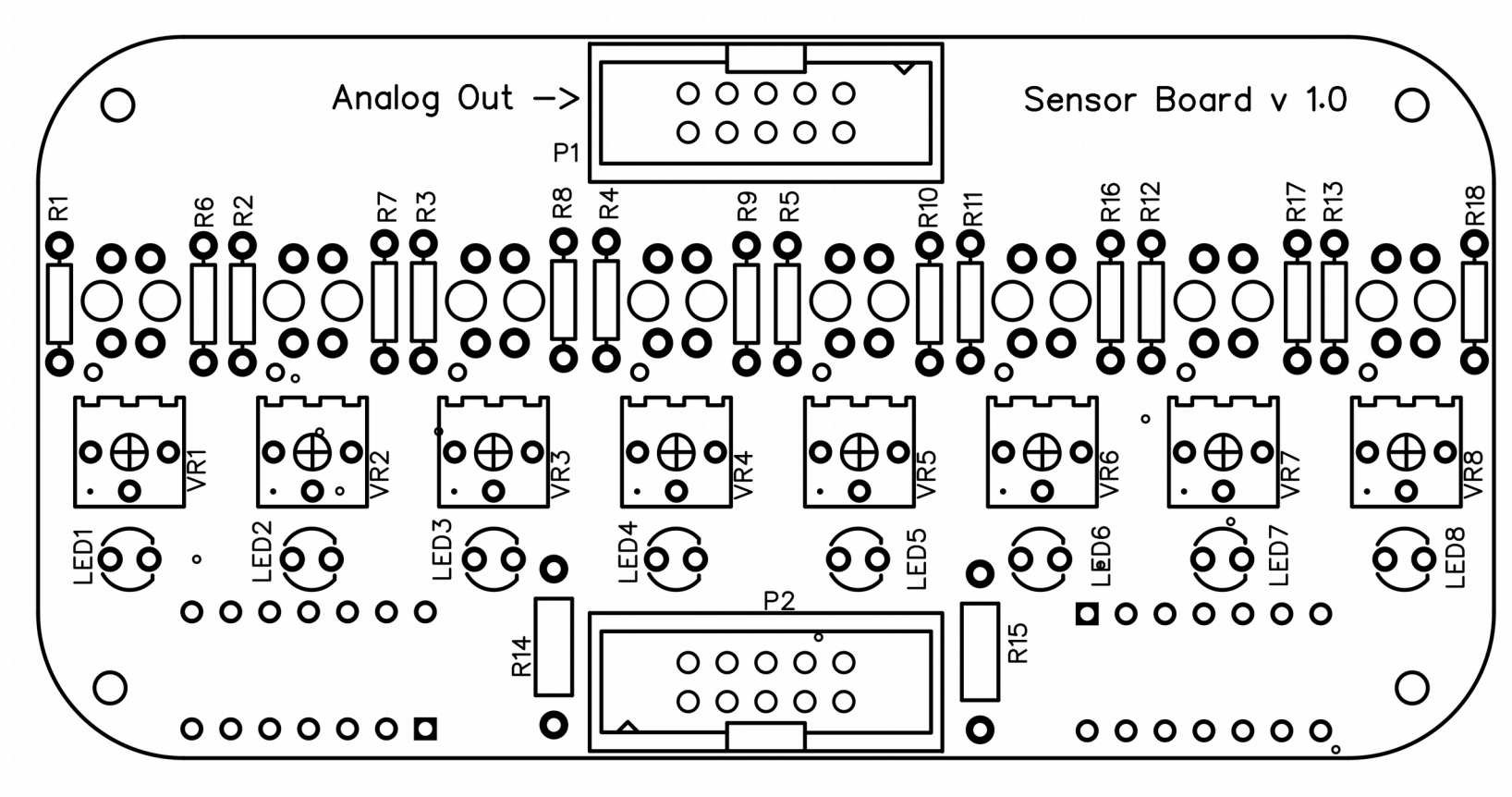
- Rated voltage
- Rated RPM
- Continuous torque
- Gear ratio

https://www.robotshop.com/blog/en/drive-motor-sizing-tool-9698

https://www.groschopp.com/understanding-continuous-speed-in-dc-motor-applications/

