Line Following Robot

by Maaskas

On this Instructable I would like to show you how I have built a Line Following Robot. Each chapter is written as a guide to help you build such a robot on your own. As visible on the video provided ***here*** the Line Following Robot is able to follow a black line in different speeds, but only maximum speed is visible on the video.

Link video: https://www.youtube.com/watch?v=eKb15k90Kxc

In order to complete this project, an intermediate knowledge of Arduino boards, Arduino coding, PID controllers and electronics is required. If you lack knowledge in any of those categories, I highly recommend researching these topics first before starting this project.

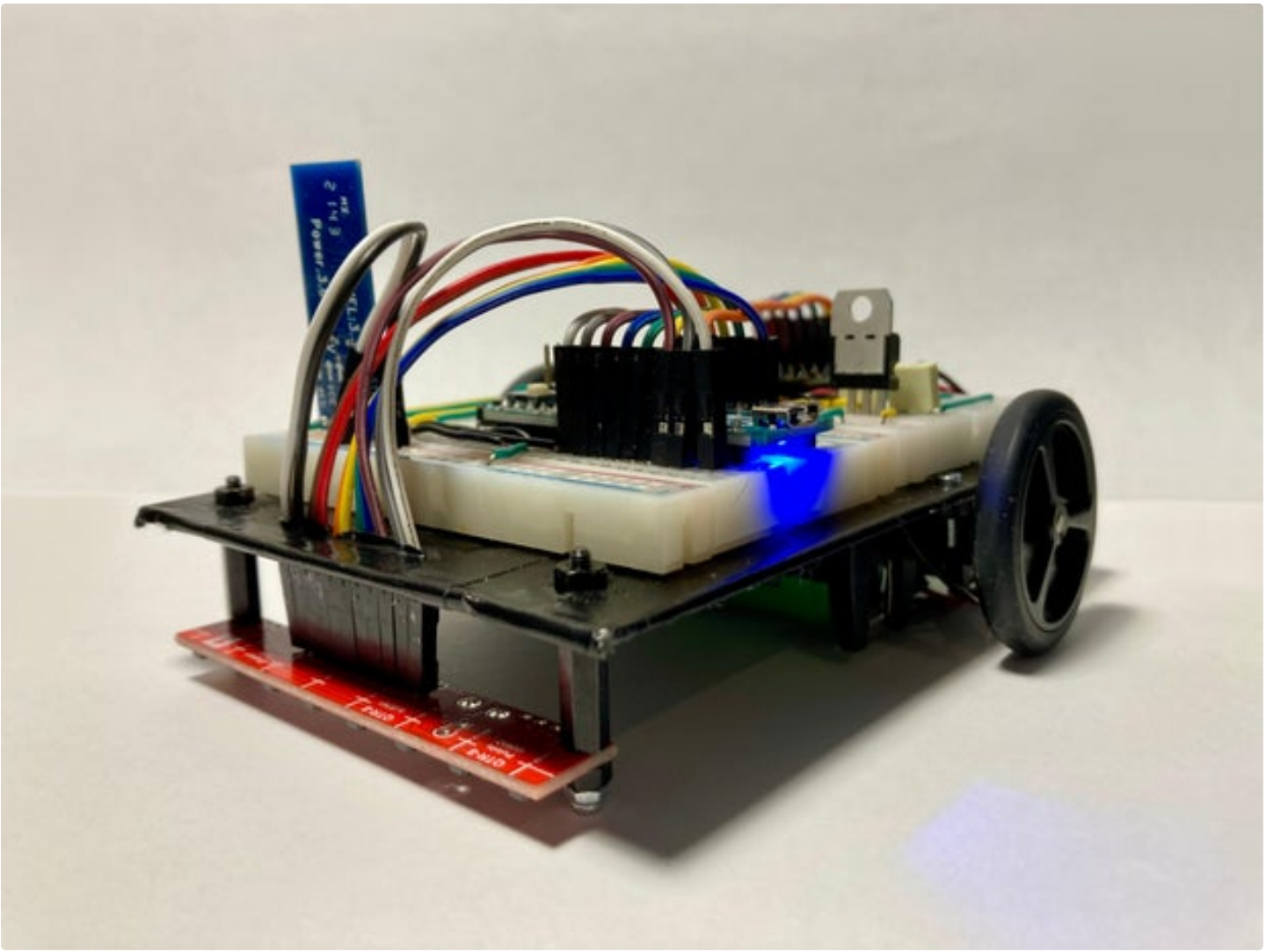
Before continuing, I would like to share that English is not my native language. I have written this Instructable to the best of my ability. I would like to ask for your forgiveness if you read any spelling or grammar issues.

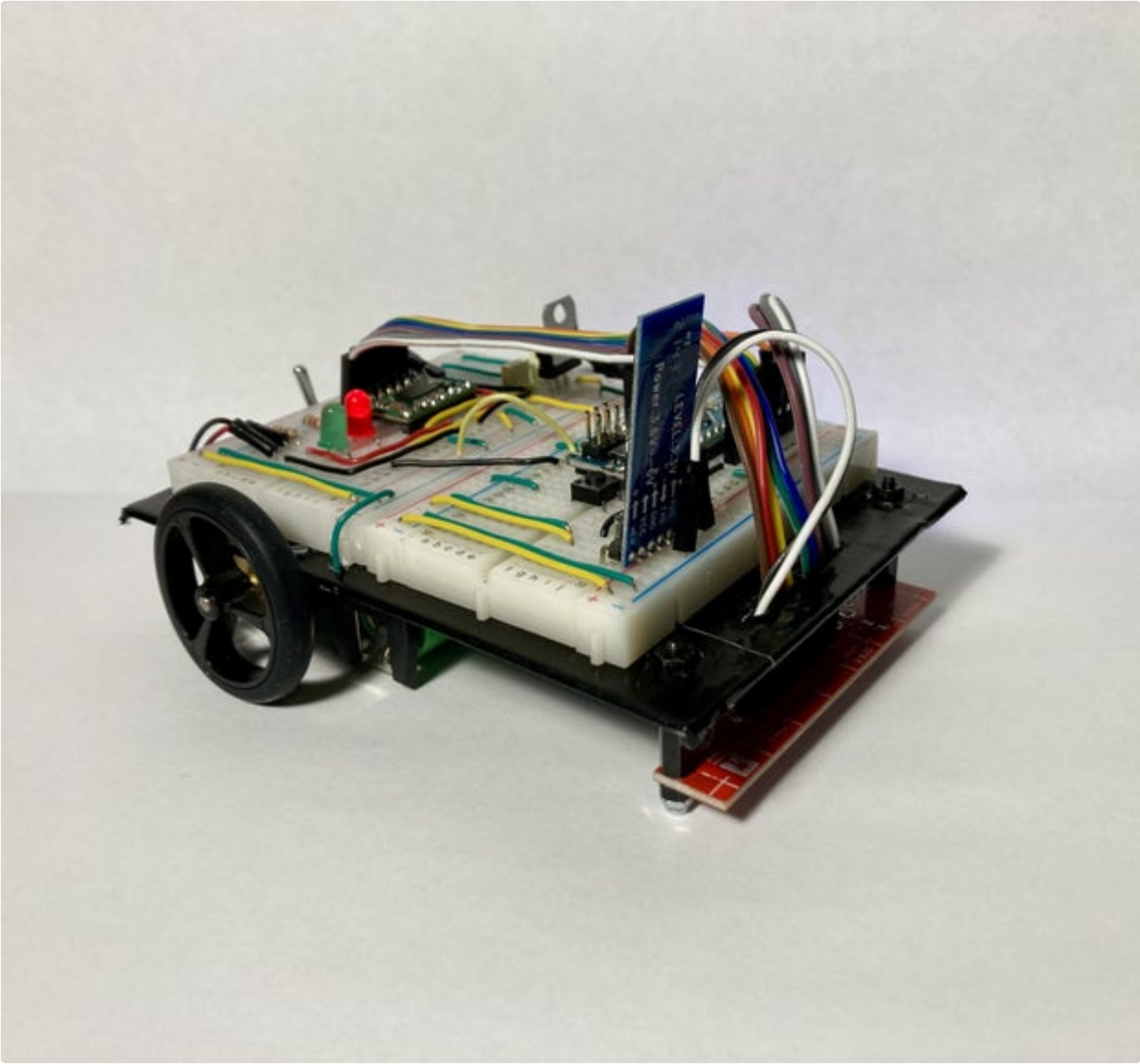
This Instructable is made possible with the help of HOGENT University in Ghent, Belgium.

