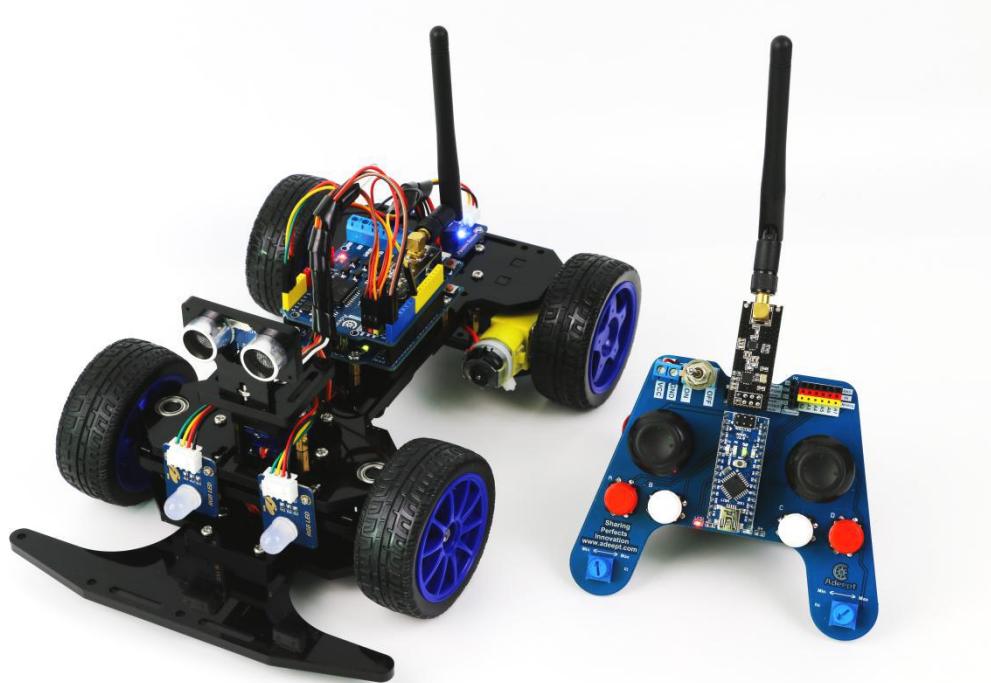


Adeept

www.adeept.com

Adeept Intelligent Remote Control Car Kit for Arduino



Warning

Please pay attention to the following issues when purchasing or using the product:

- ★ There are small components included in this kit. Swallowing mistakenly or misoperation can cause serious infection and be even fatal. When an accident occurs, please seek medical assistance immediately.
- ★ Please place the product in a safe place where an under-3-year-old cannot touch, who should not use or approach the product.
- ★ Juveniles should use the product with their parents.
- ★ Do not place the product or the components near any AC socket or other circuits, in case of potential risks of electric shock.
- ★ Do not use the product near any liquid or flame.
- ★ Do not use or store the product in an extreme environment such as extremely cold or hot and heavily humid.
- ★ Please remember to power off when the product is not in use.
- ★ Do not touch the moving or rotating part of the product.
- ★ The product may get heat at some part, which is just normal. But misoperation may cause overheat.
- ★ Misoperation may cause damage to the product. Please take care.
- ★ Do not connect the positive and negative poles of the power inversely, or the devices in the circuit may be damaged.
- ★ Please place and put the product gently. Do not smash or shock it.

About

Adeept is a technical service team of open source software and hardware. Dedicated to applying the Internet and the latest industrial technology in open source area, we strive to provide best hardware support and software service for general makers and electronic enthusiasts around the world. We aim to create infinite possibilities with sharing. No matter what field you are in, we can lead you into the electronic world and bring your ideas into reality.

The code and circuits of our product are open source. You can check on our website:

www.adeept.com

If you have any problems, feel free to send an email for technical support and assistance:

support@adeept.com

On weekdays, we usually will reply within 24 hours. Also welcome to post forums on our website.

Copyright

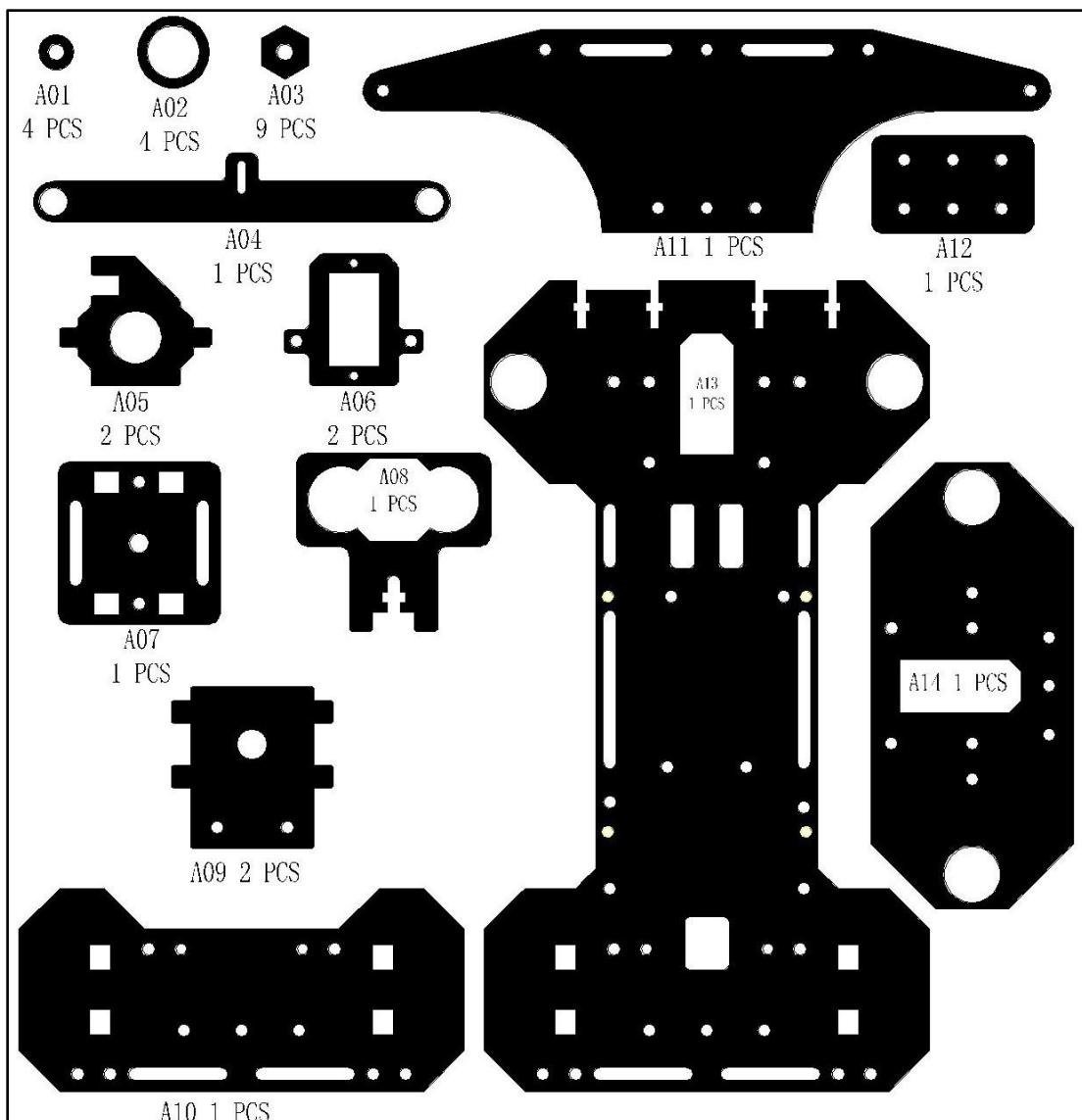
This user manual can be used for learning, DIY, refitting, etc., except for commercial purpose. The Adeept Company owns all rights of contents in the manual, including but not limited to texts, images, data, etc. Any distribution or printing should be implemented with the permission of the Company, or it will be deemed illegal.

contents

components List.....	1
Acrylic Sheets.....	1
Machinery Parts.....	2
Transmission Parts.....	2
Electronic Parts.....	3
Tools.....	4
Self-prepared Parts	4
Software & Hardware.....	5
Introduction.....	5
What is Arduino ?	5
Why Arduino?.....	6
How Should I Use Arduino?.....	6
Arduino Software (IDE).....	8
Install Library.....	11
Upload Program.....	13
Functions.....	18
Schematic Diagram of Adeekt Remote Control Shield V1.0.....	20
Schematic Diagram of Adeekt Motor Shield V1.0.....	21
Assembly.....	22
Fix the Position.....	22
Fix the Battery Holder.....	22
Rear Wheels.....	23
Front Wheels.....	25
Assemble Ultrasonic Module.....	31
RGB LED Module.....	33
Assemble PCBs (Adeekt Uno, NRF24L01 Module.)	34
Assemble the Passive Buzzer Module.....	35
Remote Control	36
Circuit Connection.....	38
Afterword.....	40

Components List

Acrylic Sheets

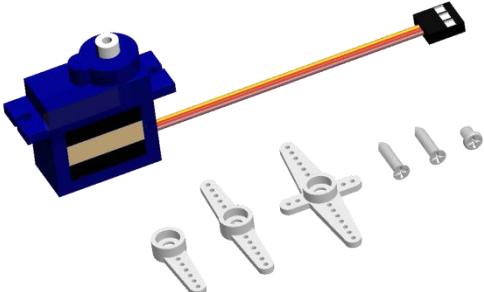
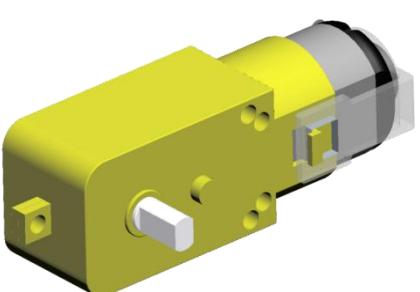


The acrylic sheet is covered with a layer of protective film. You need to remove it first.
Some holes in the acrylic sheets may have residues, so you need to clean them before using it.

Machinery Parts

M2 Nut  X10 www.adeept.com	M3 Nut  X15 www.adeept.com	M4 Nut  X2 www.adeept.com	M2*10 Screw  X10 www.adeept.com	M3*4 Screw  X4 www.adeept.com	M3*8 Screw  X28 www.adeept.com
M3*12 Screw  X5 www.adeept.com	M3*30 Screw  X4 www.adeept.com	M4*40 Screw  X2 www.adeept.com	M3*10 Countersunk Head Screw  X4 www.adeept.com	M1.4*6 Self-tapping Screw  X4 www.adeept.com	M3*6 Copper Standoff  X4 www.adeept.com
M3*12 Copper Standoff  X4 www.adeept.com	M3*30 Copper Standoff  X8 www.adeept.com	F624ZZ Bearing  X4 www.adeept.com	F687ZZ Bearing  X4 www.adeept.com	M4 Spring Washer  X8 www.adeept.com	

Transmission Parts

Servo X2 	DC Motor X2 
Front Wheel X2 	Rear Wheel X2 

Electronic Parts

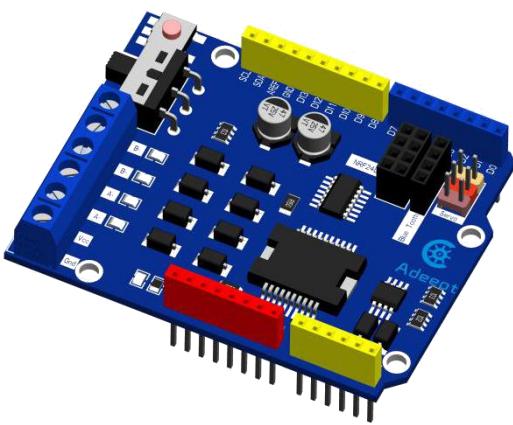
Adeekt UNO R3 X1



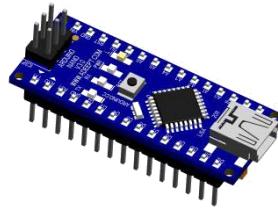
Adeekt Remote Control Shield V1.0 X1



Adeekt Motor ShieldV1.0 X1



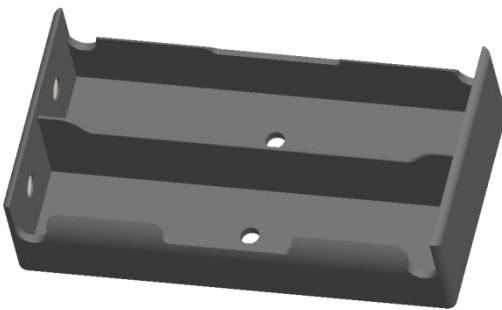
Arduino Nano X1



HC-SR04 Ultrasonic Module X1



18650X2 Battery Holder X2



Passive Buzzer X1

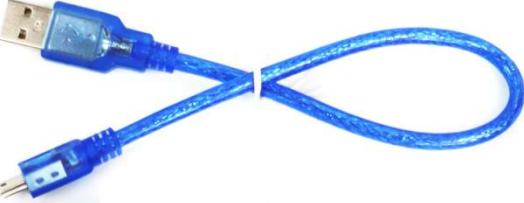
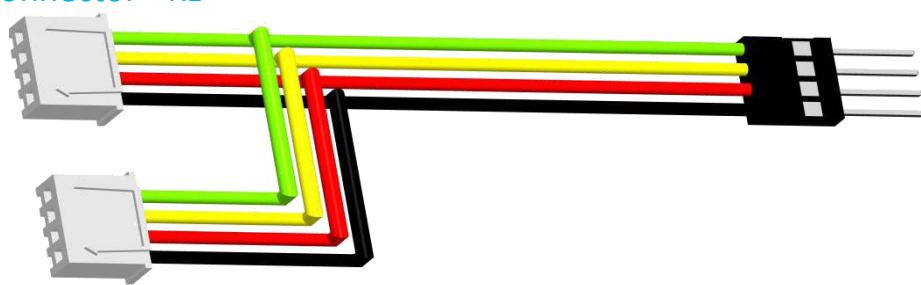


Adeekt RGB LED Module X2



NRF24L01 Module X2



Mini USB Cable X1		USB Cable X1	
LED Connector X1			
Buzzer Connector X1			
Jumper Wire M/M X4			

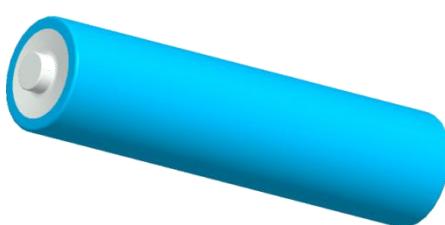
Tools

Cross Screwdriver X1	Slotted Screwdriver X1	Cross Socket Wrench X1
		

Wingding pipe X1



Self-prepared Parts

18650 Battery X4	
------------------	--

Software & Hardware

Introduction

As robots are increasingly widely used in various fields nowadays, and the intelligence evolves rapidly due to deepened innovation in recent years, they have changed greatly, and will continue to do so, our lifestyle and broadened our view of the world. Robots can work in lots of extreme and harsh environments where human beings cannot approach. And they can accomplish tasks easily that we have struggled to do. So is studying robotics more and more popular. To make the smart car robot work under the best status, it is necessary to learn more about it and seek improvements in its speed and direction.

This robot is designed for hobbyists to learn about Arduino and robotics. Smart car robots should first sense an obstacle before avoiding one. The sensor used in this car is an ultrasonic one. The car detects obstacles via the ultrasonic sensor and avoids them automatically.

This smart car is a typical robot. It's composed of three parts: sensor, actuator, and MCU. Also it can track lines – sense the leading trace and move accordingly. So the car can recognize routes automatically, select the right path, and avoid obstacle on the path. The actuator of the car is DC motors, which control the direction and speed of the car moving. As for MCU, an Arduino board is used for the core of the car.

This smart car robot is really smart in moving forward/backward and turning left/right in an unmanned manner by detecting obstacles and determining and controlling the reactions of the car by the MCU.

What is Arduino?

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges,

differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

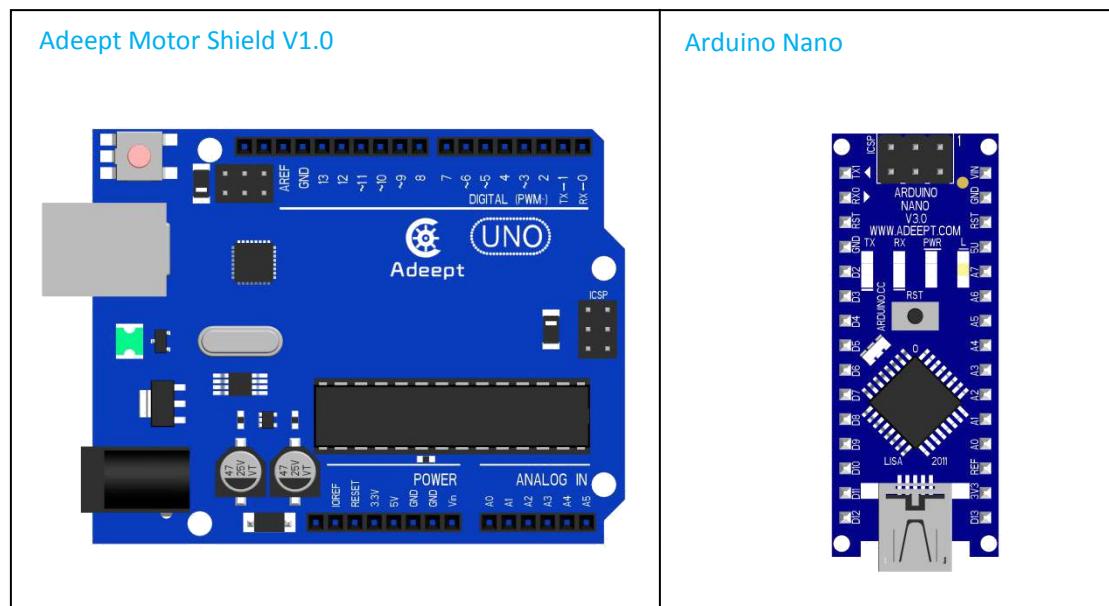
Why Arduino?

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step instructions of a kit, or sharing ideas online with other members of the Arduino community.

How Should I Use Arduino?

If you are a beginner with Arduino, Arduino learning kits on our website www.adeept.com would be a perfect step into this fantastic field!

