2. Voice control robotic arm memory learning

2.1. Function description

The robotic arm is controlled through the voice module to record the position and posture of the robotic arm that we set manually, and the movements learned and recorded can be controlled through voice control.

2.2. Start

```
#Raspberry Pi 5 master needs to enter docker first, please perform this step
#If running the script into docker fails, please refer to ROS/07, Docker tutorial
~/run_docker.sh
```

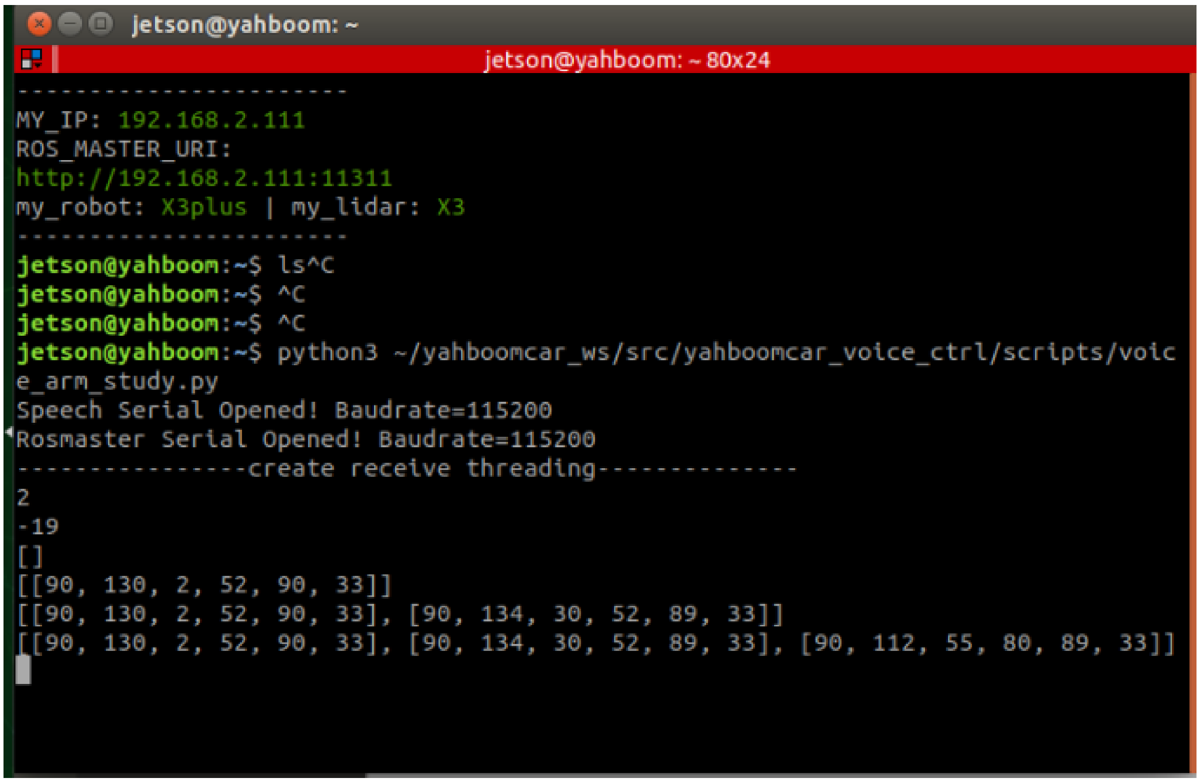
2.2.1. Function package path

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

2.2.2. Start

```
python3 ~/yahboomcar_ws/src/yahboomcar_voice_ctrl/scripts/voice_arm_study.py
```

- When the program starts, when you hear "I can start the action input", the robotic arm will turn off the torque. At this time, we can manually move each servo. After turning it to the angle we need, pass "Hello, Xiaoya "wakes up the module, and after hearing "in", we can continue to say "input completed". At this time, the robotic arm will record the current action, and the voice module will also reply "Okay, please enter the next set of actions." ", the terminal will print out the currently learned pose (here are some tips, **Each servo has a limit. When the limit is exceeded, the actions of this group will not be recorded, and the buzzer will beep once. Print on the terminal which servo exceeds the limit value. The value exceeding the limit value will be displayed as a negative number**) Up to 20 groups of actions can be recorded**. Say to the module "stop recording when the input is completed";
- After we finish recording the action group, say "Input Ended" to the module and reply "Okay. Ended". At this time, we can say "Movement Action" through voice to make the robot arm move the action just recorded;
- If you need to re-learn the recording action group, then say "clear action" to it, the module will reply "action has been cleared", and then return to the initial position. After a beep, it will announce "I can input actions", and you can start at this time Let the robot arm record the action group again.



```
-----------------------
MY_IP: 192.168.2.111
ROS_MASTER_URI:
http://192.168.2.111:11311
my_robot: X3plus | my_lidar: X3
-----------------------
jetson@yahboom:~$ ls^C
jetson@yahboom:~$ ^C
jetson@yahboom:~$ ^C
jetson@yahboom:~$ python3 ~/yahboomcar_ws/src/yahboomcar_voice_ctrl/scripts/voic
e_arm_study.py
Speech Serial Opened! Baudrate=115200
Rosmaster Serial Opened! Baudrate=115200
----------------create receive threading--------------
2
-19
[]
[[90, 130, 2, 52, 90, 33]]
[[90, 130, 2, 52, 90, 33], [90, 134, 30, 52, 89, 33]]
[[90, 130, 2, 52, 90, 33], [90, 134, 30, 52, 89, 33], [90, 112, 55, 80, 89, 33]]
```

