12. Mediapipe arm attitude control robotic arm

12.1. Use

After the program is started, the camera captures the image and lifts the right arm. After the buzzer is not heard, the robotic arm will synchronously imitate the movement of the arm (straighten, bend and close the palm).

Note: [R2] on the remote control handle has the [pause/start] function for this gameplay.

Function package path: ~/yahboomcar_ws/src/arm_mediapipe/

```
#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
```

Robot side

```
roslaunch arm_mediapipe mediaArm.launch # robot
```

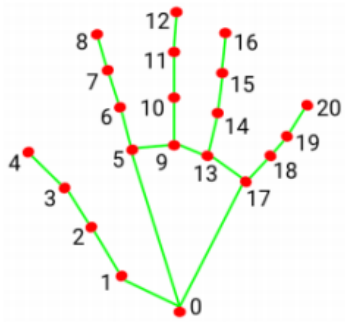
Virtual machine side (recommended)

```
rosrun arm_mediapipe PoseArm.py
```

After the program is started, press the R2 button on the handle to turn on the function. Stand in front of the camera so that the entire arm appears in the screen. At this time, the buzzer will sound. This is written in the program to prevent the robotic arm from returning after the arm is put down. Follow the movement, causing confusion in the program. We raise our elbows and wait for the buzzer to stop sounding, and the robotic arm will simulate the movement of the arm. You can straighten and bend the elbow joint, wrist joint, shoulder joint, and open/close the palm to control the robotic arm.

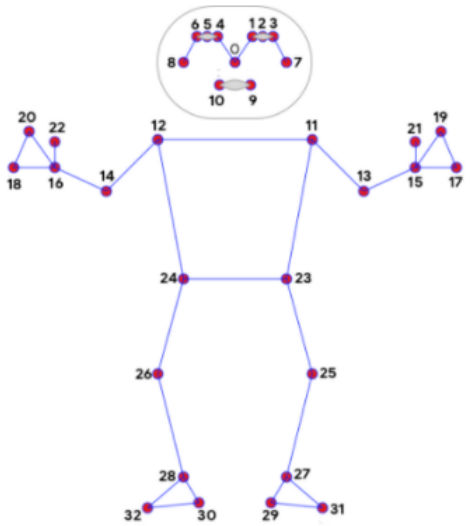
12.2. Implementation principle

MediaPipe Hands infers the 3D coordinates of 21 hand-valued joints from a frame



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

The landmark model in MediaPipe Pose predicts the positions of 33 pose coordinates (see the figure below).



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

In this program, what we need is the coordinates of that part of the right arm. By calculating the angle formed by these coordinates, we can calculate how much angle each servo of the robotic arm needs to rotate.

12.3. Core code analysis PoseArm.py

Code reference path: ~/yahboomcar_ws/src/arm_mediapipe/scripts

- Import key libraries

```
from media_library import *
```

- Obtain information about each joint of the right arm

```
frame, pointArray, lhandptArray, rhandptArray =
self.pose_detector.findHolistic(frame)
#The prototype of this pose_detector.findHolistic function is in
media_library.py
```

- Here is an explanation of how to calculate the angle formed by the shoulder joint, elbow joint, wrist joint, and palm opening and closing:

1. As can be seen from the picture above, the three points 12-11-13 form the shoulder. Therefore, the angle of the shoulder joint can be calculated by getting the coordinate values of these three points;
2. As can be seen from the above figure, the three points 11-13-15 form the elbow. Therefore, the angle of the elbow joint can be calculated by getting the coordinate values of these three points;
3. As can be seen from the above figure, the three points 13-0 (palm wrist joint)-8 (top of index finger) form the wrist joint. Therefore, by getting the coordinates of these three points, the angle of the wrist joint can be calculated. How many;
4. As can be seen from the above figure, the three points 4 (top of thumb) - 0 (palm and wrist joint) - 8 (top of index finger) form the angle of the clamping claw. Therefore, getting the coordinate values of these three points is The angle at which the clamping jaw needs to be clamped can be calculated;

For the specific implementation of the above four key joint codes, you can check the program source code.

12.4. Program flow chart