目录

感谢一篇博客教会我怎么把md导出为html
用vscode插件进行PDF转换

本来想开发一个门禁的

但是想了想，用人脸识别做一个开盖儿的垃圾桶也挺有意思的

于是最后开发了一个垃圾桶

还是有很多不足的地方，以后还需努力学习

这个是项目的地址

地址在这儿idoor_1.0

课程1：关于图片的imshow问题

最开始以为是显示的问题

但是多次调试之后发现，其实是图片尺寸的问题

解决方法参考了以下博客解决方法

最下方的图片尺寸问题的解决方法

课程2 关于图片灰度转换中的问题

我遇到的问题是，使用了Imwrite后却保存失败
也就是本地路径上并没有灰度后的图片

怎么使用灰度convert和怎么保存
`cv.imwrite()` 是很有用的一个函数
，但是如何让他保存到指定位置呢?

我的初始代码

```
cv.destroyWindow('gray_dzr')
cv.imwrite('./gray_dzr.jpg',gray_img)
```

结果将关闭特定窗口改为关闭所有窗口以后竟然就可以正常打开关闭了，太离谱

修改后的代码如下

```
cv.destroyAllWindows()
cv.imwrite('./gray_dzr.jpg',gray_img)
```

然后就可以正常保存了

捏马，竟然不改这个也可以正常允许，神魔Bug

一个小小idea

如果有多个图片，那么怎么实现多个文件的命名呢

我的代码如下：

```
for num in range(1,10):
    filename ='gray_dzr'+str(num)+'.jpg'
    file_exist=open(filename,mode='r')
    if not file_exist:
        cv.imwrite(filename,gray_img)
        file_exist.close()
    num += 1
    break
else:
        print(str(filename))
```

但是发现了许多问题，所以我使用chatgpt帮助我更正了一下

```
import os
import cv2 as cv

gray_img = ...  # Assuming you have your gray image

for num in range(1, 11):
    filename = 'gray_dzr' + str(num) + '.jpg'

    if os.path.exists(filename):
        print(f"File '{filename}' already exists.")
    else:
        cv.imwrite(filename, gray_img)
        print(f"Created file '{filename}'.")

    if num >= 9:  # To limit the loop to 1-10
        break
```

非常优美的解决方案，学习到了 `print(f"a real string '{名称}' 写法")`

也就是打印标准字符串
还学到了os的包操作检查文件是否存在

拓展一下的话：可以想一下怎么去实现自动的：
新建一个图片文件夹->放入图片

3.图片裁剪

`cv.resize(img,dsize=(length,width))` 可以实现对图片尺寸的裁剪

`ord('任意键')` 可以用来检测是否按到某一个键

4.画框（没啥好说的）

5.识别人脸

重头戏！
我觉得我没学明白

总之需要自己定义一个人脸识别函数出来

但是所需要的分类器则实际上是opencv预设好的了，如果需要拓展的话就要学习一下怎么去自己设置分类器啥的

发现**问题一**，在使用opencv自带的分类器时，不能将带有中文的路径带入进去，不然会出现打不开文件的情况

重点在思路：

先二值化（灰度处理），再进行分类器识别，最后复筛几次，获取参数，然后原图像画框

```
def face_detect_cmd():
    #1：先将原图片灰度化以后再使用
    img_gray=cv.cvtColor(img,cv.COLOR_RGB2GRAY)
    cv.namedWindow('zzy', cv.WINDOW_NORMAL)
    cv.resizeWindow('zzy', 1000, 1000)
    cv.imshow('zzy', img_gray)
    cv.waitKey(0)
    #2:使用已经训练好的分类器(需要额外学习到底怎么制作的分类器)
    face_detect = cv.CascadeClassifier('D:/opencv/sources/data/haarcascades/haarcascade_frontalface_alt2.xml')
    #3：多次识别增加准确度
    face = face_detect.detectMultiScale(img_gray,1.01,5,0,(100,100),(900,900))
    #读取face参数
    for x, y, w, h in face:
        # 画框
        cv.rectangle(img, (x, y), (x + w, y + h), color=(0, 0, 255), thickness=6)
    cv.namedWindow('zzy', cv.WINDOW_NORMAL)
    cv.resizeWindow('zzy', 1000, 1000)
    cv.imshow('zzy', img)
```

