Driver's Gaze Zone Estimation in Realistic Driving Environment by Kinect

by

Chong Luo

Thesis submitted to the University of Ottawa
In partial fulfillment of the requirements
For the M.A.Sc. degree in
Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Driver's distraction is one of the main areas, which researchers are focusing on, in design of Advanced Drivers Assistance Systems (ADASs). Head pose and eye-gaze direction are two reliable indicators of a driver's gaze and the current focus of attention. Compared with other methods that make use of head pose only, methods that combine eye information can achieve higher accuracy. The naturalistic driving environment always presents unique challenges (e.g., unstable illumination, jolts, etc.) to video-based gaze estimation and tracking systems. Some methods can achieve relatively high proficiency in the stationary laboratory environment, but they may not be suitable for the unstable driving environment. In addition, performing in real time or near-real time is another consideration for gaze estimation in an ADAS. Therefore, these special challenges need to be overcome to design ADASs.

In this thesis, we proposed a new driver's gaze zone estimation framework designed for the naturalistic driving environment. The framework combines head and eye information to estimate the gaze zone of the driver in both daytime and nighttime. The framework is composed of five main components: Facial Landmark Detection, Head Pose Estimation, Iris Center Detection, Upper Eyelid Information Extraction, and Gaze Zone Estimation. First, Constrained Local Neural Field (CLNF) is applied to obtain the facial landmarks in the image plane and the 3D model of the face in the object frame. In addition, extracting region of interest (ROI) is utilized as an optimization strategy for CLNF facial landmark detection. Second, head pose estimation can be regarded as a Perspective-n-Point (PnP) problem. Levenberg-Marquardt optimization method is used to solve the PnP problem based on the 2D landmark locations in the image plane and their corresponding 3D locations in the object frame. Third, a regression model-based method is employed to obtain the iris center from eye landmarks detected in the previous part. For upper eyelid information extraction, a quadratic function is utilized to model the upper eyelid, and the second-order coefficient is extracted. Finally, the head pose and the eye information are combined to form a feature vector, and Random Decision Forest classifier is utilized to estimate the current gaze zone of the driver from the feature vector extracted.

The experiment is carried out in the realistic driving environment in both daytime and nighttime with three volunteers by Kinect sensor V2 for Windows that is put at the back of windshield. Weighted and unweighted accuracy are utilized as evaluation metrics in gaze zone estimation. Weighted accuracy evaluates gaze zones with different significance while unweighted accuracy treats each gaze zone equally. Experiment results show that the gaze zone estimation framework proposed in this work has better performance compared to the reference in the daytime. The weighted and unweighted accuracy of gaze zone estimation reach 96.6% and 95.0% for daytime, respectively. For nighttime, the weighted and unweighted accuracy can reach 96% and 91.4%.

Acknowledgements

I would like to thank everyone who made this possible.

First of all, I must thank my parents for encouraging me and supporting me over these years. Their love and patience make who I am.

Next, I would like to thank my supervisor, Dr.Azzedine Boukerche. His continuous guidance and support during my research make this possible. He is like a father who works hard to support families in PARADISE Laboratory.

I also want to give a special appreciation to our group leader, Dr.Abdelhamid Mammeri, who is a very supportive tutor and guides my work efficiently.

Other special thanks must be given to volunteers, Tony, Felipe, and Erin participated in my experiments. It is very kind and patient for them to spare their time in my experiments in both nighttime and daytime.

Finally, I must appreciate all PARADISE team members, who supported and helped me during my research. I would like to give special thanks to Mr. Zongzhi Tang, who is a friend and gives me a lot of useful suggestions and assistance in my research. Thanks to Dr.Peng Sun for guiding me in my work and thesis writing and providing valuable suggestions. Thanks to He Li for helping me revise my thesis patiently and giving me many great advice. Thanks to Shichao Guan, Weihong Zhao, Samaneh, Zhijun Hou, Yiheng Zhao and all those who make this possible.

Table of Contents

List of Tables

List of Figures

Nomenclature

SVM	Support Vector Machine
NHTSA	National Highway Traffic Safety Administration
POS	Pose from Orthogra4phy and Scaling
IDASs	Intelligent Driver Assistance Systems
ADASs	Advanced Driver Assistance Systems
RDF	Random Decision Forest
EEG	Electroencephalogram
EOG	Electro-Oculography techniques
GRNN	Generalized Regression Neural Network
ANN	Artificial Neural Network
CNN	Convolutional Neural Networks
KNN	K-Nearest Neighbors
DOF	Degrees of Freedom
CoHMEt	Continuous Head Movement Estimation
BMPCA	Bilateral-projection Matrix Principle Component Analysis
LGO	Localized Gradient Orientation
SVR	Support Vector Regressor
LLS	Longest Line Scanning
OCEM	Occluded Circular Edge Matching
AAM	Active Appearance Model
CLM	Constrained Local Model
RANSAC	Random Sample Consensus
PPCA	Probabilistic Principal Component Analysis
PDM	Point Distribution Model
CLNF	Constrained Local Neural Field

LNF	Local Neural Field
OVA	One-Versus-All
AVA	All-Versus-All
ECOC	Error-Correcting Output-Coding
MAP	Maximizes Posterior Probability
SOP	Scaled Orthographic Projection
LR	Logistic Regressor
CCNF	Continuous Conditional Neural Field
RLMS	Regularized Landmark Mean-Shift
NU-RLMS	Non-uniform Regularised Landmark Mean Shift
ROI	Region of Interest
PnP	Perspective-n-Point
LM	Levenberg-Marquardt
ID3	Iterative Dichotomiser 3
CART	Classification and Regression Tree
SDK	Software Development Kit
IR	Infrared
HOG	Histogram of Oriented Gradients

Chapter 1

Introduction

This chapter will present the background of Advanced Driver Assistance Systems (ADASs), the motivation, the contribution and the organization of this thesis.

1.1 Background

In last decades, studies on driver safety have attracted more and more attention, and significant improvements have been obtained. Nonetheless, critical and harsh traffic accidents still happen throughout the world [76]. Several factors play vital roles in the causes of the these accidents, such as increasing use of private cars, booming of transportation, and growing usage of cellphone or other electronic devices in the car. Currently, both governments and industrials around the world have made great efforts to deal with driving safety-related issues. For example, some safety standards on driving are issued by government such as the mandatory rest in long-distance driving [76]. However, these solutions are not enough, because growing evidence shows that *drivers' inattention* (e.g., texting, drowsiness, etc.) during driving is the main factor causing car/truck crashes and incidents [78], [109]. The estimation in paper [122] shows that 80%-90% of fatal and injury crashes are related to drivers.

As defined by National Highway Traffic Safety Administration (NHTSA), distraction is a specific type of inattention that occurs when drivers divert their attention from the driving task to some other activities instead, such as calling, eating, and navigating [100]. Inattention also includes fatigue, physical, and emotional conditions of the driver [102]. Although the definition varies in different references [102], *distraction* and *drowsiness* are two main subjects in drivers' behavior analysis [76].

In 2012, 3,328 people lost their lives, and additional 421,000 people were injured in the motor vehicle crashes involving distracted drivers in America, which accounts for 16% of all motor vehicle traffic accidents [102]. After two years, the corresponding numbers reported in NHTSA's new report [103] remain the same, which are 3179, 431,000 and 16%, respectively. NHTSA also gives an estimation that there is an average of 83,000 crashes each year related to drowsy driving between 2005 and 2009 [101]. The situation is

similar in Canada, according to the Canadian Council of Motor Transport Administrators (CCMTA), the fatigue-related driving accounts for up to 21% of motor vehicle collisions every year [45].

Some statistical reports of traffic accidents provided by UK and USA show that avoiding distraction and drowsiness driving can prevent these severe crashes significantly [76]. In addition, a comprehensive survey [116] on motor vehicle crashes indicates that there is about 30% decrease of the probability of injury-related accidents when a driver is alerted of the unseen dangers by passengers. Therefore, researchers turn their focus to the driver-centric systems which can detect and analyze the inattention and distraction of drivers and then give alerts to them when they may face potential dangers or even guide the driver in dangerous situations [130]. Such systems can be called intelligent driver assistance systems (IDASs) or ADASs [130], [129].

Generally, distraction and drowsiness are two primary factors, which researchers are focusing on, in design of ADASs [76]. According to whether the vision feature is used in the design, ADASs can be divided into two categories:

1. *Visual feature-based systems*: The systems often make use of computer vision techniques to detect distraction and drowsiness. Head, face, eyes, and mouth, which are four fundamental parts of the driver, are used for feature extraction in the detection part of these systems. Eye state and blinking analysis, mouth and yawning analysis, and facial expression analysis are three major concerns in drowsiness detection researches [125], [83], [119], [63] while head movement analysis and gaze analysis are adopted in drivers' distraction detection [111], [76]. In addition, the image or video analysis combined with machine learning methods, such as Random Decision Forest (RDF) [129], is often used in vision-based systems.

2. *Non-visual feature-based systems*: This kind of system is often designed by two approaches [76]: 1) One approach is to make use of drivers' physiological information. Physiological information includes electrical activity of the brain and heart rate. Electroencephalogram (EEG) technique, which tracks and records brain wave patterns, is an effective method for detecting the drowsiness of the driver; however, this method is intrusive because several sensors are connected to the driver [86]. 2) The other one is to take advantage of vehicle parameters. Vehicle parameters often contain steering wheel movement, acceleration pedal movement, and so on. Researchers find that there exists a relationship between vehicle parameters and driver's state of drowsiness [57].

In this thesis, we will focus on visual feature-based distraction systems, which will be explained in the following section.

1.2 Motivation and Contribution

Distraction is defined as a specific category of inattention that happens when drivers turn their attention from the driving task to some other activities [100]. For example: when

the driver sends a message to someone while driving, his/her eyes will turn off the road. In addition, at least one of the hand cannot participate in the controlling of the steering wheel. These behaviors increase the probability of life-threatening collision, especially when potential danger exists. Visual-Feature based techniques are commonly utilized for Distraction Detection Systems (DDSs), and they provide a non-contact, non-invasive, and easy-setup solution compared to methods that need sensors attached on the driver.

Head pose and eye-gaze direction (i.e., the gaze direction relative to the head) are two reliable indicators of a driver's gaze and the current focus of attention [97]. Head pose and eye-gaze direction are combined to provide complete gaze information of the driver [81].

In this thesis, we proposed a new framework for gaze zone estimation in the realistic driving environment, and the experiments are conducted in both daytime and nighttime. Here are the main contributions of the thesis:

- A new gaze zone estimation framework is proposed, which combines head and eye information to estimate the gaze zone of the driver for both daytime and nighttime. Experiment results show that the gaze zone estimation framework proposed has better performance compared to the reference in the daytime. The weighted accuracy reaches 96.6%, and unweighted accuracy reaches 95.0% for daytime. For nighttime, the unweighted accuracy can reach 91.4%, and the weighted accuracy can reach 96%, which is also promising.
- Two new datasets are collected from the realistic driving environment. One dataset contains 29,000 images that are annotated manually for gaze zone estimation. The other dataset contains 2,100 images that are manually annotated for iris center detection. These datasets can be exploited further in future research.
- Microsoft Kinect for Windows V2 is employed to obtain color images in the daytime and infrared images in the nighttime. It's the first time that Kinect is used in the gaze zone estimation in the realistic driving environment.

1.3 Thesis Outline

This thesis is organized as follows:

Chapter 2 will be a literature review covering five sections, namely video-based gaze estimation, head pose detection, iris center detection, facial landmark detection, and multiclass classification techniques.

Chapter 3 will describe the proposed framework for driver's gaze zone estimation in detail. The description follows the structure of the framework, which is composed of facial landmark detection, iris center detection, upper eyelid information extraction, and gaze zone estimation.

Chapter 4 will present the experiments information in four main parts, consisting of the setup of experiments and dataset, results for iris center detection, verification of upper eyelid information, and results for gaze zone estimation.

Chapter 5 will cover conclusions for the proposed framework, and some analyses on the results in chapter 4. The possible methods for improvement and enhancement will be discussed in the future work.

Chapter 2

Literature Review

In ADASs, using head pose only for gaze estimation is not able to provide adequate details compared with using the complete gaze information (i.e., head pose and eye-gaze direction), because the former method can not distinguish the focus of the driver when he/she only has eye movements without head movements. In some practical applications, head pose and head dynamics are used for gaze estimation [82]. However, eye cues are required to obtain a comparatively more accurate performance in the naturalistic driving environment. On the other hand, in consideration of the financial cost, computational time and complexity, operation complexity and intrusiveness, the methods of eye-gaze direction estimation with coarse accuracy is enough and suitable in driving environment, compared to those with precise accuracy. Since video-based gaze estimation methods are less intrusive compared to methods where sensors are attached on the driver's skin, this chapter will review different video-based gaze estimation methods. The following sections will present video-based gaze estimation, head pose detection, iris center detection, facial landmark detection, and multiclass classification techniques, respectively.

2.1 Gaze Estimation Based on Eyes and Head

In this context, gaze can refer to either the focus point/zone of a person or the gaze direction. The task of gaze estimation is to model the relationship between the images and the gaze point/zone or gaze direction. Head pose (i.e.,position and orientation) and the orientation of the eyeball together determine the gaze direction of a person. The gaze direction is changed by rotating either or both eyeballs and head. Eyeball movement in 3D space can also be regarded as the iris/pupil movement in the 2D image space. It is a fact that a person moves the head first to a comfortable position and then move the eyeballs [69]. Therefore, head pose can decide the coarse field of view while eyeballs provide the detailed information of local gaze directions.

A gaze estimation system is regarded as a good system if it is non-intrusive, non-obstructive with easy and flexible setup and the required accuracy. Previously, there were many eye-tracking methods which are intrusive. In Electro-Oculography techniques

(EOG), sensors are attached to the skin around the eyes to measure an electric field existing when eyes rotate. By recording small differences in the skin potential around the eye, the position of the eye can be estimated [38]. Compared with systems that have sensors attached to the skin or mounted on the head, video-based systems are non-intrusive and provide more freedom for users. As mentioned before, we will focus on video-based gaze estimation methods, which are commonly used in practice to provide non-intrusiveness. These video-based methods can be broadly grouped into four categories: appearance-based, eye-model-based, feature-based and hybrid gaze estimation [69]. In the following subsections, the four categories of gaze estimation methods will be described in detail, and researches in these four categories will be reviewed.

Before the gaze estimation process, some calibration procedures are required, and these calibrations can be classified into camera calibration and personal calibration [69]. Usually, one or both kinds of calibrations are utilized by a particular system. Camera calibration refers to getting the intrinsic matrix of a camera, and personal calibration refers to the estimation of corneal curvature.

2.1.1 Feature-based Methods

Feature-based gaze estimation methods utilize local features extracted from the image, such as eye contours, eye corners, and reflections [69]. These features extracted usually have high relation with the gaze.

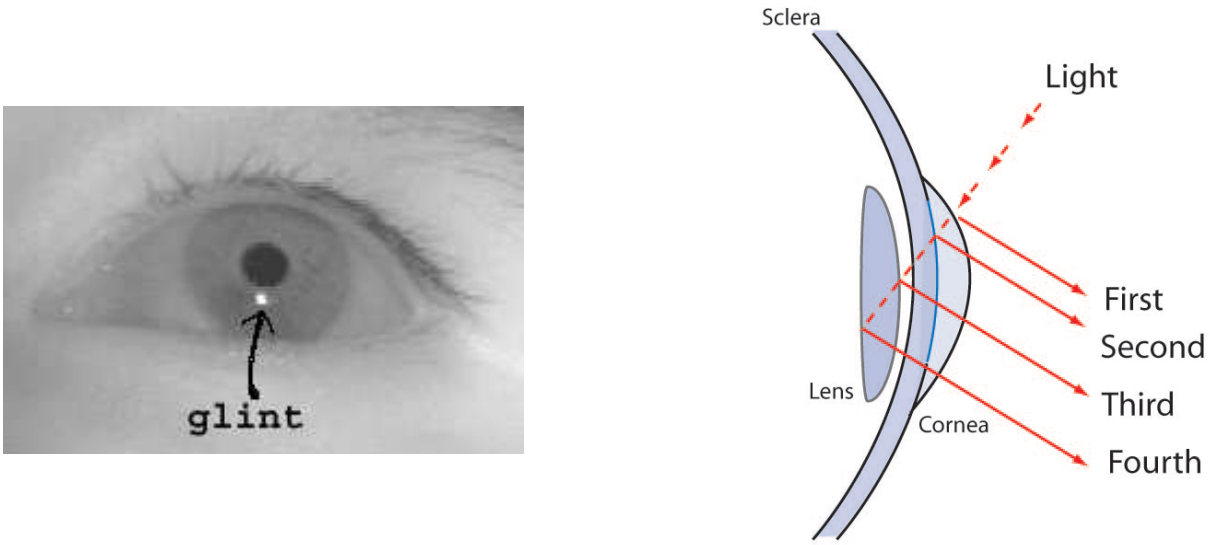


Figure 2.1: (a) Eye image with corneal reflection (glint). Taken from [155]. (b) Light reflection structure. Taken from [69].

