ÉCOLE CENTRALE DE NANTES

Foundation Masters

2020 / 2021

Driver Sleepiness detection using computer vision

---

presented by

Anqiu Hu

Zhihao REN

Supervisor :

Konstantin Akhmadeev

Contents

Abstract

Computer vision (CV) has become a necessary component of many robotics and artificial intelligence related applications. Computer vision enables machines to estimate the necessary parameters of the environment to adapt or interact with it. We use Python and learn basic and intermediate techniques from image processing, OpenCV and dlib libraries to implement a program that analyzes the input video stream and concludes whether the car driver is sleepy or is sleeping. Such a system could help prevent car accidents in cases where the driver is unable to objectively assess his condition.

In this project, we learned image processing techniques such as color space conversion, morphological manipulation, denoising, and the basics of face detection and facial marker extraction to deal with the problems arising from recognizing drivers' faces in different environments.

Late in the project, we ran into some problems with the dataset. When we managed to extract the feature matrix, it was just a combination of features per frame, not sure if this makes sense "sweat_smile". When trying to extract the label matrix, there were a lot of 0s and 1s in the annotations, and at first we thought that each number corresponded to a frame. Then we noticed that each video is about 10 minutes long and there are 10*60*30 frames in total. It seems that the numbers and frames do not correspond one-to-one. We don't know how to match them now.

Key Words : Computer vision(CV), Python, dlib libraries, OpenCV, sleepy, facial recognition

1 Background

1.1 The Drowsy Driving Problem

When drivers doze off, turning up the radio, opening the windows, drinking coffee or stopping for a break are all ways to ease the drowsiness. Many people complete the trip without incident.

However, the above approach sometimes doesn't work so well. Sleepiness often comes on suddenly, and falling asleep at the wheel is obviously a dangerous thing to do. But even if you don't fall asleep, your driving ability can be greatly impaired when sleepiness overtakes you. Fatigued driving is one of the major problems in auto accidents[1].

NHTSA estimates that 91,000 crashes involved fatigued drivers in 2017. Of those, 50,000 were injured and nearly 800 were fatal. However, the traffic safety, sleep science and public health communities believe this number is still underestimated[2]. Fatigued drivering can prevent drivers from paying attention to road conditions, impairing their ability to make good decisions or prolonging their reaction time when they need to brake or steer.

Based on the above, we try to design a program to capture and analyze the driver's facial state in real time to determine whether the driver is sleepy or not. If he is judged to be in a state of fatigue, he will be reminded to stop and rest in time by emitting a beep. Because the safest way to avoid fatigued driving is to relax.

1.2 Current status of research

Face recognition, a biometric technology for identification based on the

information of human face features. Face recognition was first studied by researchers in the 1960s, and really entered the primary application stage in the late 1990s, and its technical maturity has reached a high level since its development[3]. Not only in the field of public security, for example, the military machine department, criminal investigation to track down fugitives, crime identification, e-commerce and electronic payment in the commercial field, such as body temperature monitoring, face payment, smart home and other living scenarios, also use face recognition technology. In a word, face recognition technology has penetrated into every aspect of our life.

Currently, there are many codes shared on the web about face recognition technology. They have different focuses, from static face recognition, i.e., recognizing faces and their five senses from photos, to real-time face recognition; from simply recognizing faces to deeper applications based on recognizing faces. In more detail, there are already some ready-made codes about our project, for instance, image color channel conversion[4] , face keypoint detection[5] , keypoint detector training and testing[6],ect...

1.3 Researching significance

Although there are many off-the-shelf programs available, we hope to implement face detection and determine whether a driver is fatigued through autonomous learning. Based on the provided video, a model is trained and generated. The video set includes many special cases, such as wearing sunglasses, being in a low light environment or some different physiological characteristics. If the project goes well, we will also analyze more special cases to be more relevant to practical applications. For example, depending on the degree of fatigue, there are roughly three types of human facial expressions: yawning (opening the mouth and keeping it in this state for a relatively long time), blinking (or

slightly closing the eyes, when the number of blinks increases and the blink rate becomes slower), and nodding (nodding sleepily). By judging the face direction, position, pupil direction, eye opening and closing, blink frequency, pupil contraction rate and other data to start with, through these data, the driver's attention is calculated in real time to analyze whether the driver is driving fatigued and make timely safety warning decisions.

