6.Gesture recognition fixed action

To apply for a Baidu account, you can read the process of gesture recognition in Chapter 8. This tutorial is based on the successful recognition of gestures by gesture recognition.

1.1 Main code content

Code path：/home/jetson/Dofbot/5.AI_Visual/1.gesture_action.ipynb

```
#bgr8 to jpeg format
import enum
import cv2
def bgr8_to_jpeg(value, quality=75):
 return bytes(cv2.imencode('.jpg', value)[1])
```

```
#Import related modules
import threading
import time
from Arm_Lib import Arm_Device
# Create robot arm object
Arm = Arm_Device()
time.sleep(.1)
```

```
import cv2
import time
import demjson
import pygame
from aip import AipBodyAnalysis
from aip import AipSpeech
from PIL import Image, ImageDraw, ImageFont
import numpy
import ipywidgets.widgets as widgets
# For specific gestures, please see official information.
https://ai.baidu.com/ai-doc/BODY/4k3cpywrv
hand={'One':'number 1','Two':'number 2','Three':'number 3','Four':'number 4',
 'Five':'number 5', 'Six':'number 6','Seven':'number 7',
 'Eight':'number 8','Nine':'number 9','Fist':'Fists','Ok':'OK',
 'Prayer':'pray','Congratulation':'bow','Honour':'farewell',
 'Heart_single':'Expresses heart','Thumb_up':'like','Thumb_down':'Diss',
 'ILY':'I love you','Palm_up':'Palm up','Heart_1':'Showing your heart with your
hands 1',
 'Heart_2':'Showing your heart with your hands 2','Heart_3':'Showing your heart
with your hands 3','Rock':'Rock',
 'Insult':'Erect middle finger','Face':'Face'}
# The following keys should be replaced with your own
""" human body analysis APPID AK SK """
APP_ID = '18550528'
API_KEY = 'K6PWqtiUTKYK1fYaz13O8E3i'
SECRET_KEY = 'IDBUII1j6srF1XVNDX32I2WpuwBWczzK'
client = AipBodyAnalysis(APP_ID, API_KEY, SECRET_KEY)
```

```
g_camera = cv2.VideoCapture(0)
g_camera.set(3, 640)
g_camera.set(4, 480)
g_camera.set(5, 30) #Set frame rate
g_camera.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter.fourcc('M', 'J', 'P', 'G'))
g_camera.set(cv2.CAP_PROP_BRIGHTNESS, 40) #Set brightness -64 - 64 0.0
g_camera.set(cv2.CAP_PROP_CONTRAST, 50) #Set contrast -64 - 64 2.0
g_camera.set(cv2.CAP_PROP_EXPOSURE, 156) #Set exposure value 1.0 - 5000 156.0
ret, frame = g_camera.read()
```

```
# Define the camera display component
image_widget = widgets.Image(format='jpeg', width=600, height=500) #Set up the camera display component
display(image_widget)
image_widget.value = bgr8_to_jpeg(frame)
```

```
# Define conversion display Chinese function
def cv2ImgAddText(img, text, left, top, textColor=(0, 255, 0), textSize=20):
 if (isinstance(img, numpy.ndarray)): # Determine whether OpenCV image type
 img = Image.fromarray(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
 # Create an object that can draw on the given image
 draw = ImageDraw.Draw(img)
 # Font format
 fontStyle = ImageFont.truetype(
 "simhei.ttf", textSize, encoding="utf-8")
 # Draw text
 draw.text((left, top), text, textColor, font=fontStyle)
 # Convert back to OpenCV format
 return cv2.cvtColor(numpy.asarray(img), cv2.COLOR_RGB2BGR)
```

```
#Define the servo angle at different positions
look_at = [90, 164, 18, 0, 90, 90]
p_Prayer = [90, 90, 0, 180, 90, 180] #pray
p_Thumb_up = [90, 90, 90, 90, 90, 180] #Thumb_up
p_Heart_single = [90, 0, 180, 0, 90, 30] #One hand expresses heart
p_Eight = [90, 180, 18, 0, 90, 90] #Eight
p_Congratulation = [90, 131, 52, 0, 90, 180] #Congratulation
p_Rock = [90, 0, 90, 180, 90, 0] #rock
p_fist = [90, 90, 0, 0, 90, 0] #fist
p_horse_1 = [90, 7, 153, 19, 0, 126] #
p_horse_2 = [90, 5, 176, 0, 0, 180]
p_horse_3 = [90, 62, 158, 0, 0, 0]
global running
running = 0
```

```
# Define the mobile robot arm function to simultaneously control the movement of servos No. 1-6, p=[S1, S2, S3, S4, S5, S6]
def arm_move_6(p, s_time = 500):