Figure 2.1 shows the reflection features. When the light falls on the corneal, some reflections occur on the boundary between the lens and the corneal [53], shown in Fig-

ure 2.1(b). The corneal reflection is called a glint that is captured by a camera and forms a small white dot in the eye shown in Figure 2.1(a). Pupil-glint vector is the displacement vector between corneal reflection and pupil center, and used as a common feature in early gaze estimation systems. Feature-based methods model the relationship between features extracted from the image and current gaze information (i.e., gaze direction, gaze zone, or gaze point). Parametric models, such as polynomial, and non-parametric models, such as the neural network, are both exploited by researchers. The computation of intersection between gaze direction and object can be avoided because the model can learn the geometry relationship implicitly.

The first video-based eye tracker designed by Merchant et al. [92] uses polynomial expressions as the mapping function [38]. The authors designed such a system as early as 1974. Polynomial expressions have become one of the most popular mapping techniques [35] since then. Pupil-glint vector is utilized as the feature in their method. The authors found that the mapping from pupil-glint vector to gaze coordinate is linear in small eye rotation and becomes nonlinear in more significant eye rotation. Thus a linear mapping from pupil-glint vector to gaze coordinate is applied for small eye rotation while the nonlinear polynomial function models the relationship for larger eye rotation. Infrared light is utilized to get the pupil center by using bright-pupil effect, which will be discussed in the iris detection section.

As an alternative to parametric expressions, neural networks and their variants are also popular in feature-based methods, which assume a non-parametric form to represent the mapping from image features to gaze coordinates.

Zhu et al. [154] proposed a method based on Generalized Regression Neural Network (GRNN). The analytic mapping function is not needed, and the head movement can be implicitly taken into account. They found the pupil parameters such as orientation and r (i.e., ratio of the major and minor axes of the pupil ellipse) have a relationship with the head pose. Therefore, their method utilizes the pupil parameters, pupil-glint displacement, and glint coordinates as features and the features are mapped to the screen coordinates. The advantages of the method are that no calibration is necessary after initial training and that the gaze tracker can perform robust and accurate gaze estimation under rather significant head movement. Furthermore, the mapping function can be generalized to other individuals outside training set. Although this method is not as accurate as some commercial gaze tracker, it can handle head movements while still producing an accuracy of about 5 degrees for gaze estimation.

Later, Zhu et al. [155] proposed another method utilizing Support Vector Regression (SVR) as non-parametric mapping model between extracted features and gaze coordinates, which can construct a highly nonlinear generalized gaze mapping function that takes the head movement into consideration. Pupil-glint vector and 3D pupil position are the input of SVR. Experiment results show that their method can achieve an gaze accuracy of about 1.5 degrees under natural head movement.

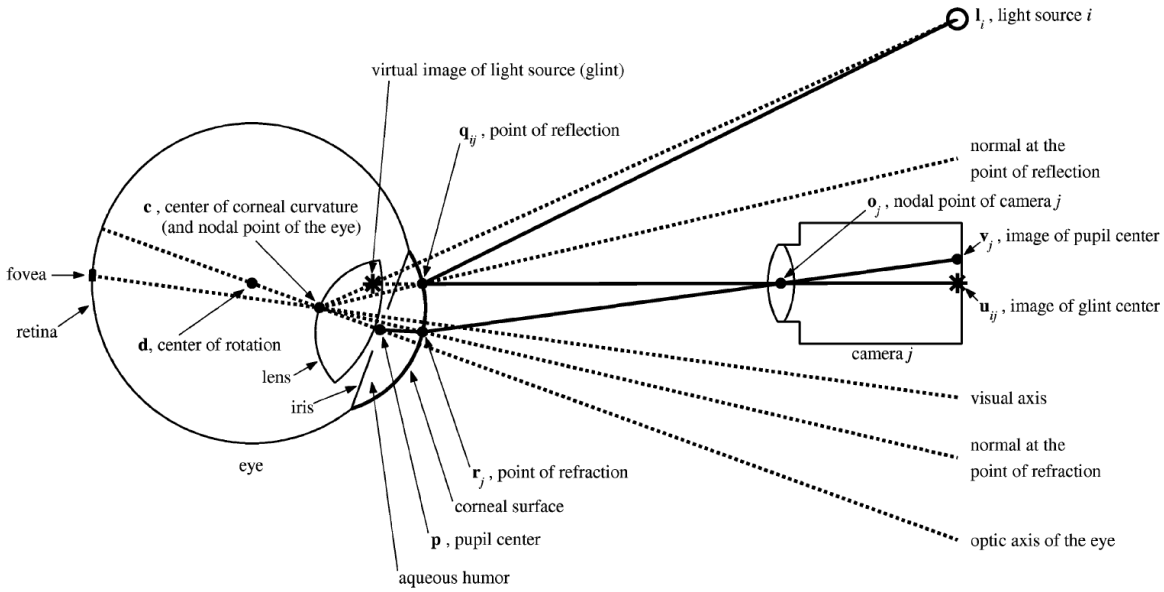


Figure 2.2: General model of the eye structure and optical properties of the eye. Taken from [65].

2.1.2 3D Model-based Methods

3D model-based methods model the common physical structures of the human eye geometrically [69] and make use of optical properties of eyes to obtain the gaze direction vector in 3D space. A general physical model of the eye structure and optical properties of the eye are presented in Figure 2.2. The gaze point/zone can be estimated by calculating the intersection of obtained gaze direction vector and the plane of a particular depth/object. The eyeball is near spherical and is usually modeled as a spherical ball with a person-specific radius R. As shown in Figure 2.2, the visible parts are pupil (the smallest circle in the middle of the iris), iris (the circle part around the pupil) and sclera (the white components of eye). In front of the iris, there is a transparent layer on the surface of the eye, which is called corneal. The corneal is not visible in the 2D image, but it plays a significant role in the structure of visual generation. Two important axes in the structure are the optical axis and the visual axis, where the visual axis is real gaze direction of the eye. The line crossing pupil center, corneal center, and eyeball center is named optical axis. The line crossing corneal center and fovea is called visual center, where fovea is a small region in the center of the retina. From the definition, visual and optical axes intersect at the corneal center. The angular offset between optical and visual axes are fixed for a particular person but may vary for different persons. Usually, calibration is applied for model-based methods to get some personal intrinsic parameters, such as the intrinsic eye parameters which include the angles between visual and optical axes, the radius of corneal and so on.

To make the eye model more suitable for the practical applications, most of the 3D model-based gaze estimation methods assume that the eyeball and corneal curvature are spherical as shown in Figure 2.3. The smaller sphere is corneal curvature with center C_P

while the exterior larger one is eyeball with center C. P represents pupil center. 3D model-based methods follow a common strategy. At first, the optical axis is estimated in 3D space by obtaining the pupil center and corneal/eyeball center. Next, the visual axis is estimated using the angle offset θ between optical and visual axes. Finally, the gaze point/zone is obtained by intersecting gaze direction with the scene geometry.

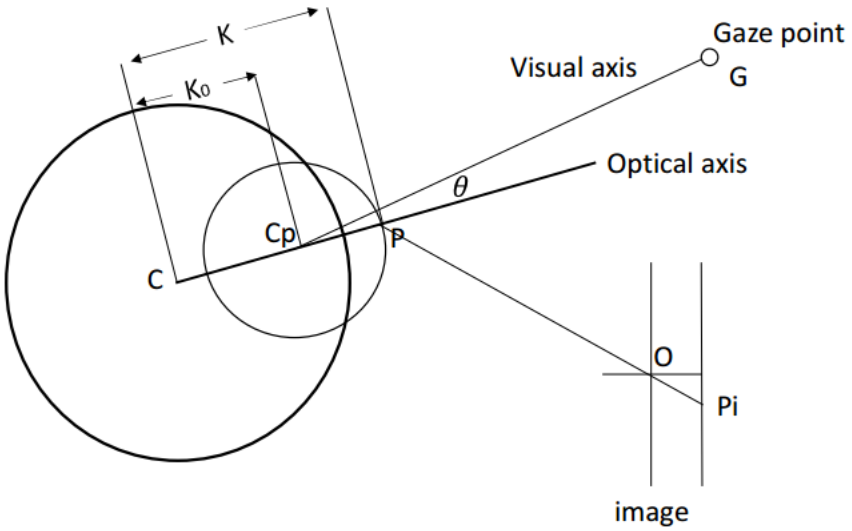


Figure 2.3: 3D eye model. Taken from [75].

Li et al. [75] designed a gaze estimation system based on the eye-mode shown in Figure 2.3 using an RGB-D camera (i.e., Kinect sensor). The eyeball center C and pupil center P are utilized to estimate the optical axis. Different from other methods, the method proposed by Li sets up a model to calibrate the eyeball center by gazing at a random target in 3D space, not predefined [75]. Before getting the 3D location of pupil center, a 3D head model is obtained from Kinect sensor. The visual axis can be estimated by adding the constant offset angle to the optical axis.

This process is carried out in an indoor environment. The results show that the method proposed can achieve good performance in every direction, with an accuracy of about 5 degrees. However, since the pupil tracking algorithm has an essential influence on the complete accuracy of gaze estimation, the method could be more accurate if the pupil center can be detected more accurately and stably. The pupil center detection method they utilized requests the pupil shown on the image as complete as possible, and this is not suitable for the realistic driving environment.

Chen et al. [37] utilized a similar 3D model-based system as Li et al. But they achieved the accuracy under 3 degrees with free head movement. The reason may be that Chen makes more personal calibration while Li uses the average parameter values.

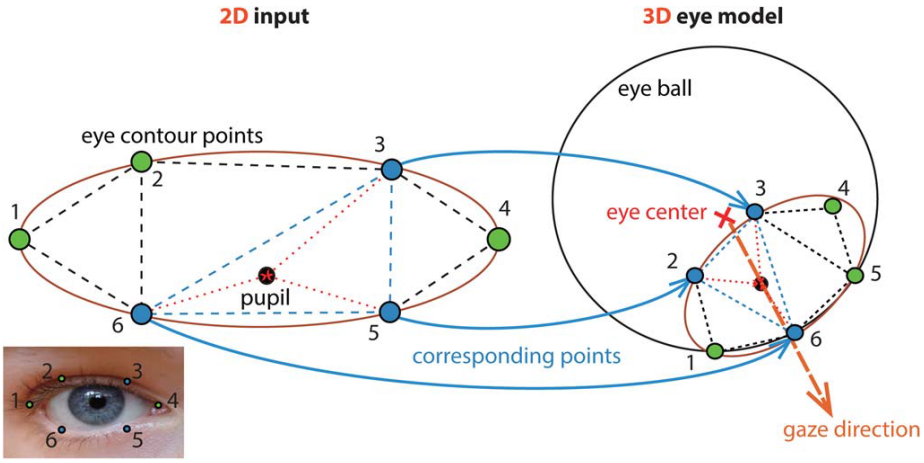


Figure 2.4: 3D gaze estimation. Taken from [136].

Vicente et al. [136] designed a driver's gaze tracking and eyes off road detection system with 3D eye model, shown in Figure 2.4. The pupil positions are segmented to form four different categories (i.e., four triangle meshes marked by dashed lines). After detecting the pupil location in the 2D image plane, the 2D pupil coordinate is mapped to corresponding coordinate in the 3D head frame. The 3D gaze direction is determined by the line cross eyeball center and pupil center in 3D space. The estimated gaze direction is then applied to detect whether the driver is looking away from a predefined space, on-road area. If the driver looks outside the space, there will be an alarm. The experiment is carried out in the real car, but they did not mention if it is in the realistic driving environment. The participators are instructed by audio to gaze specific area for 10 seconds to collect data. The false alarm rate in the on-road area is below 5%.

2.1.3 Appearance-based Methods

Feature-based and eye-model-based methods need detection of pupils or calibration of eyeball center, which are prone to introducing errors. In addition, some potential features containing useful information about gaze direction may be disregarded resulted from feature extraction. Appearance-based gaze estimation methods make use of the image data directly in a whole instead of explicitly extracting features or exploiting eye structure. The mapping relationship between image data and gaze direction or gaze points/zone are modeled directly. As a result, the variation information of different people, and the useful features will all be considered without the need of complicated calibrations and geometry information. These methods are mainly based on grayscale unit images, appearance manifold, Gaussian interpolation and cross ratios [121].

Baluja et al. [12] proposed an appearance-based non-intrusive gaze estimation system which can be customized to individual users. Artificial Neural Network (ANN) is utilized to learn the relationship between the extracted window, which contains eye region, and coordinates on the monitor that the subject is gazing at. The training data for ANN is collected by guiding the subject to gazing at a predefined path of a moving cursor on

the monitor. Totally 2000 images and their corresponding coordinates on the monitor are collected. The system can estimate the gaze coordinates from the extracted window pixels after obtaining ANN parameters from training. The system's online accuracy can reach 1.7° while the subject is allowed to move head freely. However, the subject is sitting in front of the camera while the data is collected, the movement of the head may not be enough for the wild outdoor environment.

Tan et al. [128] presented another appearance-based method which makes use of an appearance manifold model. An appearance manifold is a set of points in the high-dimensional space, representing the image data taken from different gaze coordinates [94], [99]. Every point in appearance manifold has its corresponding gaze coordinates. For a given image, the gaze coordinate can be estimated by finding the point in appearance manifold that is closest to the given image. Instead of utilizing a densely sampled spline to obtain the nearest manifold point, a nearest manifold point search technique is proposed, exploiting the topological information inherently presenting in the manifold model. Their method remains the original set of sparse appearance samples and utilizes linear interpolation among a small subset of samples to approximate the nearest manifold point. Compared with dimension reduction methods, their method keeps more useful information. The gaze estimation accuracy can reach 0.38° in the set of eye images labeled with corresponding gaze coordinates on the screen.

Previous appearance-based methods were usually carried out in a controlled laboratory environment with a lot of constraints. Recently appearance-based gaze estimation methods are usually combined with head pose estimation, which give more freedom to subject's head position and orientation [124].

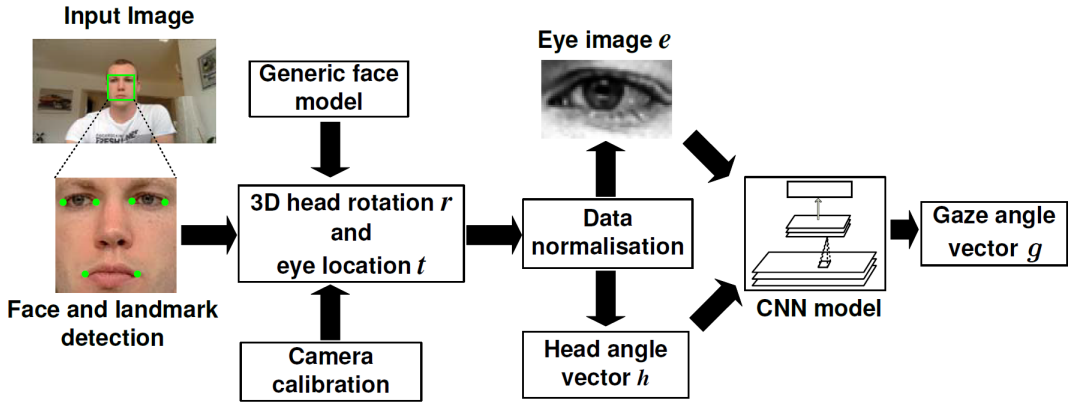


Figure 2.5: Appearacne-based method with CNN. Taken from [149].

Lai et al. [79] developed an appearance-based gaze tracking system allowing free head movement by combining head pose estimation method and appearance-based gaze estimation method. The head pose and eye region appearance are combined to represent points in high-dimensional manifold space. A random forest approach models the neighbor structure of these points, and the neighbor set is efficiently selected. After that, the best solution is

obtained from regression by L_1-optimization from those near samples. A typical camera-and-monitor system, where the subjects are guided to gaze at points on the monitor, is set up for data collection. The Experiment results show that their method can perform robust gaze tracking with free head movement and still achieve average estimation accuracy.

