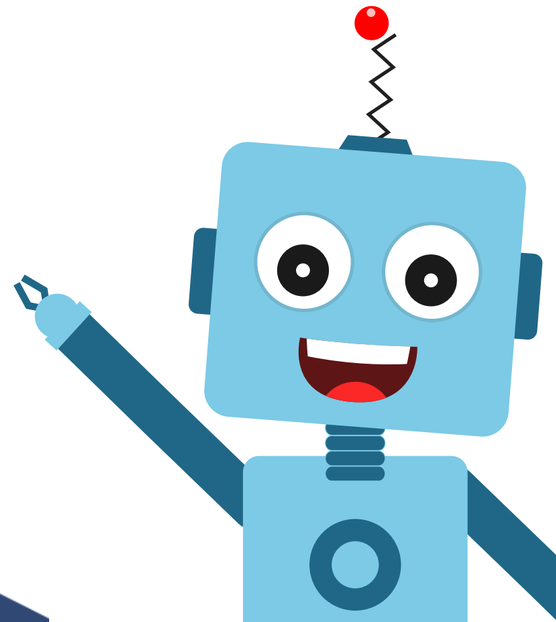
POWERED BY STEMpedia

Introduction to Sensor

Session 26



What are line-following robots? And how they work?

POWERED BY STEMpedia

- Line following robots are simple, they follow a line drawn on a surface. Just like how trains follow along a set track, they follow a dark line painted on the ground or a surface. They work by using infrared signals to detect a dark line on a bright surface or a brightly colored line on a dark surface. The line can also allow the robot to make turns.

Importance and applications of line-following robots

POWERED BY STEMpedia

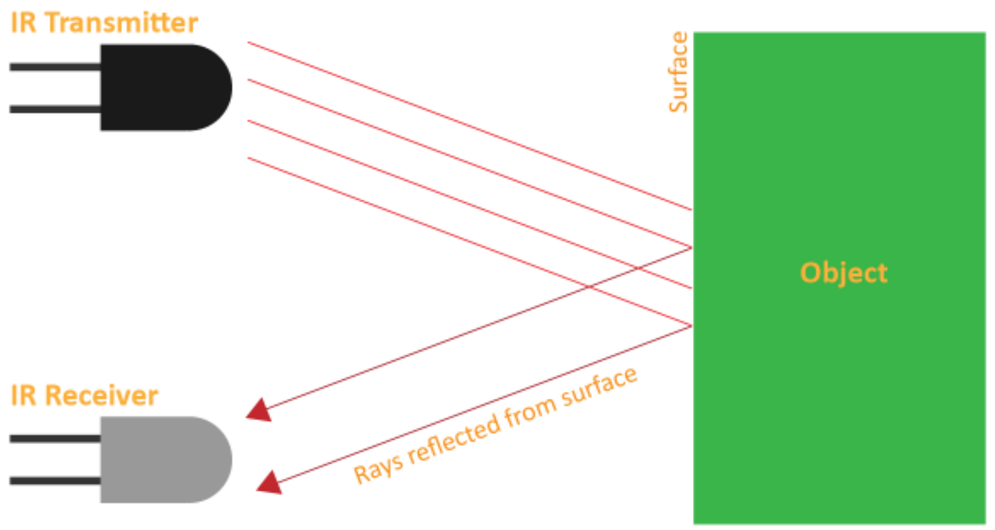
- Line following robots is very easy to build and easy to correct or maintain. Once built and turned on, they start to move along the path. This can be useful for tasks that need to be repeated often, like in factories.

