TED UNIVERSITY

CMPE 492

Updated Low-Level Design Report

Project Name: Drive Safe-Off

Team Members

Gökçe BEKAR

Rabia Esra ŞENDUR

Rumeysa OMAY

Oğuzhan Uğur SARISAKALOĞLU

Supervisor

Venera ADANOVA

Jury Members

Aslı GENÇTAV

Orkunt SABUNCU

Contents

Low-Level Design Report

1. Introduction

Driving is a task that requires the driver's full attention. Whether the driver is fully focused on the task, it can reduce the ability to detect and react to risky situations that exist or occur during driving. Drowsiness is one of the unsafe behaviors that drivers should not do while driving.

Williamson et al. conducted a study in a driving simulator and found that drivers who reported higher levels of sleepiness were more likely to have crashes and centerline crossings. In addition, slower reaction times have been associated with drowsy drivers in studies by He et al.

Drive Safe-Off is a system developed to solve this problem.It is a program that detects behaviors that indicate drowsiness by the driver - such behaviors, eye closure for a certain period of time, yawning for a long time - and warns during driving. By doing this, it aims to reduce and prevent the accidents that may occur and the deaths that may occur as a result of these accidents.

1.1 Object design trade-off

Given that drowsiness significantly increases the risk of accidents, we decided to design a system that can instantly detect drowsiness and send a warning to the driver. In order to achieve this, we decided to examine the driver's facial, eye and mouth movements in real-time while driving. Among these movements, features such as the number of blinking, eye aspect ratio, mouth aspect ratio, and yawning frequency are being examined. When any drowsiness is detected, the necessary warning is sent to the driver to prevent any accident that may be caused by the drowsiness.

Considering that the driver will be within certain physical limits and her/his mobility will be limited during driving, we used a suitable dataset for two purposes. Therefore, we used it for the detection & tracking part and the feature of the eye and mouth parts.

First of all, we detected the face and the eyes of the driver by Haar feature-based Cascade Classifiers. In addition, we implemented the tracking process for the face of the driver, taking into account the situations where detection might be insufficient. If the person's face is blocked by an object or the person turns their head in another direction while the face detector is running in a video, the face detector will likely fail since the Haar cascade classifier will not be able to recognize the face. Therefore, a good tracking algorithm will solve the problem when faced with this kind of situation.

Secondly, we used the same datasets to be able to use anomaly detection algorithms. Thanks to the various videos in the dataset, we were able to examine different types of frames. Anomaly detection helps identify events in any data set that deviate from normal behavior. For example, a situation where the driver starts blinking more than usual can be detected with anomaly detection algorithms. Machine learning is used gradually in the implementation of anomaly detection algorithms. Thanks to this, safer driving is provided by detecting the anomalies that the driver shows while driving. In addition, anomaly detection implementation is not required in the driver's mouth. If the mouth aspect ratio of the driver is higher than a certain value, it will be evaluated as yawning.

For the eye, the sum of the ear values of the driver in each frame is divided by the number of frames, if the value at t time deviates much from the average of the values obtained so far, this will be called an anomaly behavior. In short, it is planned to be applied based on the Moving Average Method.

1.2 Interface documentation guidelines

Every class has a name, description, attributes, methods and package or packages that it relates to.

The names of the classes are not established on a specific rule. Class names are made up of nouns, or strings of nouns that best express the function of the class.

Methods are named with verb phrases, fields, and parameters with noun phrases.