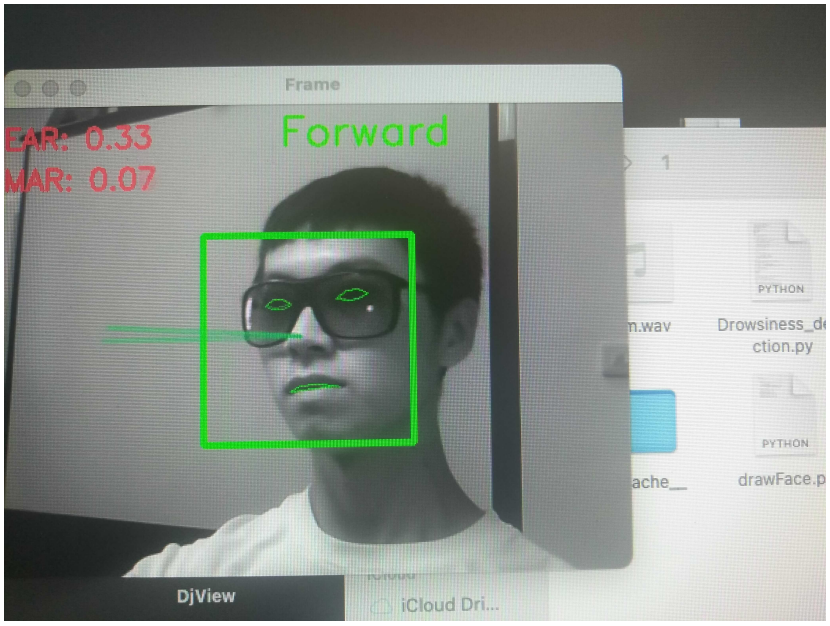


Figure 1.1: Detecting the facial features of people with sunglasses.
(Source: secret)

2 Tool : software

2.1 Computer vision

Computer vision is a branch of artificial intelligence. In simple terms, computer vision is the use of machines instead of the human eye to make measurements and judgments. Computer vision system is through the machine vision products (i.e. image ingestion device, divided into CMOS and CCD two kinds) will be ingested target into image signal, transmitted to the special image processing system, to get the morphological information of the target, according to the pixel distribution and brightness, color and other information, into digital signal; image system to these signals for various operations to extract the characteristics of the target, and then according to the discriminatory The image system performs various operations on these signals to extract the characteristics of the target, and then controls the action of the equipment on site based on the results of the discrimination[7].

Computers have the advantage of being able to perform repetitive tasks quickly without the distraction or fatigue that humans do. Convolutional neural networks can recognize objects with accuracy comparable to that of humans. Computer vision can save our money, drastically reduce our repetitive work, and help us analyze and interpret images or videos. Computer vision can also help us analyze and evaluate data in a matter of seconds.

2.2 Open-CV

OpenCV is an open source computer vision and machine learning software library that plays an important role in real-time operations. This open source library has a large number of optimized algorithms that easily process images and videos to recognize objects, faces, and more by using these algorithms.

We use cv2 in OpenCV to call the camera and capture the face information in real time.

```
import cv2
cap = cv2.VideoCapture(0)    #open the camera
cv2.circle(...)    #Circle the face with a circle
```

2.3 dlib libraries

DLib is an open source C++ library implementing a variety of machine learning algorithms, including classification, regression, clustering, data transformation, and structured prediction.

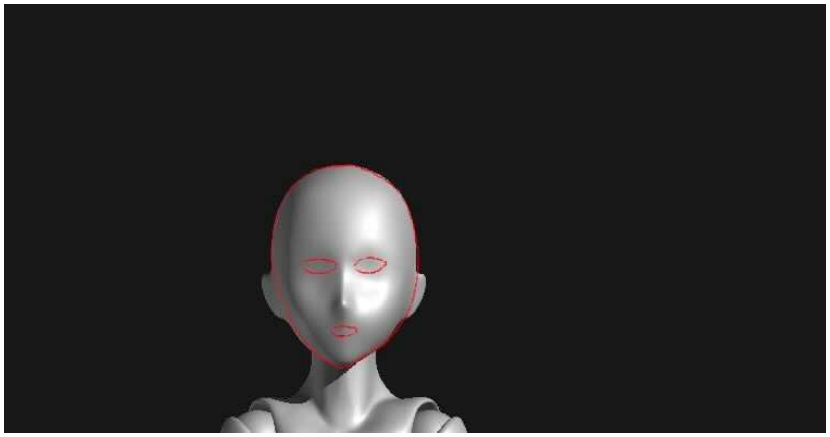


Figure 2.1: Capture facial information and mark the face, eyes and mouth with circles based on 68 facial feature points

