SunFounder DIY_Control_Robot_Arm_kit_for_Arduino-Rollarm

www.sunfounder.com

Aug 02, 2022

CONTENTS

The Rollarm Kit is an interesting and useful learning tool for Arduino and robot hobbyists. With the structural plate and code based on Arduino, it enables users to learn programming from easy to difficult, control the mechanical arm freely and perform various fun operations!

This fun mechanical arm consists of 4 axes, each controlled by a Servo. Powered by a 9V battery, the control systems is composed of SunFounder Uno board, Servo extension board with remote control. The kit includes all necessary components like structural plate, circuit boards, and connector parts.

Now let's go to get the fun!

If you have any questions, please send an email to service@sunfounder.com and we'll reply as soon as possible.

Content

CHAPTER

ONE

COMPONENT LIST AND ASSEMBLY INSTRUCTIONS

Before assembling the Rollarm, please first verify that all parts and components have been included. If there are any missing or damaged components, please contact SunFounder immediately at cs@sunfounder.com to resolve the issue as soon as possible.

Please follow the steps on the following PDF for assembly instructions:

- `[PDF]Component List and Assembly of Rollarm`

Note:

- If the Servo is already powered up, do not force the Servo shaft to avoid damaging the Servo.
- Before assembling, you need to buy 1 9V battery and fully charge them.
- The Expansion Board cannot charge the battery, so you need to buy a battery charger at the same time.

1.1 Self-Provided Components

CHAPTER

TWO

DOWNLOAD THE CODE

We have uploaded the relevant code material to github, you can download it through the link below.

- `Rollarm Code(.zip)`

CHAPTER

THREE

INSTALL AND INTRODUCE THE ARDUINO IDE

Here are the installation steps on the windows system.

For other systems, please refer to: Install Arduino IDE in different and FAQ.pdf

The code in this kit is written based on Arduino, so you need to install the IDE first. Skip it if you have done this.

Now go to arduino.cc and check the software list on the right side.

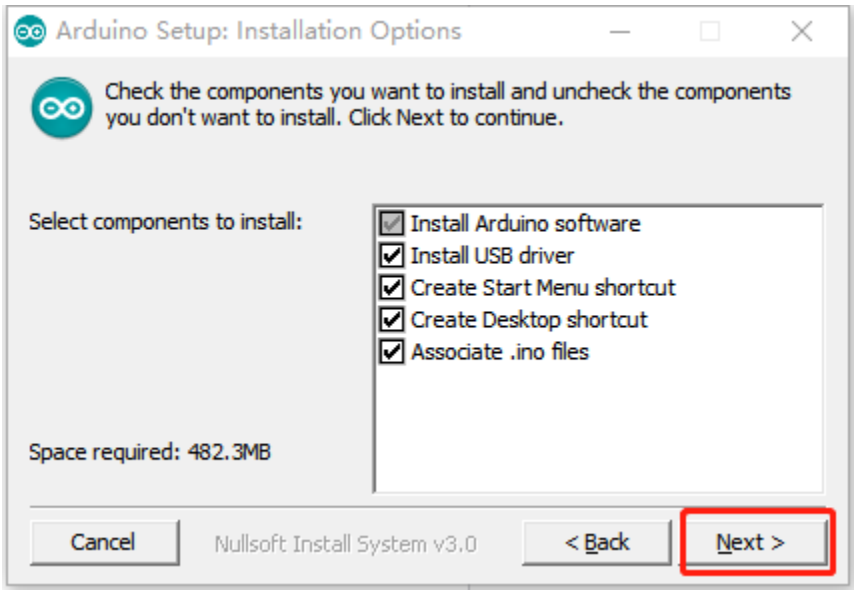
Find the one that suits your operation system and click to download. There are two versions of Arduino for Windows: Installer or ZIP file. You're recommended to download the former.

3.1 For Installer File

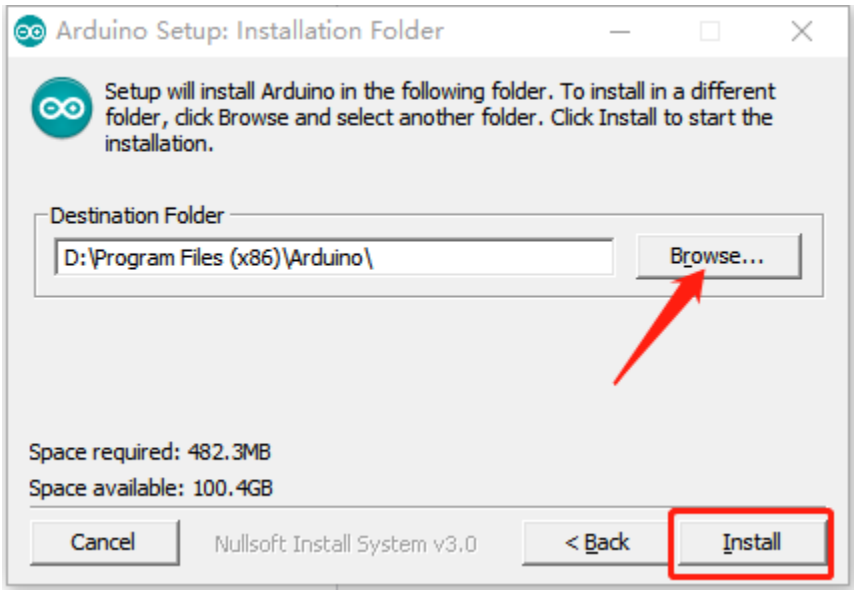
Step 1: Find the .exe file just downloaded.

Step 2: Double click the file and a window will pop up as below. Click **I Agree**.

Step 3: Click **Next.**



Step 4: Select the path to install. By default, it's set in the C disk. You can click **Browse** and choose other paths. Click **OK**. Then click Install.



Step 5: Meanwhile, it will prompts install the needed drivers, please select the 'Always trust software from "Arduino LLC" '. After the installation is done, click **Close**.

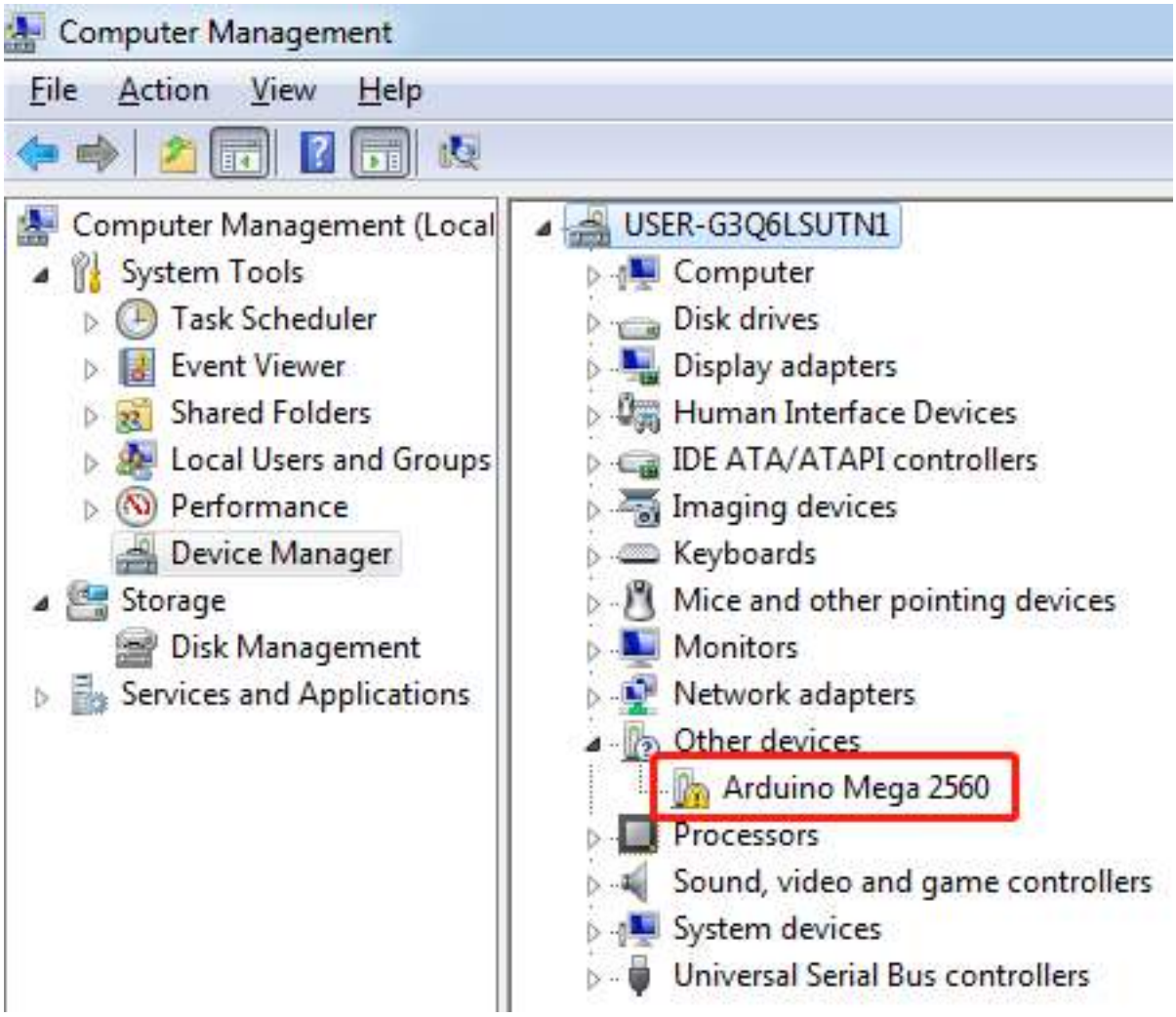
Note: The new IDE may prompt errors when you're compiling code under Windows XP. So if your computer is running on XP, you're suggested to install Arduino 1.0.5 or 1.0.6. Also you can upgrade your computer.

3.2 For ZIP File

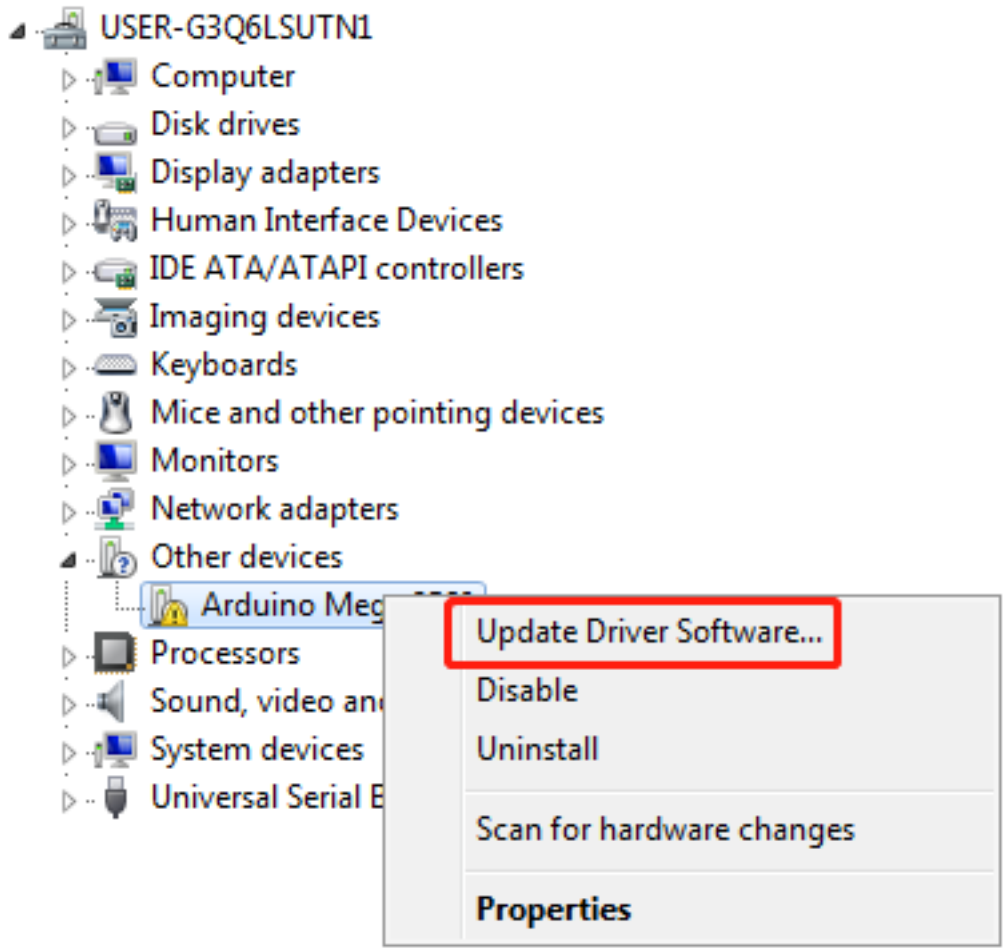
If you download the zip file before, when you connect the MCU to the computer, it may not be recognized. Then you need to install the driver manually. Take the following steps.

