This tutorial is to teach users how to install the Python module of the robotic arm into the system. If you are using the image provided by us, it has already installed the Arm_Lib module and does not need to be installed again.

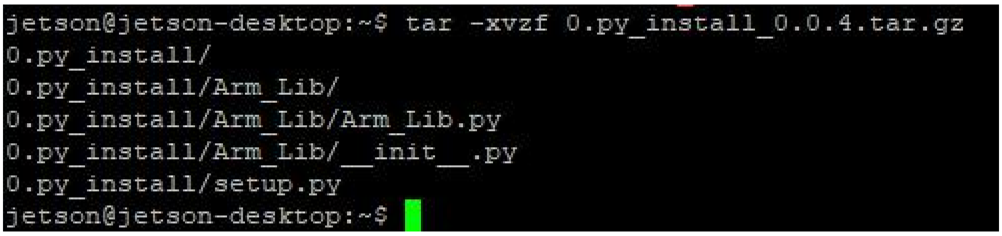
1. Transfer files

Send the 0.py_install_x.x.x.tar.gz file in the current folder to the user directory of Jetson Nano by winSCP, and enter the following command to extract file.

tar -xvzf 0.py_install_x.x.x.tar.gz

For example(0.0.4 version)

tar -xvzf 0.py_install_0.0.4.tar.gz



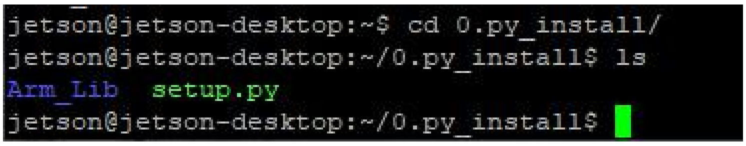
```
jetson@jetson-desktop:~$ tar -xvzf 0.py_install_0.0.4.tar.gz
0.py_install/
0.py_install/Arm_Lib/
0.py_install/Arm_Lib/Arm_Lib.py
0.py_install/Arm_Lib/__init__.py
0.py_install/setup.py
jetson@jetson-desktop:~$
```

2. Install the python library

2.1 Enter the 0.py_install directory.

There is an Arm_Lib folder and a setup.py file under this path.

cd 0.py_install/

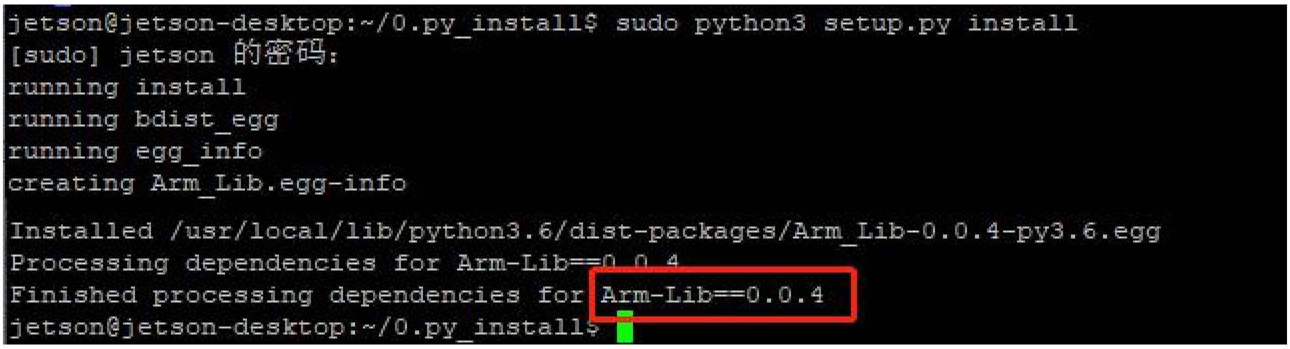


```
jetson@jetson-desktop:~$ cd 0.py_install/
jetson@jetson-desktop:~/0.py_install$ ls
Arm_Lib   setup.py
jetson@jetson-desktop:~/0.py_install$
```