需要注意的是，这个分类器一次性可以识别同一张照片中的多个脸并框选出来，如果分类效果不够理想的话可以对分类器进行重选

打开摄像头咯

关键代码很简单

```
#摄像头打开
cam=cv.VideoCapture(0)
#进行人脸识别
while True:
    flag, frame = cam.read()
    if not flag:
        break
    face_detect_cmd(frame)
    if ord('q')==cv.waitKey(0):
        break

cv.destroyAllWindows()
cam.release()
```

注意，VideoCapture()括号中的可以填视频文件，这样可以实现对指定视频的捕获
但是一定不要忘了cam.release()，这一步释放很重要

如果是VideoCapture(0)，那就是调用系统摄像头

但是有一个问题，那就是调用摄像头以后他不能自己刷新，需要我按空格它才刷新，不知道该怎么解决

原来是Waitkey的锅，只需要将waitkey的时间缩短就可以了，比如waitkey(1)

此外需要注意，read函数返回的是一个二元组，第一个值是一个bool值，第二个值是图像对象
所以 `flag, frame = cam.read()` 成对出现才是正确的写法

6.人脸的识别

所需要用到的工具有：numpy,PIL(图像处理工具库)

这是一个有一丢丢复杂的过程

找到训练文件夹->识别并提取出训练文件夹中图片的标签和特征->训练->存放

我们依次来理解一下

首先是训练文件夹，我们先在纯英文路径下存放一个图片文件夹，里面都是我们要进行特征标注的文件

其次就是提取过程了,我们用faces和ids来存放这些图片对应的面部特征和ids

那么具体是怎么提取的呢，这个需要我们自己来编写对应的函数，具体过程如下：

```
def GetImagesandLabels(Path):
    #特征值储存
    facesSamples=[]
    #标签储存
    ids=[]
    #给出所有子文件的路径
    imagePaths=[os.path.join(Path,f) for f in os.listdir(Path)]
    #加载分类器
    face_detector =cv.CascadeClassifier("D:/Edge下载/opencv/sources/data/haarcascades/haarcascade_frontalface_alt.xml")
    #遍历图片
    for imagePath in imagePaths:
        #open是打开函数，也就是将图像文件存放到PIL_img中去，convert表示对图像进行的处理方式，有0（黑白），L(灰度处理),RGB,RGBA……
        PIL_img=Image.open(imagePath).convert('L')
        #将灰度化处理的图片进行向量化，所有灰度值用uint8存放起来
        img_numpy=np.array(PIL_img,'uint8')
        faces=face_detector.detectMultiScale(img_numpy)
        #split函数的作用是将imagePath提取出来以后，先将imagePath分隔开，只取到子文件的名称，split('.')[0]则是将文件名称按照.分开，
        id = int(os.path.split(imagePath)[1].split('.')[0])
        #整理提取出的数据
        for x,y,w,h in faces:
            ids.append(id)
            facesSamples.append(img_numpy[y:y+h,x:x+w])
        print('id:',ids)
        print('faceSample:',facesSamples)
    return facesSamples,ids
```

`@Path->faces,ids` 这个函数就是一个将对应路径文件下所有的图片全部都识别出特征值并将id提取的函数

`os.path.join(path,name)` 效果就是得出path\name的一个地址的衔接效果，更好的表示地址，os.listdir(path)就是将path目录下的所有的文件名称形成一个list

分类器就不用赘述了，实现对人脸的检测

接下来对所有的face进行遍历，对每一个face都先进行灰度化处理，将图片转为向量，然后将向量传输到numpy矩阵中去

这折后就是将id提取出来,id就是文件名称而已.