 for i in range(6):
 id = i + 1
 Arm.Arm_serial_servo_write(id, p[i], s_time)
 time.sleep(.01)
 time.sleep(s_time/1000)

# Define pony sports
def horse_running():
 Arm.Arm_serial_servo_write(6, 150, 300)
 time.sleep(.3)
 Arm.Arm_serial_servo_write(6, 180, 300)
 time.sleep(.3)
```

```
global g_state_arm
g_state_arm = 0
def ctrl_arm_move(index):
 global running
 if index == "Prayer":
 arm_move_6(p_Prayer, 1000)
 time.sleep(1.5)
 arm_move_6(look_at, 1000)
 time.sleep(1)
 elif index == "Thumb_up":
 s_time = 500
 Arm.Arm_serial_servo_write(6, 180, s_time)
 time.sleep(s_time/1000)
 Arm.Arm_serial_servo_write(6, 90, s_time)
 time.sleep(s_time/1000)
 Arm.Arm_serial_servo_write(6, 180, s_time)
 time.sleep(s_time/1000)
 Arm.Arm_serial_servo_write(6, 90, s_time)
 time.sleep(s_time/1000)
 elif index == "Ok":
 s_time = 300
 Arm.Arm_serial_servo_write(4, 10, s_time)
 time.sleep(s_time/1000)
 Arm.Arm_serial_servo_write(4, 0, s_time)
 time.sleep(s_time/1000)
 Arm.Arm_serial_servo_write(4, 10, s_time)
 time.sleep(s_time/1000)
Arm.Arm_serial_servo_write(4, 0, s_time)
 time.sleep(s_time/1000)
 elif index == "Heart_single":
 arm_move_6([90, 90, 90, 90, 90, 90], 800)
 time.sleep(.1)
 arm_move_6(p_Heart_single, 1000)
 time.sleep(1)
 elif index == "Five":
 arm_move_6(look_at, 1000)
 time.sleep(.5)
 elif index == "Eight":
 s_time = 300
 arm_move_6(p_Eight, 0)
 time.sleep(1)
 Arm.Arm_serial_servo_write(2, 165, s_time)
 time.sleep(s_time/1000)
 elif index == "Rock": #rock
 Arm.Arm_serial_servo_write6_array(p_Rock, 1300)
 time.sleep(3)
 Arm.Arm_serial_servo_write6_array(look_at, 1000)
 time.sleep(1)
 elif index == "Thumb_down": #Thumb_down
 Arm.Arm_serial_servo_write6_array(p_horse_1, 1300)
 time.sleep(1)
 elif index == "Congratulation": #Congratulation
 Arm.Arm_serial_servo_write6_array(p_horse_2, 1000)
 time.sleep(1)
 running = 1
 while running == 1:
 horse_running()
 elif index == "Seven": #number 7
 Arm.Arm_Buzzer_On(8) #The buzzer automatically sounds for 0.5 seconds
 Arm.Arm_serial_servo_write6_array(p_horse_3, 1000)
 time.sleep(2)
 Arm.Arm_serial_servo_write6_array(look_at, 1000)
 time.sleep(1)

 global g_state_arm
 g_state_arm = 0
```

```
#Let the robotic arm move to the position where the camera looks forward
arm_move_6(look_at, 1000)
time.sleep(1)
```

```
def start_move_arm(index):
 # Open the robot arm control thread
 global g_state_arm
 global running
 if g_state_arm == 0:
 closeTid = threading.Thread(target = ctrl_arm_move, args = [index])
 closeTid.setDaemon(True)
 closeTid.start()
 g_state_arm = 1

 if running == 1 and index == "Seven":
 running = 0
```

```
# main process
try:
     Arm.Arm_Buzzer_On(1)
     s_time = 300
     Arm.Arm_serial_servo_write(4, 10, s_time)
     time.sleep(s_time/1000)
     Arm.Arm_serial_servo_write(4, 0, s_time)
     time.sleep(s_time/1000)
     Arm.Arm_serial_servo_write(4, 10, s_time)
     time.sleep(s_time/1000)
     Arm.Arm_serial_servo_write(4, 0, s_time)
     time.sleep(s_time/1000)

     while True:
         """1.Photograph """
```

```
        ret, frame = g_camera.read()
        #image = get_file_content('./image.jpg')
        """ 2.Invoking gesture recognition """
        raw = str(client.gesture(image_widget.value))
        text = demjson.decode(raw)
        try:
        res = text['result'][0]['classname']
        except:
            # print('Recognition results: Nothing was recognized~' )
            # img = cv2ImgAddText(frame, "Not recognized", 250, 30, (0, 0 ,
255), 30)
            img = frame
        else:
            # print('Recognition results: ' + hand[res])
            # img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
```

