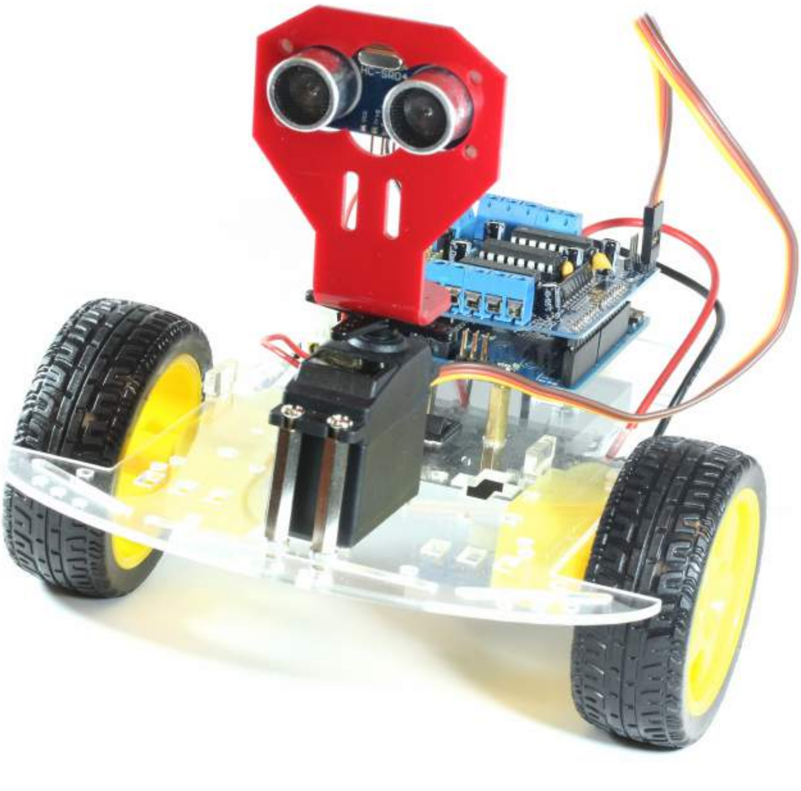
Arduino Robotics Kit With Motor Shield



Document No. ODD-008

Fall 2014

Contents

Overview

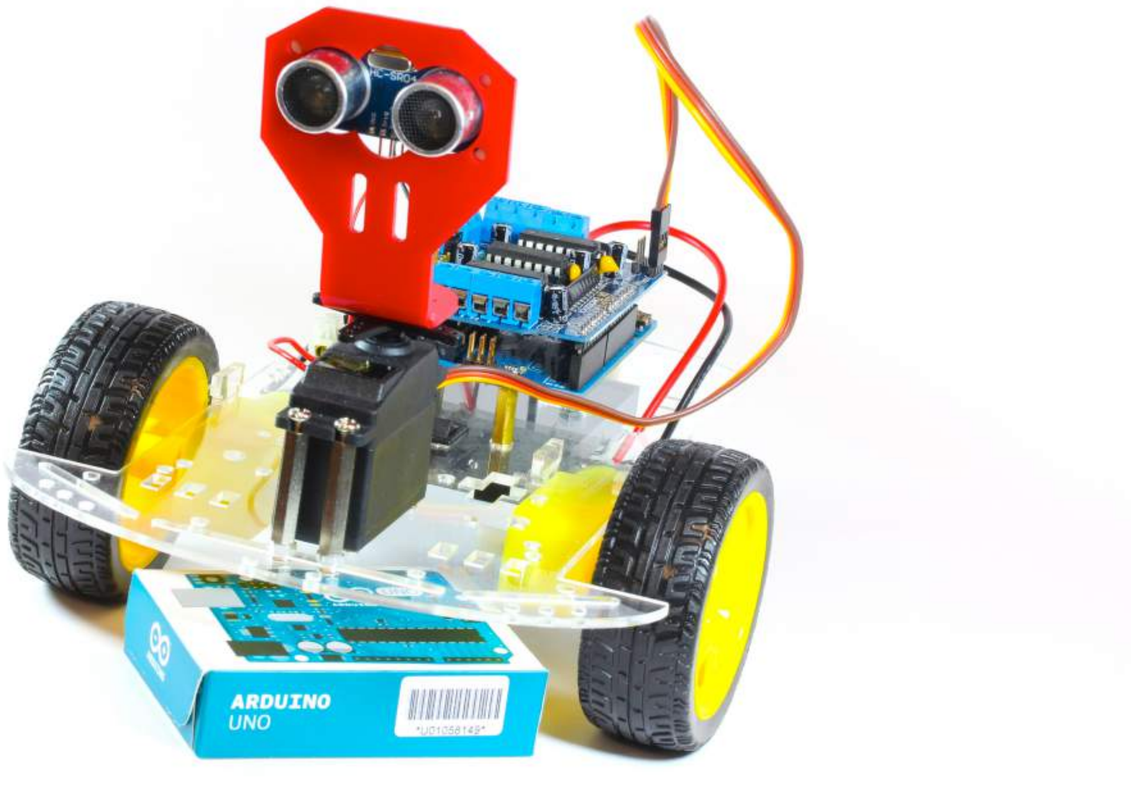
The oddWires Robotics Kit for Arduino with Motor Shield is designed to be built and used by individuals, educational institutions and anyone who wants to learn about robotics. It offers fantastic value for money including a complete, extensible chassis with an Arduino Uno to control and manage the robot.

If you want the version where you do more with LEDs, MOSFETs and so on see the oddWires Robotics Kit for Arduino. In that version you build a robot with the same chassis, but you develop a motor control board from a prototype board.

The robot itself consists of a chassis onto which you will mount an Arduino Uno together with a stackable Arduino motor shield that comes assembled. You do need to solder some pin headers to the board. This shield has terminals for the two motors, a terminal for the servo and a terminal for the 6V battery holder that will supply the power for the motors, the servo and the Arduino Uno. The connections from the ultrasonic distance sensor will be made to pins on the motor shield.

Once the motor shield is functioning, you can then add a servo-mounted ultrasonic distance sensor to enable the robot to maneuver using its own logic.

Sketches are supplied at several levels from introductory to a complete robot controller. All of these sketches are extensible and you can add all sorts of additional sensors and enhance the initial robot behavior.



oddWires Robotics Kits include a genuine Arduino Uno R3

Robot Construction Steps

- Check your kit contents list and ensure you have all components.
- Install the Arduino IDE and, if necessary USB drivers.
- Construct the chassis and the 6V power supply.
- Install the Arduino Uno and the motor shield.
- Add the servo and the ultrasonic distance sensor (there are separate sketches to test each step).
- Load the final sketch and test your completed robot.

Kit Contents

Chassis	Quantity
Paper-wrapped Acrylic Chassis Base	1
Motors with Leads	2
Wheels	2
4 x AA Battery Case	1
Hardware Pack 1	1
Hardware Pack 2	1
On/off switch	1
Arduino Uno	
Arduino Uno R3	1
USB cable	1
Motor Driver Board	
Motor Shield	1
Connecting Wire (2 colors)	2
Inline Terminal Block	1
Servo	
Servo with Lead and Accessories	1
20 way M-M Split-able Ribbon Cable (split off what you need for testing)	1
Breadboard (for testing purposes)	1
Ultrasonic Sensor	
Ultrasonic Distance Sensor Module HC-SR04	1
40 pin header (break up with snips for the size you need)	1
Ultrasonic Distance Sensor Mount	1
1 x 4 way ribbon cable F-F or 4 F-F wires	1
20 way F-M Split-able Ribbon Cable (split off what you need for testing)	1

Hardware Pack Contents

Part List	Quantity
Motor Mounts	2
Rotors	2
M3 x 30mm Screws, Pan-Headed	4
M3 x 8mm Screws, Pan-Headed	9
M3 Nuts	13
M3 x 12mm Female-Female Stand-Offs	4
M3 x 25mm Female-Female Stand-Offs	2
M3 x 15mm Female-Female Stand-Offs	4
M3 x 6mm Screws, Pan-Headed	14
On/Off Switch	1
Castor Wheel	1

These parts are used as follows:

Castor Wheel	
M3 x 15mm Female-Female Stand-Offs	4
M3 x 6mm Screws, Flat-Headed	4
M3 x 8mm Screws, Pan-Headed	4

Arduino Uno	
M3 x 10mm Female-Female Stand-Offs	4
M3 x 6mm Screws, Pan-Headed (only two stand-offs are screwed to chassis base)	6

Motors	
Motor Mounts	2
Rotors	2
M3 x 30mm Screws, Pan-Headed	4
M3 Nuts	4

Servo	
M3 x 25mm Female-Female Stand-Offs	2
M3 x 8mm Screws, Pan-Headed	2
M3 x 6mm Screws, Pan-Headed	2
M3 Nuts	2

6V 4 x AA Battery Case	
M3 x 8mm Screws, Pan-Headed	2
M3 Nuts	2

Note: hardware may be in one pack or the following may be in a separate pack:

M3 x 25mm Female-Female Stand-Offs	2
M3 x 15mm Female-Female Stand-Offs	4
M3 x 6mm Screws, Pan-Headed	14

Tools and materials required for assembly

Soldering iron
Solder
Pliers
Wire snips & wire stripper
Cross-Head Screwdriver

Install the Arduino IDE and USB Drivers

This text is based on the Getting Started text from the official Arduino site under a Creative Commons Attribution-ShareAlike 3.0 License.

Getting Started with Arduino on Windows

Download the Arduino environment

Get the latest version from the download page (http://arduino.cc/en/Main/Software).

When the download finishes, unzip the downloaded file. Make sure to preserve the folder structure. Double-click the folder to open it. There should be a few files and sub-folders inside.

Connect the board

The Arduino Uno automatically draws power from either the USB connection to the computer or an external power supply.

Connect the Arduino board to your computer using the USB cable. The green power LED (labeled **On**) should light.

