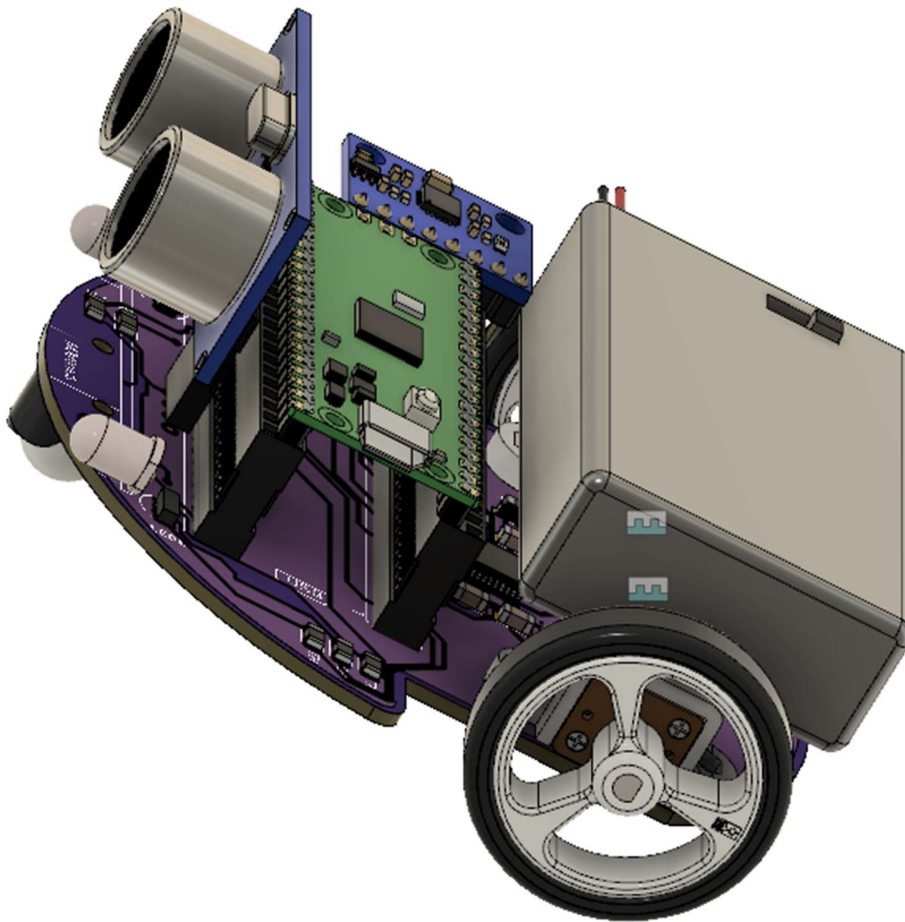




# MJtroniks Mojobot Pico User Manual

---



## Introduction

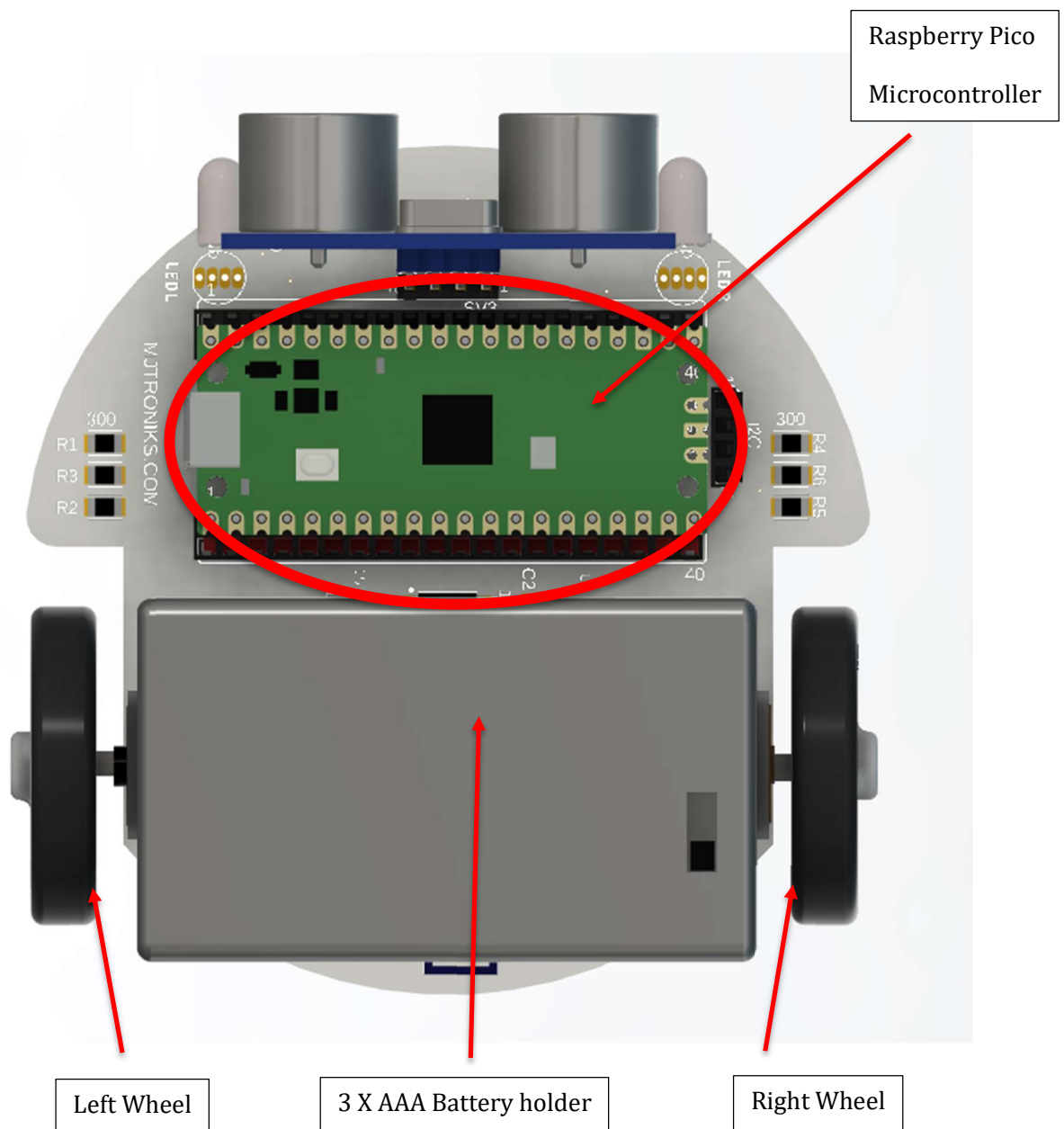
Mojobot by MJtronics is an educational and hobbyist robot, designed to facilitate learning in robotics, sensor integration, and programming.

