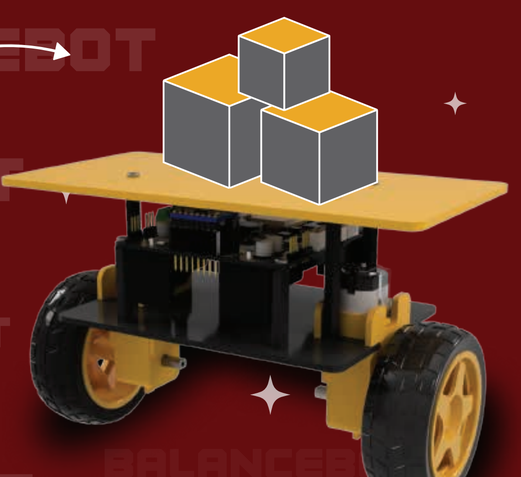
REX8

GitHub
rbt.ist/rexgithub

REX DOCS
rbt.ist/rexrdt

Buy REX
rbt.ist/rexrdt

BALANCEBOT
SETUP GUIDE



BalanceBot

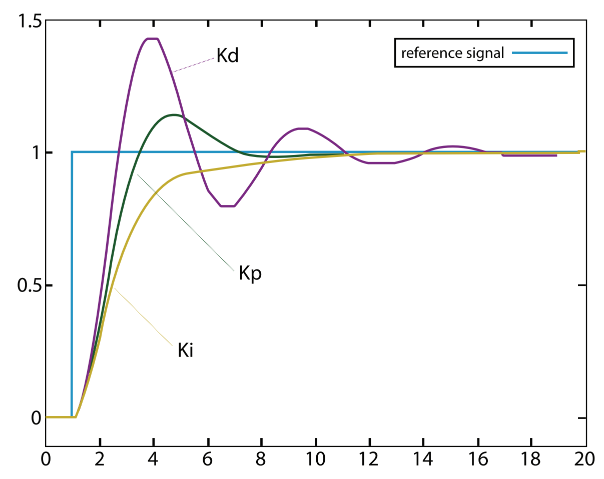
BalanceBot is a REX robot that can maintain balance against changing environmental factors, thanks to the MPU6050 acceleration sensor located on the REX board.

How does BalanceBot Stay in Balance?

Various algorithms can be used to keep BalanceBot in balance. We will use the PID algorithm to keep BalanceBot in balance.

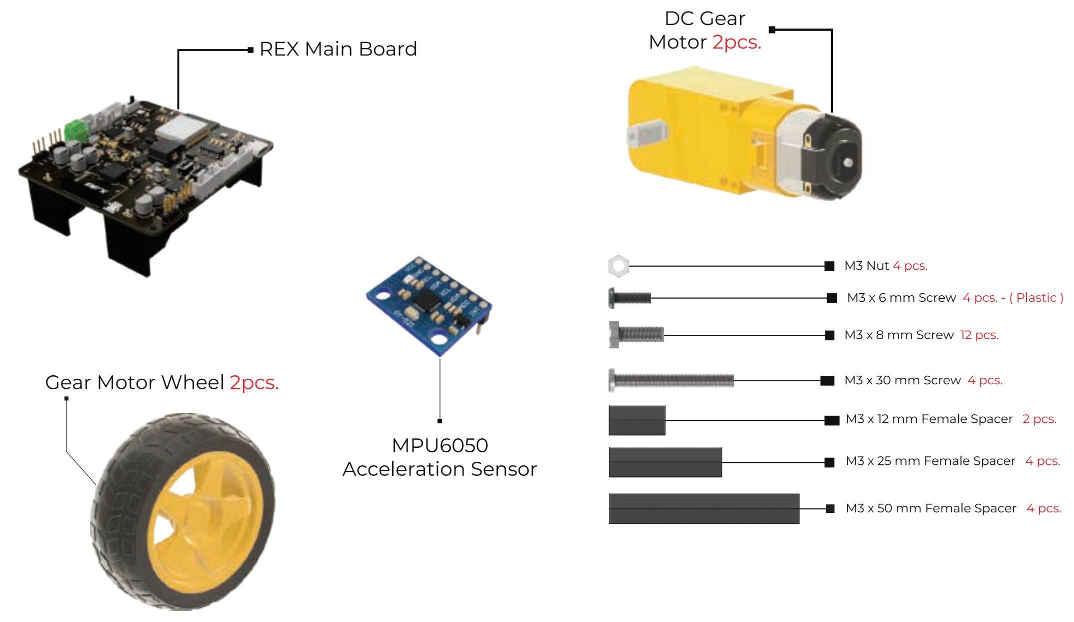
How Does PID (Proportional, Integral, Derivative) Algorithm Work?