Install the drivers

Installing drivers for the Arduino Uno with Windows8, Windows7, Vista, or XP:

- Plug in your board and wait for Windows to begin its driver installation process. After a few moments, the process will fail, despite its best efforts
- Click on the Start Menu, and open up the Control Panel.
- While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager.
- Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)"
- Right click on the "Arduino UNO (COMxx)" port and choose the "Update Driver Software" option.
- Next, choose the "Browse my computer for Driver software" option.
- Finally, navigate to and select the driver file named **"arduino.inf"**, located in the "Drivers" folder of the Arduino Software download (not the "FTDI USB Drivers" sub-directory). If you are using an old version of the IDE (1.0.3 or older), choose the Uno's driver file named **"Arduino UNO.inf"**
- Windows will finish up the driver installation from there.

When you connect the board, Windows should initiate the driver installation process (if you haven't used the computer with an Arduino board before).

On Windows 8, 7 or Vista, the driver should be automatically downloaded and installed. You can check that the drivers have been installed by opening the Windows Device Manager (in the Hardware tab of System control panel). Look for a "USB Serial Port" in the Ports section; that's the Arduino board.

Launch the Arduino application

Double-click the Arduino application. (Note: if the Arduino software loads in the wrong language, you can change it in the preferences dialog. See the environment page (http://arduino.cc/en/Guide/Environment#languages) for details.)

Getting Started with Arduino on Mac

Download the Arduino environment

Get the latest version from the download page (http://arduino.cc/en/Main/Software). When the download is finished, double click the .zip file. This will expand the Arduino application.

Install the Software

Copy the Arduino application into the Applications folder (or elsewhere on your computer). No drivers are required to be installed.

Connect the board

The Arduino Uno automatically draws power from either the USB connection to the computer or an external power supply. Connect the Arduino board to your computer using the USB cable. The green power LED (labeled **On**) should go on.

A dialog box will appear telling you that a new network interface has been detected. Click "Network Preferences...", and when it opens, simply click "Apply". The Uno or Mega 2560 will show up as "Not Configured", but it's working properly. Quit System Preferences.

Launch the Arduino application

Double-click the Arduino application. **Note:** if the Arduino software loads in the wrong language, you can change it in the preferences dialog. See the environment page for details (http://arduino.cc/en/Guide/Environment#languages)

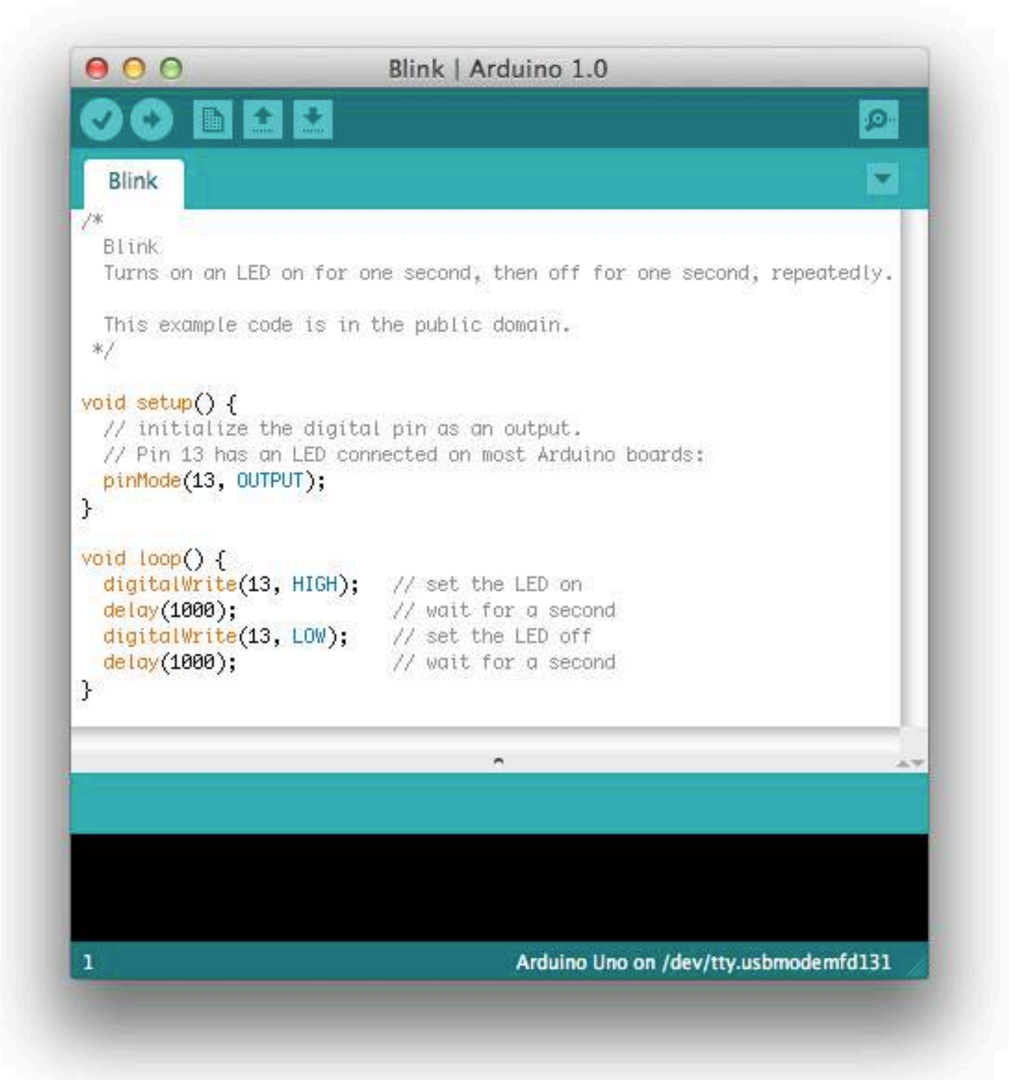
Getting Started with Arduino on Linux

Getting Started on Linux depends on your particular distribution. Details can be found here (http://playground.arduino.cc/Learning/Linux).

Your First Arduino Sketch

Open the blink example

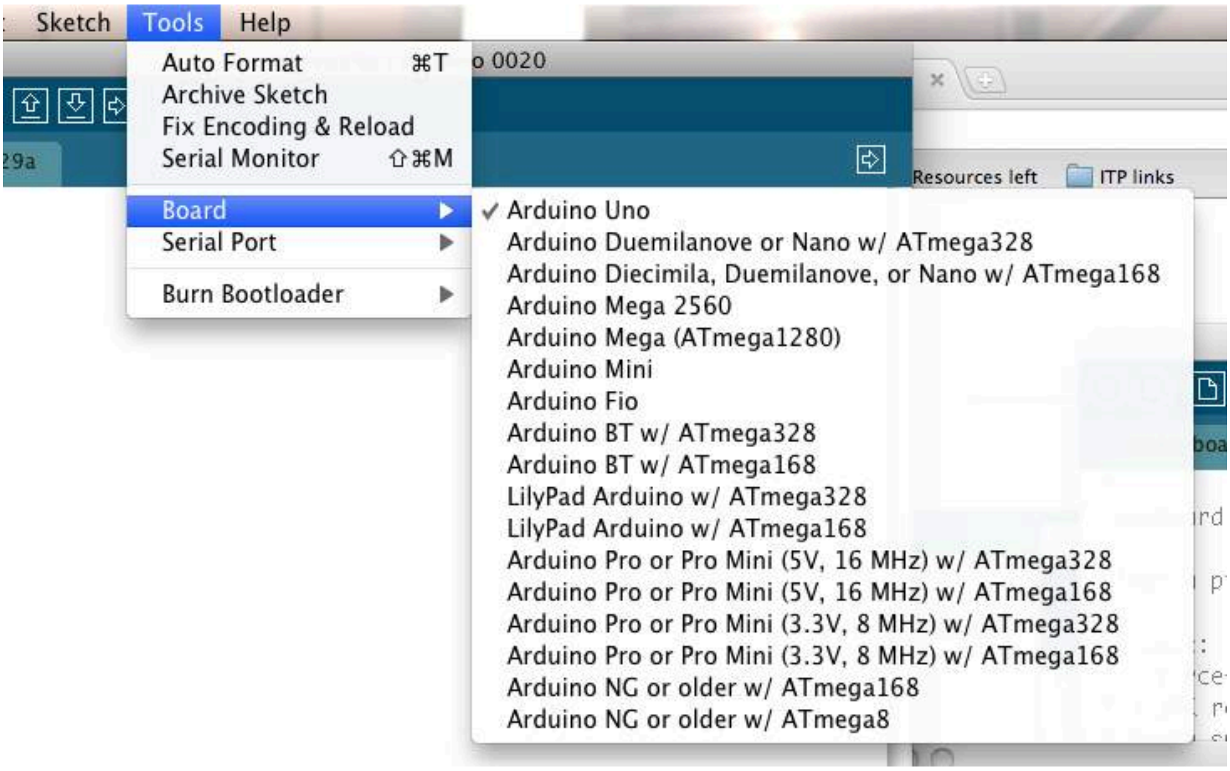
Open the LED blink example sketch: File > Examples > 1.Basics > Blink.



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

Select your board

You'll need to select the entry in the **Tools > Board** menu that corresponds to your Arduino.