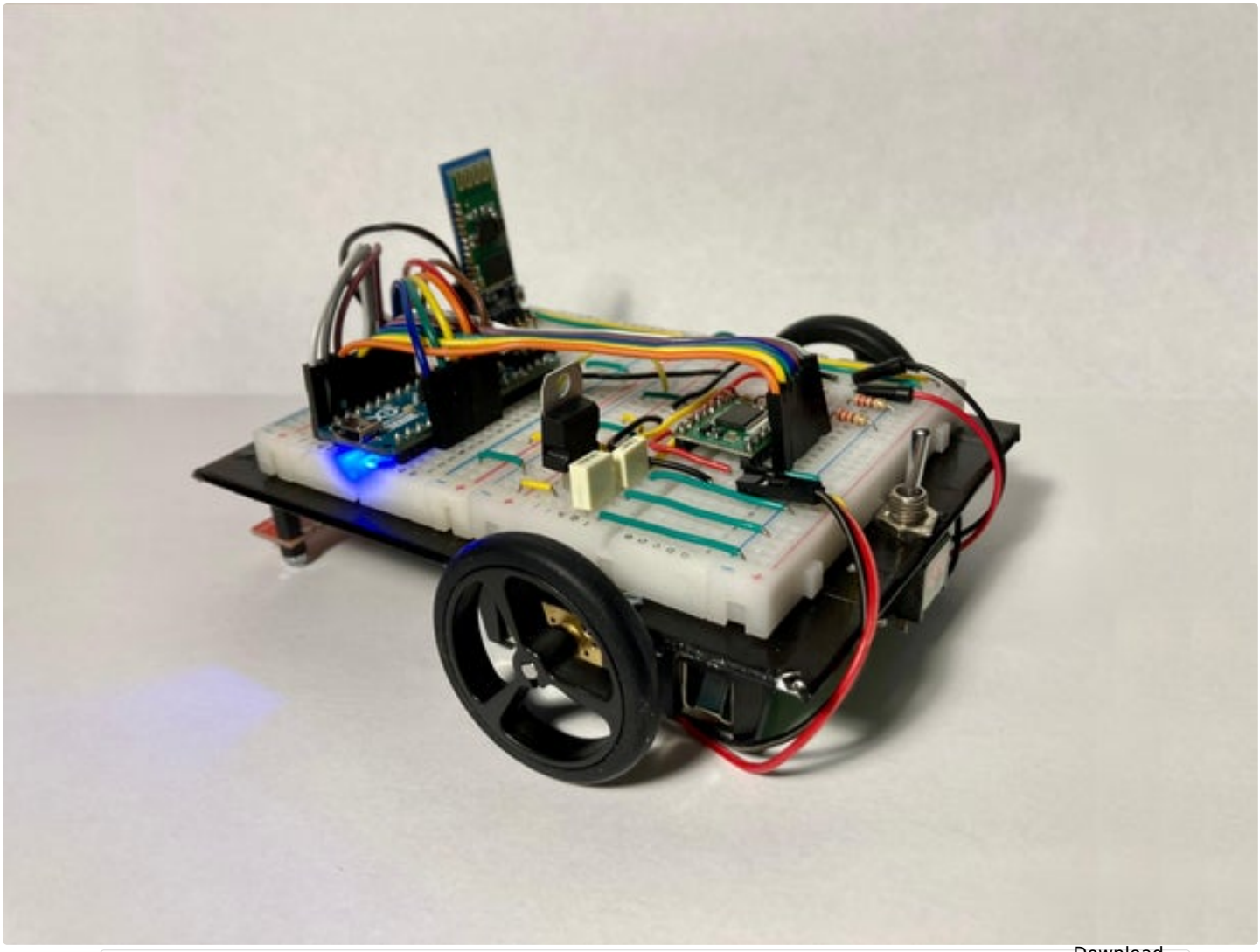
Supplies:

WARNING!

- Li-ion batteries can catch fire or explode if used incorrectly or in the case of a short circuit.
- Always connect the positive (+) and negative (-) terminals correctly.
- Never short-circuit the battery, and be careful not to accidentally short it, for example, with a keychain.
- Do not disassemble or deform the cell. If deformed, do not use it anymore.
- Do not immerse in water.
- Avoid extreme shocks or vibrations.
- Keep batteries away from fire.
- Keep batteries out of the reach of children.
- Do not use damaged chargers.
- Do not leave Li-ion batteries in a charger for more than 12 hours.
- Store the batteries in a dry and cool place.
- Dispose of empty batteries at a collection point.
- Never solder on battery cells.







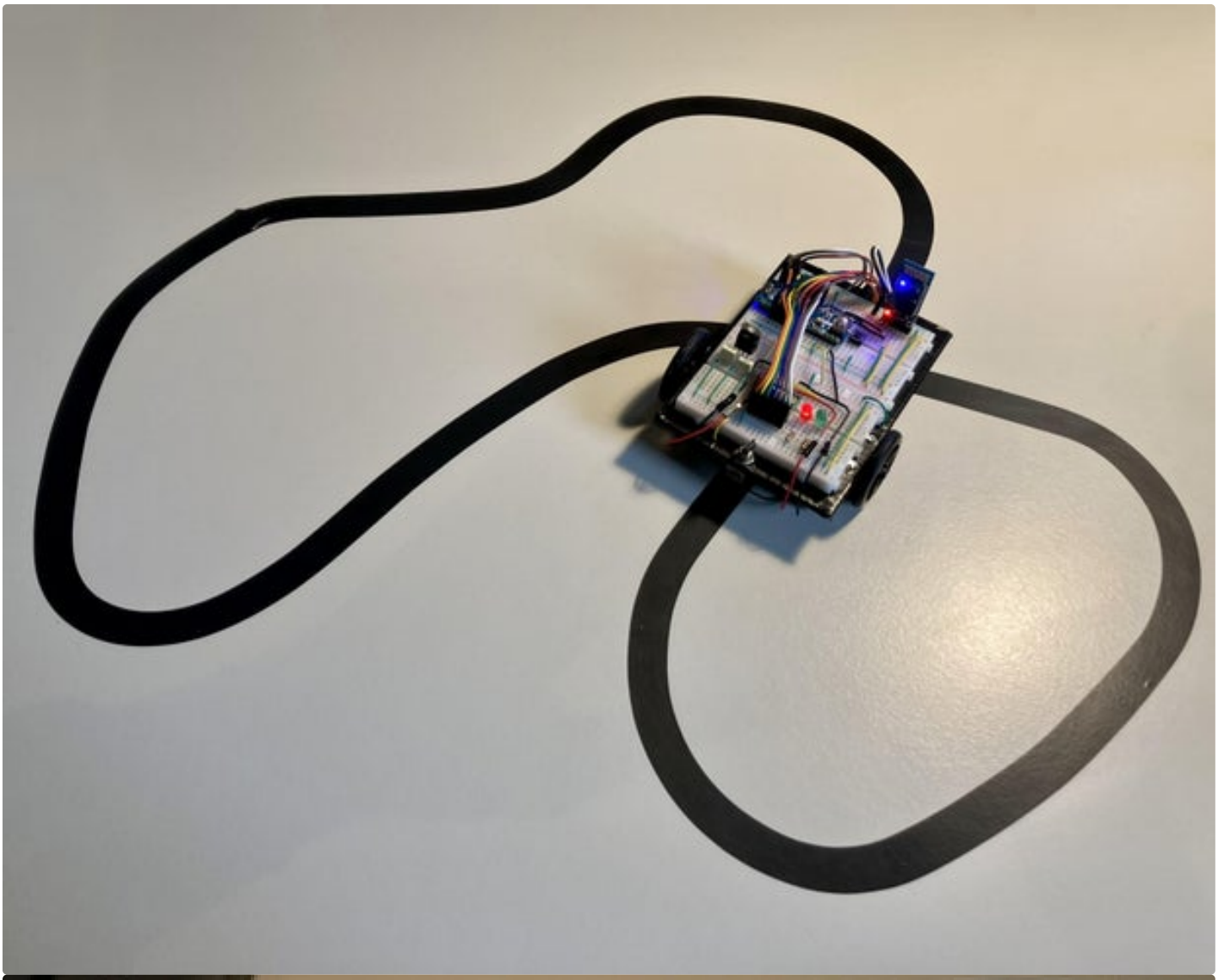
Download

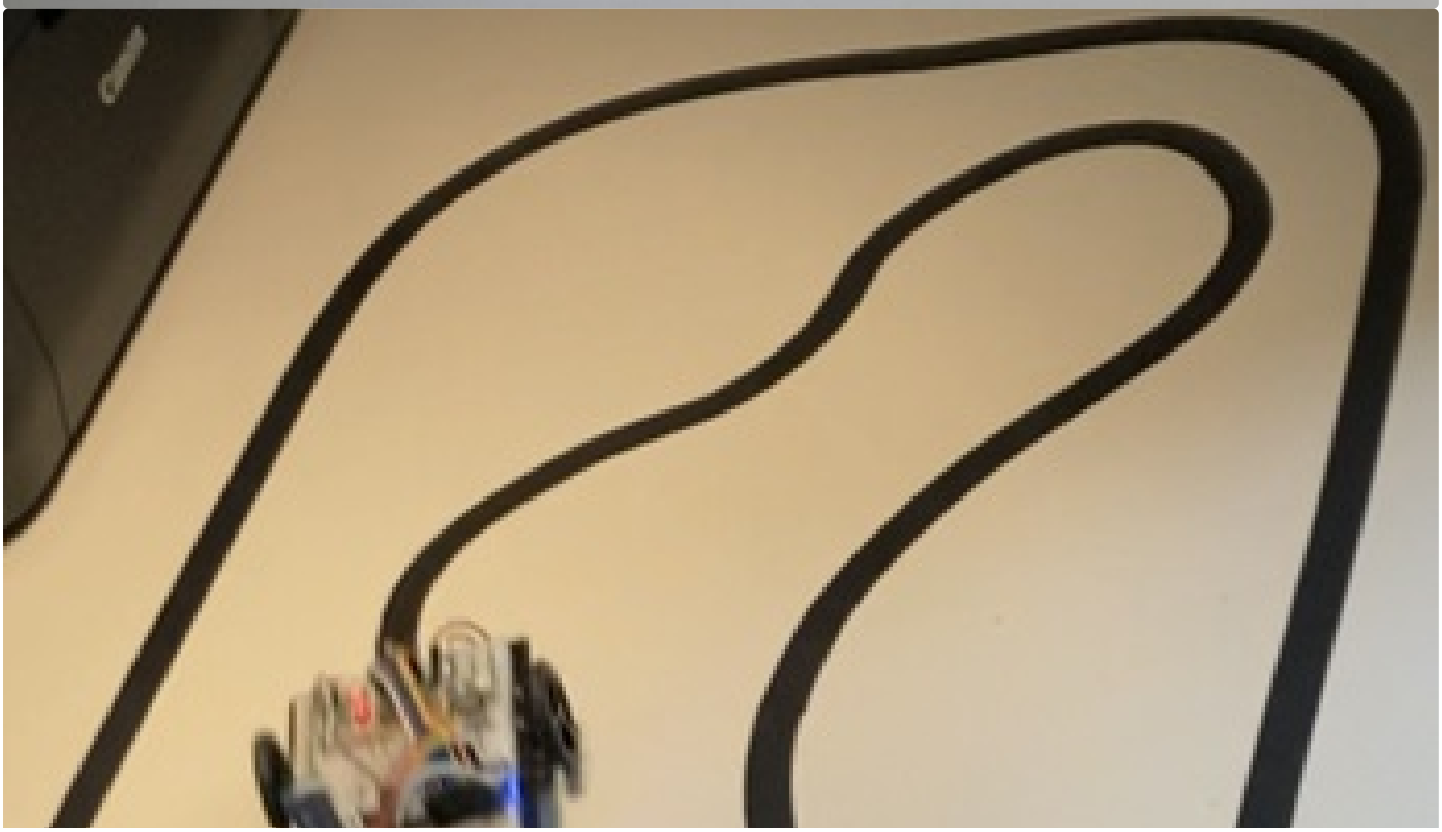
https://www.instructables.com/FG1/8CN7/LR0NCM3U/FG18CN7LR0NCM3U.pdf

Step 1: Track

The robot is designed to read a black line of a thickness of 15mm. Using regular A4 paper and a standard printer, you can print a black line of 15mm to create a track. Additionally, the robot should be able to continue forward when facing intersections. You can include those in your track design as well.