2.2.3. Core file voice_arm_study.py

- code path

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

- Core code analysis

  1. Turn off/on the servo torque. Only when the servo torque is turned off can each servo be manually turned.

```
driver.car.set_uart_servo_torque(0) #Turn off torque
driver.car.set_uart_servo_torque(1) #Turn on torque
```

2), two important lists

```
current_joints[] #The current list of each servo data, that is, the data that
needs to be recorded
study_joints[] #Study action group data
```

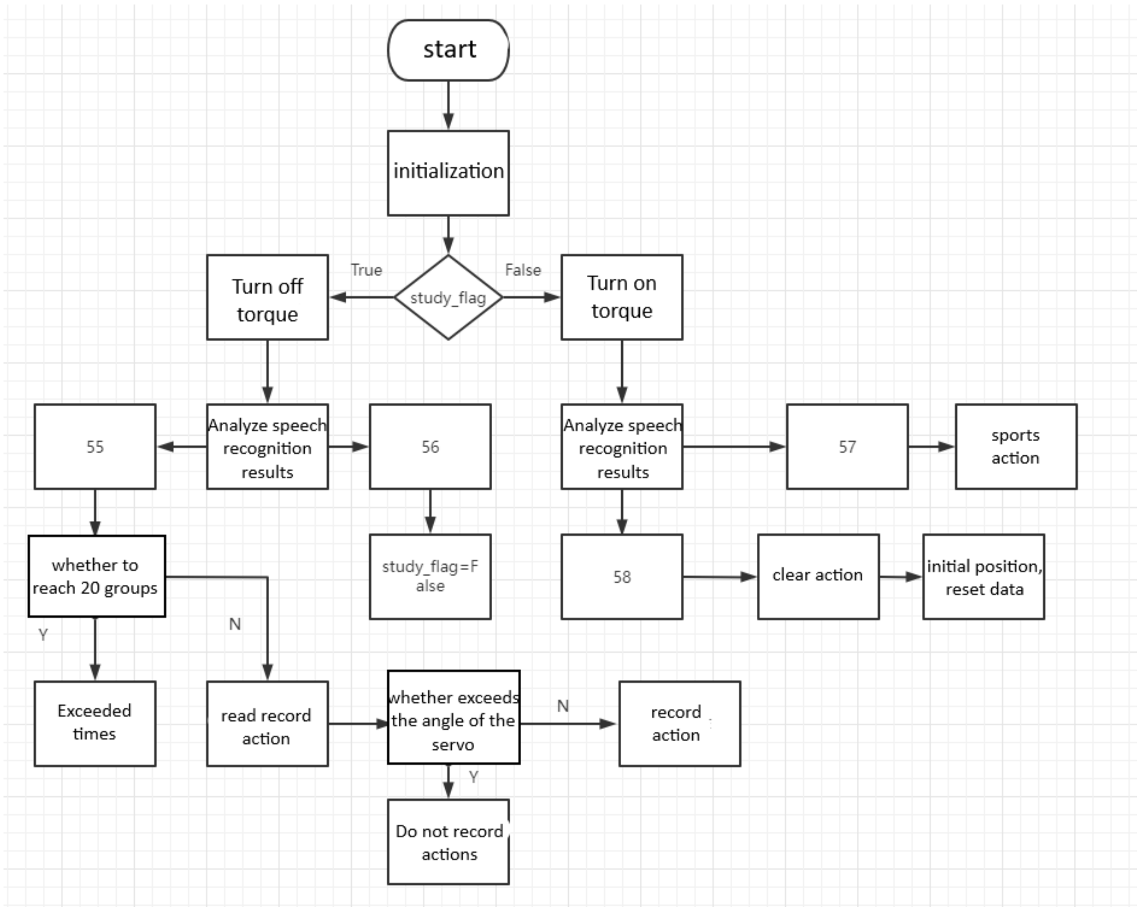
3), read each angle value function of the current steering gear

```
get_uart_servo_angle_array()
```

4), study status flag study_flag

```
study_flag = True #Study, record status
study_flag = False #Motion status
```

2.3. Program flow chart


start
initialization
study_flag
True
False
Turn off torque
Turn on torque
Analyze speech recognition results
Analyze speech recognition results
55
56
57
58
sports action
whether to reach 20 groups
study_flag=False
clear action
initial position, reset data
Y
N
Exceeded times
read record action
whether exceeds the angle of the servo
N
record action
Y
Do not record actions