2.2 Input the following command to install the python library,

When we see the prompt as shown in the figure below, it means the installation is complete

sudo python3 setup.py install

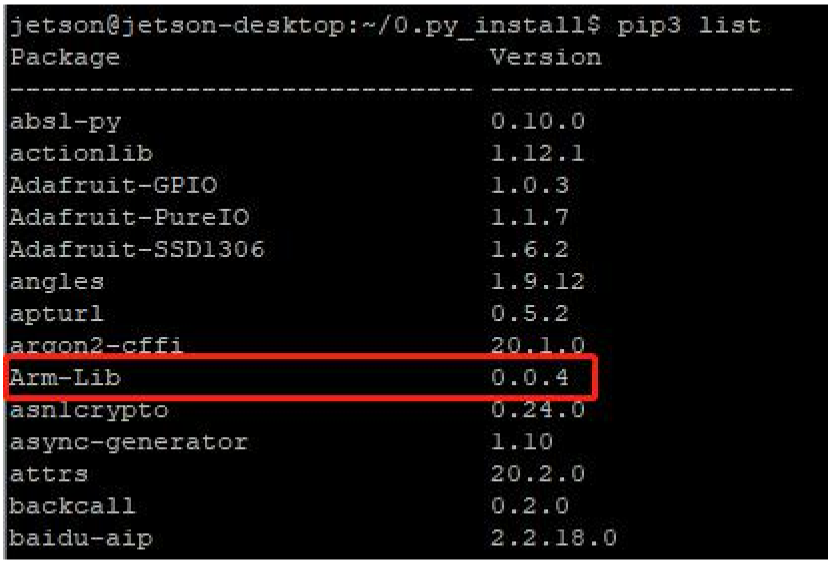


```
jetson@jetson-desktop:~/0.py_install$ sudo python3 setup.py install
[sudo] jetson 的密码:
running install
running bdist_egg
running egg_info
creating Arm_Lib.egg-info

Installed /usr/local/lib/python3.6/dist-packages/Arm_Lib-0.0.4-py3.6.egg
Processing dependencies for Arm-Lib==0.0.4
Finished processing dependencies for Arm-Lib==0.0.4
jetson@jetson-desktop:~/0.py_install$
```

2.3 Input the following command to view the module list of pip3

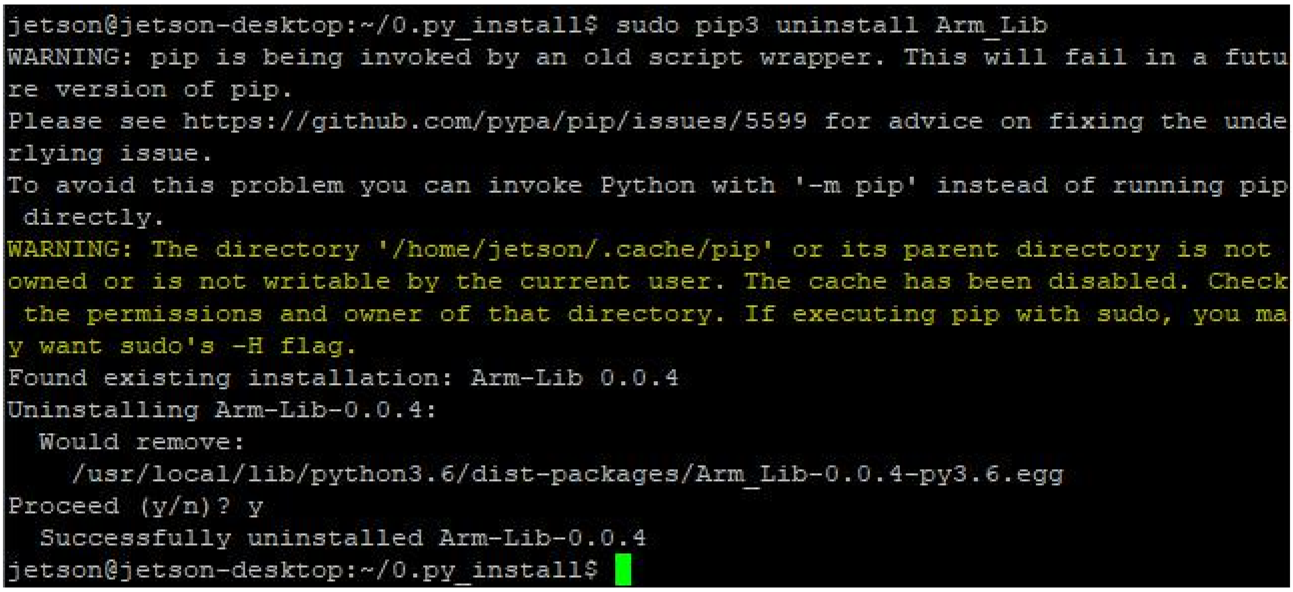
pip3 list



```
jetson@jetson-desktop:~/0.py_install$ pip3 list
Package                              Version
------------------------------ -------------------
absl-py                        0.10.0
actionlib                      1.12.1
Adafruit-GPIO                  1.0.3
Adafruit-PureIO                1.1.7
Adafruit-SSD1306               1.6.2
angles                         1.9.12
apturl                         0.5.2
argon2-cffi                    20.1.0
Arm-Lib                        0.0.4
asn1crypto                     0.24.0
async-generator                1.10
attrs                          20.2.0
backcall                       0.2.0
baidu-aip                      2.2.18.0
```