## Robot parts

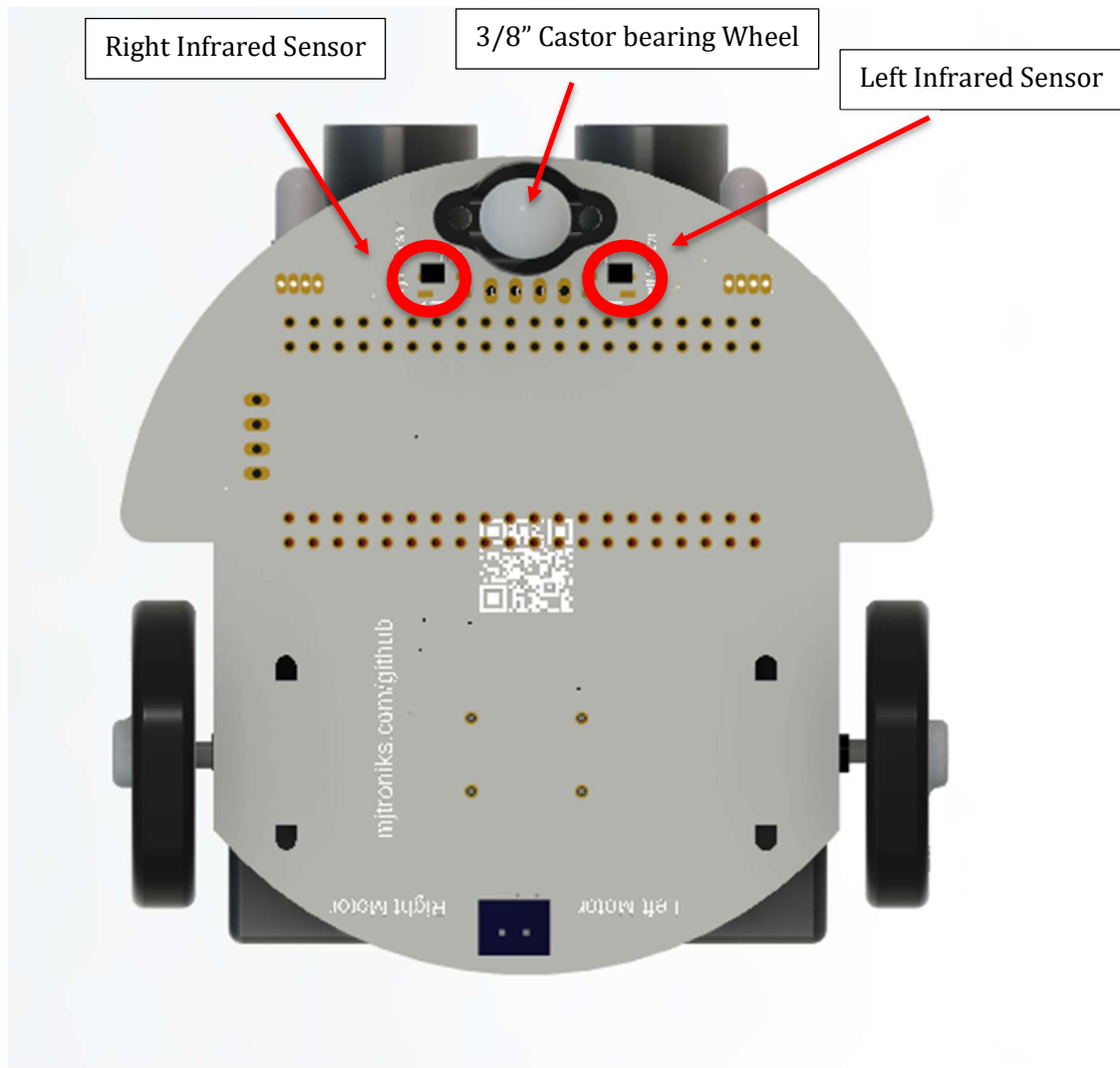
FRONT VIEW



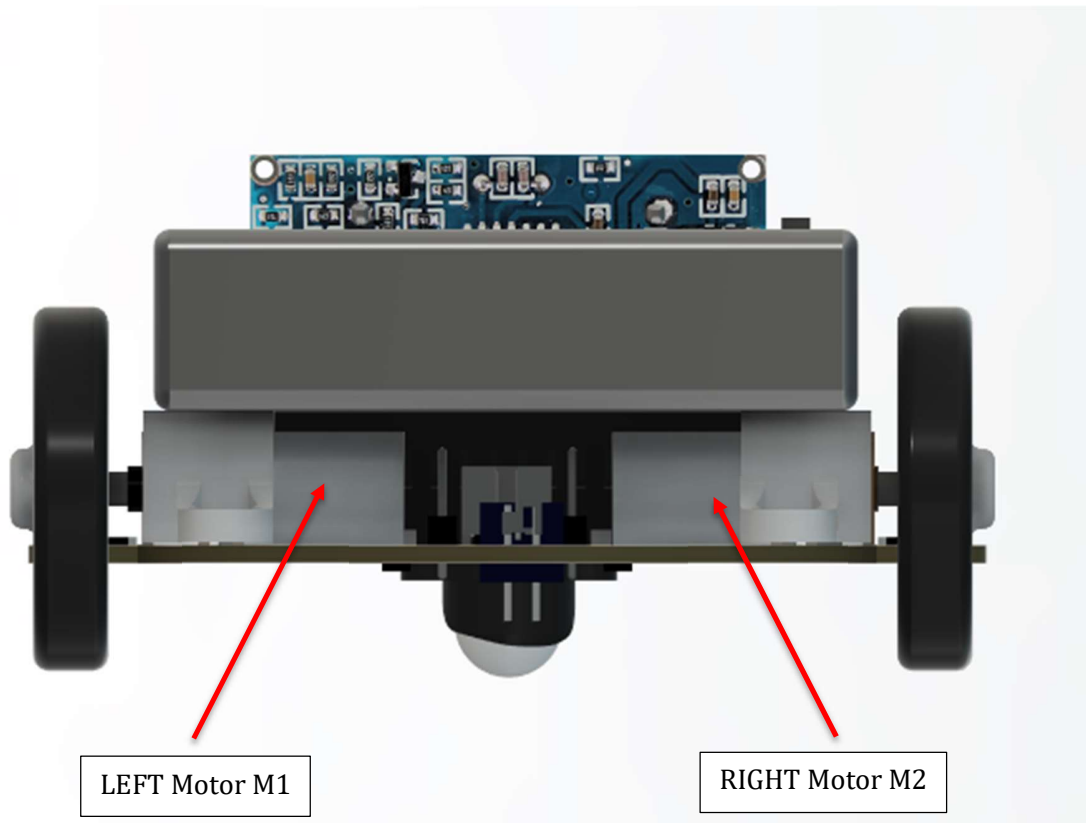
## TOP VIEW



## BOTTOM