Two types of Arduino board are used in this car kit: Adeept UNO R3 board and Arduino Nano board.



Power

The Arduino/Genuino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector. The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

Vin. The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND. Ground pins.

IOREF. This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.

PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.

LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible

to change the upper end of their range using the AREF pin and the analogReference() function.

There are a couple of other pins on the board:

AREF. Reference voltage for the analog inputs. Used with analogReference().

Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

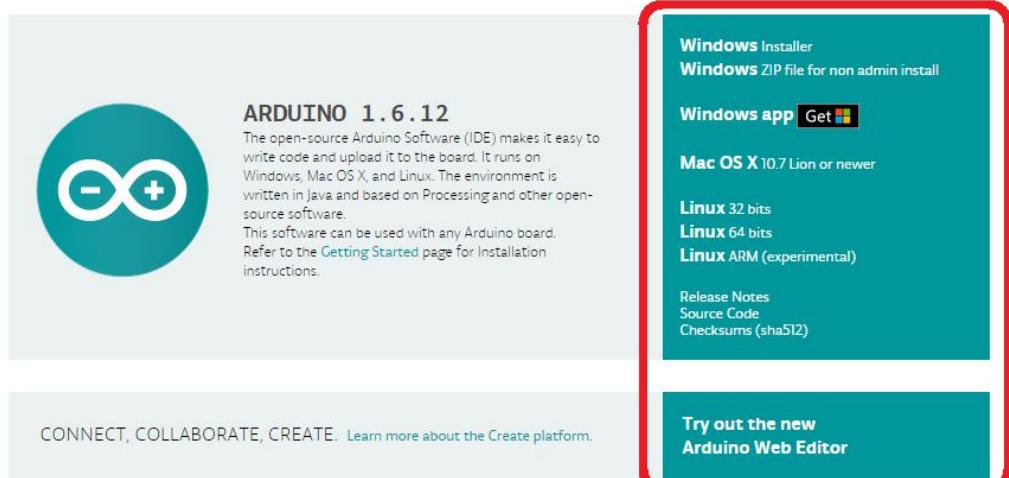
Arduino Software (IDE)

Arduino Software (IDE) is used to write and upload the code for Arduino Board. First, install Arduino software (IDE): visit <https://www.arduino.cc/en/Main/Software>. Download the corresponding installation program according to your operating system. If you are a Windows user, please select the “Windows Intaller” to download and install the driver correctly.



The screenshot shows the official Arduino website. At the top, there are two logos: the standard Arduino logo (infinity symbol with minus and plus signs) and the Genuino logo (a grid of squares with the word "Genuino" and "OUTSIDE USA" below it). To the right is a search bar with the placeholder "Search the Arduino Website" and a magnifying glass icon. Below the header is a navigation bar with links: Home, Buy, Download (which is highlighted in blue), Products, Learning, Forum, Support, and Blog. To the right of the navigation bar are "LOG IN" and "SIGN UP" buttons. At the bottom left is a "DOWNLOAD" button. On the far right is a language selection dropdown set to "ENGLISH".

Download the Arduino Software



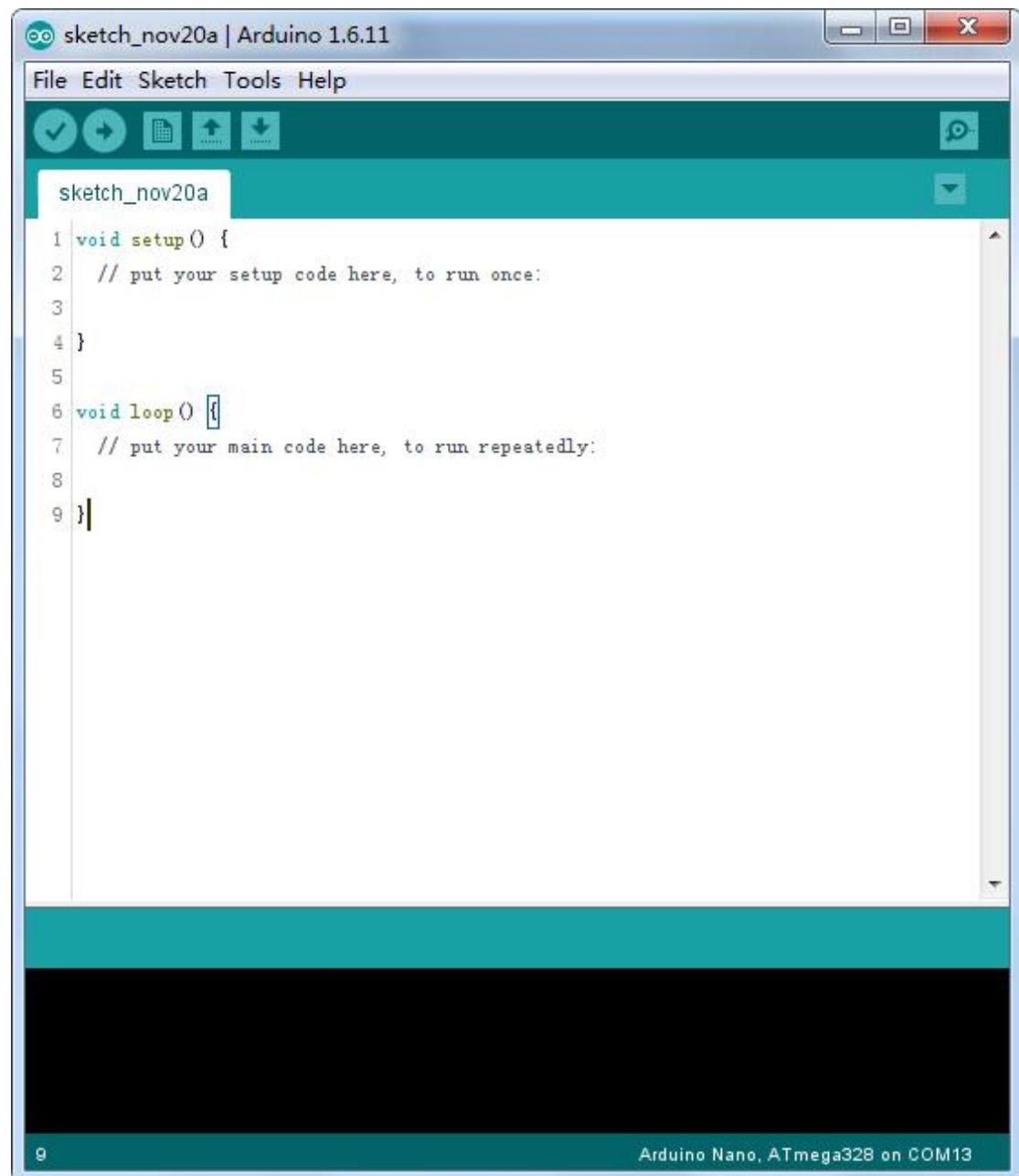
The screenshot shows the "ARDUINO 1.6.12" release notes. It includes a large circular Arduino logo on the left. The text describes the IDE as open-source software for Windows, Mac OS X, and Linux, written in Java and based on Processing. It mentions that the software can be used with any Arduino board and points to the "Getting Started" page for installation instructions. To the right, there is a section for Windows users with links for "Windows Installer" (ZIP file for non-admin install) and "Windows app" (Get). Below that are links for "Mac OS X 10.7 Lion or newer", "Linux 32 bits", "Linux 64 bits", and "Linux ARM (experimental)". At the bottom of this section are links for "Release Notes", "Source Code", and "Checksums (sha512)". A red box highlights this entire section. At the very bottom of the page is a teal banner with the text "TRY OUT THE NEW Arduino Web Editor".

After the download completes, run the installer. For Windows users, there may pop up an installation dialog box of the driver during the installation . Please agree the installation when it appears.

After installation is completed, an Arduino software shortcut will be generated on the desktop. Run the ide.



The interface of Arduino software is as follows:



The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.



Verify : Checks your code for errors when compiling it.



Upload : Compiles your code and uploads it to the configured board.

Before uploading your sketch, you need to select the correct items from the **Tools > Board** and **Tools > Port** menus. The boards are described below. On the Mac OS X, the serial port is probably something like **/dev/tty.usbmodem241** (for an Uno or Mega2560 or Leonardo) or **/dev/tty.usbserial-1B1** (for a Duemilanove or earlier USB board), or **/dev/tty.USA19QW1b1P1.1** (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably **COM1** or **COM2** (for a serial board) or **COM4, COM5, COM7**, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows **Device Manager**. On Linux, it should be **/dev/ttYACMx**, **/dev/ttYUSBx** or similar.

Once you've selected the correct serial port and board, press the upload button in the toolbar or select the **Upload** item from the **Sketch** menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecmila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is completed, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"



New: Creates a new sketch.



Open: Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: Due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead.



Save: Saves your sketch.



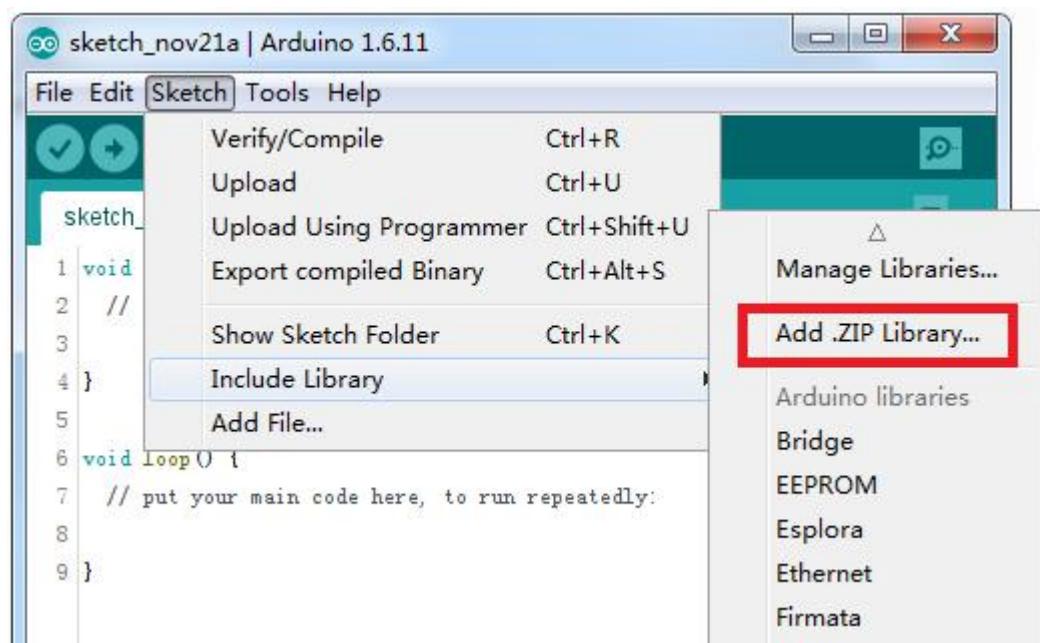
Serial Monitor: Opens the serial monitor.

Additional commands are found within the five menus: **File**, **Edit**, **Sketch**, **Tools**, and **Help**. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

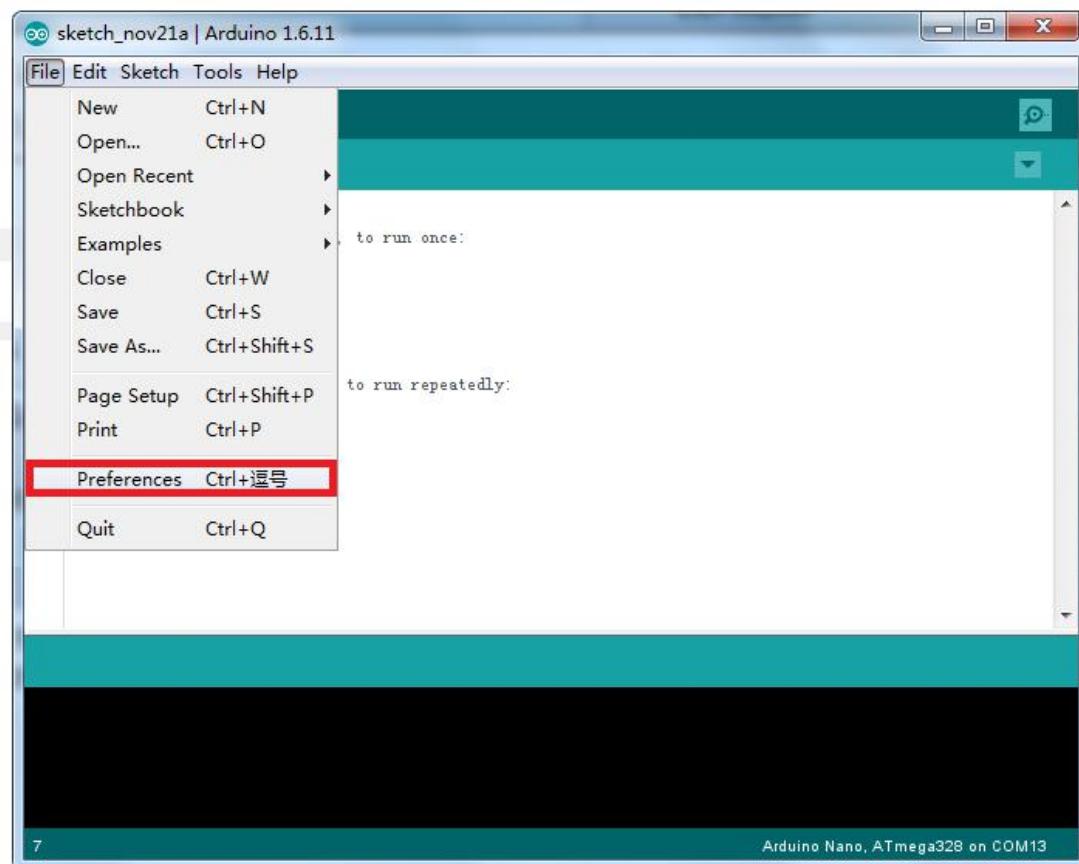
Since version 1.0, files are saved with an **.ino** file extension. Previous versions use the **.pde** extension. You may still open **.pde** named files in version 1.0 and later, and the software will automatically rename the extension to **.ino**.

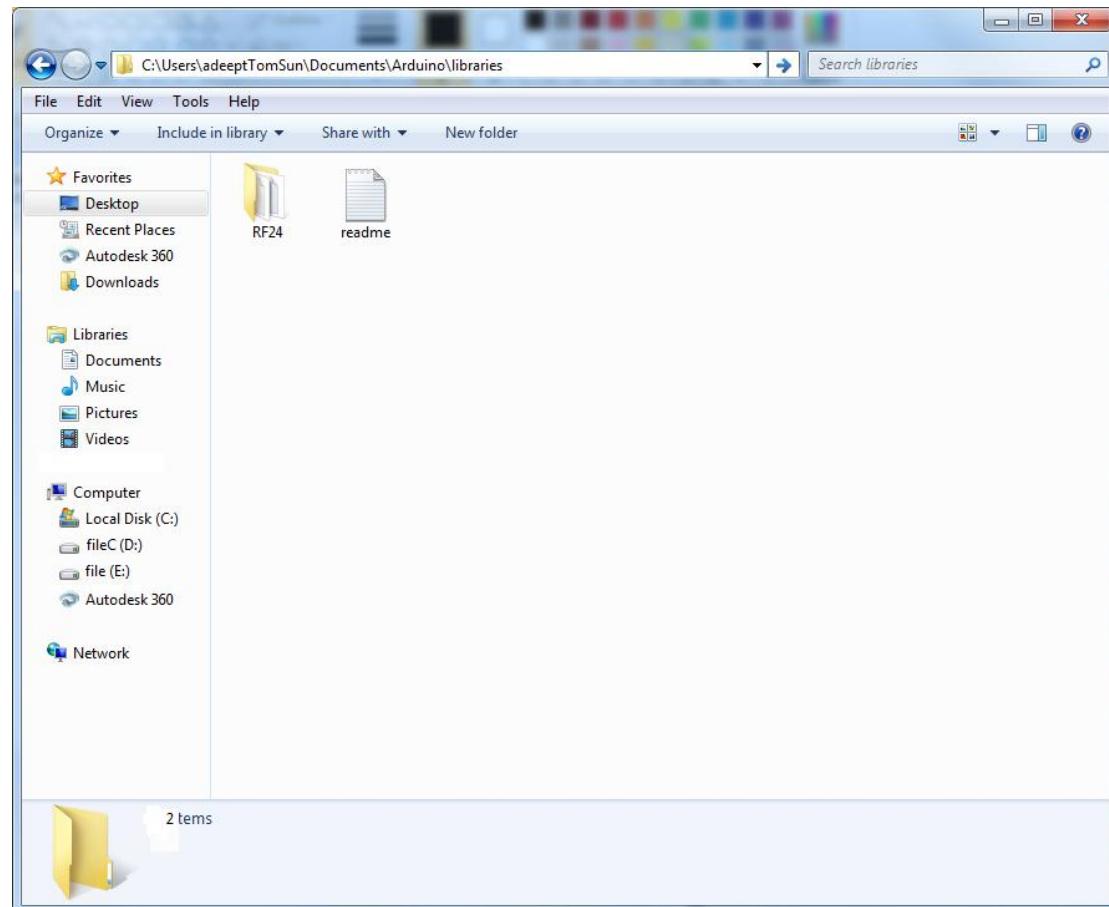
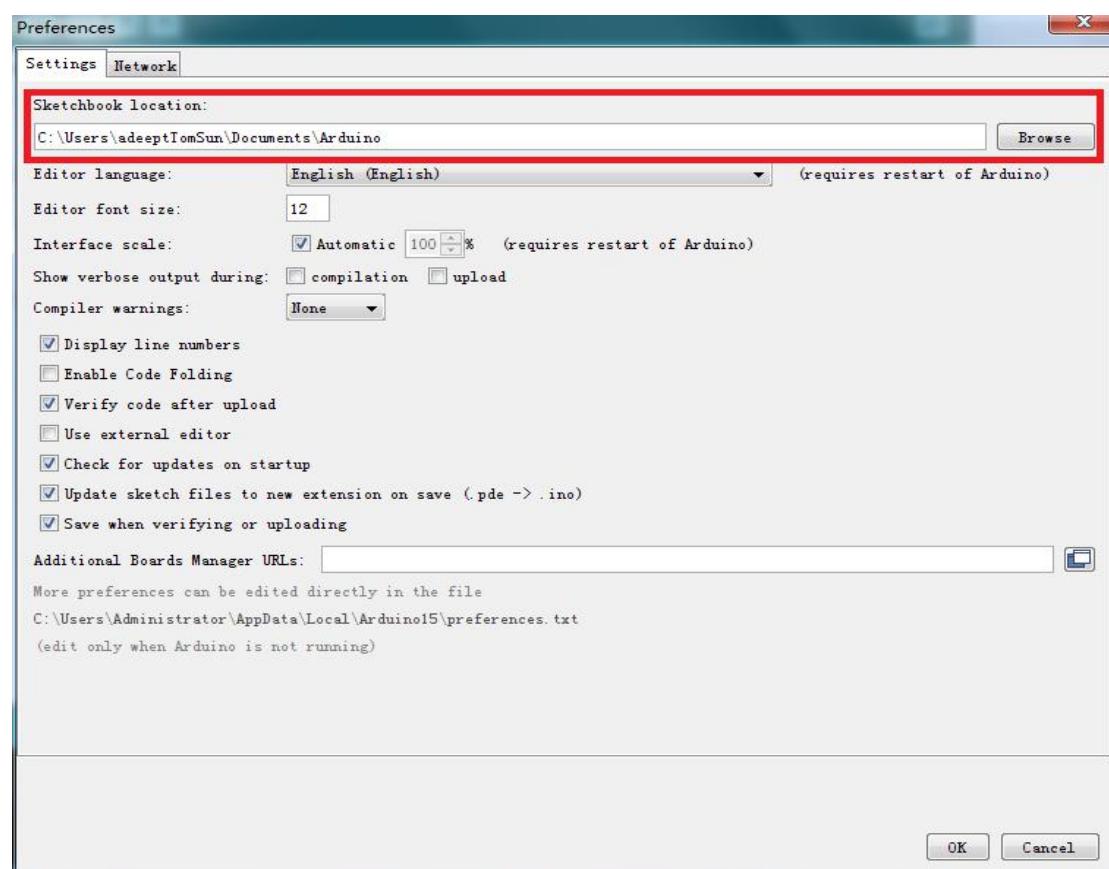
Install Library

The example sketches provided use the *RF24.ZIP* library, so you need to install it before compiling. Click **Add.ZIP Library** to add the *RF24.ZIP* to the *libraries* folder.