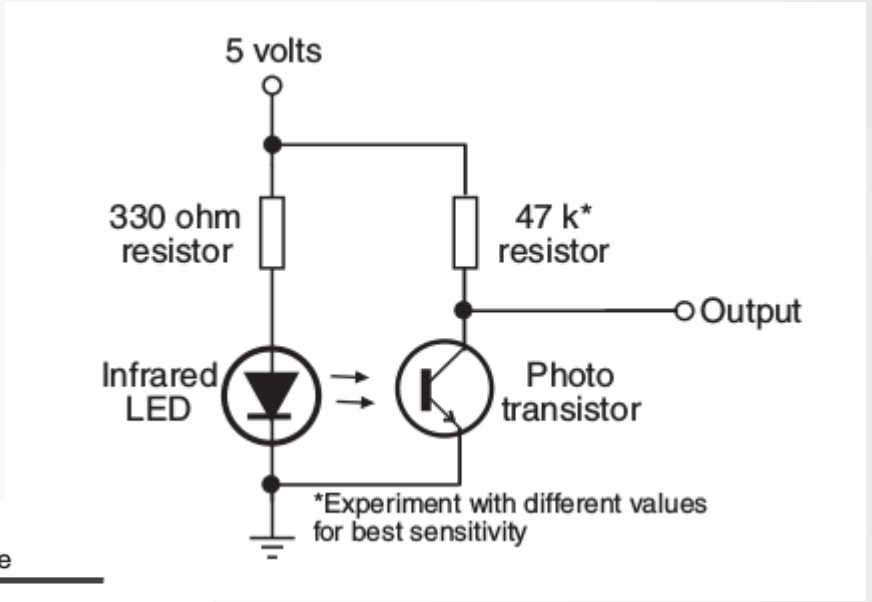
Part 3: Sensing Requirements

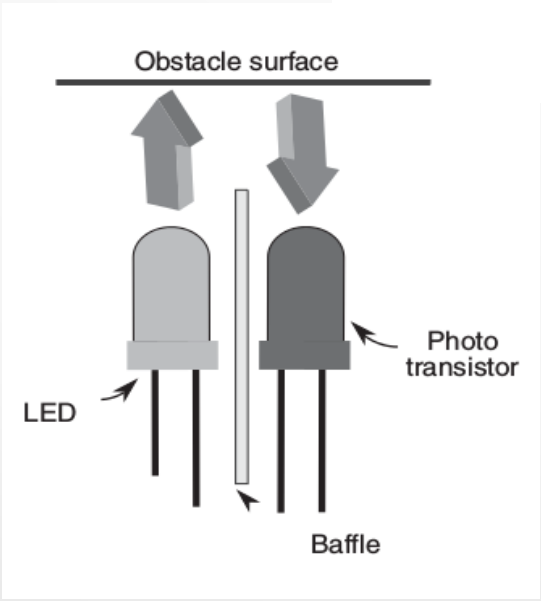
Sensor Board for Robian19



Sensors for Line Following

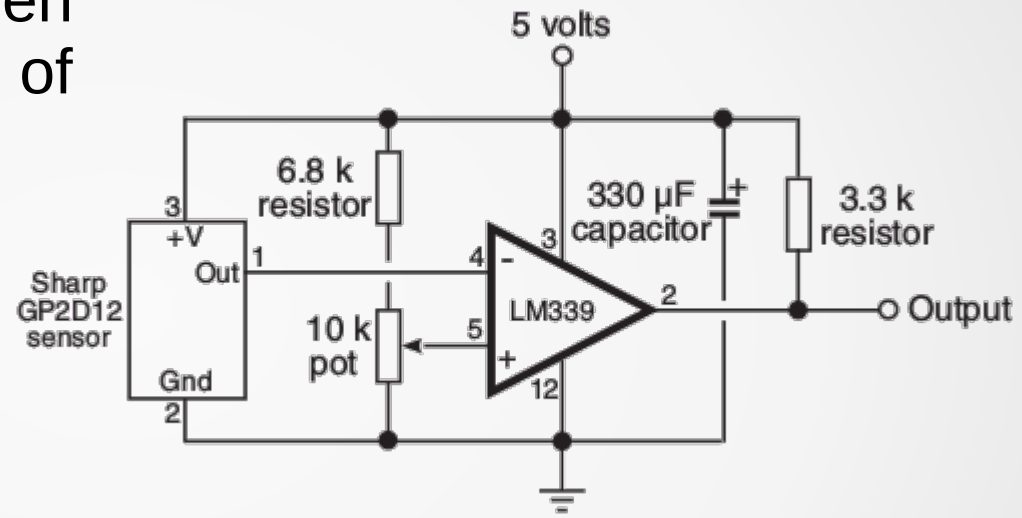
Most sensors usually provide analog outputs that can be connected to the ADC of a microcontroller





