

Internet of things PROJECTS/INTERNSHIP

# College Name

* ****SRM Institute of Science and Technology.****

# Company Name

* ****Vyorius.****

# Domain

* ****Internet of things.****

### Project topics

### IoT based on smart Agriculture System.

### World map covid-19 dashboard.

### Obstacle Avoidance Robot using Ultrasonic Sensor.

### RGB Pattern.

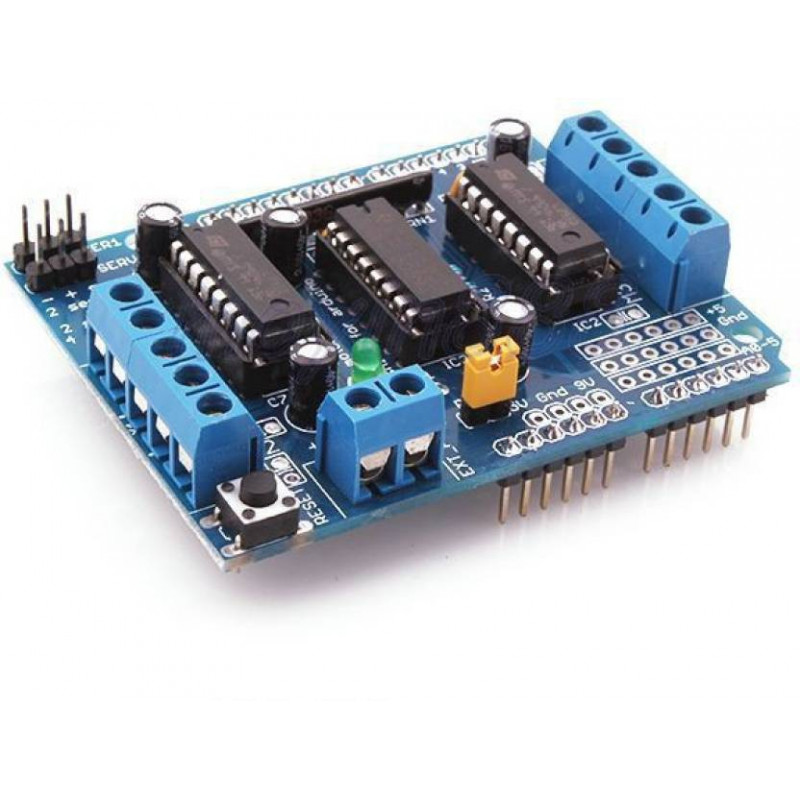
### 3.Obstacle Avoidance Robot using Ultrasonic Sensor

**Software's In detail**



* Tinkercad is a free online collection of software tools that help people all over the world think, create and make. We’re the ideal introduction to [Autodesk](https://www.autodesk.com/), the leader in 3D design, engineering and entertainment software.

**Hardware In detail**



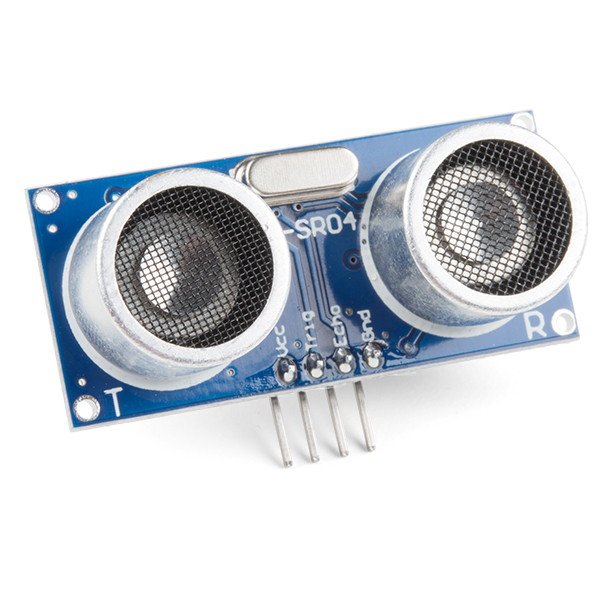
L293D shield is a driver board based on L293 IC, which can drive 4 DC motors and 2 stepper or Servo motors at the same time. Each channel of this module has the maximum current of 1.2A and doesn’t work if the voltage is more than 25v or less than 4.5v. So be careful with choosing the proper motor according to its nominal voltage and current. For more features of this shield let’s mention compatibility with Arduini UNO and MEGA, electromagnetic and thermal protection of motor and disconnecting circuit in case of unconventional voltage raise.