Zhang et al. [149] also proposed such a system combining head pose estimation method and appearance-based method. Figure 2.5 shows the framework of their proposed method. First, face and facial landmarks are detected to estimate the head pose further and obtain eye region. Second, the convolutional neural networks (CNN) is applied to learn the relationship between the combined head pose and eye images with gaze directions in the camera coordinate system. The experiment is carried out under the wild unconstrained condition. Totally 213,659 images are collected from 15 subjects during natural daily laptop use over more than 90 days, which is called MPIIGaze dataset. Their dataset has more variation in appearance and illumination, compared to existed ones. Their method achieves better performance in the most challenging cross-dataset evaluation, containing person- and pose-independent data, than some state-of-art methods. The within-dataset mean error is 6.3 degrees.

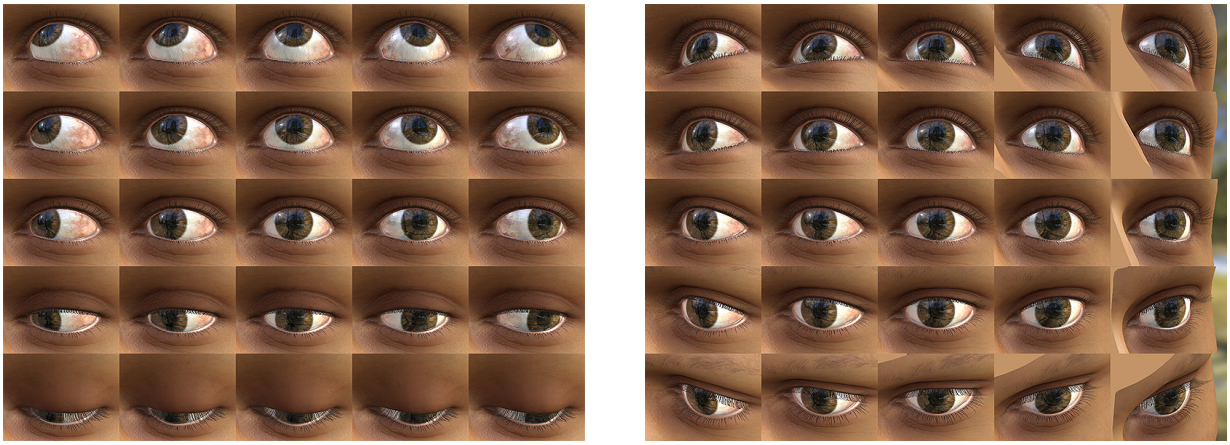


Figure 2.6: Synthesis eye images from controlled gaze direction and head pose. Taken from [146].

The vast amount of labeled training data with variable eye appearance, head pose, and illumination is an essential requirement for recent appearance-based gaze estimation. It takes a lot of time and works to collect such amount of data like Zhang et al. did. Thus, Sugano et al. [124] and Wood et al. [146] tried to make use of synthesised images for training. Sugano et al. generated 64,000 images, and Wood et al. collected one million synthesized images. The former utilizes a random regression forest while the later utilize k-Nearest Neighbors (KNN) estimator. Figure 2.6 illustrates the generated synthesis eyes. Left of figure shows renders with fixed head pose but varying gaze while right one shows fixed gaze but varying head pose. Experiment results of Sugano show that their method outperforms existing methods that use low resolution images, and the mean error of their method with cross-subject training is $6.5° \pm 1.5°$. Results from Wood show that these synthesized images

can be utilized to estimate gaze in difficult wild scenarios, even for extreme gaze angles or fully occluded pupil, and mean error is 9.95° when tested on Zhang's MPIIGaze Dataset.

2.1.4 Hybrid Methods

By combining two or more of the methods from appearance-based, feature-based, and eye-model-based methods, a trade-off can be made.

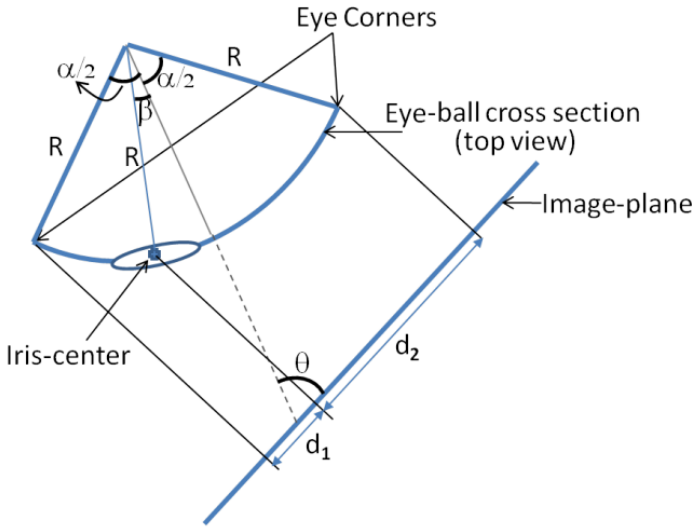


Figure 2.7: Eye model to compute horizontal gaze direction. Taken from [129].

Ashish Tawari et al. [129] designed a robust gaze zone estimation system in the naturalistic real-world driving environment, which combines feature-based and eye-model based methods. They proposed gaze-surrogate features from the eye model and upper eyelid. The eye model proposed provides horizontal gaze direction while upper eyelid provides vertical gaze information. Figure 2.7 shows the eye model they proposed.

Based on geometrical structure of the eye model, horizontal gaze direction with regard to horizontal head orientation β can be calculated by Equation 2.1 [129].

$$\beta = \theta - arccos\left(\frac{2}{d_1/d_2 + 1} sin\,(\theta)\, sin\,(\alpha/2) + cos\,(\alpha/2 + \theta)\right) \tag{2.1}$$

where α is the angle subtended by an eye in the horizontal direction, θ is yaw angle of head pose, and $\frac{d_1}{d_2}$ is the ratio of the distances of iris center from the detected corners of the eyes in the image plane [129].

Random Decision Forest (RDF) [33] classifier is then applied to classify different gaze zone of the driver according to the features obtained.

The proposed system is evaluated on their own dataset. The experiment is carried out in the naturalistic driving environment to collect the data. The experiment results show that the overall weighted accuracy is improved from 79.8% to 94.9%, when eye cues are added. The estimation accuracy for individual gaze zone is also improved.

Table 2.1 is a summary of methods for gaze estimation reviewed previously. In summary, although the feature-based methods using pupil-glint displacement are simple, having no need of complicated calibration, and relatively accurate, they are limited to particular applications since they perform not well under significant head movement. A wide-angle camera or additional camera can be used to release this limitation. However, both complexity and financial cost will increase as a consequence. The 3D model-based gaze estimation systems can tolerate natural head movements, but these methods usually require a complicated one-time system- or geometric-calibration. Some assumption or approximation can be used to simplify the calibration and computation. On the other hand, appearance-based approaches are not based on known parameters from feature extraction or many calibration parameters. They extract these information implicitly. Furthermore, they can also achieve free head movement by combining head pose estimation. They can even work person-independently, though the accuracy will decrease. However, they need extensive data for training to extract those useful information implicitly.

Compared with the person-dependent method, the person-independent method has more generalization to new individuals, and there is no need of training and calibration for particular individuals. The person-independent method is very convenient, efficient, but the accuracy will drop as a consequence. Person-dependent methods usually achieve higher accuracy. Besides, the environment has a huge influence on the results. The naturalistic driving environment always presents unique challenges to video-based gaze estimation and tracking systems. In an automobile, continuous lighting changing conditions, resulting in heavy shadows and high illumination variability, and unstable capture of data, caused by jolts, all put more difficulties on gaze estimation task. In addition, performing real-time or near-real-time is another consideration for gaze estimation in an ADAS. Therefore, the results in the stationary laboratory with stable illumination have higher proficiency than those in the wild. However, these methods may not work well in the driving environment. Unique approaches are needed to design for ADASs to overcome these special difficulties.

Paper	Method	Camera Type	Illumination	Environment	Metrics
Merchant et al.[92] 1974	Pupil-glint vector + polynomial expressions	Infrared	Unspecified	Laboratory	Angular gaze accuracy
Zhu et al.[154] 2004	Pupil-glint vector, pupil parameters, and glint coordinates + neural network	Infrared	Unspecified	Laboratory	Angular gaze accuracy
Zhu et al.[155] 2006	Pupil-glint vector and 3D pupil position + SVM	Infrared	Unspecified	Laboratory	Angular gaze accuracy
Li et al.[75] 2014	3D eye-model	Color and depth	Unspecified	Laboratory	Angular gaze accuracy
Chen et al.[37] 2008	3D eye-model	Color	Unspecified	Laboratory	Angular gaze accuracy
Vicente et al.[136] 2015	3D eye-model	Color and Infrared	Both bright and dark	In car	Eyes Off/On the Road accuracy
Baluja et al.[12] 1994	Appearance and artificial neural network	Unspecified	Unspecified	Laboratory	Angular gaze accuracy
Tan et al.[128] 2002	Appearance manifold model	Infrared	Unspecified	Laboratory	Angular gaze accuracy
Lai et al.[79] 2014	Eye appearance and head pose + random forest and regression	Color	Unspecified	Laboratory	Angular gaze accuracy
Zhang et al.[149] 2015	Eye appearance and head pose + convolutional neural network	Color	Unspecified	Wild	Angular gaze accuracy
Sugano et al.[124] 2014	Eye appearance and head pose + random regression forest	Color	Unspecified	Synthesised images	Angular gaze accuracy
Wood et al.[146] 2016	Eye appearance and head pose + KNN	Color	Unspecified	Synthesised images	Angular gaze accuracy
Ashish Tawari et al.[129] 2014	Eye information and eye-model + random forest	Color	Unspecified	Driving environment	Gaze zone accuracy
Proposed Method	Iris center, upper-eyelid, and head pose + random forest	Color and Infrared	Both bright and dark	Driving environment	Gaze zone accuracy

Table 2.1: Summary of reviewed gaze estimation methods

2.2 Head Pose Detection

In this subsection, we will review previous works on video-based head pose estimation techniques. For a better overview of the head pose estimation in computer vision, please refer to [98]. The head is commonly assumed to be a rigid object in head pose estimation, and in this thesis, head pose refers to either the orientation or joint of orientation and position of a head. The orientation has 3 degrees of freedom (DOF), which can be represented by pitch, roll, and yaw in the world frame, shown in Figure 2.8. As mentioned before, the head pose is a vital reference to know human's current attention. It is even applied for coarse gaze estimation. In addition, the gesture of people's head can convey rich and interpersonal information in a conversation [98].

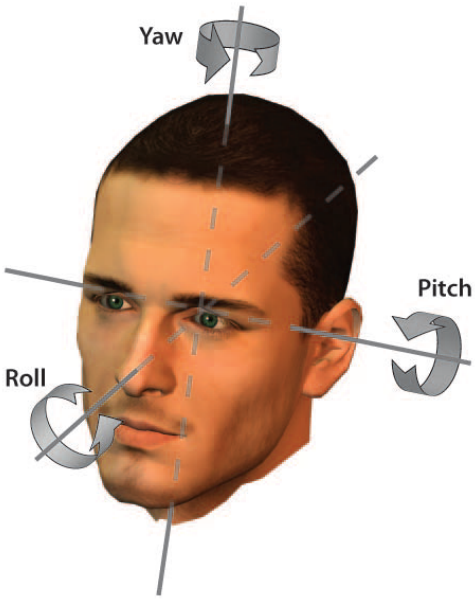


Figure 2.8: Head pose orientation. Taken from [98].

Methods for head pose estimation can be divided into three main categories in general: geometric/shape feature-based methods, appearance/texture feature-based methods, and hybrid (shape + texture) feature-based methods [130].

2.2.1 Shape-based Methods

Approaches based on shape features usually make an analysis on the geometric distribution of facial landmarks, combined with the face model, such as cylindrical, ellipsoidal and mean 3D face [91] to get the head pose. It is intuitive to assume that there is a mapping relationship between combination of the shape and distribution of facial landmarks, and the head pose.

The primary framework of the recent systems is illustrated in Figure 2.9. First, facial landmark positions in the image plane and their corresponding face model in 3D object

frame are detected/tracked. The head pose is obtained by applying Pose from Orthography and Scaling (POS) algorithm [50], which could find the projection relationship between the landmark positions in the 2D image and corresponding landmark positions in 3D face model. The projection relationship provides the face model orientation and position with respect to the camera, namely the head pose.

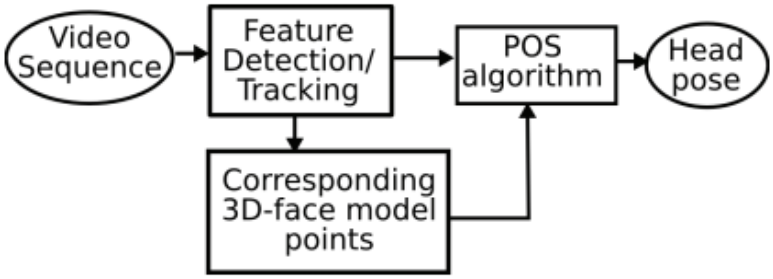


Figure 2.9: Framework for shape feature-based methods. Taken from [91].

Ohue et al. [107] proposed a shape-based head pose estimation method making use of left edge, right edge, and center edge of the face. A cylindrical face model is applied to find the yaw direction of the driver.

Tawari et al. [130] designed a Continuous Head Movement Estimation (CoHMEt) system focusing on continuously and accurately estimating driver's head pose even under significant deviations from the frontal pose. A distributed camera framework, which makes use of multiple cameras and a camera perspective selection algorithm, is proposed to increase the operating range of the head pose tracking system for drivers. The algorithm proposed independently estimate head pose from each camera stream, and then the best estimation is obtained by analyzing result of each camera. The authors utilized geometric information of the head to estimate driver's head pose and detected facial landmarks in image space and their corresponding positions in 3D face model, shown in Figure 2.10. POS algorithm requires at least four points of correspondences in general positions. The less deformable points: four eye corners, two nose corners, and a nose tip are utilized in POS algorithm.

The experiment is conducted in the naturalistic driving environment, and the data collection focuses on events that can cause significant head movement (away from the driving direction) because they are more related to driver's safety. After a thorough comparative study of different camera configurations, the system can reliably track the head movement over 96% of the time.

Martin et al. [91] designed a similar system and also conducted the experiment in the naturalistic driving environment. LISA-P Head Pose database is collected from the experiment which has head pose data from drivers of varying age, race, and gender in both daytime and nighttime. Moreover, a motion capture system is applied to obtain the ground truth of head pose. Evaluation on LISA-P database of proposed method shows that the average standard deviation is 7°. A single camera is utilized which may be the limitation of detection range.

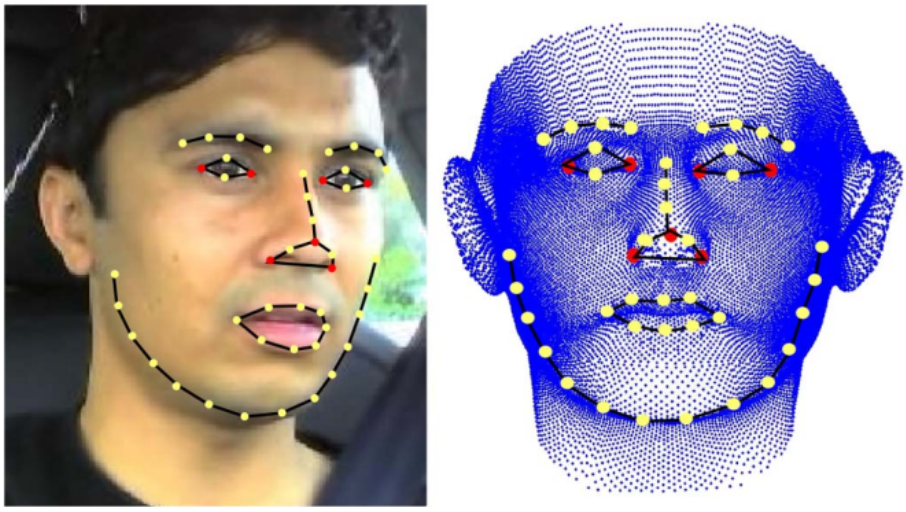


Figure 2.10: Facial landmarks and their corresponding positions in 3D face model. The solid red circles are the points utilized for the head pose calculation. Taken from [130].

