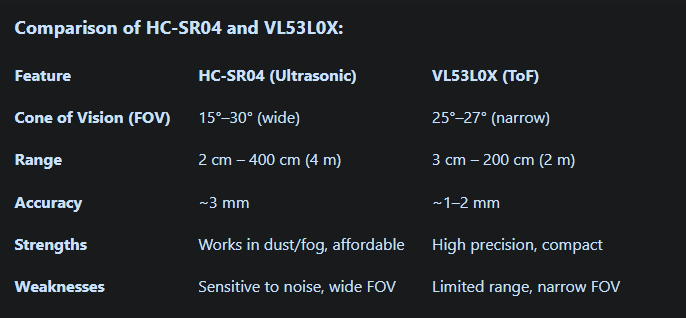
## Nav sensors

****

## LM393 Sound Sensor (all tested, all work)

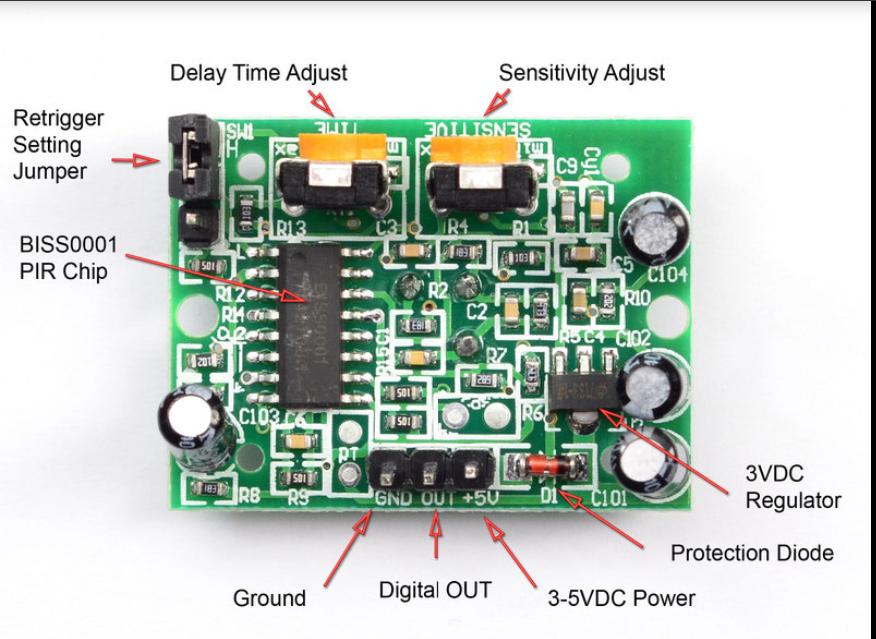
(soundsensor.ino)

* adjust the pot to adjust the sensitivity

<https://github.com/BasOnTech/Arduino-Beginners-EN/blob/master/E30-sound-sensor/E30-sound-sensor-part-2/E30-sound-sensor-part-2.ino>

## HC SR501 PIR (all tested, all work)

(sensorcode.ino)



* very finicky
  + adjust the time delay and sensitivity knobs, motion trigger jumper (place on high)

<https://lastminuteengineers.com/pir-sensor-arduino-tutorial/>

^^^^^^^ completely shows you how to adjust and use adjust switches ^^^^^^^

## ULTRASONIC

* all good

<https://projecthub.arduino.cc/Isaac100/getting-started-with-the-hc-sr04-ultrasonic-sensor-7cabe1>

## TIME OF FLIGHT

* install Adafruit\_VL53L0X on the IDE
* All 5 ToF sensors work, and different distances were tested to verify accuracy
  + Note that pins GPI01 and XSHUT were not used in testing
* <https://www.youtube.com/watch?v=EvaYQMMEtvg> (used but different code used)

#include <Wire.h>

#include "Adafruit\_VL53L0X.h"

Adafruit\_VL53L0X lox = Adafruit\_VL53L0X();

void setup() {

Serial.begin(115200);

Wire.begin(); // Initialize I2C

Serial.println("Adafruit VL53L0X test");

if (!lox.begin()) {

Serial.println(F("Failed to boot VL53L0X"));

while(1);

}

Serial.println(F("VL53L0X API Simple Ranging example\n\n"));

}

void loop() {

VL53L0X\_RangingMeasurementData\_t measure;

lox.rangingTest(&measure, false);

if (measure.RangeStatus != 4) {

Serial.print("Distance (mm): ");