Sketch
Tools
Help
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Serial Monitor ⇧⌘M
Board
Serial Port
Burn Bootloader
o 0020
Resources left
ITP links
✓ Arduino Uno
Arduino Duemilanove or Nano w/ ATmega328
Arduino Diecimila, Duemilanove, or Nano w/ ATmega168
Arduino Mega 2560
Arduino Mega (ATmega1280)
Arduino Mini
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8

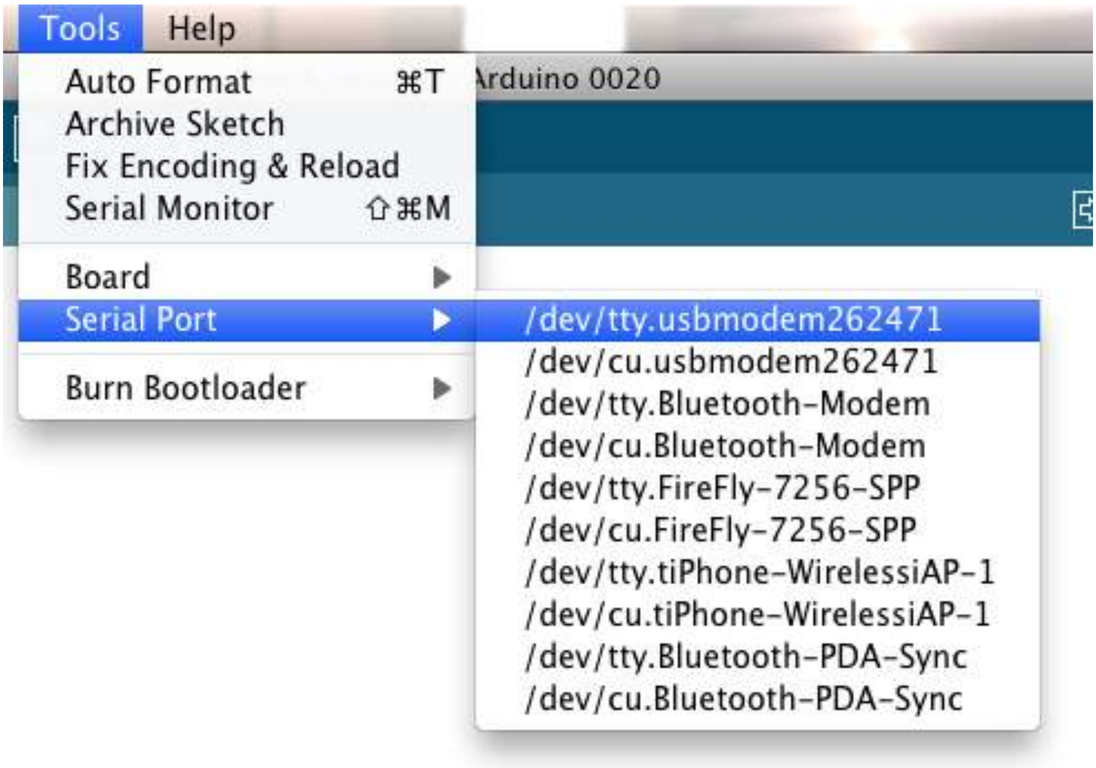
Selecting an Arduino Uno

Select your serial port

Select the serial device of the Arduino board from the **Tools > Serial Port** menu.

Select the serial device of the Arduino board from the Tools | Serial Port menu. On Windows this is likely to be **COM3** or higher (**COM1** and **COM2** are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port.

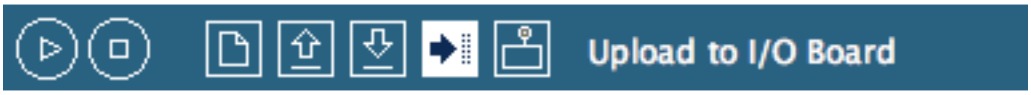
On the Mac, this should be something with **/dev/tty.usbmodem** (for the Uno).



Selecting an Uno

Upload the program

Now, simply click the "Upload" button in the environment. Wait a few seconds - you should see the RX and TX LEDs on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar.



A few seconds after the upload finishes, you should see the pin 13 (L) LED on the board start to blink. Congratulations! You've got Arduino up-and-running.

If you have problems, please see the troubleshooting suggestions http://arduino.cc/en/Guide/Troubleshooting).

Sweeping the Servo by 180 Degrees

In this section we learn how to use a servo with Arduino. We will sweep the shaft of a RC servomotor back and forth across 180 degrees.

This example makes use of the Arduino **servo library**.

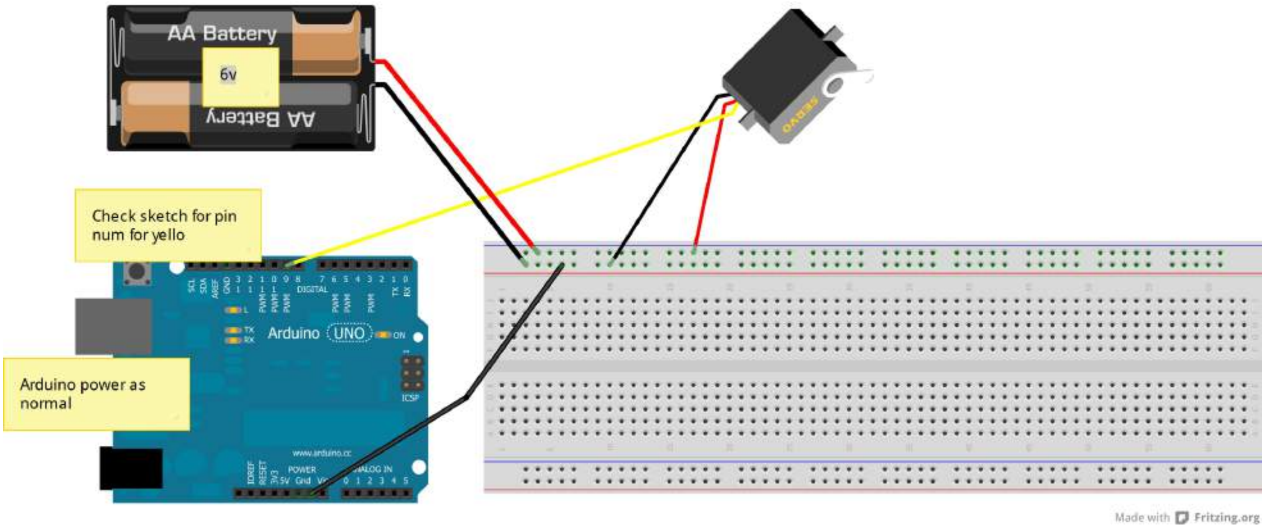
Hardware Required

- Arduino Board
- (1) Servo Motor
- Jumper wire

Circuit

Servomotors have three wires: power, ground, and signal. The power wire is typically red, and should be connected to the six-volt power supply. The ground wire is typically black or brown and should be connected to a ground pin on the Arduino board AND the 6V power supply. It is essential that there is a common ground. The signal pin is typically yellow, orange or white and should be connected to pin 9 on the Arduino board.

Click the image to enlarge



Check sketch for pin num for yello

Arduino power as normal

Image developed using Fritzing. For more circuit examples, see the Fritzing project page

Code

```
// Sweep
// by BARRAGAN <http://barraganstudio.com>
// this example code is in the public domain.


#include <Servo.h>

Servo myservo;  // create servo object to control a servo
                // a maximum of eight servo objects can be created

int pos = 0;    // variable to store the servo position

void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}


void loop()
{
  for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to 180 degrees
  {                                  // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
  for(pos = 180; pos>=1; pos-=1)     // goes from 180 degrees to 0 degrees
  {
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
}
```

HC-SR04 Ultrasonic Distance Sensor

The HC-SR04 is an ultrasonic distance sensor. It detects the distance of the closest object in front of the sensor (from 2 cm up to 3m). It works by sending out a burst of ultrasound and listening for the echo when it bounces off of an object. The Arduino board sends a short pulse to trigger the detection, then listens for a pulse on the echo pin. The duration of this second pulse is equal to the time taken by the ultrasound to travel to the object and back to the sensor. Using the speed of sound, this time can be converted to distance.

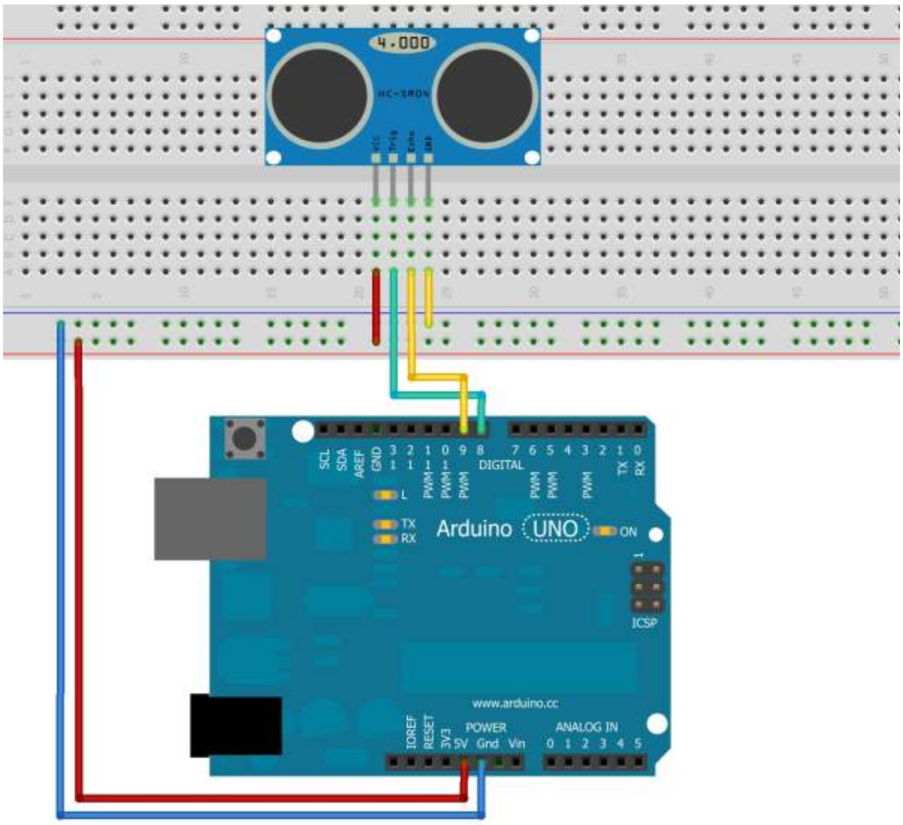
Hardware Required

- Arduino Board
- (1) HC-SR04 Ultrasonic Distance Sensor
- Jumper wire

Circuit

The 5V pin of the HC-SR04 is connected to the 5V pin on the Arduino, the GND pin is connected to the GND pin, and the Trigger pin is connected to pin 8 and the Echo pin is connected to pin 9 on the Arduino.