Other methods can be used to create a track to your liking. I have noticed that black electrically insulating tape is a perfect alternative. Since the tape has some elasticity, you can manipulate it to make clean corners by stretching. The pictures above show handmade tracks using only electrically insulating tape on a white desk. Beware, the contrast between black and white should be as high as possible for the robot to be able to detect the black line effectively. You can use all your creativity when designing your track! Enjoy!





Step 2: Chassis

Unfortunately, I cannot provide exact instructions on how to design your robot chassis. I highly recommend trying to build a chassis using your own creativity, as experimenting with different designs is remarkably easy and fun!

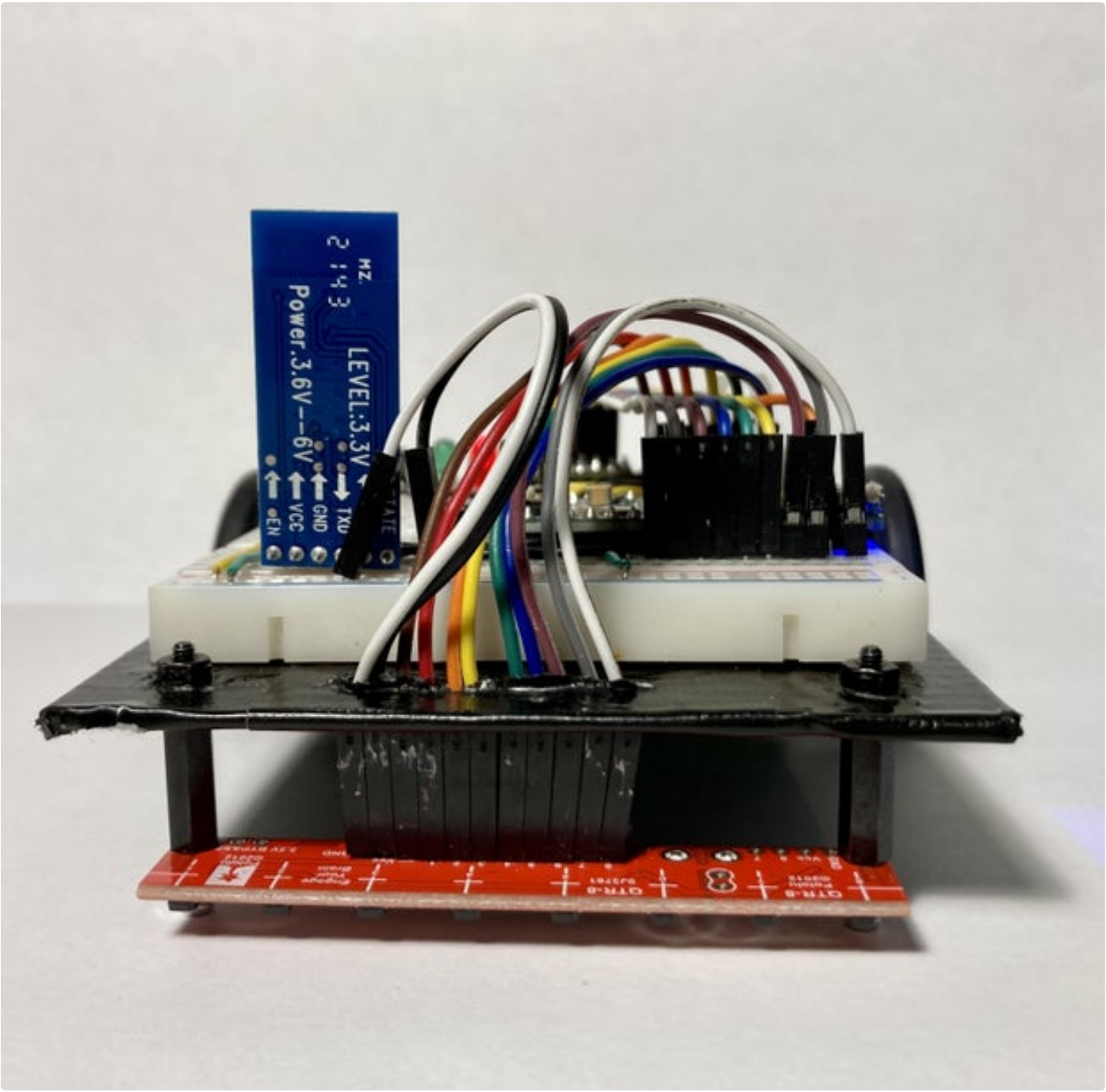
However, I will provide a list of tips that can help you create a more efficient design. In the end I will explain how I have built my own robot chassis, but again, I urge you to make something unique for yourself. If you take the following tips into regards to your design, it will be superior to the one I will show you. To get more inspiration, you can also search online for other implementations of Line Following Robots.

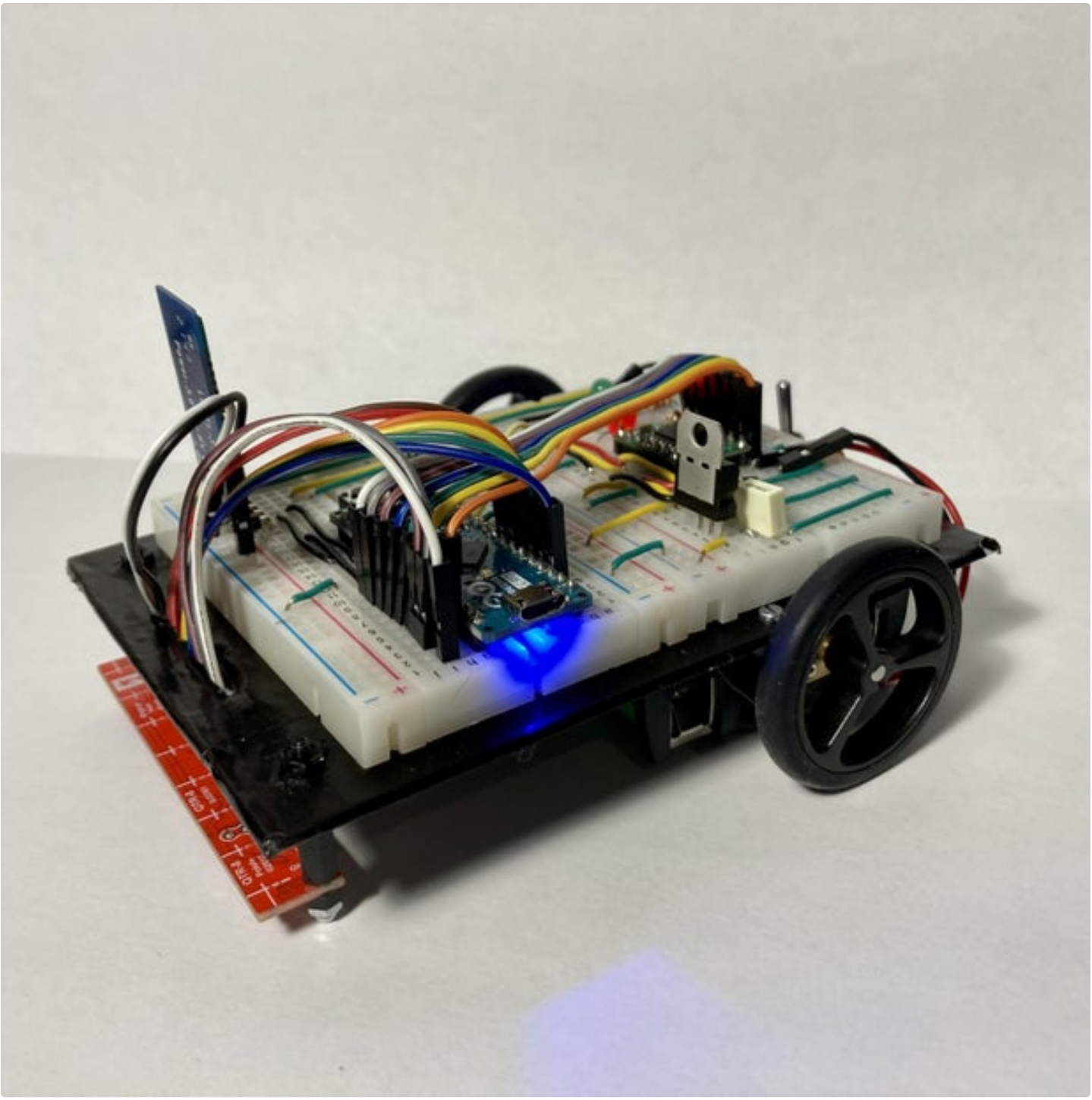
On making a successful chassis design you can take following things into account:

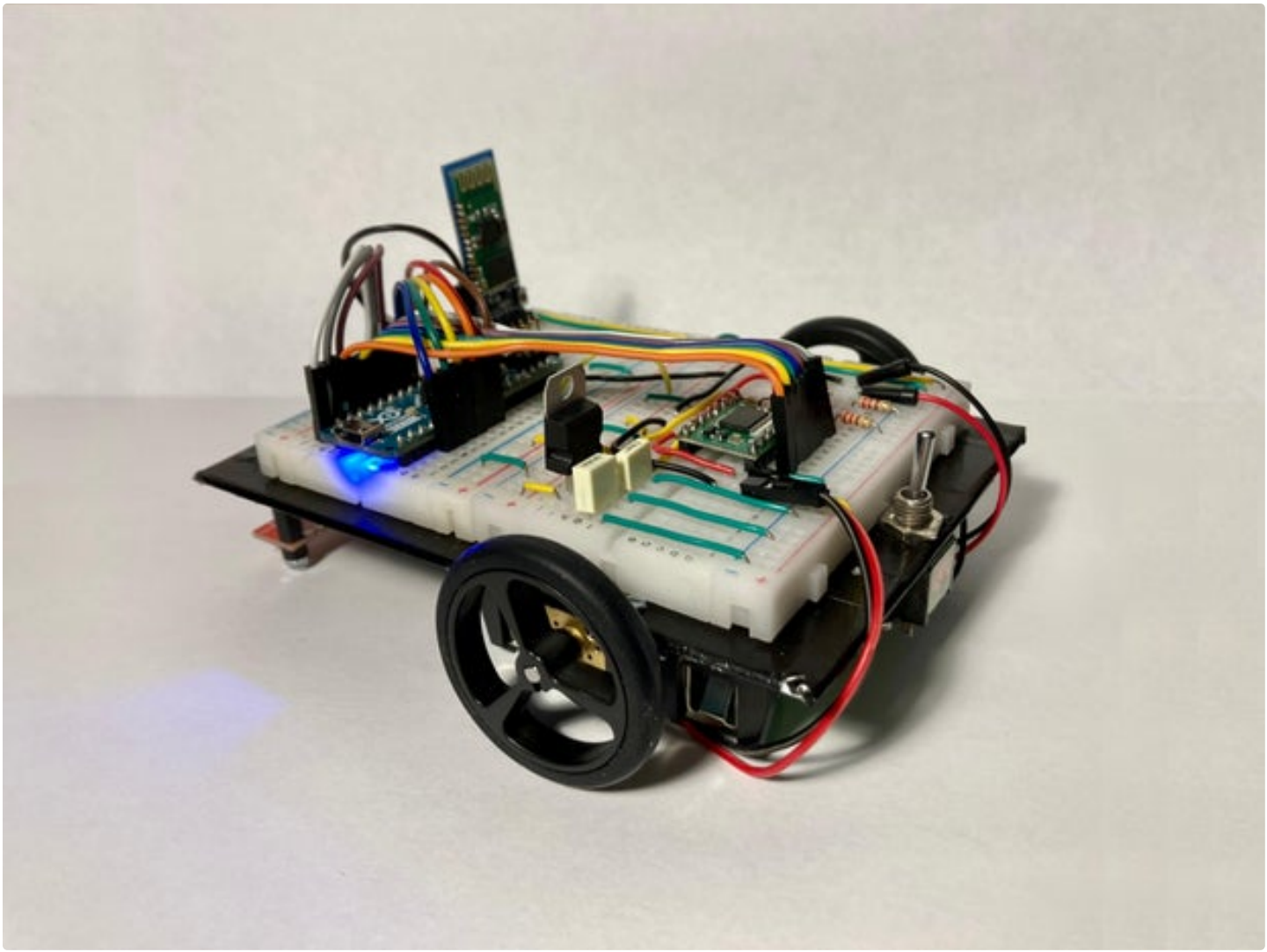
- Use non-flexible, light, electrically insulating materials: Acrylic, plastic, wood and 3D printer materials are all excellent options.
- If possible, use a 3D printer to make a custom chassis. This not only gives you the chance to make something unique, but also eliminates the use of drills, cutters and other tools that make a mess when using them.
- Make sure the center of mass is as low as possible: This prevents the robot from tilting when approaching a corner at high speed.
- Mount the heaviest components of your robot as close to the motor axles as possible: Use the mass of your components to your advantage! This allows maximum grip to the wheels, ensuring accurate position control.
- If you want the robot to be able to turn very tight corners, I recommend making the wheelbase as small as possible. The wheelbase of my robot is 85 mm. I find this sufficient for my purposes but you can go smaller as well.
- It is generally not recommended to have a large wheelbase, since this greatly impacts handling in corners in a negative way.
- Consider adapting your chassis design to fit wheels of multiple radii. You can increase robot top speed by increasing wheel radius.
- Do not use any tape, glue or other methods that permanently attach the robot components to the chassis! Use screw-bolt- or other non-permanent connections so you can undo changes when needed.
- If you want to challenge yourself, try to make your chassis design as compact as possible!
- Use breadboards only for prototyping! When the circuit is complete, use prototype boards or other methods to connect all your components more sustainably, as to prevent bad connections.
- Use a color-coding system of your wires to create structure in your circuit.

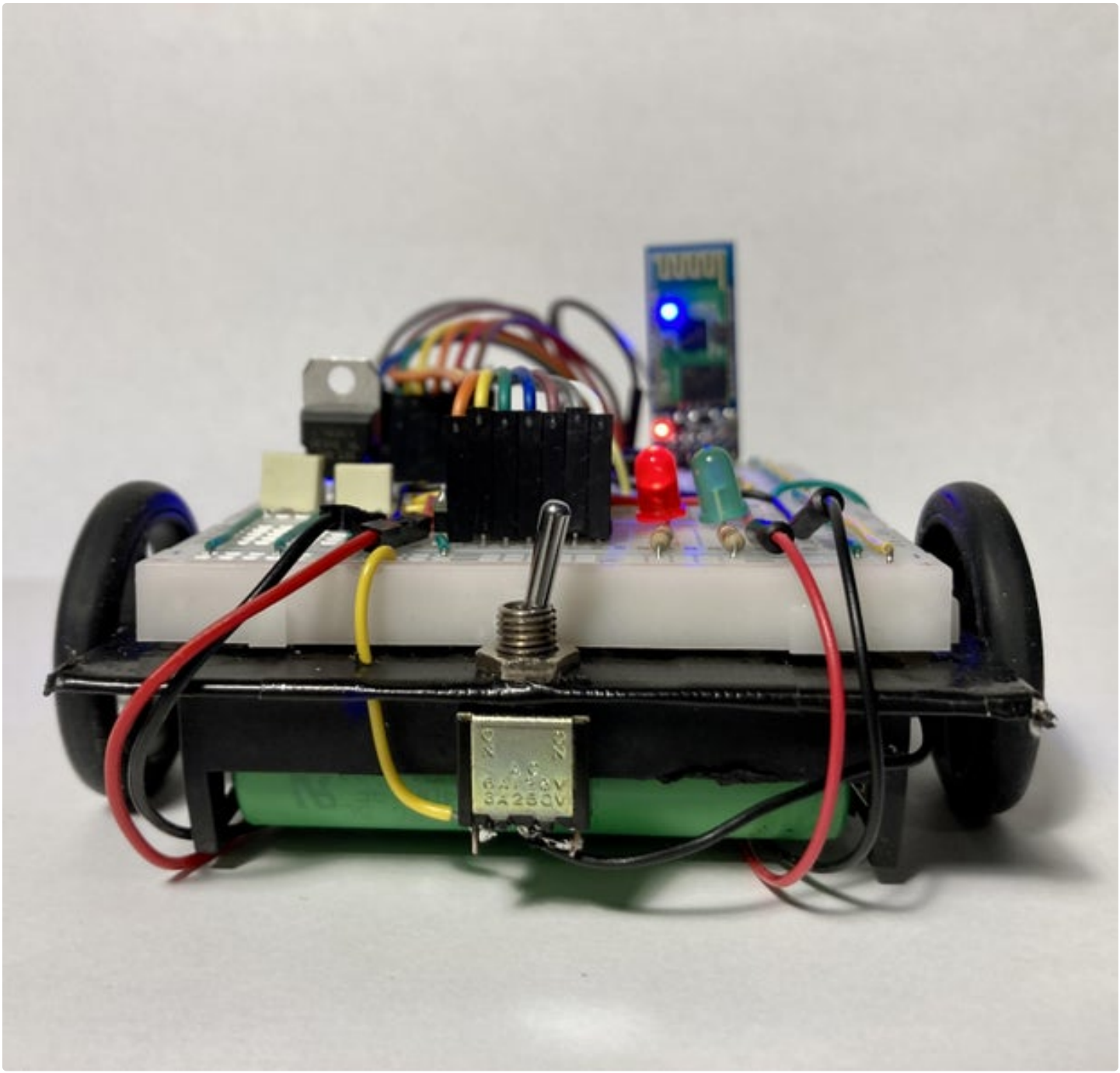
Above you can see pictures of my design of the Line Following Robot. As visible on the bottom view, the heaviest components (the batteries) are placed as close to the motor axles as possible to ensure a low center of gravity. Additionally, this also improved handling in corners.

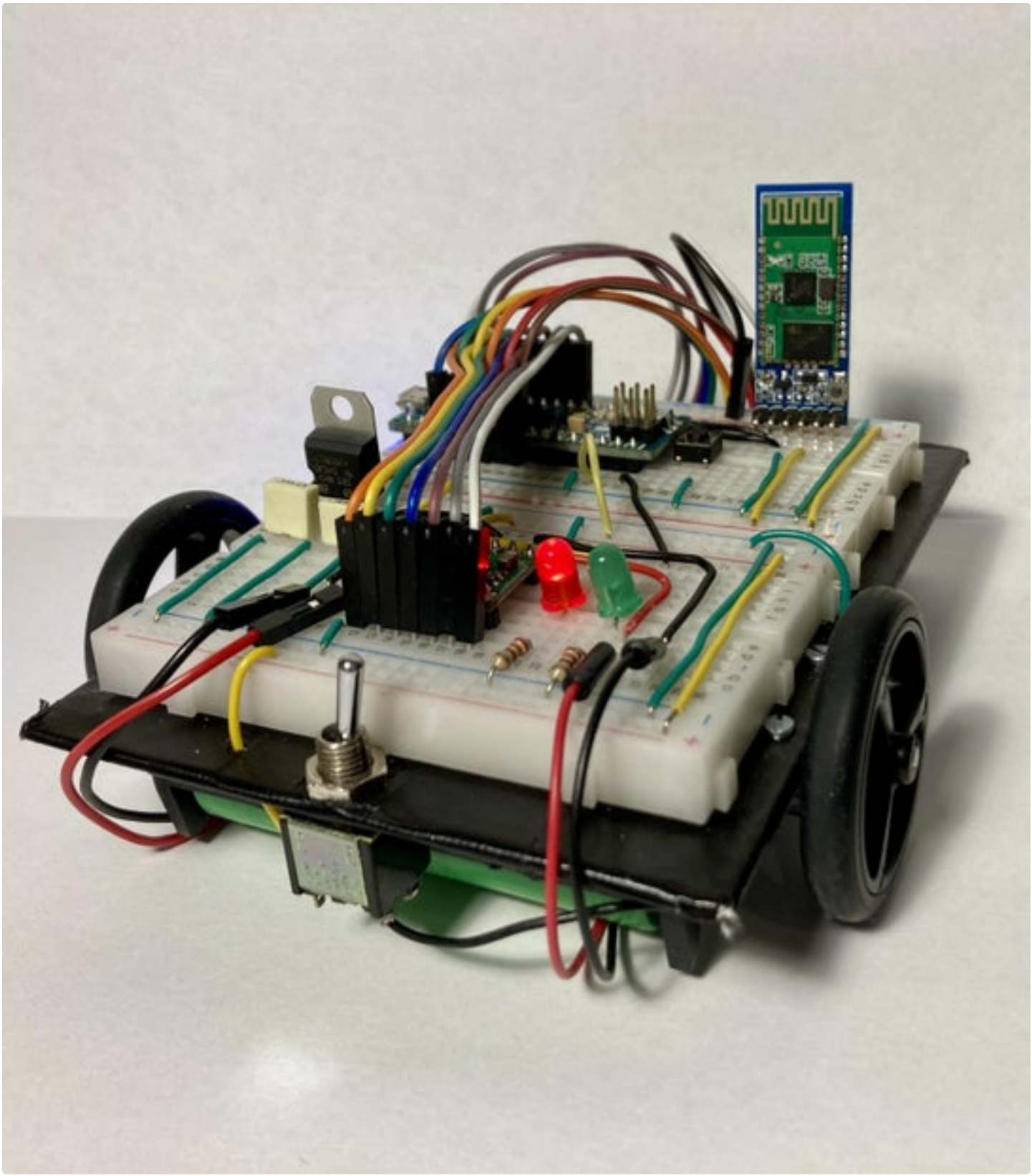
I have cut a plastic plate into a rectangle and attached all motor components to it. As visible in picture 4, the wheels seem to have a bit of camber. This is due to a poor plastic choice, which was too flexible to keep all components tidy in their place.