ATmega328P is a **high performance yet low power consumption 8-bit AVR microcontroller** that's able to achieve the most single clock cycle execution of 131 powerful instructions thanks to its advanced RISC architecture. It can commonly be found as a processor in Arduino boards such as Arduino Fio and Arduino Uno



The TowerPro SG90 9g Mini Servo is a **180° rotation servo**. It is a Digital Servo Motor that receives and processes PWM signal faster and better. It equips sophisticated internal circuitry that provides good torque, holding power, and faster updates in response to external forces.



An ultrasonic sensor is **an electronic device that measures the distance of a target object by emitting ultrasonic sound waves**, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound (i.e., the sound that humans can hear).



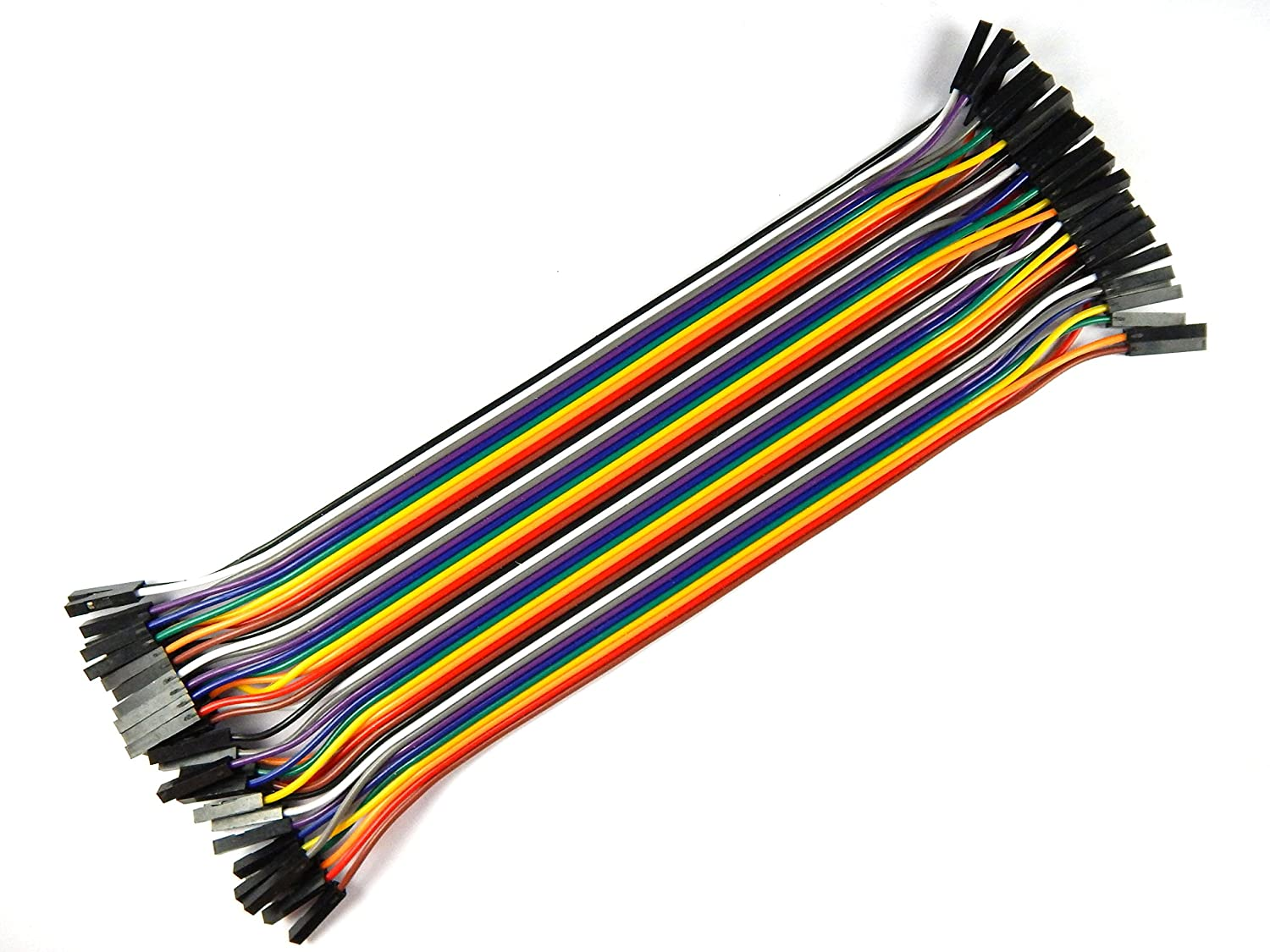
A gear motor is **an all-in-one combination of a motor and gearbox**. The addition of a gear head to a motor reduces the speed while increasing the torque output. The most important parameters in regards to gear motors are speed (rpm), torque (lb-in) and efficiency (%).