After the library is installed successfully, you can find the *RF24.ZIP* under **Sketchbook location:** on the window popped up by clicking Preferences.





Upload Program

After the preparations above, next we will upload the program (example sketches provided) to the Arduino Nano and Adeept UNO R3 boards. The car kit comprises of two parts: the remote control based on Arduino Nano and the car controller on Adeept UNO R3.

First, upload the sketch to Arduino Nano. Open the file for the remote control, the file *AdeeptRemoteControl.ino*.

AdeepsRemoteControl | Arduino 1.6.11

File Edit Sketch Tools Help

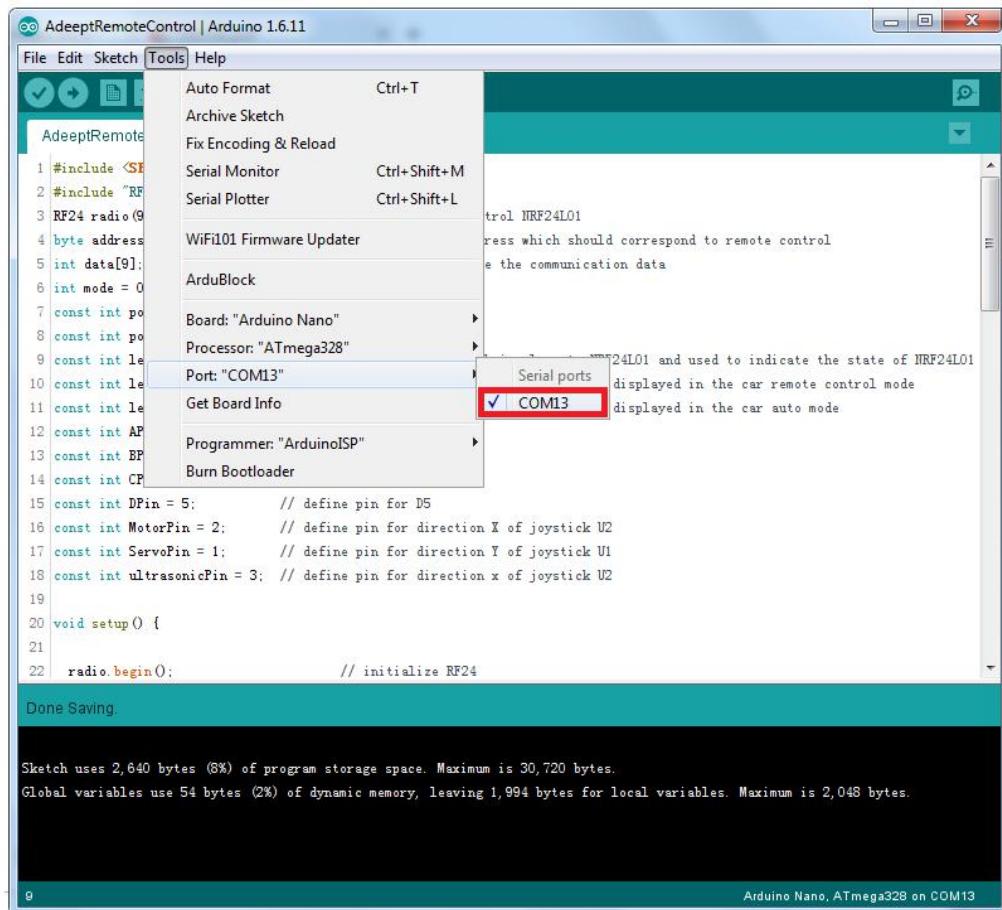
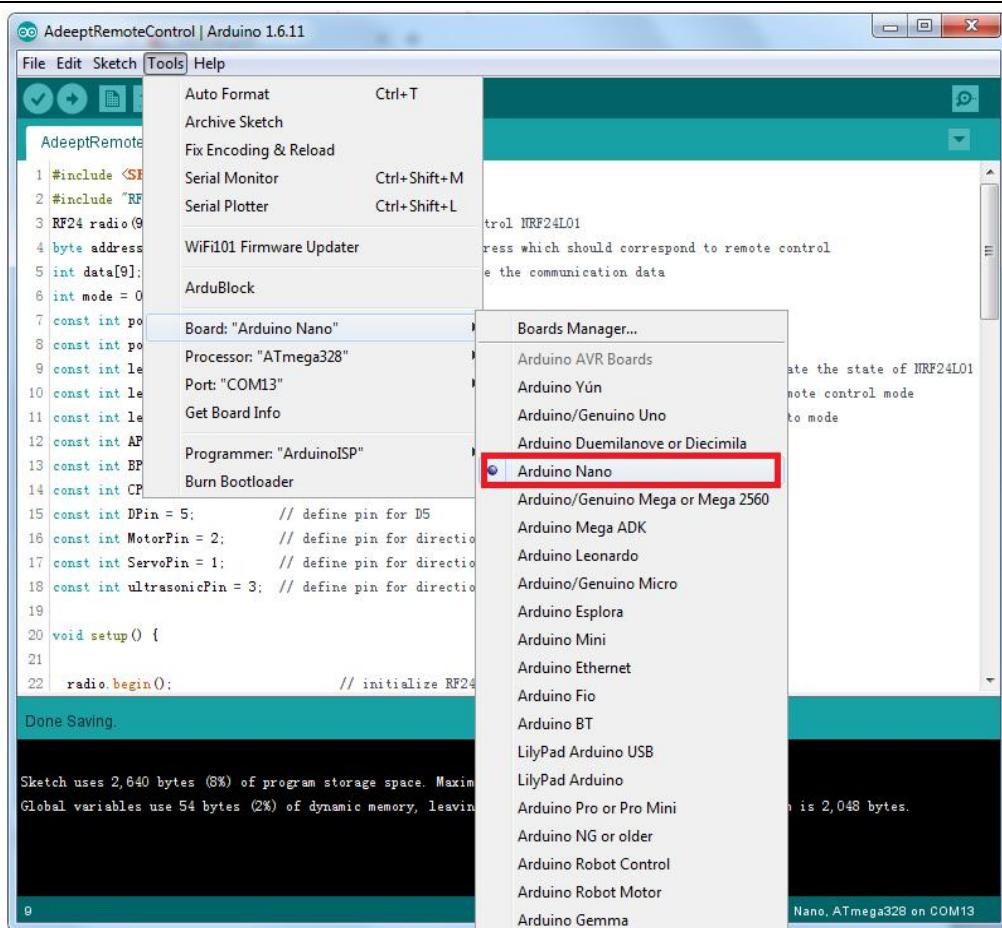
AdeepsRemoteControl §

```
1 #include <SPI.h>
2 #include "RF24.h"
3 RF24 radio(9, 10);           // define the object to control NRF24L01
4 byte addresses[5] = "00007"; // define communication address which should correspond to remote control
5 int data[9];                // define array used to save the communication data
6 int mode = 0;
7 const int pot6Pin = 5;        // define R6
8 const int pot5Pin = 4;        // define R1
9 const int led1Pin = 6;        // define pin for LED1 which is close to NRF24L01 and used to indicate the state of NRF24L01
10 const int led2Pin = 7;        // define pin for LED2 which is the mode is displayed in the car remote control mode
11 const int led3Pin = 8;        // define pin for LED3 which is the mode is displayed in the car auto mode
12 const int APin = 2;           // define pin for D2
13 const int BPin = 3;           // define pin for D3
14 const int CPin = 4;           // define pin for D4
15 const int DPin = 5;           // define pin for D5
16 const int MotorPin = 2;       // define pin for direction X of joystick U2
17 const int ServoPin = 1;        // define pin for direction Y of joystick U1
18 const int ultrasonicPin = 3; // define pin for direction x of joystick U2
19
20 void setup() {
21
22   radio.begin();              // initialize RF24
```

Done Saving.

Sketch uses 2,640 bytes (8%) of program storage space. Maximum is 30,720 bytes.
Global variables use 54 bytes (2%) of dynamic memory, leaving 1,994 bytes for local variables. Maximum is 2,048 bytes.

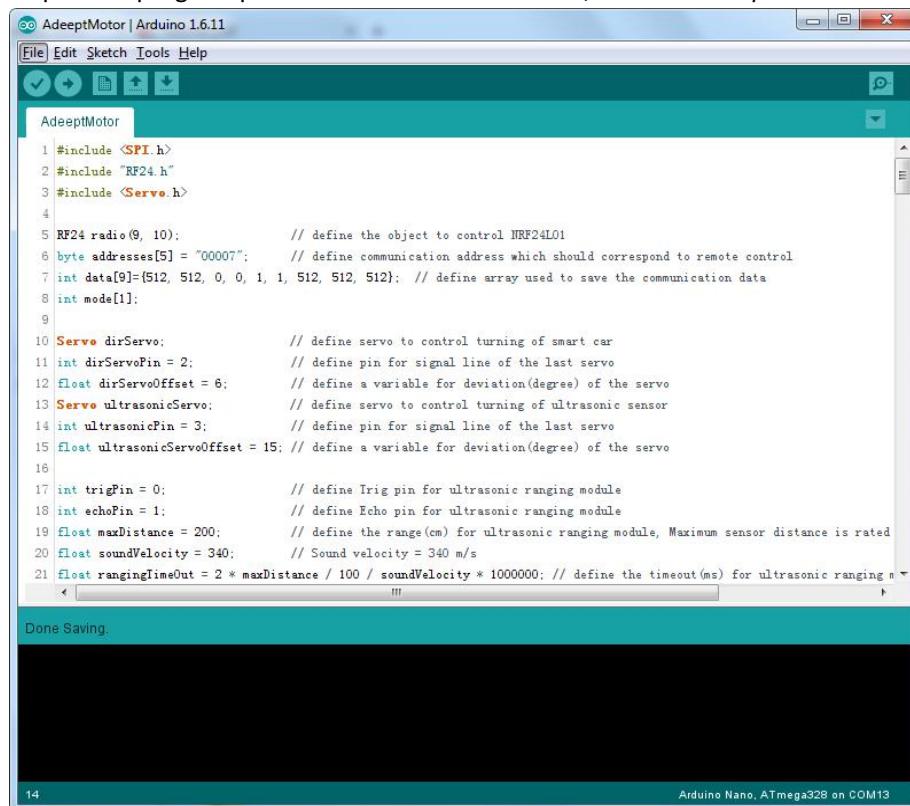
Connect the Arduino Nano to the computer. Select **Tool -> Board: "Arduino Nano"**-> **Arduino Nano**, and **Port ->COM13**. COMx is the port number assigned to the Arduino Nano and can be COM1, COM2, COM3...So it depends.



Next, click the upload button .



After the program is uploaded to the Nano successfully, upload another sketch to the Adeept UNO R3. Open the program provided for the control board, the file "AdeeptMotor.ino".



```

AdeeptMotor | Arduino 1.6.11
File Edit Sketch Tools Help

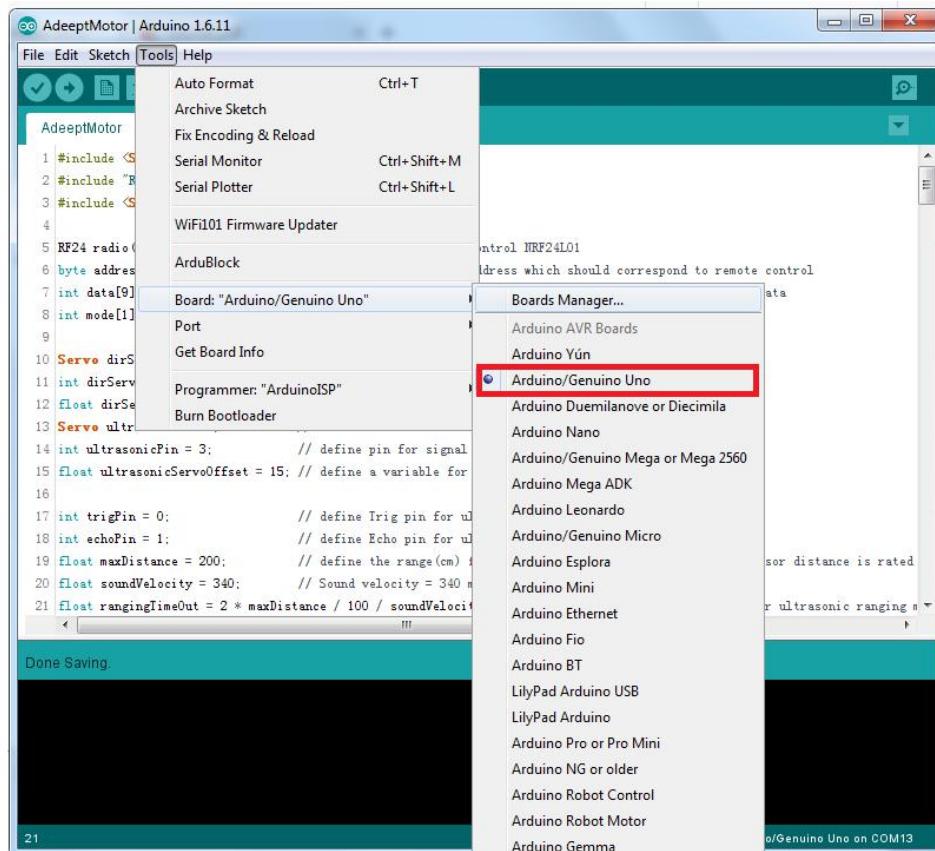
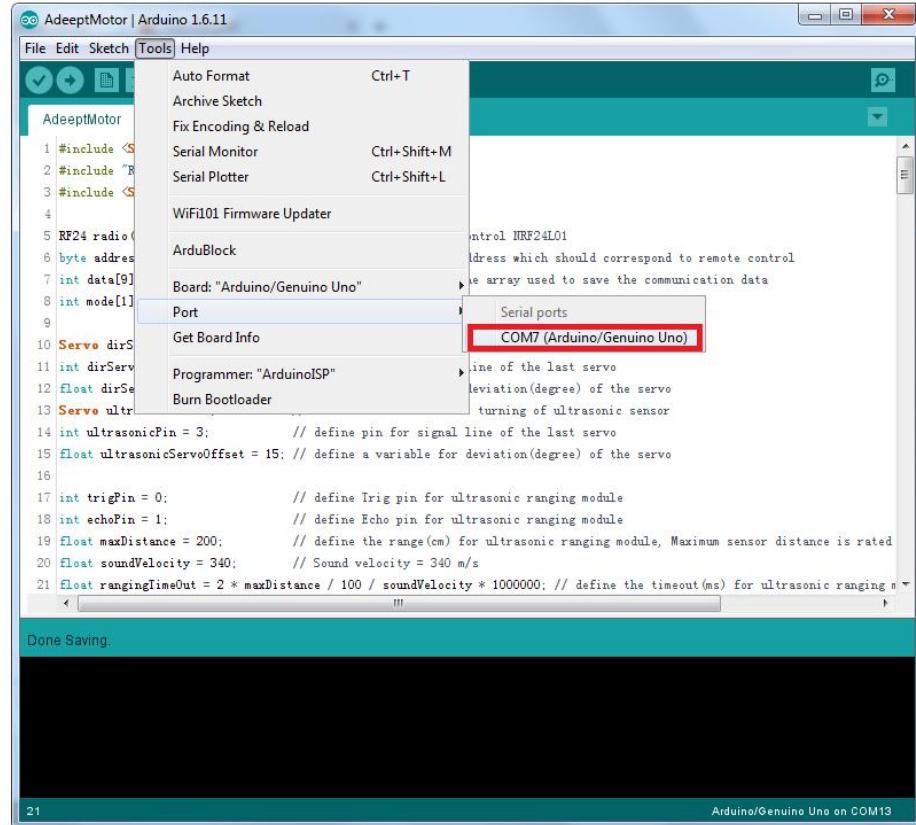
AdeeptMotor
1 #include <SPI.h>
2 #include "RF24.h"
3 #include <Servo.h>
4
5 RF24 radio(9, 10);           // define the object to control NRF24L01
6 byte addresses[5] = "00007";   // define communication address which should correspond to remote control
7 int data[9]={512, 512, 0, 0, 1, 1, 512, 512, 512}; // define array used to save the communication data
8 int mode[1];
9
10 Servo dirServo;             // define servo to control turning of smart car
11 int dirServoPin = 2;          // define pin for signal line of the last servo
12 float dirServoOffset = 6;      // define a variable for deviation(degree) of the servo
13 Servo ultrasonicServo;       // define servo to control turning of ultrasonic sensor
14 int ultrasonicPin = 3;        // define pin for signal line of the last servo
15 float ultrasonicServoOffset = 15; // define a variable for deviation(degree) of the servo
16
17 int trigPin = 0;              // define Trig pin for ultrasonic ranging module
18 int echoPin = 1;               // define Echo pin for ultrasonic ranging module
19 float maxDistance = 200;       // define the range(cm) for ultrasonic ranging module, Maximum sensor distance is rated
20 float soundVelocity = 340;     // Sound velocity = 340 m/s
21 float rangingTimeOut = 2 * maxDistance / 100 / soundVelocity * 1000000; // define the timeout(ms) for ultrasonic ranging

```

Done Saving.