The difference between the data coming to the input signal with the feeedback and the input signal is found. This difference creates the error. The error signal is sent to the PID controller, and three different formulas are applied to the error signal with three different parameters. Then, it is returned to the output signal. This process is repeated until the error is minimized.

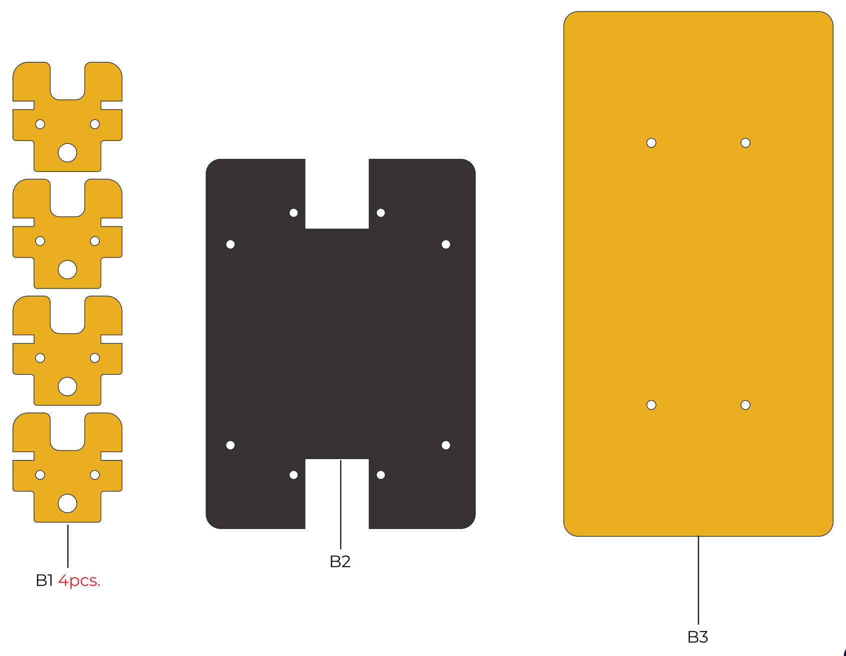


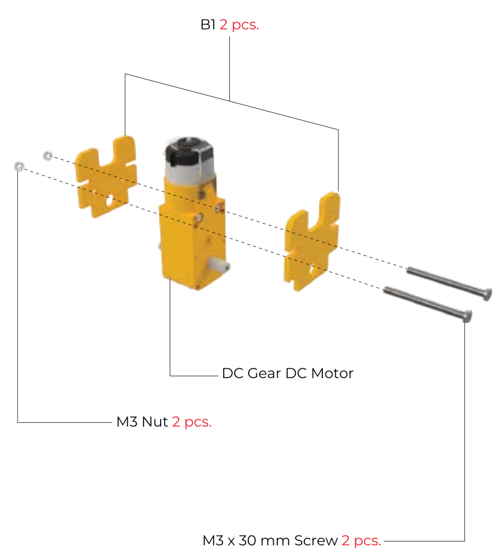
Environmental factors are calculated by using the MPU6050 (gyro/acceleration) sensor on the REX board, and they are sent to the PID. The PID generates the output signal by performing the necessary operations to keep in balance the BalanceBot.