For instance, we need to use the shape_predictor in the dlib library to capture 68 feature points of a face. And the dlib library provide a basic code that gets the input from the webcam and draws the landmarks over the face in real time.

```
import dlib
detector = dlib.get_frontal_face_detector()
preditector = dlib.shape_predictor(landmarks.dat)
#use the preditector to capture facial information
```

3 Data and Research methods

3.1 Driver drowsy detection dataset

For our training and test data, we used the Driver drowsy detection dataset from our advisor for detecting multi-stage drowsiness. The end goal is to detect not only extreme and visible cases of drowsiness but allow our system to detect softer signals of drowsiness as well. Driver drowsy detection dataset consists of both male and female drivers, with various facial characteristics, different ethnicities, and 5 different scenarios. The videos are taken in real and varying illumination conditions. The scenarios contain BareFace(NoGlasses), Glasses, Sunglasses, Night-BareFace(Night-NoGlasses) and Night-Glasses. Every video may contain 2 kinds of status: drowsy, non-drowsy. Each video is different situation with different status transition.This allowed us to obtain a sufficient amount of data for both the alert and drowsy state. For each video, we used OpenCV to extract 1 frame per second starting at the 3-minute mark until the end of the video.

There were 68 total landmarks per frame but we decided to keep the landmarks for the eyes and mouth only (Points 37–68). These were the important data points we used to extract the features for our model[9] .

3.2 Feature Extraction

Based on the facial landmarks that we extracted from the frames of the videos, we are searching for suitable features for our classification model.

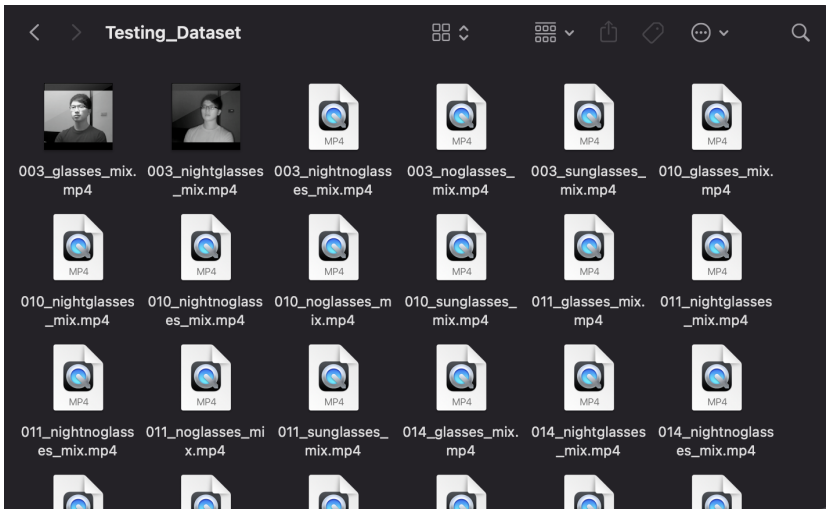


Figure 3.1: Driver drowsy detection dataset

(Source: secret)

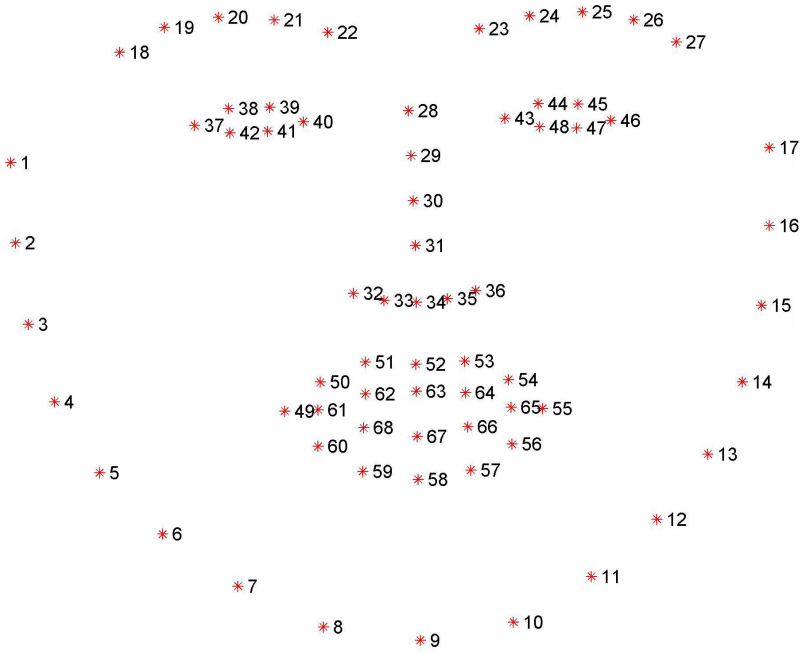
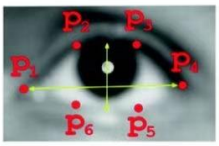


