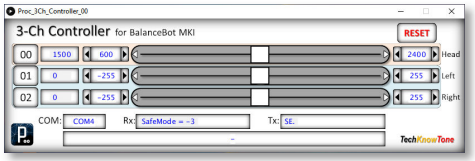
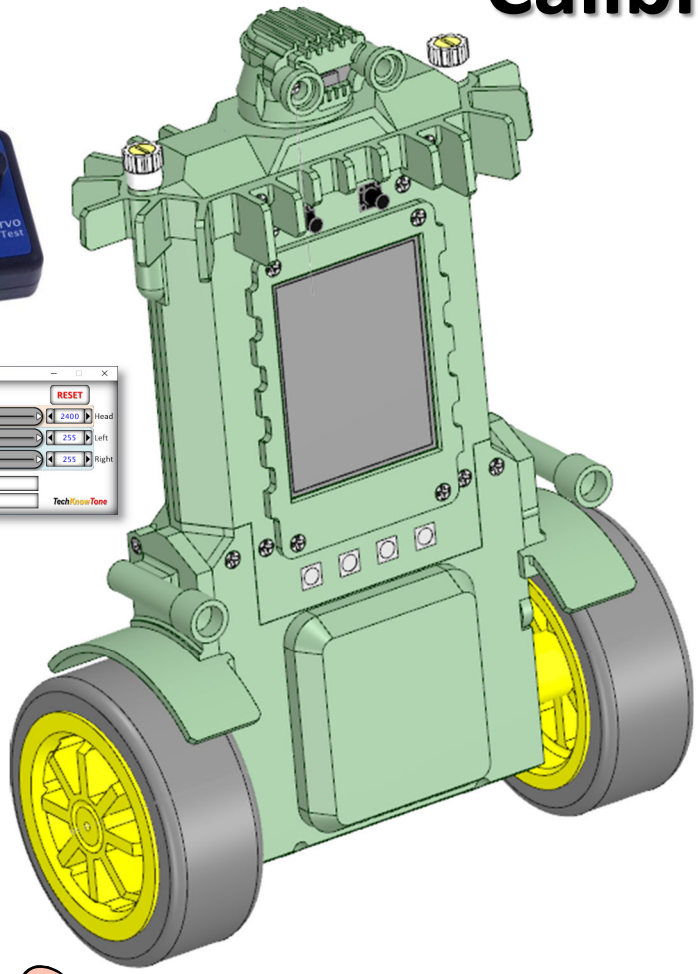
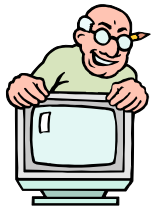
BalanceBot Mk1

Calibration



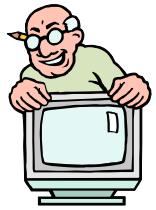




Calibration of head servo and wheel motors is an essential process.

Issue: 1.1 Released: 25/11/2025 TechKnowTone

CAUTION



Lithium batteries can be extremely dangerous, if not handled and cared for properly. This design does not include any form of current limiting circuit, like a fuse. So, care must be taken to ensure that the wiring guidelines are followed accurately, that checks are made for short-circuits, and that battery polarities are marked, and they are inserted the correct way round. Failure to do so, could result in an explosive fire.

Charging Practices: Always remove batteries from your project to charge them. Use a charger, designed for the battery used, and from a trusted supplier. Choose a flat, non-flammable surface to charge on, away from flammable materials. Never leave unattended when charging. Don't charge overnight. Monitor charging to ensure charge characteristics are as expected. Only pair batteries with similar characteristics. Do not overcharge, or leave charging for prolonged periods. This increases the risk of damage and fire.

Battery care & maintenance: Stop using a battery if it is swollen, damaged, dented or leaking. Never charge a damaged battery. Never allow a Lithium battery to discharge below 3.2 volts, as cell damage will occur. Avoid extreme temperatures. Do not charge or store batteries in very hot or cold environments. Don't cover batteries whilst charging, as this can trap heat, causing overheating.

In case of fire: Get out and stay out. If a fire starts, leave immediately, and call the fire brigade. For low voltage Lithium batteries, water is a safe extinguisher.

Built-in Monitoring: Most of my project designs include code, and circuitry, to monitor battery voltage, whilst in use. This code then seeks to alert the operator, when the battery has reached a critical low voltage, before shutting down power consuming circuitry; including the micro. Time should therefore be spent on calibrating this feature, as a precaution, for good battery management and maintenance.