Step1: Plug in the board to the computer with a 5V USB cable. After a while, a prompt message of failed installation will appear.

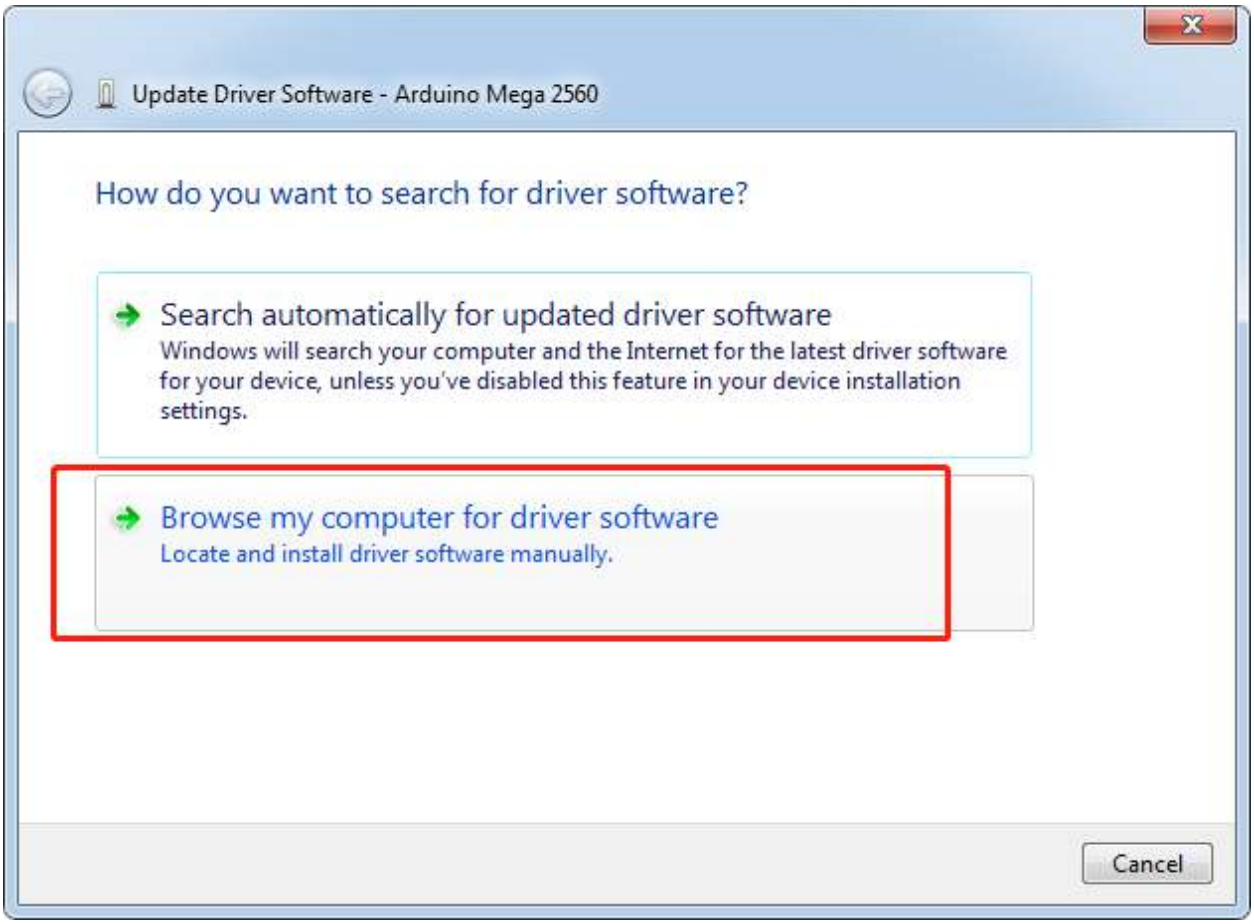
Step2: Go to the **Device Manager**. You will find under other devices, Arduino Mega 2560 with an exclamation mark appear, which means the computer did not recognize the board.



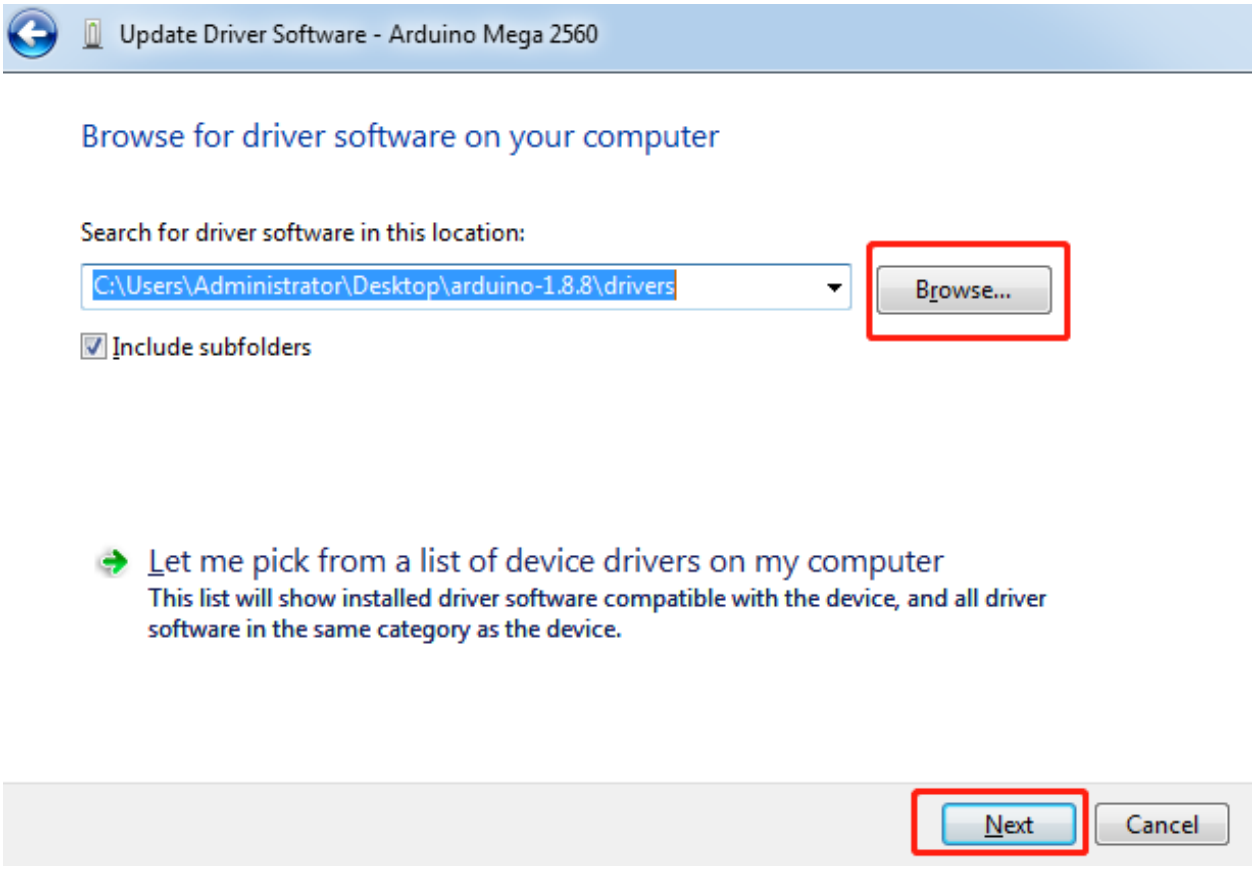
Step3: Right click on **Arduino Mega 2560** and select **Update Driver Software**.



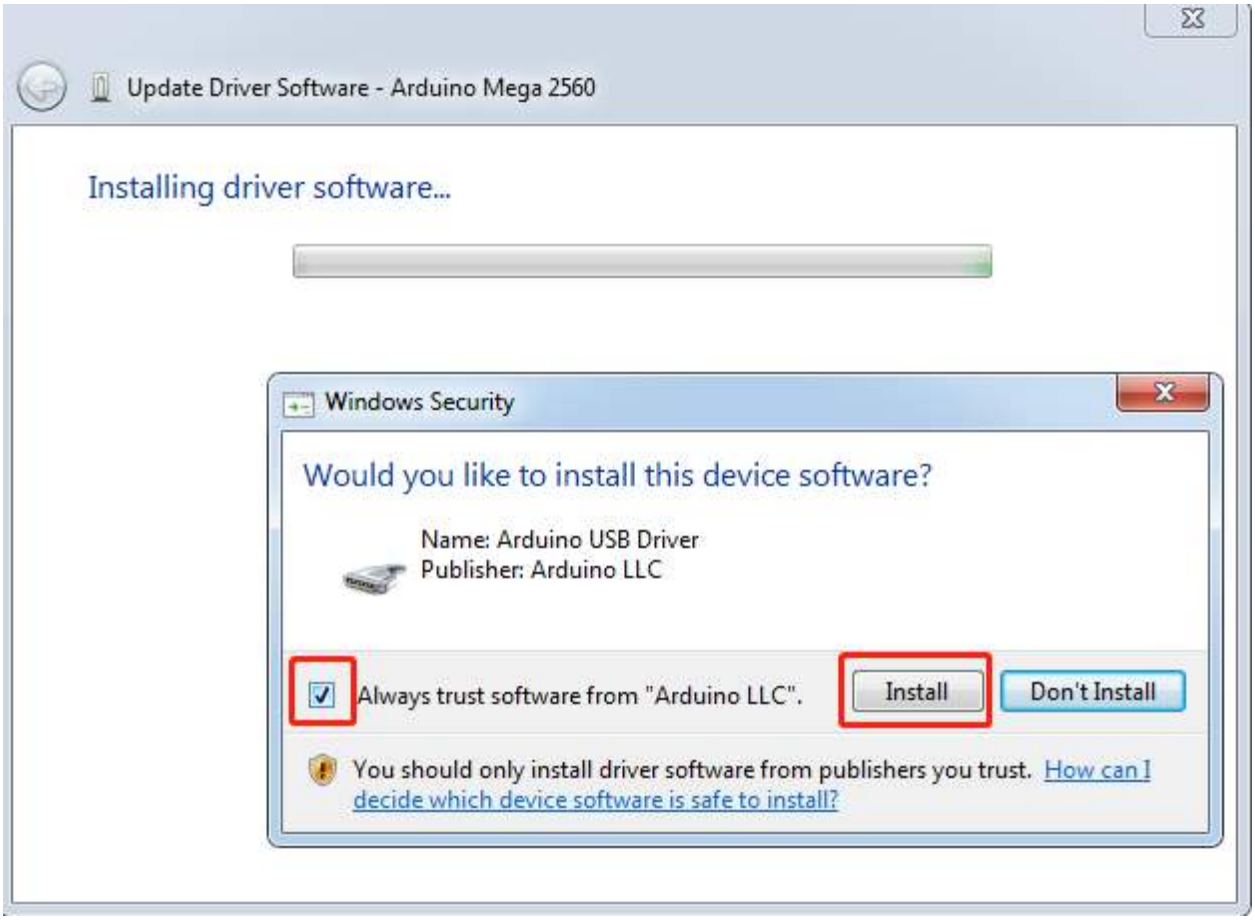
Step4: Choose the second option, **Browse my computer for Driver software**.