1.3 Engineering standards (e.g., UML and APA)

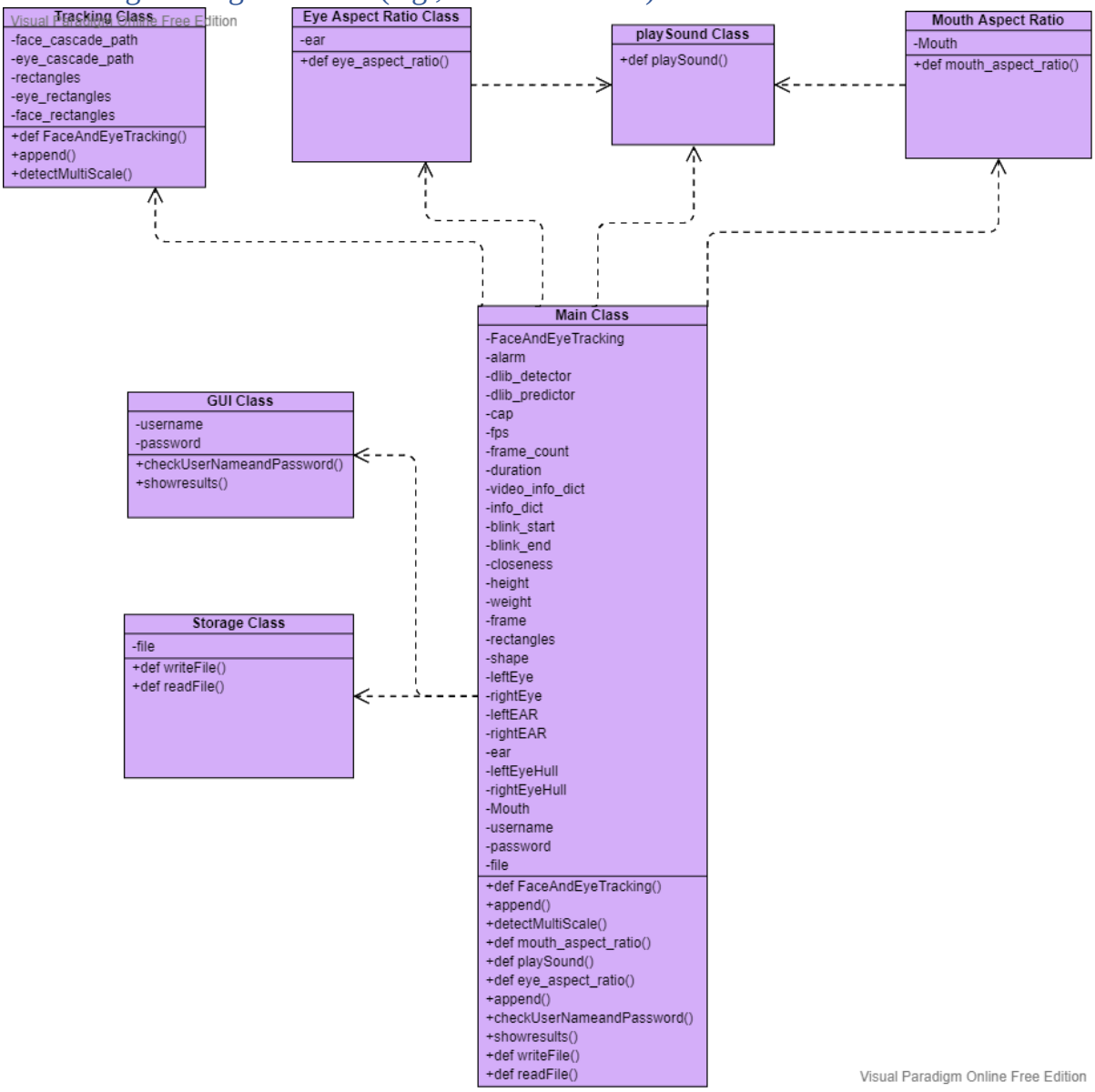


Figure 1: UML Diagram of Drive Safe-Off Project

In order to visualize object-oriented systems in our project, UML Class Diagram is used. It is a type of static structure diagram which represents the system's structure by presenting the system's classes, attributes, methods, and relationships between these objects. As can be seen in Figure 1, the UML diagram of the Drive Safe-Off project is shown. In the next reports, there may be changes such as adding or removing sections on the UML. This should be taken into account.

References of reports during technical writing of system are formatted in APA format. This standard provides readers to recognize references easily.

1.4 Definitions, acronyms, and abbreviations

UML: Unified Modeling Language

EAR: eye aspect ratio

MAR: Mouth aspect ratio

FPS: Frames per second

2. Packages

OpenCV : It is an open source library used in real-time computer vision applications.

CV2: It is the name that OpenCV developers chose when they created the binding generators,

NumPy: It is a Python library used for working with arrays.

Dlib: It is a modern C++ toolkit that contains machine learning algorithms.

distance: It computes the euclidean distance between the two vertical eye landmarks (x and y coordinates)

argparse : The argparse module makes it easy to write user-friendly command-line interfaces. It parses the defined arguments that are taken from the system.

FPS: It is used to read a number of frames from the video and see how much time has elapsed to calculate frames per second.

playSound: The playsound module is a cross-platform module that is used to play audio files.

Tracking: The class that is used to track face and eye after the detection.

Joblib: Enables us to save and load trained models.

3. Class Interfaces

When creating a class, each class consists of descriptions, attributes, and methods. Since these classes are still under development, changes can occur. The currently assumed order is as follows:

3.1 Tracking Class

Class Name: Tracking Class

Description: This class performs the necessary operations to follow after detecting the units that need to be followed (eye, mouth) on the face.