2.2.2 Appearance-based Methods

Methods based on appearance usually assume that there is a mapping relationship between holistic facial texture (intensity distribution) and 3D head pose. Often, a considerable amount of dataset for training is needed in appearance-based methods.

Guo et al. [66] introduced a fast algorithm of face detection and head pose estimation to estimate the head pose of driver in real time. Mask Transform and Support Vector Machine (SVM) classiers are used to detect the face accurately. Only the face area is utilized for head pose estimation. The algorithm proposed applies Bilateral-projection Matrix Principle Component Analysis (BMPCA) to extract features from 2D image data directly. BMPCA is a variation of Two-dimensional Principle Component Analysis (2DPCA), and it not only extracts features efficiently but also maintains more powerful and excellent performance. Experiment results on videos demonstrate that the method is rapid, robust and efficient to deal with illumination changes, glasses wearing, and different head poses with moderate rotations.

Zhu et al. [153] presented a novel method to track 3D head pose in near-real time from an image sequence captured from an uncalibrated monocular camera. The main idea of their method is to detect and track 2D face, and 3D face pose simultaneously, instead of treating them separately. 3D face pose is constrained by face dynamics utilizing Kalman filtering, and face appearance in the 2D image. Kalman filtering can constrain the 3D head pose to a smaller range of possibilities. The 3D face pose is estimated from finding the best matching between the actual face image and the projected face image from the 3D face. Face matching is formulated as an optimization problem so that the exact face location and 3D face pose can be estimated efficiently. Moreover, the use of active infrared illumination, which allows detecting eyes robustly, can constrain the detected face in the image to avoid tracking drift issue. Since the method requires explicit initialization of the frontal face, and

the tracking process is performed from this initial face pose and placement. The system may be sensitive to initial face pose and placement. Experiment result shows that the estimation error for yaw and pitch is 2.9 degrees and 3.6 degrees.

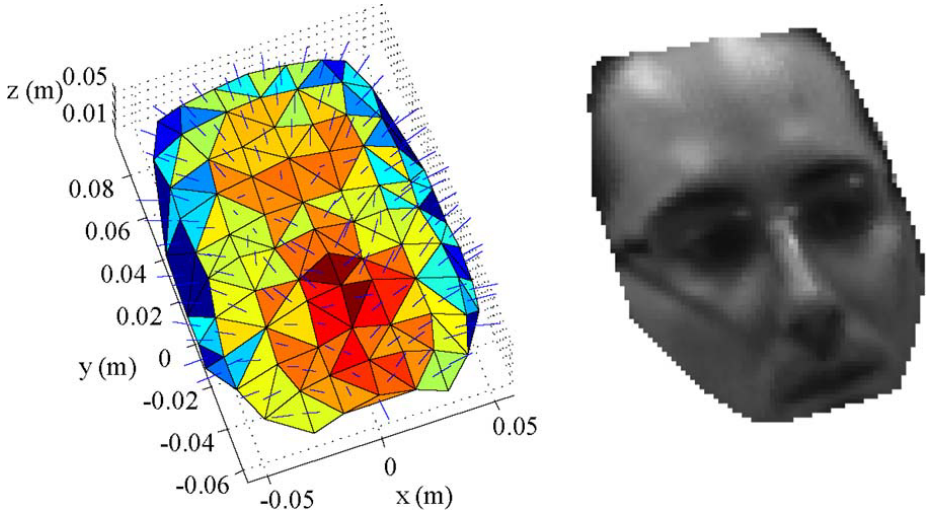


Figure 2.11: Rigid facial model used for initialization and tracking and an example of the model rendered by the tracking system. Taken from [95].

Murphy et al. [95] proposed a novel procedure for static head pose estimation and visual 3-D tracking. The system consists of three modules that detect the driver's head, initially estimate head pose, and continuously track head pose. Face area is detected by three cascaded Adaboost [142] face detectors at first in head detection module. Next, localized gradient orientation (LGO) histogram is extracted from the grayscale video image and input into three SVRs trained for head pitch, yaw, and roll orientation respectively in initial pose estimation module. The tracking module employs a new appearance-based particle filter for 3-D model tracking in an augmented reality environment, which is a virtual environment that simulates the view space of a real camera [96]. Figure 2.11 shows a rigid anthropometric texture-mapped 3D head model. The surface of the head is approximated by a set of convex polygons represented by the 3D vertices. Each vertex is assigned a texture coordinate corresponding to a position in a 2-D image texture. OpenGL-optimized graphics hardware is used to efficiently compute particle samples in real time. The system is evaluated on the dataset captured from drivers with different ages, races, and genders in both daytime and nighttime in the driving environment.

2.2.3 Hybrid Methods

Lee et al. [82] proposed a real-time driver's gaze zone estimation method based on head orientation: yaw and pitch. Shape-based and texture-based methods are combined in this method. In order to estimate a driver's head yaw, shape features (i.e., the left border, the right border, and the center of the driver's face) are extracted without the eye position. Ellipsoidal face model, instead of the cylindrical model, is then applied to model the face

based on these features, presented in Figure 2.12. The yaw angle based on the ellipsoidal model θ_{y_e} is denoted by:

$$\theta_{y\_e} = arcsin\left(\frac{c_R - x_c}{R}\right), R = (\alpha + \beta)\, r, \tag{2.2}$$

where R is the radius of rotation, and x_r, x_r, and x_c are obtained from the input image.

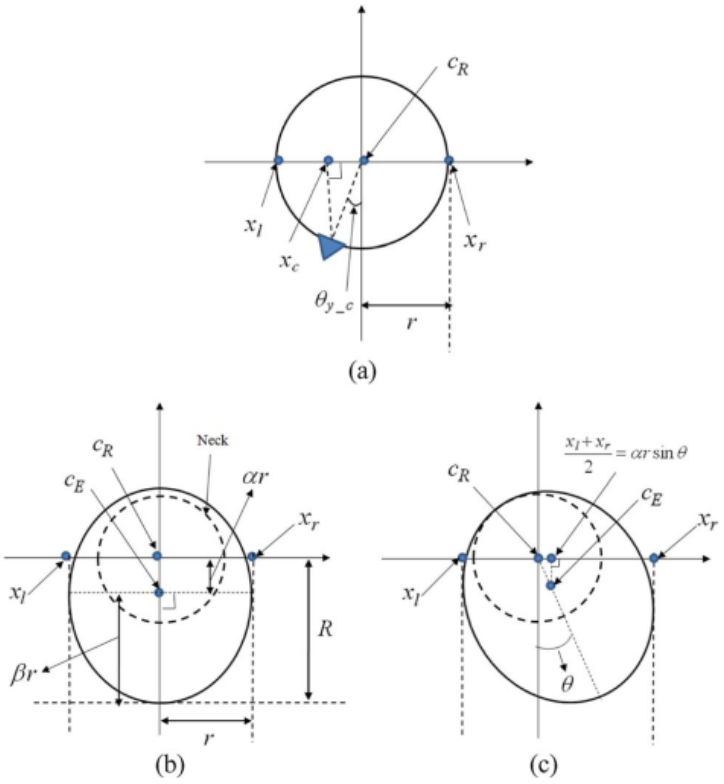


Figure 2.12: Face models: (a) Cylindrical face model. (b) Ellipsoidal face model. (c) Ellipse rotated counterclockwise by θ. Taken from [82].

To estimate the driver's head pitch, new texture features are proposed, which are the normalized mean and standard deviation obtained from the histogram of the horizontal edge projection. The driver's head pitch is estimated by passing these features in a trained SVR. Experiment results from 200000 images show that the root mean square errors of the estimated yaw and pitch angles are below seven degrees under both daylight and nighttime conditions.

In conclusion, compared with appearance-based methods, shape-based methods are intuitive and simple to implement. The challenge of shape-based methods is that they require the robust and accurate localization of facial features. Since the facial feature detection techniques are accurate enough for head estimation and multiple cameras can be used, these challenges can be released to some extent. Moreover, facial-landmark-based head pose detection can provide more information for further analysis, such as eye information

for more accurate gaze estimation. Especially in driver distraction and drowsiness analysis, eye, mouth and nose information are essential cues.

2.3 Eye Detection

As we mentioned before, eye information combined with head pose decides the accurate gaze direction of a human, which is vital to human-computer interaction system. Moreover, the state of the eye can also be used in driver's drowsiness analysis. The eye is often characterized by pupil/iris and the eyelids, and the position of the eye is commonly represented by the pupil or iris center. The appearance of the eye is hugely affected by ethnicity, head pose, viewing angle, iris position, illumination condition, occlusion of the eye by the eyelids, and so on. In addition, the appearance has large changes even for the same individual with a little variation in view angles, demonstrated in Figure 2.13. Despite active research in recent years, eye detection and tracking remains a very challenging task because of these issues.

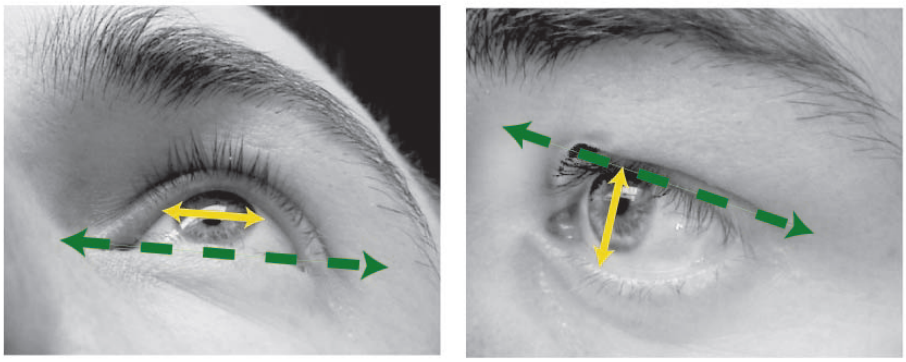


Figure 2.13: The examples of different view angle for the same individual. Taken from [69].

Shape-based, appearance-based and hybrid methods are three prominent approaches for eye detection, being either rigid or deformable [69]. The shape-based methods make use of predefined eye shape model and surrounding structures, while the appearance-based methods make use of model directly constructed from the eye appearance, which may have no practical meaning. Hybrid methods combine feature, shape, and appearance approaches to exploit their respective advantages.

2.3.1 Shape-based Methods

Shape-based methods are based on the fact that the open eye is well described by its iris, pupil and exterior contours. Different methods apply different shapes to detect eyes. Models have been proposed to model the eye shape from simple ellipse [85] to more complex models utilizing more detailed and precise modeling information [148].

Kim and Ramakrishna [77], and Perez et al. [110] estimated the center of the iris as the eye position. The shape of iris contour or limbus is modeled as a simple ellipse. They

assumed that the coarse location of the face and eye region have already been known from other methods. Their methods are to get the precise location of iris center based on the known location of the eye. The threshold is applied to eye image intensity first. Next, edge detection methods such as Longest Line Scanning (LLS), Occluded Circular Edge Matching (OCEM), and Laplacian operator are applied to detect the pupil contours. Hough transform technique [106] is another well-known method for edge detection especially for the object has a circular and ellipsoidal shape such as iris contour. Kim et al. applied the threshold by assuming people have dark iris, which may not always be true. Moreover, the circular or ellipsoidal shape constraint often works well only in the near-frontal face.

In the naturalistic environment, the shape of the eye changes significantly because of free head movement, facial expression, and eye state (open/closing). Simple ellipse is hard to perform in such condition. Figure 2.14 gives some examples of non-ellipse iris shape.

Figure 2.14: The examples of non-ellipse iris shape. Taken from [129].

Yuille et al. [148] proposed a very typical example of a complex shape-based method based on deformable eye model. The feature of the human eye is described by a parameterized eye template. The parameterized eye template proposed by Yuille et al. is illustrated in Figure 2.15.

The deformable eye model is composed of the four features:

1) The radius r of iris contour, centered at point $\overrightarrow{x_c}$.

2) The exterior contour of the eye, composed of two separate parabolic sections. It utilizes eye center $\overrightarrow{x_e}$, width $2b$ and θ to represent the orientation of eye.

3) Two points, corresponding to the centers of the whites of the eyes, represented by $\overrightarrow{x_e} + p_1(cos\theta, sin\theta)$ and $\overrightarrow{x_e} + p_2(cos\theta, sin\theta)$.

4) The areas between the exterior contour and the iris contour, namely the whites of the eye.

The model is denoted by $\vec{g} = (\overrightarrow{x_c}, \overrightarrow{x_e}, p_1, p_2, r, a, b, c, \theta)$, with a total of eleven parameters. An energy function is defined as a combination of terms due to the valley, edge, peak, image, and internal potentials. The model is then matched dynamically with the eye image by adjusting eleven parameter values to minimize the energy function, thereby

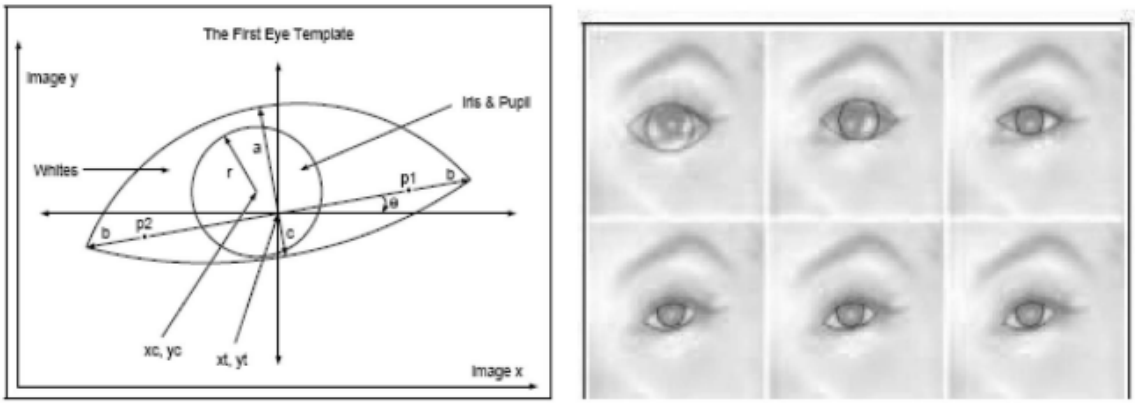


Figure 2.15: (a) A deformable template (b) Eye template at different times during the minimization. Taken from [148].

deforming itself to obtain the best match. A search strategy based on the steepest descent is applied to find the most salient components of the eye in an order. Iris is detected by valley potential, and so on. Experiment results show that the method is sensitive to the initial position of the model, and they also modeled the iris contour as a circular shape.

Lam el al. [80] extended the method proposed by Yuille et al. making use of eye corner position. The contour of the head is detected first by "snake" method. Next, the approximate position of the eyes are estimated using average anthropometric measures. They proposed an eye corner detection approach to detect the eye corners, which are applied to initiate the eye template. Experimental results show that there is about 40% improvement in average execution time when eye corner position is applied to initiate eye template.

Colombo and Bimbo [39] proposed another deformable eye model with six deformation parameters. The eye model is composed of two semi-ellipses that share the same major axis. The six parameters are $e_1 - e_6$ shown in Figure 2.16, namely the common major axis (e_1), the two minor axes (e_2, e_3), the ocular center's image coordinates (e_4, e_5), and the common orientation (e_6).

2.3.2 Appearance-based Methods

Appearance-based methods detect the eyes directly based on the statistics of eye intensity. Appearance-based methods can be divided into two categories: image template-based, and holistic methods. Image template-based methods utilize both spatial and intensity information of each pixel while holistic methods only consider the intensity distribution instead of spatial information [69]. Image template-based methods often need preprocessing such as scaling and rotation to make the detected eyes have similar size and orientation as the template images. Holistic methods extract efficient information implicitly from the

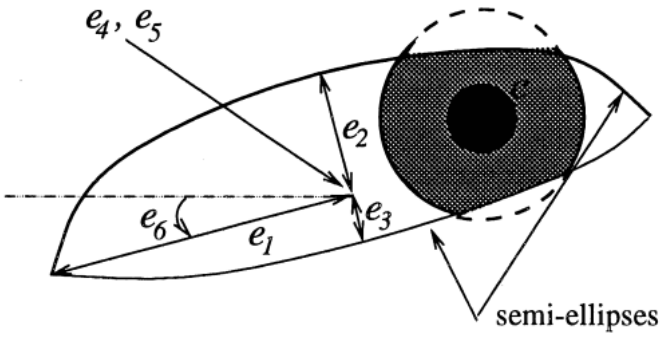


Figure 2.16: Reference template for left and right eyes. Taken from [39].

entire image appearance to solute appearance variations. These efficient information is then passed into a classifier or regression model to detect the eye.