VIEW

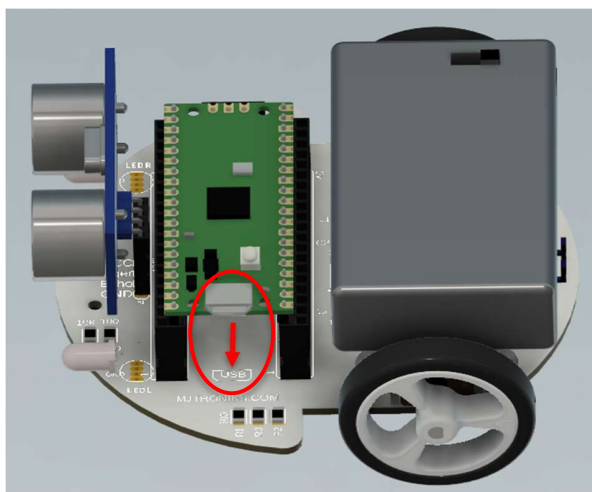


## BACK VIEW



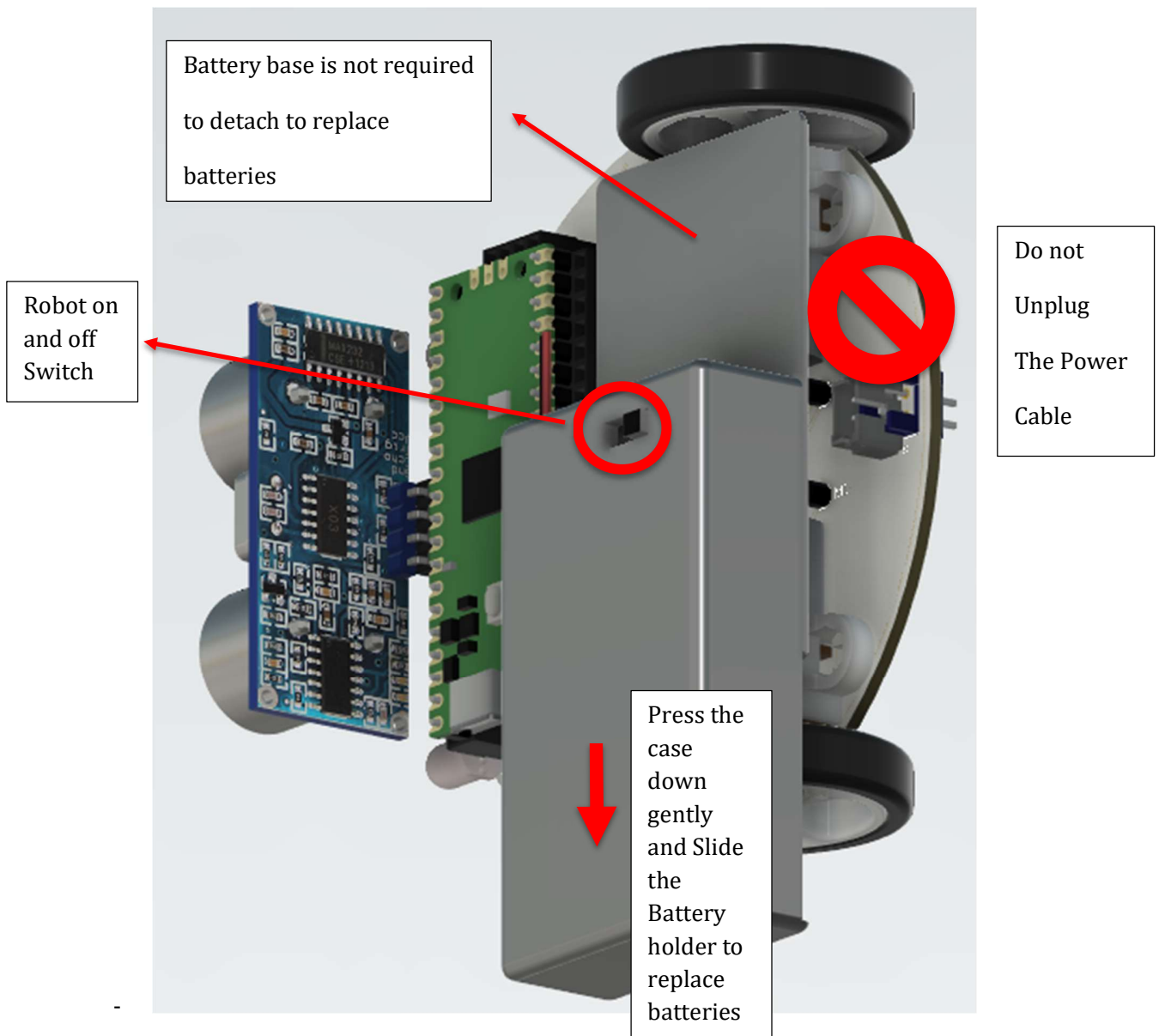
## Handling Instructions