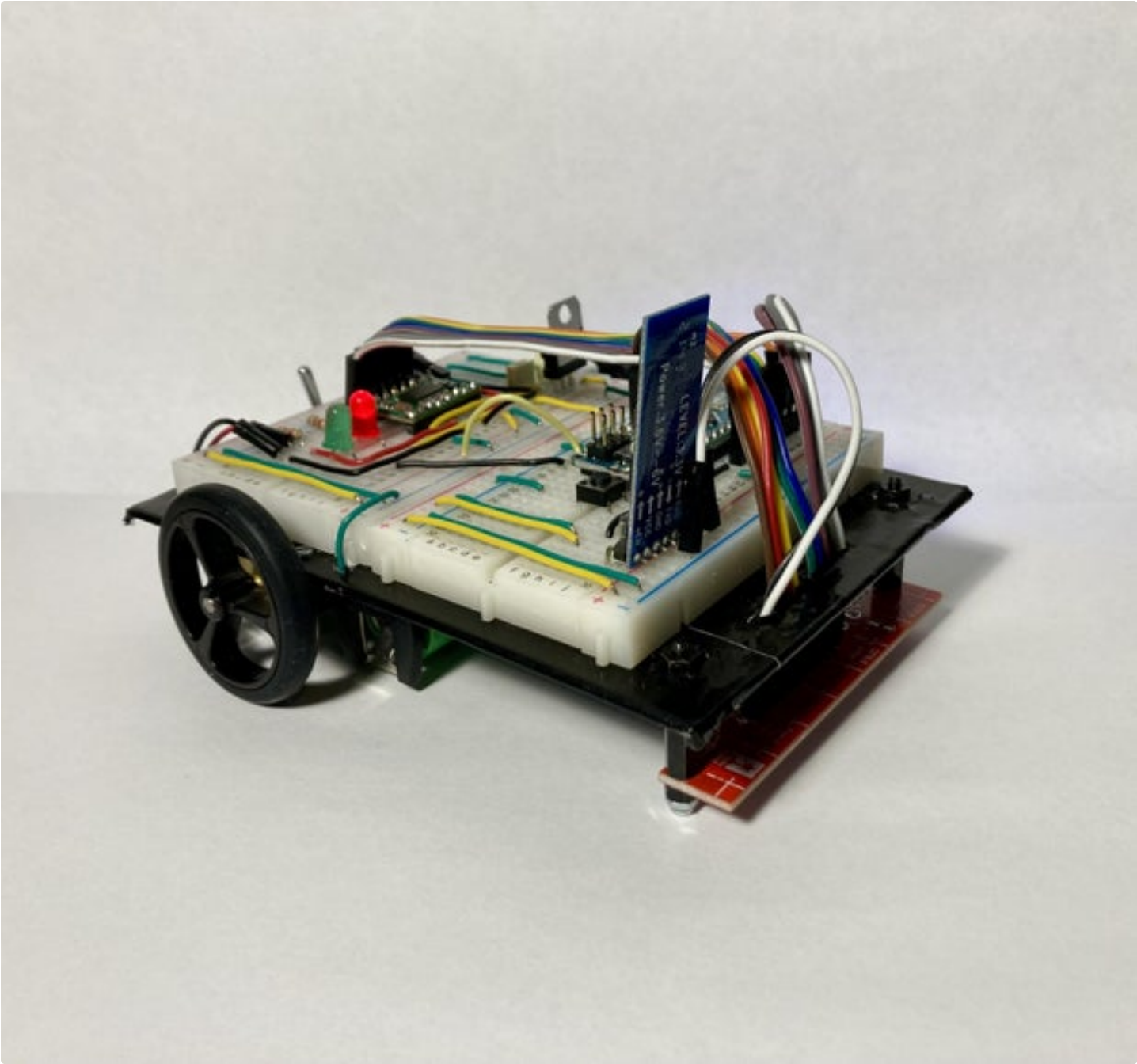

LEVEL:3.3V
Power.3.6V--6V
TXD
GND
VCC
EN
STATE

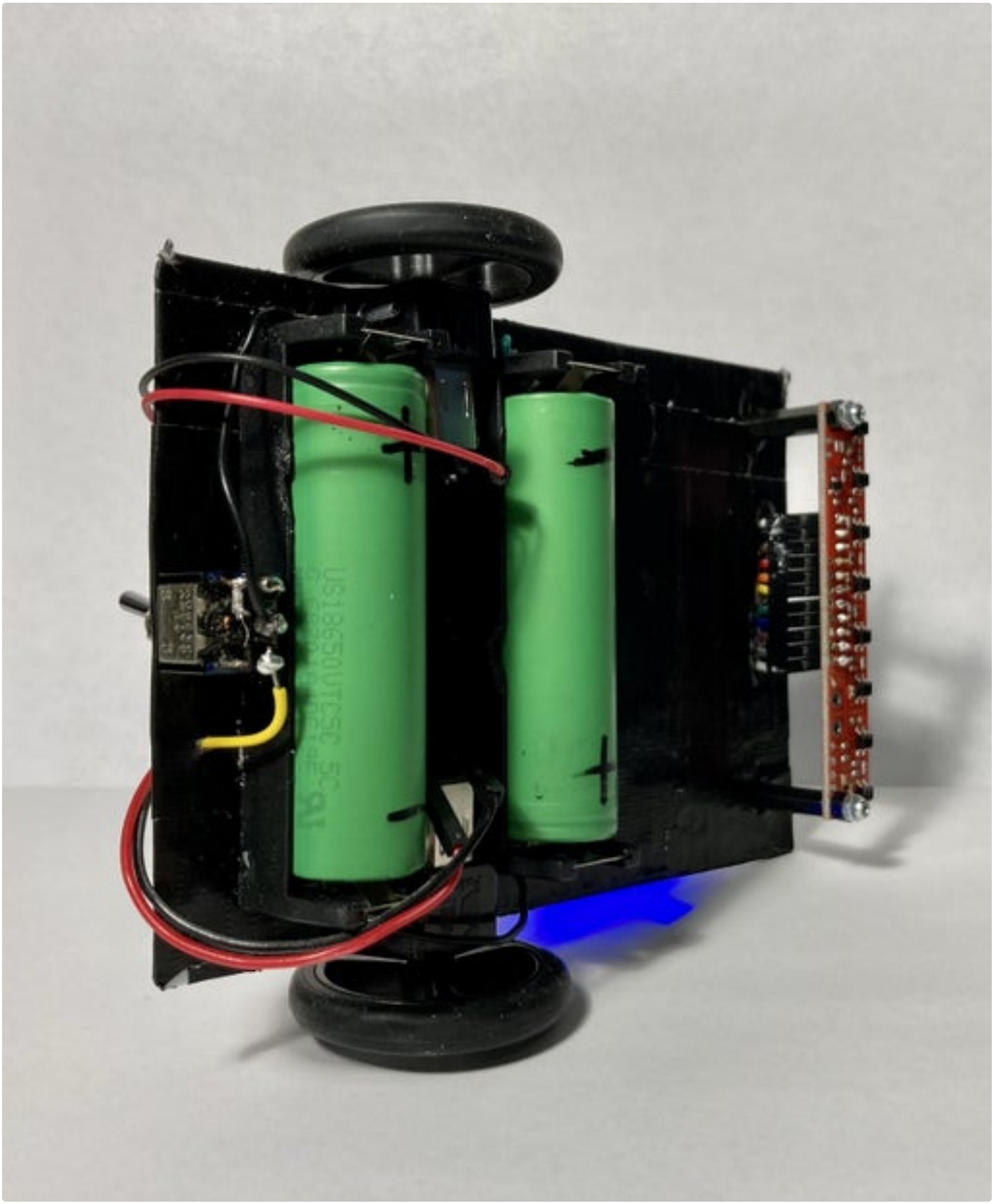


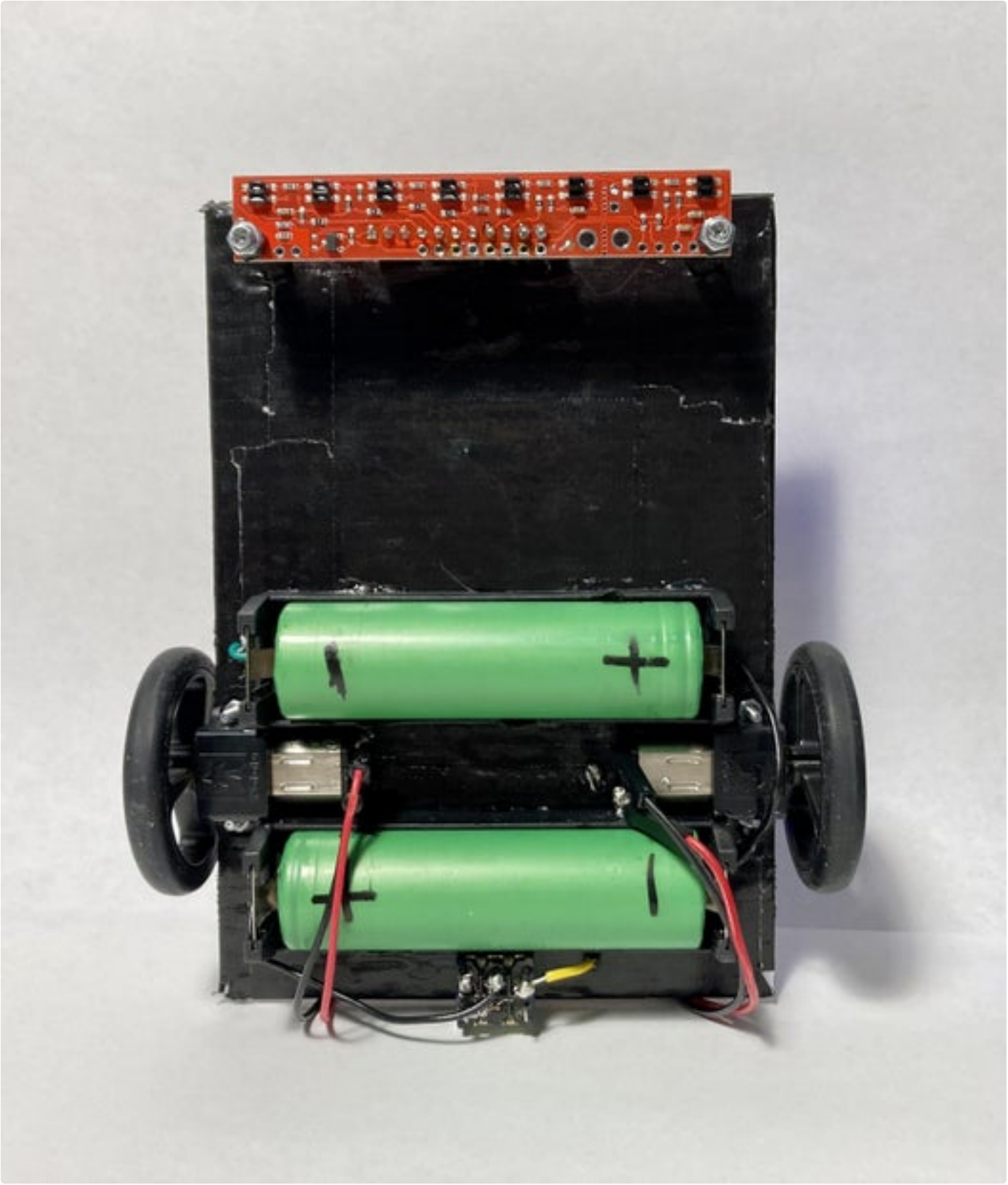










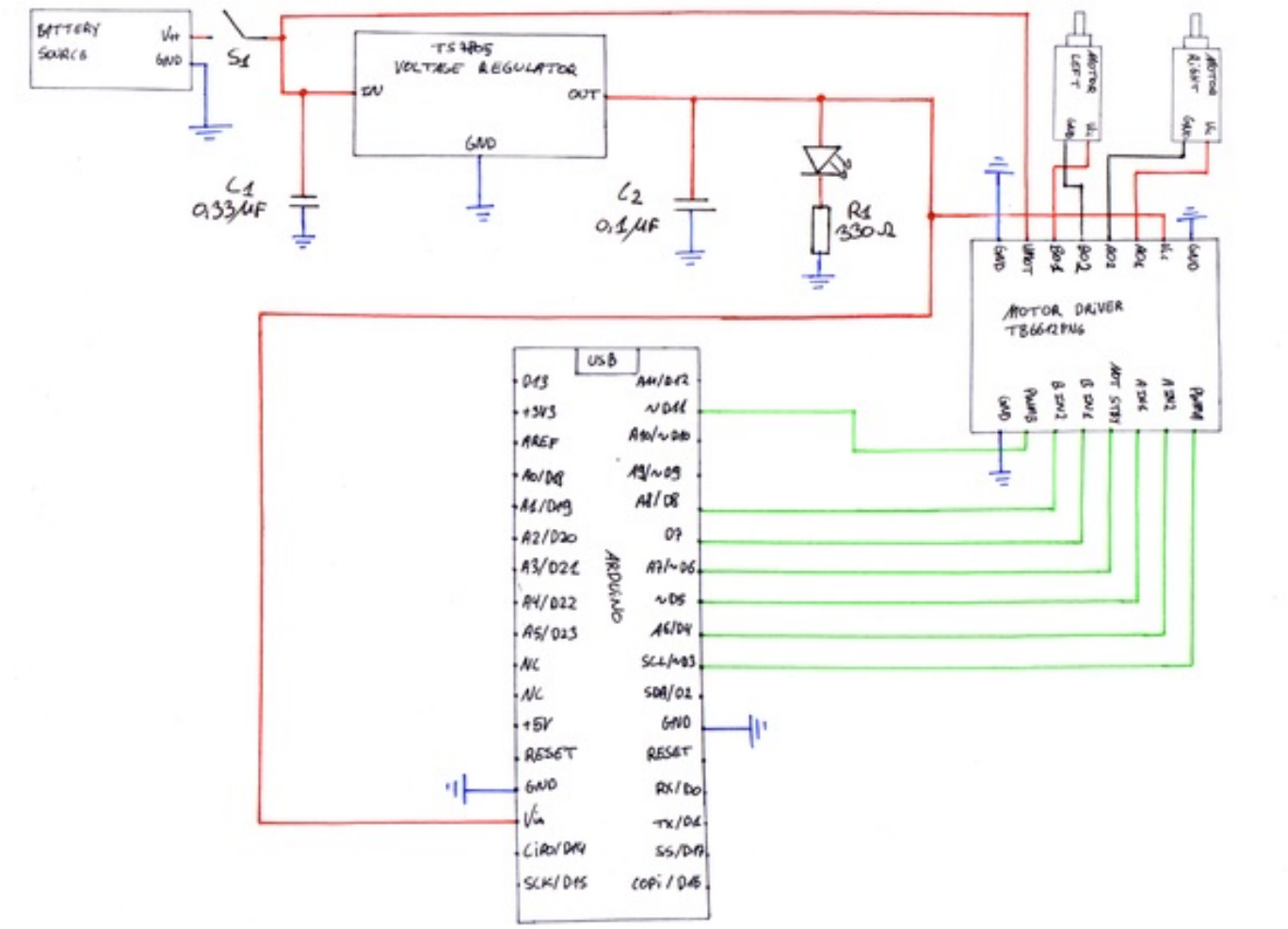


Step 3: Proof of Concept: TB6612NFG Motor Driver and DC Motors

Before coding the robot, I highly recommend creating "Proof of Concepts" to test all components if they function

correctly. A "Proof of Concept" is a simple piece of code that tests if all functions of a component (sensor, motor driver) are working as intended. This will simplify troubleshooting when your robot is making errors, since you know all components are working correctly.

To test if the TB6612FNG Motor Driver and the DC motors are functioning correctly, connect the components according to the scheme above. Next, upload the code provided below. The TB6612FNG needs PWM information to determine motor speed. In this code, using for-loops, motor Right will spin from 0 to maximum rpm clockwise, stop, and repeat the same process counterclockwise. Next, Motor Left will execute the exact cycle as motor Right. If your motors are working as described, your Motor Drivers and DC motors are confirmed to work correctly.