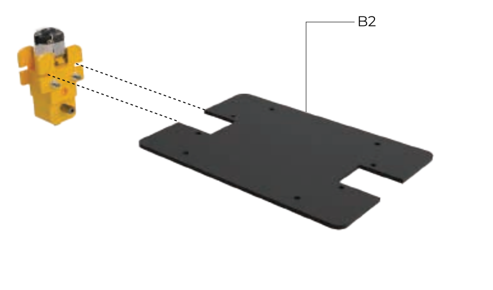
Components of BalanceBot

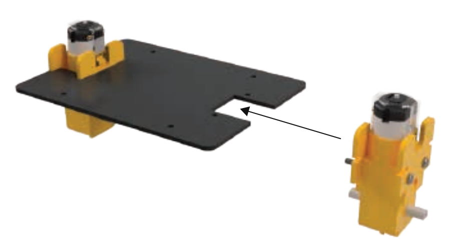

REX Main Board
DC Gear Motor 2pcs.
MPU6050 Acceleration Sensor
Gear Motor Wheel 2pcs.
M3 Nut 4 pcs.
M3 x 6 mm Screw 4 pcs. - (Plastic)
M3 x 8 mm Screw 12 pcs.
M3 x 30 mm Screw 4 pcs.
M3 x 12 mm Female Spacer 2 pcs.
M3 x 25 mm Female Spacer 4 pcs.
M3 x 50 mm Female Spacer 4 pcs.

Plexiglass Parts (Robot Chasis)