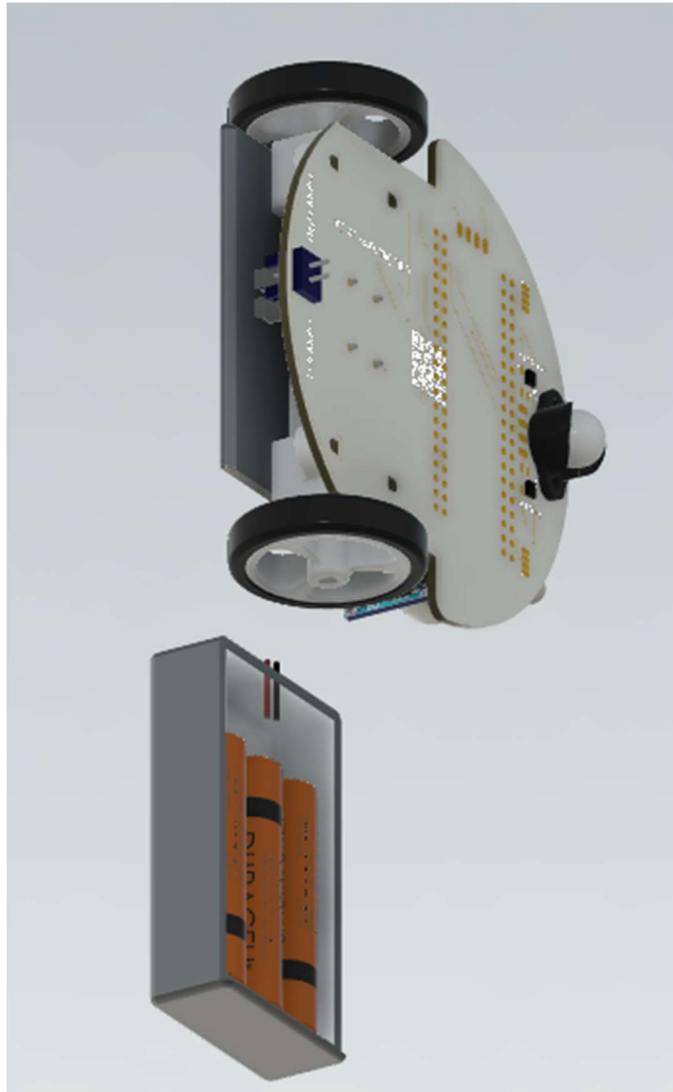
1. Microcontroller USB Alignment: Ensure the microcontroller USB matches the PCB board USB mark.