- Applications:
    1. **They are used in factories to transport materials from one place to another.**
    2. **Line following robots is also used in automatic cars in museums, malls, or amusement parks to transport people.**
    3. **Robots that automatically clean the floor of a room.**

Calibrating IR Sensors

POWERED BY STEMpedia

- Understanding the Logic
    1. **An IR sensor consists of 2 LEDs: one which transmits the IR light and one which receives the IR light. When the IR rays are transmitted, they bounce from the nearest surface and get back to the receiver LED. That's how an IR sensor detects an object.**

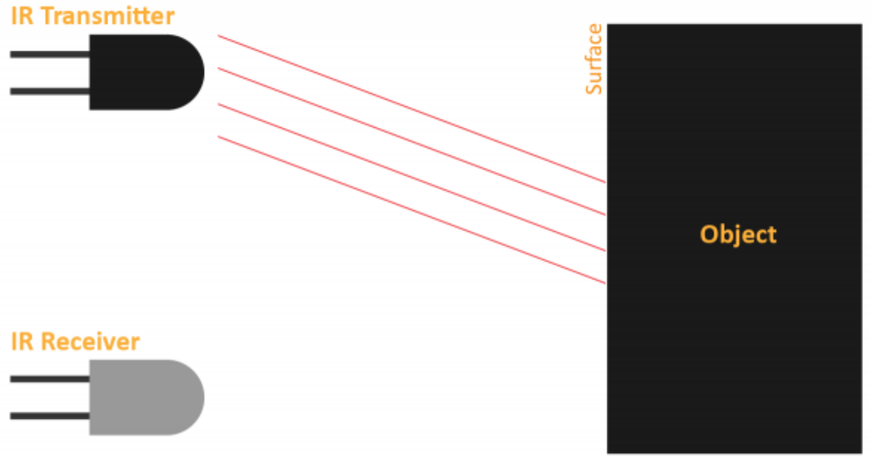


Calibrating IR Sensors

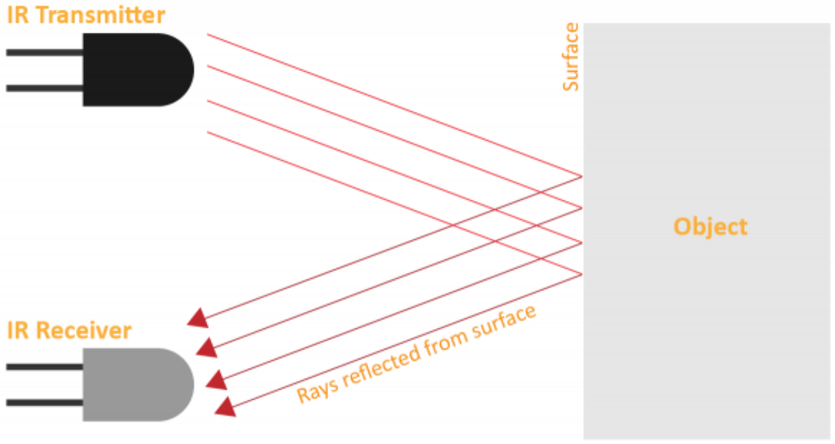
POWERED BY STEMpedia

- Understanding the Logic

1. **The dark surface will absorb more IR rays and as a result, the receiver will get fewer IR rays, and its output value is increased. This means that the sensor is active.**



2. **White or shiny objects will absorb less IR rays and as a result the receiver will get more IR rays, and output value decreases. This means that the sensor is inactive.**



Threshold Values

POWERED BY STEMpedia

We can get the sensor values in PictoBlox and based on that value we can estimate whether the surface is black or white.

1. **If the sensor detects the black line, its output value is increased. This means that the sensor is active.**
2. **If it detects the white area, its output value decreases. This means that the sensor is inactive.**

- The value above which the sensor detects the black line, we will call it the threshold value. If the sensor value is less than the threshold, it means that the sensor hasn't detected the line yet.

- Before we start using our line following robot, we need to first calibrate the IR sensors, i.e. make sure they're working fine. For that, we'll make a script in PictoBlox to calibrate them. We'll connect Quarky to PictoBlox and find out the threshold value for its IR (Infrared Right) & IL (Infrared Left) sensors.

- Let’s first find out the threshold value of the first IR sensor, i.e. the Left Sensor.