最后将所有的id和facesample提取出来汇总，这样就得到了每一个图片的id和特征矩阵

训练：就是加载识别器，将我们指定的脸的特征值通过一定的方式训练后存放到一个文件中去

```
#3.加载识别器
    recognizer=cv.face.LBPHFaceRecognizer_create()
    recognizer.train(faces,np.array(ids))
```

这里有一个小插曲,cv.face这个对象必须是要下载一个特定的包才能使用，否则无法使用的
参考链接：解决cv.face无法使用的问题

重大挫折！

在使用识别器加载过程中debug是十分艰辛的，尤其是在前面下载完了看例程，我还以为可以一模一样的写就完了，结果发现opencv的4.8版本过高，之前的 `cv.face.LBPHFaceRecognizer_create()` 函数根本无法使用，经过了漫长探索才发现，原来是初始化这个class的变量方法变了，更改以后才可以用

一个参考文章，对我很有启发，那就是查官方手册

文章源地址怎么解决LBPHFaceRecognizer_create()不存在的问题

```
代码如下：
#3.加载识别器
    recognizer= cv.face.LBPHFaceRecognizer.create()
    recognizer.train(faces,np.array(ids))
```

竟然是类对象里的方法！捏麻麻地

存放 :将训练的结果通过yml格式的文件存放下来，yml简而言之就是一种用于存放数据和标签对应的文件

本来是希望能够写一个直接新建文件夹的过程，但是发现直接这么写要出bug，所以就采用了直接write文件的方法，至于生成train文件夹的方法，我想先留白，以后再说咯

主要是看别人也开摆，我也想摆了(乐) 存放问题

7.做点有意思的应用！（比如智能门禁）

本来不知道怎么做的，但是突然想起来以前还做过一个python自动发送邮件的项目，感觉可以联系起来

思路大概是这样的:

核心:人脸识别
工作方式:
1.先将所有允许的人员录入到名单文件夹中
2.红外线传感器，检测外部是否有来往人员
3.打开摄像头：人脸检测
合法人脸：打开舵机开门
非法人脸：继续检测（并统计检测次数），超过最大次数判定为非法人员，亮LED_RED，禁止开门。
4.陌生人报警模式：如果红外检测长时间检测到有异物存在，打开摄像头，并进行摄像头截图，将截图内容保存，通过短信方式发送给guard，可以增加

存在几个难题:

1.对警报程序的编写

2.人脸识别的编写

3.自动截图程序的编写

4.上位机和单片机之间通信的问题

5.舵机的操作

6.红外传感器的检测

7.摄像头的操作

8.控制LED屏幕显示警告

对于不知道怎么通信的过程而言，我现在的解决办法是

1.通过python来写一下串口通信

2.单片机的烧录参考野火的例程

比如串口通信过程

舵机操作就参考江科大的例程

红外传感器的话还没想好，先搁置

LED还没学，学了再说

优先级:

先实现人脸识别的软件过程

再实现上位机和单片机的互相通信（基于python）

再实现对舵机的控制

再实现对红外传感器的控制

可以加一个蜂鸣器或者喇叭控制

再实现对LED的操作

接下来的过程就交给开发文档吧!

长远计划栏

1.小型化+模块化，实现不需要上位机也可以实现的系统?

2.如果不用python呢?

3.语音啥的能不能安排上

现在看来还有很多要学习的地方捏

这个开发文档用来记录开发过程中遇到的一些问
题，参考的一些文献，思路的点拨balabala

任务1：实现人脸识别的软件过程

这里我用到了下recognizer的read函数，这样可以用来读取一下我们早早训练好的白名单人员的面部特征

首先借鉴一下之前写好的人脸检测的模块（copy一下）

但是需要进行一些改编，因为初始版本仅仅完成了 识别面部->框一下 的过程,需要加上其他的一些特点，比
如 `predict` ，将识别出来的面部用识别器(`recognizer`)对比一下，得出一个误差值(`confidence`)
误差只要比我们规定的一个值小，我们就认为识别的人脸符合白名单要求

此外，我们还要在原图片中将识别出来的人脸标记上所对应的白名单上的名称，如果识别出来大于误差值，不再白名单上，我们就打一个unknown，表示识别出的人脸不符合,这里就用到了特别的函数