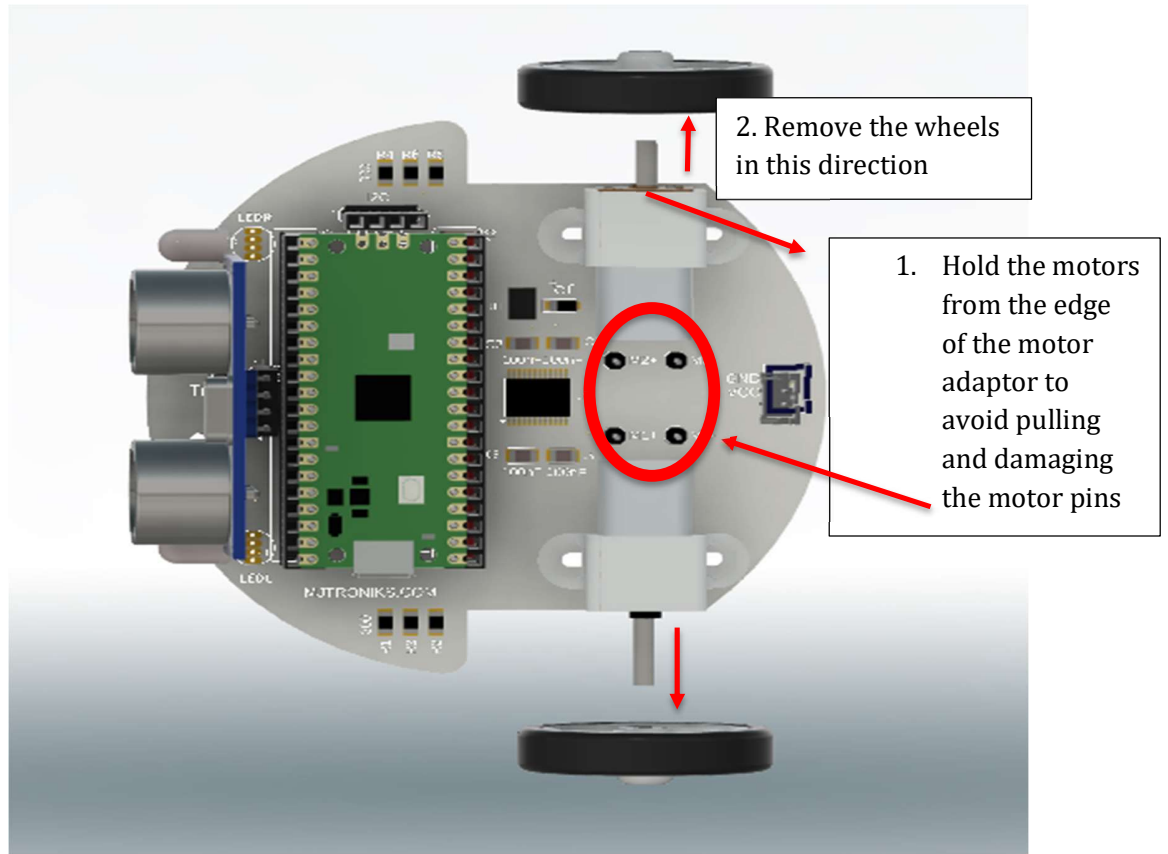
2. Battery Pack:

- Do not unplug the battery pack from the power connector.
- Remove batteries by sliding open the battery holder case.
- Use the battery holder case switch to turn the robot on and off.





3. Wheel Removal: Remove the battery holder and the proceed to remove the wheels gently, holding the motors to avoid damaging motor pins.



