10. Mediapipe gesture control robotic arm action group

10.1. Introduction

MediaPipe is a data stream processing machine learning application development framework developed and open sourced by Google. It is a graph-based data processing pipeline for building data sources that use many forms, such as video, audio, sensor data, and any time series data. MediaPipe is cross-platform and can run on embedded platforms (Raspberry Pi, etc.), mobile devices (iOS and Android), workstations and servers, and supports mobile GPU acceleration. MediaPipe provides cross-platform, customizable ML solutions for real-time and streaming.

10.2. Using

Note: [R2] of the remote controller has the function of [pause/on] for this gameplay.

The case in this section may run very slowly on the robot master. It is recommended to connect the camera on the virtual machine side and run the file[02_PoseCtrlArm.launch]. The NX master control will work better, you can try it.
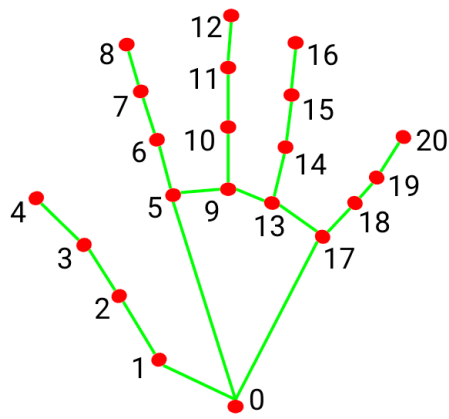
```
roslaunch arm_mediapipe mediaArm.launch          # Robots
rosrun arm_mediapipe FingerCtrl.py        # The robot can also be started in a
virtual machine, but the virtual machine needs to be equipped with a camera
```

After the program is running, press the handle's R2 key to touch the control. The camera will capture the image, there are six gestures, as follows

- Gesture Yes: Robotic arm dancing
- Gesture OK: the robotic arm shakes its head, the gripper is tight
- Gestures contempt (clenched fist, thumbs out, thumbs down): the robotic arm is in a kneeling stance, and the gripper is slack
- Gesture number 1: robotic arm nods
- Gesture rock (the index finger and the little finger are straight, the others are bent): the mechanical arm is straight and turned left and right
- Gesture number 5: robotic arm applauding

Here, when each gesture is finished, it will return to the initial position and beep, waiting for the next gesture recognition.

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

10.4. Core files

10.4.1、mediaArm.launch

```
<launch>
    <include file="$(find yahboomcar_ctrl)/launch/yahboom_joy.launch"/>
    <include file="$(find yahboomcar_bringup)/launch/yahboomcar.launch"/>
    <node pkg="web_video_server" type="web_video_server" name="web_video_server"
output="screen"/>
    <node name="msgToimg" pkg="arm_mediapipe" type="msgToimg.py" output="screen"
required="true"/>
</launch>
```

10.4.2、FingerCtrl.py

```
The implementation process here is also very simple. The main function opens the
camera to obtain data and then passes it into the process function. Inside it,
"detect palm" -> "obtain finger coordinates" -> "obtain gestures" in sequence,
and then decide what needs to be done according to the gesture results action
performed frame, lmList, bbox=
self.hand_detector.findHands(frame) #detect palm
fingers = self.hand_detector.fingersUp(lmList)  #get finger coordinates
gesture = self.hand_detector.get_gesture(lmList)    #get gesture
For the specific implementation process of the above three functions, you can
refer to the content in media_library.py
```

The implementation process here is also very simple. The main function opens the camera to obtain data and then passes it into the process function. Inside it, "detect palm" -> "obtain finger coordinates" -> "obtain gestures" in sequence, and then determine the needs according to the gesture results action performed.

10.5. Flowchart

```
                          ┌──────────┐
                          │   Start   │
                          └──────────┘
                               │
                               ▼
                    ┌─────────────────────┐
                    │ Turn on the camera, get the │
                    │ image, and enter the process │
                    │        function        │
                    └─────────────────────┘
                               │
                               ▼
                 ┌──────────────────────────┐
                 │      Detect palm:        │
                 │   frame.ImList, bbox =   │
                 │ self.hand_detector.findHands(frame) │
                 └──────────────────────────┘
                               │
                               ▼
                      ╱────────────────╲
                     ╱ Determine whether the ╲
                    ╱    remote control      ╲        Y      ┌──────────────────────────┐
                   ╱ is released and whether  ╲─────────────▶│ Enter the thread processing function: │
                    ╲  the palm is detected   ╱              │  get the coordinates of the finger │
                     ╲                       ╱               │          fingers =         │
                      ╲────────────────╱                     │ self.hand_detector.fingersUp(ImList) │
                                                             └──────────────────────────┘
                                                                          │
                                                                          ▼
                         ┌──────────────────────────┐    ┌──────────────────────────┐    ┌──────────────────────────┐
                         │ Calculate finger coordinates and gestures: │    │   Calculate the data of the  │    │ Send it to the chassis and │
                         │ fingers = self.hand_detector.fingersUp(ImList) │──▶│   motion joints of the   │──▶│      execute the       │
                         │ gesture = self.hand_detector.get_gesture(ImList) │    │ robotic arm according to │    │      corresponding     │
                         └──────────────────────────┘    │  the acquired gestures   │    │        operate:        │
                                                         └──────────────────────────┘    │ self.media_ros.pub_arm0 │
                                                                                          └──────────────────────────┘
```