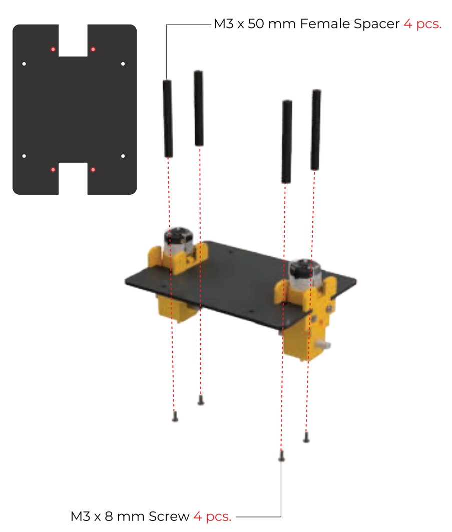


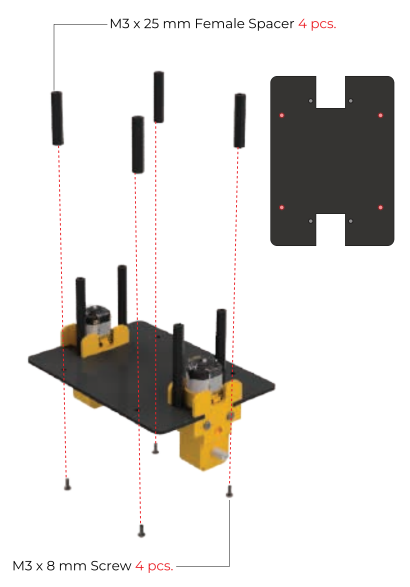


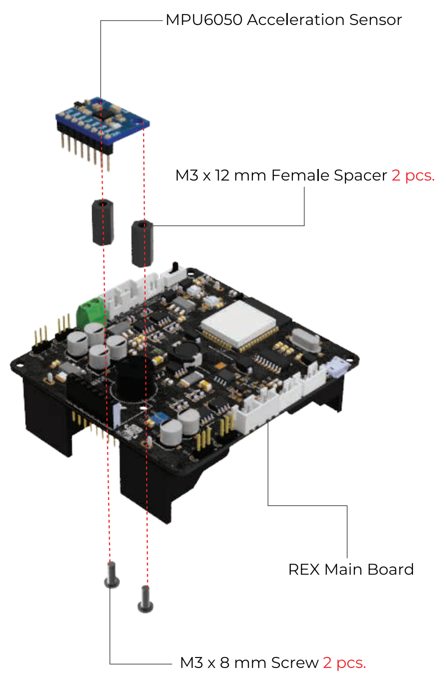


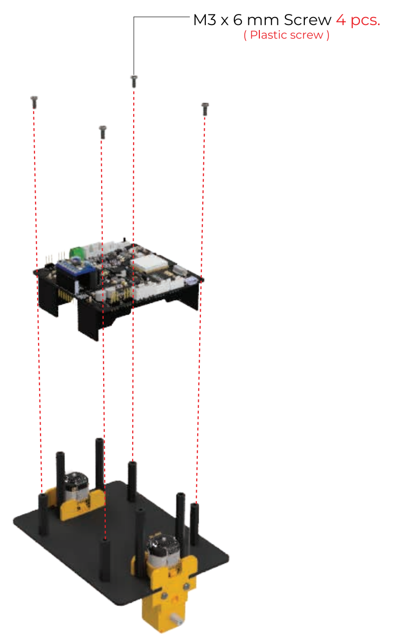


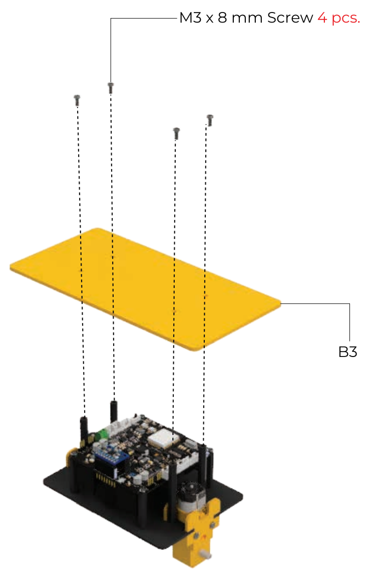
Apply the same assembly steps to other side.