Carefully dispose of batteries that have been discharged below their critical voltage.

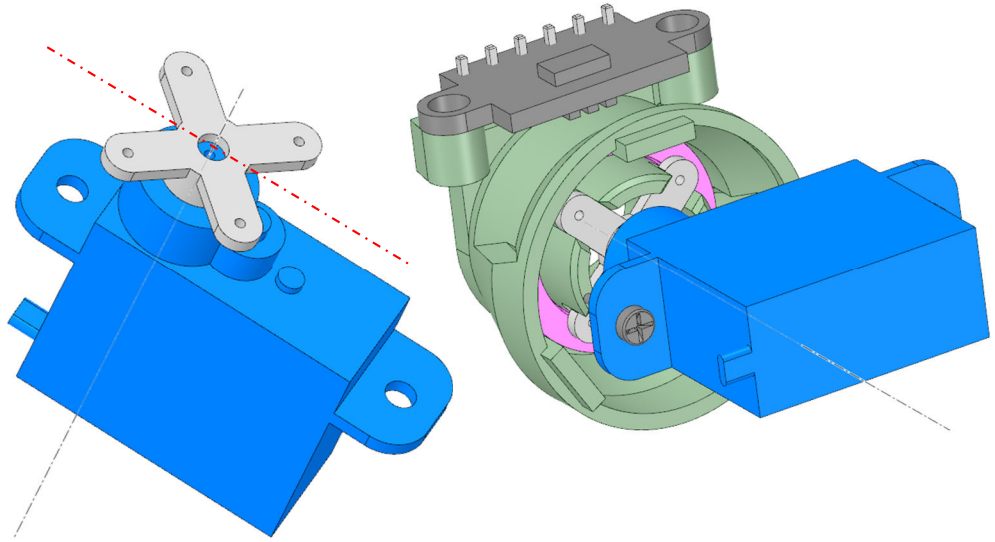
Overview

In this set of procedures we will use a Servo Consistency Tester, and a custom Windows app, written specifically for this project by me.

The servo tester is used to set the centre position, when fitting the cross shaped leaver. This is to ensure that the servo will have sufficient movement range in both directions when mounted in your robot.

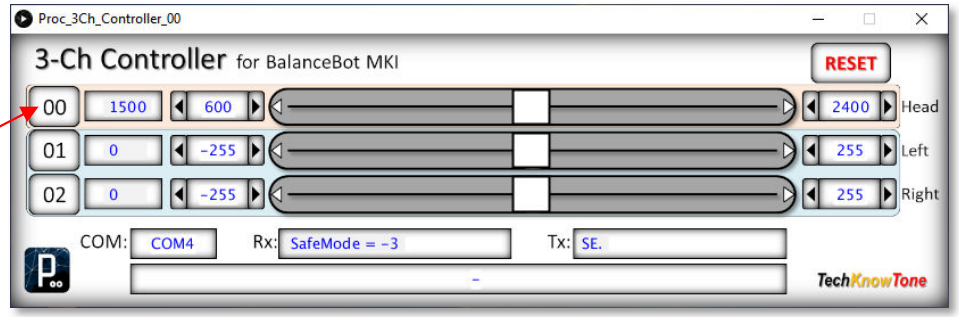
The PWM centre position of a servo is 1,500 µs.

It is simply a matter of powering up the servo tester, plugging in the servo and setting the testers output to 1500. Then fitting the servos cross shaped leaver onto the servos splined drive shaft in the correct position shown here. Given the splined drive shaft teeth positions may prevent you from getting it exactly right, but the nearest position possible will do. This cross shaped leaver mates with recesses in the neck, to drive the head.



Once the head servo is installed in the robot, and the wiring of the chassis is complete, we can then use the 3-channel app to adjust the servo angle, to determine limits to be set in code, and to test the wheel motors to see how much friction there is in their gearboxes.