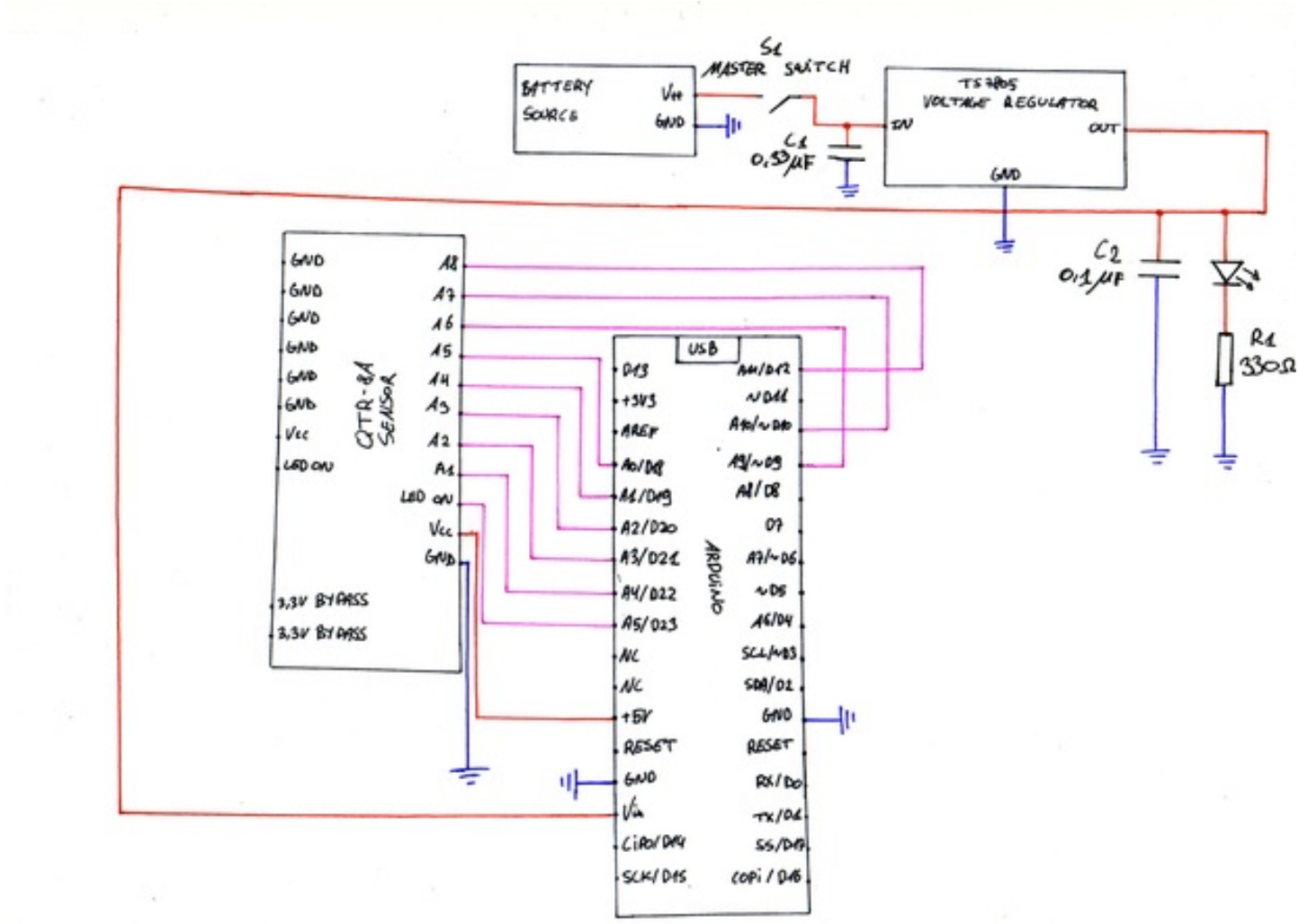


Download

https://www.instructables.com/FMM/1FQN/LR0NBY7H/FMM1FQNLR0NBY7H.ino

Step 4: Proof of Concept: QTR-8A Sensor

Since the QTR8A sensor is just an array of 8 analog sensors, a simple analogRead() will suffice to check correct functioning. In this code, all 8 sensors are read and displayed on the Serial Monitor. Exposing the sensor to light should display a value around 1000. Exposing the sensor to darkness or a black line, a value around 0 should be shown on the Serial Monitor.



Download

https://www.instructables.com/FS1/UE2Q/LR0NBYQZ/FS1UE2QLR0NBYQZ.ino

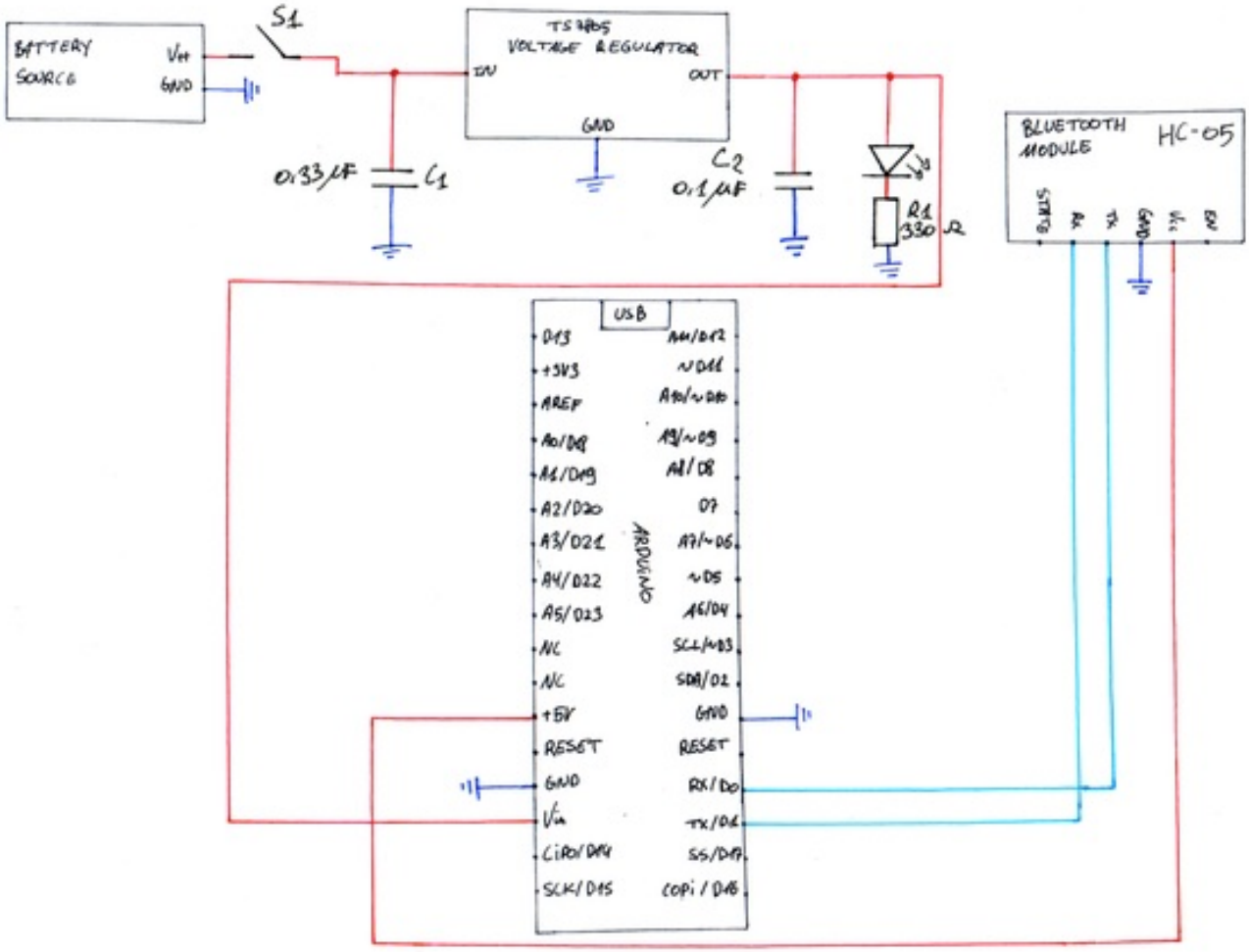
Step 5: Proof of Concept: HC-05

Although not necessarily useful for the robot to improve top speed or handling. Enabling wireless communication will benefit greatly when adjusting PID- and other parameters during testing. The tedious and time-consuming process of connecting your robot to your PC, updating and uploading code will be eliminated. Adjusting PID parameters will become much more efficient when using a HC-05 module and a Bluetooth app on PC or Android.

In this proof of concept, a basic serial communication channel will be created between the HC-05 and your Arduino. Communication will be possible in two directions (From Arduino to HC-05 and vice versa). As seen in the code provided below, 2 hardware serial ports will be used. The term Serial1 refers to the second hardware serial port available on the Arduino Micro. The Serial1 object will be used to communicate with the HC-05. Beware that your microcontroller must have at least 2 hardware serial ports for this to function correctly!

For this code to work, the baudrate of the HC-05 needs to be set to 115200 bps. More information on how to change baudrate and HC-05 name can be found on this Instructable.

The HC-05 functions as intended when you send a piece of information (text or numbers) from the Arduino to the HC-05 using the Serial Monitor. When sent, the text should arrive on the Serial Bluetooth app of your smartphone or PC. This should work the same vice versa: Sending information from the Bluetooth app should arrive on the Serial Monitor.



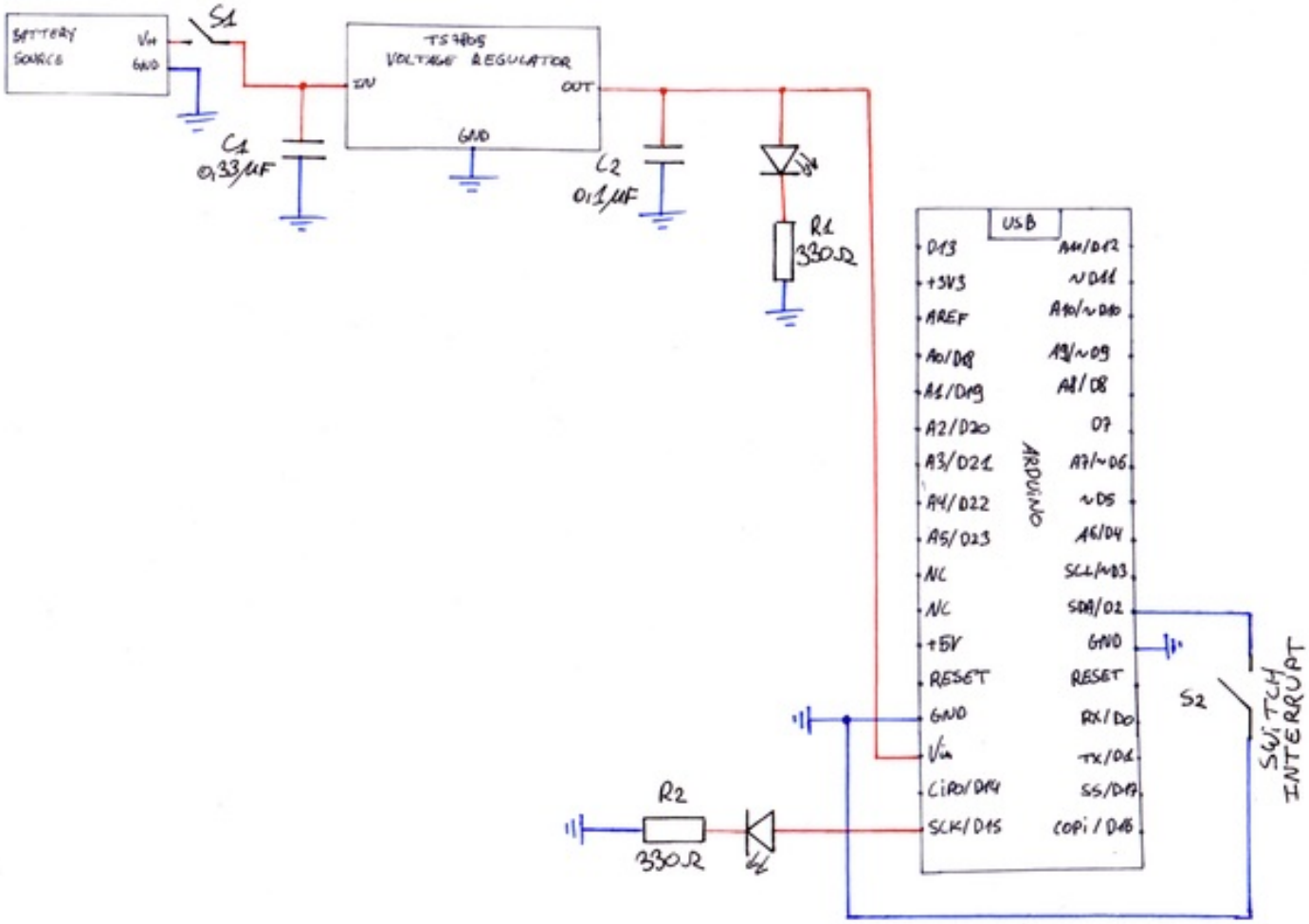
Download

https://www.instructables.com/FE6/TI5Z/LR0NBYY9/FE6TI5ZLR0NBYY9.ino

Step 6: Proof of Concept: Switch Interrupt

Starting and stopping the robot also is made possible with a physical monostable switch. Cyclically polling this switch is not a good idea. First, this can disrupt the timing of the rest of the program. Second, you risk missing pulses. Attaching the switch to an external interrupt of the microcontroller and monitoring it hardware-wise is much more convenient. The switch is debounced using software.