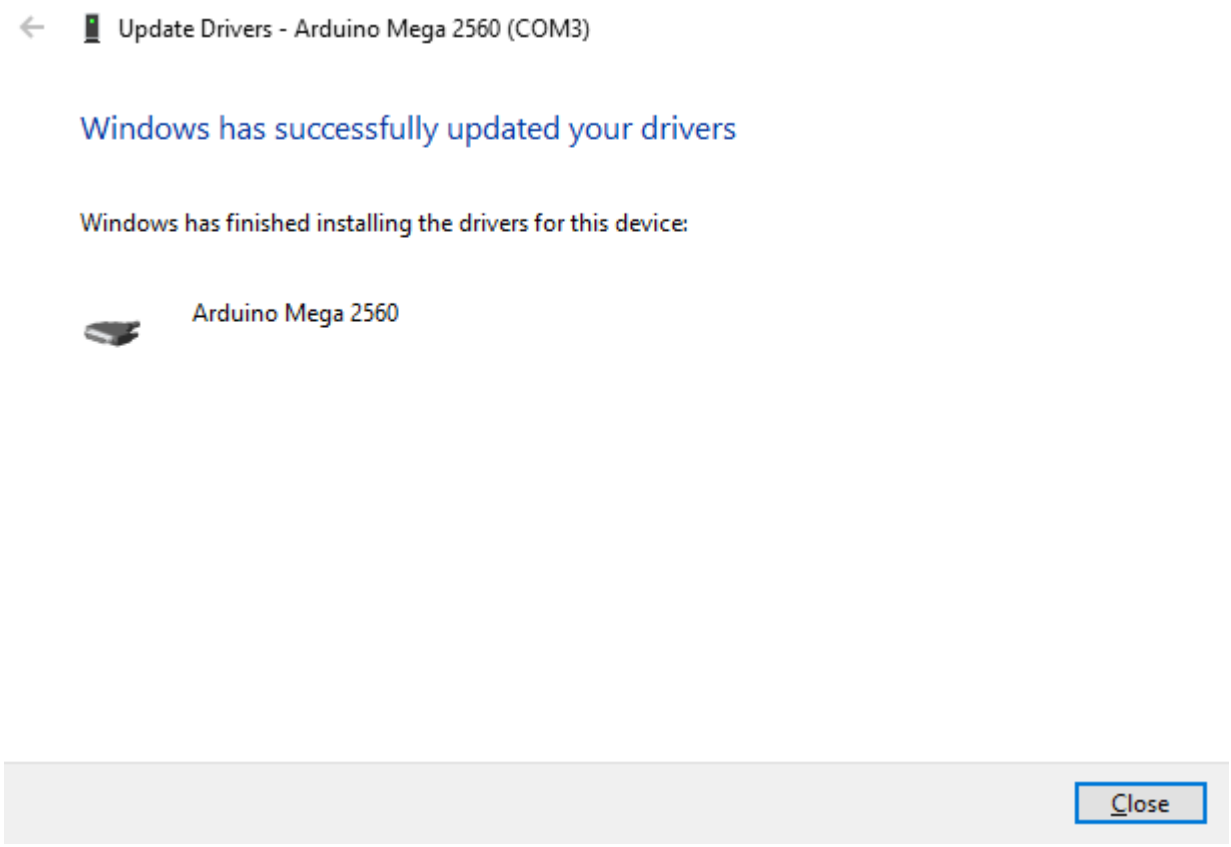
Step5: A window pops up then. Click **Browse**. Then go to the folder where you just extracted the file. Go to the **drivers** folder and click **OK** -> **Next**.



Step6: Select 'Always trust software from "Arduino LLC" ' then click Install.



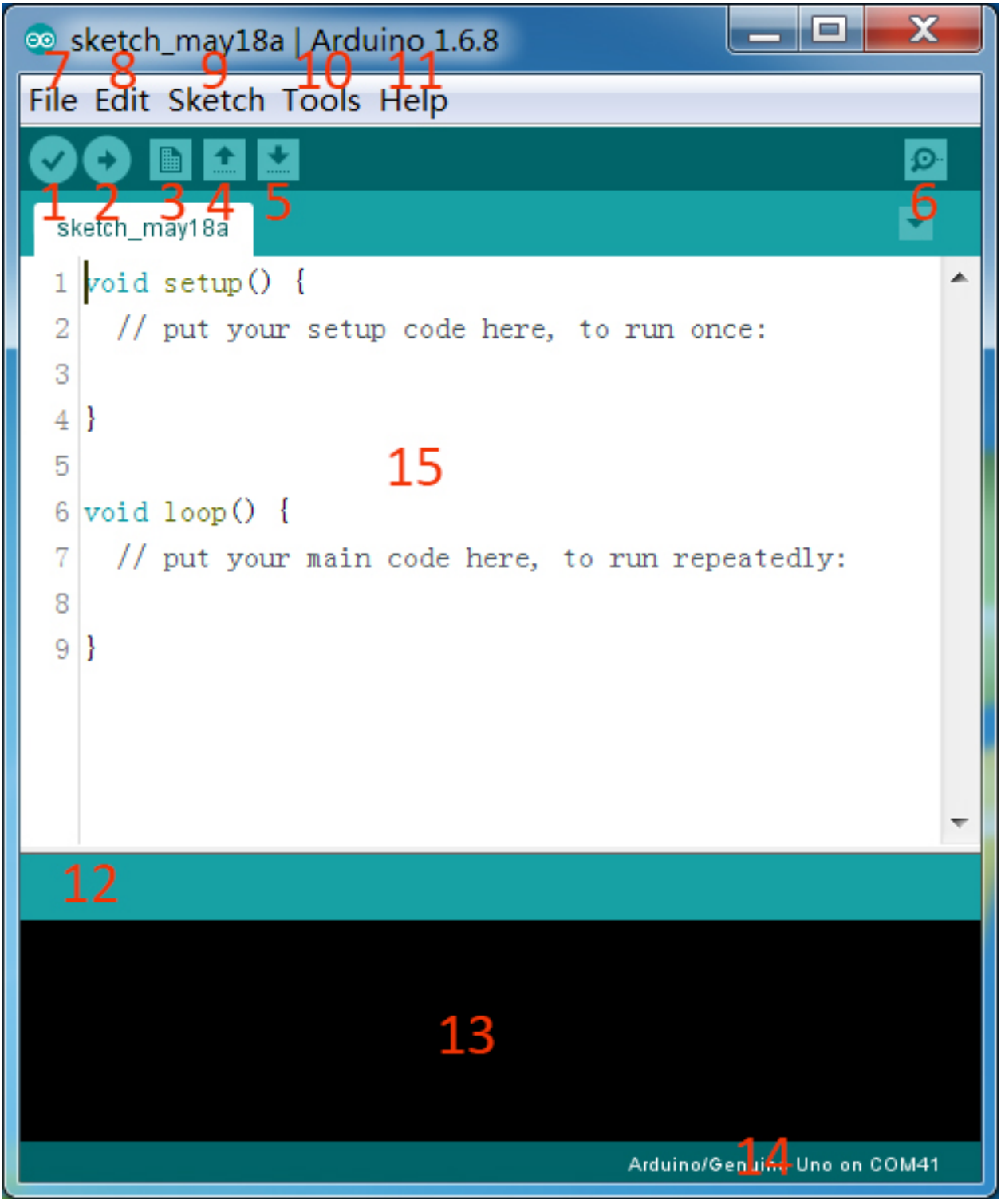
It may need a sec. Then the system prompts you the driver has been installed successfully. So the computer can recognize the board now. Click **Close**.



3.3 Open the Arduino Software (IDE)

Double-click the Arduino icon (arduino.exe) created by the installation process

Then the Arduino IDE will appear. Let's check details of the software.



1. **Verify**: Compile your code. Any syntax problem will be prompted with errors.

2. **Upload**: Upload the code to your board. When you click the button, the RX and TX LEDs on the board will flicker fast and won't stop until the upload is done.

3. **New**: Create a new code editing window.

4. **Open**: Open an .ino sketch.

5. **Save**: Save the sketch.

6. **Serial Monitor**: Click the button and a window will appear. It receives the data sent from your control board. It is very useful for debugging.

7. **File**: Click the menu and a drop-down list will appear, including file creating, opening, saving, closing, some parameter configuring, etc.

8. **Edit**: Click the menu. On the drop-down list, there are some editing operations like Cut, Copy, Paste, Find, and so on, with their corresponding shortcuts.

9. **Sketch**: Includes operations like Verify, Upload, Add files, etc. More important function is Include Library – where you can add libraries.

10. **Tool**: Includes some tools – the most frequently used Board (the board you use) and Port (the port your board is at). Every time you want to upload the code, you need to select or check them.

11. **Help**: If you're a beginner, you may check the options under the menu and get the help you need, including operations in IDE, introduction information, troubleshooting, code explanation, etc.

12. In this message area, no matter when you compile or upload, the summary message will always appear.

13. Detailed messages during compile and upload. For example, the file used lies in which path, the details of error prompts.

14. **Board and Port**: Here you can preview the board and port selected for code upload. You can select them again by **Tools -> Board / Port** if any is incorrect.