The channels on this controller correspond to:
00 - head servo angle in microseconds
01 - left-hand motor PWM drive 0 to +/-255
02 - right-hand motor PWM drive 0 to +/-255



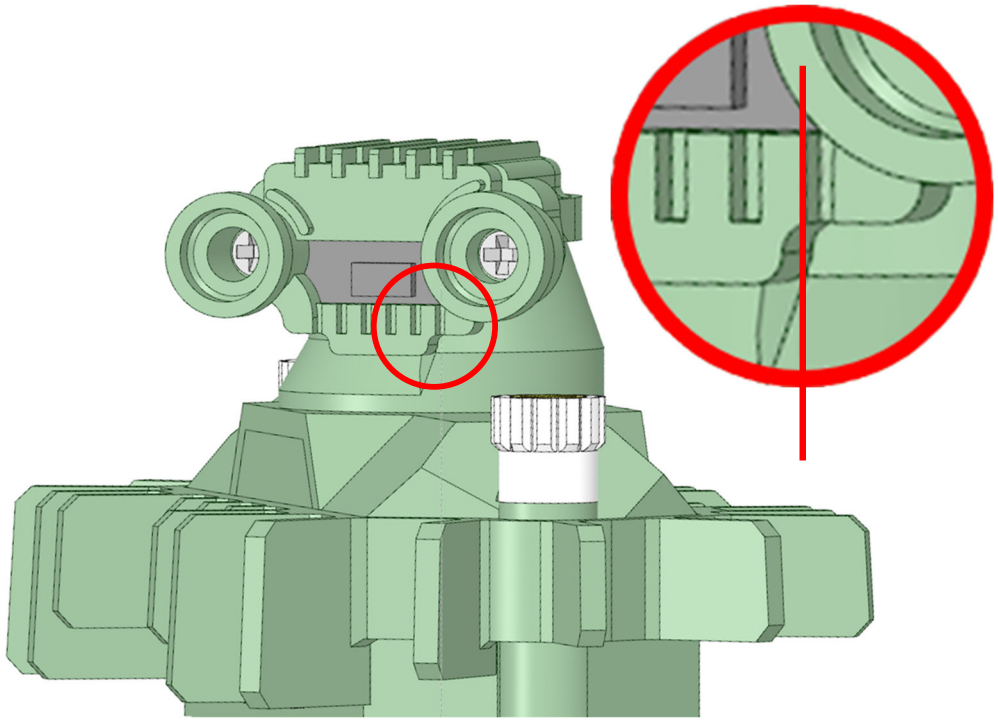
The robot needs to be in TEST mode and you need to turn ON a channel, by clicking on the channel button before it will respond to the slider values.

Head Servo Calibration

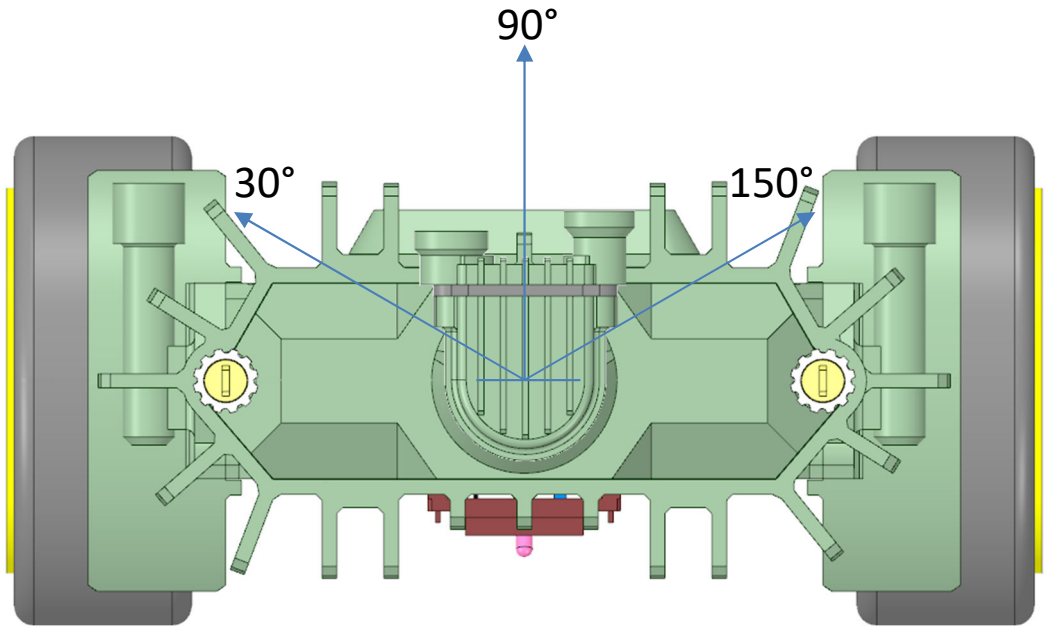
Once the robot is built, set the code into TEST mode and use the 3-channel app to control the position of the head servo, using channel 00.

