程序源代码和运行环境说明

✓ 1、程序源代码

1.1、主函数代码

> **本次课程大作业的目标是使用基于opencv完成人脸识别，为了完成这个程序需要有一个程序的主入口，这个主入口就是main.py**

首先，因为要使用opencv，所以需要引用opencv库，因此第一段程序为

```
import cv2
```

在引用完成后，我们需要使用它打开摄像头来进行人脸识别

```
# 打开摄像头
def video_demo():
    capture = cv.VideoCapture(0)
    if not capture.isOpened():
        print("Cannot open camera")
        exit()

    while (True):
        Read_ref, Read_frame = capture.read()
        Read_Image_Data(Read_frame)
        cv.imshow("image", Read_frame)
        Key_Data = cv.waitKey(1) & 0xff
        if Key_Data == 27:
            capture.release()
            break
```

这段代码的运行逻辑为

首先打开摄像头并把数据传递给**capture**这个变量

然后判断这个变量是否被打开，如果没有被打开就输出**"Cannot open camera"**并退出程序

如果摄像头被成功打开，那么进入死循环并开始执行**capture.read，**读取出来的数据存放到**Read_ref, Read_frame**

随后通过**Read_Image_Data**这个自定义函数进行读取图像的处理

然后在把读取到的图像数据在PC上显示出来，它的窗口title为"image"

随后对键盘按键进行读取并把数据存放到Key_Data当中，再然后判断Key_Data的值，如果等于27，也就是键盘上的按键ESC则退出程序

> **以上代码只是实现了打开摄像头并将其画面显示在电脑上，如想要制作人脸识别，则还需要对读取出的图像进行处理**

```
def Read_Image_Data(image_Data):
    # 读取识别的数据
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    recognizer.read('./Data/trainer/trainer.yml')
    # 读取图像
    # image_Data = cv.imread("image.jpg")
    # 把图像转换为灰度图
    gray_Data = cv.cvtColor(image_Data, cv2.COLOR_BGR2GRAY)
    # 加载人脸识别器
    face_cascade = cv.CascadeClassifier(r"./Data/haarcascade_frontalface_alt2.xml")
    # 检测灰度图中的所有面孔
    Face_Out_Data = face_cascade.detectMultiScale(gray_Data,1.01,5,0,(100,100),(300,300))
    # 对识别出来的图像进行画框
    for x, y, width, height in Face_Out_Data:
        # 人脸识别
        id, confidence = recognizer.predict(gray_Data[y:y + height, x:x + width])
        # 这里的color是 蓝 黄 红，与rgb相反，thickness设置宽度
        cv.rectangle(image_Data, (x, y), (x + width, y + height), color=(255,0,255), thickness=3)
    # 保存输出图像
    # cv.imwrite("beauty_detected.jpg", image_Data)
        name = names[id]
        cv.putText(image_Data,name,(x,y),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,255),2)

```

这段代码就是对读取出来的图像进行处理的代码，首先先读取自己提前做好的识别人脸数据的数据集

recognizer = cv2.face.LBPHFaceRecognizer_create()

recognizer.read('./Data/trainer/trainer.yml')

然后就是对输入的图像数据进行灰度处理并将其存放到gray_Data这个变量当中

随后就是加载别人做好的数据集来检测图像中的人脸，加载的数据集存放到face_cascade这个变量

再然后检测灰度图中的所有人脸数据，并存档到Face_Out_Data

然后就是对识别出来的人脸进行画框，并表明人脸的相关数据，例如ID、姓名

> **在判断完成后就可以把所有代码整合起来，形成一个main函数**

```
if __name__ == '__main__':
    video_demo()
    cv.destroyAllWindows()
```

在主函数当中，第一句话是使用**video_demo**函数，第二句话是关闭所有的打开的窗口，也就是当退出程序时，需要清除已经打开的窗口

1.2、个人数据集创建代码

> **由于需要比较准确的识别出不同的人，并且给他们打上不同的id和姓名，因此需要这样一个个人数据集创建的代码**

首先需要引用比较多的头文件

```
import cv2 as cv
import numpy as np
from PIL import Image
import os
```

引用完成后就需要对读取到的数据进行处理

```
def getImageAndLabels(Path):
    # 人脸数据
    facesSamples = []
    # 存储姓名数据
    ids = []
    # 存储图片信息
    imagePaths = [os.path.join(Path,f)for f in os.listdir(Path)]
    # 加载分类器
    face_detector = cv.CascadeClassifier('./Data/haarcascade_frontalface_default.xml')
    for imagePath in imagePaths:

        PIL_img = Image.open(imagePath).convert('L')

        img_numpy = np.array(PIL_img,'uint8')

        faces = face_detector.detectMultiScale(img_numpy)

        id = int(os.path.split(imagePath)[1].split('.')[0])

        for x,y,w,h in faces:
            ids.append(id)
            facesSamples.append(img_numpy[y:y+h,x:x+w])
    print('id:',id)
    print('fs:',facesSamples)
    return facesSamples,ids
```