15. The editing area of the IDE. You can write code here.

CHAPTER

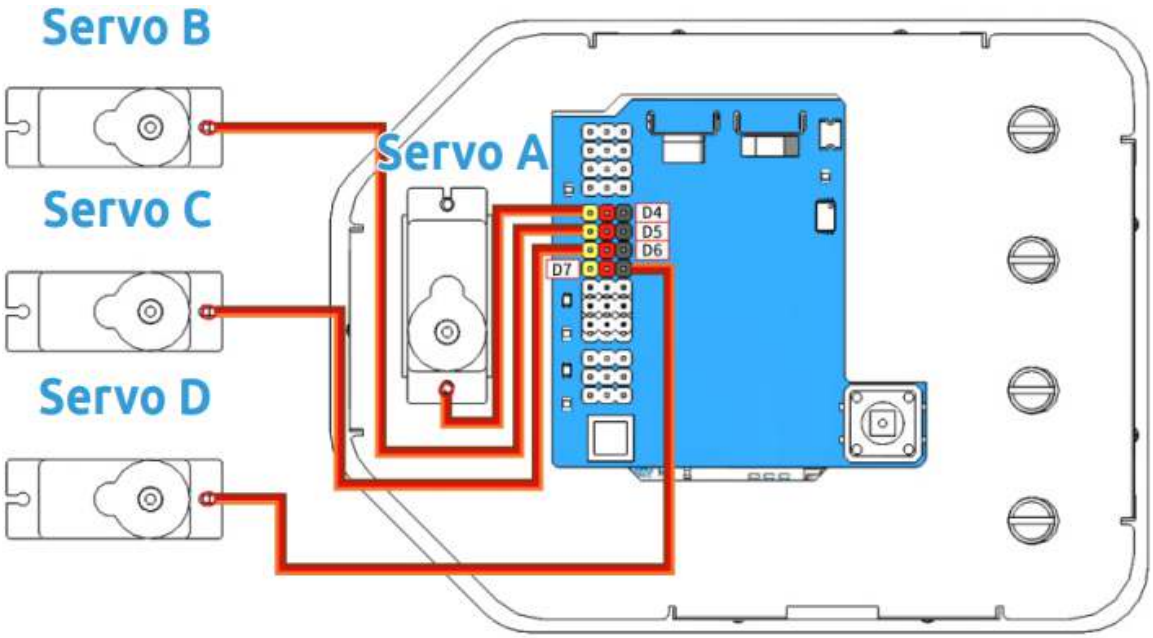
FOUR

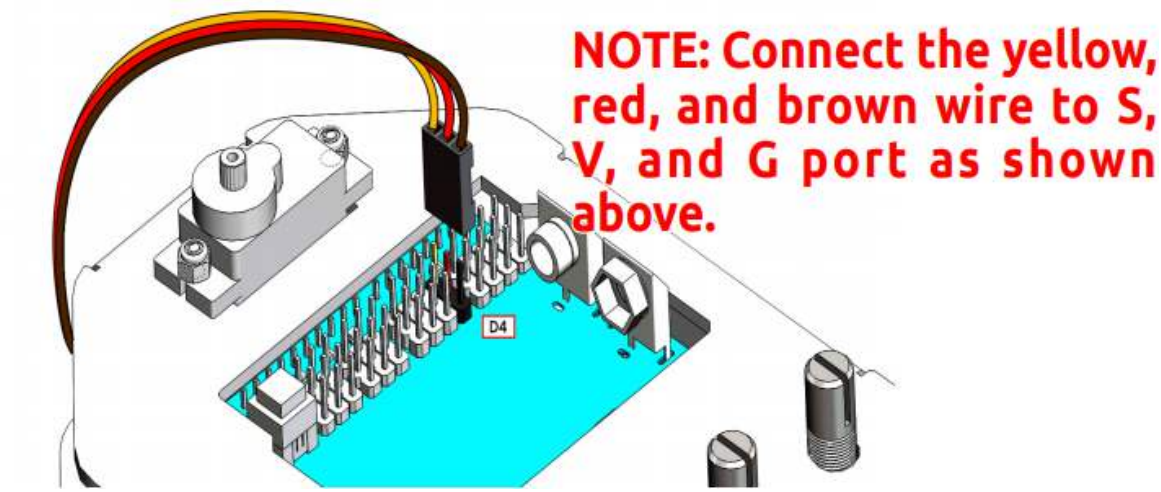
CALIBRATE THE SERVO ANGLE

Warning: You will need to write a specific angle to the Servo before you install the Servo arm for each Servo. If you skip this step, complete the assembly and power up the Servos, they may become blocked or damaged, and you may even need to disassemble the robot and reassemble the Servo parts.

Step 1: Take out the 4 Servos and label them A, B, C and D. Connect them to D4-D7 to distinguish them later.

- Servo A is connected to D4
- Servo B is connected to D5
- Servo C is connected to D6
- Servo D is connected to D7

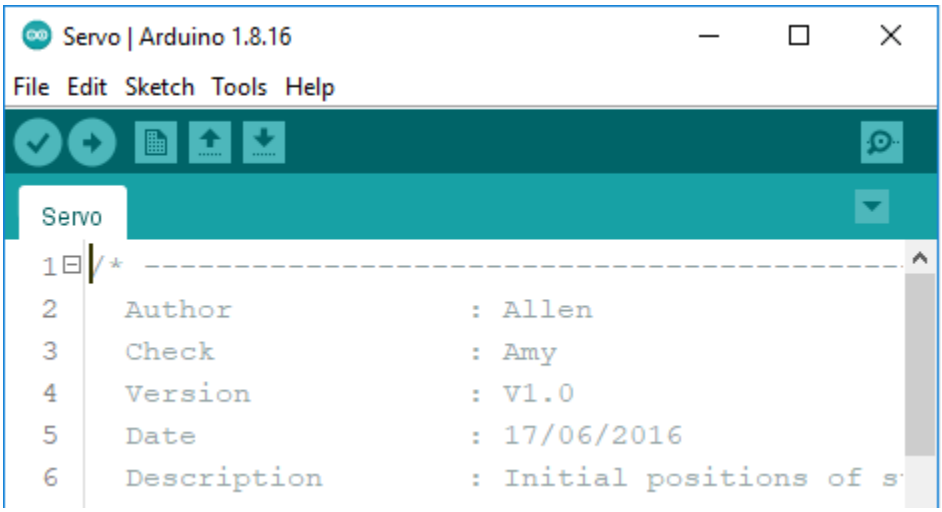




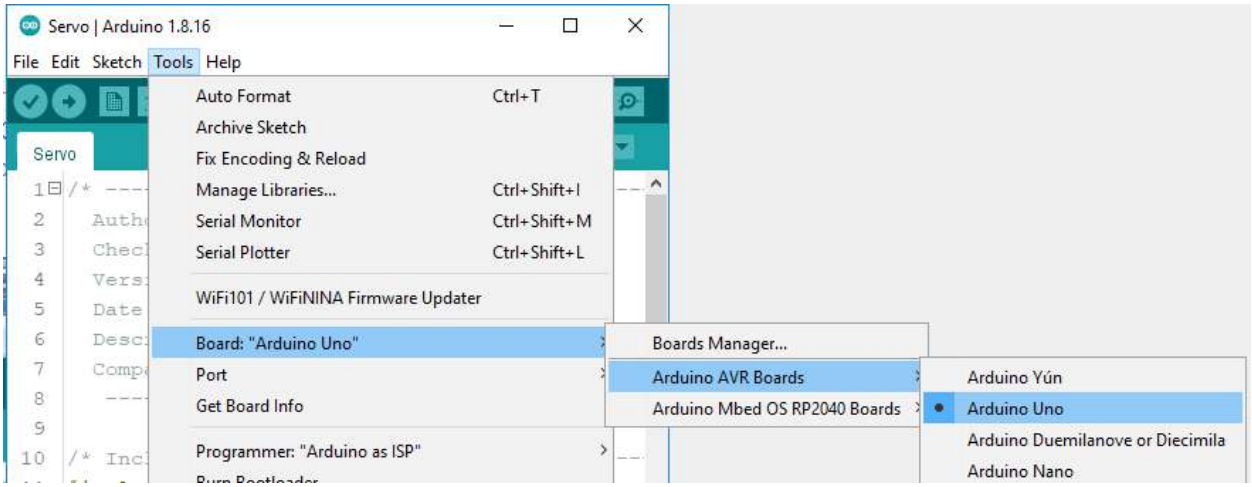
Now start uploading code to SunFounder R3 Board, this step is a one-time action, you don't need to upload the code again during the assembly process.

Step 2: Connect the SunFounder R3 Board to your PC with the USB Type B cable that comes with the kit, the computer will usually recognize your board automatically and finally assign a port.

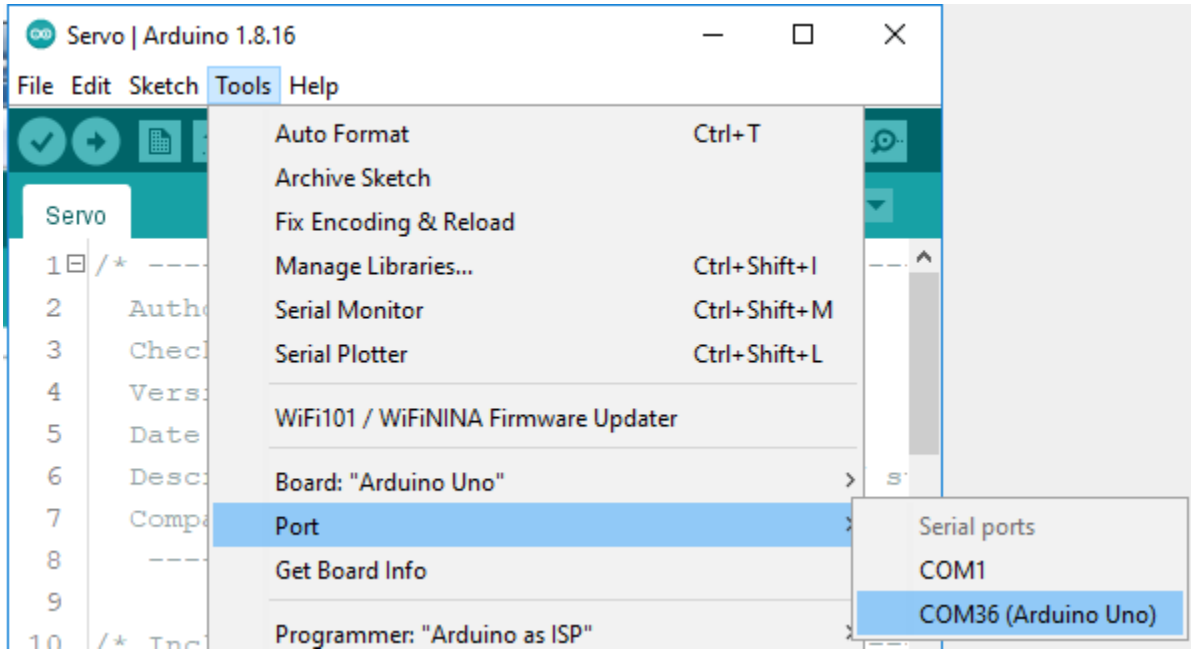
Step 3: Go to the folder `DIY Control Robot Arm kit for Arduino-Rollarm/Arduino Code/Servo` and open the file `Servo.ino`, make sure you have *Download the Code*.



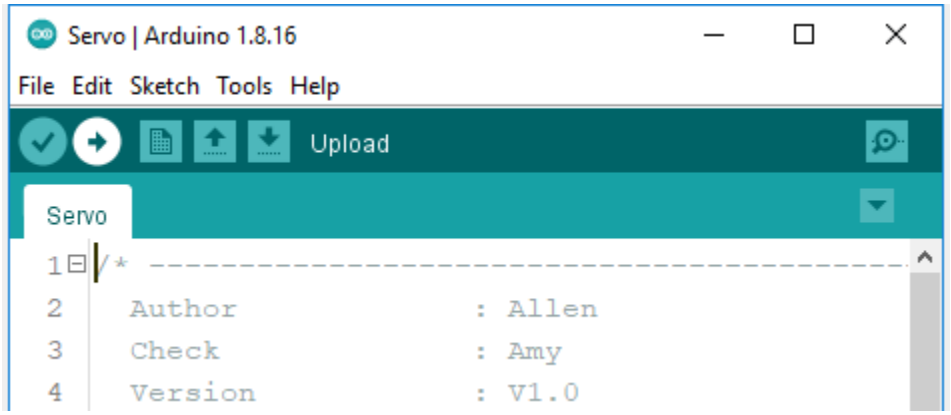
Step 4: Select the **Board**.



and **Port**, your port may not be the same as mine.



Step 5: Click **Upload.**



After the upload is completed successfully, you may hear the sound of gear moving (or may not, if the Servo shaft

happens to be at 90 degrees at the beginning; but you GENTLY spin the Servo arm and you'll find it's unmovable). So now the Servo is adjusted to 90 degrees.

Note:

- After uploading the code this time, you can unplug the Type B cable and use the battery directly or plug the Type B cable directly into the power socket, the code on the SunFounder R3 board will run again, no need to burn the code from Arduino IDE again.
- Don't force to turn the Servo when power is on, so as not to damage it.
- It is recommended that when assembling the Servo, you can unplug the other Servo cables so that you can have a better mounting position.

Now you can continue assembling Rollarm.