We want to find the three PWM µs values which set the centre and left/right positions of the head. The left/right positions are approximately +/-60° from centre. Look at the image to the right, which shows the head turned to the left and aligned with the 30° point. The same process applied to the right-hand side.

With the app determine the three values, record them and then enter them into the code in the definition table. Mine we as follows:



```
#define Head_30    885
#define Head_90    1510
#define Head_150 2121
```



Wheel Motor Calibration

With the robot on its stand, set the code into TEST mode and use the 3-channel app to control the motor PWM drive applied through channels 01 and 02.

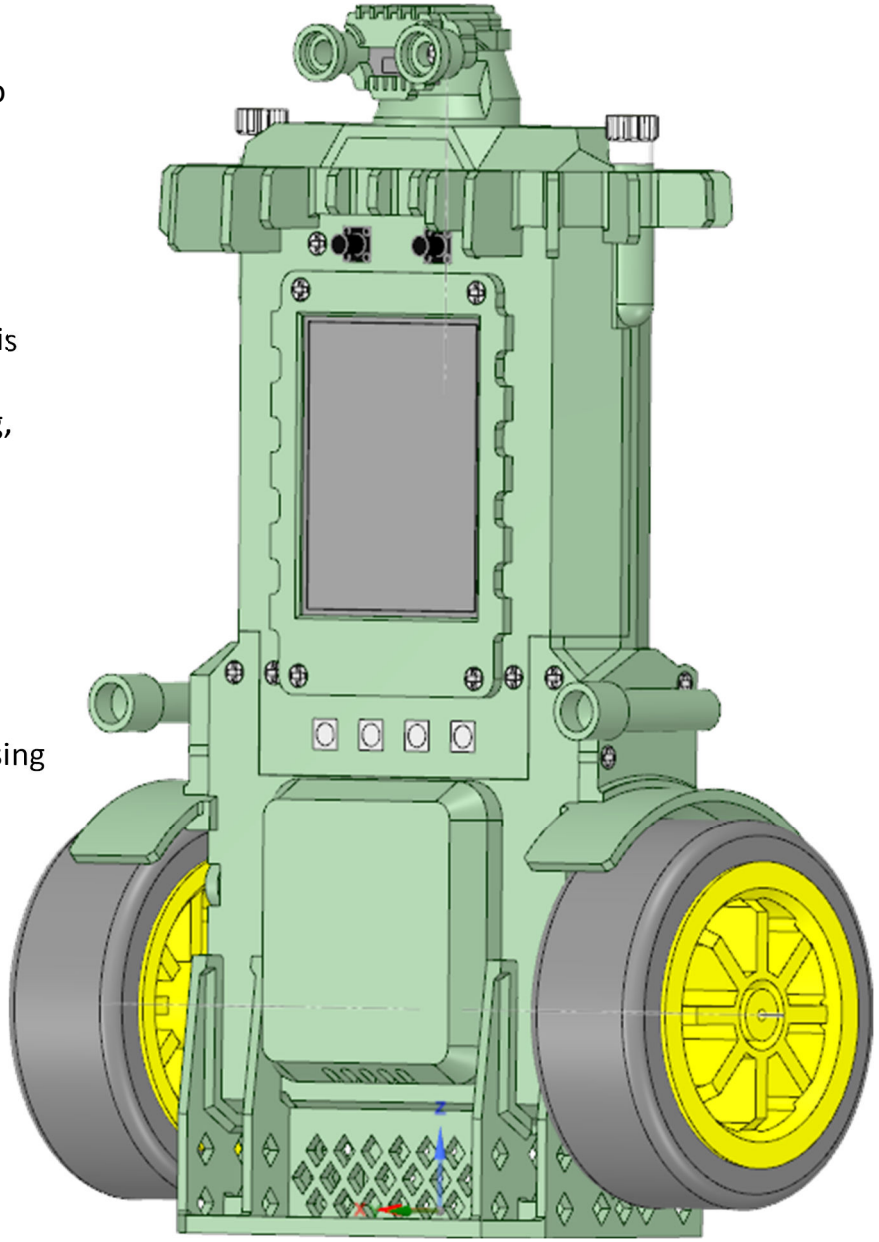
01 - left-hand motor PWM drive 0 to +/-255
02 - right-hand motor PWM drive 0 to +/-255

Adjust the app sliders to determine the point at which the motors just start to turn. It is likely that this value will be different for each motor, and when driven forward or in reverse. Use the maximum value needed, with which both wheels should start turning, either forward or backwards.

