**电子科技大学成都学院**

**实验报告**

课程名称 数字图像处理

分 院 计算机学院

年 级 2020

专　　业 人工智能

班 级 人工智能实验班

任课教师 陈东祥

学生姓名 徐何方圆 刘嘉琳 郭松铭

陈显权 蔡仁杰 吕逢春

学　　号 2041340113 2041340204 2041340219 2041340206 2041340217 2041340221

2021年12月制

摘 要

由于现代人们生活的繁忙，许多人忽视了睡眠，而导致在公众场合犯困。我们想要设计一种系统用来检测人们是否犯困，以此来减少此类事件带来的社会损失。该系统的工作可分为两个部分：检测或定位面部；预测检测到的面部中主要区域的地标。基于dlib人脸识别68特征点检测、分别获取左右眼面部标志的索引，通过opencv对视频流进行灰度化处理，检测出眼睛地标和嘴部地标来确定人的眼睛长宽比（EAR）和嘴部长宽比（MAR），以检查人是否困倦。

关键词：犯困检测，Dlib模型 ，Opencv

ABSTRACT

Due to the busy life of modern people, many people ignore sleep, resulting in sleepiness in public. We want to design a system to detect whether people are sleepy, so as to reduce the social loss caused by such events. The work of the system can be divided into two parts: detecting or locating the face; the landmark of the main area in the detected face is predicted. Based on Dlib face recognition 68 feature point detection, the indexes of left and right eye facial signs are obtained respectively. The video stream is played through OpenCV to detect eye landmarks and mouth landmarks to determine people's eye aspect ratio (ear) and mouth aspect ratio (MAR) to check whether people are sleepy.

**Key Words:**Drowsiness detection,Dilb,Opencv

目 录

[第1章 引言 1](#_Toc72071965)

[1.1 选题背景 1](#_Toc72071966)

[1.2 研究目标和意义 1](#_Toc72071967)

[1.3 研究思路 1](#_Toc72071968)

[第2章 犯困](#_Toc72071971)问题现状 2

[2.1现状概括 2](#_Toc72071970)

[第3章 犯困现象原因分析 3](#_Toc72071973)

[3.1原因分析 3](#_Toc72071970)

[第4章 研究的理论和基础 4](#_Toc72071969)

[4.1 检测犯困的评判标准 4](#_Toc72071970)

[4.2 犯困检测点一（眨眼） 5](#_Toc72071970)

[4.2.1 视觉疲劳检测原理 5](#_Toc72071970)

[4.2.2 主要代码思路 7](#_Toc72071970)

[4.3 犯困检测点二（打哈欠） 8](#_Toc72071970)

[4.3.1 打哈欠疲劳检测原理 8](#_Toc72071970)

[4.3.2 主要代码思路 11](#_Toc72071970)

[4.4 犯困检测点三（瞌睡点头） 12](#_Toc72071970)

[4.4.1 Head Pose Estimation 12](#_Toc72071970)

[4.4.2 主要代码思路 14](#_Toc72071970)

4.5 功能封装 16

4.5.1 软件架构 16

4.5.2 标准参数说明 16

4.5.3 UI界面设计 16

4.6 功能演示 18

[第5章 结论 19](#_Toc72071969)

[参考文献 20](#_Toc72071983)

[致谢 21](#_Toc72071984)

1. 引言
   1. 选题背景

日常生活中犯困的现象无处不在：学生上课的时候会打瞌睡；上班族上班的时候会犯困；驾驶员开车的时候会打瞌睡；层出不穷的犯困问题往小了说，只会影响人们的日常的生活状态，如学生犯困影响老师上课时的课堂纪律；上班打瞌睡影响工作的效率；但有些时候，无意的犯困也可能造成一些严重的后果，如一些货车司机犯困，疲劳驾驶导致的严重交通事故。根据国家公路交通安全管理局的数据，每年均涉及疲劳驾驶事故中导致超过1,550人死亡和71,000人受伤，但是实际数字可能要高得多。因此，为了避免这类事故的发生，我们制作了这个系统。它通过检查人的眼睛是否闭合或正在打哈欠来预测眼睛和嘴巴的标志，从而确定一个人是否正处于疲劳驾驶。

* 1. 研究目标和意义

对生活中人们犯困的情况进行实时检测。解决人们在日常生活中因为犯困的缘故而造成不必要的社会损失。有助于维护社会安全。

* 1. 研究思路

该系统的工作可分为两个部分：

1. 检测或定位面部
2. 预测检测到的面部中重要区域的地标。

一旦预测出结果，我们仅使用眼睛地标和嘴部地标来确定人的眼睛长宽比（EAR）和嘴部长宽比（MAR），以检查人是否困倦。dlib库内部的预训练面部界标检测器用于估计映射到面部面部结构的68-（x，y）坐标的位置。这些68-（x，y）坐标表示脸部的重要区域，例如嘴巴，左眉，右眉，左眼，右眼，鼻子和下巴。

1. 犯困问题现状
   1. 现状概况

日常生活中犯困的现象无处不在：学生上课的时候会打瞌睡；上班族上班的时候会犯困；驾驶员开车的时候会打瞌睡；层出不穷的犯困问题往小了说，只会影响人们的日常的生活状态，如学生犯困影响老师上课时的课堂纪律；上班打瞌睡影响工作的效率；但有些时候，无意的犯困也可能造成一些严重的后果，如一些货车司机犯困，疲劳驾驶导致的严重交通事故。根据国家公路交通安全管理局的数据，每年均涉及疲劳驾驶事故中导致超过1,550人死亡和71,000人受伤，但是实际数字可能要高得多。因此，为了避免这类事故的发生，我们制作了这个系统。它通过检查人的眼睛是否闭合或正在打哈欠来预测眼睛和嘴巴的标志，从而确定一个人是否正处于疲劳驾驶。

1. 犯困现象原因分析
   1. 原因分析

1、睡眠不好。如果说白天容易犯困，那么首先想的就是晚上睡眠质量的不好或者是睡眠时间不足，才会导致第二天没有精神。

2、缺乏营养。当身体缺乏一些必要的营养元素的时候也会出现困乏的现象。因为足够的营养，才能够保证你的身体健康，如果是缺乏一些营养的话，那就会使得你身体无法有太多能量支持你的人体机能正常的运转就会出现疲倦的情况。

3、贫血。相对男性来说，女性是最容易出现贫血状况的，一旦女性出现贫血的状况不仅会对自己的容貌有着变影响，而且会对精神状态也会有着影响。导致困乏，疲倦等现象的发生。

4、肝病。对于白天没有精神，比较困乏的现象。有些医生会根据你的这些状况判断你是否有肝病的情况。比如说，如果你感到疲劳容易，犯困的时候小心就是肝病在作祟。因为在临床研究表明。肝病的患者往往就会出现精神不振的情况。

5、代谢疾病。如果你也经常感到疲惫困乏的话，可能小心你的身体代谢出了问题。如果你的代谢速度过慢的话，那就可能使神经功能的调节出现一些异常的状况。同时就会伴随着犯困，疲倦的发生。

6、血脂过高。如果血脂过高的话也会引起你没有精神。因为血脂一旦过高的话，那么就会使你的血液流动速度降低，那血液运氧的能力就会出现减弱。当到达脑部氧气减少的话，那么就会使人出现缺氧的情况，那么就可能让你出现困倦乏力，没有精神的情况了。

