3.1 Raspberry Pi Control Routine

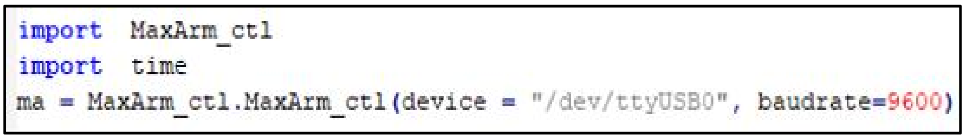
The document demonstrates the control of MaxArm's bus servo angles, setting of XYZ coordinates, control of nozzle functionality, and reading of bus servo angles and XYZ coordinates using a Raspberry Pi.

1. Working Principle

The source code of the program is located in "**10.Serial Communication/ Control Routine/ Raspberry Pi Control Routine/ Routin**e". The specific underlying implementation can be view in the "**MaxArm_ctl.py**" program file stored in "**10.Serial Communication/ Control Routine/ Raspberry Pi Routine Control/ Raspberry Pi Routin**e".

1) After hardware connection, enter the command "ls /dev/ttyUSB*" via Raspberry Pi's terminal to view the corresponding device port connecting to MaxArm. The following program uses "**/dev/ttyUSB0**" as an example.

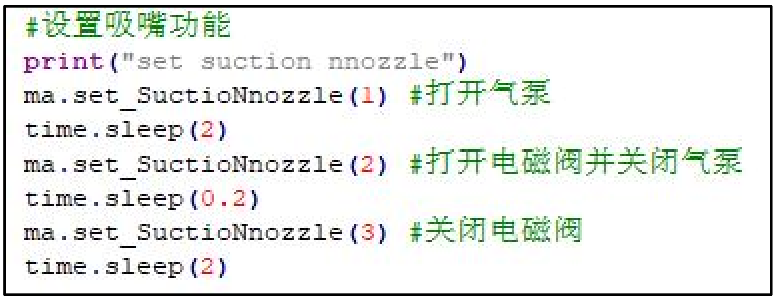
2) Import MaxArm_ctl library, then invoke the MaxArm_ctl.MaxArm_ctl() function. Input the device port checked in step 1, "/dev/ttyUSB0" and initialize the serial baud rate to 9600, otherwise communication cannot be established normally.



```
import  MaxArm_ctl
import  time
ma = MaxArm_ctl.MaxArm_ctl(device = "/dev/ttyUSB0", baudrate=9600)
```

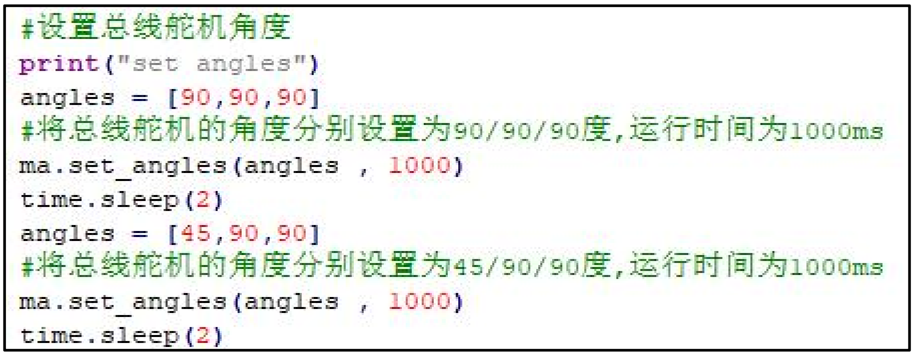
3) By calling the ma.set_SuctioNnozzle() function, the control of air pump and solenoid valve can be achieved. For example, take the code **ma.set_SuctioNnozzle(1)** as example, where the parameter "**1**" represents the current mode of operation as opening the air pump. The

duration of the venting operation is very short, so the delay for the venting operation can be reduced.