CHAPTER

FIVE

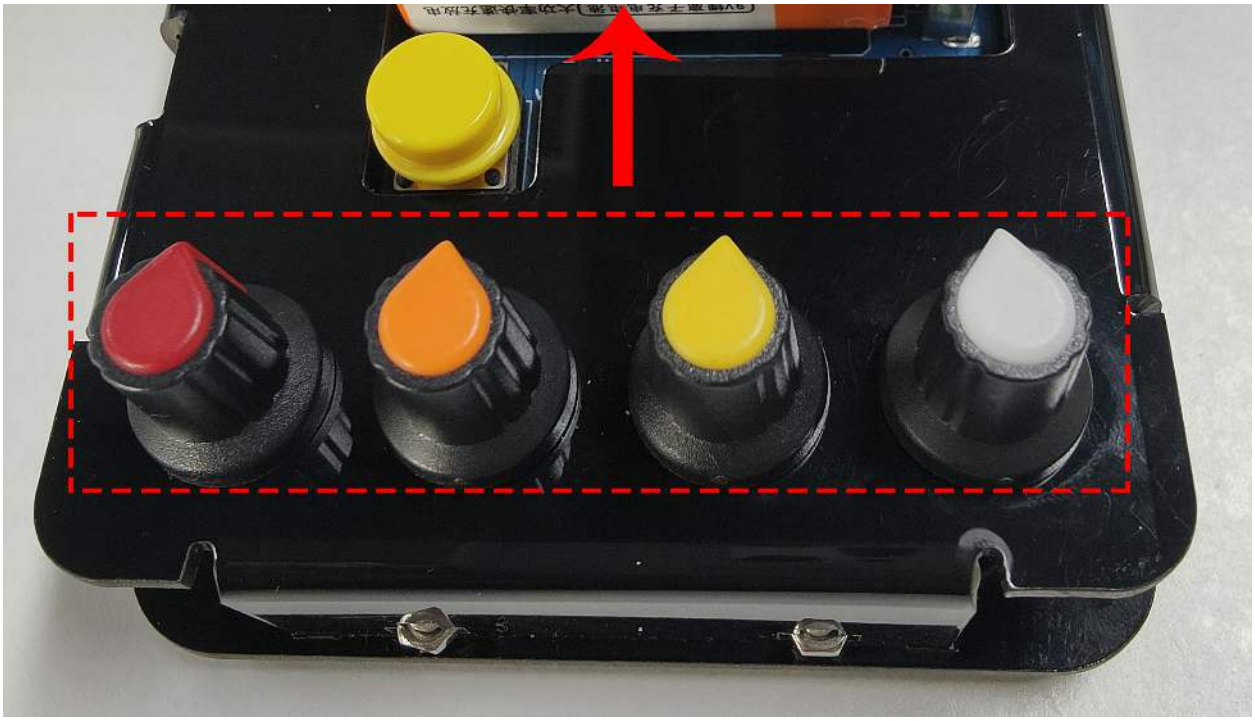
CONTROL THE ROLLARM

There are two ways to control the Rollarm: manual control (by handle), or PC control (by Labview). The detailed operations for two ways are as follows.

5.1 Manual Control

Step 1

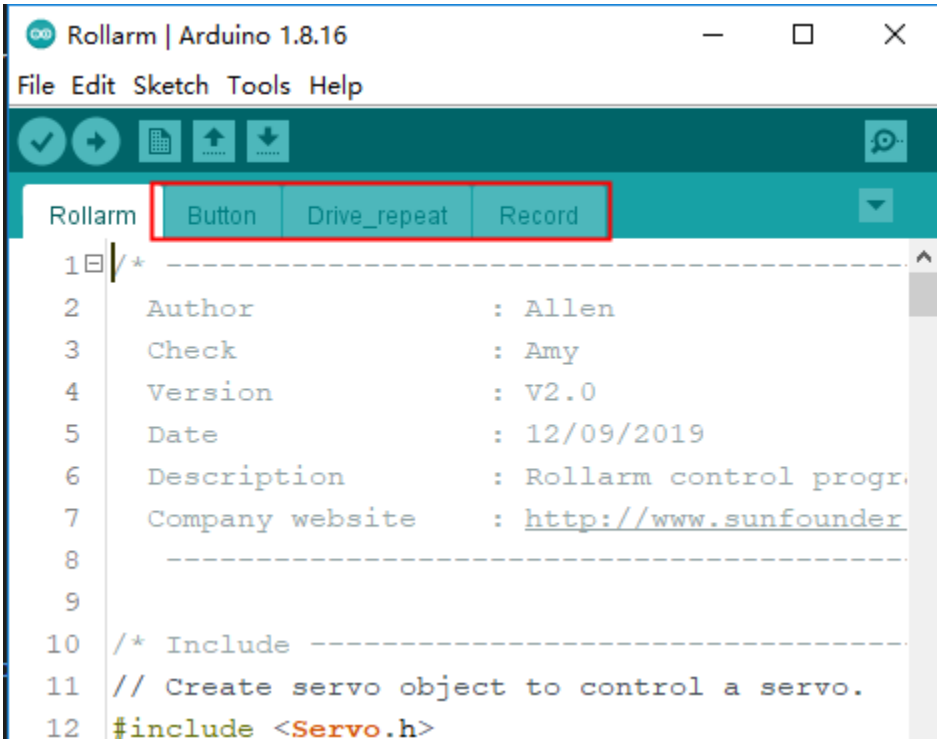
First rotate the 4 potentiometers of the Rollarm to the forward facing position, in order to keep the Rollarm as it was just assembled after uploading the code below.



Step 2

Run the `Rollarm.ino` code under the path `DIY Control Robot Arm kit for Arduino-Rollarm\\Arduino Code\\Rollarm`. There are four code files in Rollarm, `Rollarm.ino` is the main program, when the others are subprograms.

When you open the main program, the subprograms will be opened automatically



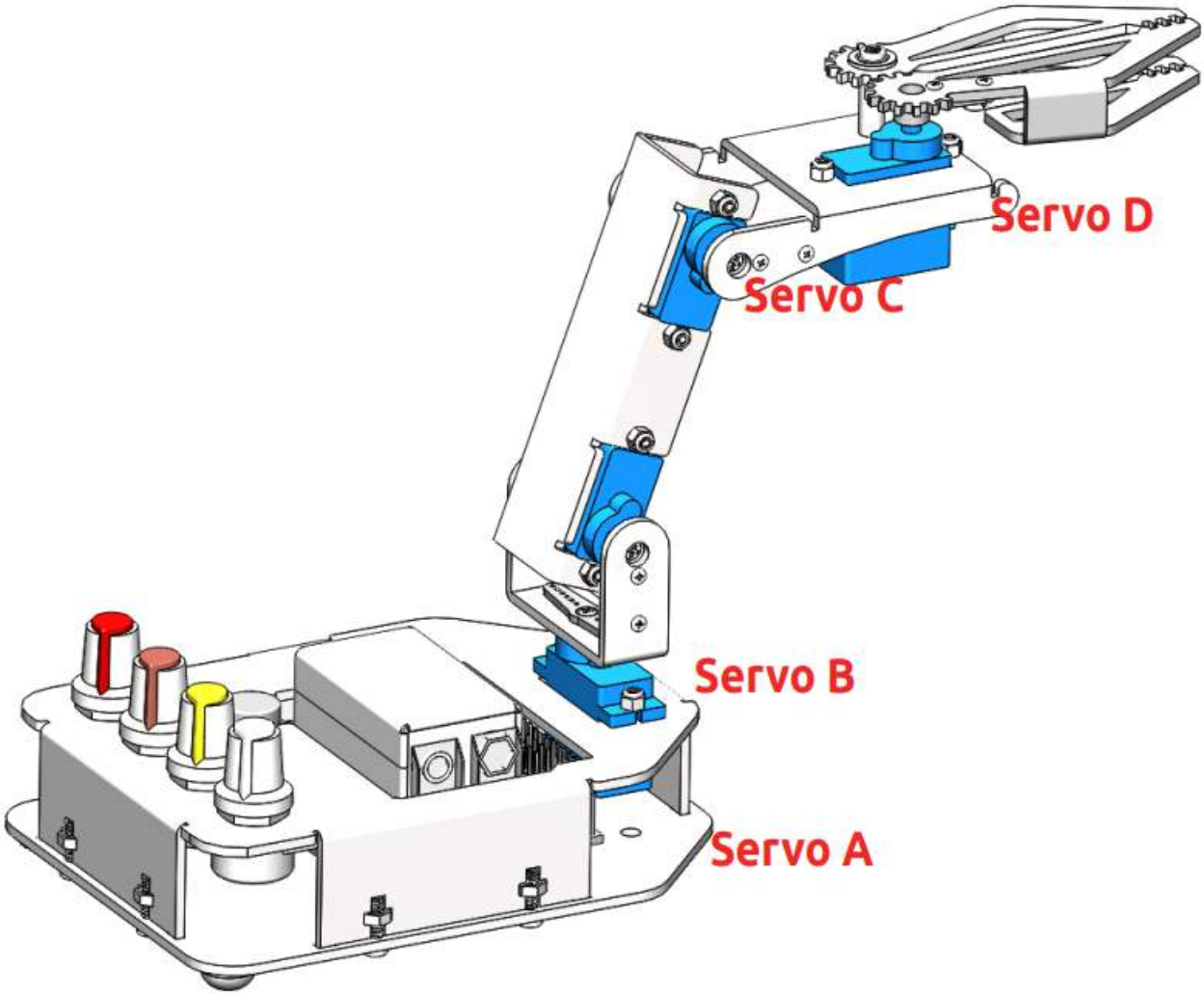
Step 3

Select the corresponding board and port, then click **Upload**.

Step 4

After the code upload, turn the power switch on, then we can try to control the Rollarm.

- **White** potentiometer to control the **Servo D**
- The **yellow** potentiometer to control **Servo C**
- The **orange** potentiometer to control **Servo B**
- The **red** potentiometer to control **Servo A**



5.1.1 Record behavior

Also `Rollarm.ino` has the function of recording actions, now look at how to use this function.

- Rotate a potentiometer to control a Servo to the desired position.
- Record this step by pressing the **yellow button** shortly.
- You can record as many steps as you like, up to a maximum of 100.
- Now press and hold the **yellow button** to get Rollarm to resume the recorded steps.

5.1.2 Code Explanation

Since the Rollarm has four Servos acting as the moving joint, we need to include a header file for driving the Servos and define them.

```
//Create Servo object to control a Servo.
#include <Servo.h>

Servo Servo_0;
Servo Servo_1;
Servo Servo_2;
Servo Servo_3;
```

After defining the function of driving the Servos, we need to read the analog value of the potentiometers and convert them into the rotating angle of the Servos.

```
//Read the values ot the potentiometers.
void ReadPot()
{
        SensVal[0] = 0;
        SensVal[1] = 0;
        SensVal[2] = 0;
        SensVal[3] = 0;

        SensVal[0] = analogRead(A0);
        SensVal[1] = analogRead(A1);
        SensVal[2] = analogRead(A2);
```

(continues on next page)

(continued from previous page)

```
        SensVal[3] = analogRead(A3);
}

//The value of the potentiometer is matched to the angle value.
void Mapping0()
{
        SensVal[0] = map(SensVal[0], 0, 1023, 10, 170);
        SensVal[1] = map(SensVal[1], 0, 1023, 10, 170);
        SensVal[2] = map(SensVal[2], 0, 1023, 10, 170);
        SensVal[3] = map(SensVal[3], 0, 1023, 100, 180);
}
```

After compiling the program, we need to make Rollarm remember the steps, which is done through pressing the button.

```
//Calculate the time the button pressed
void Button()
{
        if (digitalRead(3) == 0)
        {
                delay(10);
                if (digitalRead(3) == 0)
                {
                        KeyValue = 0;
                        while (!digitalRead(3))
                        {
                                KeyValue++;
                                delay(100);
                        }
                }
        }
}
```