7、血糖过高。那么除了血脂过高意外，那么血糖过高也会引起这种情况的发生，因为你的血糖升高的话，就会影响全身的代谢系统。从而使人体出现疲惫，劳累，困乏的现象。

8、过度的疲劳，一直没有好好休息，那就会老打瞌睡和犯困。

1. 研究的理论和基础
   1. 检测犯困的评判标准

经查阅相关文献，疲劳在人体面部表情中表现出大致三个类型：打哈欠（嘴巴张大且相对较长时间保持这一状态）、眨眼（或眼睛微闭，此时眨眼次数增多，且眨眼速度变慢）、点头（瞌睡点头）。

本实验从人脸朝向、位置、瞳孔朝向、眼睛开合度、眨眼频率、瞳孔收缩率等数据入手，并通过这些数据，实时地计算出的注意力集中程度，分析是否疲劳驾驶和及时作出安全提示。

首先通过基于dlib人脸识别68特征点检测，通过opencv对视频流进行灰度化处理，检测出人眼的位置信息。

然后将困意的检测给定三种特征（眨眼、打哈欠、瞌睡点头）。

* 1. 犯困检测一（眨眼）
     1. 视觉疲劳检测原理

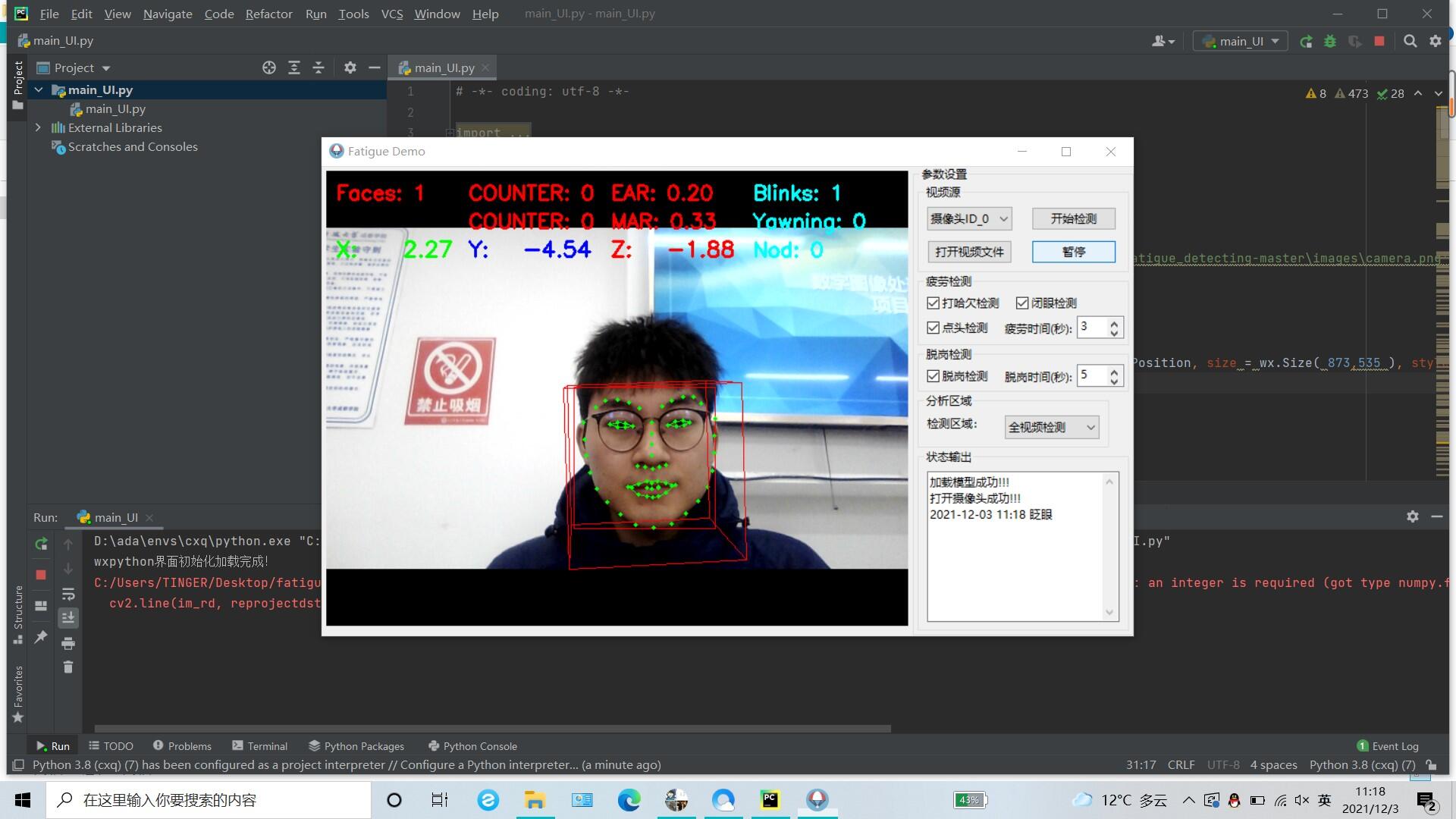
基于dlib人脸识别68特征点检测、分别获取左右眼面部标志的索引，通过opencv对视频流进行灰度化处理，检测出人眼的位置信息。

人脸特征点检测用到了dlib，dlib有两个关键函数：dlib.get\_frontal\_face\_detector()和dlib.shape\_predictor(predictor\_path)。

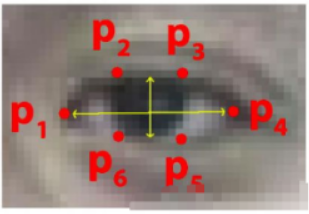
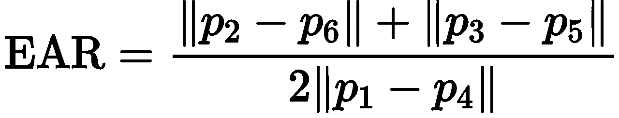
1.前者是内置的人脸检测算法，使用HOG pyramid，检测人脸区域的界限（bounds）。

2.后者是用来检测一个区域内的特征点，并输出这些特征点的坐标，它需要一个预先训练好的模型（通过文件路径的方法传入），才能正常工作。

使用开源模型shape\_predictor\_68\_face\_landmarks.dat，可以得到68个特征点位置的坐标，连起来后，可以有如图所示的效果（红色是HOG pyramid检测的结果，绿色是shape\_predictor的结果，仅把同一个器官的特征点连线）。