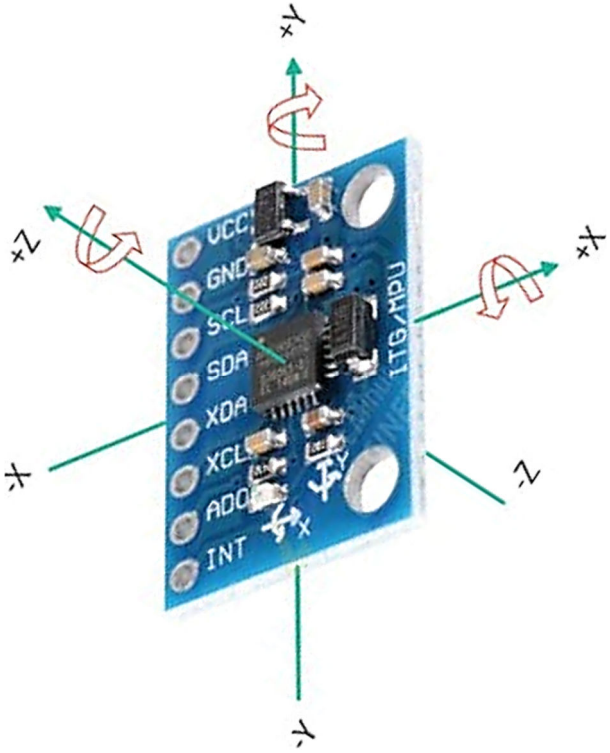
Place this value in your code as follows, under PID controller coefficients, as a floating point variable:

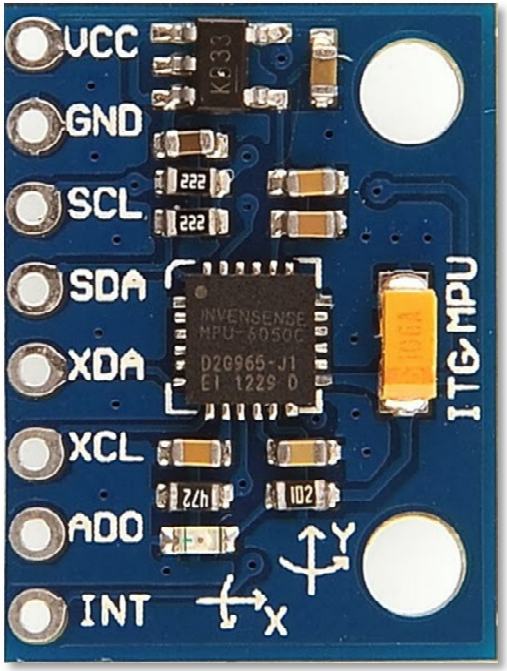
```
float PWM_StartMax = 18.0;        // PWM needed to overcome motor start stiction
```

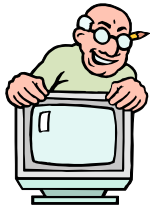
With that entered into your code, you can now move onto tuning the PID controller using the app I have provided for that. See the separate pdf file for that.



