Intelligent action control

1. Experimental purpose

This chapter will learn how to combine the online large language model to realize the process of intelligent action control of the robot dog. This case requires the use of audio equipment and cameras.
Notes:

1. **Before running this case, you need to close the startup program**, please refer to the Raspberry Pi system configuration section **9. Open and close the APP control program** This tutorial ends the startup program.
2. You need to fill in the API_KEY of the large model, please refer to the operation method of **AI large model section "1. Prerequisites for using"**.

2. Experimental steps

1. Terminal input

```
cd /home/pi/DOGZILLA/Samples/4_Big_Modle
python3 dog_agent/AIMain_en.py
```

2. Wake-up operation
   The wake-up word is: **hello yahboom**

```
pi@raspberrypi:~/DOGZILLA/Samples/4_Big_Modle $ python3 dog_agent/AIagent_go.py
serial /dev/myspeech open
start
Waiting for keyword...
```

After waking up, you will hear a "ding" sound, and then you can express your thoughts to the robot dog.

```
pi@raspberrypi:~/DOGZILLA/Samples/4_Big_Modle $ python3 dog_agent/AIagent_go.py
serial /dev/myspeech open
start
Waiting for keyword...
Keyword detected: 06.Jun 2025 03:41:17
Playing WAVE './ding.wav' : Signed 16 bit Little Endian, Rate 16000 Hz, Mono
```

3. The robot dog will be processed by the big model to understand the customer's thoughts, and then feedback the corresponding text results and audio playback.

```
Q: Please dance and tell me about yourself.
Agent
**The intelligent agent arranges actions as follows** {'function': ['Dog_Dance()
', 'play_myself_en()'], 'response': 'Dancing gracefully in my robotic body, let
me tell you a little about myself!'}
A:Dancing gracefully in my robotic body, let me tell you a little about myself!
Start executing action Dog_Dance()
### error: on_close() takes 1 positional argument but 3 were given
MPlayer UNKNOWN-12 (C) 2000-2023 MPlayer Team
do_connect: could not connect to socket
connect: No such file or directory
Failed to open LIRC support. You will not be able to use your remote control.

Playing ./dog_agent/DogAgent.mp3.
libavformat version 59.27.100 (external)
Audio only file format detected.
Load subtitles in ./dog_agent/
==========================================================================
Opening audio decoder: [mpg123] MPEG 1.0/2.0/2.5 layers I, II, III
AUDIO: 16000 Hz, 2 ch, s16le, 48.0 kbit/9.38% (ratio: 6000->64000)
Selected audio codec: [mpg123] afm: mpg123 (MPEG 1.0/2.0/2.5 layers I, II, III)
==========================================================================
AO: [pulse] 16000Hz 2ch s16le (2 bytes per sample)
```

4. At this point, the interactive process is over. If you need to express your ideas again, just wake up again.

3. Analysis of the main source code of the experiment

In the "/home/pi/DOGZILLA/Samples/4_Big_Modle/dog_agent/" path, "AIMain_en.py" is a main function entry.

```
# Main function process
    while True:
        if detect_keyword() and huanxin == 0:
            huanxin = 1

            if os.path.exists('./demos/dog_agent/myrec.wav'):
                os.remove('./demos/dog_agent/myrec.wav')
            time.sleep(0.2)

            try:

                start_recording()
                time.sleep(0.2)
                rectext = rec_wav_music_en()
                #rectext = "Tell me what can you see?"
                #print(rectext)
            except:
                print("Cannot hear command,try again")
                huanxin = 0
                continue

            if rectext != "":
                print("Q:"+rectext)

                try:
                    agent_plan_output = eval(Dog_agent_plan_en(rectext))
                    print('**The intelligent agent arranges actions as
follows**', agent_plan_output)
```

```
                response = agent_plan_output['response']
                #print('**Start speech synthesis and play**锛?+response)
            except:
                display_text = "try again..."
                print(display_text)

                huanxin = 0
                continue

            print("A:"+response)

            tts_thread = threading.Thread(target=Speak_Vioce)
            tts_thread.daemon = True
            tts_thread.start()

            for each in agent_plan_output['function']:
                print('Start executing action', each)
                try:
                    eval(each)
                except:
                    continue

            time.sleep(0.5)

        else :
            print("No information was recognized, try again")
            time.sleep(0.5)
            huanxin = 0
```