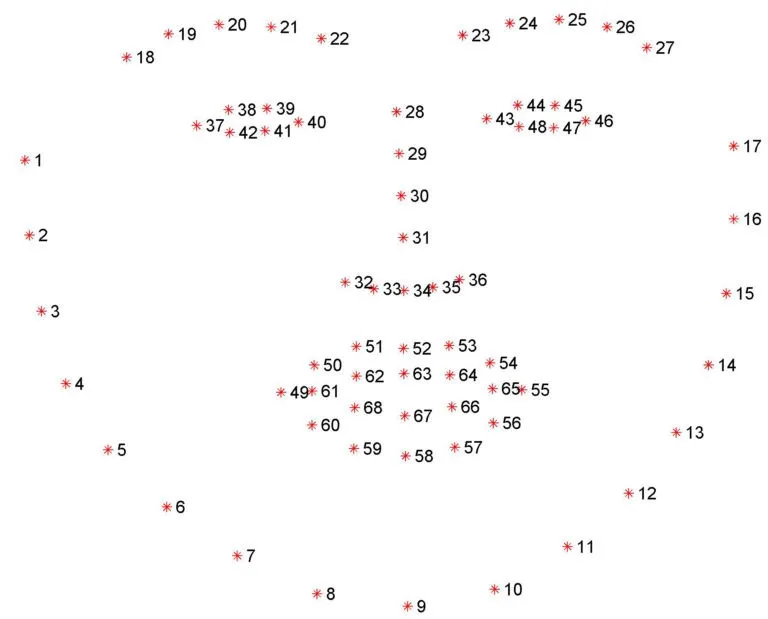
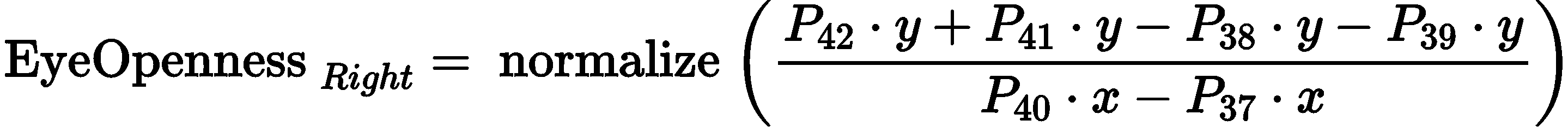
如何确定疲劳？

（1） 计算眼睛的宽高比

基本原理：计算 眼睛长宽比 Eye Aspect Ratio，EAR.当人眼睁开时，EAR在某个值上下波动，当人眼闭合时，EAR迅速下降，理论上会接近于零，当时人脸检测模型还没有这么精确。所以我们认为当EAR低于某个阈值时，眼睛处于闭合状态。为检测眨眼次数，需要设置同一次眨眼的连续帧数。眨眼速度比较快，一般1~3帧就完成了眨眼动作。两个阈值都要根据实际情况设置。

（2）当前帧两双眼睛宽高比与前一帧的差值的绝对值（EAR）大于0.2，则认为是疲劳

（68点landmark中可以看到37-42为左眼，43-48为右眼）

右眼开合度可以通过以下公式得到（左眼同理）：

通过计算38、39、42、41的纵坐标、37、40的横坐标来计算眼睛的睁开度。如：1/2\*[(y42+y41)-(y38+y39)]/(x40-x37)通过一个阈值确定眼睛是睁开还是闭上。也可以将这个值与初始的值的比值作为睁开度，根据不同程度来进行比较。睁开度从大到小为进入闭眼期，从小到大为进入睁眼期，计算 最长闭眼时间（可用帧数来代替）闭眼次数为进入闭眼、进入睁眼的次数 。通过设定单位时间内闭眼次数、闭眼时间的阈值判断人是否已经疲劳了。

* + 1. 主要代码思路

本次实验设计：计算对视频中每帧图片检测眼睛长/宽的值是否大于阈值，连续超过50次则认为已经“睡着”。（本次实验设置阈值为0.2，但每个人的眼睛大小比例不同，应该采取平均值计算方法——阈值的获取方式是：先采集30次数据，取其平均值作为默认的值。为了数据的准确，采集数据时应该平视摄像头。）