MPU 6050A Orientation



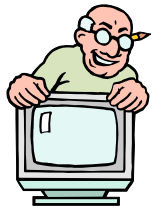


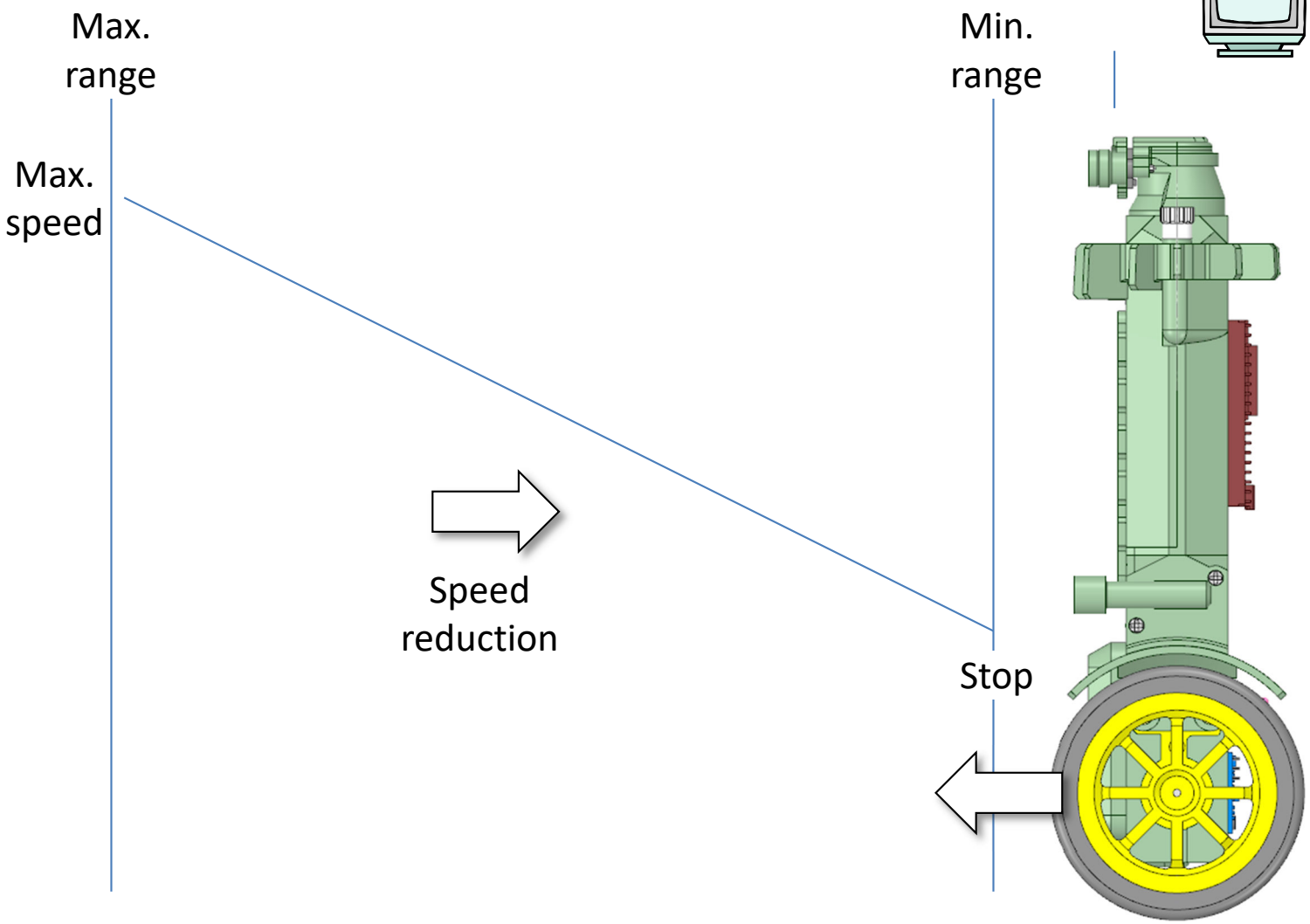


With the MPU6050 mounted on the rear as shown, the following applies:

Pitch - X gyro and Z accelerometer, –ve forwards, +ve backwards
Yaw - Z gyro and X accelerometer
Tip - Y accelerometer, -ve upright, +ve upside down
Turn - Y gyro, +ve left, -ve right

Pilot Mode





In 'Pilot' mode, with the robot static, if an object is <= minimum range then the forward direction is disabled, but turning left or right is allowed.

If the robot is moving forward and sees an object, then the range has a direct effect on the speed term, effectively reducing it to zero when the range is at a minimum value.

In the code we set the following:

Min. range = 100mm

Max. range = 500mm

Battery Voltage Health Monitoring

See 18650 discharge curve obtained from the internet. In this analysis both batteries are identical and connected in series,
Assume fully charged batteries max voltage is V_{BM} >= 8.2v max
I measured my rechargeable PP3 at 8.65v when connected and ON.
Set battery warning point at V_B = 7.00v
Set battery critical point at V_{BC} = 6.60v

ESP32 is powered from batteries connected to V_{in}.
3.3v at VADC == 4095 on 12-bit converter (4095 max).
If we use a 6k8 resistor feeding A0 and a 3k3 resistor to GND, we get a conversion factor of 10.1v == 4095 or 2.47mV/bit or 404.85
Using a Multimeter I determined the conversion factor needed to be reduced to 383.9 to display voltage correctly.