1. Program flow: detect wake-up words->listen to expression semantics->combine with big model to understand->feedback answers
2. In the path of "/home/pi/DOGZILLA/Samples/4_Big_Modle/dog_agent/", the directory structure description is as follows. Only the files related to this case are listed below

```
├── AIagent_go.py #Main program entry Chinese version
├── AIMain_en.py #Main program entry English version
├── DAgent_en.py #Intelligent agent English version
├── dog_agent.py #Intelligent agent Chinese version
├── dog_API_en.py #Online big model interface English version
├── dog_base_control.py #Basic action interface
├── dog_football_api.py #Football sports interface
├── dog_qa_api.py #Inquiry weather date related interface
├── dog_record.py #Recording interface
├── dog_speak_iat_en.pyc #Speech recognition English version
├── dog_speak_iat.py #Speech recognition Chinese version
├── dog_tts_en.pyc #Audio synthesis English version
├── dog_tts.py #Audio synthesis Chinese version
└── dog_UltraAPI.py #Online large model Chinese version interface
```

3. How to add more instructions and actions

- First open the DAgent_en.py file to add a sample instruction

```
Turn Around:Dog_Turn_Around()
Crawl:Dog_Crawl()
Squat:Dog_Squat()
Three-axis rotation:Dog_3_Axis()
pee:Dog_Pee()
sit down:Dog_Sit_Down()
wave/To greet:Dog_Wave_Hand()
stretch:Dog_Stretch()
Wave motion:Dog_Wave_Body()
Rocking motion:Dog_Swing()
handshake:Dog_Handshake()
dance:Dog_Dance()
Climb Stairs:Climb_The_Stairs()
push-up:Dog_push_up()
Display robotic arm:Dog_show_arm()
The robotic arm moves upwards:arm_up()
Robot arm grasping:arm_middle()
The robotic arm moves downwards:arm_down()
Pick up wooden blocks of the specified color, with a total of four colors: red, yellow, blue, and green,For example, pickin
Pick up wooden blocks of the specified color and place them in their corresponding positions. There are a total of four col
Kick away the balls of the designated color,There are a total of four colors for the balls: red, yellow, blue, and green. F
Scream (Surprise Scream):play_sound_surprised()
Scream (Angry Scream):play_sound_anger()
Introduce yourself: play_ryself()
Rest and wait, such as waiting for two seconds:time.sleep(2)
There are also some color related meanings: for example, the sky color is blue, apples are red, bananas are yellow, and lea
    Here are some executable action groups
【Output JSON format】
You can directly output JSON, starting from {, do not output the beginning or end of JSON containing ```
In the 'function' key, output a list of function names, where each element is a string representing the name and parameters
In the 'response' key, according to my instructions and your choreographed actions, output your reply to me in the first pe
【Here are some specific examples】    Here are some examples of command statements
My instructions: Move forward for 3 seconds, then lie down, show the robotic arm, and finally pee. You output:{'function':[
My instructions:Start exercising.You output:{'function':['Dog_Squat()','Dog_Squat()','Dog_push_up()','Dog_push_up(),Dog_Wav
My instructions:Turn around and help me pick up the yellow wooden block. You output:{'function':['Dog_Turn_Around()','caw_c
My instructions:First, perform three-axis rotation, and then kick the green ball away. You output:{'function':['Dog_Turn_Ar
My instructions:Just describe what you saw, then scream a few times and lie down. You output:{'function':['play_sound_surpr
My instructions:Move forward for 3 seconds, then move the robotic arm a few times, and finally climb the stairs.You output:
My instructions:Show the robotic arm upwards, then turn it around, and finally show the robotic arm downwards. You output:{
My instructions:If you see yellow, turn around; otherwise, dance and finally lie down. You output:{'function':['Dog_Turn_Ar
My instruction: Put the small ball in the color of an apple that I grabbed onto the trash can on the right. You output: {"f
My instructions: Take two steps forward, then sit down and introduce yourself. You output: {"function": ['Dog_forword (2) '
Assuming there are two colors in the picture, my instructions are: if there is only one color, rotate in circles; if there
Assuming a male stranger appears in the picture, my instructions are: help me keep an eye on the door. If a stranger is fou
```

- Then you need to encapsulate some action execution functions and save them in the dog_base_control.py file
  For example, encapsulate a forward function, as shown in the following figure

```
##前进 forword
def Dog_forword(delay_time):
    xgo.move_x(20)
    time.sleep(delay_time)
    xgo.stop()

## 后退 back
def Dog_back(delay_time):
    xgo.move_x(-20)
    time.sleep(delay_time)
    xgo.stop()
```