第一步：使用dlib.get\_frontal\_face\_detector() 获得脸部位置检测器

第二步：使用dlib.shape\_predictor获得脸部特征位置检测器

第三步：分别获取左右眼面部标志的索引

第四步：打开cv2 本地摄像头

第五步：从视频流进行循环，读取图片，并对图片做维度扩大，并进灰度化

第六步：使用detector(gray, 0) 进行脸部位置检测

第七步：循环脸部位置信息，使用predictor(gray, rect)获得脸部特征位置的信息

第八步：将脸部特征信息转换为数组array的格式

第九步：提取左眼和右眼坐标

第十步：构造函数计算左右眼的EAR值，使用平均值作为最终的EAR

第十一步：使用cv2.convexHull获得凸包位置，使用drawContours画出轮廓位置进行画图操作

第十二步：进行画图操作，用矩形框标注人脸

第十三步：分别计算左眼和右眼的评分求平均作为最终的评分，如果小于阈值，则加1，如果连续3次都小于阈值，则表示进行了一次眨眼活动

第十四步：进行画图操作，68个特征点标识

第十五步：进行画图操作，同时使用cv2.putText将眨眼次数进行显示

第十六步：统计总眨眼次数大于50次屏幕显示睡着。

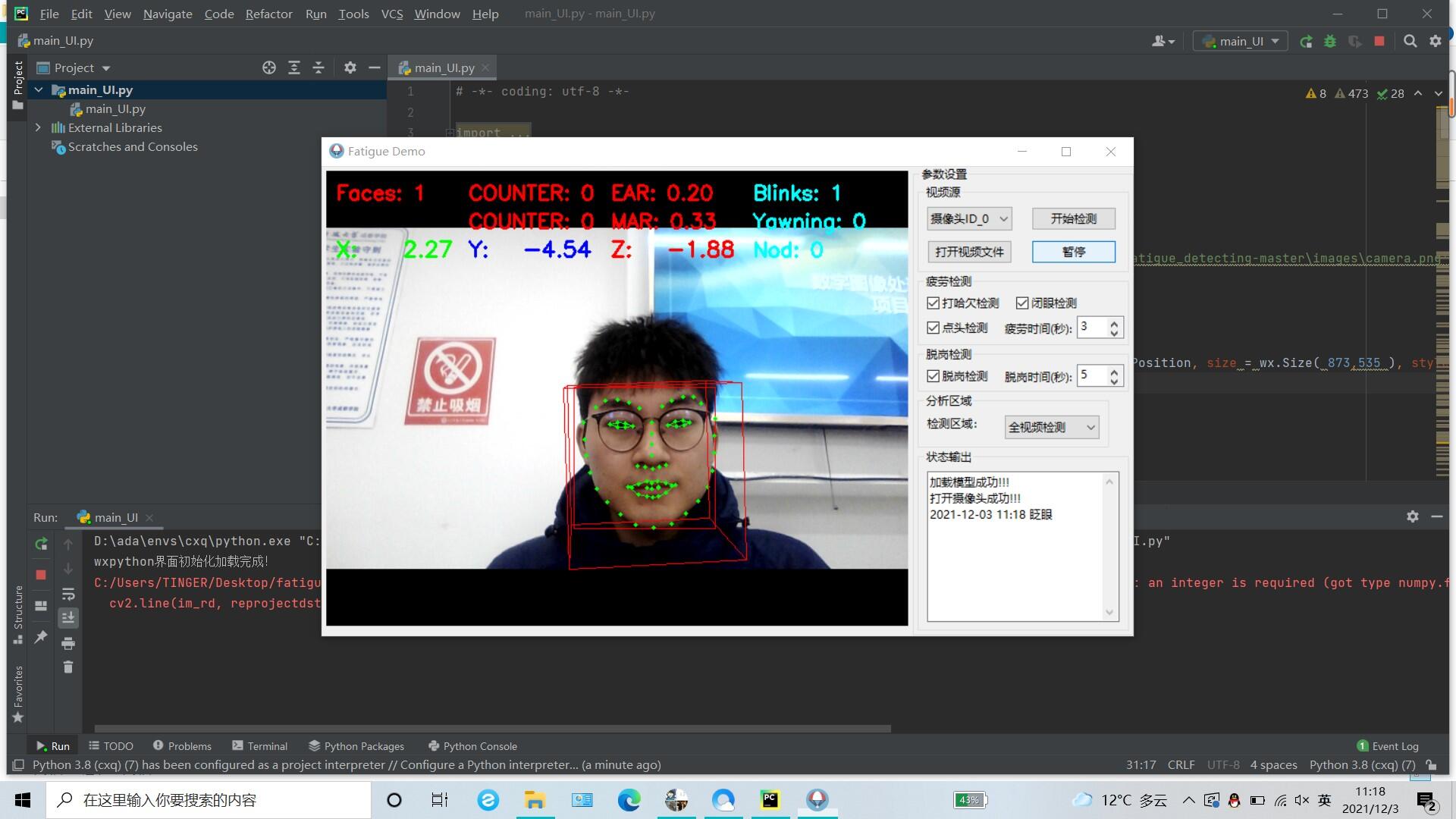
* 1. 犯困检测二（打哈欠）
     1. 打哈欠疲劳检测原理

基于dlib人脸识别68特征点检测、获取嘴部面部标志的索引，通过opencv对视频流进行灰度化处理，检测出人嘴的位置信息。

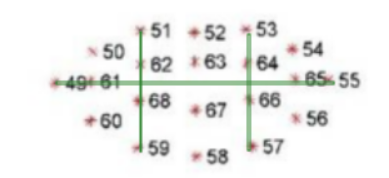
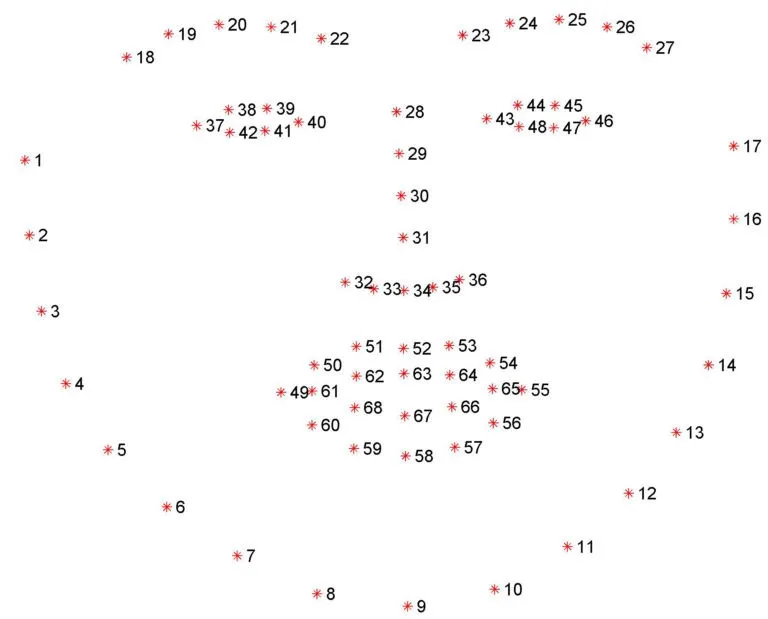
人脸特征点检测用到了dlib，dlib有两个关键函数：dlib.get\_frontal\_face\_detector()和dlib.shape\_predictor(predictor\_path)。

前者是内置的人脸检测算法，使用HOG pyramid，检测人脸区域的界限（bounds）。

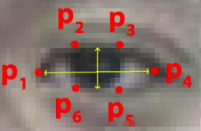
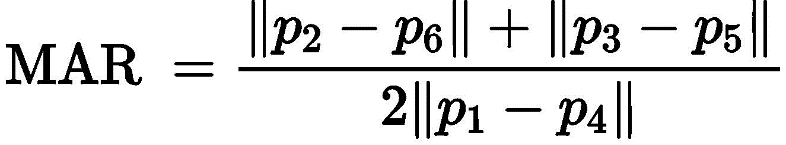
后者是用来检测一个区域内的特征点，并输出这些特征点的坐标，它需要一个预先训练好的模型（通过文件路径的方法传入），才能正常工作。

使用开源模型shape\_predictor\_68\_face\_landmarks.dat，可以得到68个特征点位置的坐标，连起来后，可以有如图所示的效果（红色是HOG pyramid检测的结果，绿色是shape\_predictor的结果，仅把同一个器官的特征点连线）。

**只要产生打哈欠的动作即归类为“疲劳”。**  
68脸部特征图lamark：

嘴部主要取六个参考点：

打哈欠可利用嘴巴处通过计算51、59、53、57、的纵坐标、49、55的横坐标来计算眼睛的睁开度。如：1/2\*[(y51+y53)-(y59+y57)]/(x55-x49)点的距离来判断是否张嘴及张嘴时间，从而确定人是否是在打哈欠，同时这个阈值应当合理，应经过大量实验，能够与正常说话或哼歌区分开来。

同眼睛相类似方法求嘴部欧式距离：

最终判定：

进行加权评分：基于眼睛和打哈欠的特征进行融合决策（打分，以模糊度表示）

* + 1. 主要代码思路

双阈值法哈欠检测——即对内轮廓进行检测：结合张口度与张口时间。  
Yawn为符合打哈欠的帧数，N为1min内总帧数，设阈值为10%，当Freq>10%时认为打了一个深度哈欠或者至少连续两个浅哈欠，此时给出疲劳提醒。

Step1：提取帧图像检测人脸，嘴部粗定位进行肤色分割；

Step2:嘴部精确定位，获取嘴部特征值K1，若k1大于阈值T1，则Step3,；否则K2=K1/2，count=0回到step1，检测下一帧。

Step3:提取嘴部内轮廓特征值K2，若K2大于阈值T2，则Step4，否则count=0,返回Step1，检测下一帧。

Step4:统计哈欠特征count=count+1,当count超过阈值且下一帧的哈欠特征消失，保存count到Yawn，Yawn(i)=count，count=0（count清0）回到Step1，否则的话也直接转回Step1。

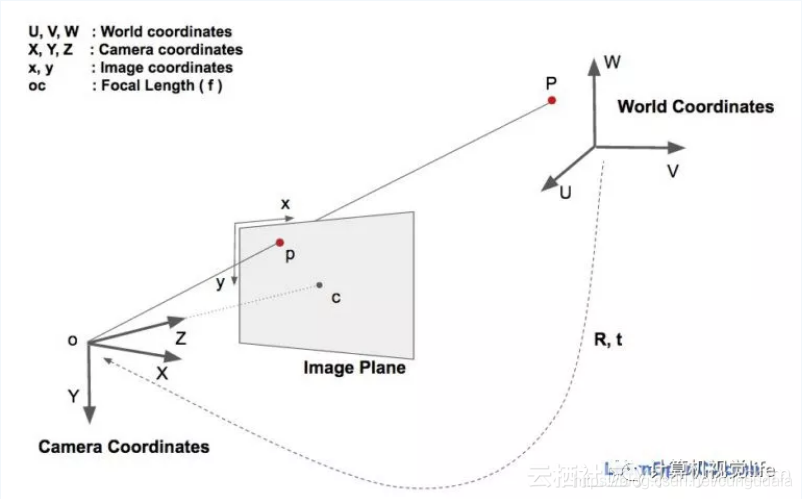
Step5:分析完1min内所有图像，计算哈欠特征总数，按照计算Freq值，超过阈值则发出疲劳提醒。（推荐阈值为0.1）