Arduino Nano, ATmega328 on COM13

Connect the Arduino UNO R3 board to the PC. Select **Tool -> Board "Arduino/Genuino Uno"**, and **Port -> COM7**. Also here is COM7, assigned to the Uno, but it can be COM1, COM2, COM3...



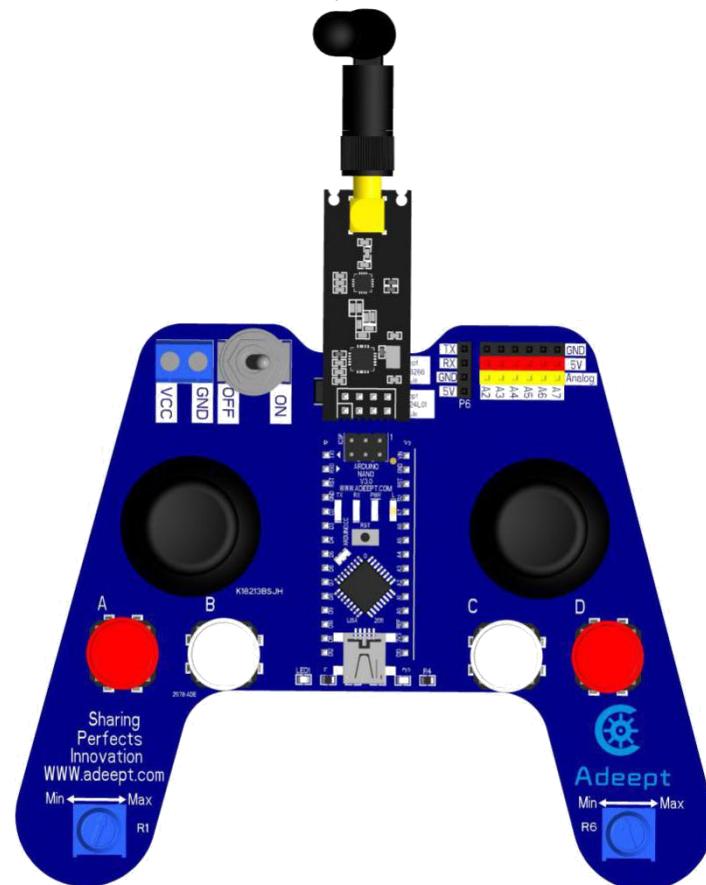
Click the button  to upload the sketch to the board.



Functions

Remote control:

1. Move the sticker rightside left and right to control the direction of the car, and forward and backward to control the left and right turning of the ultrasonic module.
2. Move the sticker leftside forward and backward to control the speed of the car going forward and backward.
3. Spin the R1 potentiometer to control the startup speed of the DC motor.
4. Spin the R6 potentiometer to fine tune the direction of the car.
5. Press Button A to switch the LED colors and status on the car, between white, red, green, blue and off. Press and hold it and the switching;
6. Press Button B and the car will switch to the remote control mode. At the same time, the LED2 will light up and LED3 go out accordingly, indicating the car now is controlled by the remote control.
7. Press Button C and the car will switch to the auto-control mode. At the same time, the LED3 will light up and LED2 go out, indicating the car now moves automatically.
8. Press Button D and the buzzer will beep. As long as you keep pressing, it will keep the beeping. The function is still active under the auto-control mode.
9. The car enters the remote control mode automatically after power on.
10. You can switch the status of the car by the remote control.



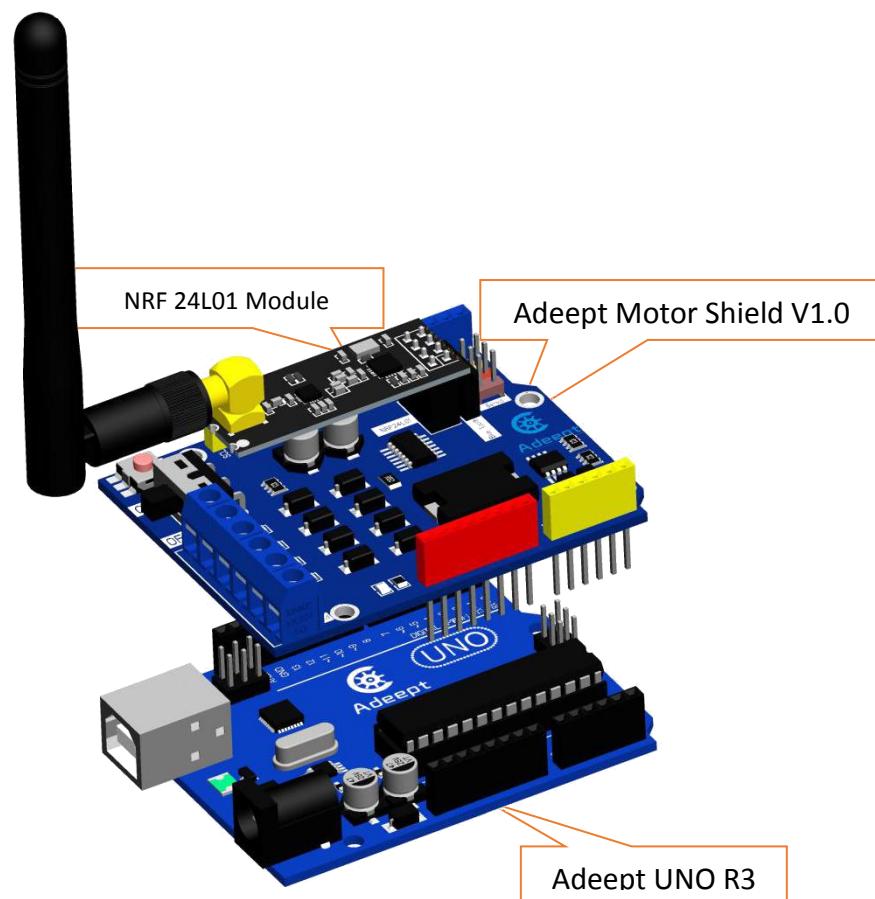
Control Board:

1. Under remote control mode:

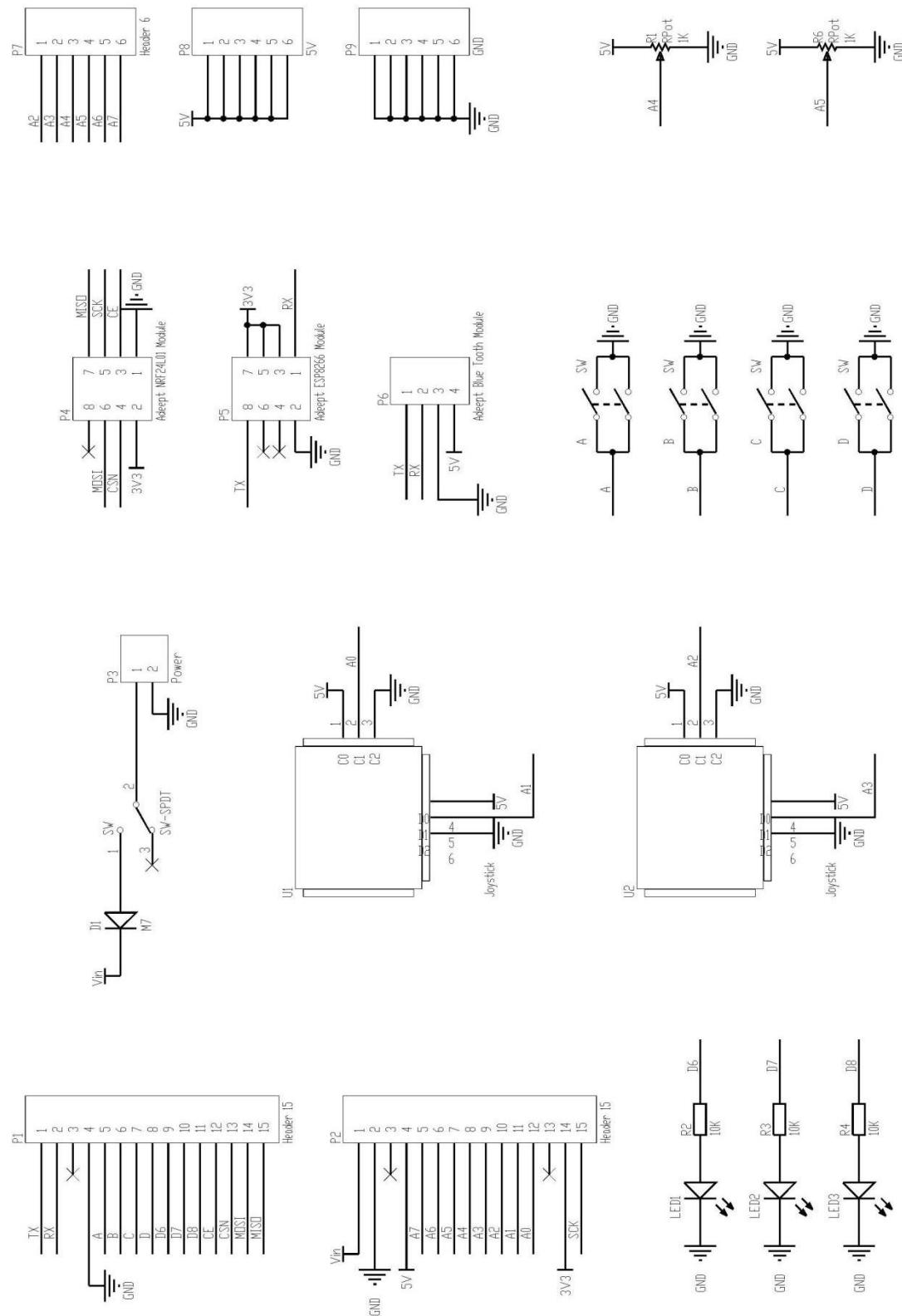
The car is completely controlled by the remote control.

2. Under auto-control mode:

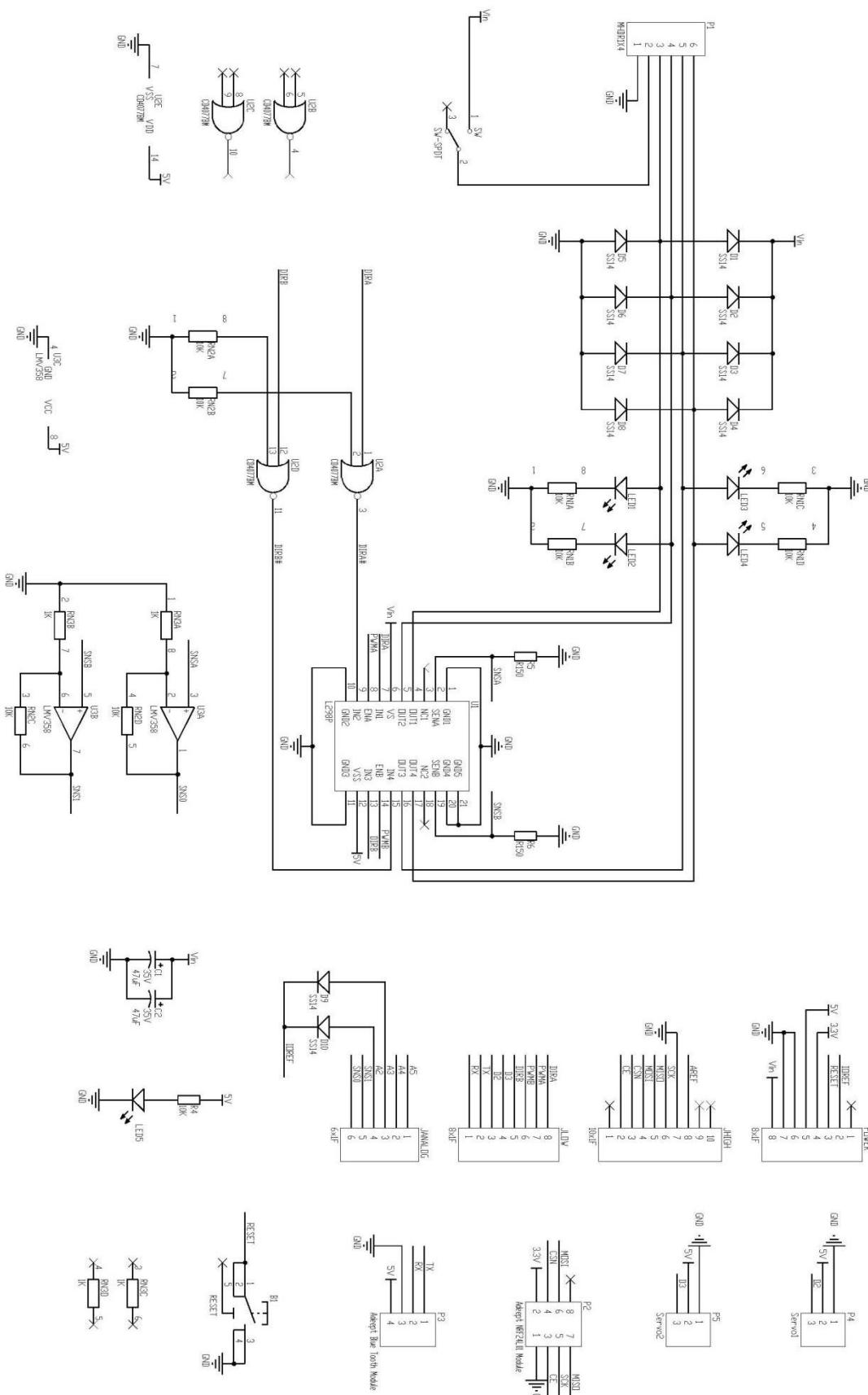
The ultrasonic module will keep detecting obstacles in front of the car. When encountering and approaching one, the car will go backward, turn to another angle bypassing the obstacle, and continue to go forward.



Schematic Diagram of Adeept Remote Control Shield V1.0



Schematic Diagram of Adeept Motor Shield V1.0

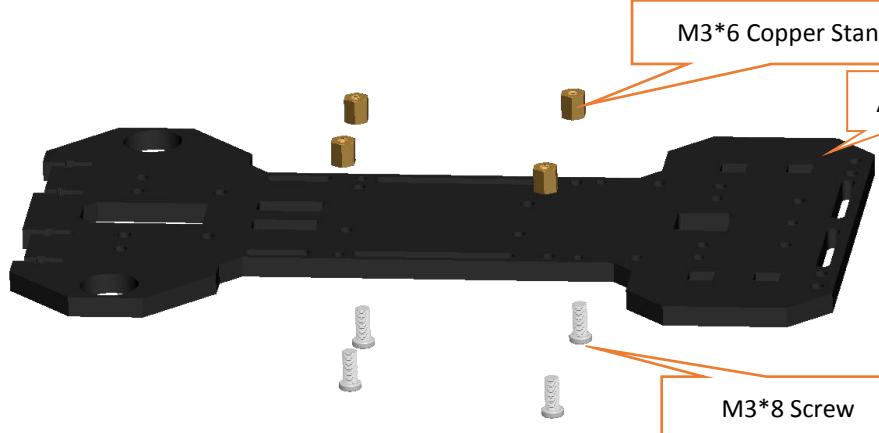


Assembly

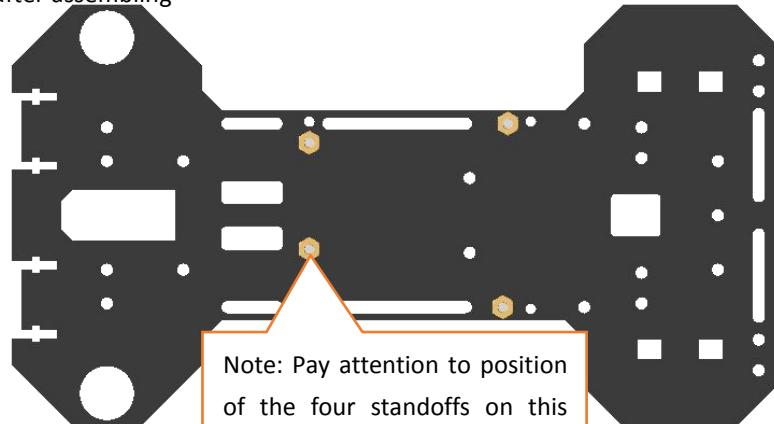
Fix the Position

Place four M3*6 copper standoffs on the holes of the A13 (main plate). Fix them by four M3*8 screws. Determine the orientation of the plate for the subsequent assembly.