## Assembly and Components

- Components Included:

- 1 x Mojobot car
- 1 x Battery Holder
- 1 x HC-SR04 Ultrasonic Sensor
- 1 x Raspberry Pi Pico
- 1 x Robot stand
- 1 x Line following map



## Pins description

### - Motors:

- Motor 1 PWM (GP10) (motor 1 speed), Direction (GP12) Forward/Backward
- Motor 2 PWM (GP11) (motor 2 speed), Direction (GP13) Forward/Backward

### - Ultrasonic Sensor:

- Trigger (GP14), Echo (GP15)

### - RGB Headlights:

- Left LED (GP22, GP21, GP20)
- Right LED (GP7, GP8, GP9)
- Infrared Control:
  - Left Sensor (GP2), Right Sensor (GP3)

### - I2C:

- SCL (GP19), SDA (GP18)

## Power Requirements

- Voltage: 3.5V to 4.5V
- Batteries: 3 X AAA batteries (ensure correct placement)

## Software Setup Overview

Detailed setup instructions in <https://github.com/mjtroniks/Mojobot/wiki>

### 1. Micropython:

- Install Thonny IDE from [thonny.org](https://thonny.org)
- Download Micropython firmware from [micropython.org/download/](https://micropython.org/download/)
- Follow the Thonny installation guide to flash the firmware onto the Raspberry Pi Pico.
- Write and execute your Micropython code within Thonny IDE.

### 2. MakeCode:

- Visit [makecode.microbit.org](https://makecode.microbit.org)
- Create a new project and select the device as Raspberry Pi Pico.

- Use the drag-and-drop blocks to create your program.
- Download the generated UF2 file and drag it to the Pico mounted as a USB drive.

### 3. C++:

- Install Visual Studio Code and CMake.
- Set up the Pico SDK and toolchain by following the official Raspberry Pi Pico C++ setup guide.
- Write your C++ code and use CMake to build the project.
- Flash the binary onto the Pico using Picoprobe or drag-and-drop method.

## Programming Tutorials

### [Micropython Introduction and Blink \(Click for detailed instructions\)](#)

To begin with Micropython, you'll need to install the necessary software and set up your environment. Here's a simple blink program to get you started:

```
import machine
import utime

led = machine.Pin(25, machine.Pin.OUT)

while True:
    led.value(1)
    utime.sleep(1)
    led.value(0)
    utime.sleep(1)
```

### [MakeCode Introduction and Blink \(Click for detailed instructions\)](#)

MakeCode is a user-friendly platform for beginners. Here's how to create a blink program:

1. Go to [makecode.microbit.org](https://makecode.microbit.org).
2. Create a new project.
3. Drag the `show leds` block and create a pattern.
4. Use `pause` to control the blinking interval.

## [C++ Introduction and Blink](#) (Click for detailed instructions)

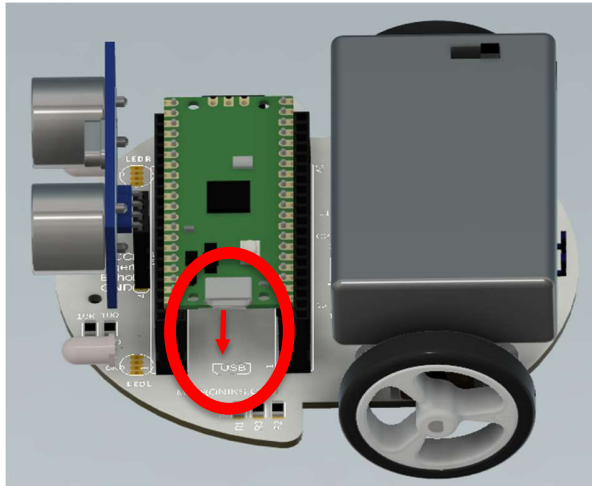
For C++, set up your Arduino development environment and write your first program to blink an LED:

```
#include "pico/stdlib.h"
```