Figure 3.2: The full set of facial landmarks that can be detected via dlib library (Source : `https://hackaday.io/project/27552-blinktotext/log/68360-eye-blink-detection-algorithms`)

- **Eye Aspect Ratio(EAR) :** EAR, as the name suggests, is the ratio of the length of the eyes to the width of the eyes. The eye aspect ratio is an estimate of the open state of the eye. When people is drowsy, their eyes are likely to get smaller and they are likely to blink more. Based on this hypothesis, we expected our model to predict the class as drowsy if the eye aspect ratio for an individual over successive frames started to decline i.e. their eyes started to be more closed or they were blinking faster. The following equation is calculated, and if the eye aspect ratio is detected to be below 0.25, we will determine that the driver is in a fatigued state[9] .

```
def eye_aspect_ratio(eye):
  A = dist.euclidean(eye[1],eye[5])
  B = dist.euclidean(eye[2],eye[4])
#compute the euclidean distances between the two sets of
#vertical eye landmarks (x,y)-coordinates
  C = dist.euclidean(eye[0],eye[3])
  ear = (A + B) / (2.0 * C)
#compute the eye aspect ratio
  EYE_AR_THRESH = 0.25
#set the threshold 0.25
#A --eye[1]-point 38, eye[5]-point 42
#B --eye[2]-point 39, eye[4]-point 41
#C --eye[0]-point 37, eye[3]-point 40
#the points are in the Fig. 3.2
```



$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

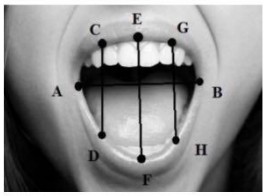
Figure 3.3: eye aspect ratio (Source: https://laptrinhx.com/drowsiness-detection-with-machine-learning-1320742370/)

- **Mouth Aspect Ratio(MAR):** Similar to the EAR, the MAR, as you would expect, measures the ratio of the length of the mouth to the width of the mouth. Our hypothesis was that as an individual becomes drowsy, they are likely to yawn and lose control over their mouth, making their MAR to be higher than usual in this state[10] .We calculate the MAR based on the following equation, but the project is not currently progressing here.

```
def mouth_aspect_ratio(mouth):

  A = dist.euclidean(mouth[1], mouth[7])
  B = dist.euclidean(mouth[2], mouth[6])
  C = dist.euclidean(mouth[3], mouth[5])
  D = dist.euclidean(mouth[0], mouth[4])
  mar = (A + B + C) / (3.0 * D)
  return mar

# construct the argument parse and parse the arguments
```



$$MAR = \frac{|EF|}{|AB|}$$

Figure 3.4: mouth aspect ratio (Source: `https://laptrinhx.com/drowsiness-detection-with-machine-learning-1320742370/`)

- **Pose Estimation:** We need six points of the face , is nose tip, chin, extreme left and right points of lips, and the left corner of the left eye and right corner of the right eye. We take standard 3D coordinates of these facial landmarks and try to estimate the rational and translational vectors at the nose tip. Now, for an accurate estimate, we need to intrinsic parameters of the camera like focal length, optical center, and radial distortion parameters. After obtaining the required vectors we can project those 3D points on a 2D surface that is our image.

  In computer vision the pose of an object refers to its relative orientation and position with respect to a camera. The pose estimation problem described there is often referred to as Perspective-n-Point problem or PNP in computer vision. In this problem the goal is to find the pose of an object when we have a camera. A 3D rigid object has only two kinds of motions with respect to a camera.

  1. Translation : Moving the camera from its current 3D location (X, Y, Z) to a new 3D location (X', Y', Z') is called translation.
  2. Rotation : You can also rotate the camera about the X, Y and Z axes. A rotation, therefore, also has three degrees of freedom. There are

many ways of representing rotation [11] .