Assemble the following components

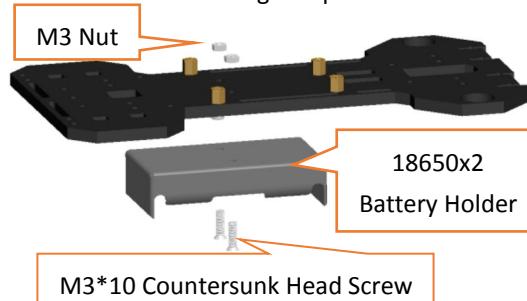


Effect diagram after assembling

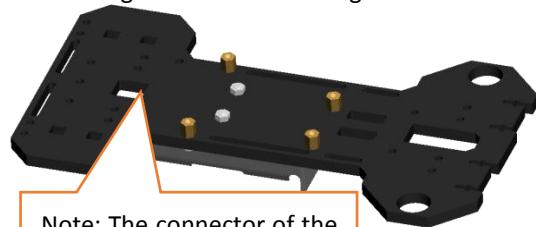


Fix the Battery Holder

Assemble the following components

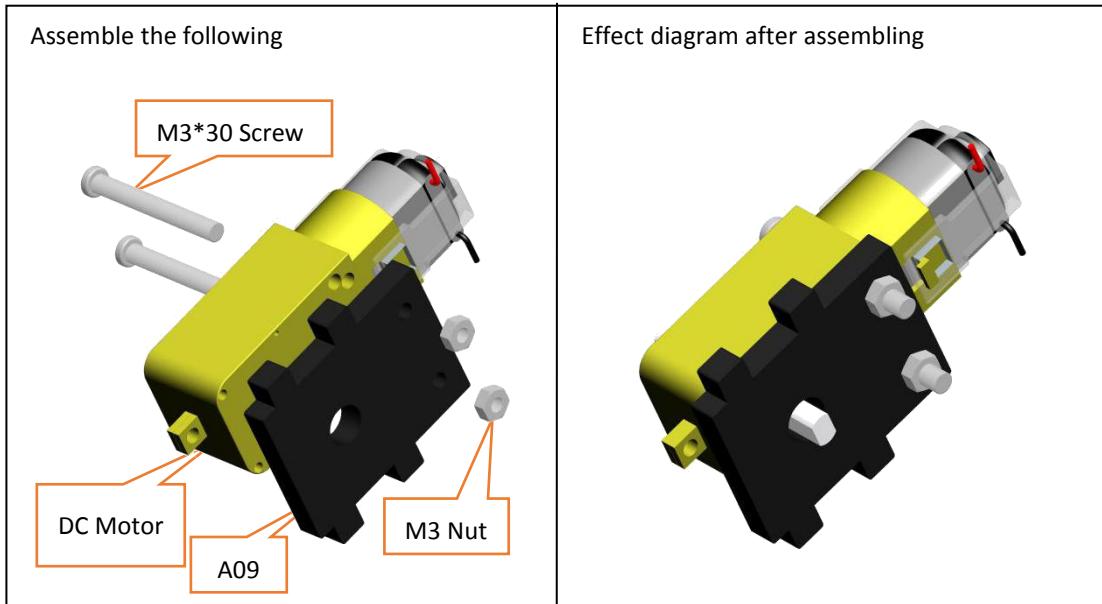


Effect diagram after assembling

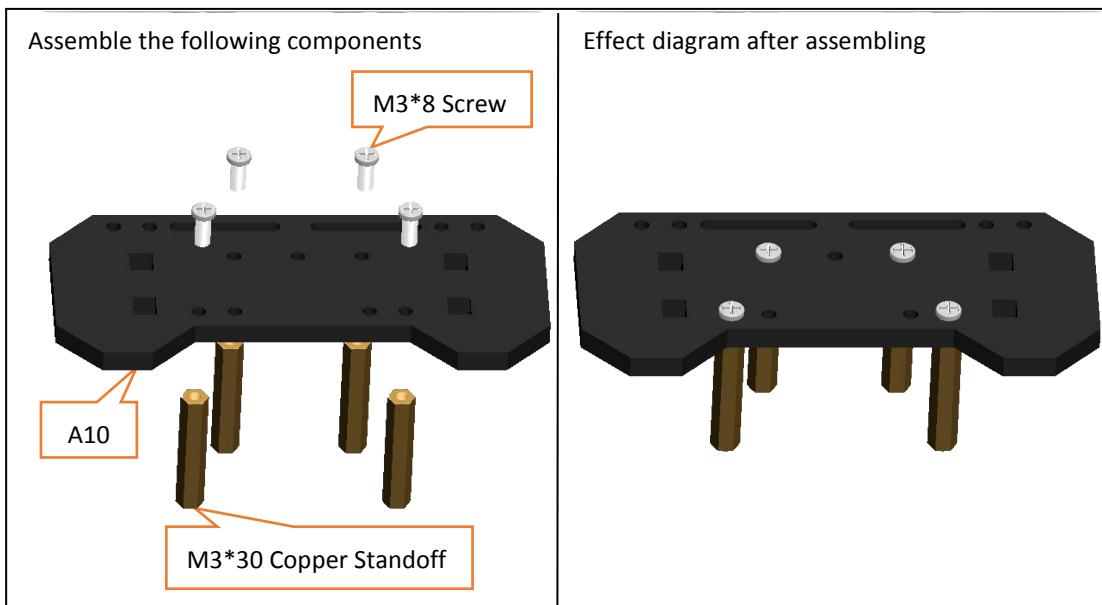


Rear Wheels

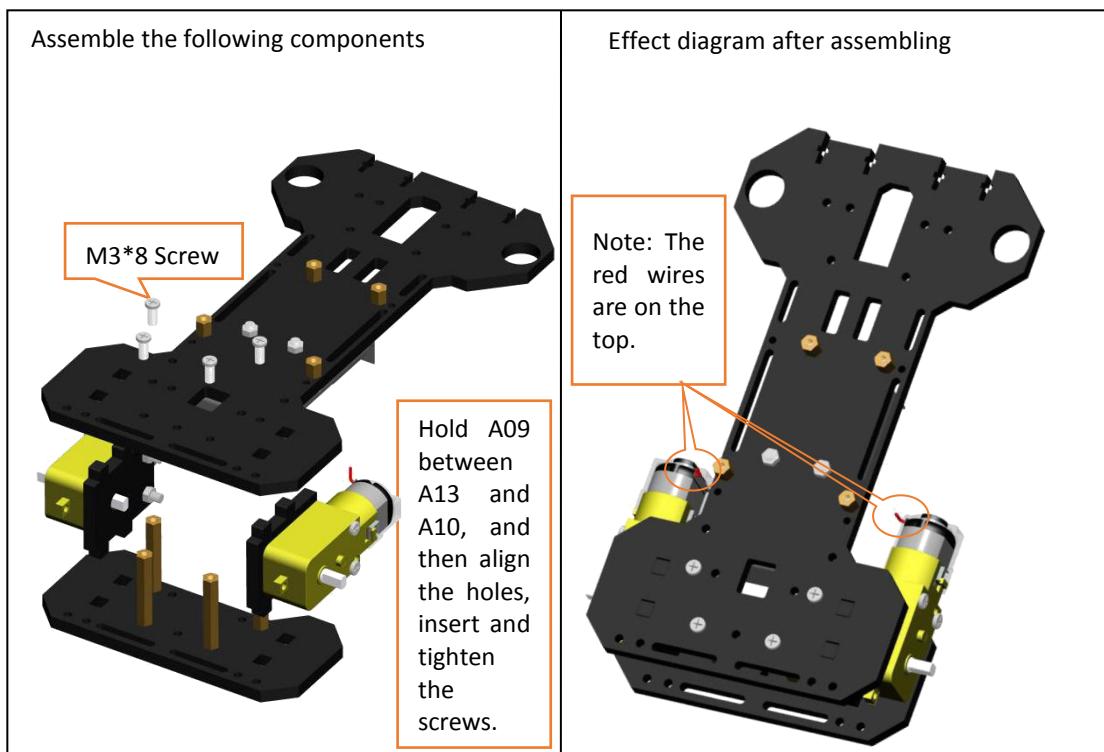
3.1 Fix the two DC motors on the A09 plate. Try to place the motor right in the middle position and along the middle axis of the plate.



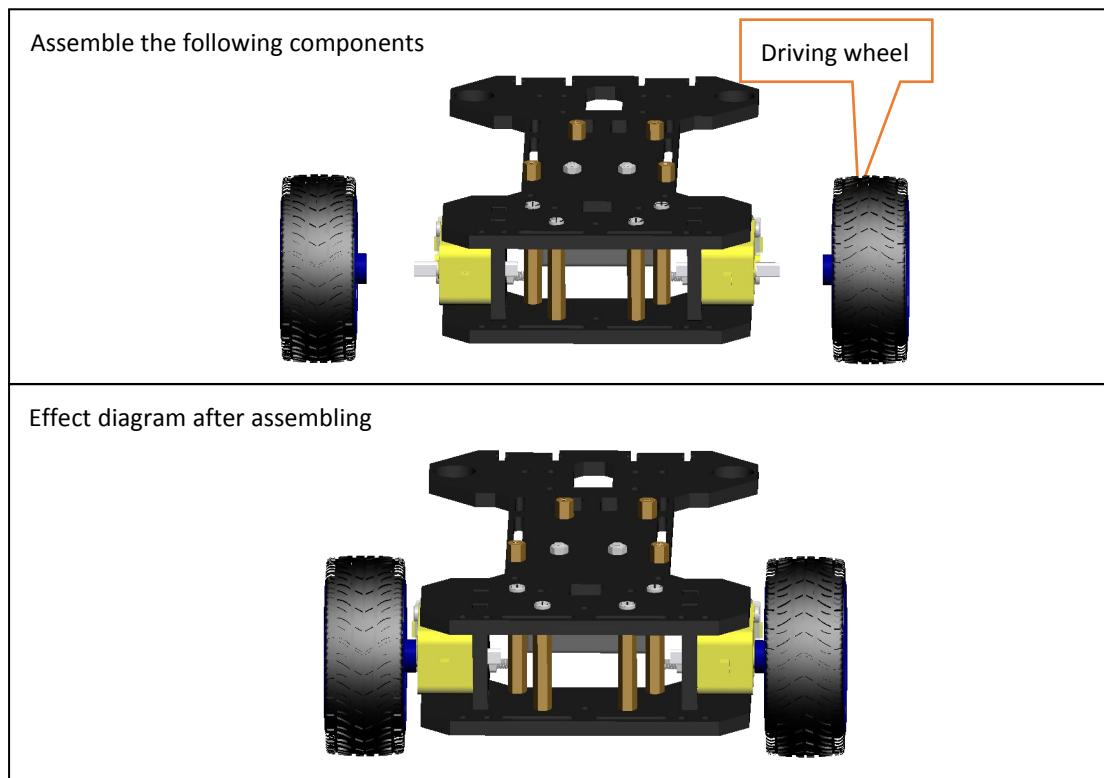
3.2 Fix the M3*30 copper standoffs onto the Plate A10.



3.3 Fix the part assembled in previous 3.1 and 3.2 under the A13 (main plate).

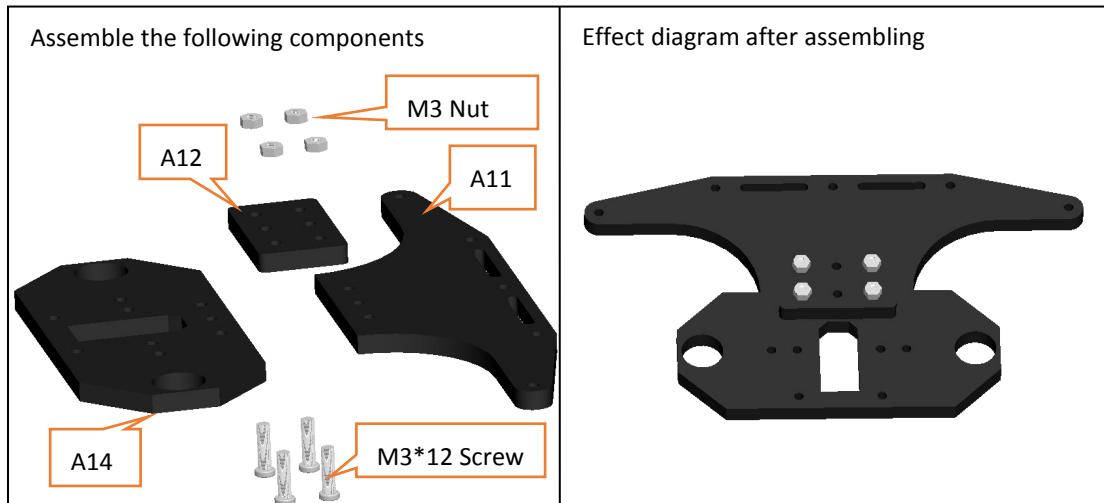


3.4 Press the rear wheels into the shaft of the motors to the farthest.



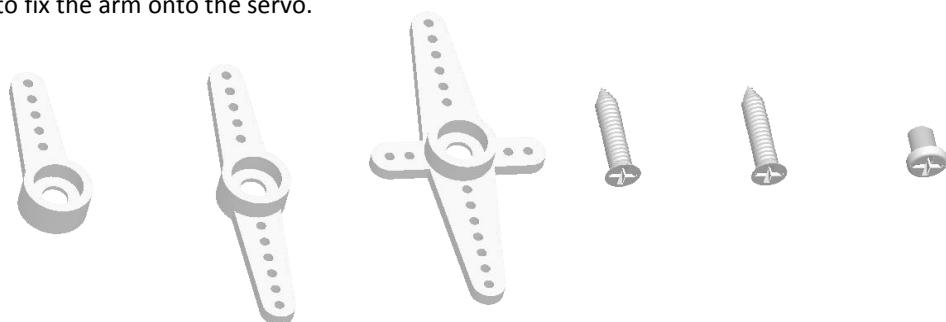
Front Wheels

4.1 Connect Plate A11 and A14 with A12.



4.2 Adjust the servo with built-in rocker arms.

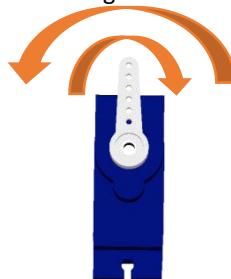
First, learn the structure. The servo can connect the rocker arm and spin to drive components bound with the arm. There are 3 types of rocker arms and 3 screws in the package. The smallest screw is to fix the arm onto the servo.



Mount and remove the rocker arm.

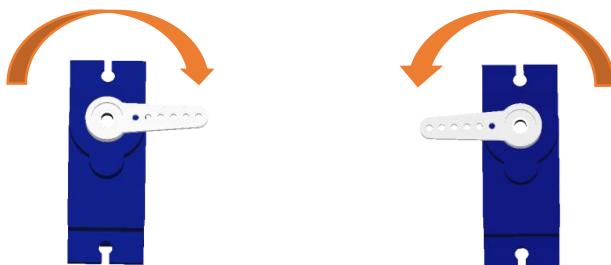


Rotate the rocker arm between 0 and 180 degrees.

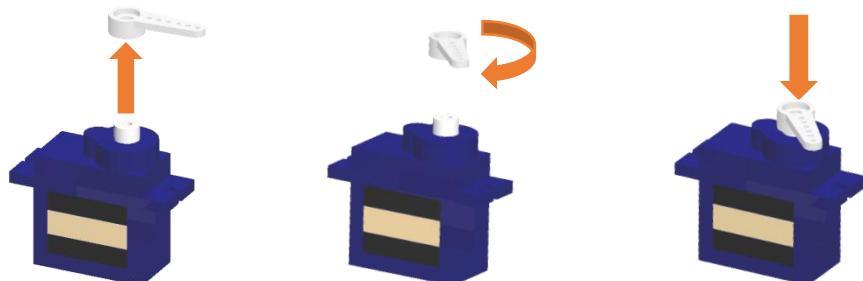


Now adjust the servo. This step is to make the servo shaft in the middle, so the component connected to the servo can be driven to move in a certain scope as needed.

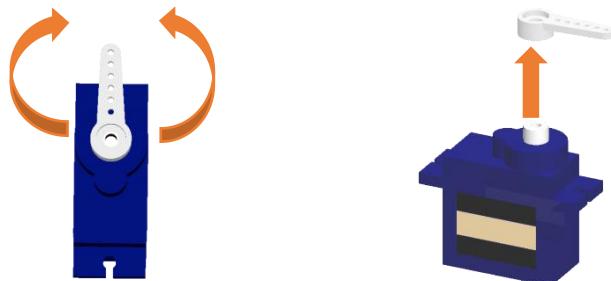
Adjust the rocker arm to make it rotate to an almost equal angle towards left and right.



If the angle is not nearly the same, please remove the arm and install it again. Repeat the step until nearly the SAME degree.



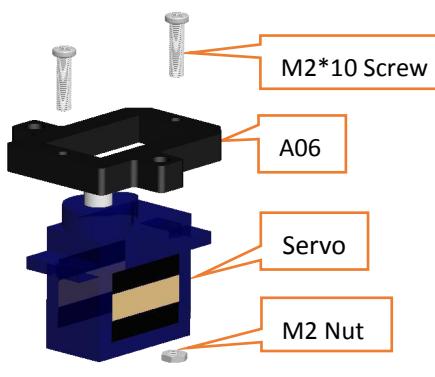
After the adjusting, the rocker arm should be in the middle axis. Remove the arm.



Make sure all servos have been adjusted and DO NOT spin the servo shaft before the whole assembly is done for the car. If you move it accidentally, readjust before the assembly.

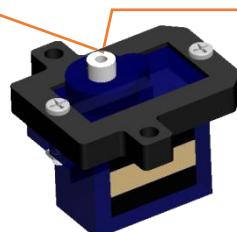
4.3 Assemble servo to Plate A06 (2 groups).