* 1. 犯困检测三（瞌睡点头）
     1. Head Pose Estimation

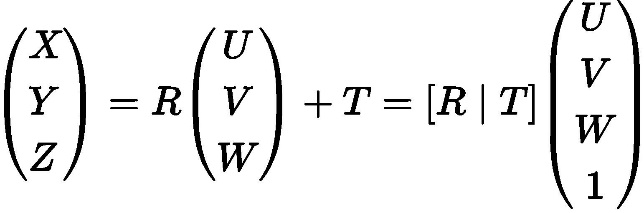
一种比较经典的 Head Pose Estimation 算法的步骤一般为：2D人脸关键点检测；3D人脸模型匹配；求解3D点和对应2D点的转换关系；根据旋转矩阵求解欧拉角。

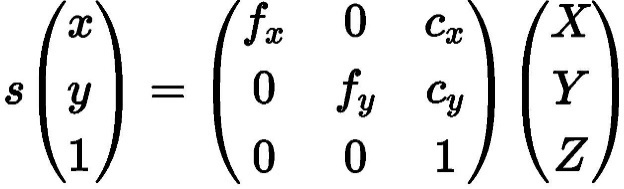
众所周知一个物体相对于相机的姿态可以使用旋转矩阵和平移矩阵来表示。

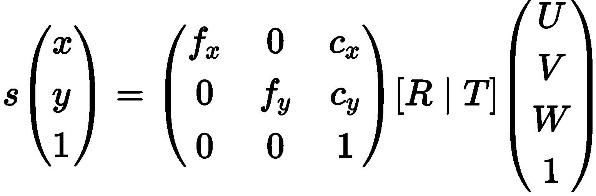
平移矩阵：物体相对于相机的空间位置关系矩阵，用T表示；

旋转矩阵：物体相对于相机的空间姿态关系矩阵，用R表示。

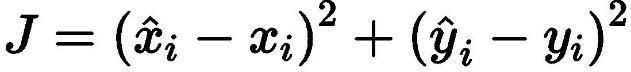
如此看来必然少不了坐标系转换。讲点人性，继续上图

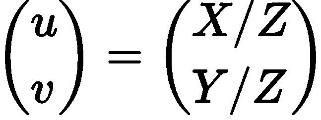
于是世界坐标系(UVW)、相机坐标系(XYZ)、图像中心坐标系(uv)和像素坐标系(xy)四兄弟闪亮登场。如果相机完美无瑕，老三可以回家洗洗睡觉，关系也相对简单。

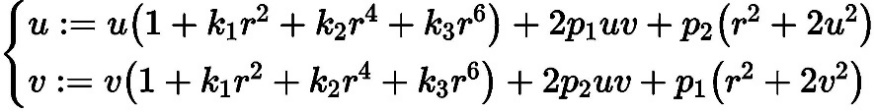
世界坐标系到相机坐标系：

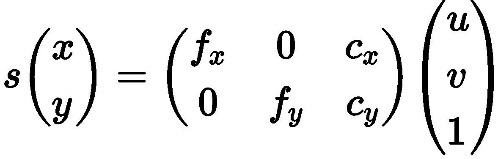
相机坐标系到像素坐标系：

因此像素坐标系和世界坐标系的关系如下：

上式的求解可用DLT(Direct Linear Transform)算法结合最小二乘进行迭代求解，最小二乘的目标函数可为

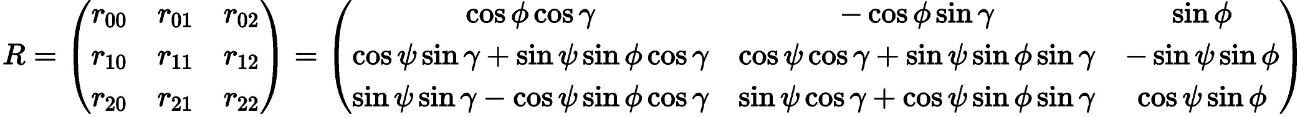
带^的变量为预测值，其余为测量值。  
可是相机也很无奈，她不完美，总有点瑕疵，比如径向和切向畸变，那关系就要稍微复杂一些，叫醒阿三继续推导：

相机坐标系要先转换到图像中心坐标系：

然后再被折磨一番(计算考虑畸变)：

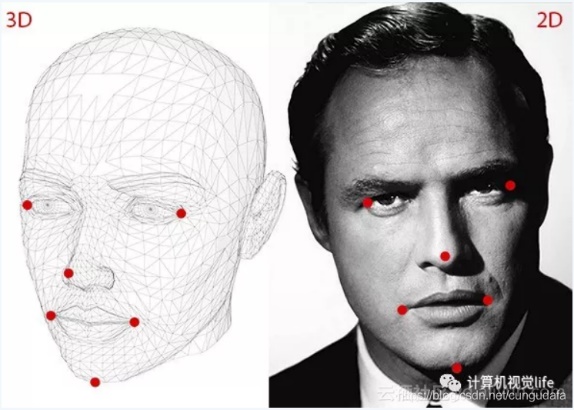
最后图像中心坐标系到像素坐标系:

看来只要知道世界坐标系内点的位置、像素坐标位置和相机参数就可以搞定旋转和平移矩阵，可上面的关系分明是非线性的，这可怎么解啊？其实OpenCV已经给我们提供了求解PnP问题的函数solvePnp()，一步轻松到位。

得到旋转矩阵后，就可以开心地去见欧拉角了：

世界坐标系中点的位置怎么得到呢？

总不能每时每刻都要测一下人脸各个点在空间的位置

从各种论文中发现，原来大牛们在算法里面内置了一个3D人脸模型，把关键点的空间位置都标出来，就充当真实脸的空间位置；可是大牛又觉得这样不太合理，一个3D人脸模型不能表示所有人的脸，对所有人采用一个模型得到的精度肯定不好，于是便有了3DMM(3D Morphable Model)，对不同人可以拟合出对应的3D脸模型，这样关键点的空间位置就比较准确了，Head Pose Estimation 的精度提上去了。