Attributes:

String face_cascade_path
String eye_cascade_path
double[] rectangles
double[] eye_rectangles
double[] face_rectangles

Methods:

double[] FaceAndEyeTracking() : This method tracks the face and the eye.
append() : ????
detectMultiScale() : ???

3.2 Eye Aspect Ratio Class

Class Name: Eye Aspect Ratio Class

Description: This class calculates the EAR value of the eye. It is a fixed value when the eye is open, but approaches 0 as the eye is closed.

Attributes:

double[] ear

Methods:

int eye_aspect_ratio() : This method calculates EAR value of the eye.

3.3 playSound Class

Class Name: playSound Class

Description: This class warns you when a behavior that should not have happened is detected. These behaviors are long-term eye closure and long-term yawning.

Attributes:

Methods:

void playSound() : This method play sound for warning.

3.4 Storage Class

Class Name: Storage Class

Description: This class responsible for writing and reading file. These files keep data shared while tracking, username and password.

Attributes:

File file

Methods:

- void writeFile(File file) : This methods writes the data to the file.
- void readFile(String pathToFile): This method reads the data in the file.

3.5 MouthAspectRatio Class

Class Name: MouthAspectRatio Class

Description: This class calculates the MAR value of the mouth. Mouth aspect ratio is almost zero when the mouth is closed, and the aspect ratio of the mouth increases slightly when the mouth is open.

Attributes:

double[] Mouth

Methods:

double mouth_aspect_ratio() : This method calculates MAR value of the mouth.

3.6 GUI Class

Class Name: GUI Class

Description: This class handles everything related to the GUI.

Attributes:

String username
String password
File file
Storage storage

Methods:

boolean checkUserNameandPassword(username,password): This method checks if username and password match.
void showresults(File file): This methods shows the results in the saved files.

3.7 Main Class

Class Name: Main Class

Description: The carry method of the system handles entire intersection of classes and method provides an singular execution from the beginning to the end of the driving sesion.

Attributes:
FaceAndEyeTracking
alarm
fhog_object_detector dlib_detector
shape_predictor dlib_predictor
VideoCapture cap
float fps
int frame_count
float duration
dict video_info_dict
dict info_dict
int blink_start
int blink_end
int closeness
int height
int weight
ndarray frame
list rectangles
full_object_detection shape
ndarray leftEye
ndarray rightEye
float64 leftEAR
float64 rightEAR
int ear
int Mouth
ndarray leftEyeHull
ndarray rightEyeHull

Methods:
def FaceAndEyeTracking()
append()
detectMultiScale()
def mouth_aspect_ratio()
def playSound()
def eye_aspect_ratio()
append()

4. Methodology of Tracking and Examining Eye State

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. It can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of a specific object in an image. Face-detection algorithms focus on the detection of frontal human faces. In this task, OpenCV (Open-Source Computer Vision Library) which is an open-source computer vision and machine learning software library are used. OpenCV contains many pre-trained classifiers for face, eyes, mouth, etc. In this system, face and eye classifiers are used in order to detect human faces and eyes. We perform the face detection for each frame of the videos in our data set. The first process of our system is to detect the face and the eyes of the driver by Haar feature-based Cascade Classifiers. Haar feature-based cascade classifiers are an effective object detection method proposed by Paul Viola and Michael Jones. It is a machine learning-based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Furthermore, tracking algorithms are usually faster than detection algorithms. The reason for this is that while tracking an object detected in the previous frame, there is a lot of information about the appearance of the object. Also, thanks to the previous frame, the position, direction and speed of its movement are known. Thus, in the next frame, it can use all this information to predict the object's position in the next frame. To find the object accurately, a small search around the expected location of the object can easily find the object's new location. The initial bounding box supplied by the user is taken as a positive example for the object, and many image patches outside the bounding box are treated as the background. With a new frame applied, the classifier is run at the pixel facing a small neighborhood around the current location to generate a potential positive sample.

Finally, the biggest reason why we use tracking after detection is to improve performance. Tracking can help when detection fails: If the person's face is blocked by an object or the person turns their head in another direction while the face detector is running in a video, the face detector will likely fail since the Haar cascade classifier will not be able to recognize the face. Therefore, a good tracking algorithm will solve the problem when faced with this kind of situation.

Figure 2 shows the results of our tracking algorithm for human eyes. Accuracy measured 96.6% in good light conditions and without glasses.

	Number of Frames(1000)
Detected Eye	966,00
Undetected Eye	34
Error Rate	0,034
Accuracy	96,60%

Figure 2: Accuracy of the tracking eye object.