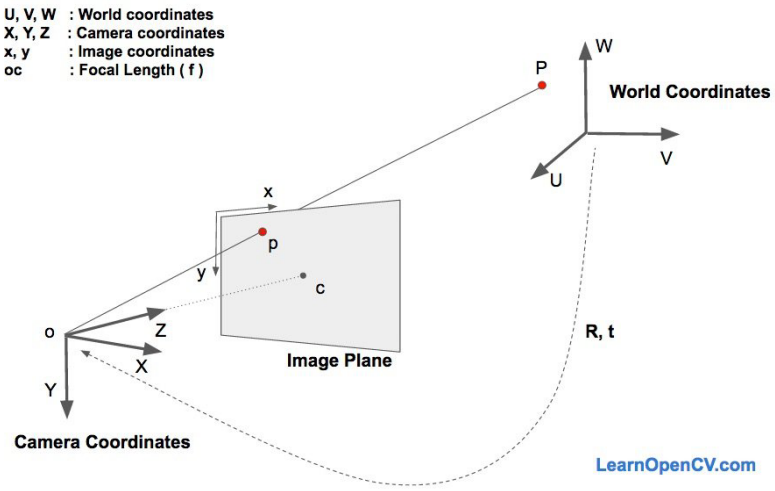


Figure 3.5: mouth aspect ratio

(Source : `http://aiuai.cn/aifarm913.html`)

4 Difficulties

4.1 Different physiological characteristics

Some people have small eyes and may have an EAR ratio that is less than the set fatigue level, so we need to design dynamic ratio changes.

4.2 head angle changing

When our head angle changes, our EAR changes slightly, and it is especially important to exclude this distracting factor. For example, when we are backing up, we need to look at the rearview mirror for a longer time, and this is the time to avoid mismeasurement.

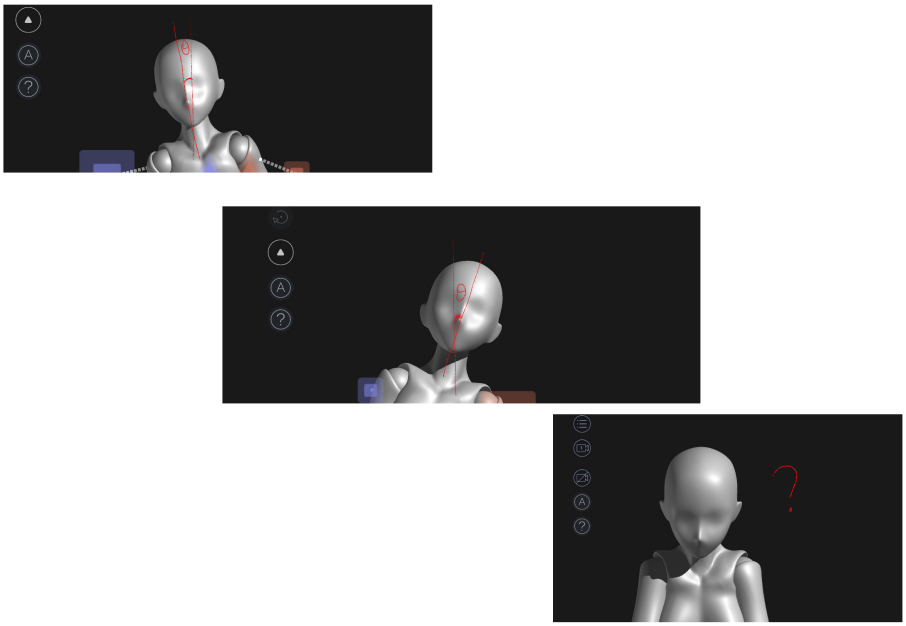


Figure 4.1: head angle changing

4.3 Training face recognition models

It is very difficult to train proprietary models and build your own cascade classifier based on existing datasets. This is where our work currently comes to a halt.

Conclusion

Through our project, we learned how to call Python's OpenCV library and dlib library to understand some techniques of image processing, such as color space conversion, extraction of facial markers, etc. We achieved to capture facial detail information of human face dynamically and extract facial features. However, we encountered difficulties in training the model and had no idea how to write Python code to implement the training process and it was difficult to continue.

References

[1] https://www.cdc.gov/sleep/features/drowsy-driving.html[Z]

[2] https://www.nhtsa.gov/risky-driving/drowsy-driving[Z]

[3] https://developer.aliyun.com/article/778706[Z]

[4] https://blog.csdn.net/hongbin_xu/article/details/78347484[Z]

[5] One Millisecond Face Alignment with an Ensemble of Regression Trees by Vahid Kazemi and Josephine Sullivan, CVPR 2014.[D]

[6] https://blog.csdn.net/jcx1314/article/details/65937839[Z]

[7] https://en.wikipedia.org/wiki/Computer_vision[Z]

[8] https://towardsdatascience.com/seeing-the-world-through-computers-eyes-and-why-it-matters-to-you-40152124d20[Z]

[9] https://hackaday.io/project/27552-blinktotext/log/68360-eye-blink-detection-algorithms[Z]

[10] https://laptrinhx.com/drowsiness-detection-with-machine-learning-1320742370/[Z]

[11] http://aiuai.cn/aifarm913.html[Z]