这段代码的运行逻辑为

首先创建相关的人脸数据**facesSamples**和姓名数据**ids**，还有图片的信息**imagePaths**

然后通过加载别人制作好的人脸分类器对人脸进行识别并将其存放到**face_detector** 这个变量

然后对识别样本中的人脸数量进行循环处理（如果Path当中只有一张图片，那么样本数量就是1）

识别出来后将其**id**和相关**faces**数据保存

然后就是保存读取出来的图片样本的框的大小

然后输出相关数据的**id**和**facesSamples**

> **数据处理函数完成后，需要对此函数进行调用并且需要读取的数据集里面有相关的数据才可以进行识别**

```
if __name__ == '__main__':
    Path = './Data/MyPath/'
    faces,ids = getImageAndLabels(Path)
    recognizer = cv.face.LBPHFaceRecognizer_create()
    recognizer.train(faces,np.array(ids))
    recognizer.write('./Data/trainer/trainer.yml')
```

读取**./Data/MyPath/**的数据，然后传入**getImageAndLabels**函数，并得到**faces,ids** ，随后对读取出来的所有数据写入**./Data/trainer/trainer.yml**然后给主函数调用

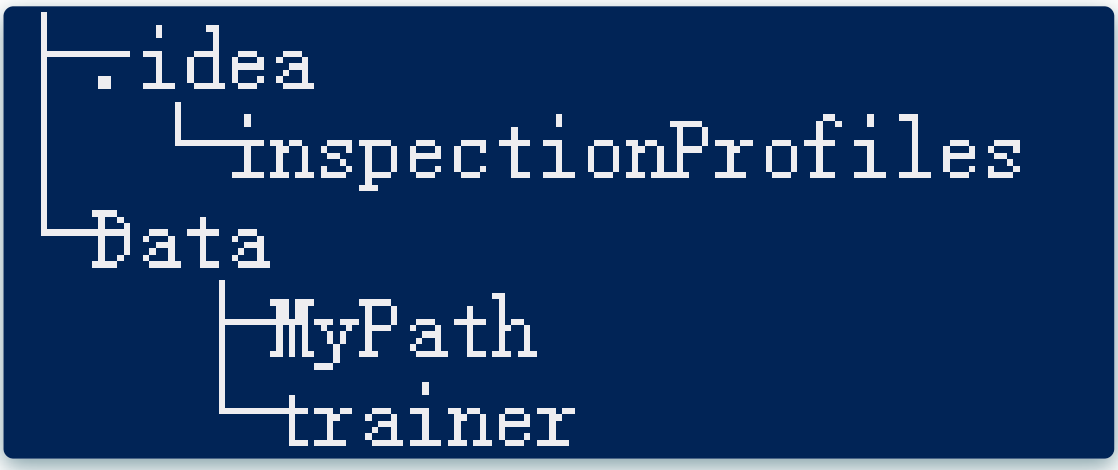
✓ 2、运行环境说明

> 本程序编写基于Python3.11环境

> 本程序需要安装较多的软件包才可以正常运行
>
> 但是主要的安装包为以下

软件包	版本	最新版本
PIL-Tools	1.1.0	
Pillow	10.0.1	
certifi	2023.7.22	2023.7.22
chardet	5.2.0	
charset-normalizer	3.3.1	3.3.1
contourpy	1.1.1	
cycler	0.11.0	▲ 0.12.1
fonttools	4.42.1	▲ 4.43.1
idna	3.4	
kiwisolver	1.4.5	1.4.5
matplotlib	3.8.0	
numpy	1.26.1	
opencv-contrib-python	4.8.1.78	
opencv-python	4.8.1.78	
packaging	23.1	
pip	23.3.1	
pyparsing	3.1.1	3.1.1
python-dateutil	2.8.2	2.8.2
requests	2.31.0	2.31.0
setuptools	65.5.1	
six	1.16.0	
urllib3	2.0.7	2.0.7
wheel	0.38.4	▲ 0.41.2

> 本程序的文件结构为



其中，在程序的根目录一共有2个**py**文件，main.py是进行打开摄像头进行图像识别的主程序，main2.py是进行个人数据集创建的程序

在Data文件夹下的MyPath文件夹当中是个人的训练集图片

在Data文件夹下的trainer文件夹当中是训练完成后输出的trainer.yml文件

在Data文件夹当中的文件是别人训练好的相关数据集