We can tell which part of the code the Rollarm is performing by reading the value upon pressing the button. When the value is larger than 10, it means Rollarm is repeating the steps. When it is between 0 and 10, it means Rollarm is remembering. And when it is 0, it means Rollarm is being controlled by the potentiometers. The specific program is as follows:

```
//Check the button.
static int Flag = 1;
Button();

//The time of pressing the button is not long then record the action.
if ((KeyValue < 10) && (KeyValue > 0))
{
        KeyValue = 0;
        Record();
        Mapping1();
}
//Long press the button and open the auto mode,start repeating the action.
else if (KeyValue > 10)
{
        if (Flag == 1)
        {
                Flag = 0;
                Calculate();
        }
```

(continues on next page)

(continued from previous page)

```
        Drive_init();
        delay(3000);
        for (int i = 1; i < Time; i++)
        {
                Drive_repeat(i);
                delay(500);
        }
}
//Did not press the button, open the manual mode.
else
{
        ReadPot();
        Mapping0();
```

Next, we are going to call the function to write the value of the Servo rotating angle. However, it is not merely about writing the values directly; the difference between two adjacent rotating values will also be written into the Servos. Here we take a Servo program for example.

```
//The first axis.
if (Dif0[n] > 0)
{
        for (int j = Joint0[n - 1]; j <= Joint0[n]; j++)
        {
                Servo_0.write(j);
                delay(10);
        }
}
else
{
        for (int j = Joint0[n - 1]; j >= Joint0[n]; j--)
        {
                Servo_0.write(j);
                delay(10);
        }
}
```

5.2 PC Control

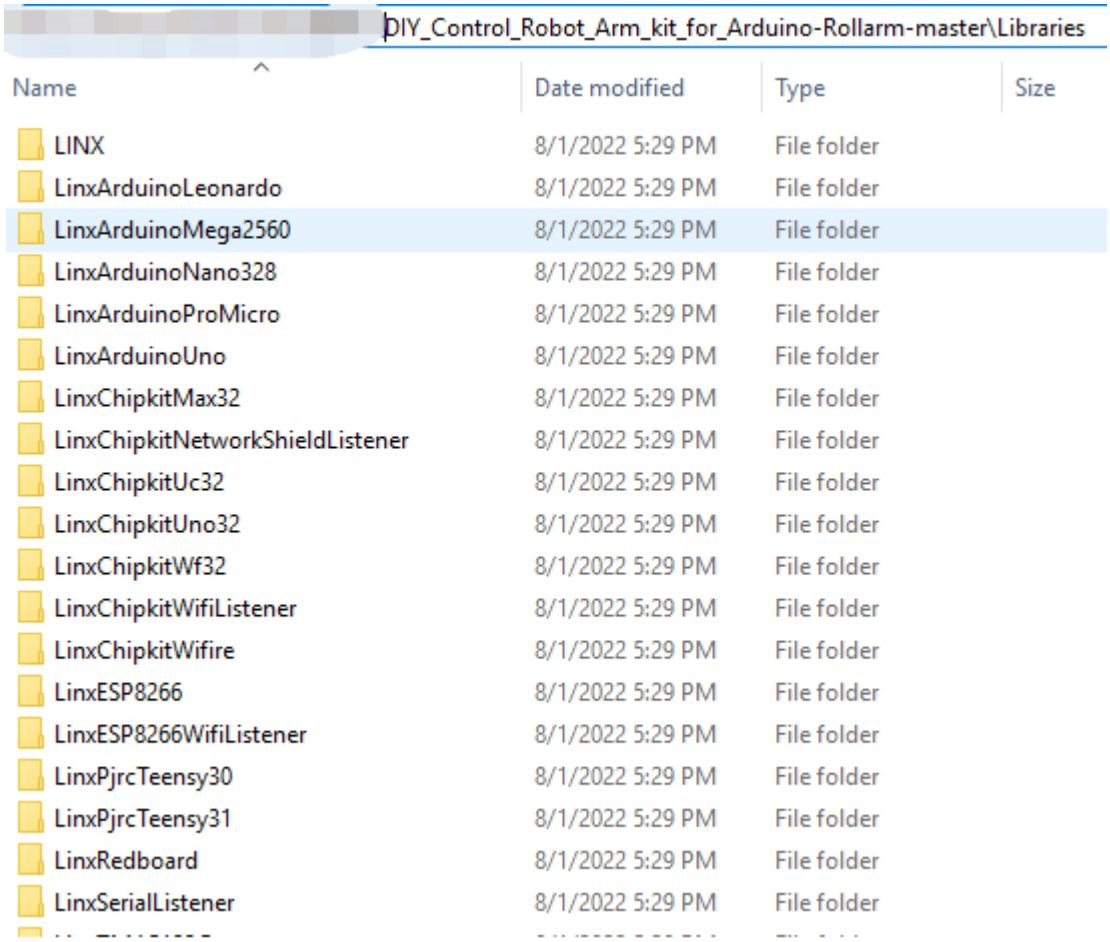
Rollarm can be controlled from your computer using a desktop application generated by LabVIEW.

Using this app, you can control Rollarm to different postures and see the corresponding angle of each servo.

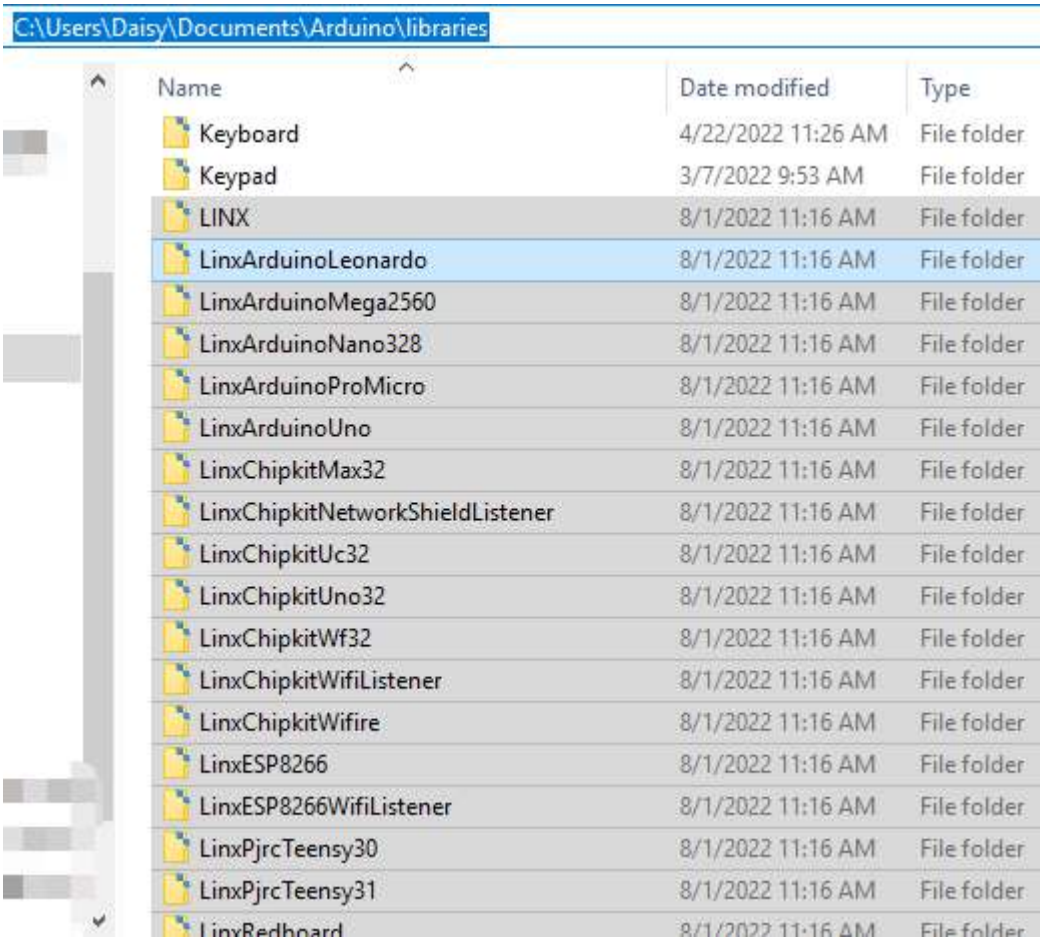
5.2.1 1. Add the Libraries

Here we need to add LINX related libraries to `Arduino/libraries`. LabVIEW LINX is an add-on to LabVIEW that adds the tools for development with Raspberry Pi, BeagleBoard and Arduino.

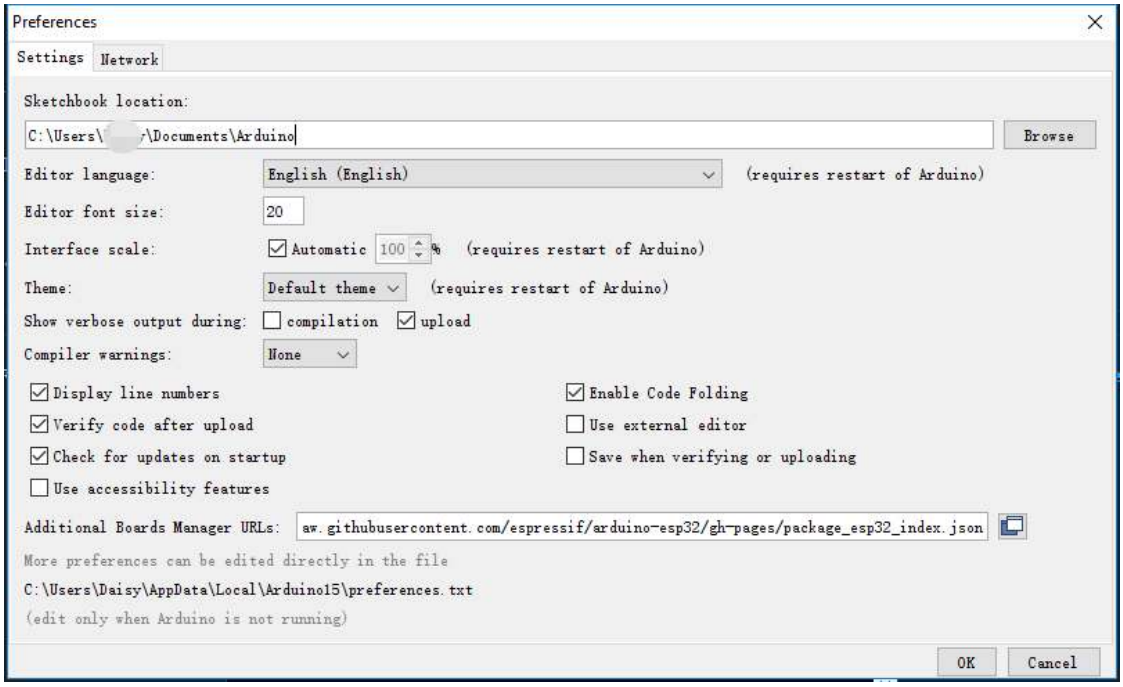
Go to the `DIY_Control_Robot_Arm_kit_for_Arduino-Rollarm\Libraries` path and copy all the folders.



Paste all the folders you just copied into the `C:\Users\xxx\Documents\Arduino\libraries` path.