Image developed using Fritzing. For more circuit examples, see the Fritzing project page



Code

```
/*
HC-SR04 for Arduino

Original project from http://www.swanrobotics.com

This project demonstrates the HC-SR
The distance presented in the code is in mm, but you can uncomment the line
for distance in inches.
The schematics for this project can be found on http://www.swanrobotics.com

This example code is in the public domain.
*/

const int TriggerPin = 8;      //Trig pin
const int EchoPin = 9;         //Echo pin
long Duration = 0;

void setup(){
  pinMode(TriggerPin,OUTPUT);  // Trigger is an output pin
  pinMode(EchoPin,INPUT);      // Echo is an input pin
  Serial.begin(9600);          // Serial Output
}

void loop(){
  digitalWrite(TriggerPin, LOW);
  delayMicroseconds(2);
  digitalWrite(TriggerPin, HIGH);          // Trigger pin to HIGH
  delayMicroseconds(10);                   // 10us high
  digitalWrite(TriggerPin, LOW);           // Trigger pin to HIGH

  Duration = pulseIn(EchoPin,HIGH);        // Waits for the echo pin to get
high
                                           // returns the Duration in
microseconds
  long Distance_mm = Distance(Duration);   // Use function to calculate the
distance

  Serial.print("Distance = ");             // Output to serial
  Serial.print(Distance_mm);
  Serial.println(" mm");

  delay(1000);                             // Wait to do next measurement
}

long Distance(long time)
{
    // Calculates the Distance in mm
    // ((time)*(Speed of sound))/ toward and backward of object) * 10

    long DistanceCalc;                      // Calculation variable
    DistanceCalc = ((time /2.9) / 2);     // Actual calculation in mm
    //DistanceCalc = time / 74 / 2;         // Actual calculation in inches
    return DistanceCalc;                    // return calculated value
}
```

Constructing the Chassis

Identify the chassis components:

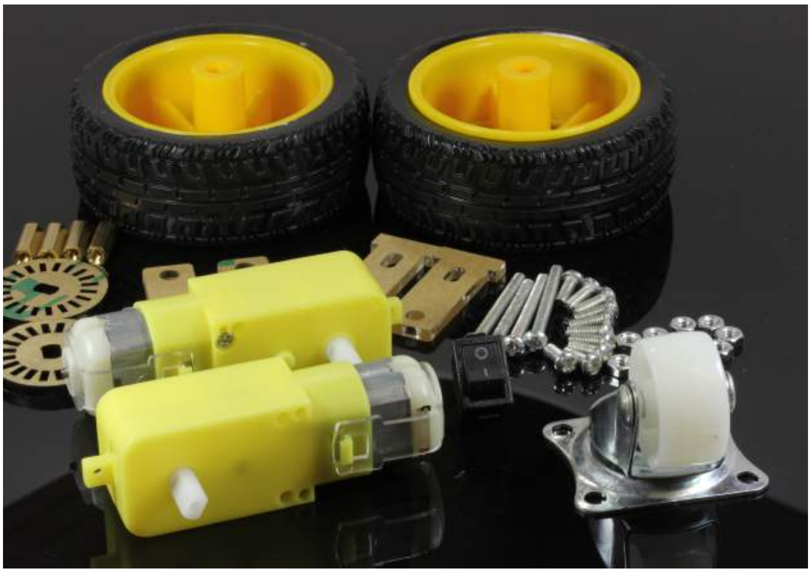
1 x chassis plate

2 x wheels

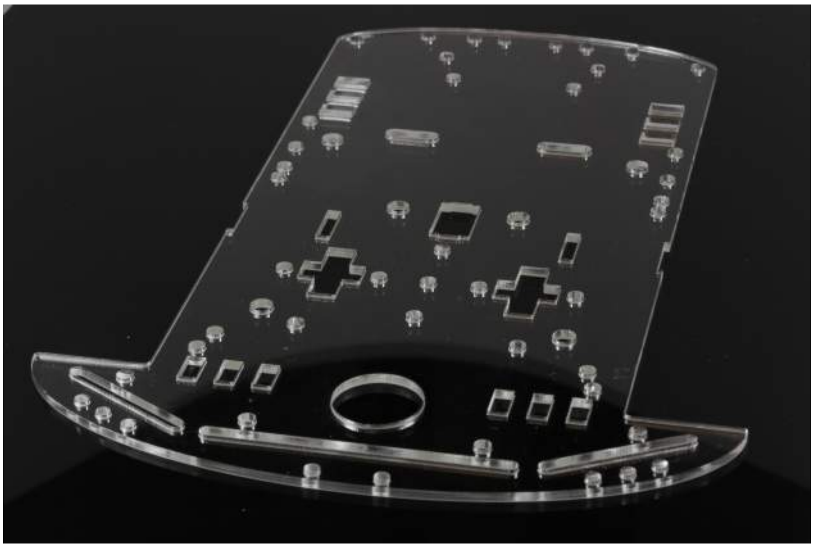
2 x motors

1 x castor wheel

1 x hardware pack 1

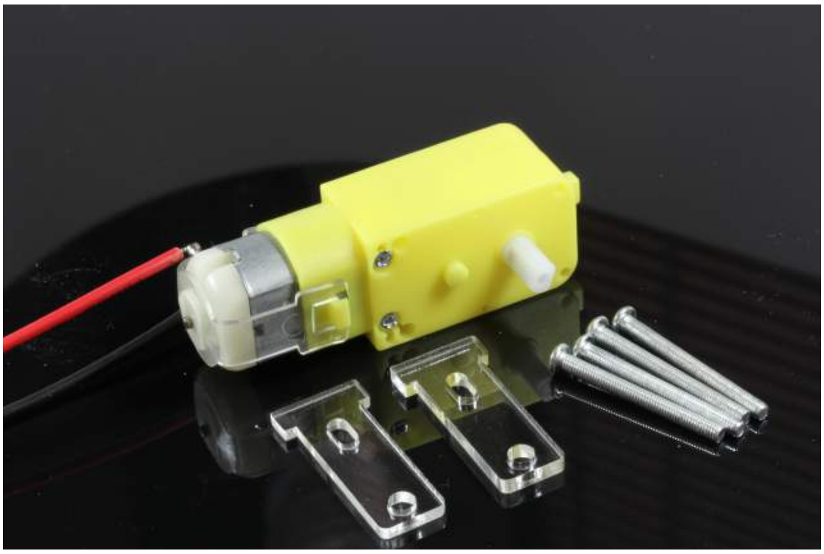


Here's the chassis plate with its protective paper removed.

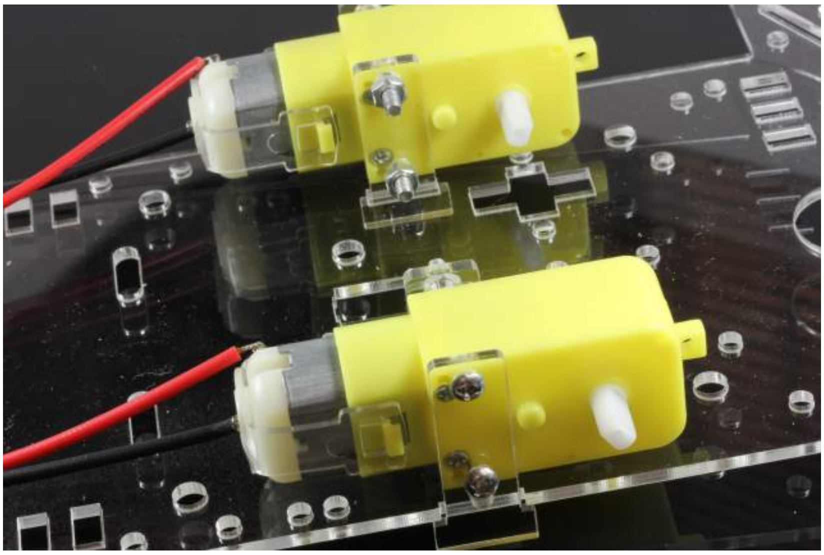


Here's the motor, mounting plates and two long machine screws & two nuts. Solder the black & red leads to the motors as shown.

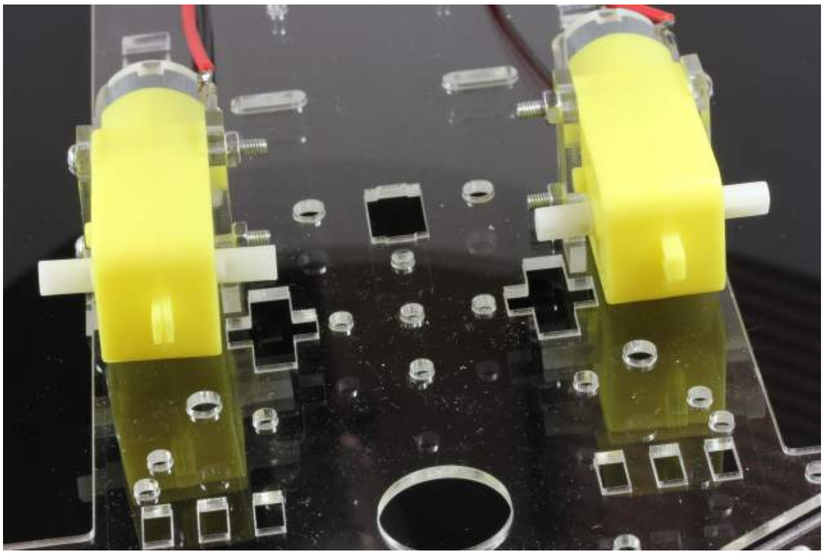
Slip the two mounts into the positions shown in the next photo. The mounts should be positioned so that the piece that will hold the motors to the base will be on the top of the finished chassis. Loosely tighten the nuts until you have the motors aligned, and then tighten. You should not over-tighten anything, as the base is acrylic not metal!



Both of the motors attached with mounts (note mirror image). Keep the screw heads on the outside to avoid the wheels being fouled by the screw ends.



Here's another view of the motors installed on the chassis base.