- Finally, add the interface and usage of the encapsulated function to the DAgent_en.py file.

```
from dog_API_en import *

#DOGZILLA  lite  Action choreography agent description

AGENT_SYS_PROMPT = '''
You are my mechanical dog butler. Please output the corresponding function to be run and your reply to me in JSON format according to my instructions

【Here is an introduction to all built-in functions】
Forward movement:Dog_forword(time)  #Among them, time represents the number of seconds of the action,Advance 1 second:Dog_forword(1)
Step back action:Dog_back(time)  #Among them, time represents the number of seconds of the action,Step back for 1 second:Dog_forword(1)
Left translation action:Dog_Left_move(time)  #Among them, time represents the number of seconds of the action,Left shift for 1 second:Dog_Left_move(1)
Right translation action:Dog_Rihgt_move(time)  #Among them, time represents the number of seconds of the action,Right translation for 1 second:Dog_Rihgt_move(1)
Left rotation action:Dog_LeftTurn(time) #Among them, time represents the number of seconds of the action,Rotate left for 1 second:Dog_Rihgt_move(1)
Right rotation action:Dog_RightTurn(time) #Among them, time represents the number of seconds of the action,Rotate right for 1 second:Dog_Rihgt_move(1)
Looking up action:Dog_Looking_up()
```

4. How to replace the large model interface
   This involves a lot of DIY operations, and it is not recommended for novices to replace the model.

The large model used in this function is OpenRouter

- You can first start from the python version program of the platform corresponding to the interface, and fill in the necessary information according to the platform interface and instructions.
- Then encapsulate the executable file into a function. You can refer to the method of "dog_API_en.py" and put it in the directory of "/home/pi/DOGZILLA/Samples/4_Big_Modle/dog_agent/". For example, the added file name is "mychatgpt.py"

```
xinghou_UltraAPI.py   dog_record.py   audio.py   dog_API_en.py   DAgent_en.py
from openai import OpenAI
import base64
import os,sys
current_dir = os.path.dirname(os.path.abspath(__file__))
parent_dir = os.path.dirname(current_dir)
sys.path.append(parent_dir)
from API_KEY import *


def dogGPT_en(inputtext):
    client = OpenAI(
    base_url="https://openrouter.ai/api/v1",
    api_key=openAI_KEY,
    )

    completion = client.chat.completions.create(

    model="google/gemini-2.5-pro-exp-03-25:free",
    #model="qwen/qwen2.5-vl-32b-instruct:free",
    #model="meta-llama/llama-4-maverick:free",
    #model="nvidia/llama-3.1-nemotron-ultra-253b-v1:free",
    messages=[
        {
        "role": "user",
        "content": [
            {
            "type": "text",
            "text": inputtext
```

- Open the file **DAgent_en.py**

```
from dog_API_en import *

```

- Take this tutorial as an example: then add from dog_API_en import * from mychatgpt import * in the head

```
API_KEY.py   DAgent_en.py
from dog_API_en import *
from mychatgpt import *
```

- Then find this place and replace it with your own encapsulated API function interface.

5. If you want to change the threshold for recording start and the duration of recording, you can change this file and enter in the terminal

```
nano /home/pi/DOGZILLA/Samples/4_Big_Modle/dog_agent/dog_record.py
```

Change the recording part of this file as shown in the figure below

```
quitmark = 0
automark = True
def start_recording(timel = 3,save_file=SAVE_FILE):
    global automark,quitmark
    start_threshold = 3000 #30000    Start threshold
    end_threshold = 1500 #2000      Stop threshold
    endlast = 15  # The total number of times detected is lower than the stop threshold, and the recording is automatically stopped
    max_record_time = 10 #The maximum recording duration

    CHUNK = 1024
    FORMAT = pyaudio.paInt16
    CHANNELS = 1
    RATE = 16000
    WAVE_OUTPUT_FILENAME = save_file
```

Parameter meaning:

- start_threshold = 3000 #Start recording when a sound higher than this value is detected. This value changes according to the environment
- end_threshold = 1500 #Sound lower than this value is detected. This value changes according to the environment
- endlast = 15 #Stop recording when the number of sounds lower than end_threshold is detected. Here it is 15 times
- max_record_time = 5 #The duration of the recorded audio, here is 5
  Note: start_threshold must be greater than end_threshold (start_threshold>end_threshold)
  Generally, the ideal value of end_threshold is half of start_threshold, which can be adjusted according to your own environment.

6. If you feel that the recorded audio cannot be recognized by the online large model because the sound is too small, you can adjust the value here to amplify the recorded audio.
   Terminal input

```
nano /home/pi/DOGZILLA/Samples/4_Big_Modle/dog_agent/dog_record.py
```