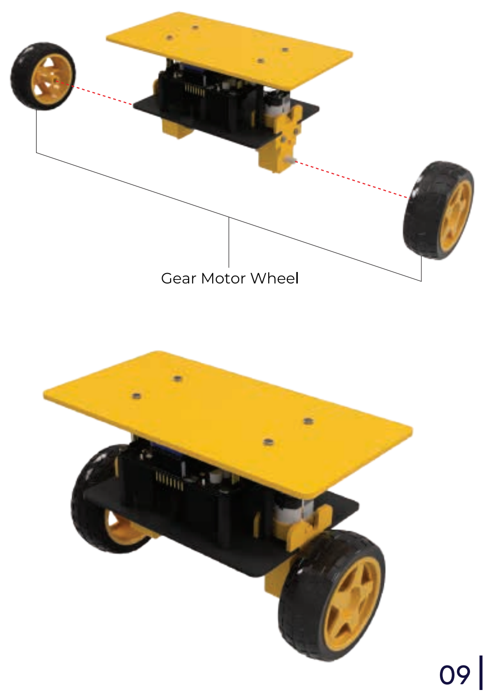


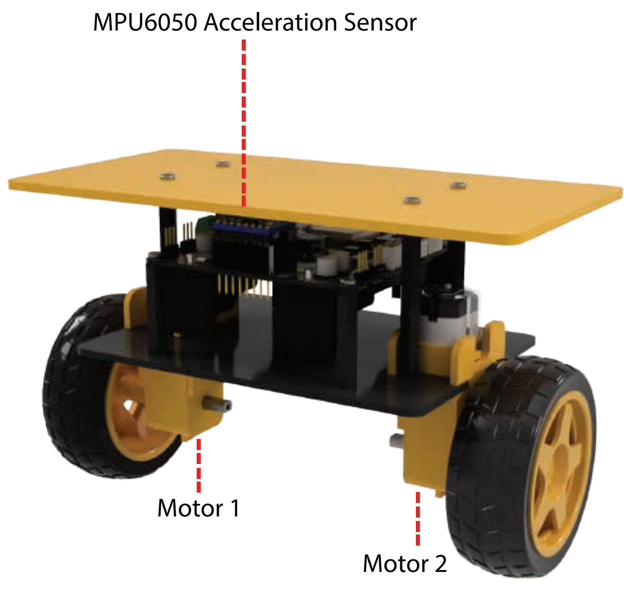






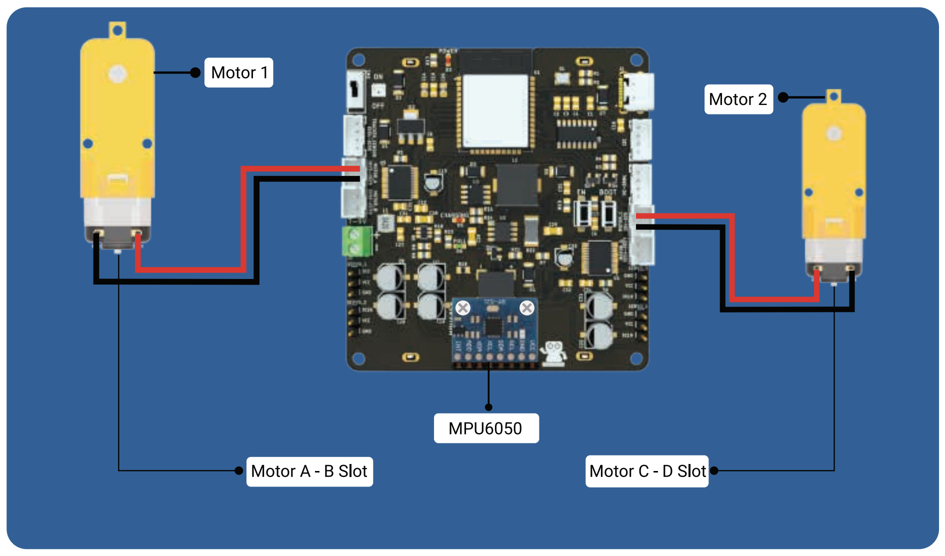




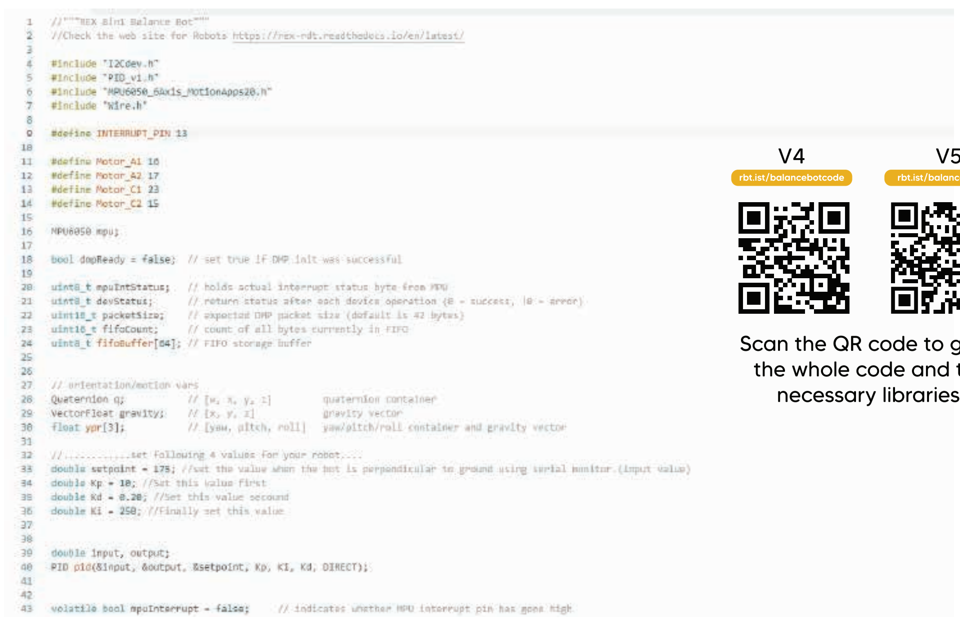


The Circuit Diagram

After assembling the acrylic pieces, you can proceed with circuit installation as shown in the diagram below.



Arduino Code



```
//"""REX 8in1 Balance Bot"""
//Check the web site for Robots https://rex-rdt.readthedocs.io/en/latest/

#include "I2Cdev.h"
#include "PID_v1.h"
#include "MPU6050_6Axis_MotionApps20.h"
#include "Wire.h"

#define INTERRUPT_PIN 13

#define Motor_A1 16
#define Motor_A2 17
#define Motor_C1 23
#define Motor_C2 15

MPU6050 mpu;

bool dmpReady = false;  // set true if DMP init was successful

uint8_t mpuIntStatus;   // holds actual interrupt status byte from MPU
uint8_t devStatus;      // return status after each device operation (0 = success, !0 = error)
uint16_t packetSize;    // expected DMP packet size (default is 42 bytes)
uint16_t fifoCount;     // count of all bytes currently in FIFO
uint8_t fifoBuffer[64]; // FIFO storage buffer


// orientation/motion vars
Quaternion q;           // [w, x, y, z]         quaternion container
VectorFloat gravity;    // [x, y, z]            gravity vector
float ypr[3];           // [yaw, pitch, roll]   yaw/pitch/roll container and gravity vector

//.............set following 4 values for your robot.....
double setpoint = 175; //set the value when the bot is perpendicular to ground using serial monitor.(input value)
double Kp = 10; //Set this value first
double Kd = 0.20; //Set this value second
double Ki = 250; //Finally set this value


double input, output;
PID pid(&input, &output, &setpoint, Kp, KI, Kd, DIRECT);


volatile bool mpuInterrupt = false;     // indicates whether MPU interrupt pin has gone high
```

V4	V5
rbt.ist/balancebotcode	rbt.ist/balancebotV5

Scan the QR code to go to the whole code and the necessary libraries.