MAX: V_M = 8.2v, gives A0 = 3148 on ADC (V_M* 383.9)

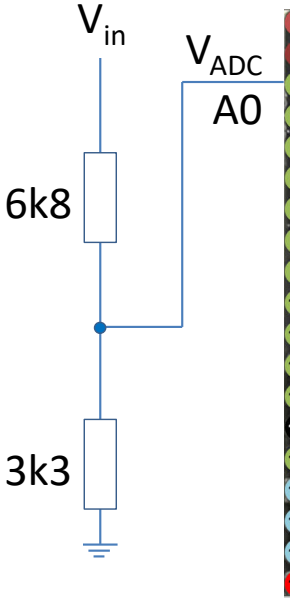
WARNING: V_B = 7.0v, gives A0 = 2687on ADC (V_B* 383.9)

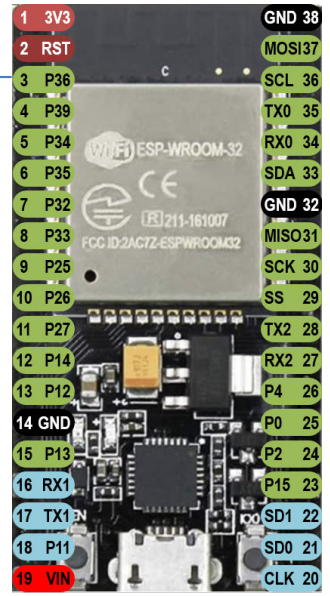
CRITICAL: V_{BC} = 6.6v, gives A0 = 2534 on ADC (V_{BC}* 383.9)

The code will sample the battery voltage on power-up to ensure it is sufficient, then at every 40ms interval, calculating an average (1/20) to remove noise.

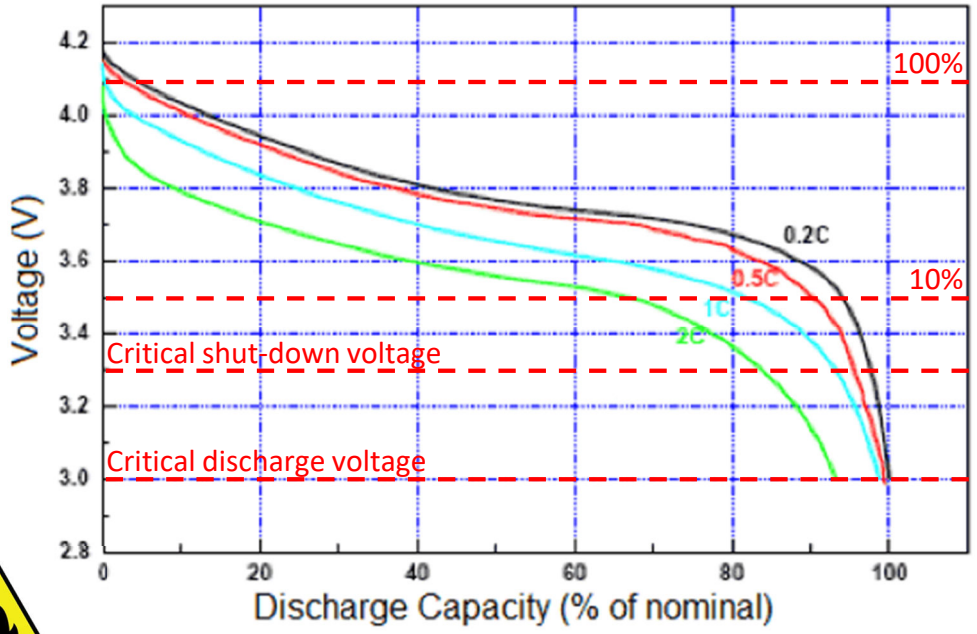
Given the relatively light current drawn I have assumed a linear discharge curve ranging from 8.2v (100%) to 6.6v (0%) capacity. The rate of discharge is monitored and used to actively predict the life of the battery in use.

Note: If connected to USB port with internal battery switched OFF the ADC will read a value 5 volts (A0 = 1919) or less. So if the micro starts with such a low reading it knows that it is on USB power.





18650 Lithium Battery
Discharge Profile



Discharge: 3.0V cutoff at room temperature.

Issue: 1.1 Released: 25/11/2025 TechKnowTone