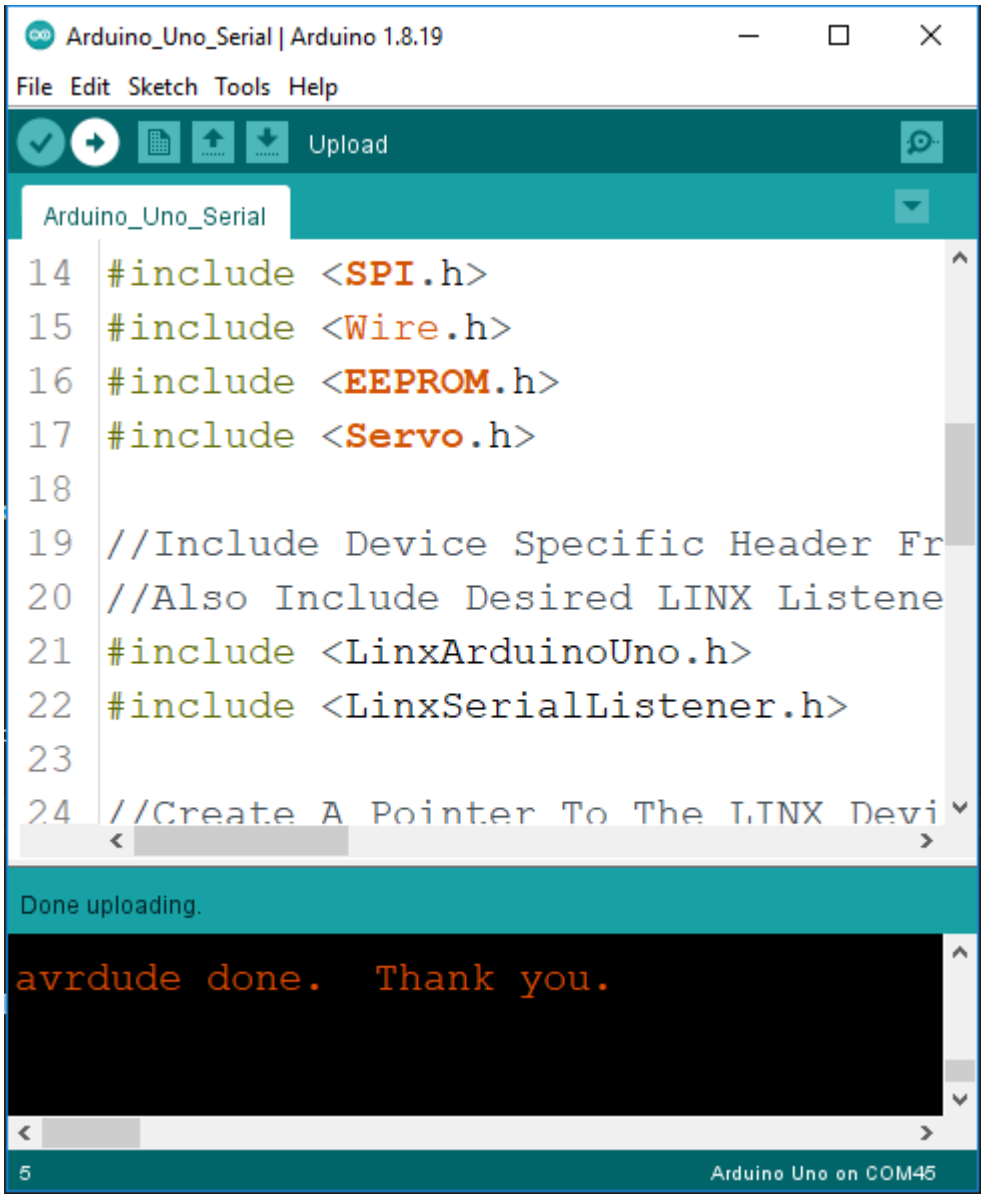
The Arduino IDE uses this path by default to store user-added libraries. If yours isn't this path, go to **File > Preferences** to locate or change the location of your `libraries/` folder.



5.2.2 2. Run the Code

Open the `.ino` file in the `DIY_Control_Robot_Arm_kit_for_Arduino-Rollarm\Arduino Code\Arduino_Uno_Serial` path.

After selecting the correct board and port, upload it to the Arduino board.

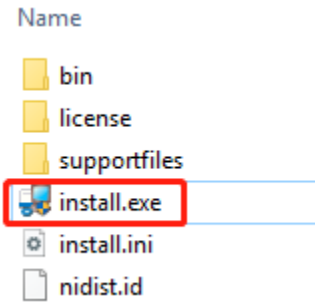


5.2.3 3. Install the APP

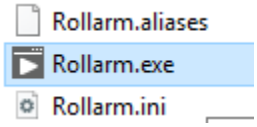
Download the installation package.

- `Rollarm-Labview`

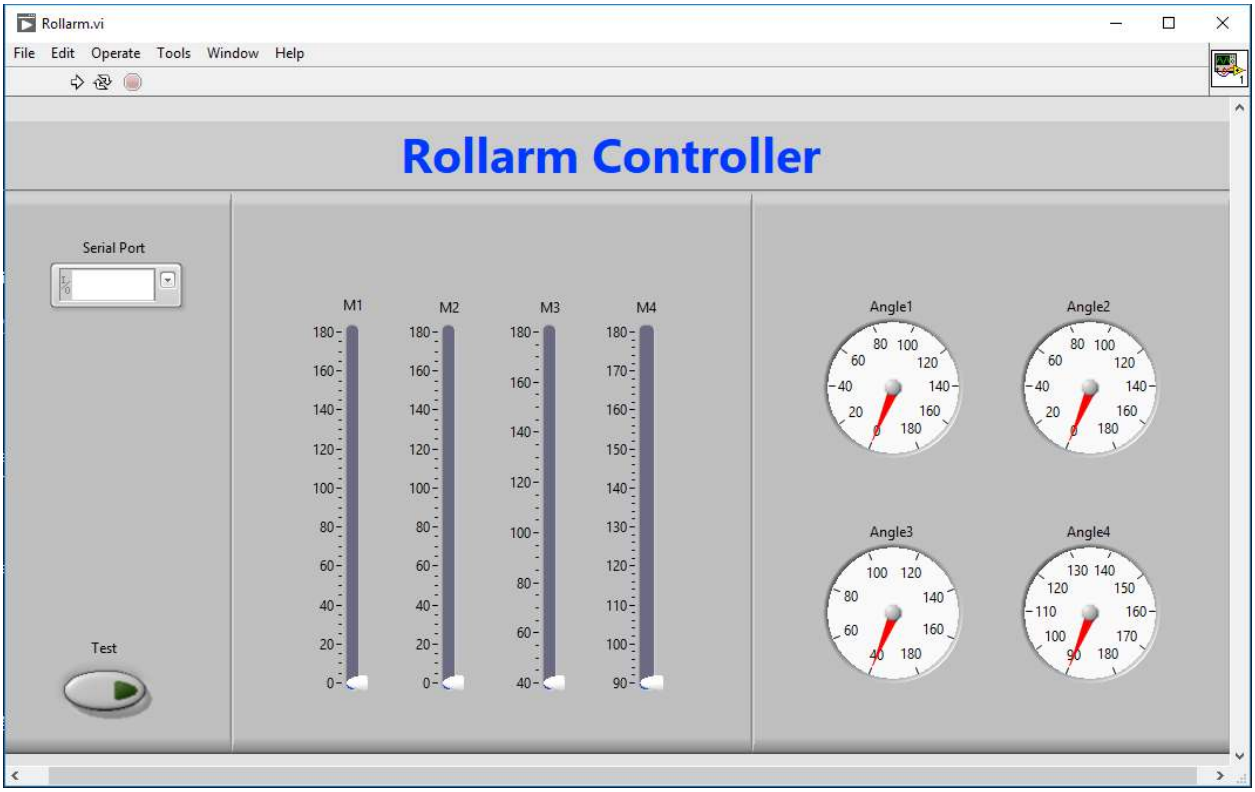
Unzip the file after downloading and double-click the `install.exe` file in the folder `Volume` to install it.



When the installation of this software is complete, locate **Rollarm** in the left corner of your computer to launch it. Or, go to the installation directory we just used and double-click `Rollarm.exe` to open it. The default installation directory is: `C:\\Program Files (x86)\Rollarm`.

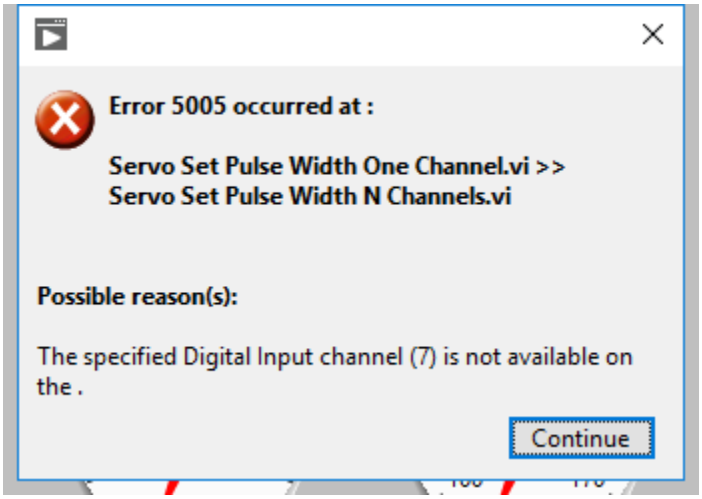


The following interface will show up.



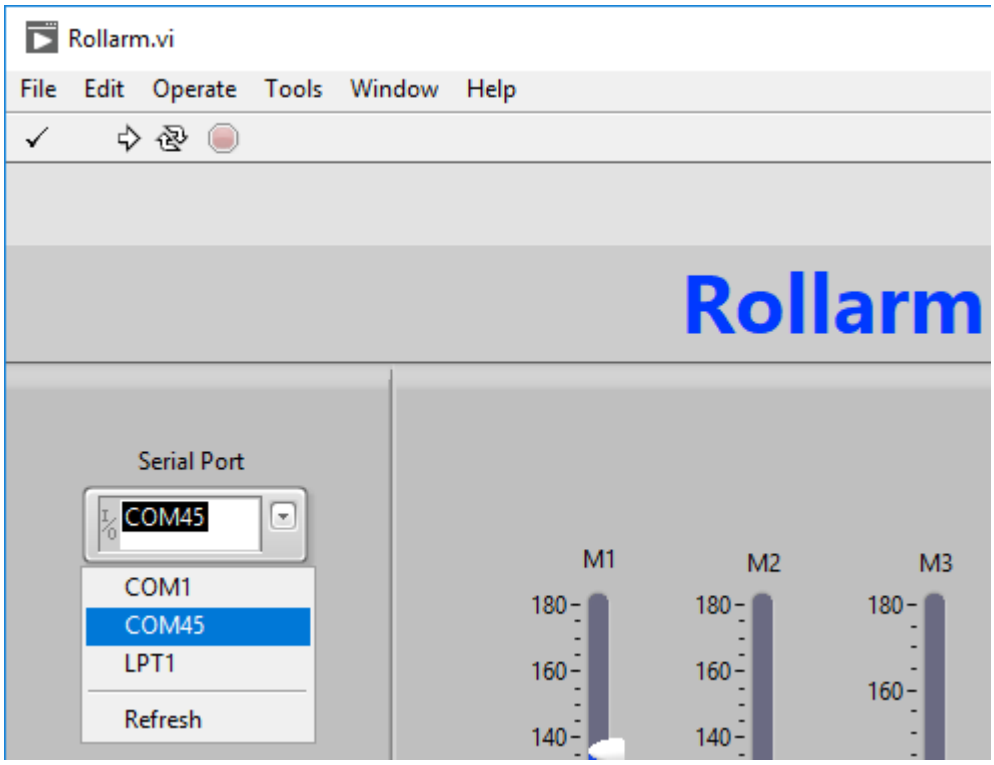
Note: When you just open or stop the app, the following error message may appear, please click **Continue** to ignore

it and continue the next steps.