A **wheel** is a circular component that is intended to rotate on an axle bearing. The wheel is one of the key components of the wheel and axle which is one of the six simple machines. Wheels, in conjunction with axles, allow heavy objects to be moved easily facilitating movement or transportation while supporting a load, or performing labour in machines. Wheels are also used for other purposes, such as a ship's wheel, steering wheel, potter's wheel and flywheel.

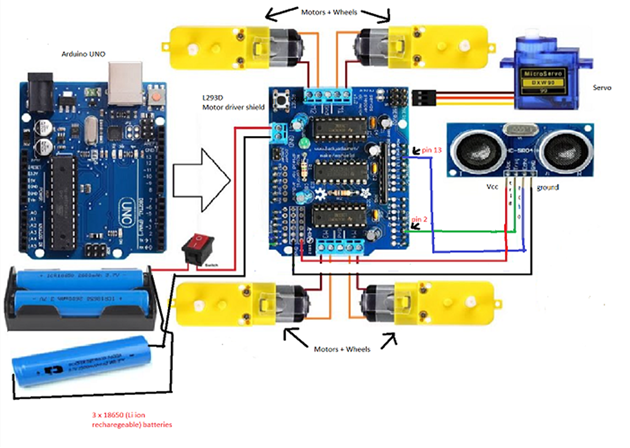


18650 batteries are lithium-ion batteries. They get their name from their size: 18mm by 65mm. A High Drain battery basically means the battery gives out a larger amount of current than regular batteries. 18650 batteries can have anywhere from 300 to 500 charging cycles, which is insane!



A jump wire (also known as jumper, jumper wire, jumper cable, DuPont wire or cable) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test.

**CIRCUIT VIEW**



**CODE**

#include <AFMotor.h> //Import library to control motor shield

#include <Servo.h> //Import library to control the servo

AF\_DCMotor rightBack(1);//Create an object to control each motor

AF\_DCMotor rightFront(2);

AF\_DCMotor leftFront(3);

AF\_DCMotor leftBack(4);

Servo servoLook; //Create an object to control the servo

byte trig = 2; //Assign the ultrasonic sensor pins

byte echo = 13;

byte maxDist = 150; //Maximum sensing distance (Objects further than this distance are ignored)

byte stopDist = 50; //Minimum distance from an object to stop in cm

float timeOut = 2\*(maxDist+10)/100/340\*1000000 // Maximum time to wait for a return signal

byte motorSpeed = 55; //The maximum motor speed

int motorOffset = 10; //Factor to account for one side being more powerful

int turnSpeed = 50; //Amount to add to motor speed when turning

void setup()

{

rightBack.setSpeed(motorSpeed); //Set the motors to the motor speed

rightFront.setSpeed(motorSpeed);

leftFront.setSpeed(motorSpeed+motorOffset);

leftBack.setSpeed(motorSpeed+motorOffset);

rightBack.run(RELEASE); //Ensure all motors are stopped

rightFront.run(RELEASE);

leftFront.run(RELEASE);

leftBack.run(RELEASE);

servoLook.attach(10); //Assign the servo pin

pinMode(trig,OUTPUT); //Assign ultrasonic sensor pin modes

pinMode(echo,INPUT);

}

void loop()

{

servoLook.write(90); //Set the servo to look straight ahead

delay(750);

int distance = getDistance(); //Check that there are no objects ahead

if(distance >= stopDist) //If there are no objects within the stopping distance, move forward

