**————————–**

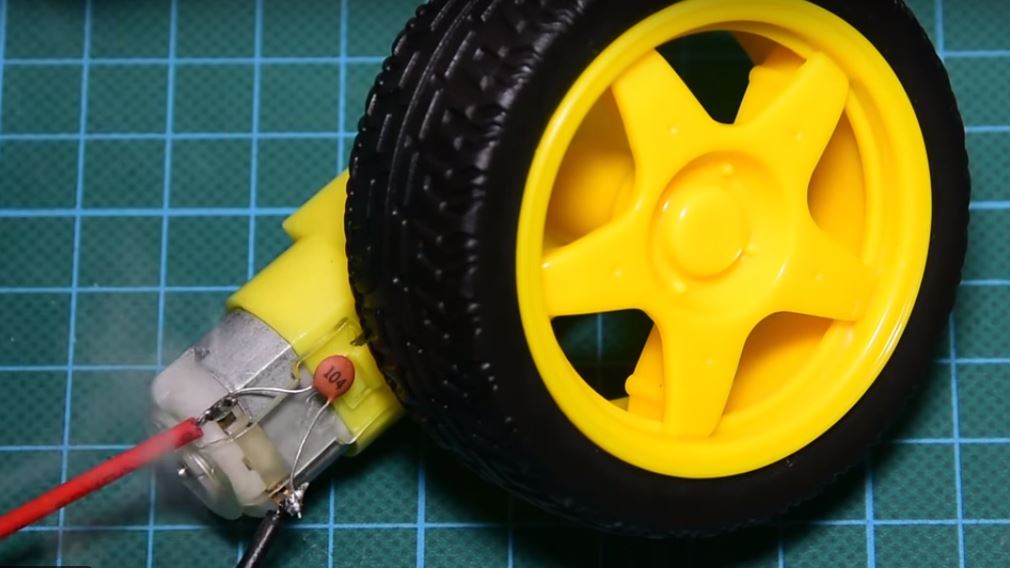
**Building the Robot**

**————————–**

To begin, we start by coupling the chassis of the robot. The robot kit, contains the chassis, two geared DC motors, the wheels, the front wheel,  battery holder, some screws, and wires. The kit also contains a guide which can be followed in steps like we will be doing to build the robot.

**Step 1. Connect the motor and wheels to the chassis.**

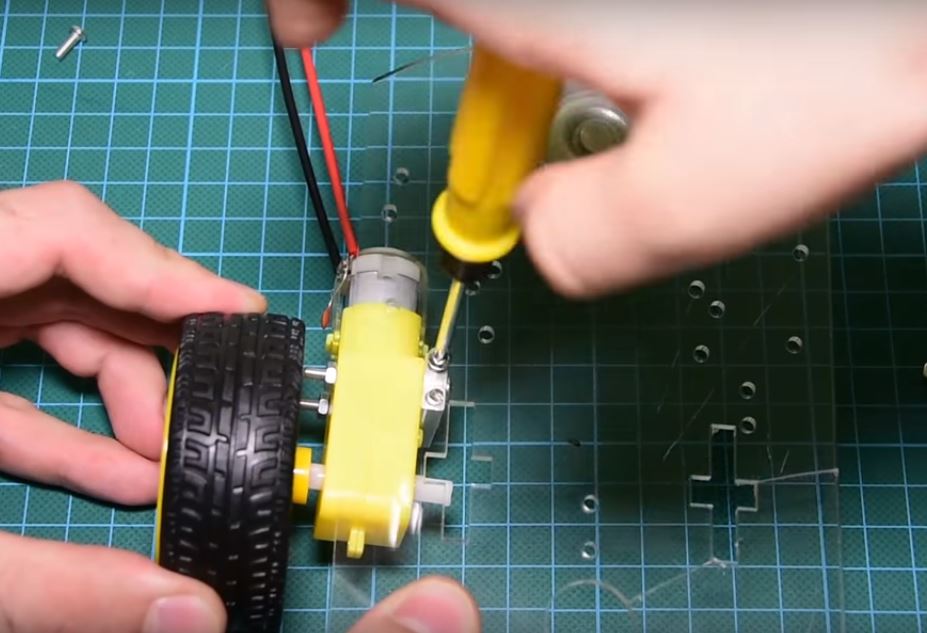
To complete this step, we start by soldering the thick red and black wires to the positive and negative terminals of the motors and in order to reduce the electrical noise that the motor produces, we solder the small ceramic capacitors to the motors as shown in the image below. Since we are using ceramic capacitors which do not have polarity, just solder one leg to each motor terminal as the orientation doesn’t matter.