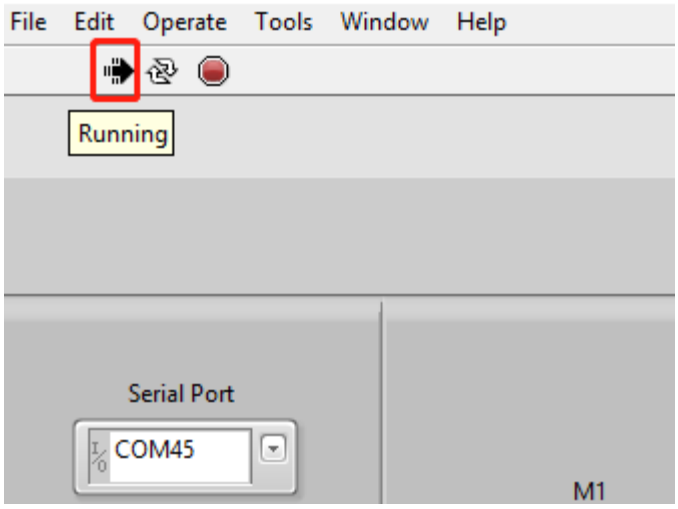


5.2.4 4. Use the APP

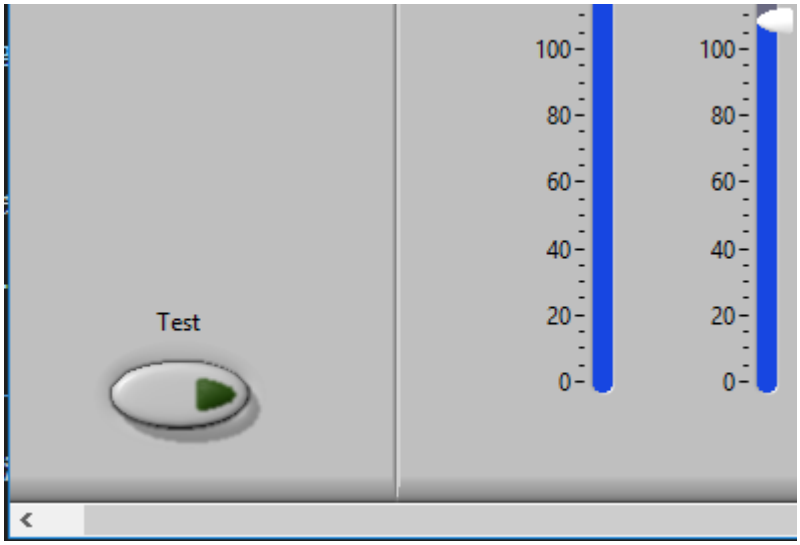
Choose COM port, the same one you picked in the Arduino IDE.



Run this app.

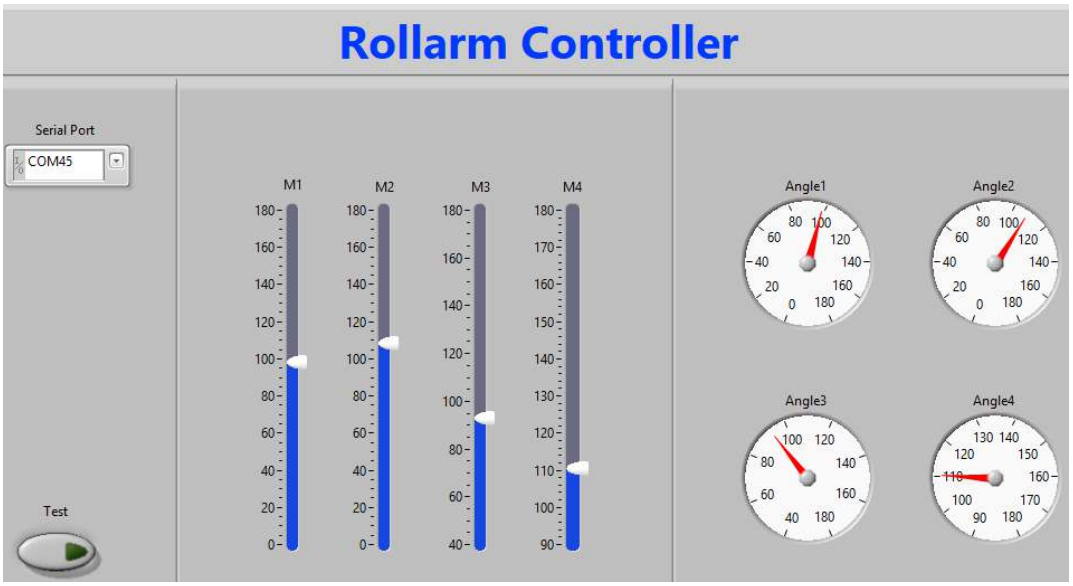


You can test whether the app and Rollarm are communicating properly by clicking the Test button. The LED on pin 13 will light up when you click this button, which means the communication is working.



Now you can use M1~M4 to control Rollarm's 4 servos.

- M1: Control the Rollarm to turn left and right, the angle range is 0-180°.
- M2: Control the forward extension and retraction, the angle range is 0-180°.
- M3: Control the Rollarm's head up/down, the angle range is 40-180°.
- M4: Control the Rollarm open/close, the angle range is 90-180°.


Rollarm Controller
Serial Port
COM45
Test
M1
M2
M3
M4
Angle1
Angle2
Angle3
Angle4

CHAPTER

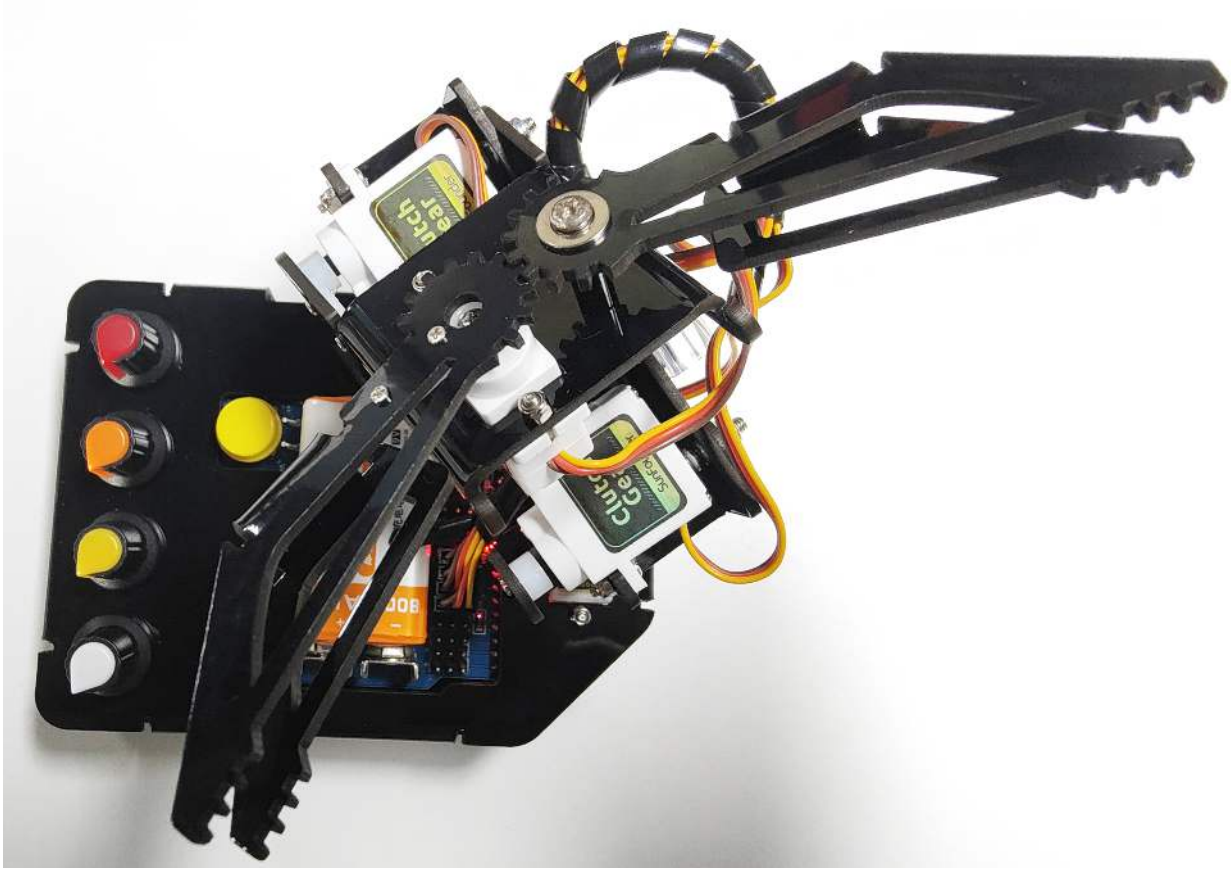
SIX

FAQ

6.1 About the Assembly

Q1: After assembly and program download, the Rollarm's four axes are in wired position, some maybe out of control. What should I do?

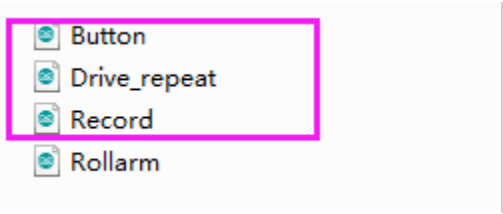
- Remember to power on and calibrate each Servo before assembly, please refer to *Calibrate the Servo Angle*.
- If it has been calibrated, but after running `Rollarm.ino`, it is at the angle shown below, but if you screw all 4 potentiometers to the forward-facing position and can return to the way it was just assembled, it is normal.



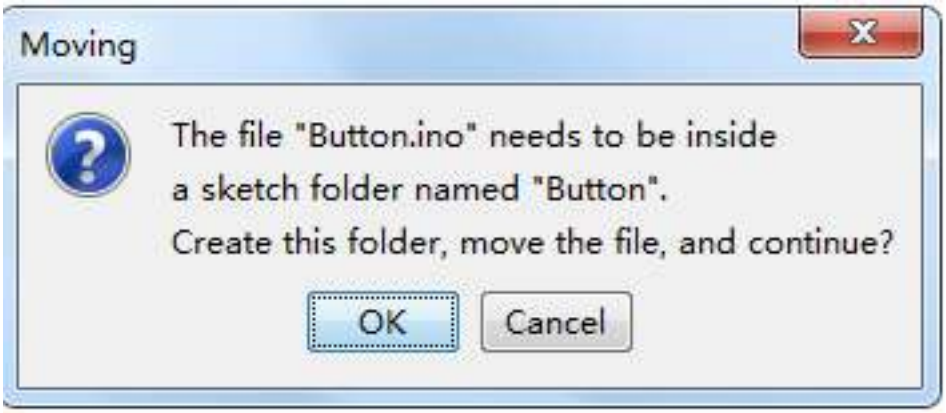
6.2 About the Arduino Code Control

Q1: When I open a program, it prompts me that a new folder should be created. After I click Yes and a new folder is created, the main program reports an error when I want to open the main program. What's going wrong?

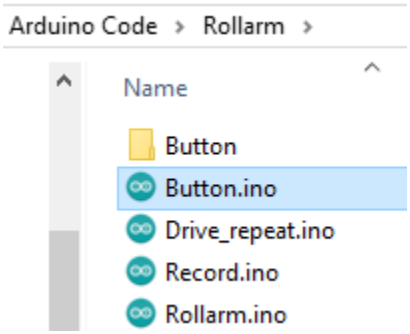
A: DO NOT open these subprograms under `Arduino code\\Rollarm` separately:



If you open the subprograms separately, a window will pop up like this:



If a new folder has been created for the subprogram, please cut the subprogram file to the original directory `Arduino code\\Rollarm`. Reopen the main program:



CHAPTER

SEVEN

THANK YOU

Thanks to the evaluators who evaluated our products, the veterans who provided suggestions for the tutorial, and the users who have been following and supporting us. Your valuable suggestions to us are our motivation to provide better products!

Particular Thanks

- Len Davisson
- Kalen Daniel
- Juan Delacosta

Now, could you spare a little time to fill out this questionnaire?

Note: After submitting the questionnaire, please go back to the top to view the results.

CHAPTER

EIGHT

COPYRIGHT NOTICE