Assemble the following components

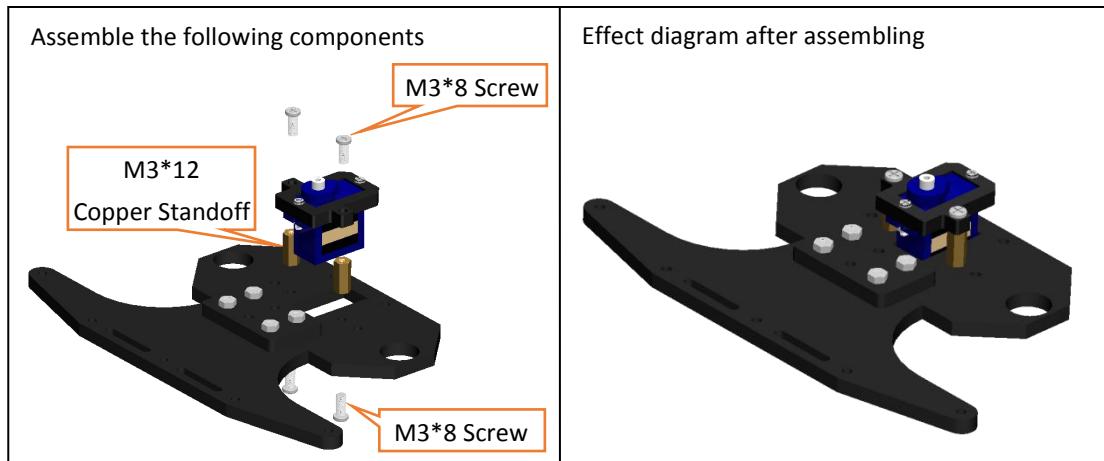


Effect diagram after assembling

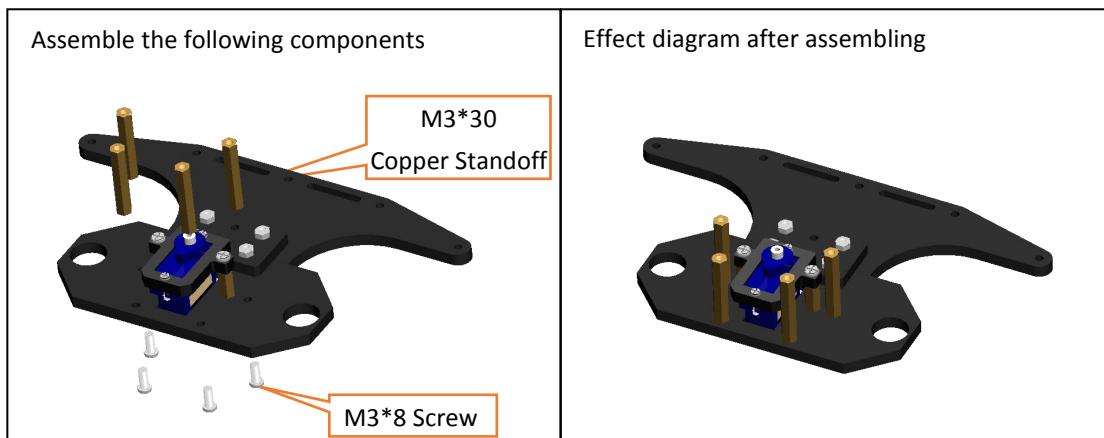
Note: The servo shaft should be closer to the bulges on the plate, and the nuts underneath.



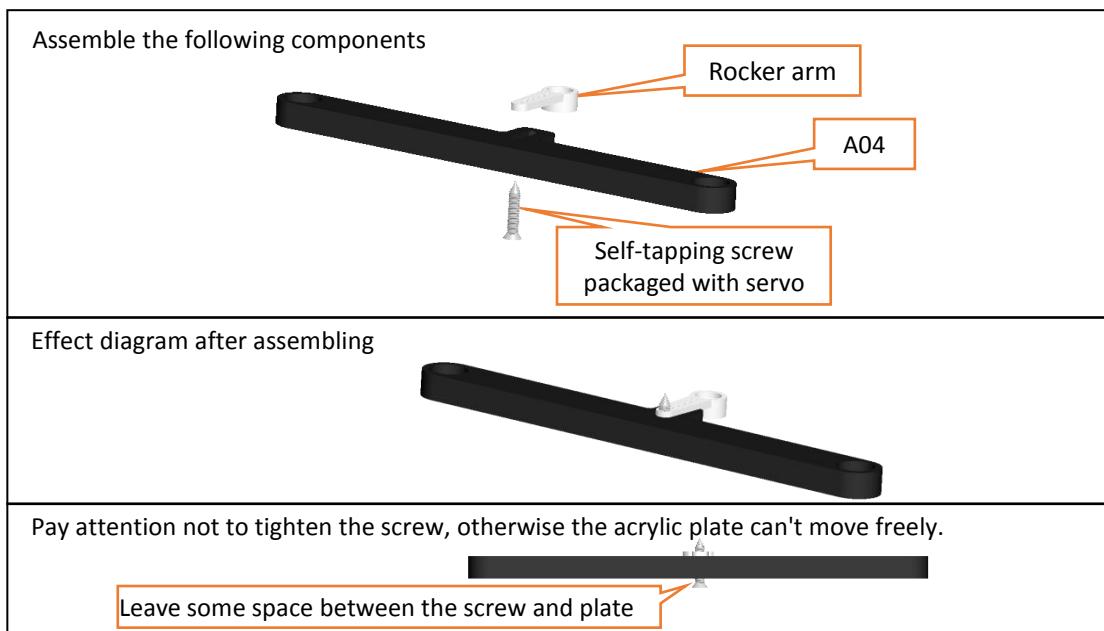
4.4 Fix a servo on Plate A14.



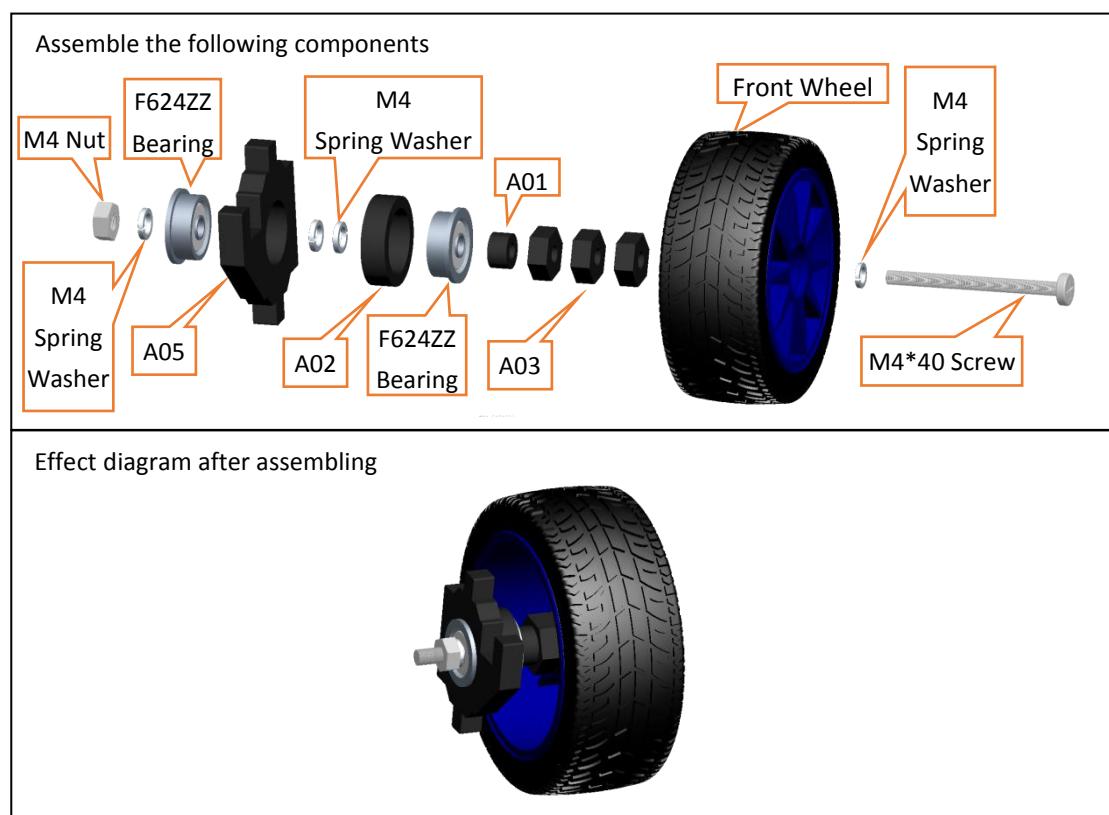
4.5 Fix 4 M3*30 standoffs .



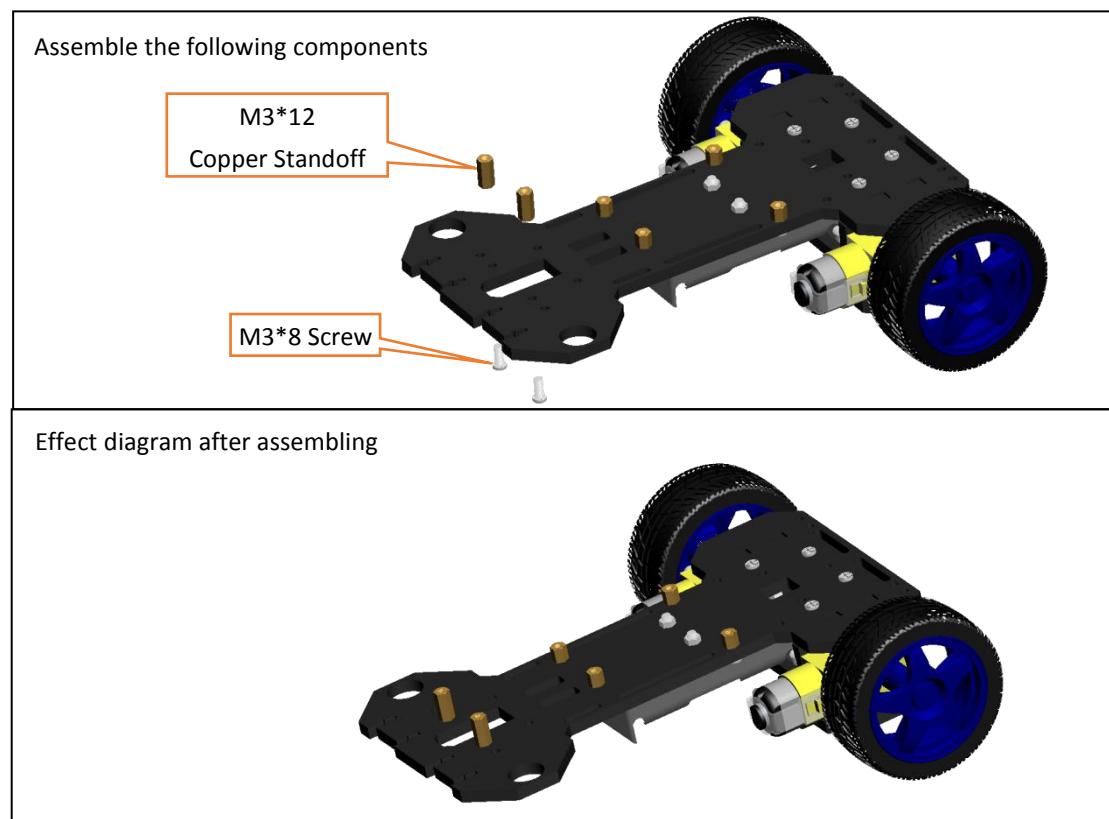
4.6 Connect rocker arm and Plate A04.



4.7 Install the front wheel (2 groups).



4.8 Fix 2 M3*6 standoffs on Plate A13.



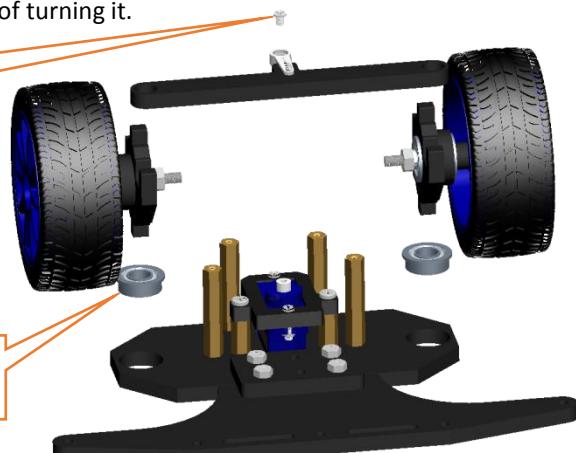
4.9 Assemble front wheels and steering part.

Assemble the following components

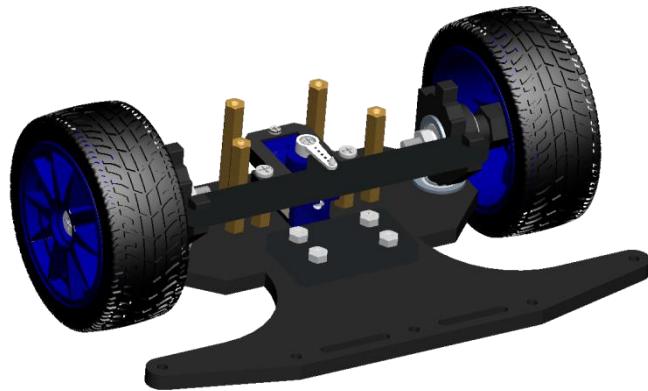
The rocker arm should be mounted almost perpendicular to the steering plate, so it can turn to a nearly equal degree towards left and right. But small deviation is acceptable. If not, remove the arm and reassemble again, instead of turning it.

Fixing screw packaged with servo

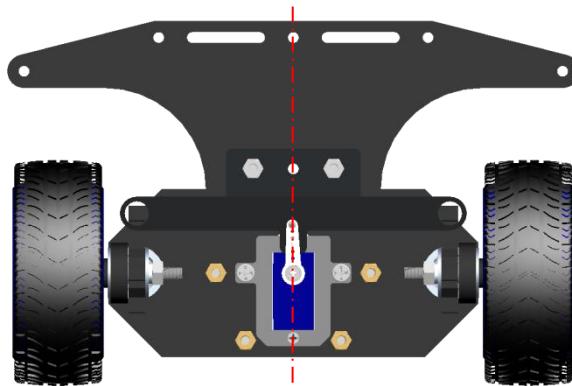
F687ZZ Bearing



Effect diagram after assembling

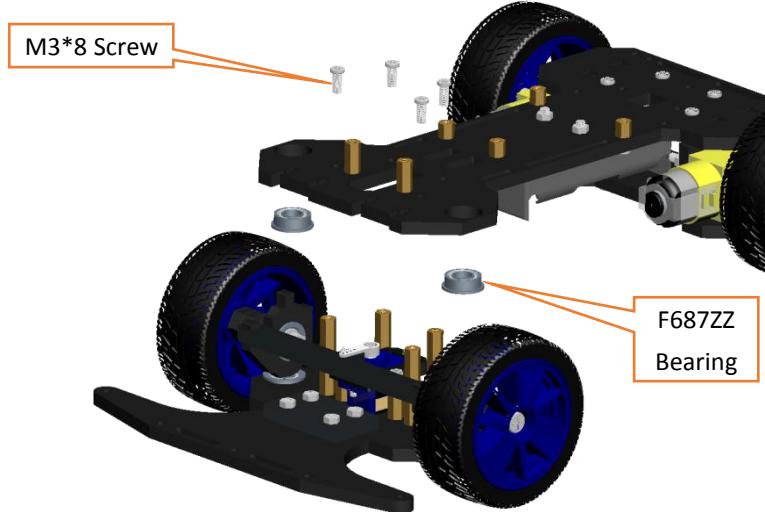


From the top view, the rocker arm should be in the middle of the rotation range, so it can rotate 90 degrees towards left and right. Otherwise, you need to readjust it again based on Step 4.2 above.

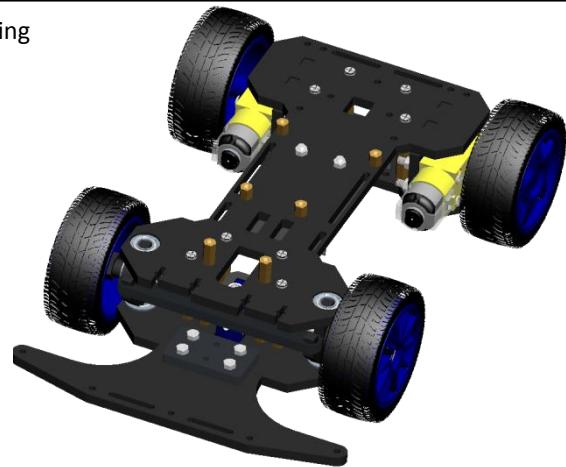


4.10 Fix the front wheels and the steering part.

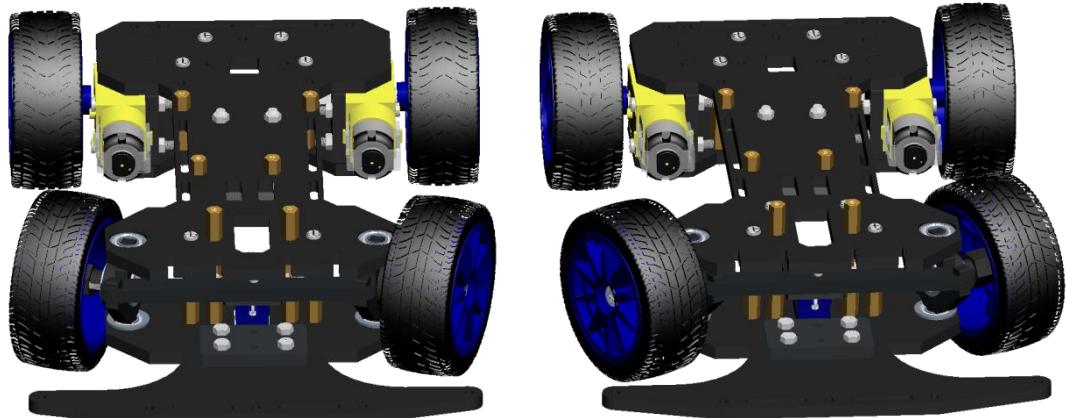
Assemble the following components



Effect diagram after assembling

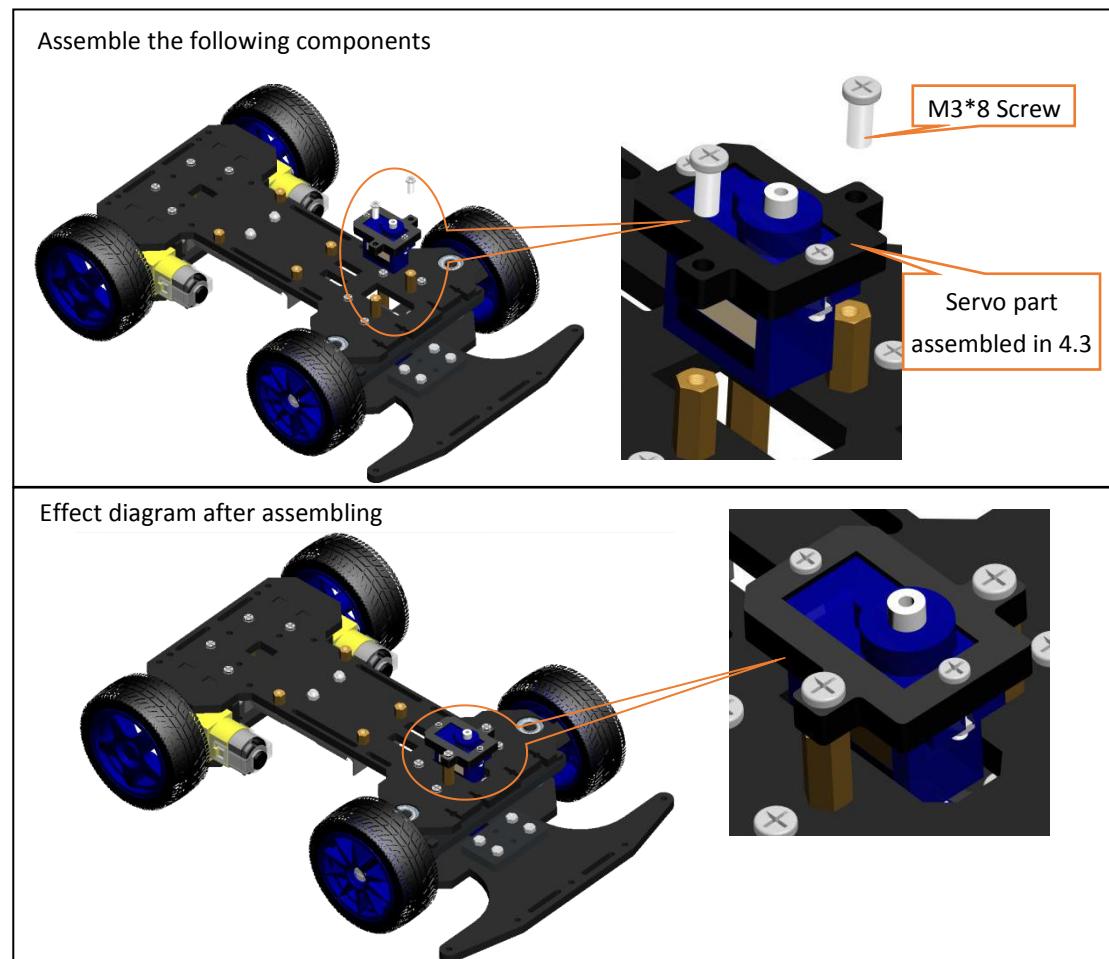


The two wheels can turn left and right.