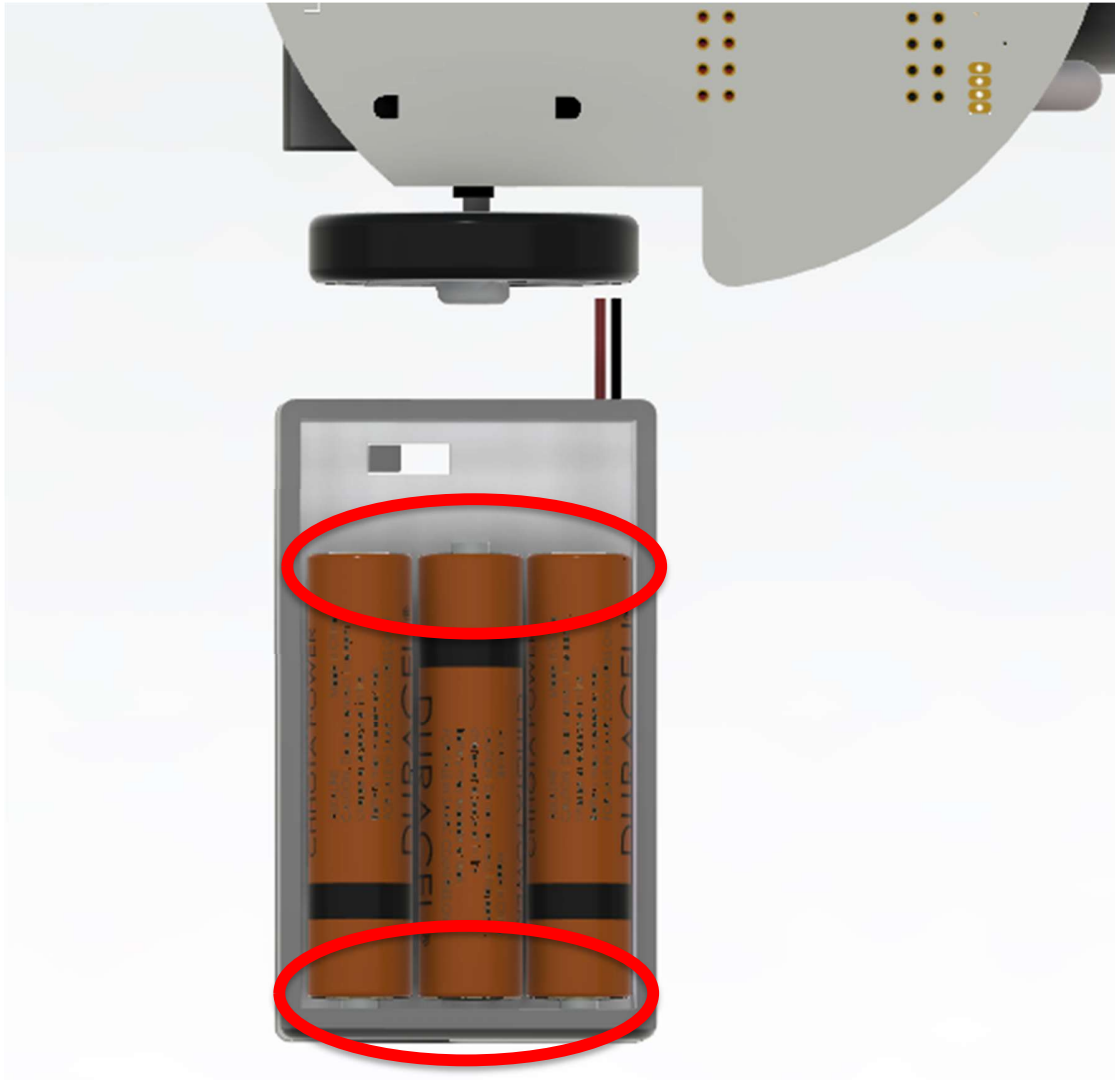
```
int main() {  
    const uint LED_PIN = 25;  
    gpio_init(LED_PIN);  
    gpio_set_dir(LED_PIN, GPIO_OUT);  
  
    while (true) {  
        gpio_put(LED_PIN, 1);  
        sleep_ms(1000);  
        gpio_put(LED_PIN, 0);  
        sleep_ms(1000);  
    }  
    return 0;  
}
```

## Troubleshooting

1. Microcontroller Placement: Check proper placement of the microcontroller.



2. Battery Polarity: Ensure batteries are correctly placed with proper polarity.



3. When switching programming languages the UF2 image fails to upload, to resolve Re upload the image using Thony IDE as described in [this tutorial](#) section Steps to resolve

For further details and images, refer to the MJtronics Mojobot GitHub Repository (<https://github.com/mjtroniks/Mojobot>).