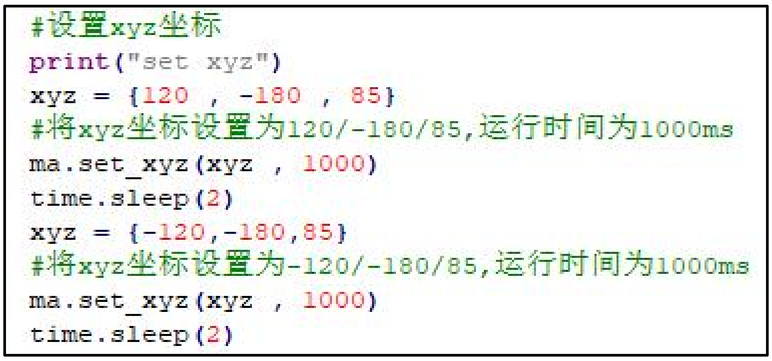
```
#设置吸嘴功能
print("set suction nnozzle")
ma.set_SuctioNnozzle(1) #打开气泵
time.sleep(2)
ma.set_SuctioNnozzle(2) #打开电磁阀并关闭气泵
time.sleep(0.2)
ma.set_SuctioNnozzle(3) #关闭电磁阀
time.sleep(2)
```

4) By calling the **ma.set_angles()** function, the control of the bus servo angles can be achieved. For example, using the code **ma_ctl.set_angles(angles , 1000)**, where the first parameter "**angles**" represents the rotation angles. This parameter is a list of length 3, corresponding to the angles required for the rotation of the 3 bus servos. The range of rotation angles is 0°-240°, and the second parameter "1000" represents the running time (in milliseconds), with a duration of 1000 milliseconds in this case.



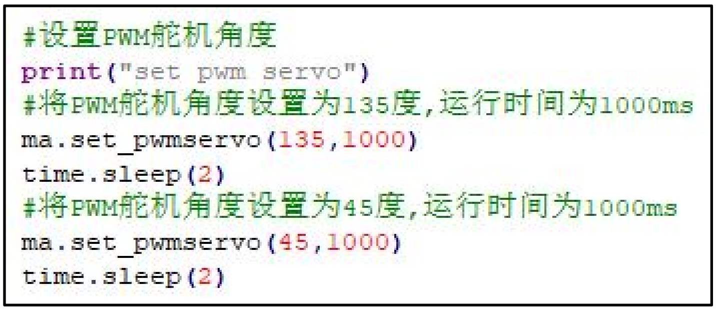
```
#设置总线舵机角度
print("set angles")
angles = [90,90,90]
#将总线舵机的角度分别设置为90/90/90度,运行时间为1000ms
ma.set_angles(angles , 1000)
time.sleep(2)
angles = [45,90,90]
#将总线舵机的角度分别设置为45/90/90度,运行时间为1000ms
ma.set_angles(angles , 1000)
time.sleep(2)
```

5) By calling the **ma.set_xyz()** function, you can set the XYZ coordinates. For example, using the code **ma.set_xyz(xyz , 1000)**, where the first parameter "**xyz**" represents the XYZ coordinates to be set, and it is a list of length 3, and the second parameter "1000" represents the duration of operation (in milliseconds), with a duration of 1000 milliseconds in this case.



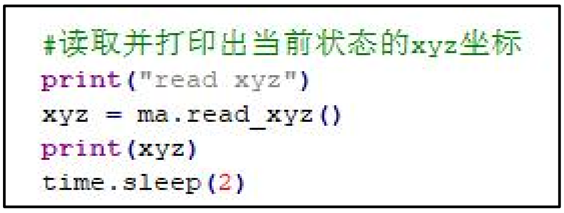
```
#设置xyz坐标
print("set xyz")
xyz = {120 , -180 , 85}
#将xyz坐标设置为120/-180/85,运行时间为1000ms
ma.set_xyz(xyz , 1000)
time.sleep(2)
xyz = {-120,-180,85}
#将xyz坐标设置为-120/-180/85,运行时间为1000ms
ma.set_xyz(xyz , 1000)
time.sleep(2)
```

6) By calling the **ma.set_pwmservo()** function, you can control the angle of PWM servos. For example, using the code **ma.set_pwmservo(135, 1000)**, where the first parameter "**135**" represents the angle of rotation for the PWM servo, and the second parameter "**1000**" represents the duration of operation (in milliseconds), with a duration of 1000 milliseconds in this case.