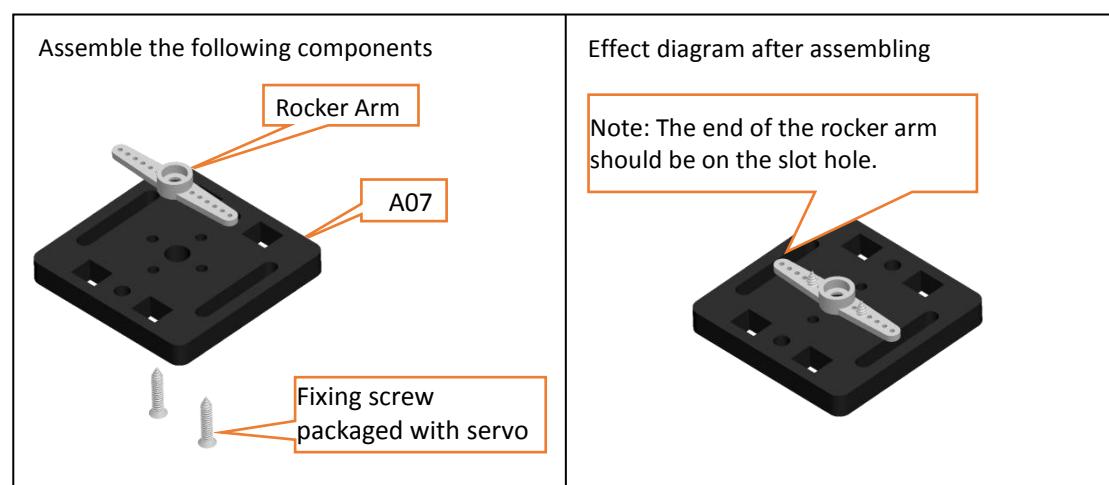


Assemble Ultrasonic Module

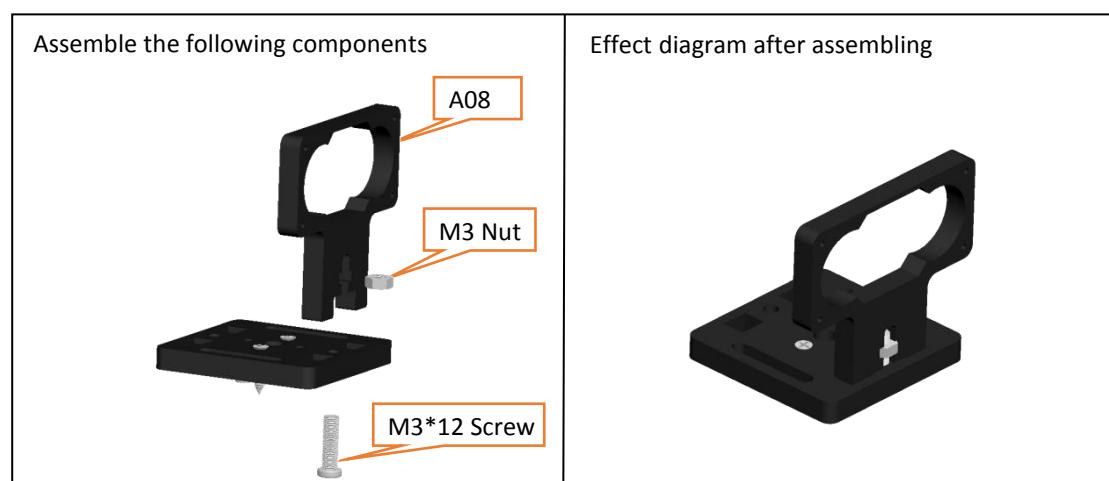
5.1 Assemble the servo.



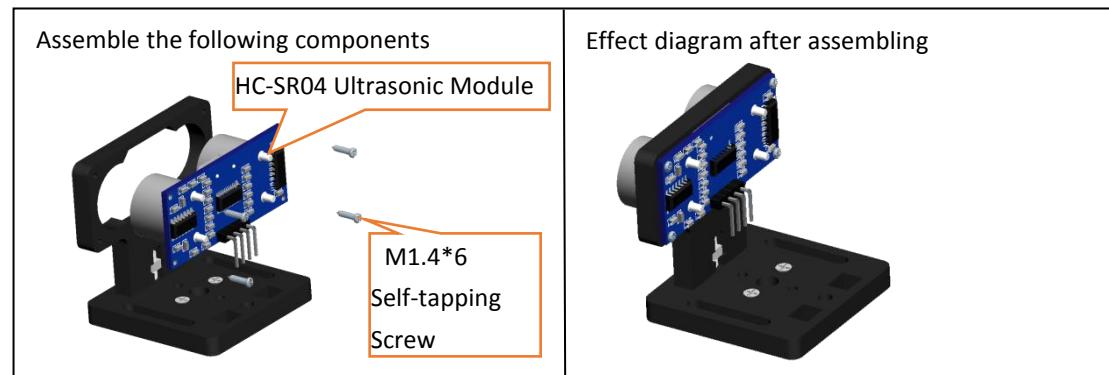
5.2 Assemble the rocker arm and Plate A07.



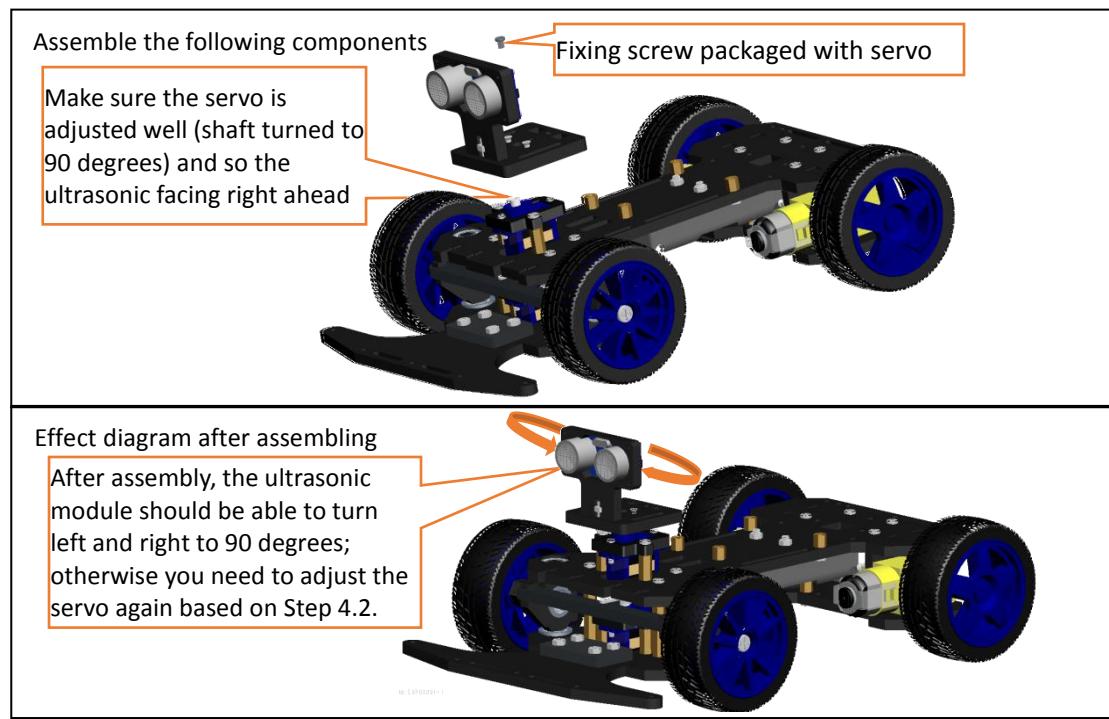
5.3 Fix the Plate A08 and A07.



5.4 Fix the Ultrasonic Module on the part assembled in Step 5.3.



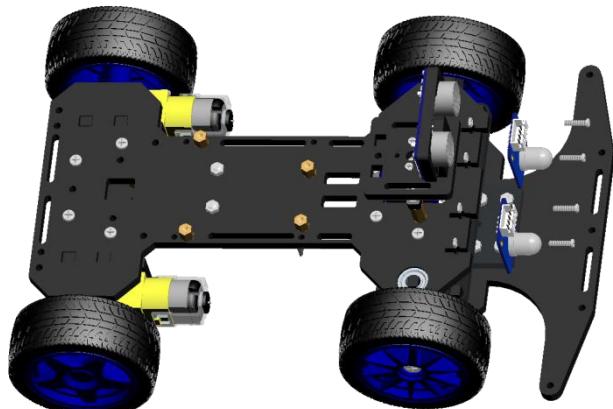
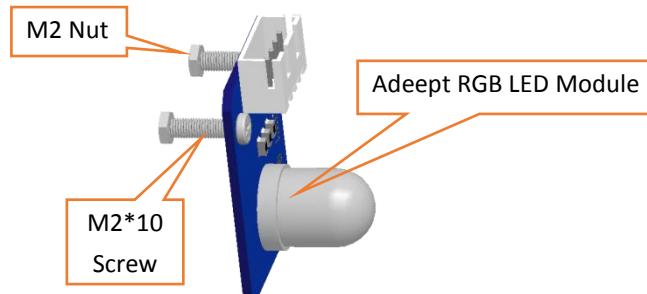
5.5 Mount the part assembled in Step 5.4 to the car body.



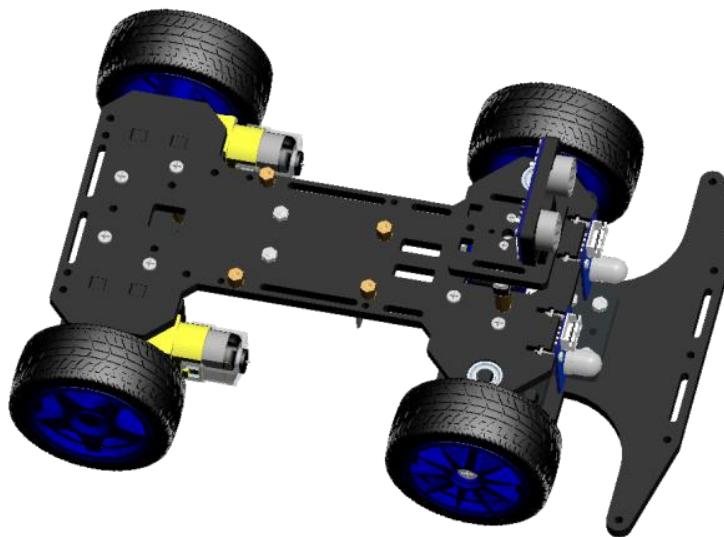
RGB LED Module

Assemble the following components

First assemble the M2*10 screw and M2 nut into the LED module, with the screw just inserted into the nut (as shown in the figure). Then hold the LED, press the nuts into the holes of the module and tighten them.

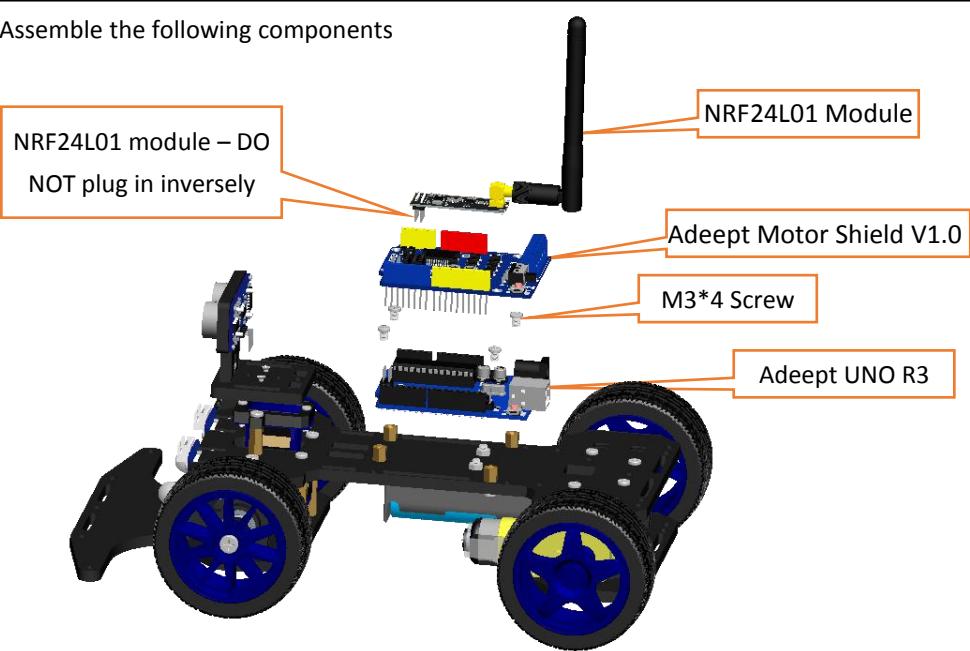


Effect diagram after assembling

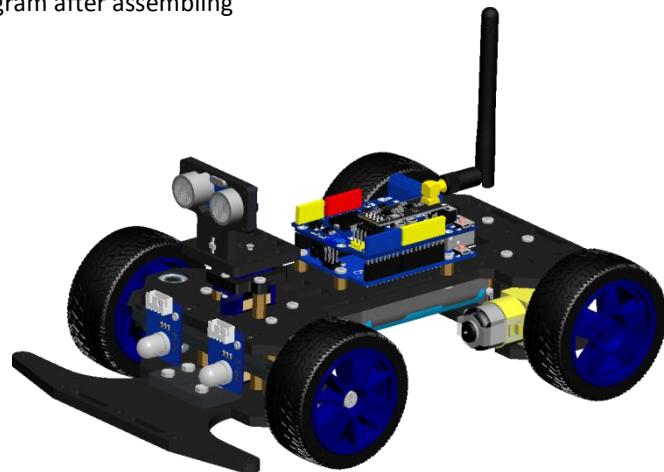


Assemble PCBs (Adeept Uno,NRF24L01 Module.)