Serial.println(measure.RangeMilliMeter);

} else {

Serial.println("Out of range");

}

delay(100);

}

* Four sensor ToF

#include <Wire.h>

#include "Adafruit\_VL53L0X.h"

// Define sensor objects with position labels

Adafruit\_VL53L0X loxFarRight = Adafruit\_VL53L0X(); // Pin 4

Adafruit\_VL53L0X loxMidRight = Adafruit\_VL53L0X(); // Pin 5

Adafruit\_VL53L0X loxMidLeft = Adafruit\_VL53L0X(); // Pin 6

Adafruit\_VL53L0X loxFarLeft = Adafruit\_VL53L0X(); // Pin 7

// Define XSHUT pins for each position

#define XSHUT\_FAR\_RIGHT 4

#define XSHUT\_MID\_RIGHT 5

#define XSHUT\_MID\_LEFT 6

#define XSHUT\_FAR\_LEFT 7

void setup() {

Serial.begin(9600);

Wire.begin(); // Initialize I2C

// Set XSHUT pins as outputs

pinMode(XSHUT\_FAR\_RIGHT, OUTPUT);

pinMode(XSHUT\_MID\_RIGHT, OUTPUT);

pinMode(XSHUT\_MID\_LEFT, OUTPUT);

pinMode(XSHUT\_FAR\_LEFT, OUTPUT);

// Initialize sensors in order (prevents address conflicts)

initializeSensor(loxFarRight, XSHUT\_FAR\_RIGHT, 0x30, "Far Right");

initializeSensor(loxMidRight, XSHUT\_MID\_RIGHT, 0x31, "Middle Right");

initializeSensor(loxMidLeft, XSHUT\_MID\_LEFT, 0x32, "Middle Left");

initializeSensor(loxFarLeft, XSHUT\_FAR\_LEFT, 0x33, "Far Left");

Serial.println(F("All sensors initialized!"));

}

void loop() {

// Read and print distances with position labels

readSensor(loxFarRight, "Far Right");

readSensor(loxMidRight, "Middle Right");

readSensor(loxMidLeft, "Middle Left");

readSensor(loxFarLeft, "Far Left");

delay(1000); // Adjust for robot responsiveness

}

// Function to initialize a sensor with a unique address and label

void initializeSensor(Adafruit\_VL53L0X &sensor, int xshutPin, uint8\_t newAddress, const char\* position) {

// Reset the sensor (pull XSHUT low, then high)

digitalWrite(xshutPin, LOW);

delay(10);

digitalWrite(xshutPin, HIGH);

delay(10);

// Initialize with the new address

if (!sensor.begin(newAddress)) {

Serial.print(position);

Serial.println(F(" sensor failed to boot!"));

while (1); // Halt if a sensor fails

}

Serial.print(position);

Serial.print(F(" sensor initialized at address 0x"));

Serial.println(newAddress, HEX);

}

// Function to read and print sensor data

void readSensor(Adafruit\_VL53L0X &sensor, const char\* position) {

VL53L0X\_RangingMeasurementData\_t measure;

sensor.rangingTest(&measure, false); // Get distance

if (measure.RangeStatus != 4) {

Serial.print(position);

Serial.print(": ");

Serial.print(measure.RangeMilliMeter);

Serial.println(" mm");

} else {

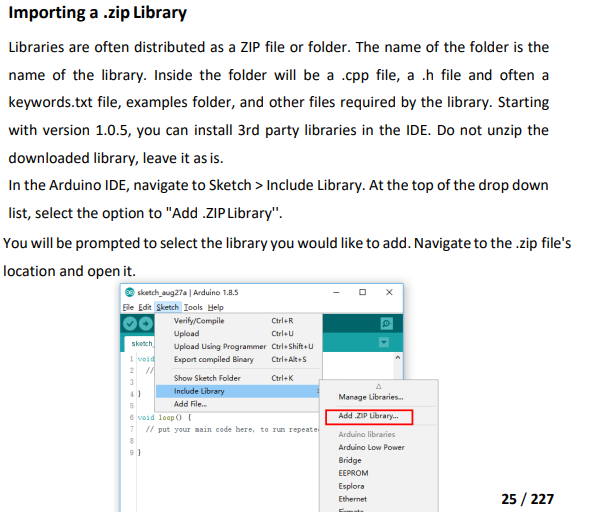
Serial.print(position);

Serial.println(": Out of range");