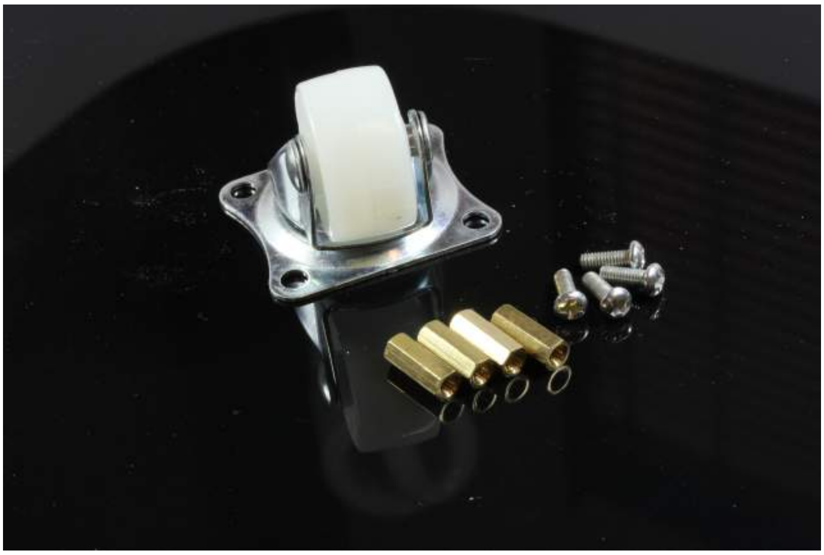


Mounting the castor wheel requires:

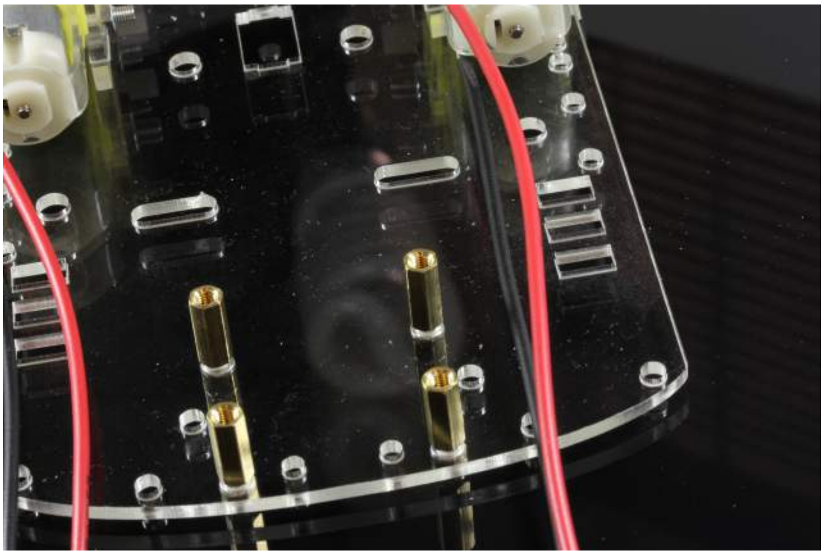
4 x 15 mm standoffs

4 x pan-head screws to attach the standoffs to the base

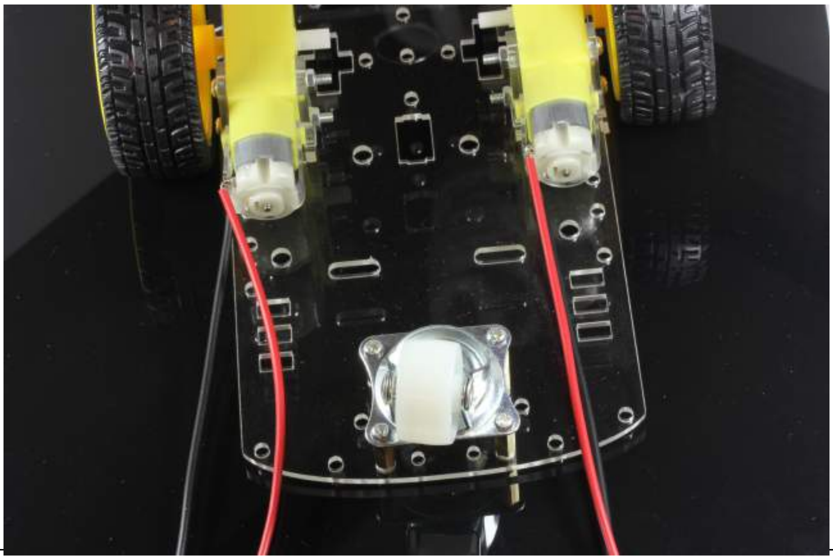
4 x pan-head screws to attach the castor to the mounts



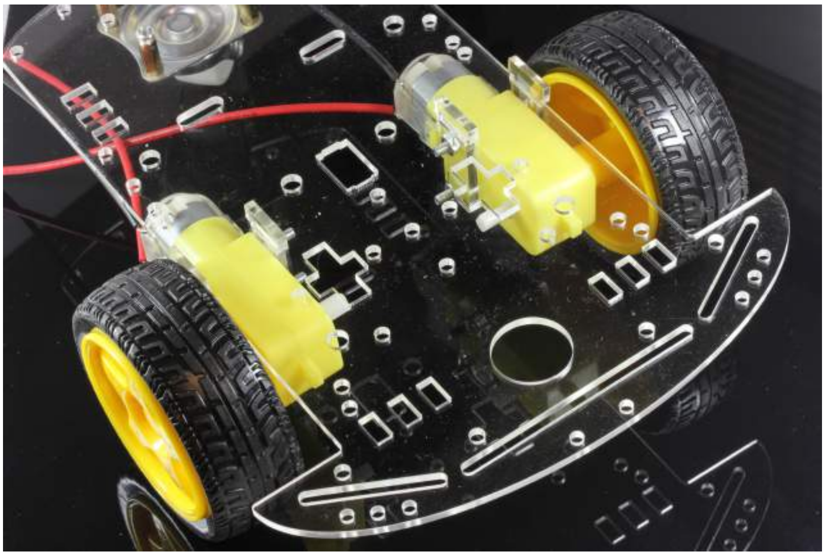
Screw the stand-offs to the chassis with the pan-head screws.



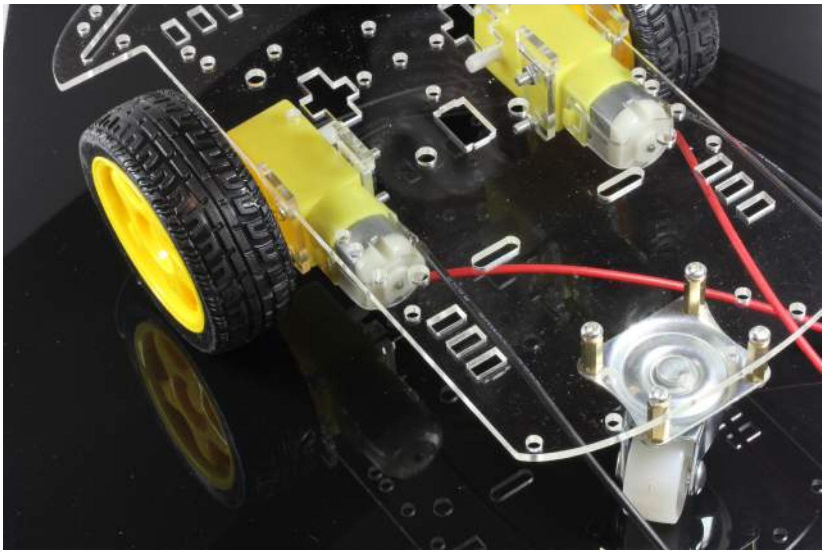
Mount the castor wheel with more screws.



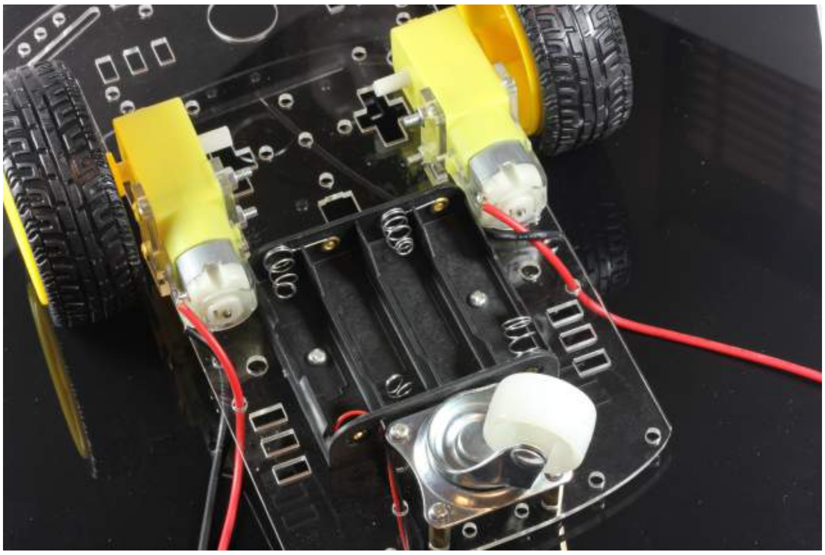
Here is the assembled chassis.



Another view.

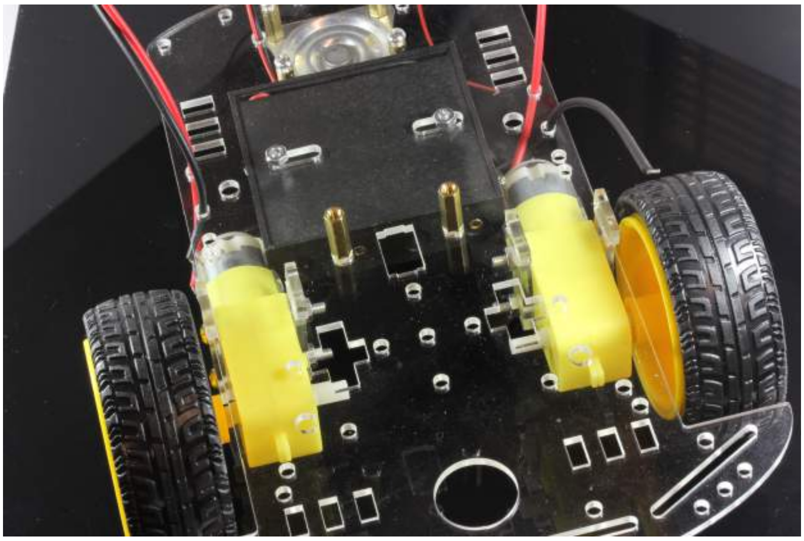


Battery box installed underneath.