Using an ISR, the button is debounced and a function GreenLed() is called. The proof of concept works correctly if a green LED turns on when pressing the pushbutton. If the button is pressed again, the LED will turn off.

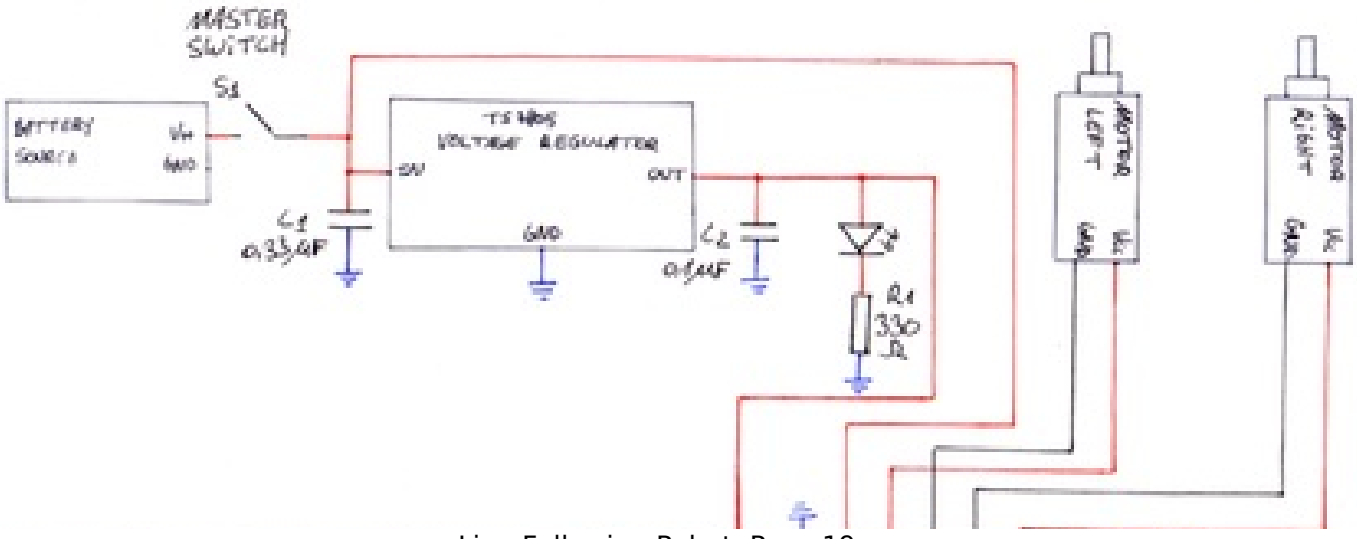


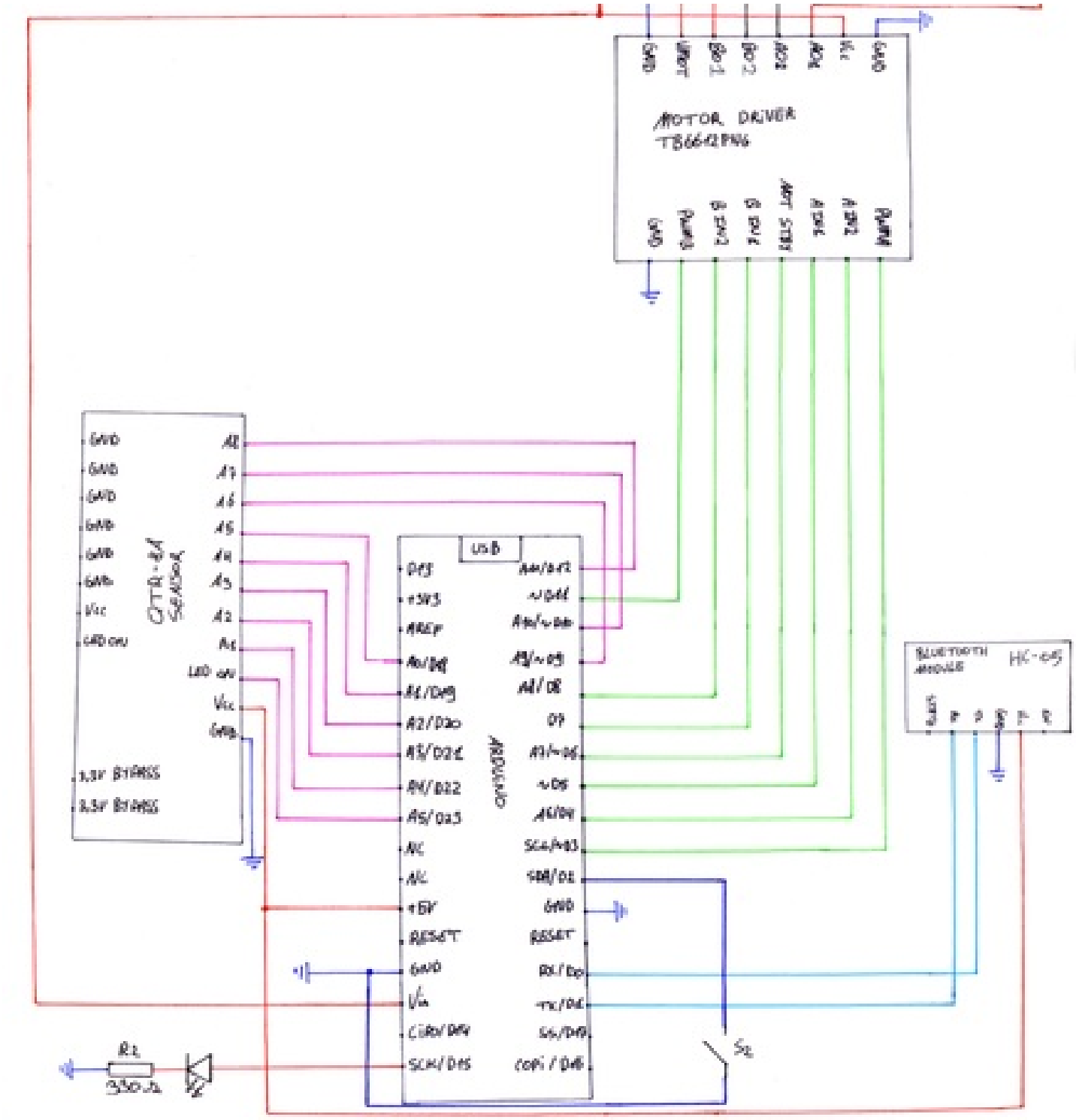
Download

https://www.instructables.com/FR0/L7WH/LR0NBZMV/FR0L7WHLR0NBZMV.ino

Step 7: Electrical Scheme

The final electrical scheme consist of a summation of all Proof of Concepts listed before. This is all you need to get the robot to work.





Step 8: PID Coding

After successfully completing all proof on concepts and finishing the electrical scheme, the next step is to upload the Arduino code. I have provided an additional manual on how to operate the robot, adjust PID- and other parameters down below. I recommend reading this manual before continuing to the next chapter "Testing".

This code uses following libraries:

- EEPROMAnything.h: To write data in EEPROM Memory.

- SerialCommand.cpp: Enables ability to read incoming data and execute it as instructions.
- SerialCommand.h: Enables ability to read incoming data and execute it as instructions.

In the main loop, all sensor values get normalized and stored into 1 variable “dalpunt”. Variable “dalpunt” will be the input of the PID controller. Next, PID formulas determine the output. Output information is used to send PWM information to the Motor Driver.

Additionally, other functions are added:

- onSet(): Function to read instructions and store them in EEPROM memory.
- onStart() and onStop(): Functions to start and stop the robot.
- onDebug(): Function to display all parameters of the robot on the Bluetooth Serial Monitor.
- onCalibrate(): Function that's responsible for storing calibration values in EEPROM memory.
- onUnknownCommand(): Sends error message to Serial Monitor.

Download
https://www.instructables.com/FEM/604U/LR0NC0WW/FEM604ULR0NC0WW.ino

Download
https://www.instructables.com/FOJ/5MBW/LR0NCLL9/FOJ5MBWLR0NCLL9.h

Download
https://www.instructables.com/F06/WS1Y/LR0NCLOO/F06WS1YLR0NCLOO.cpp

Download
https://www.instructables.com/FOA/865H/LR0NCM5O/FOA865HLR0NCM5O.pdf

Step 9: Testing

The robot should be able to complete a full lap of your designed track using only following parameters: Power, Kp and Diff. I would recommend starting off easy with following parameter values:

- Power -> 50
- Kp -> 10
- Diff -> 0,5

If the robot is successful in completing multiple rounds without stopping, you can continue to increase Power and Kp. Have fun in this phase! I encourage you to mess around with all parameters and see what happens! For example, give Kp an extreme value of 150 and observe how the robot behaves.

After some time, you probably will start to realize the great addition of the HC-05 Bluetooth module. Adapting parameters becomes much less of a chore compared to the traditional wired connection with your PC.

At some point you will notice controlling the robot with only a P controller will run into its limitations. Increasing Power and P parameter only will get you so far. You can now start experimenting with I and D values. In my own experience, I have noticed a PD controller will work substantially better than only a P controller.

Step 10: Conclusion

Thank you for reading my Instructable! I hope you had fun creating your own Line Following Robot! I would like to thank the University of HOGENT in Ghent, Belgium for their cooperation to make this project into reality.