Assemble the following components



Effect diagram after assembling

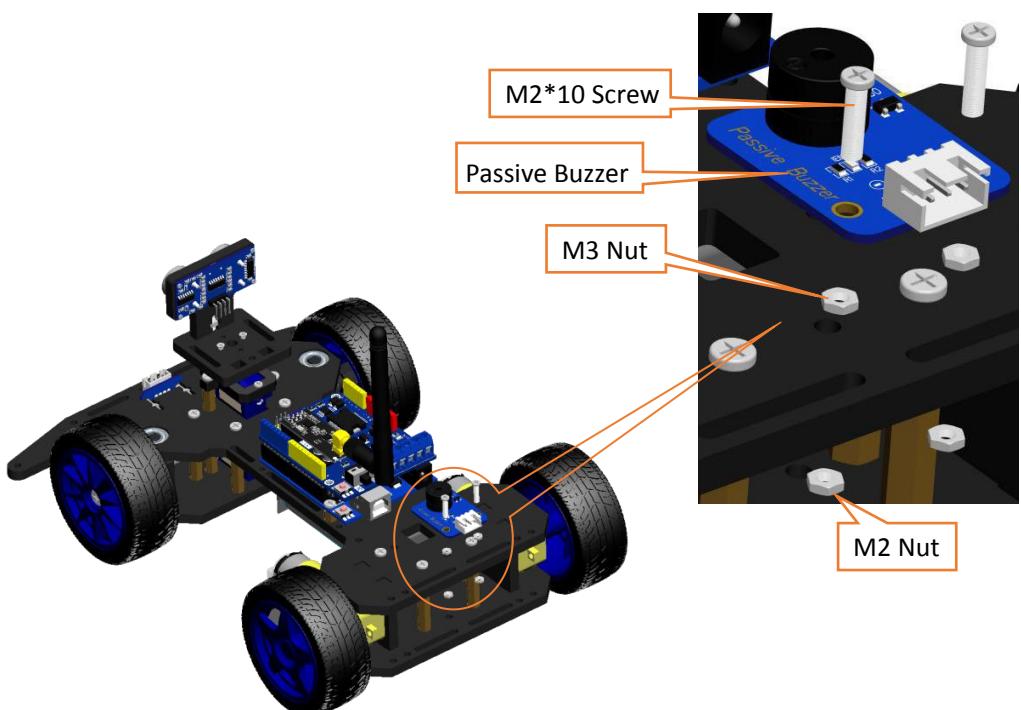


Assemble the Passive Buzzer Module

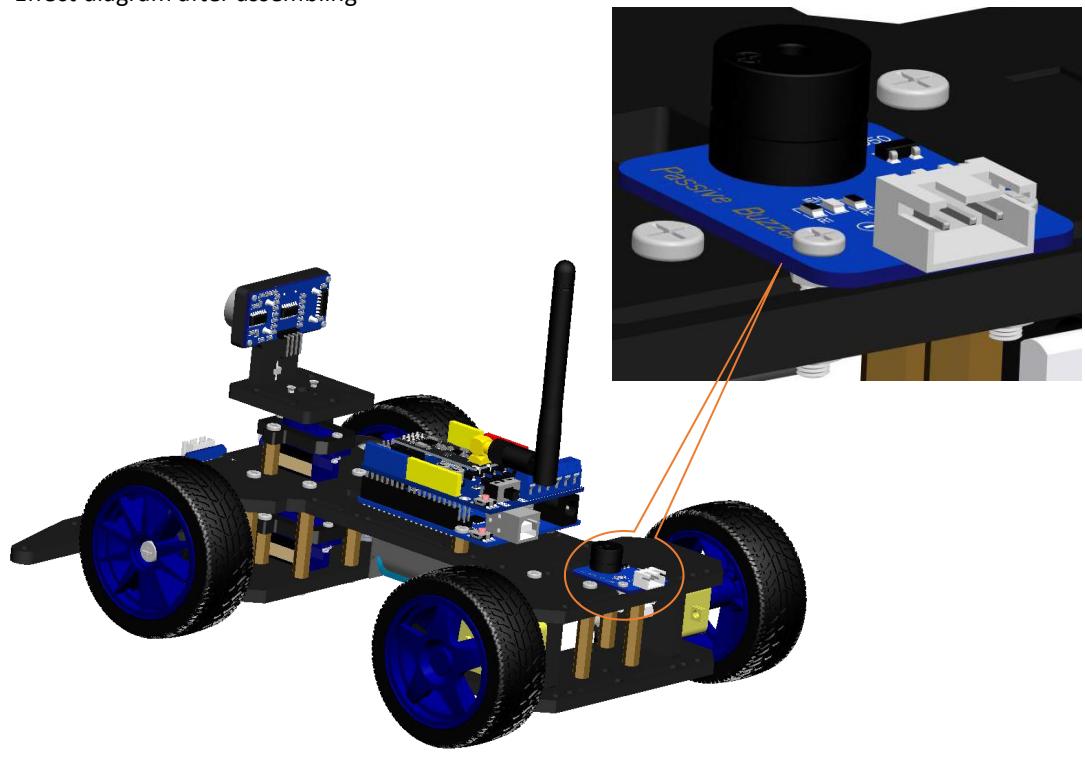
Assemble the following components

Put an M3 nut on the car first, and then insert two M2*10 screws through the buzzer module, M3 nut, the long slot on the plate at the back of the car, and then an M2 nut below the plate in turn. During the course, you can hold the nut underneath, insert the screw through these, and tighten them.

The M3 nut here is to support the module in height.

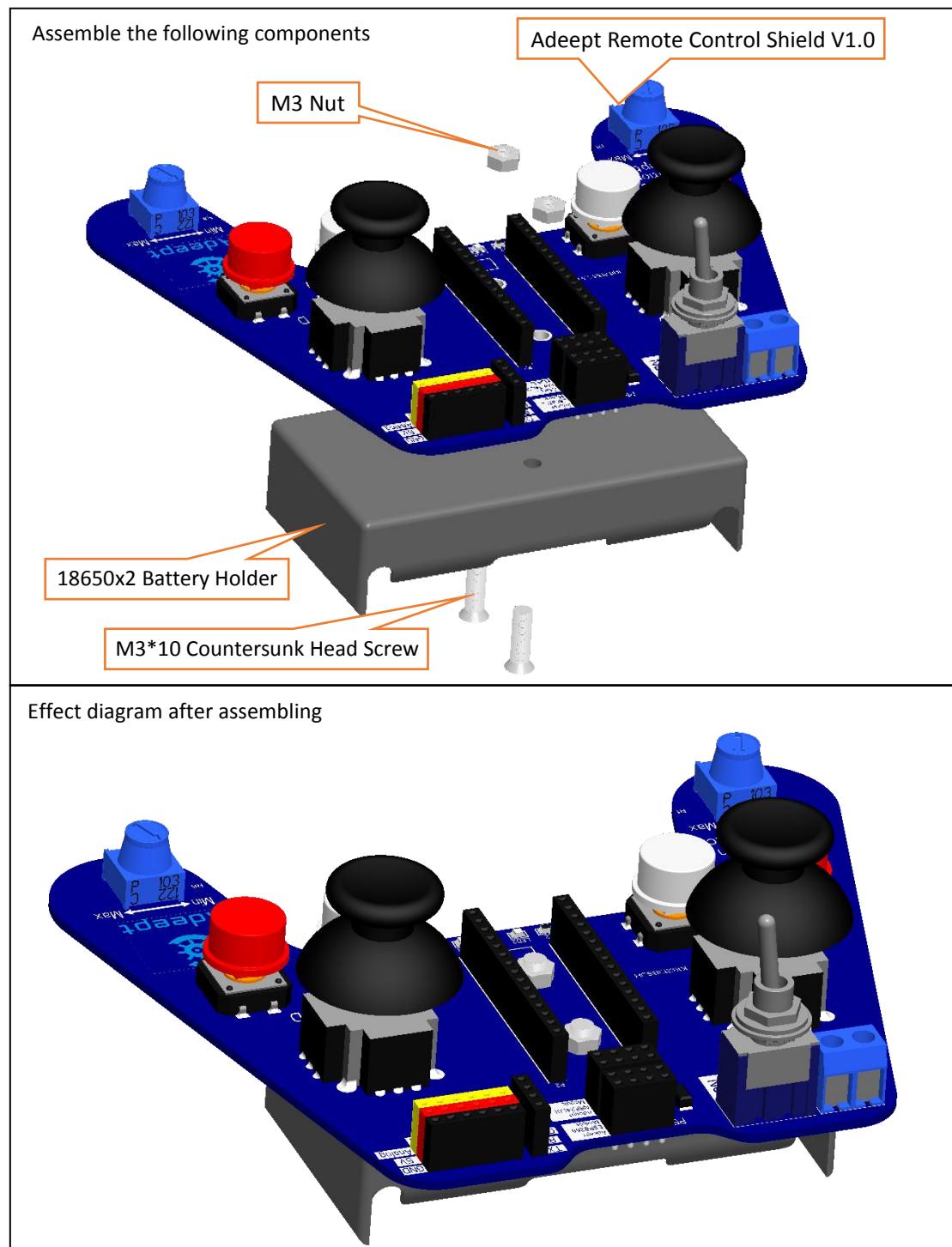


Effect diagram after assembling



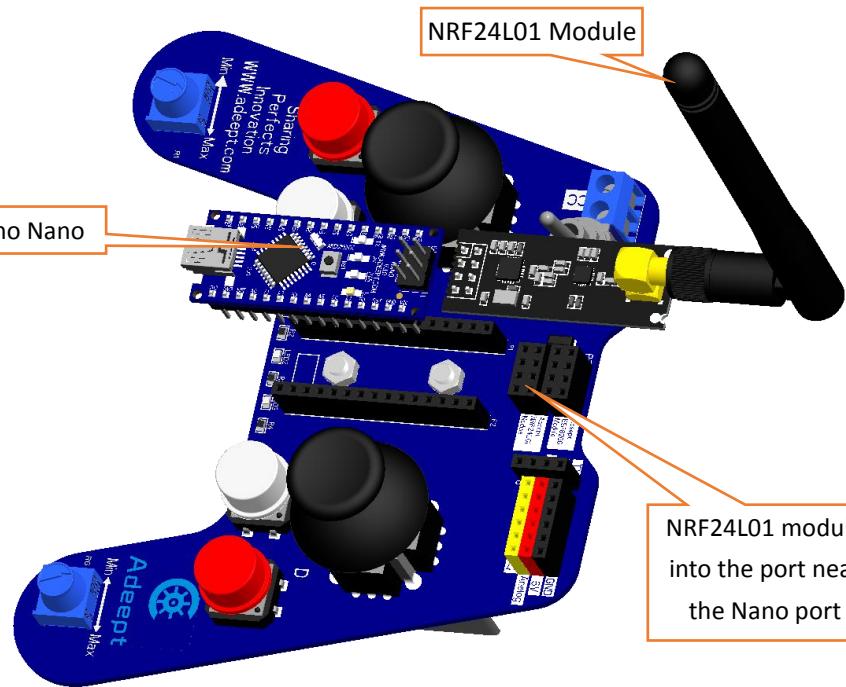
Remote Control

9.1 Fix the 18650x2 Battery Holder and Adeept Remote Control Shield V1.0.

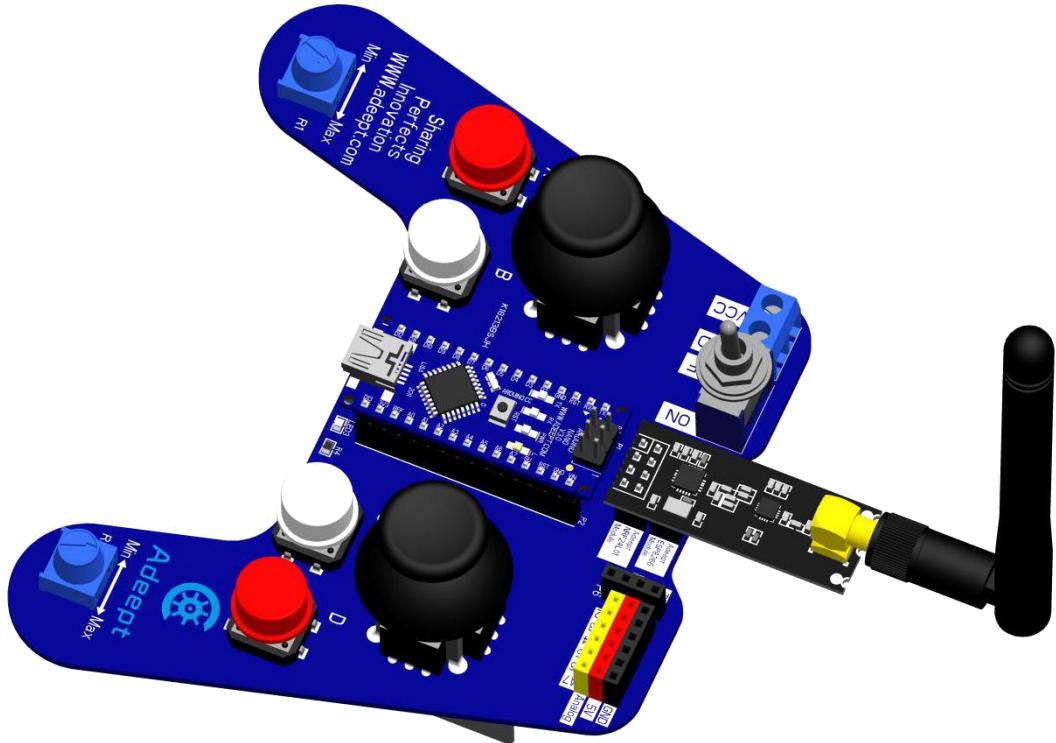


9.2 Plug the NRF24L01 Module and Arduino Nano board onto the Adeept Remote Control Shield V1.0.

Assemble the following components

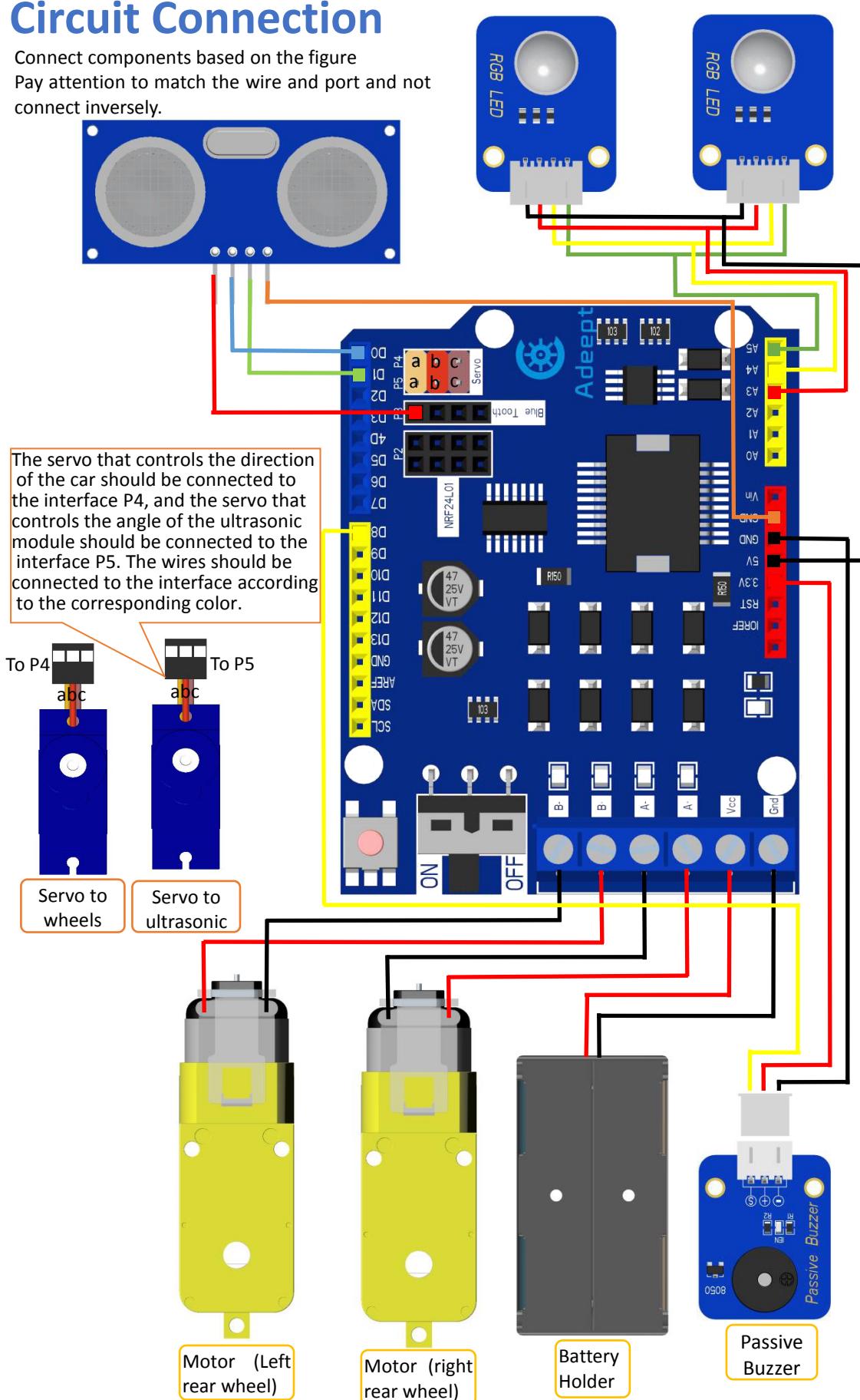


Effect diagram after assembling

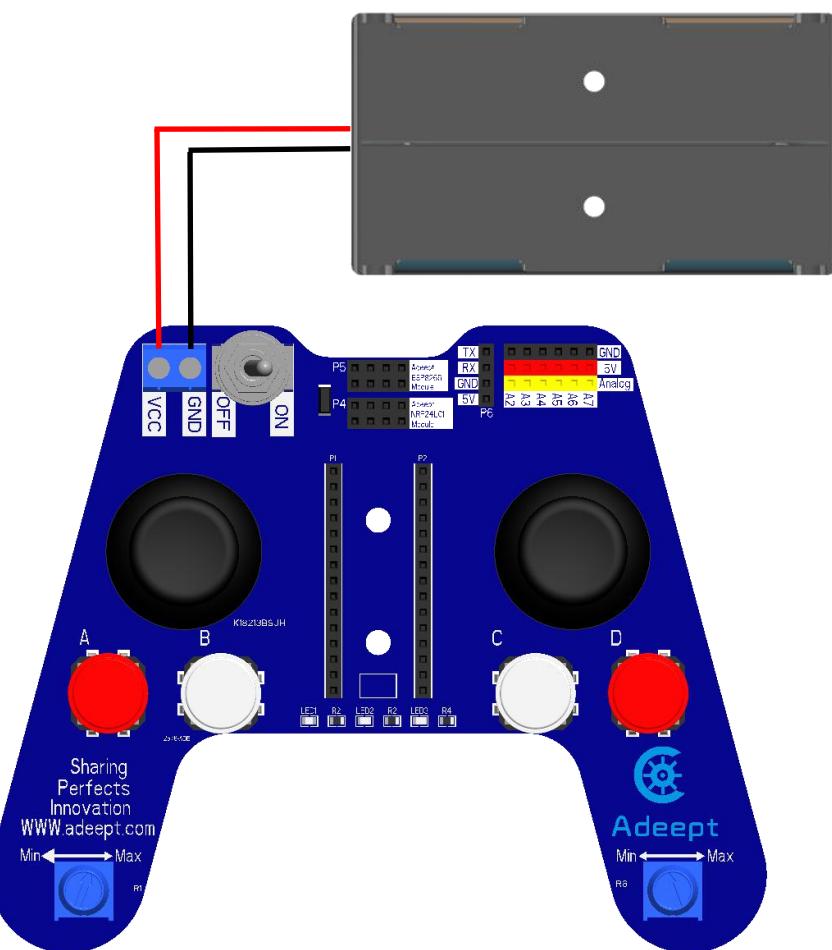


Circuit Connection

Connect components based on the figure
 Pay attention to match the wire and port and not connect inversely.



Circuit Connection of the Remote Control:



Afterword

Thanks for purchasing our product and reading the manual! If you spot any errors or have any ideas or questions for the product and this guide, welcome to contact us! We will correct them if any as quickly as possible.

After completing all projects in the guide, you should have some knowledge of the book and Arduino, thus you can try to change the car into other projects by adding more Adeept modules or changing the code for extended functions.

For more information about Arduino, Raspberry Pi, smart car robot, or robotics, etc., please follow our website www.adeept.com. We will introduce more cost-effective, innovative and intriguing products!

Thanks again for choose Adeept product!