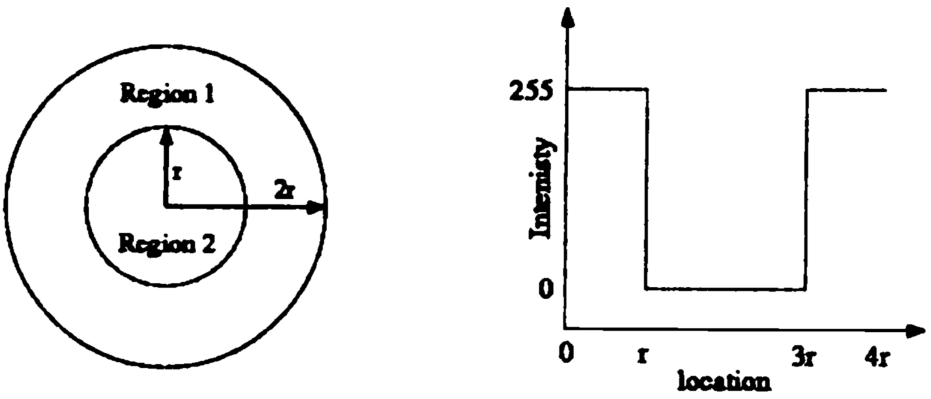


Figure 2.17: The blob geometry and across-section of the intensity model for an ideal valley. Taken from [67].

Hallinan [67] proposed an intensity model combined with blob operator shown in Figure 2.17, which is composed of two regions, corresponding to dark iris and white sclera in the eye. For each region, the frontal view of the ideal eye has uniform intensity, but in practice, the histogram of intensity is affected by Gaussian noise. Therefore, the author makes use of statistical measures to match the ideal template with eye image to solve intensity variation issues. Before the matching procedure, the input image is smoothed by a non-linear diffusion operator [105] to enforce the region to be constant piecewisely.

2.3.3 Hybrid Methods

Hybrid methods take advantage of both shape-based and appearance-based methods to reduce their respective disadvantages.

Local patch-based methods have shown promising results in object detection, recognition recently [129]. These methods combine local patch appearance with shape model. Algorithms such as Active Appearance Model (AAM) [41], Constrained Local Model (CLM) [46] have been applied to detect facial landmarks including eye contour, nose and mouth contour. These local patch-based methods can solve head rotation, and illumination change issues more efficiently. These methods can be used to detect eye contour and provide accurate location of the eye. The iris center detection is then applied based on the accurate location of the eye and eye corners.

Based on eye contour landmarks obtained from AAM, Ishikawa et al. [127] proposed a two-part iris detector: template matching and edge-based iris refinement. Two templates are applied to estimate the initial location and radius of iris. The black disk template is matched with the intensity image while the ring template is matched with the vertical edge image. The scale of AAM is applied to obtain the radius of both templates. The estimate of iris center is then refined by an edge-based method iteratively until the estimate of the iris center converges. Edges are detected first, and then an ellipse is fitted to the detected edges to refine iris center.

Based on eye contour landmarks captured from CLM, Tawari et al. [129] proposed an iris detection algorithm based on regression models and the histogram of oriented gradients (HOG) feature. A sequence of regression matrixes $S = (R_1, ..., R_K)$ are trained offline and applied on eye region to update the estimation location iteratively by reducing the residual errors between the estimated iris position and the ground-truth. This method is based on the known landmarks of the eye to get the eye region and initial estimation of iris center.

Li et al. proposed [85] a hybrid eye-tracking algorithm that combines feature-based and model-based methods, called "starburst". Infrared videos are applied for eye-tracking, and captured from a cheap eye-tracking device mounted on object's head. The algorithm is to detect the pupil contour and center, and corneal reflection. The corneal reflection is removed through thresholding before pupil detection. First, the method detects the pupil based on feature points extracting strategy proposed. The feature points are detected iteratively until the geometric center of feature points converges. The center is called start point and is manually determined or set as the center of infrared eye image. The iteration consists of two stages. In stage one, the intensity derivatives along 18 rays extending radially away from start point, are independently evaluated pixel by pixel. The derivative evaluation along each ray stops until a threshold $\phi = 20$ is reached, and a feature point is defined at that pixel. In stage two, calculate intensity derivative along the ray from each feature point obtained from stage one to start point. The derivative calculation for each feature point stops when the threshold is reached, and a new feature point is placed at that pixel. After stage two, the start point is changed to the geometric center of all feature points. Finally, an ellipse model is applied to fit these candidate feature points to find the pupil contour. The Random Sample Consensus (RANSAC) algorithm is utilized for model

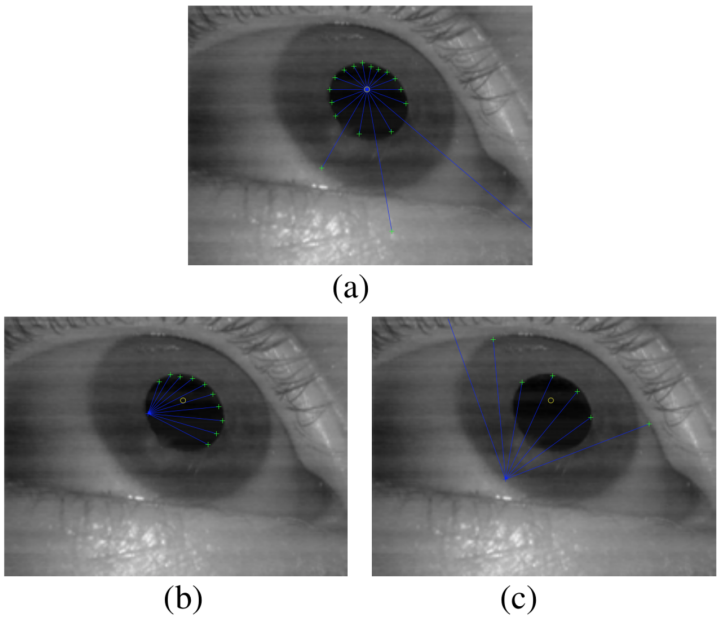


Figure 2.18: Feature detection. Taken from [85].

fitting [56] instead of least-squares fitting method [152] used commonly. The proposed starburst algorithm detects pupil more accurately than pure feature-based methods do and spends much less time than pure model-based methods.

2.3.4 Other Methods

In addition to categories described in the previous section, there exist some other methods which are difficult to classify.

Some methods make use of characteristics of the infrared image, such as bright/dark pupil effect. When an infrared light source is placed near the optical axis of the camera, the eye image shows bright pupil, because the ray near optical axis enters the pupil and will be reflected by the retina and captured by the camera. Thus, the pupil is bright in the image. When the infrared light source is placed away from camera optical axis, the ray reflected by retina cannot be captured by the camera. Thus, the pupil is dark in the image. Figure 3.8b shows the bright/dark effect of the pupil in the infrared image.

The pupil can be detected quickly and robustly by detecting the dark and bright pupil changes dynamically. In a simple situation, preprocessing such as face detection is not needed in such method because most of the time only the pupil has such bright/dark effect and the background can be removed easily.

Tomono et al. [132] proposed a system composed of a CCD camera and two near infrared light sources. The two infrared light sources have different wavelengths, and are placed near and away from camera optical axis respectively, to generate bright and dark pupil effects simultaneously. The captured data is filtered to obtain intensity data of two different

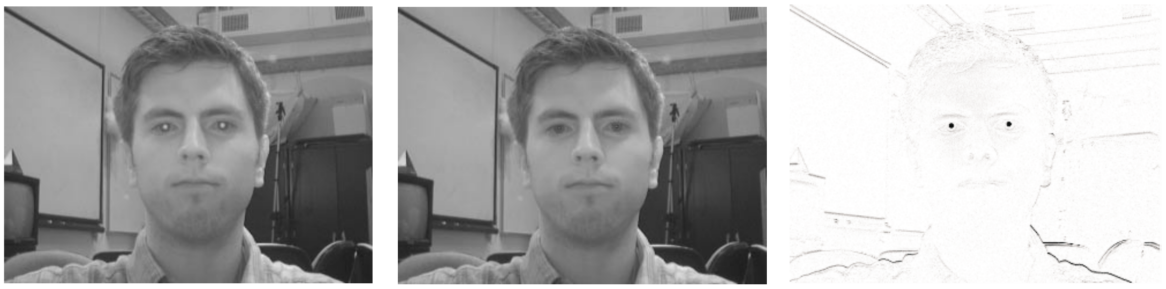


Figure 2.19: Bright/dark pupil effect images and their subtract image. Taken from [70].

illuminations. Two images with the bright pupil and dark pupil effect are captured. The pupil is then extracted from the subtraction of the two images.

However, in a more complex environment, issues such as eye closure/occlusion, potential interference from the object which has similar pupil effect, and external illuminations will make these methods work poorly. Therefore, pupil-effect-based methods are combined with other methods.

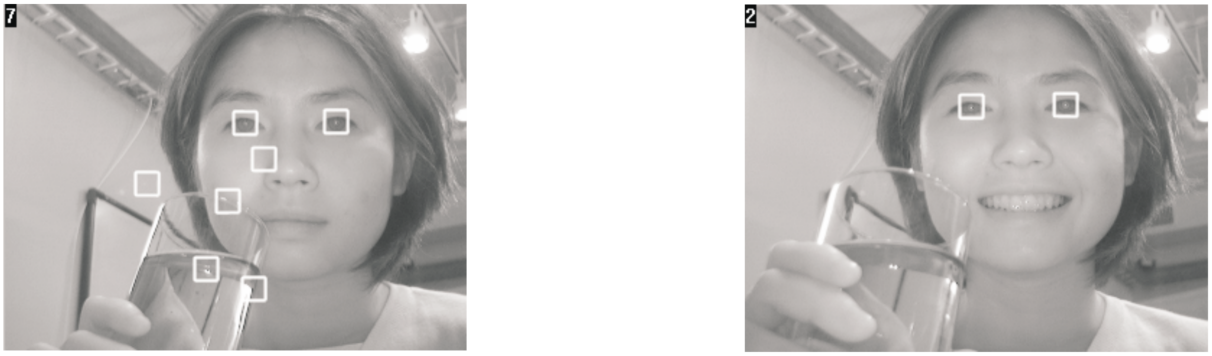


Figure 2.20: Tracked candinates and results after classification. Taken from [70].

Haro et al. [70] also made use of bright/dark pupil effect the same way as method proposed by Tomono et al. However, they considered a more complicated situation where some reflective moving objects exist and can be confused as pupil effect, such as a glass of water, shown in Figure 2.20. Thus, the pupil effect method is combined with the appearance-based method to find out the pupil from a set of candidates. The classifier is trained from a dataset containing positive images with a pupil and negative images without a pupil. Probabilistic principal component analysis (PPCA) [131], [93] is applied to reduce the dimension of intensity vector and obtain the probability of a candidate region to include the pupil. The probability information is then combined with temporal information by considering the estimate results of Kalman tracker, which models the movement of pupils.

Zhu et al. [156] proposed a real-time and robust method for eye tracking under variable and realistic lighting circumstance and various face orientations. The pupil-effect-based

method is combined with SVM-based object recognition method and mean-shift-based object tracking method to solve issues in the real world.

In this subsection, we reviewed different methods applied in eye detection. Deformable shape-based methods can model eyes well, but they are usually sensitive to initialization and can not handle various head pose and eye occlusions well. Moreover, they need higher image contrast and computational consumption. Compared with holistic appearance-based eye detection methods, local patch-based methods are more discriminative, more robust to various head pose and illumination. Local patch-based methods combine local patch appearance and shape model together. Instead of detecting eye only, they are usually used to detect facial landmarks. These facial landmarks include eye contour, mouth contour, face contour, and nose. Thus, eye contour can be achieved and is combined with iris detection method further to get a whole description of the eye. In addition, other information can be applied to head pose analysis and drowsiness analysis. The active infrared illumination approaches, which make use of pupil effect or combine pupil effect with other methods, are effective in the room environment. However, in the realistic driving environment, they need a particular setup of two infrared sources with the different wavelengths, and the outdoor sunlight forms more illumination interference. Therefore, they are more suitable for the indoor environment or the outdoor environment at night.

2.4 Facial Landmark Detection

Facial landmark detection techniques are applied to many computer vision applications, such as face recognition, face tracking, head pose estimation, face animation, and 3D face modeling [36]. It is a very challegeing task because of the non-rigid shape of human faces. Facial expressions, illumination variation, head pose changes, partial occlusions, and face difference among humans in both appearance and shape are all critical issues that make person-independent facial landmark detection a difficult problem. Existing methods can be classified into three categories: deformable model-based, regression-based, and other methods [143].

Deformable model-based approaches are the most popular and have achieved great performance in facial landmark detection and tracking [62]. Traditional models such as CLM and AAM are all possible deformable models. These methods combine the geometry feature and appearance feature of a face. Usually, a parametrized shape model is used to model the face shape. The face appearance is utilized either holistically or locally. Methods based on AAM and 3D Morphable Model (3DMM) [15] extract features from the whole face appearance holistically. Methods based on CLM and Active Shape Model (ASM) [42] extract the respective local appearance feature around each feature landmark position. Thus, each feature has its own appearance model. The facial landmark positions are detected by fitting the deformable model in the given image. Usually, an initial estimation of the model points in an image is obtained from detected face region. Thus, the face shape is refined guided by the appearance around currently estimated landmarks, holistically or locally.

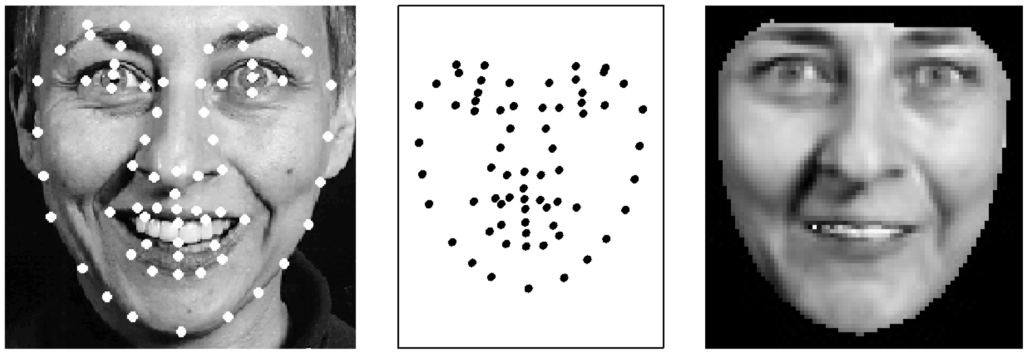


Figure 2.21: Left: an annotated training image. Right: corresponding shape-free patch. Middle: facial landmark locations. Taken from [41].

Cootes et al. [41] proposed a facial landmark detection method based on AAM. A linear shape model called Point Distribution Model (PDM) [42] is used to model the shape of a face, and eigen-analysis is utilized to build a linear texture model from horizontal scanned intensity vector. The texture model is trained from annotated images. Figure 2.21 shows an example in the training set, where important facial landmark locations are annotated manually on the original face image, and corresponding shape-free patch is extracted. The training face image is warped to make the landmark points match the mean shape and to generate the shape-free patch. Eigen-analysis is applied on the shape-free patch to build texture model.

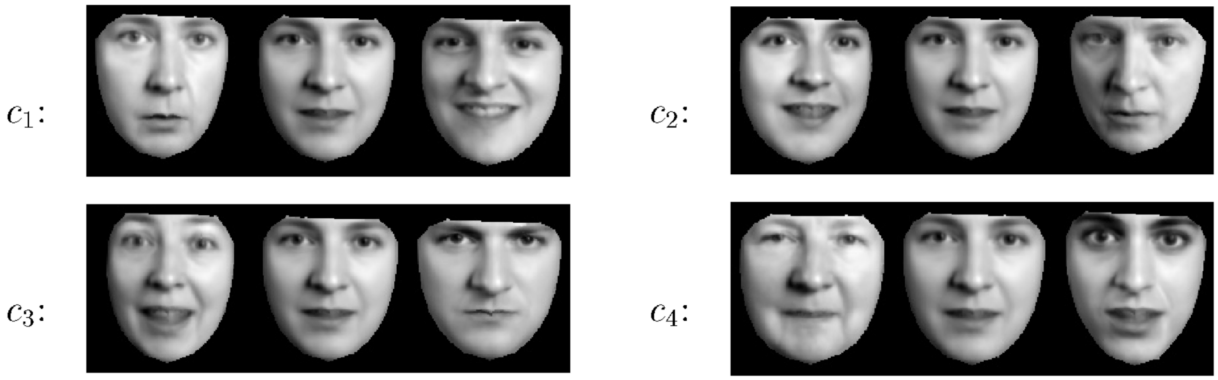


Figure 2.22: Effect of varying first four facial appearance model parameters, c1-c4 by ± 3 standarad deviations from the mean. Taken from [41].