- Take out the track from your Quarky kit and bring the black line pretty close to the sensor. Observe the change in the variable value on the Stage. Note this value. Then bring the white space close to the sensor. You’ll notice that the value decreases. Note down the new value.

- The threshold value will be in between these two values. E.g., if the value when the sensor detects the black line is 2000 and when it detects the white space is 500, then we will keep the threshold at average of the noted values i.e., **(2000 + 500)/2**.

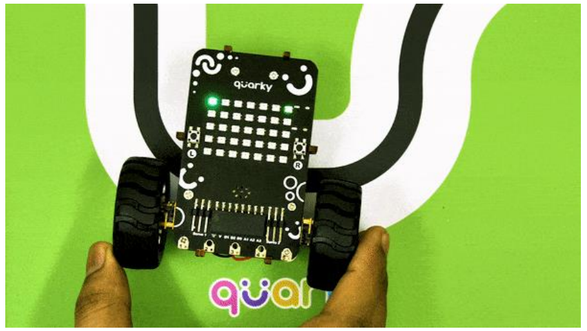
Please give the threshold value as per your calibration as it can be different for everyone, based on the surrounding light and surface.

Calibrating the IR Sensors

POWERED BY STEMpedia

- **Your calibration value is HIGH**: The black region will not be detected in this case. Reduce the calibration value.

- **Your calibration value is LOW**: The white region will not be detected in this case. Therefore, increase the calibration value.



- **Your calibration value is OK**. The white and black regions are detected accurately.

Activity: Calibration of IR Sensor

POWERED BY STEMpedia

- Connect Quarky to PictoBlox using a **USB cable**,
  - Select Python Coding as the coding environment and Quarky as the Board,
  - Open PictoBlox and create a new file, select the Serial port to establish a connection and press Connect, then define an object for Quarky.
- We need to import the time module in Python and use it.
- Let’s define Threshold Value.
- We will be writing the while loop by providing the condition as while True.
- Determine if left IR sensor is active/inactive; if active, illuminate designated LED. Syntax: Set LED
- Now, for adding a **time delay** of 0.1 seconds.
- Determine if left IR sensor is active/inactive; if active, illuminate designated LED. Syntax: Set LED.
- Now, for adding a **time delay** of 0.1 seconds, we will be using **time.sleep()** function

Activity: Calibration of IR Sensor

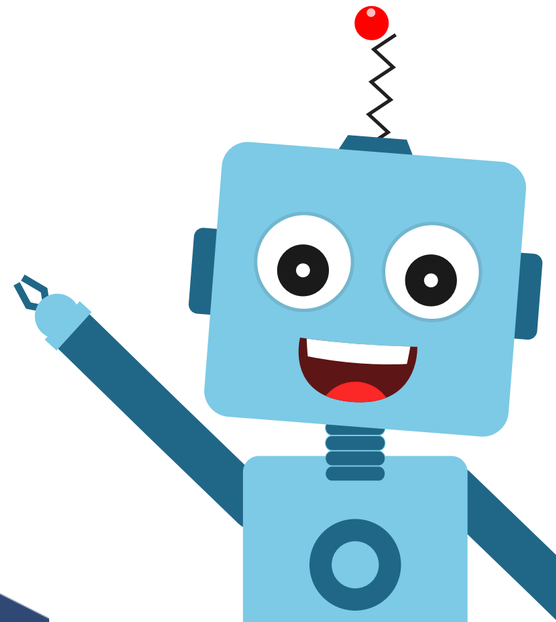
POWERED BY STEMpedia

```
import time
quarky = Quarky()
quarky.setirthreshold("IRL", 300)
quarky.setirthreshold("IRR", 300)
while True:
  if quarky.getirstate("IRL"):
    quarky.setled(1, 1, (0, 255, 0), 50)
  else:
    quarky.setled(1, 1, (255, 0, 0), 50)
  time.sleep(0.1)
  if quarky.getirstate("IRR"):
    quarky.setled(7, 1, (0, 255, 0), 50)
  else:
    quarky.setled(7, 1, (255, 0, 0), 50)
  time.sleep(0.1)
```