如何确定疲劳？

思路一：可利用姿态估计结果（如Pitch的读数）来判断是否点头及点头幅度

思路二：或用鼻尖处30号点的前后移动值（或是方差，方差表示一个单位时间数据的偏离程度，程度越大，则表示发生点头动作的概率越大、点头幅度越大）

* + 1. 主要代码思路

第一步：2D人脸关键点检测；

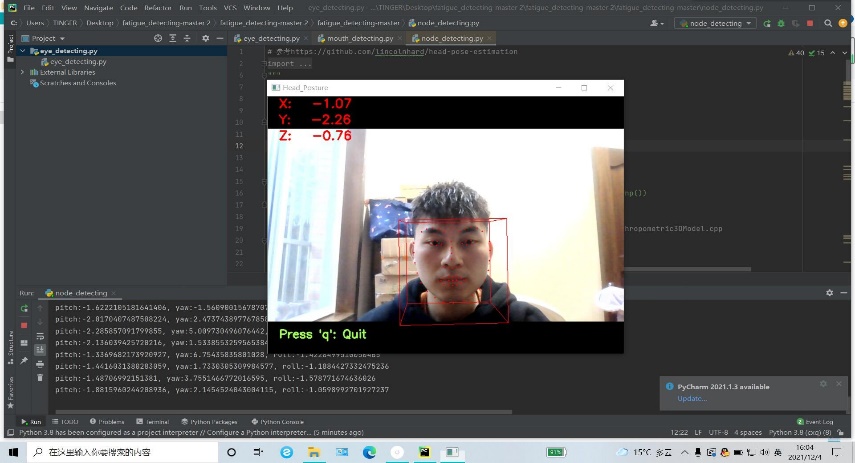
第二步：3D人脸模型匹配；

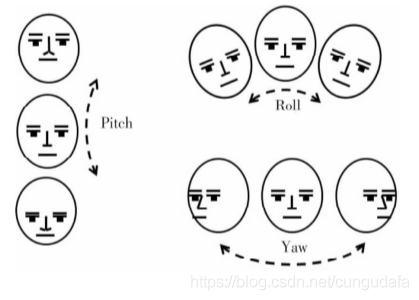
第三步：求解3D点和对应2D点的转换关系；

第四步：根据旋转矩阵求解欧拉角

头部姿态显示效果：

点头如何判断是否是瞌睡：

主要参数：欧拉角

头部姿态判断打瞌睡得到实时头部姿态的旋转角度过后,为头部旋转角度的3个参数Yaw,Pitch和Roll的示意图,驾驶员在打瞌睡时，显然头部会做类似于点头和倾斜的动作.而根据一般人的打瞌睡时表现出来的头部姿态，显然很少会在Yaw上有动作，而主要集中在Pitch和Roll的行为.设定参数阈值为0.3,在一个时间段内10 s内，当I PitchI≥20°或者|Rolll≥20°的时间比例超过0.3时,就认为驾驶员处于打瞌睡的状态,发出预警。

* 1. 功能封装
     1. 软件架构

环境：Win10、Python3.7、anaconda3、JupyterNotebook

技术：

Opencv：图像处理

Dlib：一个很经典的用于图像处理的开源库，shape\_predictor\_68\_face\_landmarks.dat是一个用于人脸68个关键点检测的dat模型库，使用这个模型库可以很方便地进行人脸检测，并进行简单的应用。

Numpy：基于Python的n维数值计算扩展。

Imutils ：一系列使得opencv 便利的功能，包括图像旋转、缩放、平移，骨架化、边缘检测、显示

matplotlib 图像（imutils.opencv2matplotlib(image）。

* + 1. 标准参数说明

疲劳认定标准：

- 眨眼：连续3帧内，眼睛长宽比为 0.2

- 打哈欠：连续3帧内，嘴部长宽比为 0.5

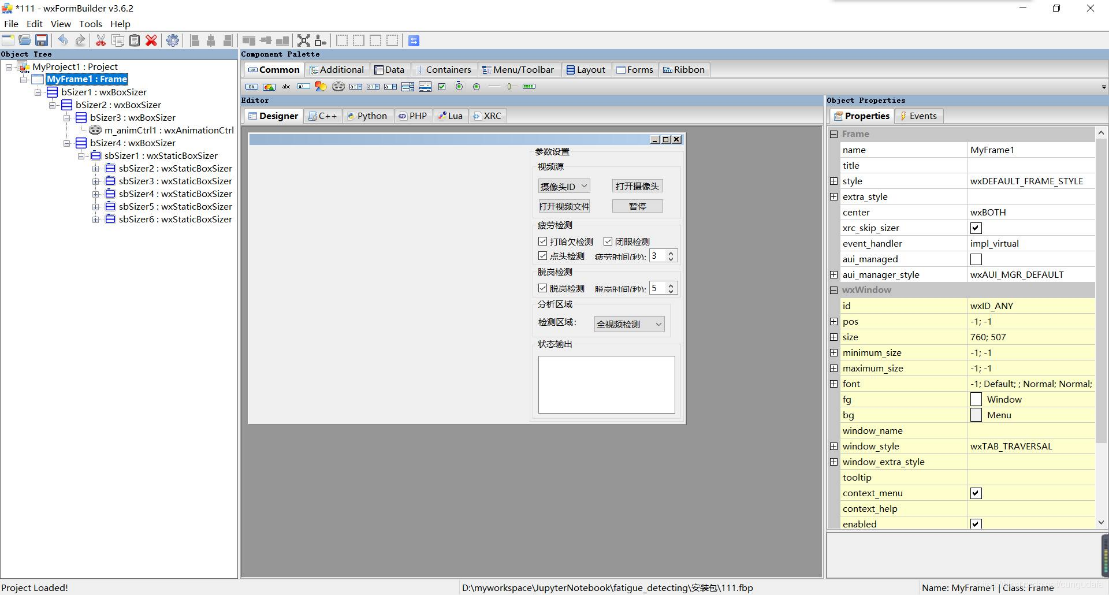
- 瞌睡点头：连续3帧内，pitch（x）旋转角为 0.3

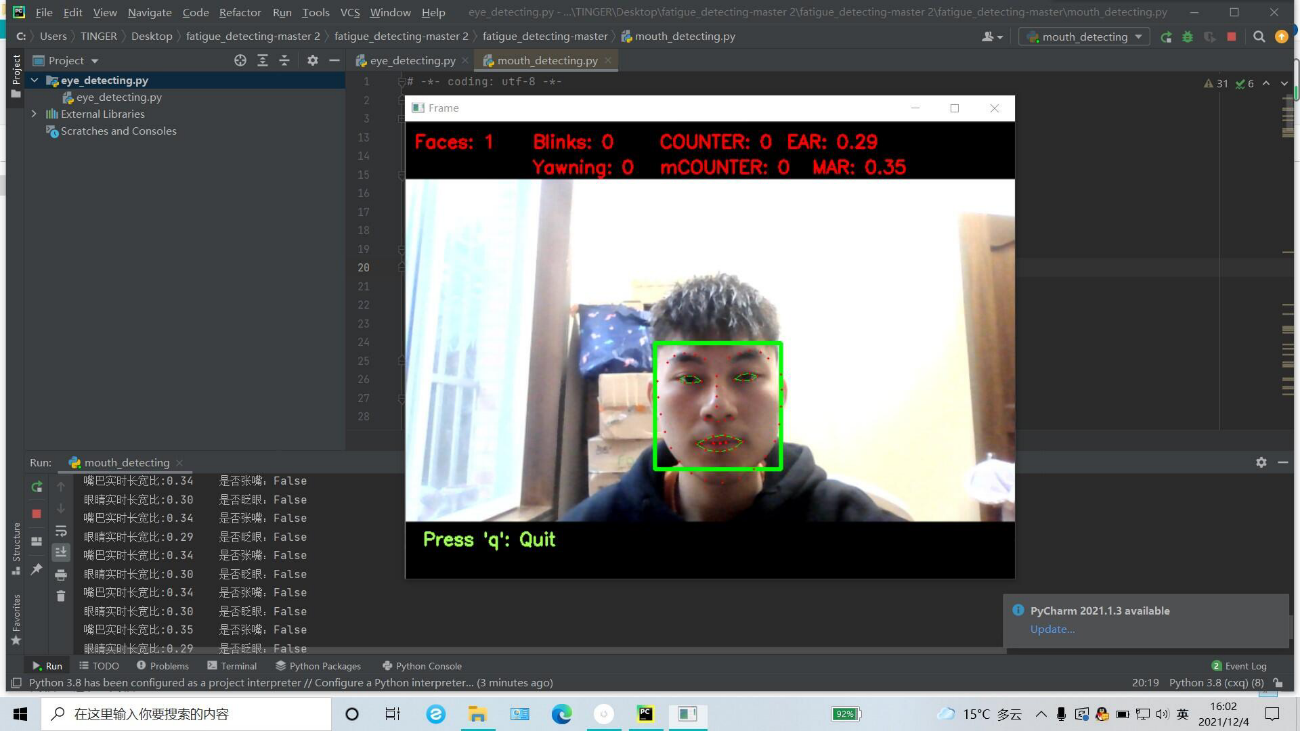
(真实运用中需要根据不同人的眼睛大小进行检测，人的眼睛大小，俯仰头习惯都不一样，这只是一个参考值)