The variation of face texture is based on the mean face texture. Figure 2.22 shows the synthesized face texture generated by the model parameters. The objective of AAM fitting is to minimize the difference between the testing image and the synthesized image. Instead of using simple linear regression to optimize, the method they proposed utilizes the learned

correlation between errors in model parameters and the resulting residual texture errors. Experiment results show that the method can converge rapidly and reliably. Because of the flexible and simple framework of AAM, there also exists a lot of extensions and variation in AAM based facial landmark detection methods, such as [133], [151], [73]. These extensions try to improve the method in terms of the efficiency, discrimination, and robustness when applied to practical applications [143].

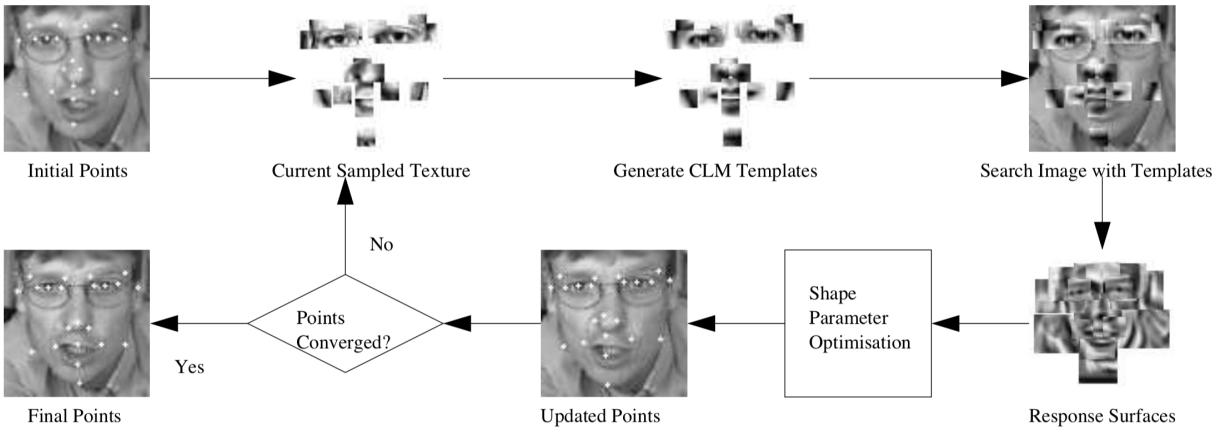


Figure 2.23: CLM search algorithm. Taken from [46].

Cristinacce and Cootes [46] proposed the CLM for facial landmark detection. They utilized the same shape model (i.e., PDM) as AAM [41] to model the parameterized face shape. However, the appearance model is built from the local patch around each landmark locations instead of the whole face region. Figure 2.23 demonstrates the fitting procedure. The shape model is initialized based on the previous frame or the face region. The texture patch around each landmark is sampled from the given image. A set of appearance model/template is generated based on current estimation. The templates are correlated with the given image to generate response surfaces. An optimization method is used to update shape parameters to maximize the sum of responses. Landmark positions are obtained iteratively. Experiment results show that CLM search algorithm is more robust and more accurate than the original AAM search method. There also exist a lot of extensions and variations in CLM model-based facial landmark detection methods, such as [120], [11], [40]. These extensions and variations combine different shape models, appearance models, local detectors and fitting strategies with a similar framework.

Baltrusaitis et al. [11] proposed a CLM-based method for robust facial landmark detection in the wild, called Constrained Local Neural Field (CLNF). Their contribution and novelties are in the novel local detector (i.e., Local Neural Field (LNF)) and fitting strategy (i.e., Non-Uniform Regularised Mean-Shift (NU-RLMS)) proposed. LNF with the edge feature is used as local detector to evaluate the probability of a landmark being aligned at a particular location. Different from traditional linear SVM-LR, LNF can better model the non-linear relationship existing in the wild environment and also makes use of spatial relationship. NU-RLMS method is utilized as fitting strategy and leads to more accurate

and reliable fitting because this fitting strategy takes the reliability of each local detector into consideration. Experiment results on a number of publicly available datasets show that their method performs better than state-of-art approaches when performing landmark detection and tracking in unseen lighting conditions and the wild.

Regression-based methods learn a regression function that directly maps image appearance (feature) to the target output (shape) shown in Equation 2.3 [36], [143]:

$$\mathcal{M} : \mathbf{Feat}(\mathcal{I}) \mapsto \mathbf{x} \in \mathbb{R}^{2N}, \tag{2.3}$$

where $\mathbf{Feat}(\mathcal{I})$ is the appearance feature of image $\mathcal{I}$, and $\mathbf{x}$ is the shape. $\mathcal{M}$ represents the mapping from feature to shape.

Cao et al. [36] proposed an explicit shape regression method for facial landmark detection. They directly learned a vectorial regression function to find the landmarks from the appearance of a face. Instead of using a shape model to constrain the face shape, the constraint of face shape is learned implicitly. Two-level boosted regression, shape-indexed features, and a correlation-based feature selection method are proposed to learn regression models from a large training set. Boosted regression [61] combines the weak regressors in a cascaded manner. A sequence of weak regressors guide the update of face shape based on previously estimated shape and shape-indexed features. The regressors are aimed to minimize the alignment error regressively. Experiment results show their method has better accuracy and efficiency than many state-of-the-art methods on some challenging datasets.

Valstar et al. [134] proposed a novel approach based on Boosted Regression combined with Markov Networks, called BoRMaN. SVR is utilized as the weak regressor to learn a mapping between appearance around a point and the locations of the points. Markov Random Field is utilized to constrain the possible face shape by exploiting the constellations that facial landmarks can form. The joint probability of pairwise relations and the relative positions of two landmarks in a configuration is modeled by Markov Random Field to prevent unfeasible facial landmark location combinations. Experiment results demonstrate that their method can achieve good accuracy and robustness under conditions with illumination variation, moderate head pose, and occlusions caused by glasses. However, they did not mention how the method works under large head variation in the wild.

In addition to CLM-based, AAM-based, and regression-based methods, there are also many other methods for facial landmark detection which may make use of 3D deformable model [15], depth information [31], [10] and so on. Because of the popularity of deep learning in machine learning, Luo et al. [89] even applied deep learning algorithm in facial landmark detection. For a more systematic and detailed summary of facial landmark detection methods, please refer to [143]. From the survey of Wang et al. [143] CLM-based and AAM-based methods are the most popular and attract a lot of researchers to work on extending and improving CLM-based and AAM-based methods. Furthermore, regression-based methods became popular in recent years and can also achieve promising performance. However, regression-based methods focus more on the frontal or near-frontal images, while some CLM-based variations can obtain good performance in large head pose variation and hand-over-face gestures, which is more suitable for the realistic driving environment.

2.5 Multiclass Classification Methods

To estimate the gaze zone y_i from the feature vector $\mathbf{x_i}$ extracted from image $\mathbf{I}^i$, supervised classification method is applied. Supervised classification method is to learn a model $\mathbf{H}$ from a labeled training set $(\mathbf{x_i}, y_i)$ shown in Equation 2.4, and the model is further applied to classify unseen example.

$$y_i = \mathbf{H}(\mathbf{x_i}) \tag{2.4}$$

where $y_i \in \{1, ..., K\}$ is the discrete class label of i^{th} example, K is the overall class number, $\mathbf{x_i} \in \mathbb{R}^N$ is the vector data of i^{th} example.

Classification methods can be divided into two categories according to the number of classes to be classified. A binary classifier is used to classify two classes with $K = 2$ while multiclass classifier is used to classify three or more classes with $K \geq 3$. Various successful algorithms have been proposed for binary classification, and these binary classifiers can be extended to multiclass case naturally or by applying special strategies such as decomposing and hierarchical classification. Popular methods for binary classification include decision trees [34], [112], [147], neural networks [14], KNN [13], Naive Bayes classifiers [115], and SVMs [44].

The decision trees, KNN, Naive Bayes algorithm can naturally solve binary or multi-class classification problems. Neural networks can be extended by adding multiple binary neurons in the output layer, and class label can be coded to the binary codeword by one-per-class coding and distributed output coding [52].

Instead of naturally extending these binary classifiers, the multiclass classification task can be decomposed into multiple binary classification tasks which can be efficiently implemented by binary classifiers [6]. The decomposing strategy includes One-Versus-All (OVA), All-Versus-All (AVA), Error-Correcting Output-Coding (ECOC), and Generalized Coding [6].

In One-Versus-All strategy, each class is discriminated from the other $K - 1$ classes by a binary classifier [114]. Thus, K binary classifiers are needed to generate negative and positive output. Only positive output will give information which class a new example belongs to, and only one binary classifier will have a positive output.

In All-Versus-All strategy, each class is discriminated against each other class by binary classifiers [59], [71]. Overall $\frac{\mathbf{K(K-1)}}{2}$ binary classifiers are required to classify between any two classes. Voting is applied to the results of these classifiers, and the class with maximum votes will be the label of the new example.

In ECOC strategy, a class is coded according to a binary matrix M shown in Figure 2.24. The K rows of M are codewords representing K classes, and N columns are utilized to train N binary classifiers. Hamming distance is applied to calculate the distance between the resulting codeword from the N binary classifiers and the codeword for each class. The class has minimum hamming distance will be the class of the input.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
Class 1	0	0	0	0	0	0	0
Class 2	0	1	1	0	0	1	1
Class 3	0	1	1	1	1	0	0
Class 4	1	0	1	1	0	1	0
Class 5	1	1	0	1	0	0	1

Figure 2.24: Example of 5 classes and 7 bit codewords. Taken from [6].

In Generalized Coding strategy, the binary code in ECOC is extended to the code taking three values $\{-1, 0, +1\}$ for each bit. Bit value $\{0\}$ then represents the case where the classifier will ignore that class.

Hierarchical Classification is another method to solve multiclass classification problem, which hierarchically divides the output space and arranges the classes in a binary tree shown in Figure 2.25 [6]. At each parent node, multiple classes are split into two child clusters by a simple binary classifier. Repeat the split procedure until a node contains only one class, and this node is called leaf node. For a new vector, it starts from the root and be classified continuously until it reaches a leaf node, the class of the new vector is labeled the same as class contained in that leaf node.

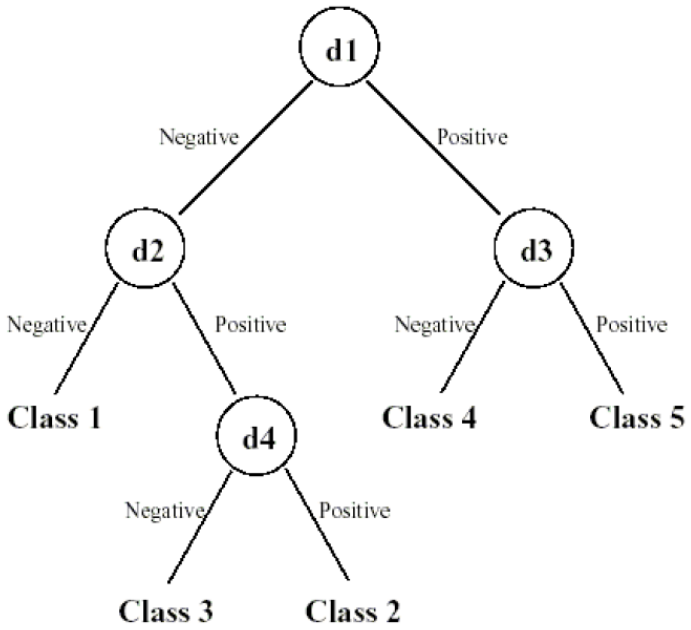


Figure 2.25: Example of hierarchical classification tree for 5-class classification. Taken from [6].

Ensemble methods including boosting and bagging (bootstrap and aggregation) [32] are usually used to generate multiple version of a classifier and aggregate these multiple

classifiers together to get a strong classifier. In bagging strategy, to take every single classifier's output into consideration, the aggregation makes a plurality vote, and the class that has the most votes will be the final classification result. The multiple version of a classifier is achieved by randomly select N samples with replacement from overall N samples in the training set for each classifier, which is also called bootstrapping. Thus, each classifier has similar but different dataset with each other for training. Experiment results show that bagging method achieves a substantial gain in accuracy [32].

Based on bagging decision trees, applying bagging on decision trees classifier, RDF improves the method by adding another strategy, namely random feature selection. Instead of using all N features in vector $\mathbf{x_i}$, only m, $m < N$ features are randomly selected to train a decision tree which reduces the correlation between each tree and put more randomness in the overall classifier. Thus, each tree is trained from the values of a random vector sampled independently. Experiment results show that RDF is more robust with respect to noise and obtain as good accuracy as Adaboost proposed by Freund and Schapire [58].

In this section, we described different multiclass classification methods. Among all multiclass classification methods, RDF has been successfully and widely applied to various problems [123]. It has become a major data analysis method, which performs better compared to many standard methods [72], [51]. The popularity of RDF is based on the fact that it can be applied to a wide range of classification problems, even if they are nonlinear and involve complex high-order interaction effects [123]. In addition, RDF has been implemented by many open libraries such as OpenCV [3].

2.6 Summary

This chapter reviewed five components, namely video-based gaze estimation, head pose detection, iris center detection, facial landmark detection, and multiclass classification techniques.

For gaze estimation, although the feature-based methods utilizing pupil-glint displacement are simple, having no need of complicated calibration, and relatively accurate, they are limited to particular applications because they perform poorly under significant head movement. The 3D model-based gaze estimation systems can tolerate natural head movements, but these methods usually require a complicated one-time system or geometric calibration. On the other hand, appearance-based approaches are not based on known parameters from feature extraction and many calibration parameters. They extract these information implicitly. Moreover, they can also achieve free head movement by combining head pose estimation. However, they need extensive data for training to extract those useful information implicitly.

Compared with the person-dependent method, the person-independent method has more generalization to new individuals, but the accuracy will drop as a consequence. Person-dependent methods usually achieve higher accuracy. In addition, the naturalistic driving environment always presents unique challenges to video-based gaze estimation and tracking systems. In an automobile, continuous lighting changing conditions, resulting

in heavy shadows and high illumination variability, and unstable capture of data, caused by jolts, all add more difficulties on gaze estimation task. Furthermore, performing in real time or near-real time is another consideration for gaze estimation in an ADAS. The results in the stationary laboratory with stable illumination have higher proficiency than those in the wild. However, these methods may work poorly in the driving environment. Therefore, specific approaches need to be designed for ADASs to overcome these unique difficulties.

For head pose estimation, compared with appearance-based methods, shape-based methods are intuitive and simple to implement. The challenge of shape-based methods is that they require the robust and accurate localization of facial features. Since the facial feature detection techniques are accurate enough for head estimation and multiple cameras can be used, these challenges can be released to some extent. Furthermore, facial-landmark-based head pose detection can provide more information for further analysis, such as eye information for more accurate gaze estimation. Especially in driver distraction and drowsiness analysis, eye, mouth and nose information are critical cues.

For eye detection, deformable shape-based methods can model shape of eyes well, but these methods are usually sensitive to initialization and can not handle various head pose and eye occlusions well. Compared with holistic appearance-based eye detection methods, local patch-based methods are more discriminative, more robust to various head pose, and various illumination. Instead of detecting eye only, local patch-based methods are usually used to detect facial landmarks. These facial landmarks include eye contour, mouth contour, face contour, and nose. Eye contour can be achieved and combined with iris detection method further to get a whole description of eyes. Moreover, other information can be applied to head pose analysis and drowsiness analysis. The active infrared illumination approaches are effective in the room environment but not in the realistic driving environment, because they need a particular setup of two infrared sources with different wavelengths, and the outdoor sunlight forms more illumination interference. Thus they are more suitable for the indoor environment or outdoor environment at night.