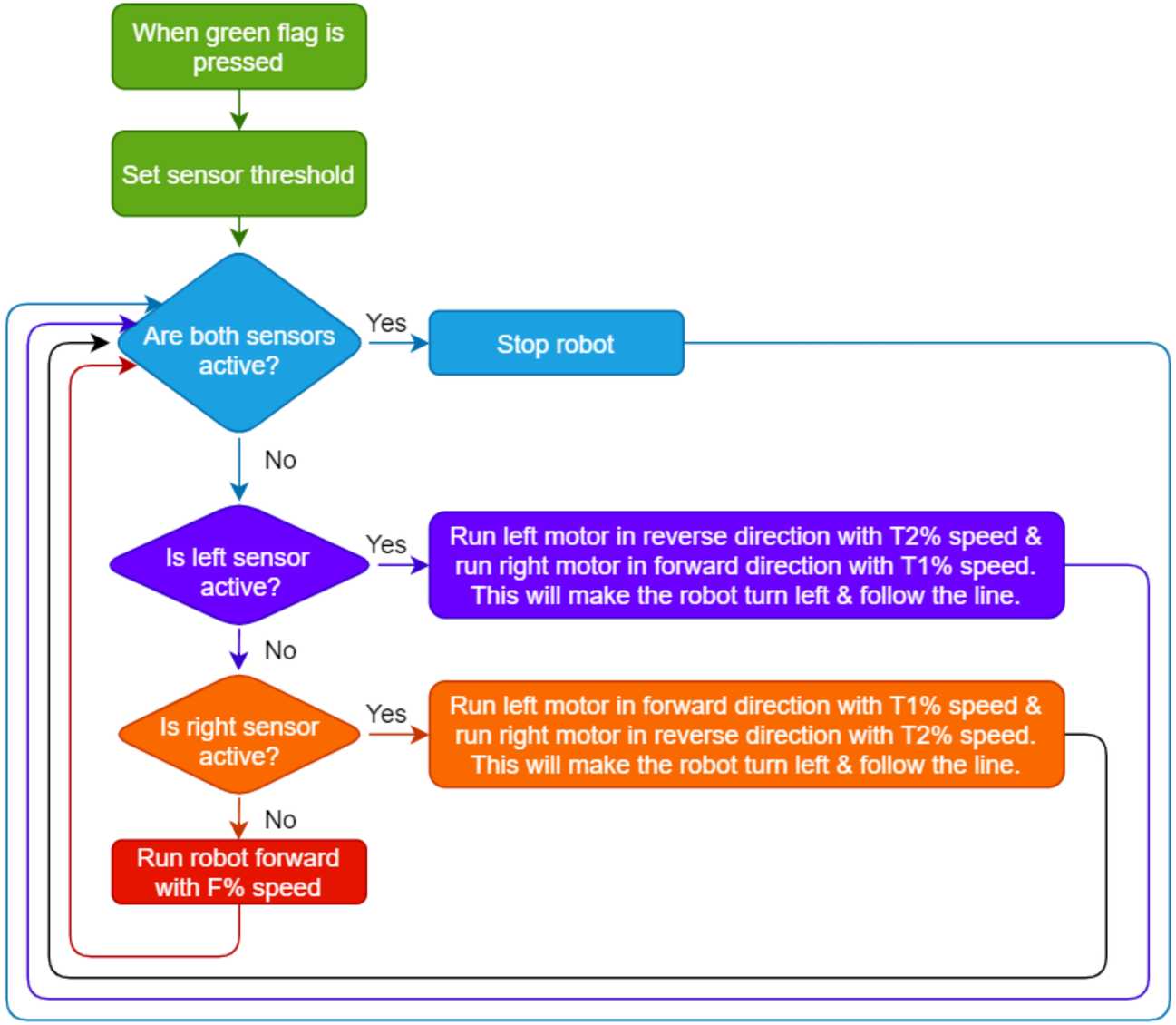
Final Output

POWERED BY STEMpedia

Logic of Line Following Robot



Logic of Making Line Following Robot



Understanding the Logic

POWERED BY STEMpedia

- First, we will set the left and right IR sensor threshold.
- If both the sensors are active, it means that the robot is at a crossroad and therefore must stop.
- If only the left sensor is active, this means that the robot is drifting towards its right and we need to bring it back on track. Therefore, we must make it move a little towards the left.
- If only the right sensor is active, this means that the robot is drifting towards its left. Therefore, we must make it move a little towards the right.
- If none of the sensor is active, it means that the black line is in between the sensors i.e., both the sensors are above white surface. Therefore, it should keep moving forward.
- Repeat steps 2-5.

Understanding the Logic

POWERED BY STEMpedia

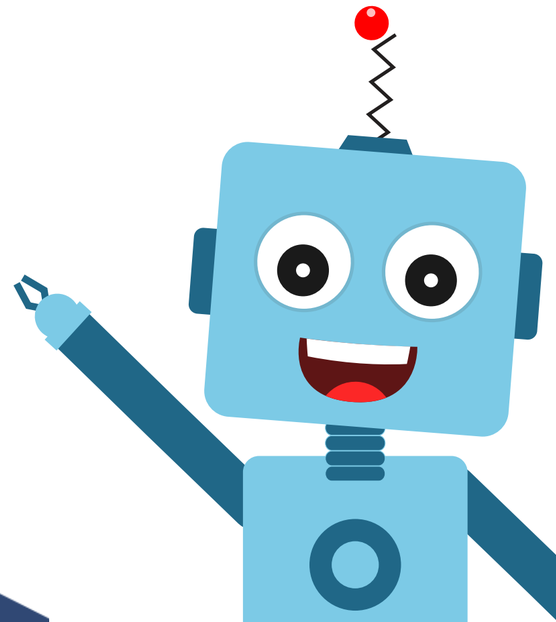
There are 3 important things we will use in a line following robot:

quarky.initializelinefollower(F,T1,T2)

1. F: The speed with which the robot will move forward when it has not detected a black line.
2. T1 & T2: When the robot is following the line and if one of the sensors say the left one, detects the black line, then the robot is off track and it has to turn left in order to get back on the track. And we know how to turn the robot left. The left motor moves backward and the right moves forward. But if both are moving at the same speed, then the robot motion will become jerky and inefficient. Hence, we will have two speeds for turning T1 and T2.