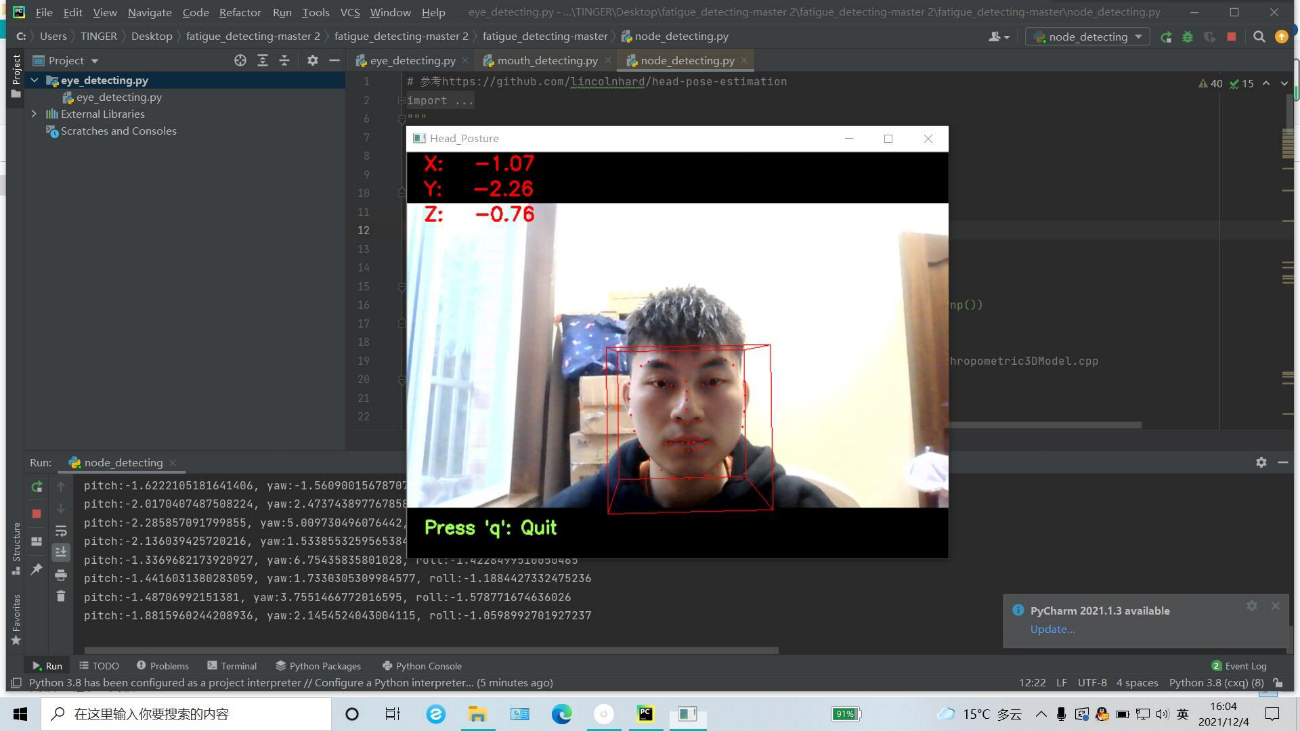
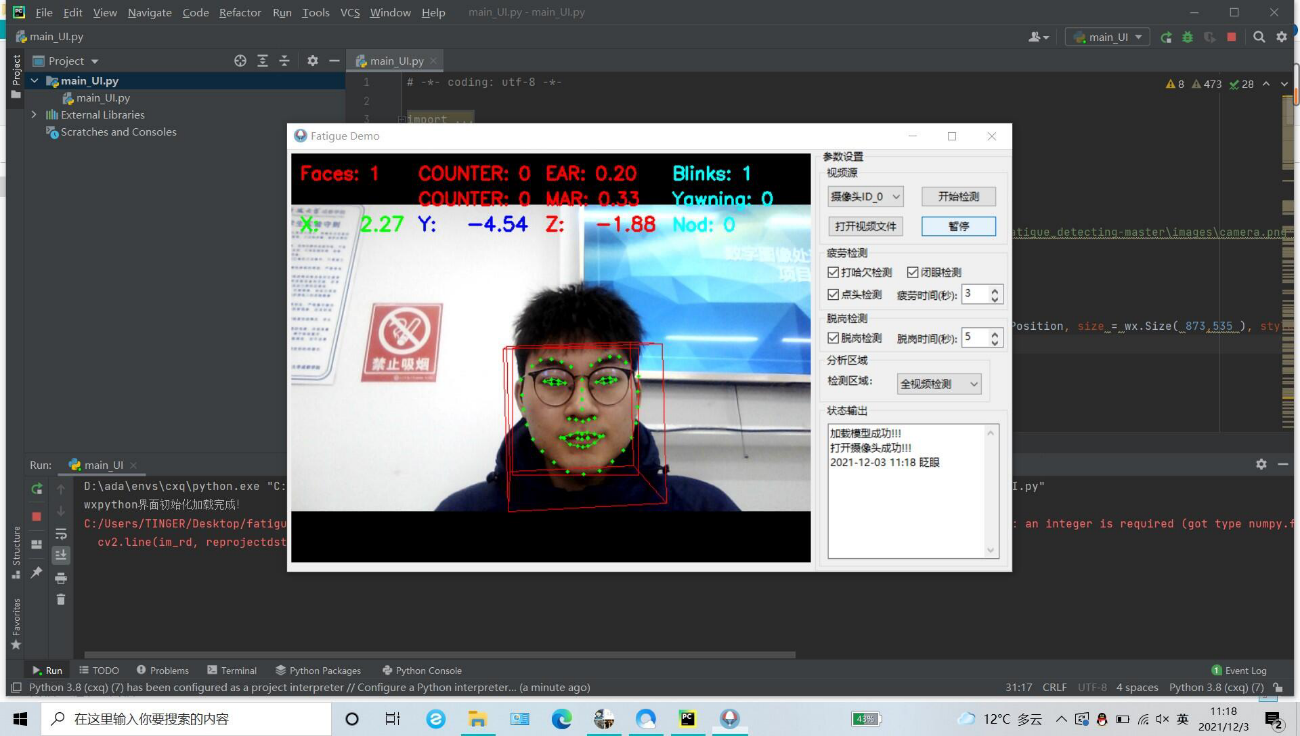
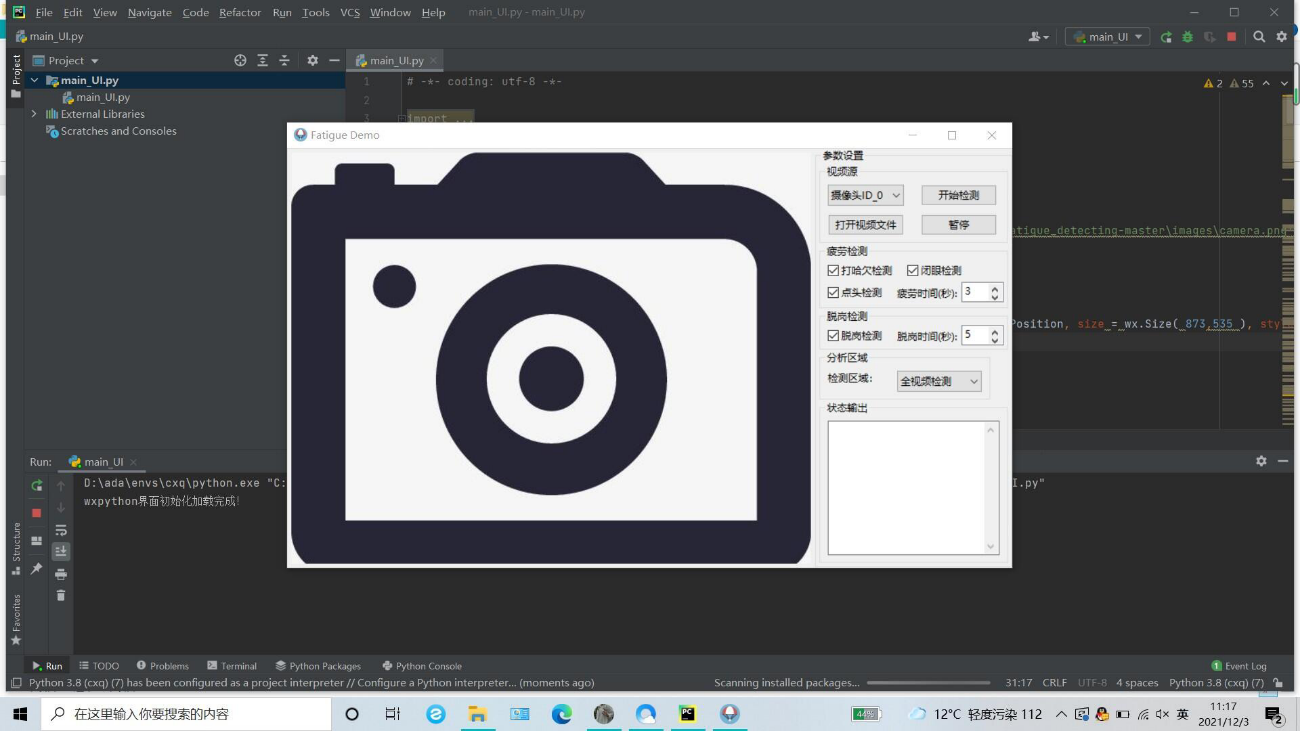
* + 1. UI界面设计