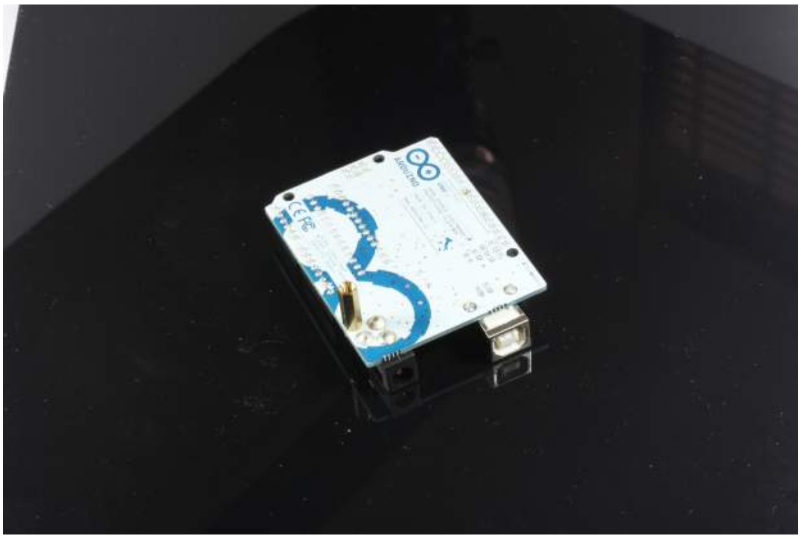


Installing the Arduino Uno

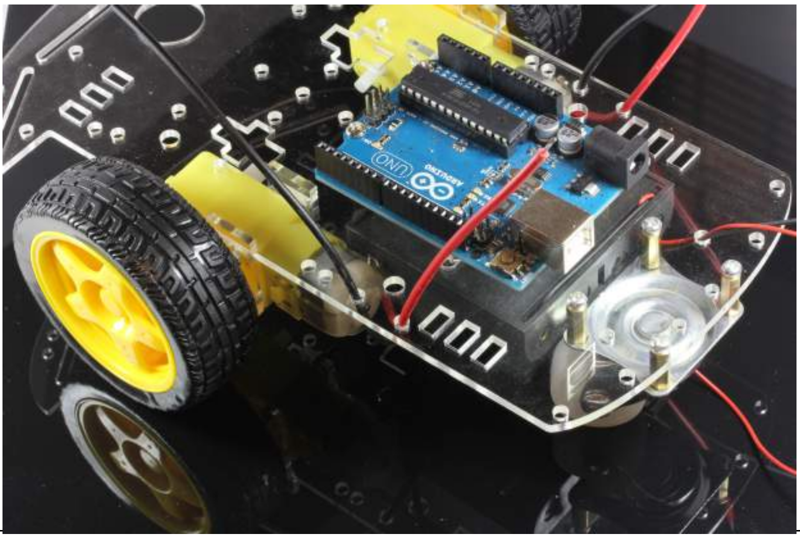
Insert two of the smaller standoffs from Hardware Pack 2 in the position shown in the photo and fix with a couple of 6mm M3 screws.



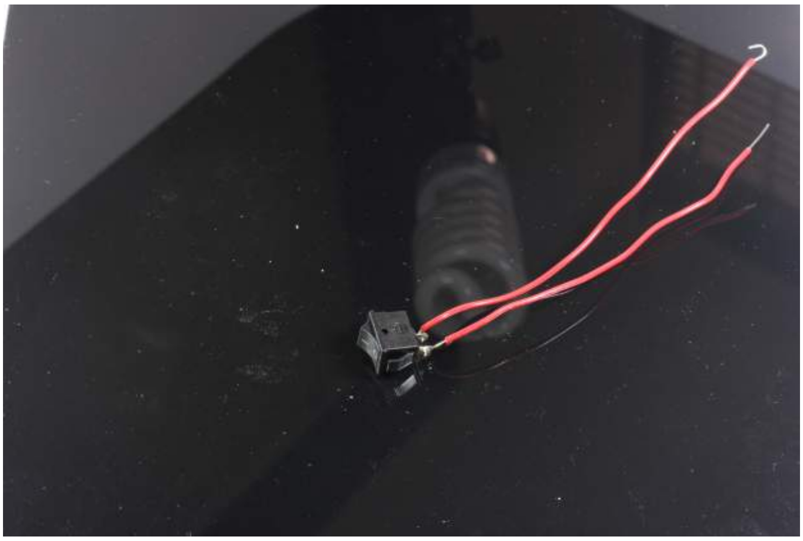
Add a standoff to the Uno in the position shown.



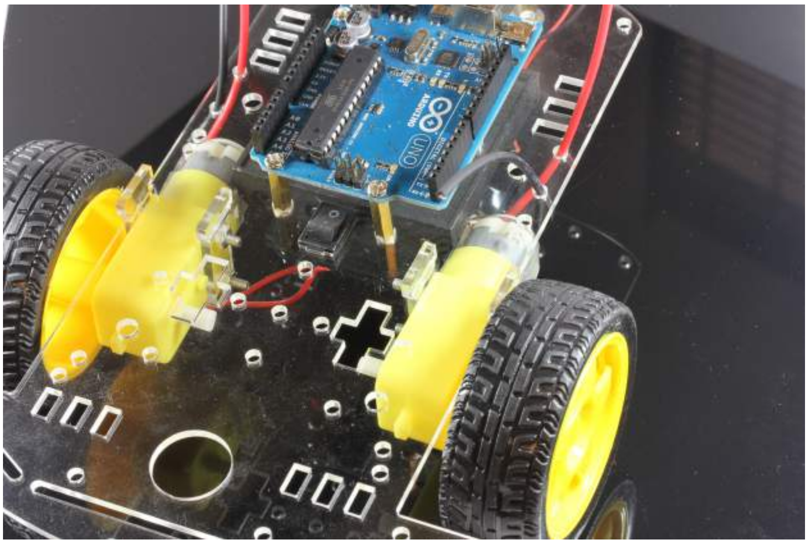
Then fix the Uno into position using another two M3 screws.



Solder two red wires cut from the supplied connecting wire to the switch.



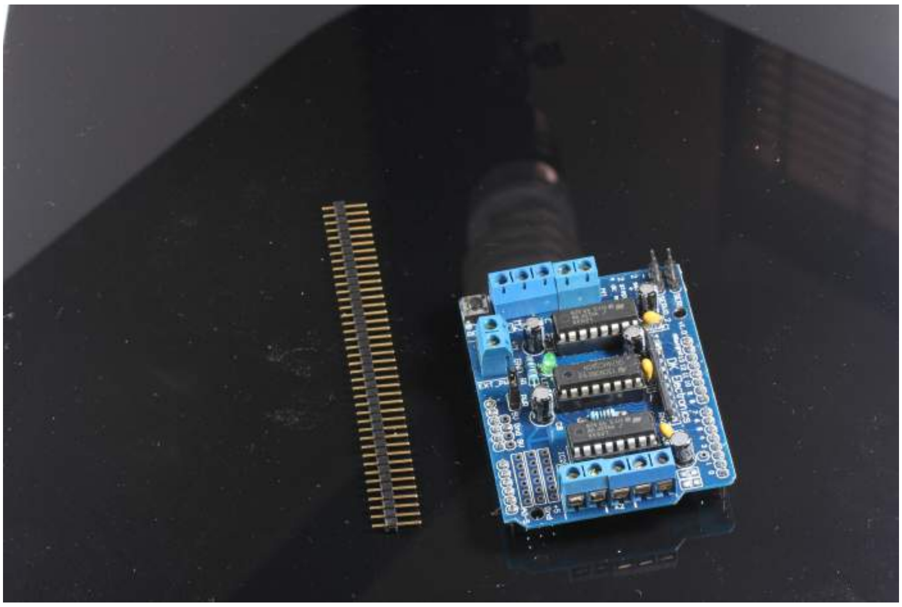
Snap the switch into place just in front of the Uno.



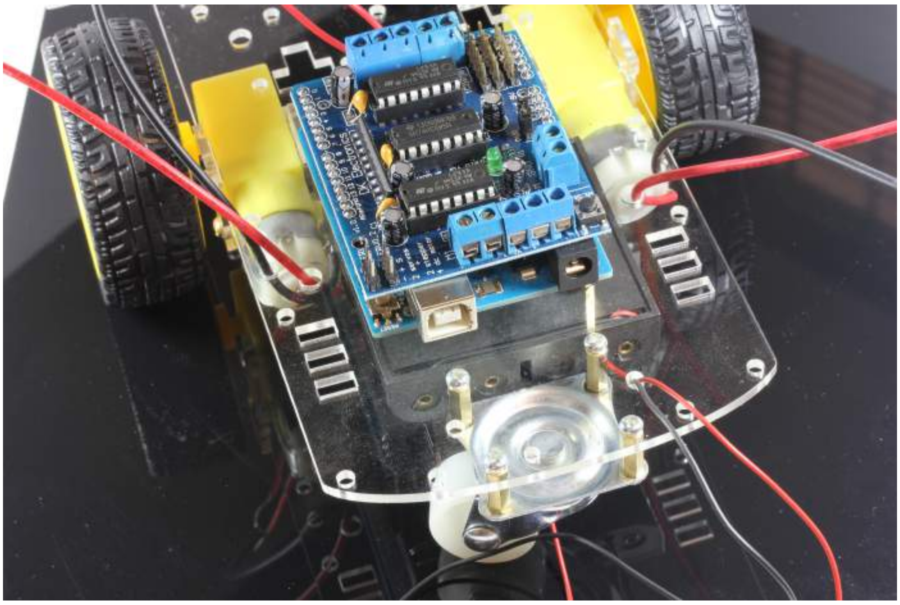
Installing the Motor Shield

Snip off 3 x 6 pin strips from the 40 pin header (tip: hold onto the small piece when you snip so it doesn't go flying off).