Sensors for Line Following

Alternatively this output can be fed to an amplifier and then connected to a digital IO pin of the microcontroller.



Which approach is better?

Analog vs Digital Interface

- Digital Interface is
  - Easy
  - Fast response
  - Needs Manual Calibration
- Analogue Interface
  - Needs design choices
  - Has slower response?
  - But can be calibrated by software

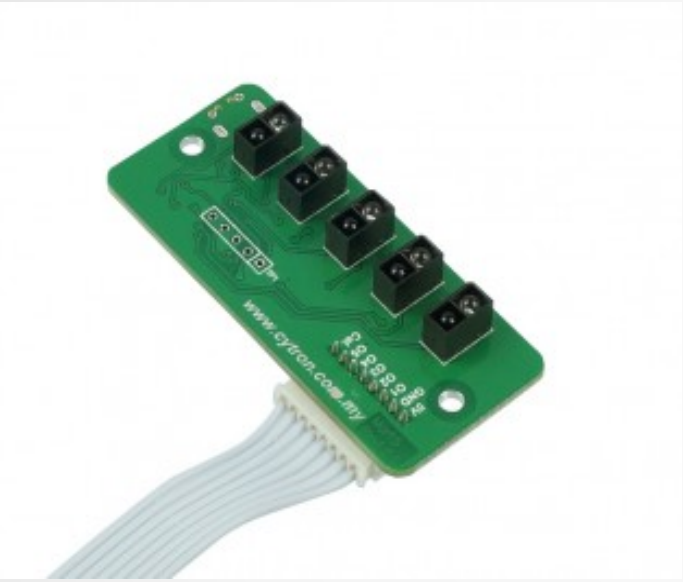
Analog Sensor Calibration

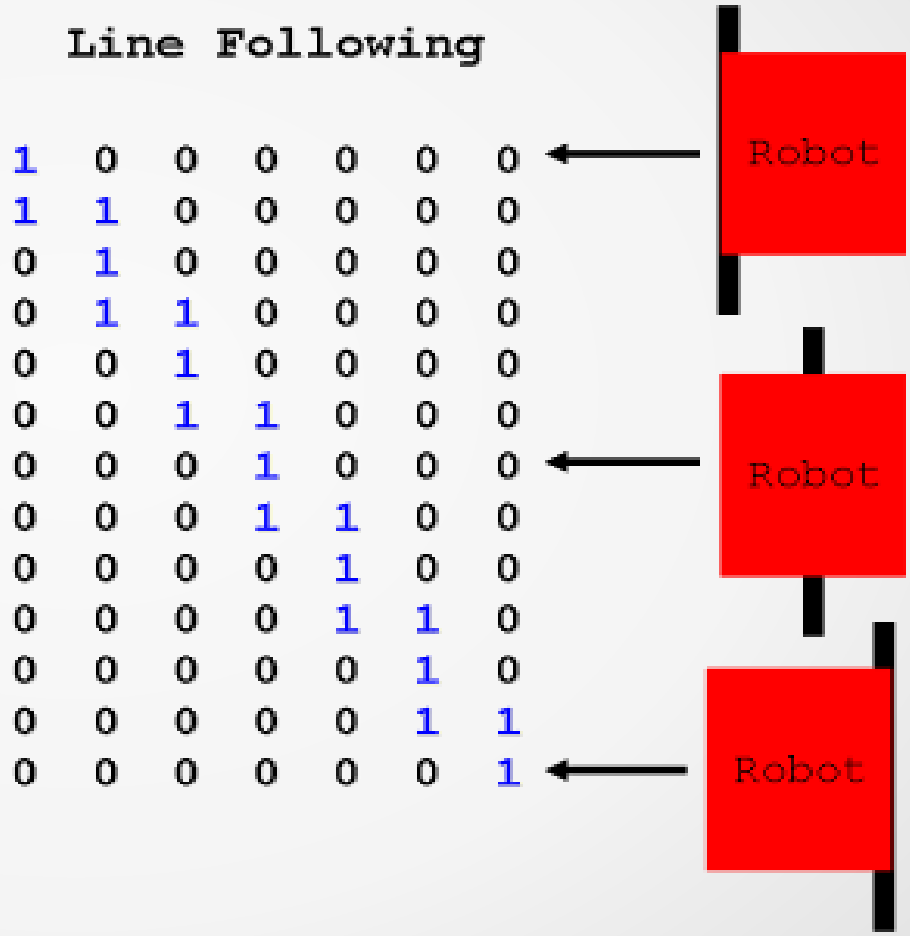
- Outputs from the analog sensors are in the range (0 to 1023).
  - `value = analogRead(channel);`
- You will have multiple sensors for line sensing and you will need to calibrate each one individually!!
- What is sensor calibration?
- Finding threshold values for the two cases (Why 2 thresholds?)
  - Sensor on the line
  - Sensor off the line

Auto-Calibration of Analog Sensors

- The robot starts calibrating its sensors when commanded (e.g. on a button press).
- Move the sensors over the line while the MCU records values in arrays
- Calculate the thresholds
- Save the thresholds in EPROM and load them in future.
  - Include EEPROM.h
  - Use functions EEPROM.get(), EEPROM.get()

Decisions Based on Line Sensor Output





Decisions Based on Line Sensor Output

- Issue different control commands to the robot based on the binary pattern of the sensor array

00100	0x04	The line is in the center. GO FORWARD
00001	0x01	Line on right-most sensor. GO HARD RIGHT
00010	0x02	Line on mid-right sensor. GO RIGHT
00110	0x06	Line on mid-right sensor. GO RIGHT
01100	0x0C	Line on mid-left sensor. GO LEFT
01000	0x08	Line on mid-left sensor. GO LEFT
10000	0x10	Line on left-most sensor. GO HARD LEFT

```
if((LFSensor[0]== 0 )&&(LFSensor[1]== 0 )&&(LFSensor[2]== 0 )&&(LFSensor[3]== 0 )&&(LFSensor[4]== 1 ))
   GoHardRight();
```