{

moveForward();

}

while(distance >= stopDist) //Keep checking the object distance until it is within the minimum stopping distance

{

distance = getDistance();

delay(250);

}

stopMove(); //Stop the motors

int turnDir = checkDirection(); //Check the left and right object distances and get the turning instruction

switch (turnDir) //Turn left, turn around or turn right depending on the instruction

{

case 0: //Turn left

turnLeft (400);

break;

case 1: //Turn around

turnLeft (700);

break;

case 2: //Turn right

turnRight (400);

break;

}

}

void moveForward() //Set all motors to run forward

{

rightBack.run(FORWARD);

rightFront.run(FORWARD);

leftFront.run(FORWARD);

leftBack.run(FORWARD);

}

void stopMove() //Set all motors to stop

{

rightBack.run(RELEASE);

rightFront.run(RELEASE);

leftFront.run(RELEASE);

leftBack.run(RELEASE);

}

void turnLeft(int duration) //Set motors to turn left for the specified duration then stop

{

rightBack.setSpeed(motorSpeed+turnSpeed); //Set the motors to the motor speed

rightFront.setSpeed(motorSpeed+turnSpeed);

leftFront.setSpeed(motorSpeed+motorOffset+turnSpeed);

leftBack.setSpeed(motorSpeed+motorOffset+turnSpeed);

rightBack.run(FORWARD);

rightFront.run(FORWARD);

leftFront.run(BACKWARD);

leftBack.run(BACKWARD);

delay(duration);

rightBack.setSpeed(motorSpeed); //Set the motors to the motor speed

rightFront.setSpeed(motorSpeed);

leftFront.setSpeed(motorSpeed+motorOffset);

leftBack.setSpeed(motorSpeed+motorOffset);

rightBack.run(RELEASE);

rightFront.run(RELEASE);

leftFront.run(RELEASE);

leftBack.run(RELEASE);

}

void turnRight(int duration) //Set motors to turn right for the specified duration then stop

{

rightBack.setSpeed(motorSpeed+turnSpeed); //Set the motors to the motor speed

rightFront.setSpeed(motorSpeed+turnSpeed);

leftFront.setSpeed(motorSpeed+motorOffset+turnSpeed);

leftBack.setSpeed(motorSpeed+motorOffset+turnSpeed);

rightBack.run(BACKWARD);

rightFront.run(BACKWARD);

leftFront.run(FORWARD);

leftBack.run(FORWARD);

delay(duration);

rightBack.setSpeed(motorSpeed); //Set the motors to the motor speed

rightFront.setSpeed(motorSpeed);

leftFront.setSpeed(motorSpeed+motorOffset);

leftBack.setSpeed(motorSpeed+motorOffset);

rightBack.run(RELEASE);

rightFront.run(RELEASE);

leftFront.run(RELEASE);

leftBack.run(RELEASE);

}

int getDistance() //Measure the distance to an object

{

unsigned long pulseTime; //Create a variable to store the pulse travel time

int distance; //Create a variable to store the calculated distance

digitalWrite(trig, HIGH); //Generate a 10 microsecond pulse

delayMicroseconds(10);

digitalWrite(trig, LOW);

pulseTime = pulseIn(echo, HIGH, timeOut); //Measure the time for the pulse to return

distance = (float)pulseTime \* 340 / 2 / 10000; //Calculate the object distance based on the pulse time

return distance;

}

int checkDirection() //Check the left and right directions and decide which way to turn

{

int distances [2] = {0,0}; //Left and right distances

int turnDir = 1; //Direction to turn, 0 left, 1 reverse, 2 right

servoLook.write(180); //Turn servo to look left

delay(500);

distances [0] = getDistance(); //Get the left object distance

servoLook.write(0); //Turn servo to look right

delay(1000);

distances [1] = getDistance(); //Get the right object distance

if (distances[0]>=200 && distances[1]>=200) //If both directions are clear, turn left

turnDir = 0;

else if (distances[0]<=stopDist && distances[1]<=stopDist) //If both directions are blocked, turn around

turnDir = 1;

else if (distances[0]>=distances[1]) //If left has more space, turn left

turnDir = 0;

else if (distances[0]<distances[1]) //If right has more space, turn right

turnDir = 2;

return turnDir;

}