Solder the three strips to the bottom right hand corner of the shield where the rows are labeled A0-A5, GND & 5V. You could simply solder the connecting wires later but this way you get to easily use these pins.

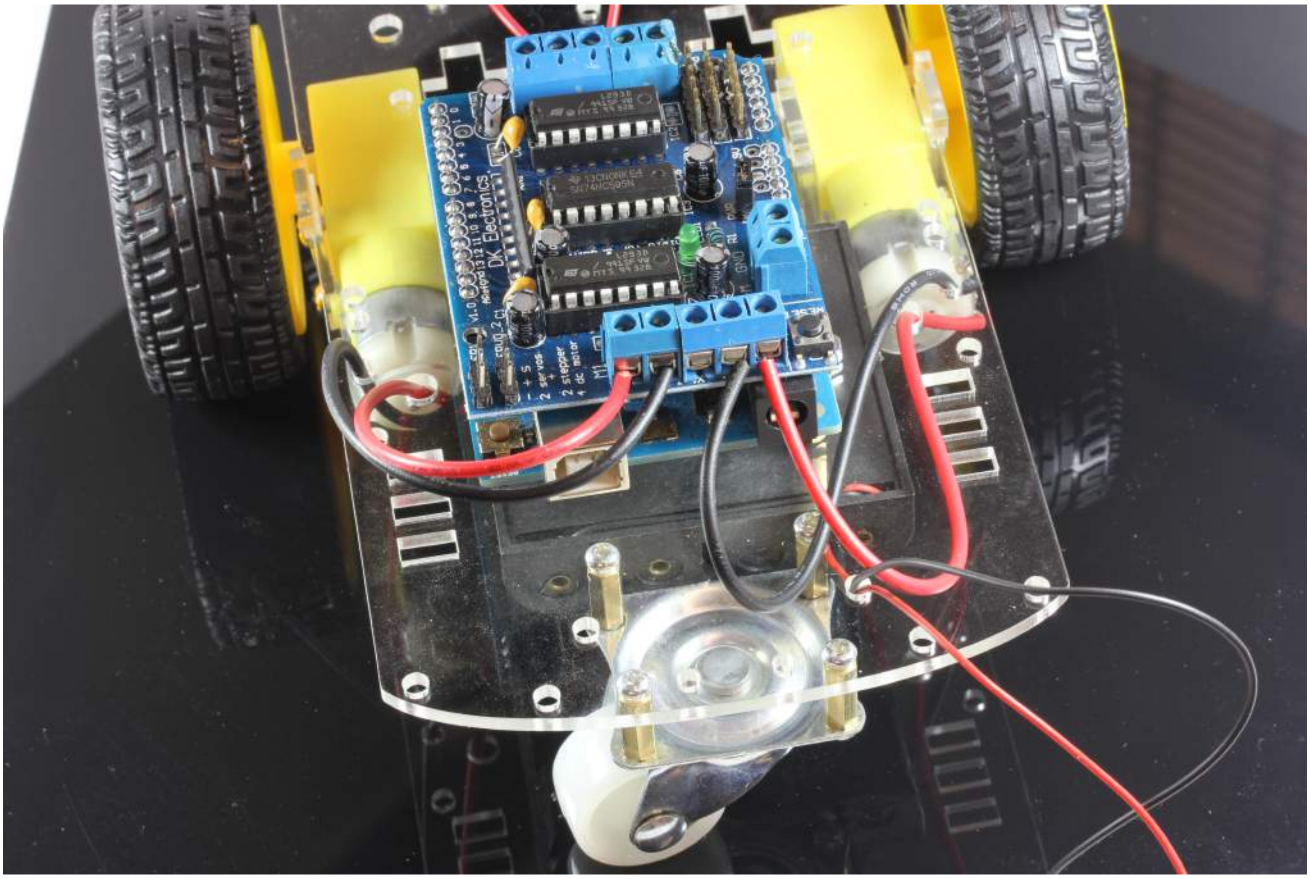


Here's the motor shield installed on top of the Un. Ensure you leave the shield evenly positioned. The USB & Power connections are a different height; fully inserting can stress the motor shield board.

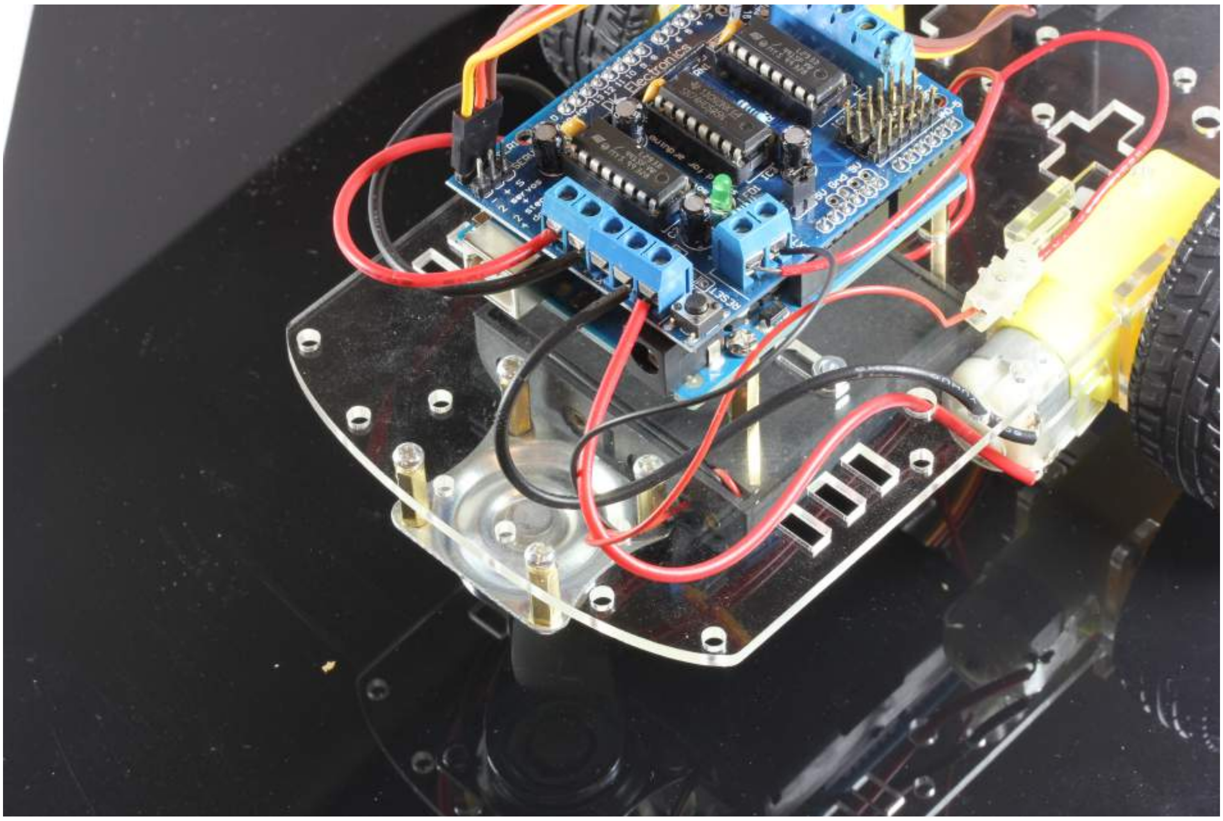


Very important: do NOT connect the power to the motor shield incorrectly or you could damage the board.

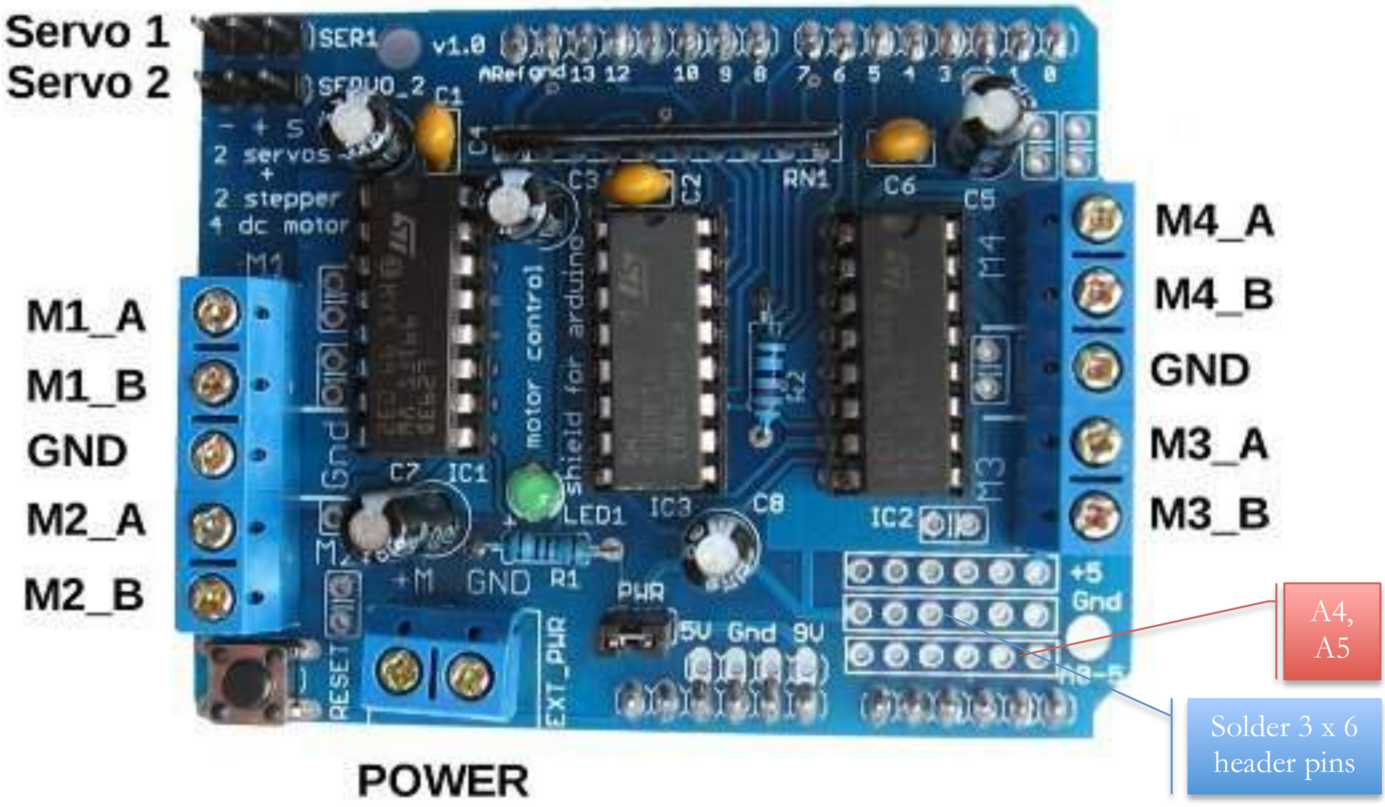
1. Connect Motor 1 to M1_A & M1_B. Connect the Red from first motor to M1_A & the Black to M1_B.
2. Connect Motor 2 to M2. Connect the Red from second motor to M2_B & the Black to M2_A. Note the reversal – each motor must run the opposite to the other otherwise the robot will go arund in circles.