`cv2.putText（）`，效果是将特定的文字标注到图片上

使用范例在这篇博客里:cv2.putText使用方法

遇到点问题，重复识别咋整，也就是无法精准识别

所以必须要有一个识别后的标准来确定一个统一的人脸图像

20：51

发现不能用return的方式来实现识别程序的终止，准备后面用usart的方式从单片机终止识别

20：53
我去，对啊，应该做成一张一张的人脸识别，这样就不用担心多个人同时识别的问题了！

总之今天先解决了识别的问题了

解决一下通信的问题：

我现在的思路是先让单片机一直向上位机发消息，直到回应为止

就先让单片机一直发"ask"，直到上位机识别吧

新知识：可以 `pip install pyserial` 这个包，从而可以在python上对stm32的外设进行一些操作

操作教学：如何用python进行串口通信

果然不会一帆风顺，在用python进行串口通信的情况下，出现了串口无法打开的情况，我们称之为 `BUG1`

报错如下: `could not open port 'COM4': PermissionError(13, '拒绝访问。', None, 5)`

查阅资料后发现，可能是占用导致的问题，在运行python的同时我还打开了其他的串口助手，从而导致了无法正常打开串口

python串口调试的参考资料

关闭了野火的串口调试助手后，发现python可以正常的打开串口了

难点：如何用python实现对单片机的收发信息的控制(啊啊啊啊啊烦的一)

现在的问题是，单片机和上位机各自管各自的，根本发不到一块儿去，我测

稍微调试一下，发现好像可以操作一下，我把最初版本的那个 `send string` 给改成了 `Send byte` 首先解决了在通信过程中,*二进制通信码*和*我本来要读取的那个字符格式*因为传输原因不匹配的情况.

接下来就是让python发出一个启动信号，让单片机的LED_G点亮!

当前的问题是，python中接收的数据都是二进制的源码，需要进行一定的进制更换，比如改成十进制或者十六进制进行操作

为什么说必须进行**进制更换**，因为试了好几次，把KEIL单片机的发送的源码改了又改，发现统一都是二进制传输的十六进制的数字文件，捏麻麻地

总之我把所有python传输和接收的数据统一为了同一种模式 `/x01` ，还点了个灯，这样就顺利解决了启动的问题！~~原神，启动!~~

当然还是有发送信息不畅，有报错的情况出现，这个还得好好观察下才行

发现问题，如果传输的是一个string的话该如何通信呢，是个好问题

先不管了，进入下一个课题

将python控制通信与人脸识别联系起来

先训练一个我自己的人脸的特征，用到之前的那个人脸特征提取模块昂

ok，识别大概做好了，现在先让软件可以根据识别的结果来点亮灯!

做成这样，只要是识别成功了，就先熄灯，然后亮起绿灯。失败就亮红灯

碰到了问题：无法连续接收信息

问询chatgpt我的代码编写中的问题，发现是由于未清除之前的缓冲区的信息，又马上接收下一波信息，导致无法正常接收。

这是修改的代码，每一次receive前都清空缓冲区

```
USART_ClearFlag(DEBUG_USARTx, USART_FLAG_RXNE); // 清除接收缓冲区状态位
respon = USART_ReceiveData(DEBUG_USARTx); // 接收新指令
```

结果改了还是无法解决这个问题

捏马的
我怀疑是指令区一直被占用，无法write了
重新弄一遍试试看呢

重新配置一遍中断函数咋也不好使
!!!!!!!!!

我测，原来是python的b `\x06` 不是0x06，十六进制打错来，我测

为了方便测试，必须得弄一个让单片机识别完成以后自动初始化的过程才行

OK

大功告成
今天将人脸识别和点灯对应起来，对的人点绿灯，错的人点红灯，空闲时间点白灯，等待工作点黄灯，初始态为蓝灯

通过一些简单的延时，将识别过程和点灯结合起来力！
今天就先到这里吧，明天调试舵机，把舵机和人脸识别联系起来

今天进行一个PWM输出的学习捏

先进行简单的呼吸灯调教

分解下需要用到的知识：TIM的简单运用，PWM波的输出运用

找找教程先:

基础定时器:TIM6,TIM7(这里就贴一个教程链接就行)基本TIM教程

通用定时器：TIMx

高级定时器：TIM1,TIM8

CK_PSC:提供的时钟频率

PSC:预分频数

ARR:重载周期数

需要牢记的公式！！！！！！

PWM频率公式： `Freq=CK_PSC/(PSC+1)/(ARR+1)`
PWM频率实际上就是电平从有效电平到无效电平再到有效电平的切换频率

占空比公式： `Duty=CCR/(ARR+1)` ,CCR就是有效电平的百分比

分辨率公式： `Reso=1/(ARR+1)`

`setCompare` 是一个十分重要的函数，可以帮助我们改变CCR的值，这样就可以实现呼吸灯的效果了

详情就请参见写的bsp_pwm_led文档，现在总算学（抄）完了，但是不知怎么来把野火开发板的5v开关打开，气死我了
怎么打开5V开关

TNND还是不行，我服了

无奈记录下，我用ADC测试了一下VCC的那一个GPIO端口，捏麻麻地，网上说有FT管的接开漏输出，引脚就能输出5V，可是试了下结果反而变成0.3v，我服了，用推挽输出能正常3V3

心累，不知道咋解决了

我测，ADC输入的引脚电压不是FT管的，还好还好，我测，差点把ADC烧了，好险

还是没找到好的解决办法呢

会不会是pwm输出不对？没有机械转的声音，我估计电源和pwm输出都有可能有问题，我先试试做一个呼吸灯效果。

发现问题：PWM输出失败，看来得先把呼吸灯效果实现才行

哇擦，把GPIO——Init放后面了就可以正常呼吸灯，好神奇，什么原理

通过求助好哥们，好哥们说开发板的5V口子一般都是正常的，上电就能用，我明天试试看

发现问题：原来是我初始化OC口错了，应该是用OC3_Init函数

？为毛本来可以动的舵机，动了一会儿又动不了了，不会是TIM烧坏了吧我超

原来是给的参数超出范围，就停转了，改回去就好了

话说为毛我的电机一直转啊

原来是我买的是360度电机，CCR=500的时候反向转速最大，1500转速为0，2500正向转速最大，关于舵机停不下来的问题

接下来的思路：买个红外传感模块，初步感应有人到位，然后再买一个蜂鸣器（？），用来表示识别成功与否

要不要买个小垃圾桶来试试手呢?

