Lesson 5 The Movement of Robotic Arm and Chassis car

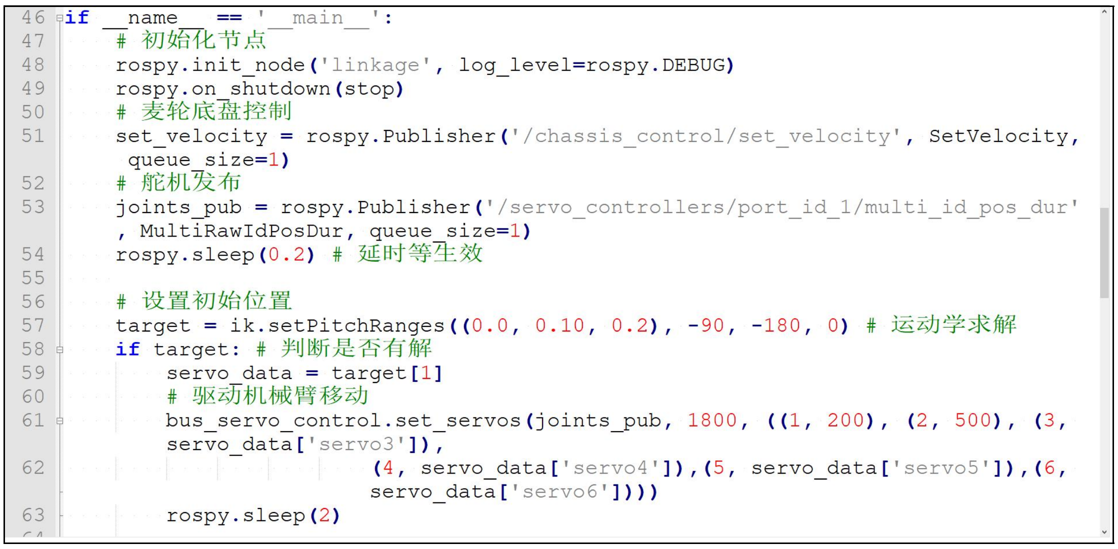
1. Working Principle

Based on the previous four lessons, chassis is added. In this lesson, the robotic arm changes posture and chassis car moves at the same time so ad to keep the end of robotic arm motionless.

According to the inverse kinematics, By adjusting the value of y-axis coordinate and converting them to the value of ID3, ID4, ID5 and ID6 servos, which can make the end of robotic arm motionless.

The chassis is set to motion mode and adjust corresponding movemnt position according to the posture of robotic arm.

The source code of program is located in : **/home/ubuntu/armpi_pro/src/armpi_pro_demo/kinematics_demo/link.py**



```
if __name__ == '__main__':
    # 初始化节点
    rospy.init_node('linkage', log_level=rospy.DEBUG)
    rospy.on_shutdown(stop)
    # 麦轮底盘控制
    set_velocity = rospy.Publisher('/chassis_control/set_velocity', SetVelocity,
     queue_size=1)
    # 舵机发布
    joints_pub = rospy.Publisher('/servo_controllers/port_id_1/multi_id_pos_dur'
    , MultiRawIdPosDur, queue_size=1)
    rospy.sleep(0.2) # 延时等生效
    
    # 设置初始位置
    target = ik.setPitchRanges((0.0, 0.10, 0.2), -90, -180, 0) # 运动学求解
    if target: # 判断是否有解
        servo_data = target[1]
        # 驱动机械臂移动
        bus_servo_control.set_servos(joints_pub, 1800, ((1, 200), (2, 500), (3,
        servo_data['servo3']),
                (4, servo_data['servo4']),(5, servo_data['servo5']),(6,
                servo_data['servo6'])))
        rospy.sleep(2)
```

2. Operation Steps

It should be case sensitive when entering command and the “Tab” key can be used to complete the keywords.

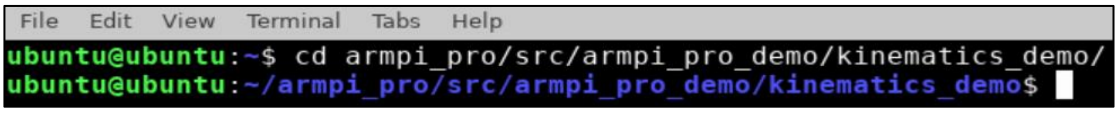
1) Turn on ArmPi Pro and connect to the system desktop via No Machine.