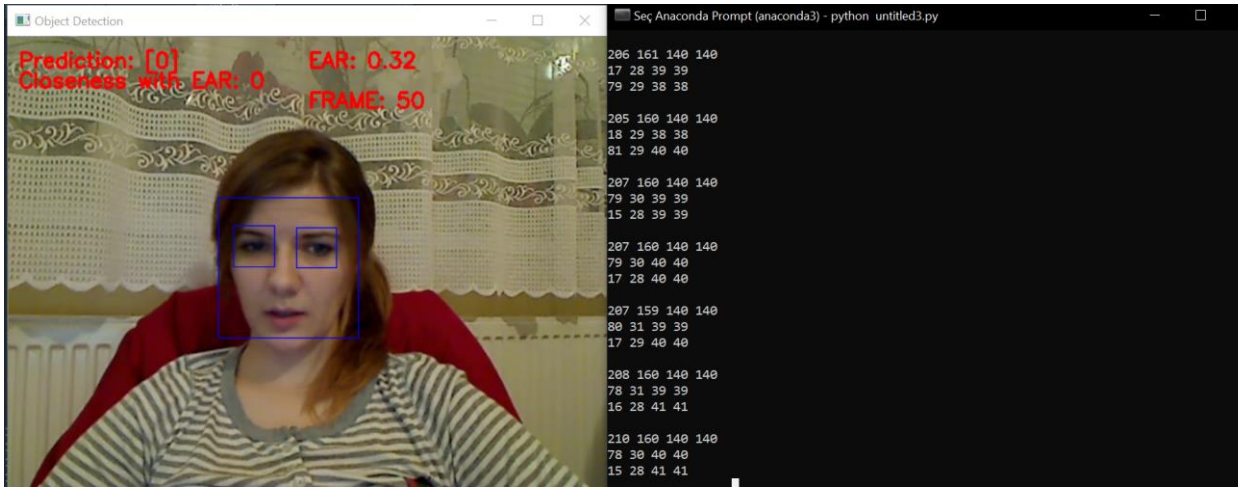


Figure 3: Coordinates of human face and eyes that are detected and tracked.

In figure 3 which is given above, when the first line which starts with 206 is examined, the first number shows the x coordinate and the second number shows the y coordinate of the human face. The third and fourth numbers represent the width and length of the face. The numbers in the second and third lines represent the coordinates, widths, and lengths of the eyes. As the face and eyes continue to be tracked in each frame, these numbers change accordingly.

Training

In order to train the model, data with labels were used. The way each frame is labeled is as follows:

frame ID : blink ID : NF : LE_FC : LE_NV : RE_FC : RE_NV : F_X : F_Y: F_W : F_H : LE_LX : LE_LY : LE_RX : LE_RY: RE_LX : RE_LY : RE_RX : RE_RY

frame ID - Frame counter based on which the time-stamp can be obtained (separate file).

blink ID - Unique blink ID, eye blink interval is defined as a sequence of the same blink ID frames.

non frontal face (NF) - While subject is looking sideways and eye blink occurred, given variable changes from X to N.

left eye (LE), right eye (RE), face (F)eye fully closed (FC) - If subject's eyes are closed from 90% to 100%, given flag will change from X to C.

eye not visible (NV) - While subject's eye is not visible because of hand, bad light conditions, hair or even too fast head movement, this variable changes from X to N.

face bounding box (F_X,F_Y,F_W,F_H) - x and y coordinates, width, height**l**

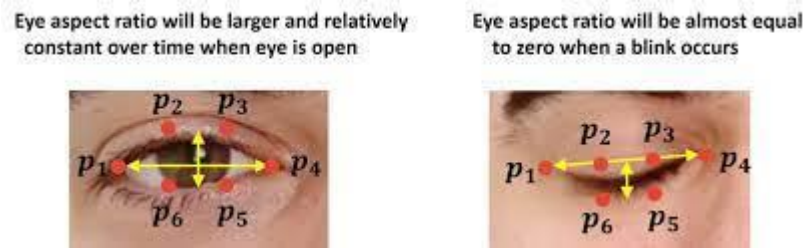
left and right eye corners positions - RX (right corner x coordinate), LY (left corner y coordinate)

So if a frame consist a blink it will be like:

2851:9:X:X:X:X:X:240:204:138:122:258:224:283:225:320:226:347:224

As seen above, the blink ID keeps track of how many times the eye has been closed until then. -1 means open eye.

After detecting the position of the eye in the tracking part, the ear value was calculated according to the points around the eye.