3. How to uninstall Arm_Lib library

sudo pip3 uninstall Arm_Lib



```
jetson@jetson-desktop:~/0.py_install$ sudo pip3 uninstall Arm_Lib
WARNING: pip is being invoked by an old script wrapper. This will fail in a futu
re version of pip.
Please see https://github.com/pypa/pip/issues/5599 for advice on fixing the unde
rlying issue.
To avoid this problem you can invoke Python with '-m pip' instead of running pip
 directly.
WARNING: The directory '/home/jetson/.cache/pip' or its parent directory is not
owned or is not writable by the current user. The cache has been disabled. Check
 the permissions and owner of that directory. If executing pip with sudo, you ma
y want sudo's -H flag.
Found existing installation: Arm-Lib 0.0.4
Uninstalling Arm-Lib-0.0.4:
  Would remove:
    /usr/local/lib/python3.6/dist-packages/Arm_Lib-0.0.4-py3.6.egg
Proceed (y/n)? y
  Successfully uninstalled Arm-Lib-0.0.4
jetson@jetson-desktop:~/0.py_install$ 
```

4. Arm_Lib library API

Arm_Action_Mode(self, mode)

# Control action group operation. 0: Stop   1: Single run 2: Run in loop

Arm_Action_Study(self)

# Record the current action once in study mode

Arm_Button_Mode(self, mode)

# Set the mode of K1 button, 0: default mode 1: study mode

Arm_Buzzer_Off(self)
# Close buzzer

Arm_Buzzer_On(self, delay=255)
# Open the buzzer, delay defaults to 0xff, which control buzzer keeps beeping.
# delay=1~50. The buzzer turns off automatically after 100 milliseconds, and the maximum delay time is 5 seconds.

Arm_Clear_Action(self)
# Clear action

Arm_PWM_servo_write(self, id, angle)
# Control PWM servo.
id:1-6 (0 means control all servos). angle: 0-180

Arm_RGB_set(self, red, green, blue)
# Control RGB light color

Arm_Read_Action_Num(self)
# Read the number of saved action groups 0

Arm_get_hardversion(self)
# Read the hardware version number

Arm_ping_servo(self, id)
# Read the servo status.
In normally, it will return 0xda; If it didn't read date, it will return 0x00; other values means servo errors.

Arm_reset(self)
# Restart drive board

Arm_serial_servo_read(self, id)
#Read the specified servo angle
  id: 1-6 returns 0-180. When read error, it will return None

Arm_serial_servo_read_any(self, id)
# Read bus servo angle
id: 1-250
Return value: 0-180
Arm_serial_servo_write(self, id, angle, time)
# Control bus servo
id: 1-6 (0 is to send 6 servos)

angle: 0-180°
time: servo rotation time

Arm_serial_servo_write6(self, s1, s2, s3, s4, s5, s6, time)
# Control 6 bus servos at the same time
S1~S5 angle:0-180°
S6 angle:0-270°
time: servo rotation time

Arm_serial_servo_write6_array(self, joints, time)
# joints: an array, which stores the angle data of 6 servos
time: servo rotation time

Arm_serial_servo_write_any(self, id, angle, time)
# Control any bus servo
id: 1-250 (0 is broadcast)
angle: 0-180°
time: servo rotation time

Arm_serial_servo_write_offset_state(self)
# Read the status of the position offset of the set bus servo,
0 means no corresponding servo ID, 1 means success, 2 means set failure or it is out of range.

Arm_serial_servo_write_offset_switch(self, id)
# Set the bus servo mid-position offset.
id: 1-6 servo
0: Restore the factory default median

Arm_serial_set_id(self, id)
# Set the number of the bus servo on expansion board

Arm_serial_set_torque(self, onoff)
# Control torque of all servo
1: Turn on the torque
0: Turn off the torque (we can manually change the angle of the servo)