

Manual: Line Following Robot

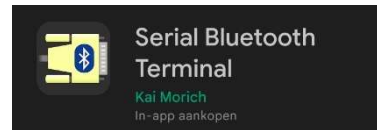
Introduction

Welcome to the manual of my implementation of a Line Following Robot! This manual will show you how to operate the robot using a free serial Bluetooth app on an Android phone.

Unfortunately, this app is not available on Apple App store. Please consider downloading a Bluetooth serial app on your Windows PC.

Serial Bluetooth app

On the Play Store, type “Serial Bluetooth Terminal” in the search bar. Download and install.

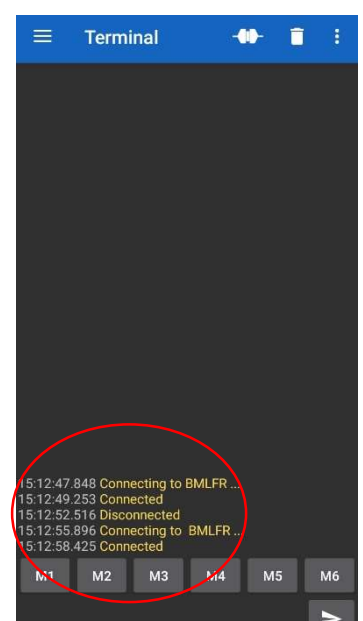
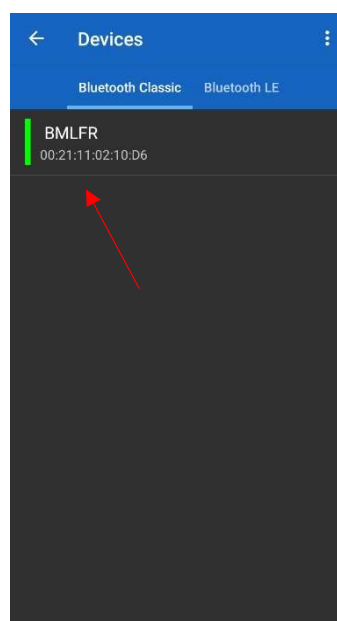
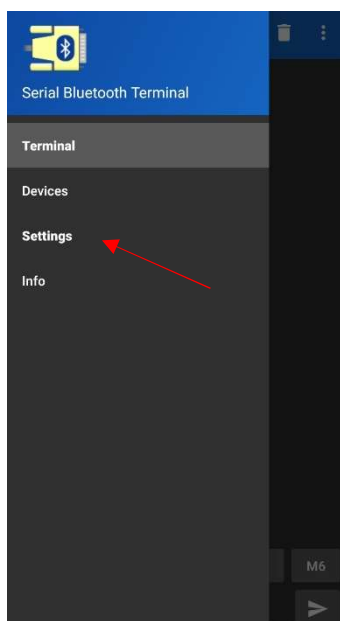


This app enables wireless communication between your smartphone and the HC-05. Execute following steps to establish a connection between the HC-05 and your smartphone:

1. Connect the HC-05 to your phone via Bluetooth functionality.
2. Open the Serial Bluetooth Terminal app: Open Settings: Tap the name of the HC-05 module. When successfully connected, your app will send you a notification.

Note that my module is called “BMLFR” (Bluetooth Module Line Following Robot). If you want to be able to change your Bluetooth module name and baudrate, you can check out this Instructable:

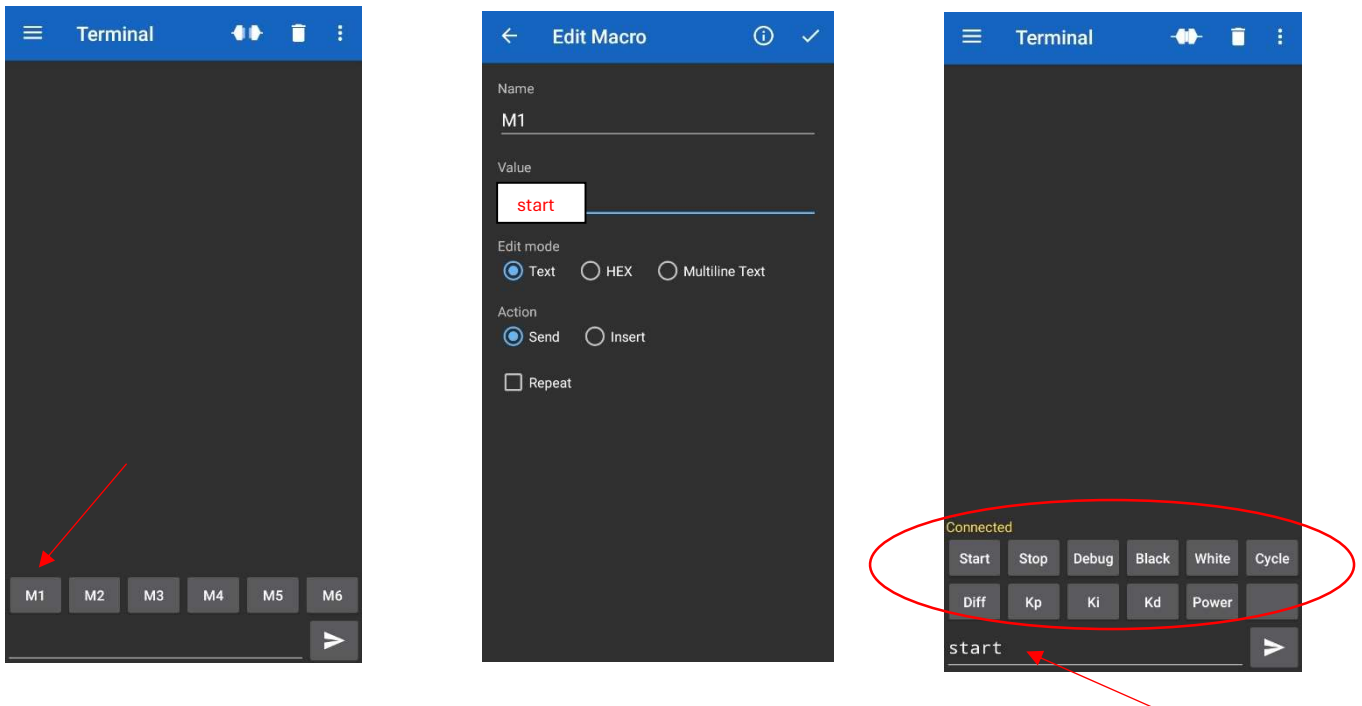
<https://www.instructables.com/AT-command-mode-of-HC-05-Bluetooth-module/>



Macros are useful tools that can help you write robot instructions more efficiently. You can assign a certain value to a Macro. When pressing a Macro, your assigned value will appear in the text box. This eliminates the time-consuming task of writing every command by hand.

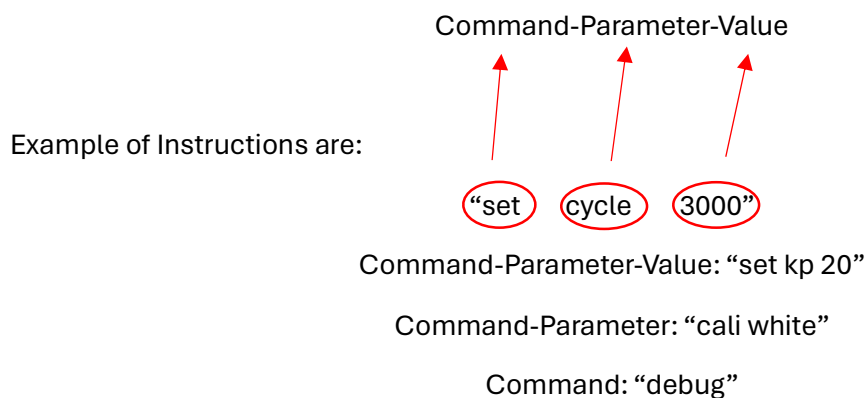
For example: I assigned the string “start” to Macro M1. When pressing the Start button, “start” will appear in the textbox.