2) Click

and select

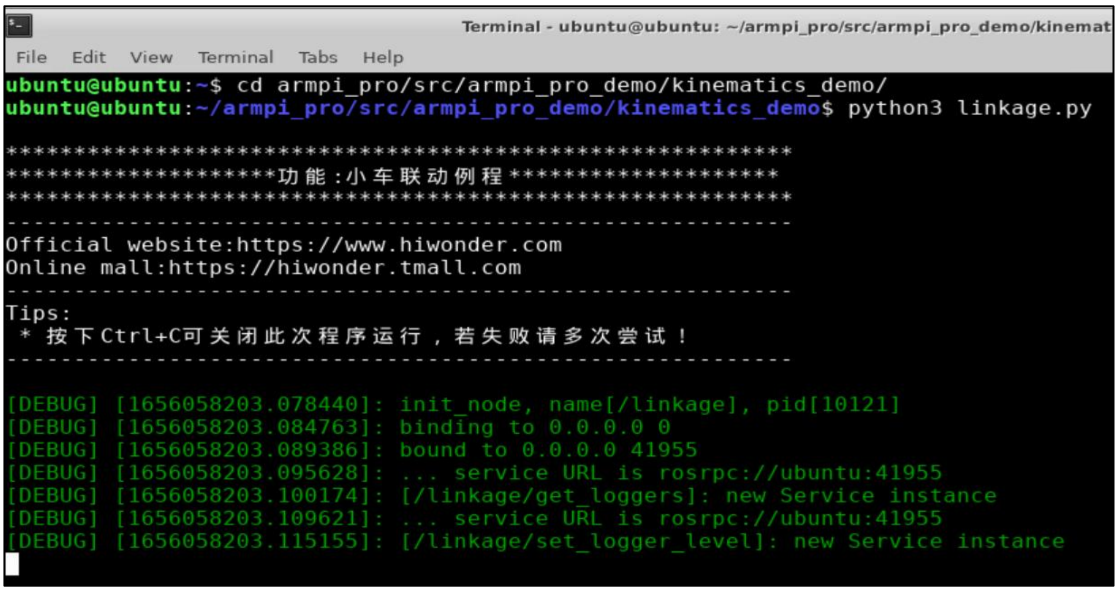
in pop-up interface to open the terminal.

3) Enter command “cd armpi_pro/src/armpi_pro_demo/kinematics_demo/” and press “Enter” to access to the directory of game programmings.



```
ubuntu@ubuntu:~$ cd armpi_pro/src/armpi_pro_demo/kinematics_demo/
ubuntu@ubuntu:~/armpi_pro/src/armpi_pro_demo/kinematics_demo$
```

4) Enter command “python3 linkage.py” and press “Enter” to start game. The following prompt will show up in terminal.


Terminal - ubuntu@ubuntu: ~/armpi_pro/src/armpi_pro_demo/kinemat
File Edit View Terminal Tabs Help

```
ubuntu@ubuntu:~$ cd armpi_pro/src/armpi_pro_demo/kinematics_demo/
ubuntu@ubuntu:~/armpi_pro/src/armpi_pro_demo/kinematics_demo$ python3 linkage.py

**********************************************************
********************功能:小车联动例程********************
**********************************************************
----------------------------------------------------------
Official website:https://www.hiwonder.com
Online mall:https://hiwonder.tmall.com
----------------------------------------------------------
Tips:
 * 按下Ctrl+C可关闭此次程序运行，若失败请多次尝试！
----------------------------------------------------------

[DEBUG] [1656058203.078440]: init_node, name[/linkage], pid[10121]
[DEBUG] [1656058203.084763]: binding to 0.0.0.0 0
[DEBUG] [1656058203.089386]: bound to 0.0.0.0 41955
[DEBUG] [1656058203.095628]: ... service URL is rosrpc://ubuntu:41955
[DEBUG] [1656058203.100174]: [/linkage/get_loggers]: new Service instance
[DEBUG] [1656058203.109621]: ... service URL is rosrpc://ubuntu:41955
[DEBUG] [1656058203.115155]: [/linkage/set_logger_level]: new Service instance
```

5) If want to exit the game, please press “Ctrl+C” in the terminal. If fail to exit, please try several times!

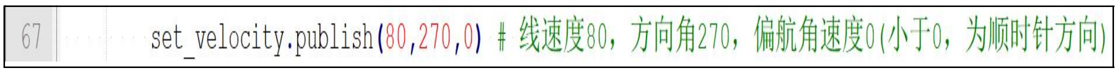
3. Project Outcome

After starting game, ArmPi Pro will constantly change the posture of robotic arm in a constant contractions and expansions, which make the end of robotic arm motionless.

4. Kinematics Analysis

4.1 Chassis Kinematics Analysis

The function "set_velocity.publish()" is used to control robot chassis to move constantly.



```
        set_velocity.publish(80,270,0) # 线速度80，方向角270，偏航角速度0(小于0，为顺时针方向)
```

Take the code "set_velocity.publish(80,270,0)" as example:

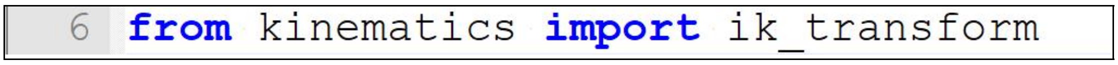
The first parameter "80" is the linear velocity.

The second parameter "270" is the direction angle.