}

}

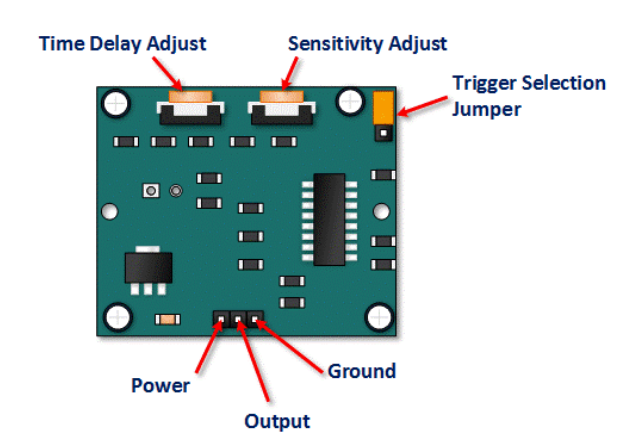
## IR Sensor with Remote control from kit

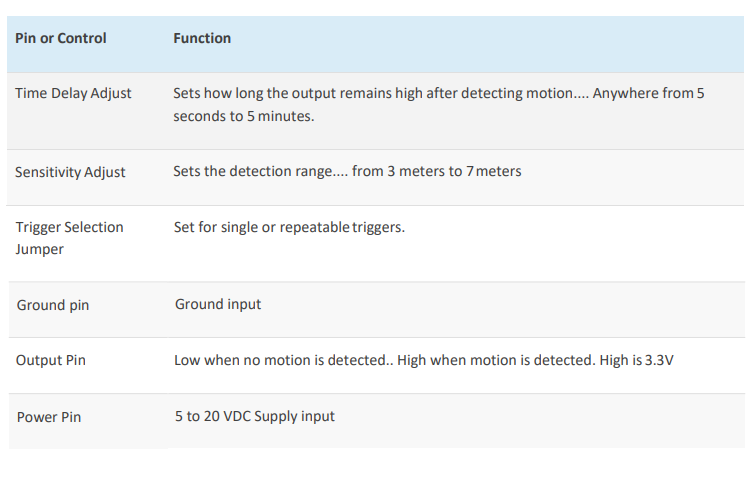


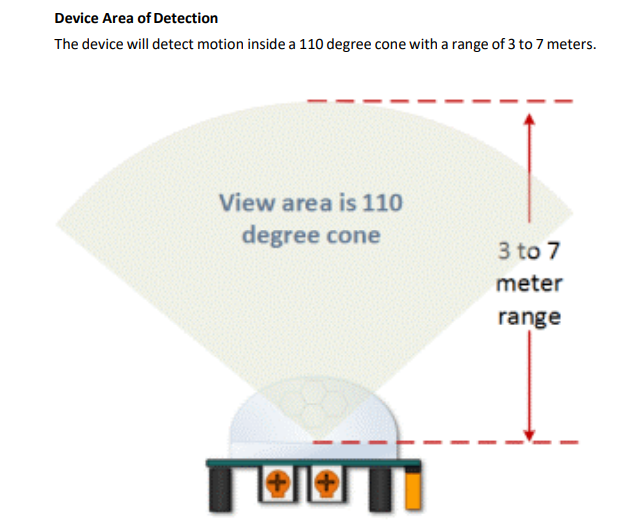
The library is given with Elegoo mega kit document, and is on Sheldon’s computer for testing the remote control.

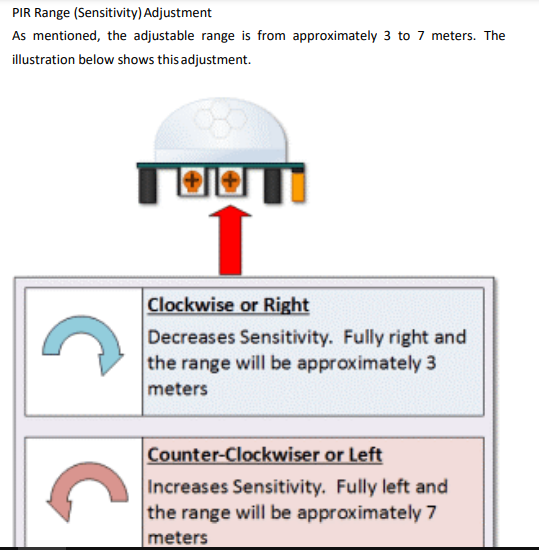
* Works and can tell which input is pressed on the remote