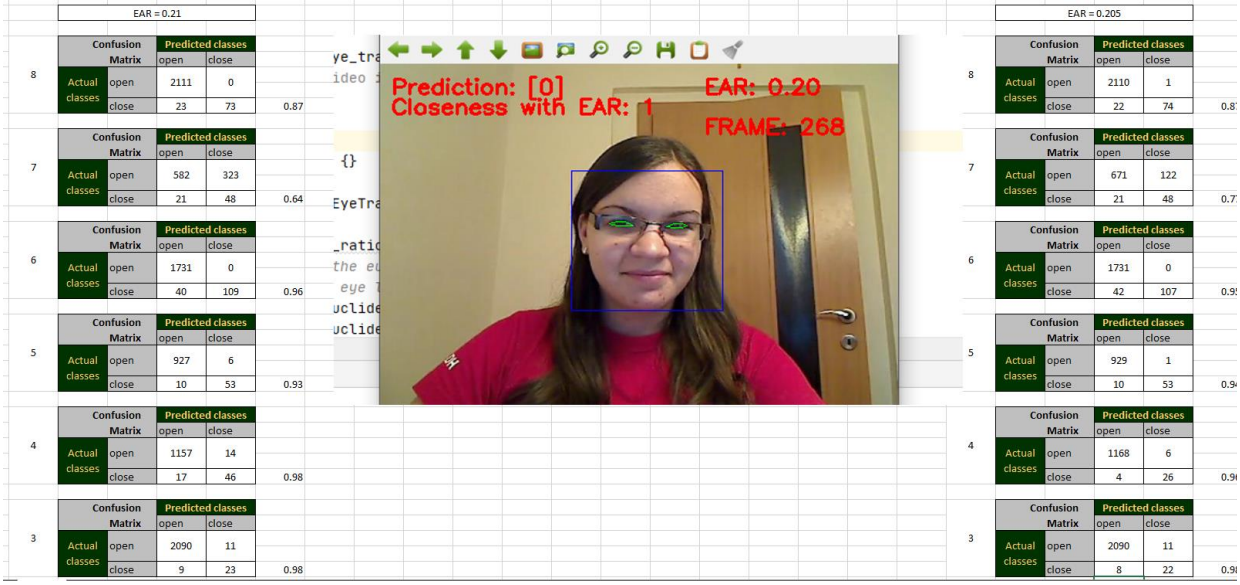
$$\mathrm{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

During the training, ear value was calculated for each frame. The calculated value was defined as closed if it was lower than the initial constant, and open if it was higher.

The predicts are not close (0) or close (1). This binary resulting leads us to use logistic regression model because logistic regression is used when the dependent variable(target) is categorical. In the training process model is fitted to predict state of eye with given eye aspect ratio.

Since how long the eye is closed is important to us, the next step is to detect the closeness of the eye by making unsupervised-anomaly detection. Ear value has a higher value with the eye open, and a lower value with the eye closed. For this reason, the driver's ear values coming from each frame while driving are summed up and divided by the total frame result until that time. We plan to detect anomaly behavior in this way, since there will be an average decrease when the eye is closed.

After training process, we are going to achieve persistence of model with Joblib library. Dump and load function enables us to save and load functionality to process. In tthis way, pretrained models can be provided to any session of driving.



Finally, we investigated trained models. For this purpose, we summarize the model by comparing the result of their labels. One way that helps us to improve is the modeling confusion matrices. We deploy our model on videos with 8 different people. As a result, the model challenging with a test video. The cause is that the ear aspect ratio diversify within people. As seen in he figure above, a person with low distinction on eye aspect ratio that predict close is fails. Considering this distinction, to determine the constant improved the model a little. At the current state, we tried to solve this problem with specify the more common eye aspect ratio during training. However, the newer approaches should be able to avoid the importance of driver's diversity.

5. Glossary

Session: The time interval between driving starts and ends.

User: It is the person who will drive the vehicle in a certain time period.

Supervisor: The person has the administrative authority of the system environment.

Detection: It is a computer technology being used in a variety of applications that identifies human faces in digital images.

Tracking: It is the faster and more accurate determination of the location of the object with the detection made in the previous frame and the information obtained.

Anomaly Detection: It is the identification of uncommon situations that raise suspicions by differing significantly from the major part of the data.

6. Reference

1. Williamson, A.; Friswell, R.; Olivier, J.; Grzebieta, R. Are drivers aware of sleepiness and increasing crashrisk while driving? Accid. Anal. Prev.2014,70, 225–234.

2. He, J.; Choi, W.; Yang, Y.; Lu, J.; Wu, X.; Peng, K. Detection of driver drowsiness using wearable devices: A feasibility study of the proximity sensor. Appl. Ergon.2017,65, 473–480