The third parameter "0" is the yaw rate.(When it is samller than 0, its direction is clockwise)

4.2 Robotic Arm Inverse Kinematics Analsis

Before get the inverse kinematics solution, the relevant libraries need to be imported.



```
from kinematics import ik_transform
```

Before robotic arm starts moving, calculate the position robotic amr through inverse kinemaics

```
        target = ik.setPitchRanges((0.0, 0.20, 0.20), -90, -180, 0) #
        运动学求解
```

Take the code “target = ik.setPitchRanges((0.0, 0.20, 0.20), -90, -180, 0)” as example,

In the first parameter “**(0, 0.20, 0.20**”, “0” is the position in x-axis. “0.2” is the position in y-axis. “0.20” is the position in y-axis.

The second parameter “-90” is the picth angle.

The third paramter “-180” is the range of picth angle.

The fourth parameter “0” is the range of pitch angle.

Before controlling robotic arm to mve, the relevant libraries need to be imported:

```
from armpi_pro import bus_servo_control
from hiwonder_servo_msgs.msg import MultiRawIdPosDur
```

In this program, the robotic arm is controlled to move to the traget position by calling bus_servo_control.set_servos() function.

```
        # 驱动机械臂移动
        bus_servo_control.set_servos(joints_pub, 1800, ((1, 200), (2, 500
        ), (3, servo_data['servo3']),
                                (4, servo_data['servo4']),(5, servo_data['servo5'
                                ]),(6, servo_data['servo6'])))
```

Robotic arm controlling takes the code “bus_servo_control.set_servos(joints_pub, 1800, ((1, 200), (2, 500), (3, servo_data['servo3']),(4, servo_data['servo4']),(5, servo_data['servo5']),(6, servo_data['servo6'])))” as example.

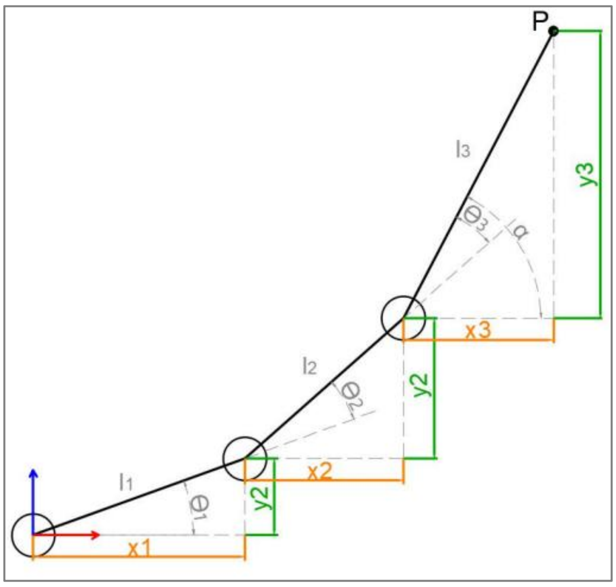
The first parameter “joints_pub” is to publish servo controlling node message.

The second parameter “1800” is the running time.

In the third parameter “((1,200),(2,500),(3, servo_data['servo3']), (4, servo_data['servo4']), (5, servo_data['servo5']), (6, x_dis)”, “1” is the servo number, “200” is the servo angle. Same for the remaining code.

4.3 Inverse Kinematics Calculation

The process of inverse kinematics calculate involves amount of matrix manipulations but they will not be explained in this lesson, For a better understanding of inverse kinematics principle, the robotic arm is analyzed here using the geometric method.



The model of the robotic arm is simplified by removing the base pant-tilt and actuator parts to get the body of the robotic arm. From the above figure, the coordinates (x,y) of the end point P of the robotic arm can also be regarded as (x1+x2+x3, y1+y2+y3).

The θ_1, θ_2 and θ_3 in the figure are the servo angles to be solved, and α is the angle between the claw and the horizontal plane. From the above figure, it can be seen that the pitch angle of claw = $\alpha=\theta_1+\theta_2+\theta$, according to which the following equation can be listed.

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3)$$

Among them, x and y are given by the user, and l_1, l_2, and l_3 are the inherent properties of the mechanical structure of the robotic arm.

To facilitate the calculation, the known parts are treated for overall consideration:

$$m = l_3 \cos(\alpha) - x$$

$$n = l_3 \sin(\alpha) - y$$

Substituting m and n into the existing equations and simplifying them:

$$l_2 = (l_1\, cos\, \theta_1 + m)^2 + (l_1\, sin\, \theta_1 + n)^2$$

$$sin\, \theta_1 = \frac{(-b \pm \sqrt{b^2 - 4ac})}{2a}$$

$$a = m^2 + n^2$$

$$b = \frac{-n \cdot ({l_1}^2 - {l_2}^2 - m^2 - n^2)}{l_2}$$

$$c = (\frac{{l_1}^2 - {l_2}^2 - m^2 - n^2}{2l_2})^2$$

Accordingly, the angle of θ_1 can be found, and θ_2 and θ_3 can be obtained, so that the angles of the three servos can be solved.