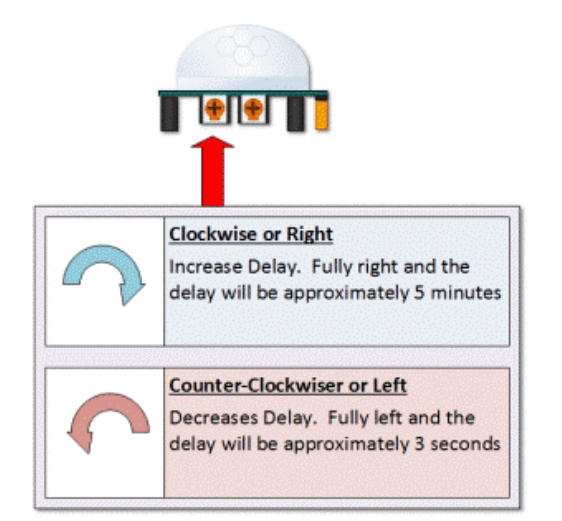
## HC-SR501 PIR Sensor

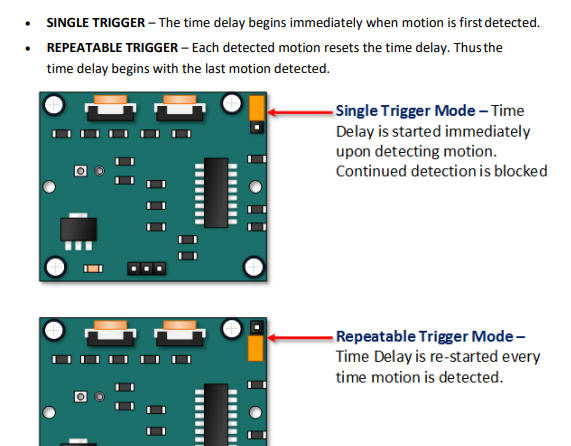
****

****

****

****

****

****

## GYRO MPU9250 /GY-521

* install mpu9250 gyro library on arduino ide
* vcc to 5v, gnd to gnd, scl to \_\_\_\_\_, sda to \_\_\_\_

Revised   
#include <Wire.h>

// MPU-6050 I2C address

const int MPU\_addr = 0x68;

// Gyroscope and accelerometer vars

int16\_t AcX, AcY, AcZ, GyX, GyY, GyZ;

float gyroX, gyroY, gyroZ; // Angular velocity in °/s

float accX, accY, accZ; // Accelerometer data in g

// Orientation variables

float roll, pitch, yaw; // Roll, pitch, and yaw angles

//Note: Dependent on how the sensor is place affects these (roll and pitch are switched currently)

float rollAcc, pitchAcc; // Roll and pitch from accelerometer

// Complementary filter variables

float alpha = 0.98; // Weight for gyroscope data (tune this value if needed)

// Timing variables

float dt; // Time interval

unsigned long prevTime; // Previous time stamp

// Gyroscope sensitivity scale factor (for ±250°/s range)

const float GYRO\_SCALE = 131.0;

// Accelerometer sensitivity scale factor (for ±2g range)

const float ACCEL\_SCALE = 16384.0;

void setup() {

Wire.begin();

Serial.begin(9600);

// Initialize MPU-6050

Wire.beginTransmission(MPU\_addr);

Wire.write(0x6B); // PWR\_MGMT\_1 register

Wire.write(0); // Wake up the MPU-6050

Wire.endTransmission(true);

// Initialize variables

roll = 0;

pitch = 0;

yaw = 0;

prevTime = micros();

}

void loop() {

// Read accelerometer and gyroscope data

Wire.beginTransmission(MPU\_addr);

Wire.write(0x3B); // Start with register 0x3B (ACCEL\_XOUT\_H)

Wire.endTransmission(false);

Wire.requestFrom(MPU\_addr, 14, true); // Request 14 registers

// Read accelerometer data

AcX = Wire.read() << 8 | Wire.read();

AcY = Wire.read() << 8 | Wire.read();

AcZ = Wire.read() << 8 | Wire.read();

// Read temperature data (not used in this example)

Wire.read() << 8 | Wire.read();

// Read gyroscope data

GyX = Wire.read() << 8 | Wire.read();

GyY = Wire.read() << 8 | Wire.read();

GyZ = Wire.read() << 8 | Wire.read();

// Convert raw accelerometer data to g

accX = AcX / ACCEL\_SCALE;

accY = AcY / ACCEL\_SCALE;

accZ = AcZ / ACCEL\_SCALE;