    1. T1 will be the speed with which the motor will move forward and
    2. T2 will be the speed with which the motor will move backward.
    3. We will have to set F, T1, and T2 during the programming and calibrate it for effective line following.

Now, let's make the script for Quarky based on this logic.

Activity: Line Following Robot



Activity: Line Following Robot

POWERED BY STEMpedia

1. Open PictoBlox and create a new file, select the Serial port to establish a connection and press Connect, then define an object for Quarky.
2. Select The Bluetooth Port from the Connect option. Make sure that the quarky is running and the Pictoblox Link software is running on the device. Select the Quarky and connect it by clicking on connect.
3. Select Python Coding as the coding environment and Quarky as the Board,
4. We learned how to determine threshold values in a previous activity. The syntax for setting these values for the left and right IR sensors is:
5. Let’s define Threshold Value.
6. We will be writing the while loop by providing the condition as while True.
7. Now, write the function which performs the automatic line following logic with the parameters specified in the initializelinefollower () function.
8. Press Run to run the code.

Activity: Line Following Robot(Output)

POWERED BY STEMpedia

```
sprite = Sprite('Tobi')
quarky = Quarky()

quarky.setirthreshold("IRL", 3000)
quarky.setirthreshold("IRR", 3000)
quarky.initializelinefollower(35, 40, 10)

while True:
  if not (quarky.getirstate(35) and
quarky.getirstate(34)):
    quarky.dolinefollowing()
  else:
    quarky.stoprobot()
```

POWERED BY
STEMpedia