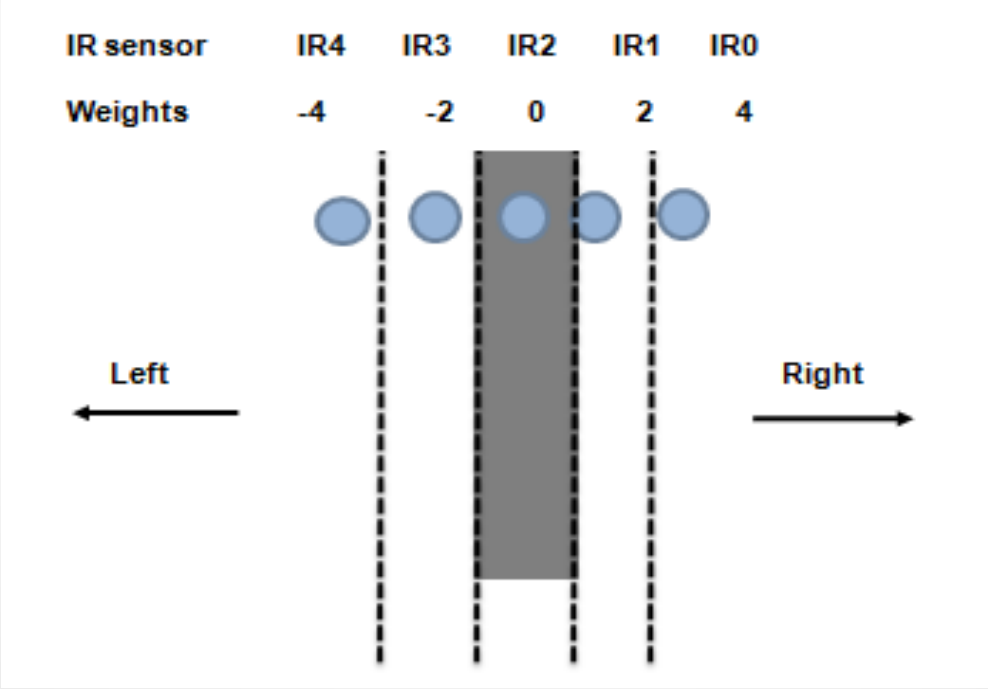
Issues With This Approach

- Leads to cluttered code.
- A very small set of control commands do not allow robust line following
- The robot oscillates about the line (zig-zag motion)and often loses the line

A Better Approach

Assign weights to each sensor and using a formula calculate a numeric ‘Error’ value

$$Current\ deviation = \frac{\sum_0^4 sensor\ reading\ (0/1)\ *\ corresponding\ weight}{number\ of\ sensors\ reading\ 0}$$



https://www.mathworks.com/help/supportpkg/arduino/examples/arduino-robot-line-follower-application.html?prodcode=SL

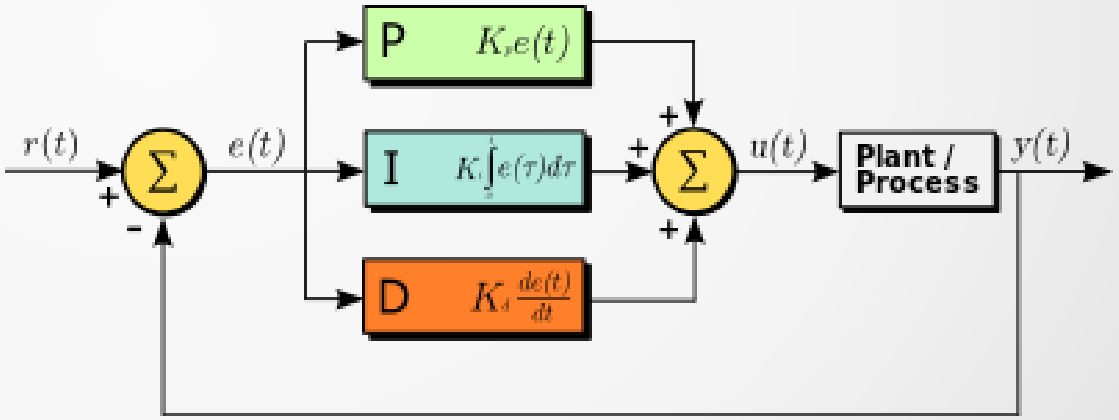
Benefits of this Approach

- Now we have one numeric position ‘Error’ value which is positive if the robot deviates to the right (e.g.) and negative when it deviates to the left!
- The magnitude is proportional to the deviation
- Easier to code and making the program more elegant!!
- Last but not the least it can be fed to our PID controller

What is PID Controller?