I have created a Macro for every instruction the robot can execute. The last image shows all Macro’s I’ve created for easy robot parameter control.



Robot Instructions

The Arduino is coded to read incoming text information and execute it as instructions. The instructions need to be constructed the following way:



Note that some instructions only require a Command to function correctly (For example: “debug”). Others only require a Command and a Parameter (For example: “cali white”). **Beware the instructions are case sensitive!**

When an instruction is sent correctly, the Arduino will send a confirmation message to the Serial Bluetooth app. For example:

Your instruction: "set kp 20"

Arduino: "Kp set to 20"

If the Arduino does not recognize your command, you will receive an error message on the Serial Bluetooth app.

For example:

Your (faulty) command: "robot on"

Arduino: 'Unknown Command: "robot on"'

When an instruction is successfully sent, its corresponding value gets saved into the EEPROM memory of the microcontroller. When power is completely disconnected of the robot (and therefore the Arduino), it is not necessary to resend all new instructions.

All available **Commands** are:

- "start": The Line Following Robot starts moving. (No Parameter and Value needed)
- "stop": The Line Following Robot stops moving. (No Parameter and Value needed)
- "debug": Displays all Robot parameters on the serial Bluetooth app for information purposes. This can be useful when debugging. (No Parameter and Value needed) More information see chapter "Debug" below.
- "set": This instruction needs a Parameter and a Value to function correctly.
- "cali": Short for Calibration. This instruction only works with Parameters: "black" and "white."

All available **Parameters** for Command "set" are:

- "cycle": Adjust the cycle time of the Arduino.
- "power": Adjust the speed of the Robot.
- "diff": Affects intensity of Parameter "Power" depending on the error amount.
- "kp": Adjust Kp value.
- "ki": Adjust Ki value.
- "kd": Adjust Kd value.

All available **Parameters** for Command "cali":

- "black": Calibration of black values. (Only available with "cali" Command.)
- "white": Calibration of white values. (Only available with "cali" Command.)

All available **Values** of Command “set” are:

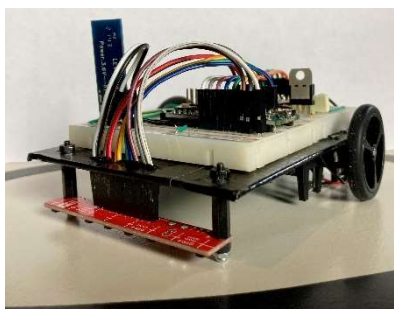
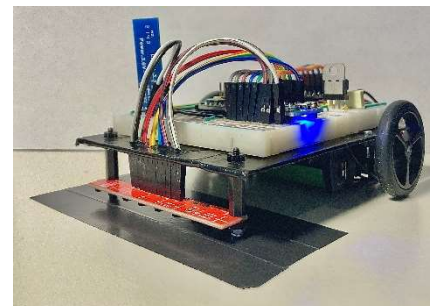
- “cycle”: Executed in milliseconds. Possible between 1-10000000.
- “power”: Possible between 1-255.
- “diff”: Possible between 0-1. You can assign values up to 2 numbers after the decimal point.
- “kp”: Possible between 0-100.
- “ki”: Executed in seconds. Possible between 0-1000. You can assign values up to 2 numbers after the decimal point.
- “kd”: Executed in seconds. Possible between 0-3. You can assign values up to 2 numbers after the decimal point.

Calibration

For the robot to be able to detect the black line properly, calibration is needed. This, however, is only necessary when track conditions change. Calibration is recommended every time you place the robot on a new kind of track. It can also be useful for debugging purposes: If your robot is not functioning as intended, you can always try recalibrating.

To calibrate black values, place the robot on a black surface with the same color intensity as the black line from your designed track. Send the following instruction using the Bluetooth terminal: “cali black”. After successfully calibrating, the Arduino will send a confirmation message to the Serial Bluetooth terminal: “Calibrating black... Done!”.

