

Facial Recognition

Facial Recognition

1. Experimental Purpose
2. Experimental Steps
3. Experimental Results
4. Experimental Source Code Analysis

This tutorial is specific to the CM5 version and cannot be used directly with the CM4 version

1. Experimental Purpose

Learn how the robot dog can recognize faces after collecting facial image data.

2. Experimental Steps

1. Log in to the robot dog system, exit the robot dog program, and enter "ip (the robot dog's IP address):8888" in the browser. Once logged in, enter the password "yahboom".

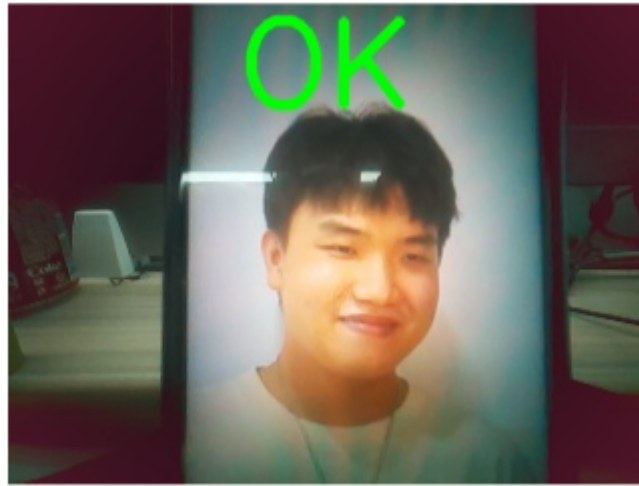


Password:

2. Log in and run the **face_recognition_Class.ipynb** file in the **cd ~/DOGZILLA_Lite_class/5.AI Visual Recognition Course/18. Face Recognition** directory.
3. Run the program

3. Experimental Results

1. The screen will display a live image. Press the button in the upper-right corner of the screen to capture a face.
2. Each time a face is captured, the screen will display "OK," but this does not indicate successful recognition.



开始人脸录入
拍照成功

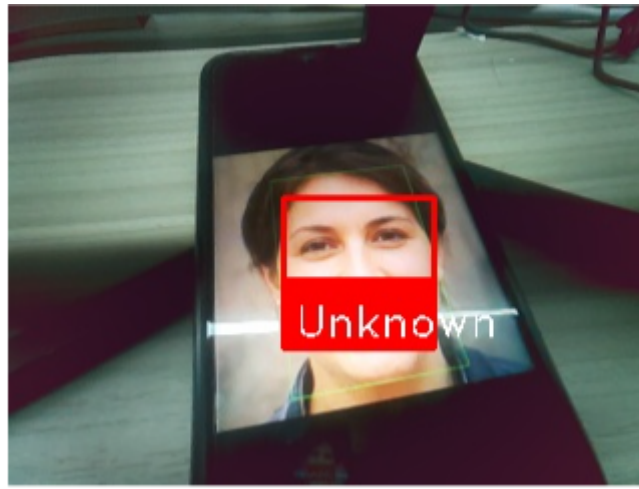
3. Press the button in the upper-left corner of the screen to perform face recognition. The program will display two results: 1. Successful capture, indicating face recognition is possible.
2. Failed capture, indicating that face recognition is unsuccessful and requires re-capturing.
4. After a captured face is recognized, the screen will display the image's serial number.

Encoded:



开始人脸录入
拍照成功
拍照成功
退出人脸录入模式，开始识别
./Face_P/1.jpg
./Face_P/2.jpg
Face input successful

Unencoded:



5. Restarting this program will not save previously encoded faces; you will need to re-encode them for recognition.

4. Experimental Source Code Analysis

```
def take_photo():
    global i
    show_ok = False # 控制是否显示OK字样 Control whether the word 'OK' is
displayed
    ok_display_time = 0 # 记录显示OK的时间 Record the time when OK is displayed

    while True:
        frame = picam2.capture_array()

        #cv2.imshow('frame', frame)
        image_widget.value = bgr8_to_jpeg(frame)

        b, g, r = cv2.split(frame)
        img = cv2.merge((r, g, b))
        imgok = Image.fromarray(img)
        mydisplay.ShowImage(imgok)

        if button.press_d():
            cv2.imwrite(f"./Face_P/{i}.jpg", frame)

while (1):
    frame = picam2.capture_array()
    face_locations = face_recognition.face_locations(frame)
    face_encodings = face_recognition.face_encodings(frame, face_locations)
    # 在这个视频帧中循环遍历每个人脸 Loop through each face in this video frame
    for (top, right, bottom, left), face_encoding in zip(
        face_locations, face_encodings):
        # 看看面部是否与已知人脸相匹配。 Check if the face matches a known face.
        for i, v in enumerate(total_face_encoding):
            match = face_recognition.compare_faces(
                [v], face_encoding, tolerance=0.5)
            name = "Unknown"
            if match[0]:
                total_image_name[i] =
total_image_name[i].replace(folder_path, "")
                name = total_image_name[i] #str(i+1)
                name = 'NO.' + name
                break
```

```

# 画出一个框，框住脸 Draw a frame to enclose the face
cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)
# 画出一个带名字的标签，放在框下 Draw a label with a name and place it under
the box
cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255),
cv2.FILLED)
font = cv2.FONT_HERSHEY_DUPLEX
try:
    cv2.putText(frame, name, (left + 6, bottom - 6), font, 0.7,
(255, 255, 255), 1)
except :
    name = "Unknown"
    cv2.putText(frame, name, (left + 6, bottom - 6), font, 0.7,
(255, 255, 255), 1)
# 显示结果图像 Display result image
image_widget.value = bgr8_to_jpeg(frame)

```

take_photo(): This function is mainly used to collect faces.

while (1): This loop is used to identify the collected faces. If the face is recognized, the face number will be displayed on the screen. If it cannot be recognized, "Unknown" will be displayed.