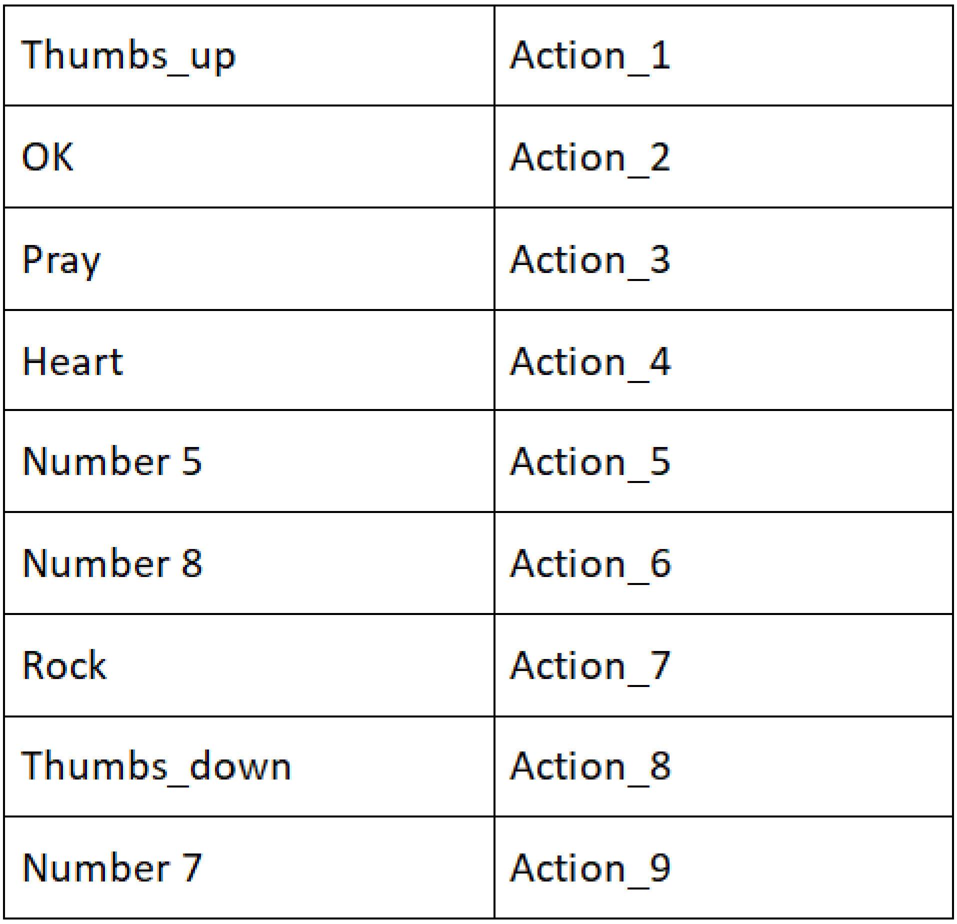
```
        if res == 'Prayer': # 1 pray
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == 'Thumb_up':# 2 like
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == 'Ok': # 3 OK
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == 'Heart_single': # 4 One hand expresses heart
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == 'Five': # 5 number 5
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == "Eight": # number 8
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)

        elif res == "Rock": # rock
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == "Congratulation": # Congratulation
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == "Seven": # number 7
            print('Recognition results: ' + hand[res])
            img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
            start_move_arm(res)
        elif res == "Thumb_down": # Thumb_down
            print('Recognition results: ' + hand[res])
```

```
              img = cv2ImgAddText(frame, hand[res], 250, 30, (0, 255 , 0), 30)
              start_move_arm(res)

          else:
             img = frame
 image_widget.value = bgr8_to_jpeg(img)
except KeyboardInterrupt:
     print(" Program closed! ")
     pass
```

If the set gesture action is recognized, the robotic arm will perform the corresponding action. The current gestures and actions correspond to the following:



Thumbs_up	Action_1
OK	Action_2
Pray	Action_3
Heart	Action_4
Number 5	Action_5
Number 8	Action_6
Rock	Action_7
Thumbs_down	Action_8
Number 7	Action_9