å¨è¿éæå¥å¾çæè¿°可视化界面工具下载：

1. 新建画布：先选择forms里面的Frame,开始的时候必须用这个，其他控件都是在里面。
2. 布局设置：在Layout里面选择一个布局。布局就是用来将Frame分成单独的小方格。各种布局都不一样，有的是只有行没有列，有的是行列都可以。根据需要添加即可。
3. 添加控件：在common里面选择基础的按钮等控件。在这里很难控制好位置，因为只有选定好了在哪个方格里，添加的时候才能添加。添加后的位置不好变化。所以，必须先设计好正规布局，然后再往里面添加小控件。
4. 右侧设置属性（Properties）：名称，相对位置等。
5. 点击事件：填写事件函数名。
6. 导出设计界面：选中MyProject2：Project，在属性（Properties）中代码规则（code\_generation）勾选上python。

综上，本次UI界面设计：

1. wxpython加载界面：（添加封面图和图标）
   1. 功能演示



1. 结论

对21世纪快节奏的人们而言，犯困成为生活中必不可少的一种现象。无论何种身份的人们，都会出现犯困的现象。针对这一现象的发生，我们设计了一个能自动检测人们犯困现象的系统。该系统涉及面较为广泛，不仅仅适用于对学生上课犯困现象的检测，对上班一族工作时犯困、对驾驶员驾车时犯困和对那些匆匆忙忙赶路人犯困等现象都可以进行有效的检测。针对学生而言，学习压力大、上课不感兴趣导致的上课犯困，老师上课不能照顾到每一位同学是否有犯困现象，因此使用该系统进行检测，可以发现学生在某些课程上犯困的数量或者上课犯困的时间段。例如某些课程学生犯困数达到了百分十百九十，可能说明该老师的教学方式不便于学生的理解和学习，老师能及时发现并修改教学方式，能从根源解决问题；又如一节课45分钟，学生可能到后20分钟左右开始犯困，老师可针对学生犯困的这一时间段，在后半节课开展一些趣味课堂，或者让学生自主练习等方式使学生不至于老师在授课时睡觉，提高一些教学质量。针对驾驶员而言，该系统可以有效检测驾驶员驾车时的状态，一旦驾驶员处于疲劳、瞌睡状态该系统就会对驾驶员做出提前预警，驾驶员因疲劳有可能在驾驶中突然进入梦乡，在此关键时刻该系统就会发出尖锐的报警声及时惊醒驾驶员避免交通事故的发生。尤其是那些拉货跑长途的大货车司机，很容易出现疲劳驾驶致使交通意外的发生，该系统能有效检测驾驶员驾车时犯困现象，即使报警等处理，杜绝意外的发生。

参考文献

[1]Drivers are falling asleep behind the wheels. Prevalence of drowsy driving crashes: https://www.nsc.org/road-safety/safety-topics/fatigued-driving

[2]Facial landmarks with dlib, OpenCV and Python: https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/

[3]Eye blink detection with OpenCV, Python, and dlib: https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/

[4]Drowsiness Detection with OpenCV: https://www.pyimagesearch.com/2017/05/08/drowsiness-detection-opencv/

[5]Real-Time Eye Blink Detection using Facial Landmarks: http://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf

致谢

我们要首先感谢指导老师陈东祥，指导我们学习数字图像处理，从而为我们打开了一扇通往新世界的大门。我们刚接触数字图像处理这门课程前对其一无所知，接触到了许多新的知识，如：算子、滤波器等知识。相信这些知识在以后的学习生活中会给我们带来巨大的帮助。同时我们还要感谢亲爱的组员们，在我们互相配合一起努力下完成了这一篇论文。