In this example, the robot will follow a black line of electrical tape placed on a white table. Therefore, the robot needs to be calibrated on a black patch of tape.



To calibrate white values, place the robot on a full white surface of your designed track, without a black line. Send the following instruction using the Bluetooth terminal: “cali white”. After successfully calibrating, the Arduino will send a confirmation message to the Serial Bluetooth terminal: “Calibrating white... Done!”

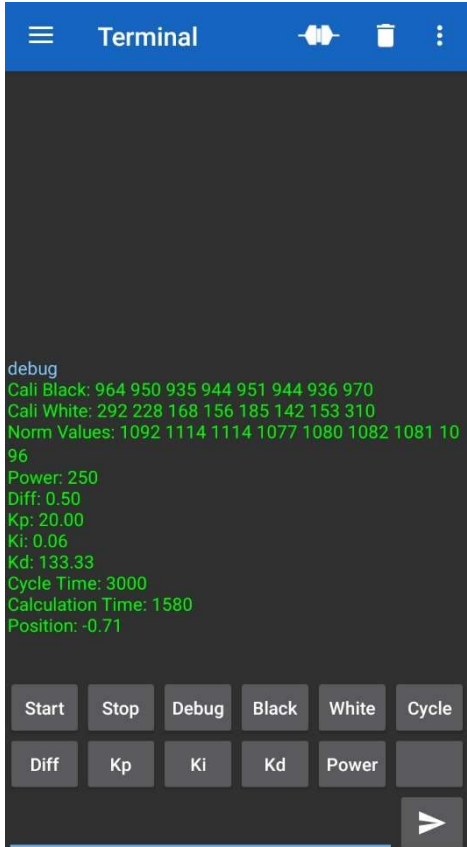
In this example, the robot is placed on the white table itself to get correct calibration values.

Debug

As mentioned before, sending Instruction “debug” will make the Arduino send all robot parameter values for easy debugging. Some values have not been mentioned before:

- Cali Black: Displays all calibrated black values. Since the QTR8A is an array of 8 sensors, logically 8 sensors will be calibrated. Ideally, all “cali black” values should approach a value of 1000.
- Cali White: Displays all 8 calibrated white values. Ideally, all cali white values should approach a value of 0.
- Norm Values: All Black and White sensor values after normalization (PID purposes, see Instructable).
- Calculation Time: The full time, in microseconds, to read and implement all code in the Arduino. It is highly recommended to keep this time as short as possible.
- Cycle Time: Recommended to keep at 2 x Calculation Time.
- Position: Represents a value between -30 and 30:
 - If position = 0, the robot is exactly in the middle of the black line.
 - If position = -30, the robot is placed far left.
 - If position = 30, the robot is placed far right.

The robot uses a PID controller to get position to stay at 0 when moving on the track.



```
Terminal
debug
Cali Black: 964 950 935 944 951 944 936 970
Cali White: 292 228 168 156 185 142 153 310
Norm Values: 1092 1114 1114 1077 1080 1082 1081 1096
Power: 250
Diff: 0.50
Kp: 20.00
Ki: 0.06
Kd: 133.33
Cycle Time: 3000
Calculation Time: 1580
Position: -0.71
```

Start Stop Debug Black White Cycle
Diff Kp Ki Kd Power

First Time Starting

Before starting, it is recommended to calibrate first and set all parameters to the following values:

- Power: 50
- Diff: 0.5
- Kp: 10
- Ki: 0
- Kd: 0
- Cycle Time: 2 x Calculation time

When the robot is able to complete a full lap without stopping, it is highly encouraged to experiment with all values! Assign extreme values to all Parameters and find out how the robot reacts. Above all else, have fun!

Acknowledgments

Thank you for reading the Manual of my implementation of the Line Following Robot. For more information, check out my Instructable using following link:

<https://www.instructables.com/Line-Following-Robot-8/>