Multimodal Large Model+robotic arm gripping (Voice Version)

Before running the function, you need to close the App and large programs. For the closing method, refer to [4. Preparation] - [1. Manage APP control services].

1. Function Description

After the program runs, wake up the voice module and input robotic arm gripping commands by voice. Based on the type of object to be gripped, the large model will start external programs to control the robotic arm to grip the target object.

2. Startup

Users with Jetson-Nano mainboard version need to enter the docker container first and then input the following command. Users with Orin mainboard can directly open the terminal and input the following command:

```
ros2 launch largemodel largemodel_control.launch.py
```

The robotic arm gripping has the following action commands:

- **Grip by machine code ID**: Sort machine codes 1/2/3/4
- **Grip by machine code height**: Remove machine codes with height higher than x cm (where x represents height), for example: Remove machine codes with height higher than 5 cm
- **Grip by color block height**: Remove y color blocks with height higher than x cm (where x represents height, y represents color with values red/green/yellow/blue), for example: Remove red blocks with height higher than 5 cm
- **Grip other objects**: Grip the small yellow duck on the desktop, grip the blue cuboid on the desktop

2.1. Grip by Machine Code ID

2.1.1. Startup

After waking up the module, input by voice:

```
Sort machine code 2
```

The program will open two terminals, starting the machine code recognition program and gripping program respectively. After recognizing machine code 2, the gripping program will control the robotic arm to descend and grip machine code 2, then place it at the set position, and finally return to the machine code recognition posture to prepare for the next gripping. This continues until no more machine code 2 is recognized, then it reports back to the large model and the task ends.

2.1.2. Task Planning

Plan the action function apriltag_sort(2), where the parameter 2 represents the ID of the machine code to be gripped.

2.1.3. Core Code Analysis

You can refer to the content in **2.1.3. Core Code Analysis** from tutorial [17. AI Model - Text Version] - [Multimodal Large Model+robotic arm gripping]. The voice version and text version have the same action functions, only the task command input method is different.

2.2. Grip by Machine Code Height

2.2.1. Startup

After waking up the module, input by voice:

```
Remove machine codes with height higher than 3 cm
```

Then two terminals will open, starting the machine code recognition height calculation program and gripping program respectively. After recognizing machine codes with height higher than 3 cm, the gripping program will control the robotic arm to descend and grip the abnormal height machine codes, then place them at the set position, and finally return to the machine code recognition posture to prepare for the next gripping. This continues until no more abnormal height machine codes are recognized, then it reports back to the large model and the task ends.

2.2.2. Task Planning

Plan the action function apriltag_remove_higher(30.0), where the parameter 30 represents the height threshold. Machine codes with height higher than this value will be removed, unit is millimeters mm.

2.2.3. Core Code Analysis

You can refer to the content in **2.2.3. Core Code Analysis** from tutorial [17. AI Model - Text Version] - [Multimodal Large Model+robotic arm gripping]. The voice version and text version have the same action functions, only the task command input method is different.

2.3. Grip by Color Block Height

2.3.1. Startup

After waking up the module, input by voice:

```
Remove green blocks with height higher than 3 cm
```

Then two terminals will open, starting the color block recognition height calculation program and gripping program respectively. After recognizing green blocks with height higher than 3 cm, the gripping program will control the robotic arm to descend and grip these blocks, then place them at the set position, and finally return to the recognition posture to prepare for the next gripping. This continues until no more abnormal height green blocks are recognized, then it reports back to the large model and the task ends.

2.3.2. Task Planning

Plan the action function color_remove_higher('green',30.0), 'green' indicates the color block to be sorted is green blocks, the parameter 30 represents the height threshold. Machine codes with height higher than this value will be removed, unit is millimeters mm.

2.3.3. Core Code Analysis

You can refer to the content in **2.3.3. Core Code Analysis** from tutorial [17. AI Model - Text Version] - [Multimodal Large Model+robotic arm gripping]. The voice version and text version have the same action functions, only the task command input method is different

2.4. Grip Other Objects

2.4.1. Startup

After waking up the module, input by voice:

```
Grip the small yellow duck on the desktop
```

The program will first take a photo to determine the position of the small yellow duck, then open two terminals, starting the KCF tracking positioning program and object gripping program respectively. After calculation, the robotic arm will descend to grip the small yellow duck and finally return to the initial posture.

2.4.2. Task Planning

1. Call seehat() to check the position of the small yellow duck;
2. Call grasp_obj(x1,y1,x2,y2) gripping function, where x1,y1,x2,y2 represent the top-left and bottom-right corner coordinates of the small yellow duck's outer bounding box.

2.4.3. Core Code Analysis

You can refer to the content in **2.4.3. Core Code Analysis** from tutorial [17. AI Model - Text Version] - [Multimodal Large Model+robotic arm gripping]. The voice version and text version have the same action functions, only the task command input method is different