3. Connect GND (black wire) of battery box to GND of Power Terminals (EXT_PWR) on Motor Shield.
4. Install on/off switch using inline connector (battery box red wire to on switch, other red wire from switch to +M on shield).

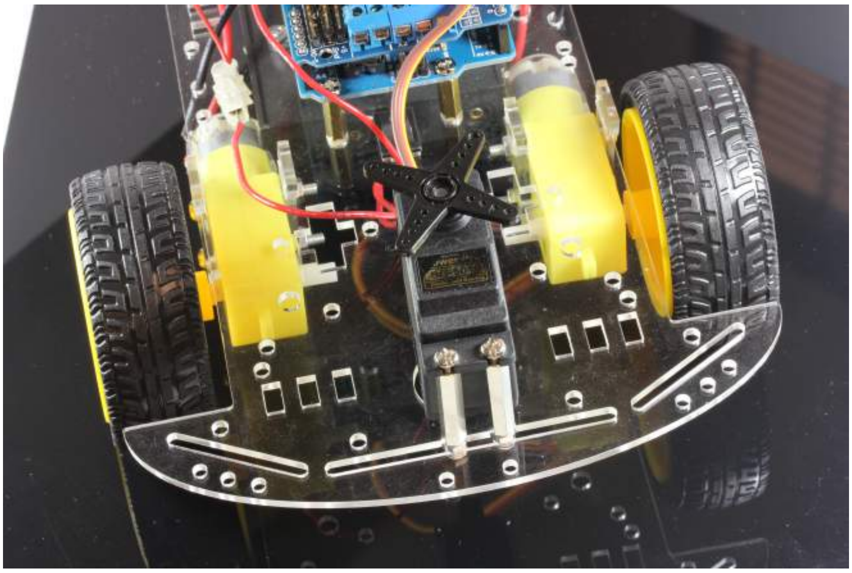


5. Connect Servo to SER1 (Brown to -, Red to +, Yellow to S)
6. Connect Ultrasonic Sensor to the A4, A5 +5V, and GND on the motor shield. You will need to solder header pins into these holes to be able to connect the ultrasonic pins to the motor shield.

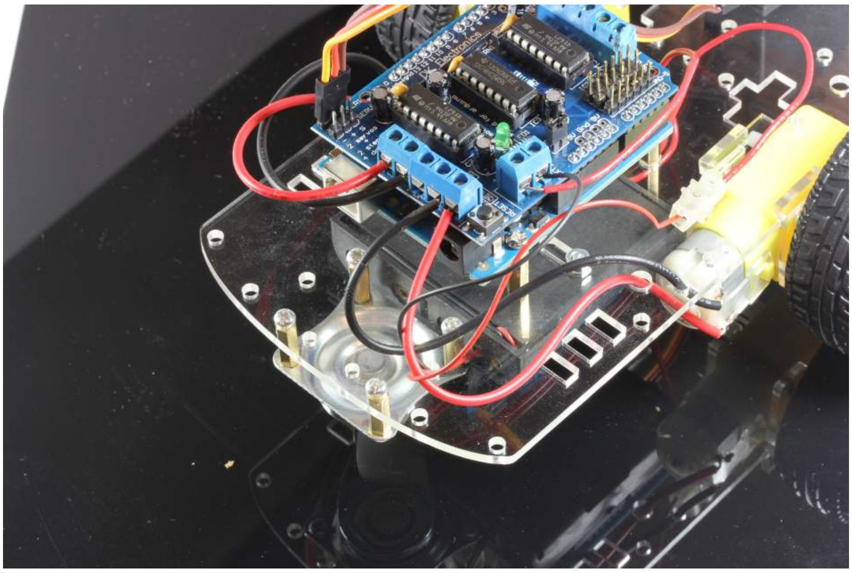


Installing the Servo

Servo attachment. Note the nut below the standoff as a spacer.



Servo connected to Servo 1.



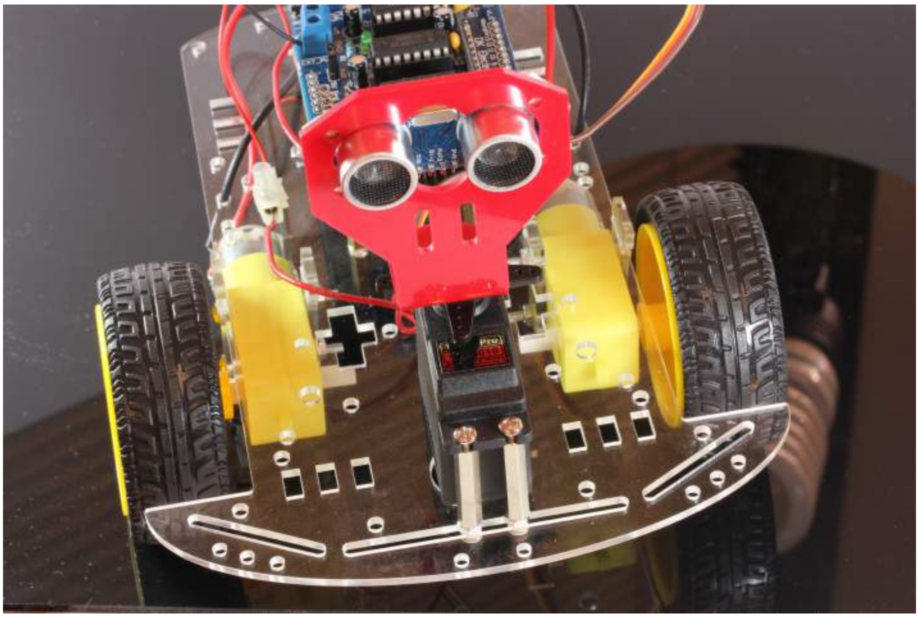
Brown	GND
Red	Power
Orange	Signal

Installing the Ultrasonic Distance Sensor

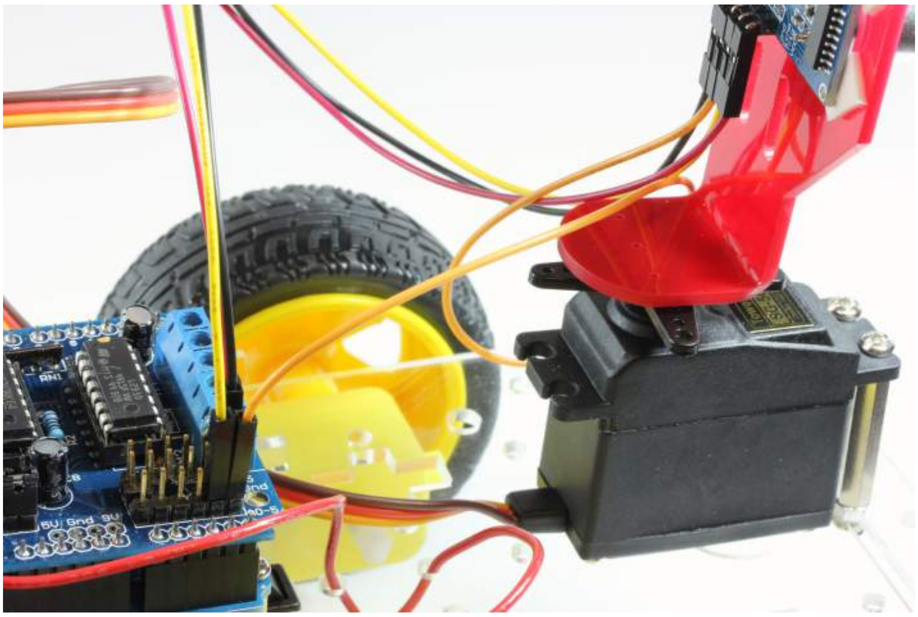
Use the female-female connecting wires to connect the Ultrasonic sensor as follows:

Sensor	**Arduino**
Vcc	Arduino +5V
Trig	Arduino Pin A4
Echo	Arduino Pin A5
GND	Arduino GND

The ultrasonic distance sensor mount is attached to the servo mount and the mount is connected to the servo using double-sided tape – you can use contact cement or superglue for more permanence.



Here are the connections from the HC-SR04 Ultrasonic Distance Sensor to the Motor Shield.



Final Construction and Testing

There are several sketches supplied that may be helpful in testing. In addition, there is a completed sketch. You will be able to enhance this sketch as you develop your robot car further. You can find the download on the oddWires site for the kit.

Note: Remember to download the AFMotor and NewPing libraries. These are used in the supplied sketch and you will get a lot of error message have not installed these libraries!

Ideas for Extending your Robot

Here are a few ideas for extending your Arduino-based robot.

LDR (moth)

Use a Light Dependent Resistor as part of a voltage divider to sense the light. Read the values from an Arduino analog pin and move the robot to the source of light.

PIR Sensor (movement sensor)

Use a PIR sensor to detect movement. Have it chase a person moving around the room.

IR Remote Control

Use a TSOP3848 IR receiver in conjunction with a remote control (or build your own with a TSAL7400 IR LED and an Arduino) to remotely control your robot.