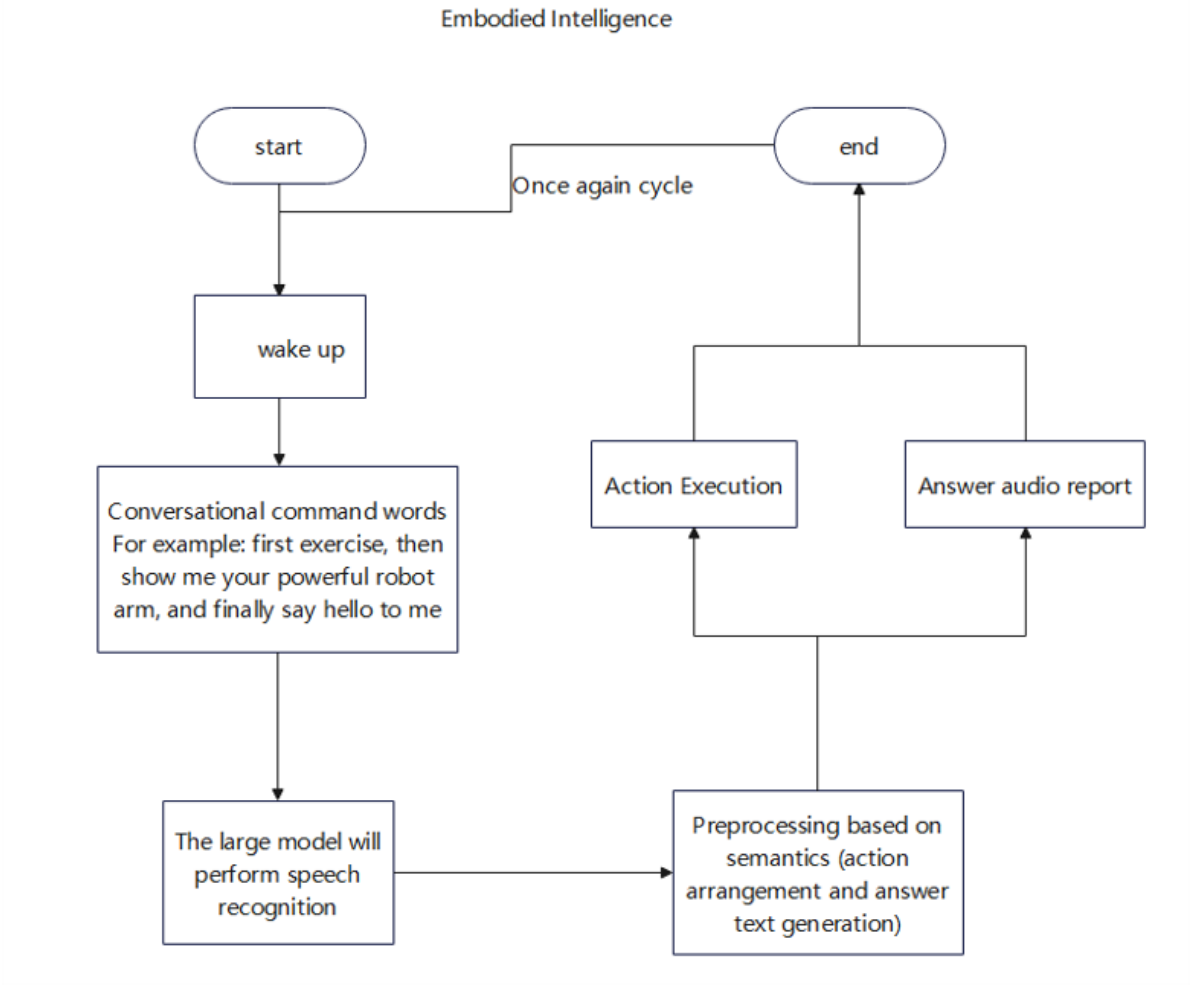
```
        wf.close()
        print(f"The recording has been saved as: {WAVE_OUTPUT_FILENAME}")

        amplify_audio_librosa("recorded_audio.wav", "recorded_audio.wav", gain_factor=5.0) #放大它 Enlarge it
```

Here it is amplified 5 times, here you can make an adjustment according to the distance of the sound source.
Note: If the distance is too far to record audio at all, adjusting the parameters here will be meaningless.
It is recommended that the distance from the sound source of the recorded audio should not be greater than 1.5m.

4. Experimental summary

Based on the above description, the flowchart of this case is as follows:



If you don't know what this case means, here are some reference examples

Examples:

1. A series of action instructions, such as: dance, turn in a circle, push-ups, move forward for 3s, wave your hand and get down.
2. Some action instruction sentences, such as: first show your body, then turn in a circle and lie down, and finally say hello to me.
3. Action instructions + some questions, such as: dance, bark twice, and finally tell me today's date.

Notes

1. If this error occurs when the program starts, you can press "ctrl+C" to end the program and then restart it.

```
serial /dev/myspeech open
Network check failed: HTTPConnectionPool(host='www.baidu.com', port=80): Max retries exc
eeded with url: / (Caused by NewConnectionError('<urllib3.connection.HTTPConnection obje
ct at 0x7fff84058610>: Failed to establish a new connection: [Errno -3] Temporary failur
e in name resolution'))
检测网络没连上，请重启网络
```

2. If you want to terminate this case, press "ctrl+C" to end the program.