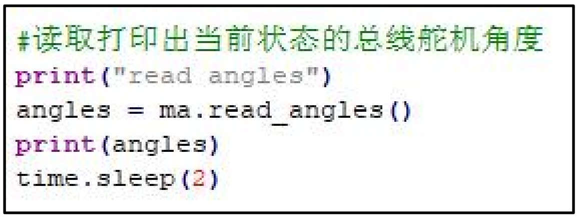
```
#设置PWM舵机角度
print("set pwm servo")
#将PWM舵机角度设置为135度,运行时间为1000ms
ma.set_pwmservo(135,1000)
time.sleep(2)
#将PWM舵机角度设置为45度,运行时间为1000ms
ma.set_pwmservo(45,1000)
time.sleep(2)
```

7) By calling the **ma.read_xyz()** function, you can obtain the current XYZ coordinates of MaxArm. Retrieve the current XYZ coordinate information and print it out.



```
#读取并打印出当前状态的xyz坐标
print("read xyz")
xyz = ma.read_xyz()
print(xyz)
time.sleep(2)
```

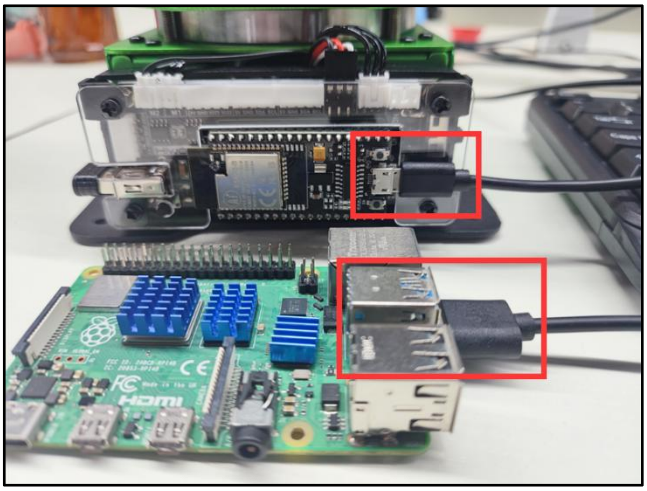
8) By calling the **ma.read_angles()** function, you can obtain the current angles of the three bus servos on MaxArm. Retrieve the current information of the bus servo angles and print it out.



```
#读取打印出当前状态的总线舵机角度
print("read angles")
angles = ma.read_angles()
print(angles)
time.sleep(2)
```

2. Preparation

Please ensure that MaxArm is connected to the USB port of Raspberry Pi.



3. Operation Instructions

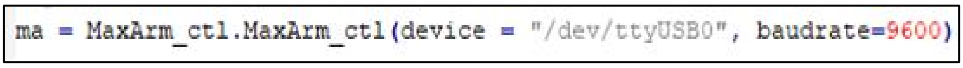
The entered command should be case sensitive and “Tab” can be used to complement key words.

1) Start MaxArm robotic arm and connect to Raspberry Pi system desktop via VNC. The, transfer the routine code to the “/home/MaxArm/” directory.

2) Click on in the upper left corner of the desktop, or press “Ctrl+Alt+T” to open the LX terminal.

3) Input the command “**ls /dev/ttyUSB***” to view the corresponding device port.

4) Replace the device port number “**/dev/ttyUSB0**” in the “main.py file with the port number obtained in step 3).



```
ma = MaxArm_ctl.MaxArm_ctl(device = "/dev/ttyUSB0", baudrate=9600)
```

5) Enter the command “**cd /home/MaxArm**” and press Enter to access the directory where the program is stored.

6) Enter the command “**cd /home/MaxArm**” and press Enter to start the program.

7) If you want to close this file, enter “**Ctrl+C**” in the LX terminal. If the operation fails, you can try pressing "**Ctrl+C**" multiple times

4. Program Outcome

While the program is running:

1) Open the air pump to start suction of an object, then release the object after 2 seconds.

2) Rotate the three bus servos to 90 degrees, then wait for 2 seconds before rotating them to 45, 90 , 90 degrees respectively.

3) Change the XYZ coordinates of MaxArm to 120/-180/85, then wait for two seconds before changing the XYZ coordinates to -120, -180, 85.

4) Rotate the PWM servo angle to 135 degrees, then wait for 2 seconds before rotating it to 45 degrees.

5) Finally, print the current angles of the bus servos and XYZ coordinates in the serial port.