For facial landmark detection, CLM-based and AAM-based methods are the most popular and attract a lot of researchers to work on extending and improving these methods. In addition, regression-based methods became popular in recent years and can also achieve promising performance. However, regression-based methods focus more on the frontal or near-frontal images, while some CLM-based variations can obtain good performance in large head pose variation and hand-over-face gestures, which is more suitable for the realistic driving environment.

For multiclass classification, RDF is the most popular method and has been successfully applied to a wide range of classification problems. Besides, RDF has been implemented by many open libraries such as OpenCV.

Chapter 3

System Framework

To have a real-time gaze zone estimation system in the naturalistic driving environment, the framework of our system is proposed, as shown in Figure 3.1.

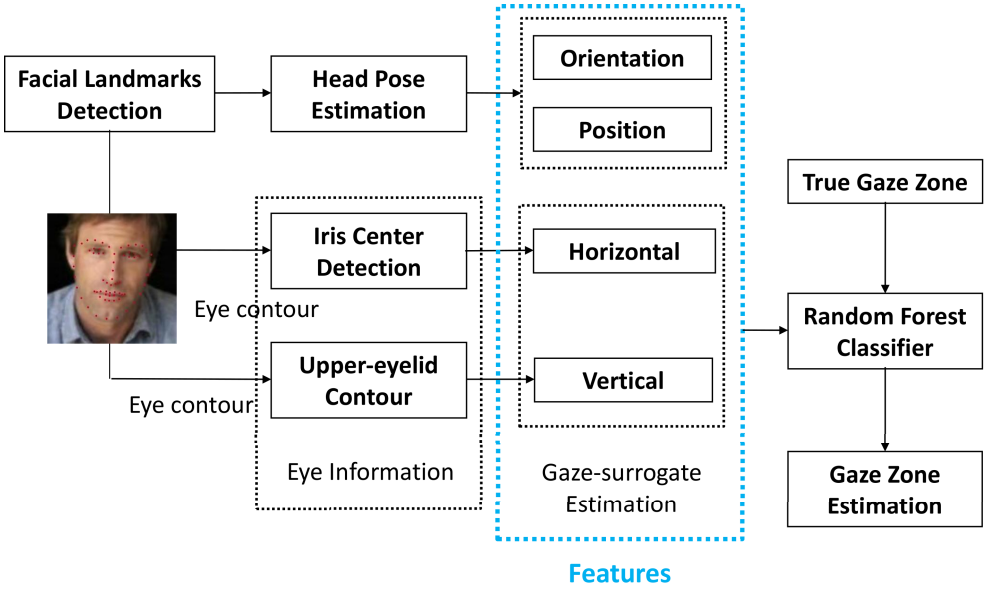


Figure 3.1: Proposed framework for driver's gaze zone estimation in the realistic driving environment

The framework is composed of five main components: Facial Landmark Detection, Head Pose Estimation, Iris Center Detection, Upper Eyelid Information Extraction, and Gaze Zone Estimation. First, Constrained Local Neural Field (CLNF) [9] method is used to detect facial landmarks, which consist of 68 red points describing the eyes, nose, mouse, and exterior face contour of a person, as shown in the face image of Figure 3.1. Next, we estimate the head pose, detect iris center, and extract upper eyelid information based on the facial landmarks obtained previously. Furthermore, we extract horizontal and vertical gaze features. These features will finally function as the input of the Random Decision Forest (RDF) classifier to obtain the gaze zone estimation. This system can work in both

daytime and nighttime since color data is captured in the daytime and infrared data is captured in the nighttime. In this chapter, we will follow the framework proposed, and discuss these five parts in detail in the following sections. Table 3.1 shows a summary of the mathematical symbols applied in this chapter.

Section	Symbol	Description
Section 3.1	$\mathbf{p}$	CLM model parameter
	$\mathbf{x}_i$	i^{th} coordinate
	$\mathcal{I}$	an image
	l_i	a discrete random variable
	$\mathbf{X}$	PDM instance
	$\bar{\mathbf{X}}$	mean position of facial landmarks in the 3D object frame
	$\mathbf{\Phi}$	deformation coefficient matrix of PDM
	$\mathbf{q}$	PDM parameter
	$\mathbf{\Lambda}$, $\tilde{\mathbf{\Lambda}}^{-1}$	covariance matrix
	s	the scaling factor
	$\mathbf{t}$	translation factor
	$\mathbf{R}_{2D}$	the first two rows of a full rotation matrix
	$\mathbf{w}$	a vector representing an axis-angle rotation
	θ	the amount of rotation in radians
	$\hat{\mathbf{n}}$	rotation axis of axis-angle rotation
	I	identity matrix
	$\boldsymbol{\eta}, \eta_0$	the coefficients of LR
	$\boldsymbol{\alpha}, \boldsymbol{\Theta}, \boldsymbol{\beta}, \boldsymbol{\gamma}$	LNF model parameters
Section 3.2	$\boldsymbol{p}_i^{Obj}$, $\boldsymbol{p}_i^{Img}$	landmarks in 3D object frame and 2D image frame
	$\boldsymbol{K}$	the intrinsic matrix of camera
	$\boldsymbol{R}$	the rotation matrix
	$\boldsymbol{t}$	the translation
	$\boldsymbol{r}_i$	current residual vector
Section 3.3	$\boldsymbol{p}_k, \boldsymbol{p}_{k-1}$	the currently and previously estimated locations
	$\boldsymbol{R}_{k-1}$	regression matrix
	$\mathcal{I}$	an image
	$\boldsymbol{S}$	a set of regression matrices
	K	the iteration number
	$Gmag, Gdir$	the magnitude and direction of gradient
	G_x, G_y	horizontal and vertical gradients
	$\boldsymbol{p}_*^i, \boldsymbol{p}_0^i$	true and initial estimation of iris center locations
Section 3.4	p_k	the probability of k^{th} category
	K	the total number of possible categories for an attribute
	C_k	the amount of k^{th} category
	T_i	the i^{th} potential split point

Table 3.1: A summary of the mathematical symbols

3.1 Facial Landmark Detection

Among the various versions of CLM method, CLNF is chosen to be our facial landmark detector. CLNF is the name given to a CLM instance that uses LNF local detectors and NU-RLMS fitting method, proposed by Baltruvsaitis [9]. This variation is chosen because it performs better than state-of-art approaches when performing facial landmark detection and tracking in unseen lighting conditions and the wild. This CLM instance will be presented later.

CLM-based facial landmark detection approach includes three main parts: a statistical shape model, local detectors, and a CLM fitting method. Shape model and local detectors both are trained offline and then used for online facial landmark detection. These three parts will be presented in the following subsections.

CLM fitting is to iteratively find the model parameter, $\mathbf{p}$, minimizing the misalignment of all landmarks. In a probabilistic way, the error function or the energy function can be expressed in Equation 3.1 [120],

$$\varepsilon\left(\mathbf{p}\right) = \mathcal{R}\left(\mathbf{p}\right) + \sum_{i=1}^{n} \mathcal{D}_i\left(\mathbf{x}_i; \mathcal{I}\right), \tag{3.1}$$

where $\mathcal{R}\left(\mathbf{p}\right)$ is the regularization term that penalizes complex or unlikely deformation, and $\mathcal{D}_i\left(\mathbf{x}_i; \mathcal{I}\right)$ stands for the amount of misalignment the i^{th} landmark is experiencing at coordinate $\mathbf{x}_i$ in the image $\mathcal{I}$ (data term). Model parameter $\mathbf{p}$ decides the location of $\mathbf{x}_i$ according to the shape model chosen. The form of Equation 3.1 is related to the form of statistical shape model and local detectors chosen, which will be described later.

An alternative way to interpret CLM fitting purpose is to find out the parameter $\mathbf{p}$ which maximizes posterior probability in condition that all landmarks are aligned in image $\mathcal{I}$, shown in Equation 3.2 [120],

$$\begin{aligned} p\left(\mathbf{p} \mid \{l_i = 1\}_{i=1}^{n}, \mathcal{I}\right) &= \frac{p(\mathbf{p}) p(\{l_i = 1\}_{i=1}^{n}, \mathcal{I}|\mathbf{p})}{p(\{l_i = 1\}_{i=1}^{n}, \mathcal{I})} \\ &\propto p\left(\mathbf{p}\right) \prod_{i=1}^{n} p\left(l_i = 1 \mid \mathbf{x}_i, \mathcal{I}\right), \end{aligned} \tag{3.2}$$

where $l_i \in \{1, -1\}$ is a discrete random variable representing whether the i^{th} landmark position is aligned or misaligned, and $p(\mathbf{p})$ is the prior probability of parameter $\mathbf{p}$. $p(l_i = 1|\mathbf{x}_i, \mathcal{I})$ denotes the probability of a landmark being aligned at image position $\mathbf{x}_i$. Moreover, it is assumed that all the local detectors are conditionally independent of each other, which is different from the method that exploits appearance model holistically. The joint probability of all landmarks being aligned at position $\mathbf{x}_i$ in image $\mathcal{I}$ is represented as $\prod_{i=1}^{n} p\left(l_i = 1 \mid \mathbf{x}_i, \mathcal{I}\right)$.

In the above equation, the prior probability $p(\mathbf{p})$ can be obtained from the shape model chosen, and the local alignment probability $p(l_i = 1|\mathbf{x}_i, \mathcal{I})$ can be computed from the response map created by local detectors for each landmark independently.

Equation 3.2 and Equation 3.1 will be equivalent if the regularization and misalignment error take the following forms [9]:

$$\mathcal{R}(\mathbf{p}) = -ln\{p(\mathbf{p})\} \tag{3.3}$$

$$\mathcal{D}_i(\mathbf{x}_i;\mathcal{I}) = -ln\{p(l_i = 1 \mid \mathbf{x}_i, \mathcal{I})\} \tag{3.4}$$

There are many general optimization approaches that can be applied to minimize the error function in Equation 3.1, such as Newton method [120]. However, in consideration of speed, accuracy, and stability, they are not suitable in practice because of slow convergence and unstability [120]. Therefore, an optimization method designed for CLM fitting is required for the real-time, accurate and stable facial landmark detection.

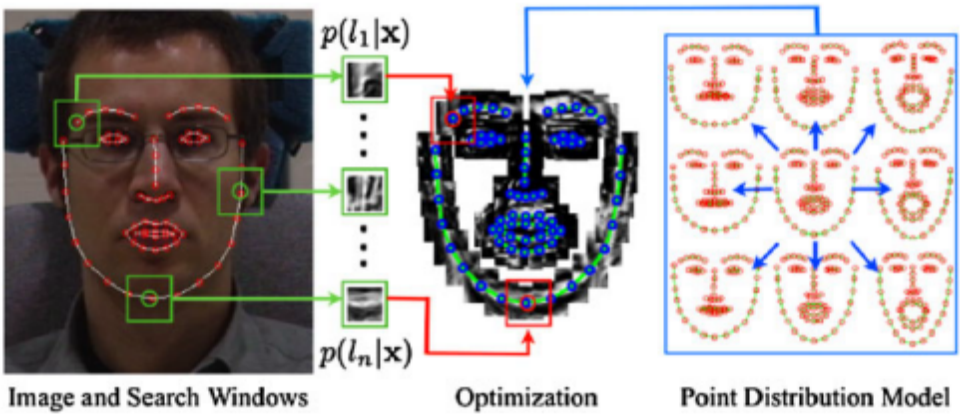


Figure 3.2: CLM fitting diagram and its two steps: (1) obtain local detector response maps $\{p(l_i = 1|\mathbf{x}_i, \mathcal{I})\}_{i=1}^{n}$ from the intresting area around every currently estimated landmark locations, and (2) update facial shape model parameters to optimize point alignments under constraint of shape model. Taken from [120]

Figure 3.2 shows a general diagram of CLM fitting process. Two steps are included in the diagram, and they will be performed iteratively until some conditions (e.g., maximum iteration steps) are met. Firstly, the approach performs an exhaustive local search around the currently estimated landmark locations respectively. Thus the response maps will be computed around each estimate of landmark locations by local detectors. Secondly, an optimization strategy will be employed over these response maps. The optimization strategy often applies an approximation method on the response maps instead of using the maps directly. The approximation methods include Isotropic Gaussian Estimate [42], Anisotropic Gaussian Estimate [104], [144], A Gaussian Mixture Model Estimate [64], and Kernel Density Estimate [120].

3.1.1 Shape Model

Shape model describes the possible deformation of a face. It can also constrain the joint motion of landmarks, guide CLM fitting procedure, and function as the prior probability

$p(\mathbf{p})$ of the model parameter $\mathbf{p}$ because the shape model can evaluate the possibility of the face shape. Both head pose (i.e., orientation and location) and facial expression need to be described by the shape model, and they are also called global and local parameters respectively.

The typical choice of facial landmarks is shown in Figure 3.3, which models the outline of the face and the critical facial features for emotion recognition, including eyebrows, eyes, nose and mouth.

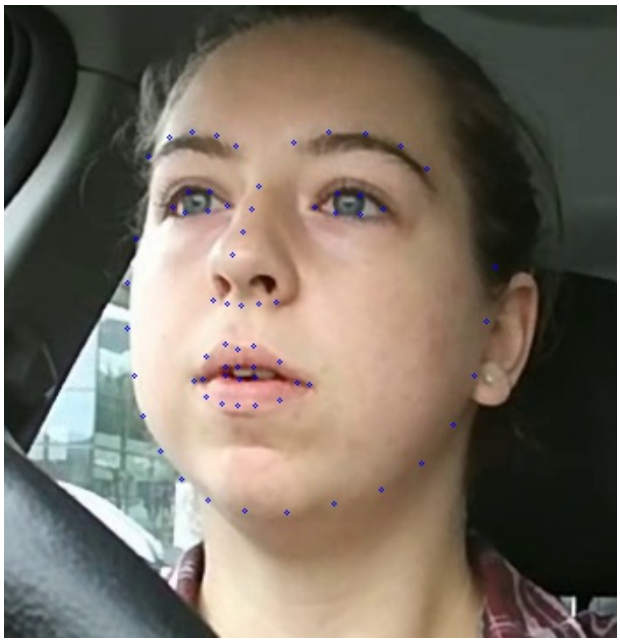


Figure 3.3: Facial landmarks containing 68 points.

- **Deformable Shape Model**

The most popular model used in deformable shape models is Point Distribution Model (PDM) proposed by Cootes and Taylor [42]. PDM is a linear model that linearly approximates deformation of non-rigid objects. Equation 3.5 shows the working mechanism of PDM parameter $\mathbf{q}$ generating a model instance $\mathbf{X}$, which represents the 3D landmark positions in a face. It is obvious that the coordinates are put into a column vector here in Equation 3.6 [42].

$$\mathbf{X} = \bar{\mathbf{X}} + \mathbf{\Phi q} \tag{3.5}$$

$$\mathbf{X} = [X_1, X_2, ..., X_n, Y_1, Y_2, ..., Y_n, Z_1, Z_2, ..., Z_n]^T \tag{3.6}$$

Above $\bar{\mathbf{X}}$ represents the mean position of the facial landmarks in the 3D object coordinate frame (denoted in the same form as Equation 3.6). The non-rigid deformation in a face is controlled by m dimensional parameter $\mathbf{q}$ and $m \times n$ deformation coefficient matrix $\mathbf{\Phi}$.

Here m denotes the number of components needed to control the linear deformation of the face. Since the obtained images have two dimensions, a projection method is needed to project the 3D model to the image plane. The projection method will be described in the following part of this section.

Both mean landmark positions $\bar{\mathbf{X}}$ and coefficient matrix $\mathbf{\Phi}$ can be learned offline from the manually-labeled face images using Principal Component Analysis (PCA) [43], [145]. PCA can reduce the dimensionality of the data from current dimension to some smaller dimension. Finally, only m main components are needed to describe a face using a shape model.

When PCA is applied to the PDM, the probability of non-rigid shape parameter $\mathbf{q}$ is often assumed to have a zero mean Gaussian distribution with covariance matrix $\mathbf{\Lambda}$, which results in the prior probability in Equation 3.7 [9],

$$p(\mathbf{q}) = \mathcal{N}(\mathbf{q}; \mathbf{0}, \mathbf{\Lambda}) = \frac{1}{\sqrt{(2\pi)^m \left|\mathbf{\Lambda}\right|}} exp\left\{-\frac{1}{2}(\mathbf{q}^T \mathbf{\Lambda}^{-1} \mathbf{q})\right\}, \tag{3.7}$$

where $\mathbf{\Lambda} = diag([\lambda_1; ...; \lambda_m])$ is learned from the training dataset and based on the amount of shape deformation that the i^{th} parameter can explain. The higher value the λ_i gains, the more dominant the i^{th} component is.