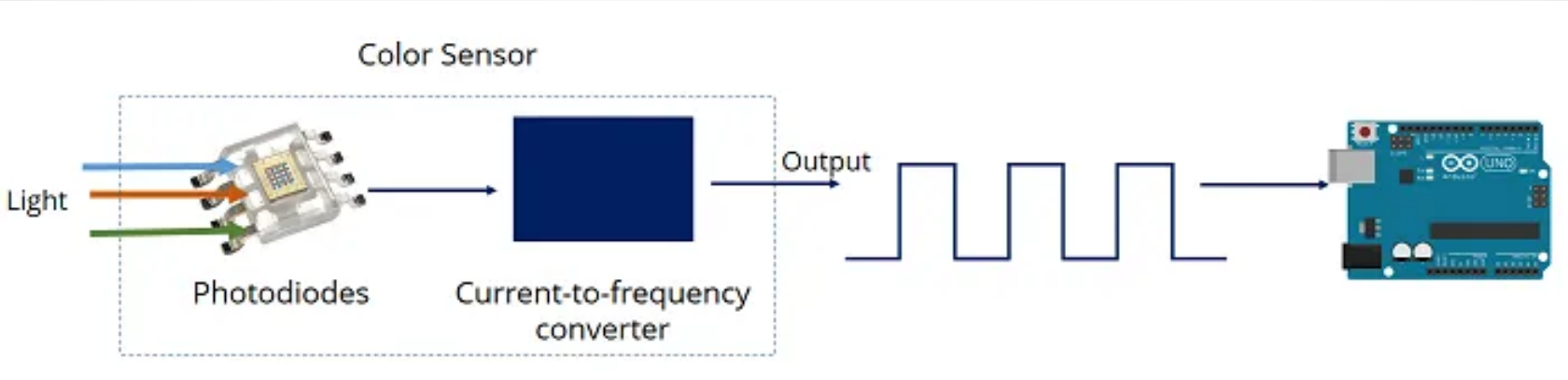
A Proportional–Integral–Derivative (PID) controller is a control loop feedback mechanism (controller) commonly used in industrial control systems. A PID controller continuously calculates an error value e(t) as the difference between a desired setpoint and a measured process variable y(t)and applies a correction based on proportional, integral, and derivative terms (sometimes denoted P, I, and D respectively) which give their name to the controller type.

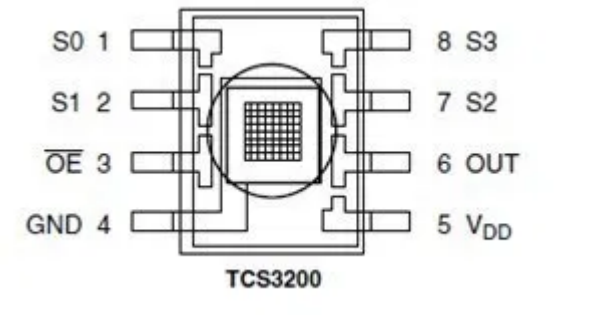
$$u(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(\tau)\,d\tau + K_\mathrm{d} \frac{de(t)}{dt},$$



Color Sensor TCS3200

Color Sensor Interfacing





https://randomnerdtutorials.com/arduino-color-sensor-tcs230-tcs3200/

Color Sensor Interfacing

Pin Name	I/O	Description
GND (4)		Power supply ground
OE (3)	I	Enable for output frequency (active low)
OUT (6)	O	Output frequency
S0, S1 (1, 2)	I	Output frequency scaling selection inputs
S2, S3 (7, 8)	I	Photodiode type selection inputs
VDD (5)		Voltage supply

Color Sensor Interfacing

Filter selection

To select the color read by the photodiode, you use the control pins S2 and S3. As the photodiodes are connected in parallel, setting the S2 and S3 LOW and HIGH in different combinations allows you to select different photodidodes. Take a look at the table below:

Photodiode type	**S2**	**S3**
Red	LOW	LOW
Blue	LOW	HIGH
No filter (clear)	HIGH	LOW
Green	HIGH	HIGH

Color Sensor Interfacing

Part of your Complex Engineering Problem

References

https://www.mouser.com/catalog/specsheets/TCS3200-E11.pdf
https://howtomechatronics.com/tutorials/arduino/arduino-color-sensing-tutorial-tcs230-tcs3200-color-sensor/
https://randomnerdtutorials.com/arduino-color-sensor-tcs230-tcs3200/
https://github.com/MajicDesigns/MD_TCS230
https://github.com/CloudyPadmal/TCS3200_Color_Sensor/blob/master/Color/Color.cpp