1.Voice control robotic arm movement

Command word table

Command words	Command words
Clamp the clip	Lift the arm up
Open the clip	Put the arm down
Action A	Arm left
Action B	Arm right
Action C	Action D
Action E	Reset
Go ahead	Back
Turn left	Turn right
Enter A mode	Enter B mode
Robot stop	Robot sleep
Red light up	Green light up
Blue light up	Yellow light up
light A	light B
light C	display power
Warning	

1.1、Function Description

Voice control the motion status of the car, RGB light bar, robotic arm action group, etc.

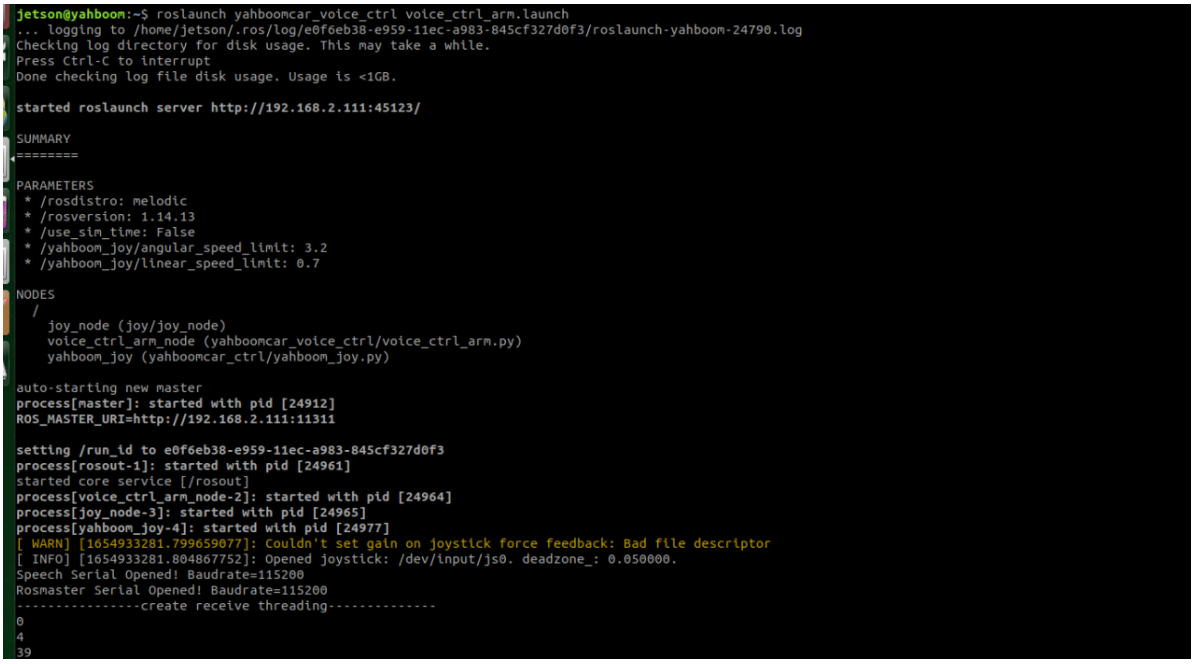
1.2 Start

1.2.1 Ros package path

```
~/yahboomcar_ws/src/yahboomcar_voice_ctrl/
```

1.2.2 Start

```
roslaunch yahboomcar_voice_ctrl voice_ctrl_arm.launch
```



1.2.3 Code `voice_ctrl_arm.py`

- Code path

```
~/yahboomcar/src/yahboomcar_voice_ctrl/scripts
```

- Code analysis:

  1） import the relevant library files

  ```
  from Speech_Lib import Speech
  from voice_arm_library import *
  from Rosmaster_Lib import Rosmaster
  ```

  Speech_Lib: Voice module library, path:

  ```
  ~/software/py_install_V0.0.1/py_install/Speech_Lib
  ```

  voice_arm_library：Robot arm action group library， path：

  ```
  ~/yahboomcar/src/yahboomcar_voice_ctrl/scripts
  ```

  Rosmaster_Lib: Rosmaster driver library, path：

  ```
  ~/software/py_install/Rosmaster_Lib
  ```

  2）Create voice recognition objects, drive control objects and robotic arm action objects

  ```
  spe = Speech()
  self.car = Rosmaster()
  voice_arm = Voice_Arm()
  ```

3) Main function: According to the result of the voice recognition, the relative code is executed.
In this course, we take returning the robotic arm to the initial position as an example.

```
speech_r = spe.speech_read()
            if speech_r!=999:
                print(speech_r)
            #print(speech_r)
            if speech_r == 49 :
             spe.void_write(45)
             voice_arm.init_pose()
```

`voice_arm.init_pose()` it is the program that needs to be executed. At this time, it will jump to the `voice_arm_library` library and execute the function init_pose() inside. In the init_pose() function, the following function will be executed.

```
def init_pose(self):
    self.arm_joint.joints =[90.0, 145.0, 0.0, 0.0, 90.0, 31.0]
    self.pubPoint.publish(self.arm_joint)
```

Defines the angle that each joint needs to execute, takes `TargetAngle` as the topic, and publishes the data; then returns to the main function, subscribes to the TargetAngle topic; receives the data, enters the callback function, and sends it to the underlying control through the `self.car.set_uart_servo_angle` function , which drives the servo.

1.3 Flowchart