Next, We attach the front wheel. My little nephew helps me with this, he owns the small hands in the picture. This project is great for kids!

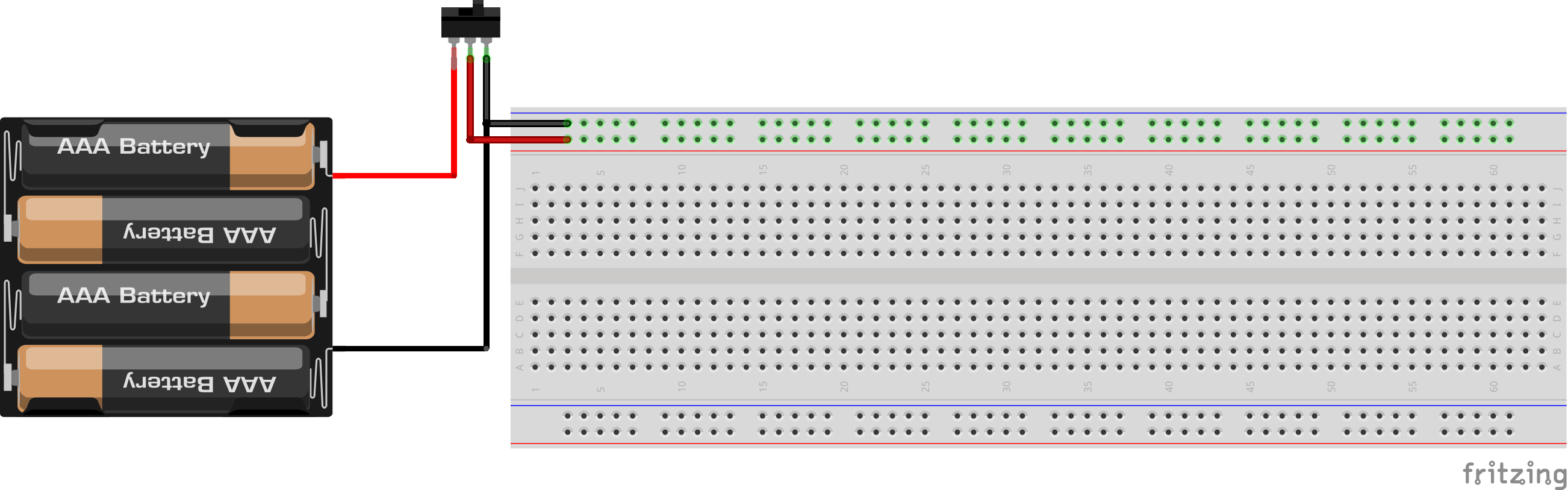


Next, we attach the rear wheels to the chassis in accordance with the instruction leaflet provided attached to the kit.



**Step 2: Prepare the Switch and connect the Power Source**

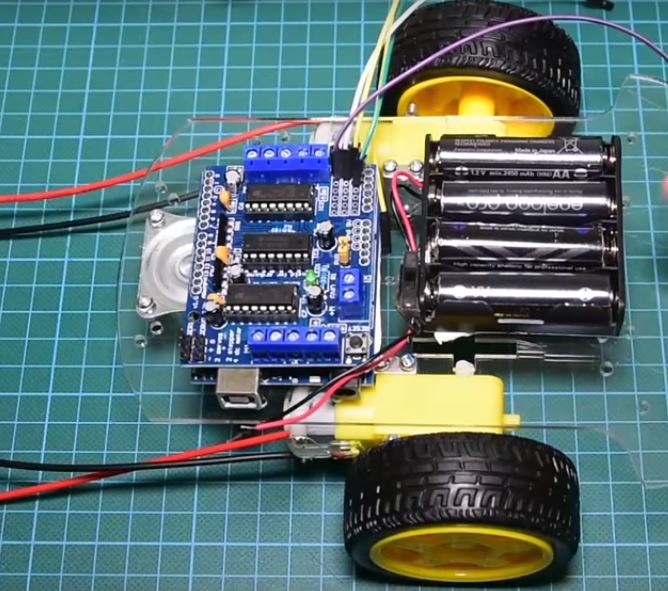
Next, we add a switch to the battery holder so that we will be able to turn the robot on or off. The switch is connected according to the schematics shown below and attached to the case using a hot glue. The Battery case is attached to the chassis using a double-sided tape to ensure everything sticks together.