先买两个舵机试试

总算到货了，在此之前我先把LCD给开动了来

开动LCD

NMD，怎么这么xx复杂，受不了一点

经过了两天的折磨，大概能把时序图啥的弄懂了，目前先做到能够正常使用它吧

现在只做到了调用函数实现文字，但是接下来的目标是能够在LCD上做到操作像素点和显示图像，还是任重道远的

总之先记录一下这个idea，以后慢慢学，现在先学会最基本的使用方法

将LCD信息和识别成功与否联系起来，可以方便辨识（待办）

用舵机把垃圾盖弄起来！

NMD想了半天还是直接顶起来最省事

可惜没有双面胶，明天去买点

ok,买来了，今天上午的任务就是把两个垃圾桶给驱动一下开盖程序

重新设置一个TIM的PWM输出口

查了一下DATAsheet，我们就采用TIM4_CH3，也就是PB8,来接入绿色垃圾桶

为了方便起见，把Delay函数解耦出来了，方便其他的模块进行调用

在进行识别的时候出现了重复识别，开盖时间与识别时间错开导致的时间混乱，所以需要调整一下识别结果传递的时间

在将LCD的各个函数与我要用到的功能函数模块化的时候碰到了一些奇怪的BUG，比如有结构体明明在头文件定义了，却还是显示未定义

查询了网上的资料，看来多半是重复定义了头文件

把LCD的几个头文件单列出来，防止重复引用，然后就可正常编译了

正式调整一下成果

在把LCD加入以后竟然出现了HardFault,无法在打开LCD后接着打开LED了

是通讯的超时判定导致了无法正常使用，删除即可

下一个问题

在复位一次后竟然无法正常运转了

应该是时许的问题，有延迟，delay有些乱用了

列出主要的毛病：

1.LCD刷新和接收上位机信号对不上

2.LED亮灯时间有点晚，调快点

深入一下，就是说，必须得在收发信号处做一个中断才行

在python上实现一个不断重置的FLAG位，确保是在识别后给出的结果而不是上一次识别的结果。适用于连续的识别

真是成也中断败也中断，中断导致我不能把接收的信号人为设置为RESET

从上位机入手呢？也不对啊

那为什么没有如预期一样，开关一次盖子以后停下呢

MD，怎么整，我去思考一下对中断的理解吧

！！果然是中断理解的问题，修改了对中断的设定以后，问题顺利解决了！

分享一下对中断的理解的文章：

中断的理解

怎么用GIT啊啊啊啊啊啊啊啊啊啊啊

没办法，只有硬学了
先把一些文件推送一遍试试看，建立了自己的第一个代码仓库这里！
我测，感觉没弄懂，有点难办

目前学习顺序我想可以这样

新建一个代码仓库->学一下git的推送和提交的模板->学一下git的远程操作

现在会提交了，所以先不管了，起码交作业莫得问题了（嘻嘻）

开始学习真正的识别系统，作为垃圾分类使用

找到一个教程利用opencv进行训练

接下来就交给下一个开发文档吧！

更新一下

重新把文档整理了下，提交到了idoor项目里去
地址在这儿idoor_1.0

怎么没有云分支，我测什么情况

创建了个新的，现在试试看呢？

这是用来学习物体检测的过程教学昂

1.原理学习

参考的文献1：检测原理

不过这里是基于R-CNN网络进行的识别，通过其他的一些教程了解到，还可以使用YOLO5模型来进行物体的识别

YOLO的介绍
先了解了原理再进行复刻，学习的效果会好很多

复刻的源地址：
yolo5识别,环境配置aconda,项目环境配置，aconda安装路线更改

鉴定为看不懂一点，不过大概有点印象了，Ok开始操作

话不多说，开始copy

第一步，配置环境

这里记录一下conda启动相关的配置路径
conda create --prefix=D:\MINI_conda_storage\envs\yolo5 python==3.8.5

更新一下

发现这个yolo5版本有点落后，于是把之前安装的包都删除了，现在安装anconda后学习yolo8

打开官方YOLOv8的网址，查看安装和使用的方法
YOLOv8官网

安装了ANCONDA，取代之前的MINICONDA，首先创建一个python3.8以上的虚拟环境，我们把它取名为YOLO8

牛魔的，竟然必须用-p命令创建，不然都给哥们安到C盘了

记录一下地址:D:\ANCONDA\envs\YOLOv8

命令如下 `conda create -p D:\ANCONDA_ENVS\YOLOv8 python=3.8`

竟然不能重命名一个简短点的代称，我测

启动代码 `conda activate D:\ANCONDA_ENVS\YOLOv8`

关闭代码 `conda deactivate D:\ANCONDA_ENVS\YOLOv8`

接下来先检查一下显卡的安装情况

`Nvidia-smi` 查看安装的N卡的CUDA版本，我的电脑是11.7的版本，可以安装pytorch的

打开pytorch官网，查找一下cuda11.7应该怎么安装
pytorch官网版本查询

记录下载的指令，在conda的命令台下载

```
conda install pytorch==2.0.0 torchvision==0.15.0 torchaudio==2.0.0 pytorch-cuda=11.7 -c pytorch -c nvidia
```

下载pytorch2.0

安装不起，国内没有镜像源，先试试安装其他版本吧

```
conda install pytorch==1.13.0 torchvision==0.14.0 torchaudio==0.13.0 pytorch-cuda=11.7 -c pytorch -c nvidia
```

这个是pytorch1.13版本

还是不行，我决定删掉清华源，使用官方源

在下载完了pytorch后,我用conda下载了yolo8的模型，现在可以开始学习了（应该）

用pycharm打开了yolo的文件夹，首先配置好解释器，我最开始配置解释器的时候，conda找不到可执行文件，这里解决方法po出来了
关于conda可执行文件找不到的解决办法

查询了一下目录，我把源码和虚拟环境放到一起了我测，很烦
我得用pycharm把打开一下才行

NMD，安装的源码都不知道安装到哪里去了，重装一遍
conda包的卸载

NMD,C盘爆满，清一下内存，我测，清不了一点，G，以后安个固态

查找一下anconda位置查找conda包位置

现在先去github把源码下载一下吧

采用git clone的方式，打开源码的话就用pycharm的conda环境即可

推荐一个好中好的笔记为什么要用conda

在检查好了requirements后，我顿悟了配置环境的意思，配置的其实就是一个安装、运行代码的环境，所以我采用git clone的方式之后，再用conda来安装和核对requirements，最后显示安装成了
我测，太艰难了

妈的，C盘爆内存了，删库，明天重新安装一遍