// Convert raw gyroscope data to °/s

gyroX = GyX / GYRO\_SCALE;

gyroY = GyY / GYRO\_SCALE;

gyroZ = GyZ / GYRO\_SCALE;

// Calculate time interval

unsigned long currentTime = micros();

dt = (currentTime - prevTime) / 1000000.0; // Convert to seconds

prevTime = currentTime;

// Calculate roll and pitch from accelerometer

rollAcc = atan2(accY, accZ) \* 180 / PI;

pitchAcc = atan2(-accX, sqrt(accY \* accY + accZ \* accZ)) \* 180 / PI;

// Apply complementary filter to combine gyroscope and accelerometer data

roll = alpha \* (roll + gyroX \* dt) + (1 - alpha) \* rollAcc;

pitch = alpha \* (pitch + gyroY \* dt) + (1 - alpha) \* pitchAcc;

yaw += gyroZ \* dt; // Yaw is only from gyroscope (no accelerometer correction)

// Print orientation

Serial.print("Pitch: ");

Serial.print(roll);

Serial.print(" | Roll: ");

Serial.print(pitch);

Serial.print(" | Yaw: ");

Serial.println(yaw);

delay(10); // Small delay for stability

}

// MPU-6050 Short Example Sketch

//www.elegoo.com

//2016.12.9

#include<Wire.h>

const int MPU\_addr = 0x68; // I2C address of the MPU-6050

int16\_t GyX, GyY, GyZ;

void setup(){

Wire.begin();

Wire.beginTransmission(MPU\_addr);

Wire.write(0x6B); // PWR\_MGMT\_1 register

Wire.write(0); // set to zero (wakes up the MPU-6050)

Wire.endTransmission(true);

Serial.begin(9600);

}

void loop(){

Wire.beginTransmission(MPU\_addr);

Wire.write(0x3B); // starting with register 0x3B (ACCEL\_XOUT\_H)

Wire.endTransmission(false);

Wire.requestFrom(MPU\_addr,14,true); // request a total of 14 registers

GyX = Wire.read()<<8|Wire.read(); // 0x43 (GYRO\_XOUT\_H) & 0x44 (GYRO\_XOUT\_L)

GyY = Wire.read()<<8|Wire.read(); // 0x45 (GYRO\_YOUT\_H) & 0x46 (GYRO\_YOUT\_L)

GyZ = Wire.read()<<8|Wire.read(); // 0x47 (GYRO\_ZOUT\_H) & 0x48 (GYRO\_ZOUT\_L)

Serial.print("X = "); Serial.println(GyX);

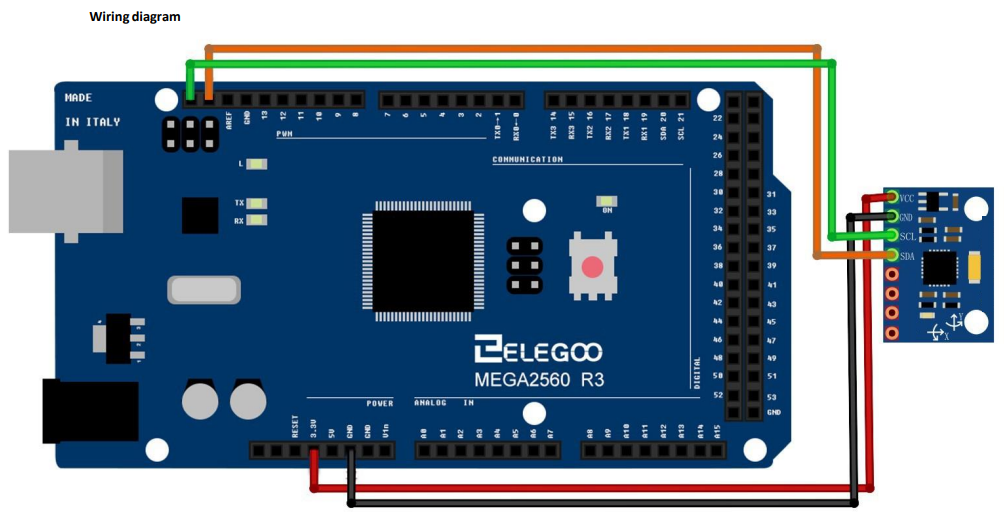
Serial.print("Y = "); Serial.println(GyY);

Serial.print("Z = "); Serial.println(GyZ);

Serial.println();

delay(500);

}



vgggggg