Switch Connection

**Step 3: Mount other Parts on the chassis.**

The next step is to mount other parts of the robot before we start connecting their wires. The motor shield is stacked on the Arduino and it is mounted on the chassis using a double-sided tape.  The current requirements of the motors are often higher than what the Arduino can provide, that is why it’s important to use the motor shield as it is equipped with additional circuitry to provide up to 600mA current to each of the motors. This shield provides power to the motors and the servo motor and makes our life much easier. The Ultrasonic sensor is also mounted on the top of the servo motor which is then mounted on the chassis using a double-sided tape. I could have easily made a 3D print part in order to achieve these things but since most of the viewers don’t have access to a 3D printer yet, I decided to go with this simple solution.



Coupling the robot

With this part done, we are through with coupling the robot and can move to wiring the components.

Schematics

Wire up the components together as shown in the image below.

Schematics

To simplify the connections, below is a pin map of how the components connect, one to the other.

**Ultrasonic Sensor Motor Shield**

VCC 5v

Gnd Gnd

Trig A4

Echo A5

**Servo Motor Shield(Servo\_2 port)**

Signal(yellow wire) S

Vcc(Red wire) +

Gnd(Black wire) -

Double check your connections to be sure everything is as it should be. Slide the switch to turn it on and you should see a green light come up on the motor driver, indicating power presence. If this does not occur, take a look at the schematics and correct any error.

With everything confirmed, we can then proceed to take a look at the code for this project.

**——-**

**Code**

**——**

The code uses three libraries. Two of them must be downloaded in order for the program to compile. The first one is the motor shield driver from Adafruit. The second library is the NewPing library for the supersonic distance sensor. You can find the links for both libraries in the description of the video.

——————–  
LIBRARIES  
——————–

Motor Shield Library: <https://github.com/adafruit/Adafruit-Motor-Shield-library>

New Ping Library: <https://bitbucket.org/teckel12/arduino-new-ping/wiki/Home#!download-install>

Let’s now take a quick look at the code in order to see how it works.

The first thing we do in the code is to include the libraries that have been downloaded into the code.

//////////////////////////////////////////////

// Arduino ROBOT v0.1 //

// //

// http://www.educ8s.tv //

/////////////////////////////////////////////

#include <AFMotor.h>

#include <NewPing.h>

#include <Servo.h>

Next, we declare the pins to which or ultrasonic sensor is connected to, and some variables which will be used to store info as the code runs then we set the speed of the motors. You can set any value up to 255.

#define TRIG\_PIN A4

#define ECHO\_PIN A5

#define MAX\_DISTANCE 200

#define MAX\_SPEED 190 // sets speed of DC motors

#define MAX\_SPEED\_OFFSET 20

Next, we Initialize the servo motor by creating an object of the servo library. We Also initialize the Motors to which the wheels are connected using the AF library.

AF\_DCMotor motor1(1, MOTOR12\_1KHZ);

AF\_DCMotor motor2(3, MOTOR12\_1KHZ);

Servo myservo;

We then move into the void setup() function where we initialize the servo motor and we set it to “look” straight. In my case, this was at 115 degrees angle, but you might need to do some maths or trial and error to get the degrees which signify the ultrasonic sensor is forward faced. Next, we read the distance a few times in order to get a valid distance measurement.

**void** setup() {

myservo.attach(9);

myservo.write(115);

delay(2000);

distance = readPing();

delay(100);

distance = readPing();

delay(100);

distance = readPing();

delay(100);

distance = readPing();

delay(100);

}

Now we go to the loop function which executes every 40 ms. If the distance we measured is less than or equal to 15 cm, we stop the motors, reverse for 300ms, stop,  look right and left and measure the distance in each direction. If the distance in one direction is greater than the other we turn the robot to the direction with the farthest/greatest distance and instruct it to move forward.

**void** loop() {

**int** distanceR = 0;

**int** distanceL = 0;

delay(40);

**if**(distance<=15)

{

moveStop();

delay(100);

moveBackward();

delay(300);

moveStop();

delay(200);

distanceR = lookRight();

delay(200);

distanceL = lookLeft();

delay(200);

**if**(distanceR>=distanceL)

{

turnRight();

moveStop();

}**else**

{

turnLeft();

moveStop();

}

}**else**

{

moveForward();

}

distance = readPing();

}

That’s the simple algorithm I developed for this robot so far. As always you can find the code of the project in a link in the description of the video and also by clicking the link below.  As earlier mentioned, feel free to modify the code to suit your needs.

——————–