Therefore the prior probability in Equation 3.3 is obtained and the regularization term is expressed in following equation [9]:

$$\mathcal{R}(\boldsymbol{q}) = -ln\left\{\frac{1}{\sqrt{(2\pi)^m \left|\mathbf{\Lambda}\right|}} exp\left\{-\frac{1}{2}(\mathbf{q}^T \mathbf{\Lambda}^{-1} \mathbf{q})\right\}\right\} \tag{3.8}$$

$$= ln\sqrt{(2\pi)^m \left|\mathbf{\Lambda}\right|} + \frac{1}{2}(\mathbf{q}^T \mathbf{\Lambda}^{-1} \mathbf{q}). \tag{3.9}$$

Since the constant term has no influence on the minimization of $\mathcal{R}(\boldsymbol{q})$, it can be ignored and the regularization term turns into [9]:

$$\mathcal{R}(\boldsymbol{q}) = \frac{1}{2}(\mathbf{q}^T \mathbf{\Lambda}^{-1} \mathbf{q}) \propto \|\mathbf{q}\|^2_{\mathbf{\Lambda}^{-1}}, \tag{3.10}$$

where $\|\mathbf{q}\|_{\mathbf{\Lambda}^{-1}}$ is equal to $\sqrt{\mathbf{q}^T \mathbf{\Lambda}^{-1} \mathbf{q}}$, which represents the Mahalanobis distance with a covariance matrix $\mathbf{\Lambda}^{-1}$. Since covariance matrix $\mathbf{\Lambda}^{-1}$ here is diagonal, it reduces to a normalised Euclidean distance.

- **Projection Method**

As mentioned before, a projection method is needed to project the 3D face model to the 2D image frame. Nonetheless, a weak perspective projection is sufficient in the proposed framework, instead of a full (or true) perspective camera model. The weak perspective camera model is a linear approximation of full perspective projection, and it is also called Scaled Orthographic Projection (SOP) [50]. In this approximation, it assumes that all

points on the object are placed in the same plane, which means these points have the same depth from the camera, as shown in Figure 3.4. This assumption is reasonable because the points on the face have little variation depth with respect to the depth between the face and camera. The following equation can be utilized to project i^{th} landmark point in PDM

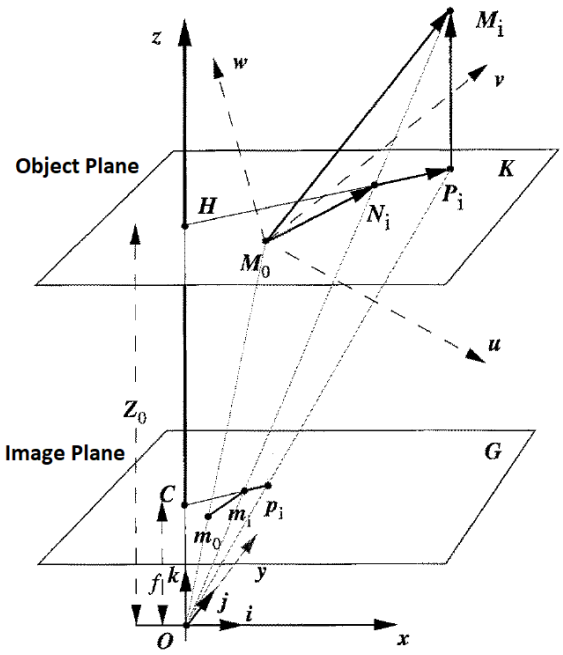


Figure 3.4: An illustration of an object point M_i, its full perspective projection point m_i, and its weak projection point p_i in the image plane. Taken from [50]

model to the image plane according to SOP [120]:

$$\mathbf{x}_i = s \cdot \mathbf{R}_{2D} \cdot (\bar{\mathbf{X}}_\mathbf{i} + \mathbf{\Phi_i q}) + \mathbf{t}, \tag{3.11}$$

where $\mathbf{q}$ is non-rigid parameter (or local parameter), and $s, \mathbf{w}, \mathbf{t}$ are rigid motion parameters (or global parameters). When these parameters are combined, parameter $\mathbf{p} = [s, \mathbf{w}, \mathbf{t}, \mathbf{q}]$ is formed, which controls both local deformation and global head pose. Here $s = \frac{f}{z_0}$ is the scaling factor and $\mathbf{t} = [t_x, t_y]^T$ is the translation factor. $\mathbf{R}_{2D}$ is the first two rows of a full rotation matrix, which is described by an axis-angle rotation. An axis-angle rotation is represented by a vector $\mathbf{w} = [w_x, w_y, w_z]^T = \theta\hat{\mathbf{n}}$, where the magnitude of the vector ($\|\mathbf{w}\| = \theta$) denotes the amount of rotation in radians around the $\hat{\mathbf{n}} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$ axis. Rodriguez's formula [126], [8] in Equation 3.12 displays the relationship between the Axis-angle representation and a rotation matrix.

$$\mathbf{R}(\hat{\mathbf{n}}, \theta) = I + sin(\theta)[\hat{\mathbf{n}}]_\times + (1 - cos(\theta))[\hat{\mathbf{n}}]_\times^2 \tag{3.12}$$

where $[\hat{\mathbf{n}}]_\times$ is the matrix form of the cross product operator with the vector $\hat{\mathbf{n}} = (\hat{n_x}, \hat{n_y}, \hat{n_z})$, and has form shown below:

$$[\hat{\mathbf{n}}]_\times = \begin{bmatrix} 0 & -\hat{n}_z & \hat{n}_y \\ \hat{n}_z & 0 & -\hat{n}_x \\ -\hat{n}_y & \hat{n}_x & 0 \end{bmatrix}. \tag{3.13}$$

Since projection and PDM model are combined, the following equation can be defined [9]:

$$\mathcal{P}_{SOP}(\mathbf{p}) = T_{s,\mathbf{w},\mathbf{t}}(\bar{\mathbf{X}} + \mathbf{\Phi q}), \tag{3.14}$$

where $\mathcal{P}_{SOP}$ represents the SOP projection, and the following equation in the form of homogeneous coordinates can be obtained [9]:

$$\mathcal{T}_{s,\mathbf{w},\mathbf{t}}(\mathbf{X}) = [s \cdot \mathbf{R}_{2D}|\mathbf{t}] \cdot \begin{bmatrix} X_1 & X_2 & ... & X_n \\ Y_1 & Y_2 & ... & Y_n \\ Z_1 & Z_2 & ... & Z_n \\ 1 & 1 & ... & 1 \end{bmatrix}. \tag{3.15}$$

Usually global parameter $s, \mathbf{w}, \mathbf{t}$ is non-informative; therefore, the non-informative prior probability can be used for global parameters, which assumes all rigid deformations are equally likely. $\tilde{\mathbf{\Lambda}}^{-1}$ can now be described as [9]:

$$\tilde{\mathbf{\Lambda}}^{-1} = diag([0; 0; 0; 0; 0; 0; \lambda_1^{-1}; ...; \lambda_m^{-1}]),$$

which leads to the following regularization factor [9]:

$$\mathcal{R}(\mathbf{p}) = \|\mathbf{p}\|^2_{\tilde{\mathbf{\Lambda}}^{-1}}. \tag{3.16}$$

3.1.2 Local Detector

The most important part of CLM facial landmark detection method is the local appearance model (also called local detector or patch expert), which models the appearance feature of the area around landmark positions, independently. Different landmark locations have their own local detectors. In addition, the local detector can provide the probability that the currently estimated position is aligned or misaligned to the true landmark position based on patch around it. After performing an exhaustive local search using the local detector in the region of interest, response map $\{p(l_i = 1|\mathbf{x}_i, \mathcal{I})\}_{i=1}^n$ is obtained.

A local detector contains two components: a classifier that discriminates aligned locations from misaligned ones and a Logistic Regressor that gets an approximate probabilistic output from the output of the classifier, enforcing the output in the range of 0 and 1. Every time a local detector is applied to an image patch, a scalar is obtained, representing the probability that the current estimated location is aligned to the ground truth from the intensity in that patch. The local detector is applied to the $n \times n$ patch around each pixel among the region of interest, which generates a response map, as shown in Figure 3.2

and 3.6. The brighter or the hotter position indicates the greater possibility. An alternative discription will be that the local detector can find the relationship between the patch intensity around a particular location and the likelihood of that location being aligned to the ground truth. Generally, a linear Support Vector Machine (SVM) and a Logistic Regressor (LR) are combined to function as a local detector [144], [120], [74]. There also exist a variety of classification approaches such as GentleBoost [87], [60]. When using SVM as a classifier, the probability of alignment at a particular landmark position, $\mathbf{x}_i$, can be modeled as follows [9]:

$$p(l_i|\mathbf{x}_i, \mathcal{I}) = \frac{1}{1 + e^{\boldsymbol{\eta} C_i(\mathbf{x}_i;\mathcal{I}) + \eta_0}}, \tag{3.17}$$

where $C_i(\mathbf{x}_i;\mathcal{I})$ is the result of a SVM, shown in Equation 3.18 [144]. $\boldsymbol{\eta}$ and η_0 are the coefficients of LR.

$$C_i(\mathbf{x}_i;\mathcal{I}) = \mathbf{w}_i^T \mathcal{P}(\mathcal{W}(\mathbf{x}_i;\mathcal{I})) + b_i \tag{3.18}$$

where $\mathbf{w}_i$ represents the gain, and b_i denotes the bias. $\mathcal{P}(\boldsymbol{z})$ is a function that normalizes vector $\boldsymbol{z}$ to zero mean and unit variance, reducing the sensitivity of a patch image to intensity variation. $\mathcal{W}(\mathbf{x}_i;\mathcal{I})$ is the vector format of all intensity in a $n \times n$ patch centered around $\mathbf{x}_i$. Both $\mathbf{w_i}$ and b_i can be learned offline from the training dataset that contains both positive and negative patch images. Positive patch images are the patches centered at true landmark position while negative patch images are the patches far away from the ground-truth location. Computational cost is always one of the main concerns in facial landmark detection, and linear SVM can solve this problem easily because efficient convolution can be used to reduce the computational cost and accelerate the process.

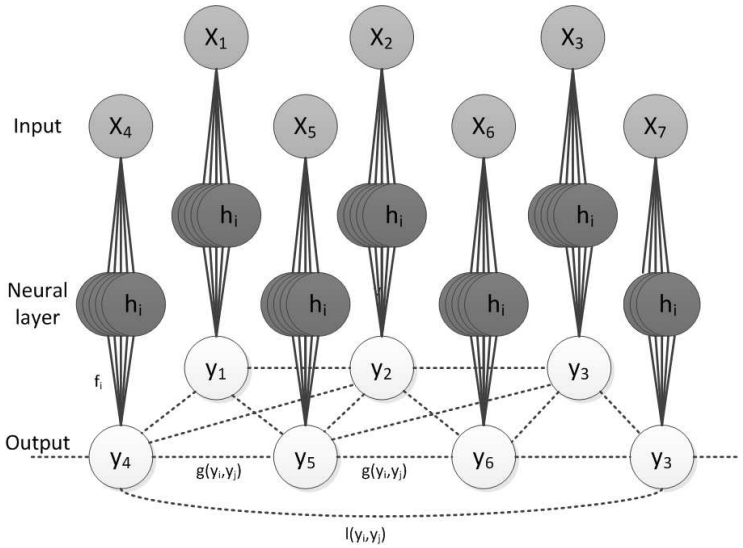


Figure 3.5: LNF model. Taken from [9]

Although SVM-LR local detector is popular for speed and training convenience, it has its limitations. This local detector utilizes a linear SVM, which is too simple when it comes to a more complicated facial landmark detection environment where intensity variation is

high. Therefore, an efficient detector handing non-linear relationship is needed. Recently, Baltruvsaitis proposed a new local detector in [9], which shows good performance in CLM facial landmark detection. The local detector he proposed is called Local Neural Field (LNF), which is an instance of Continuous Conditional Neural Field (CCNF). LNF makes use of a hidden non-linear layer and spatial relationship among pixels to solve problems in the complex environment. Also, simple convolutions can be used to simplify and speed up the computation like what linear SVM-LR does, leading to near-real-time performance.

As a local detector, LNF learns the non-linear relationship between pixel values and response map. It also combines the spatial relationship among the pixels in the area of interest around the currently estimated landmark. The spatial relationship consists of both spatial similarity and spatial sparsity. Spatial similarity means that near pixels share a similar probability of alignment while spatial sparsity punishes the sharp response to make the response map smoother. Figure 3.5 illustrates the structure of LNF with edge features. The input of LNF is a set of patch feature for every pixel and patch feature $\mathbf{x}_i$ is a vectored intensity of a 2D patch. The input is represented by $\boldsymbol{x} = \{\mathbf{x_1}, ..., \mathbf{x_N}\}$. The output of LNF is a set of alignment probability (i.e., response map) represented by $\boldsymbol{y} = \{y_1, ..., y_N\}$, where y_i is a scalar between 0 and 1. Vertex feature f_k describes the relationship between $\mathbf{x}_i$ and y_i through a neural layer $\boldsymbol{\Theta}$. A single neural layer contains multiple neurons described by vertex feature f_k, and weight coefficient $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_{K_1})$ denotes the importance and reliability of every neuron. K_1 is the number of neurons in a single neural layer, which depends on the realistic situation. The edge features applied here are g_k and l_k, which describe the similarity and sparsity respectively in a response map.

The mathematical model of LNF is described by the probability density function shown in Equation 3.19, which models the probability distribution of output under the condition of observed value $\mathbf{x}$ [9].

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{exp(\Psi)}{\int_{-\infty}^{\infty} exp(\Psi) d\boldsymbol{y}} \tag{3.19}$$

The model is controlled by a series of potential functions Ψ, and the denominator $\int_{-\infty}^{\infty} exp(\Psi) d\mathbf{y}$ is used for normalization.

The potential function is given by Equation 3.20 [9],

$$\Psi = \sum_i \sum_{k=1}^{K_1} \alpha_k f_k(y_i, \boldsymbol{x}, \boldsymbol{\theta}_k) + \sum_{i,j} \sum_{k=1}^{K_2} \beta_k g_k(y_i, y_j) + \sum_{i,j} \sum_{k=1}^{K_3} \gamma_k l_k(y_i, y_j), \tag{3.20}$$

where $\boldsymbol{\alpha} = \{\alpha_1, ..., \alpha_{K_1}\}$, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_{K_1}\}$, $\boldsymbol{\beta} = \{\beta_1, ..., \beta_{K_2}\}$, and $\boldsymbol{\gamma} = \{\gamma_1, ..., \gamma_{K_3}\}$ are model parameters. f_k, g_k, and l_k are defined in the following equations [9]:

$$f_k(y_i, \boldsymbol{x}, \boldsymbol{\theta}_k) = -(y_i - h(\boldsymbol{\theta}_k, \mathbf{x}_i))^2 \tag{3.21}$$

$$h(\boldsymbol{\theta}, \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}}} \tag{3.22}$$

$$g_k(y_i, y_j) = -\frac{1}{2} S_{i,j}^{(g_k)} (y_i - y_j)^2 \tag{3.23}$$

$$l_k(y_i, y_j) = -\frac{1}{2} S_{i,j}^{(l_k)} (y_i + y_j)^2 \tag{3.24}$$

These model parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}$ and $\boldsymbol{\Theta}$ are obtained from training, which maximize the conditional likelihood of LNF model.

$$(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}}, \hat{\boldsymbol{\Theta}}) = \arg \max_{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Theta}} \sum_{q=1}^{M} logP(\boldsymbol{y}^{(\mathbf{q})} | \boldsymbol{x}^{(\mathbf{q})}) \tag{3.25}$$

where M is training number. For more details about the Formula Derivation of the way to get the locally optimal model parameters, please refer to the reference [9].

When the local detector is applied to the area of interest, the response map can be obtained from the inference of $\boldsymbol{y}$, which maximizes the conditional probability under the condition of observed $\boldsymbol{x}$ (i.e., the pixel values in the area of interest).

$$\hat{\boldsymbol{y}} = \arg \max_{\boldsymbol{y}} P(\boldsymbol{y} | \boldsymbol{x}) \tag{3.26}$$

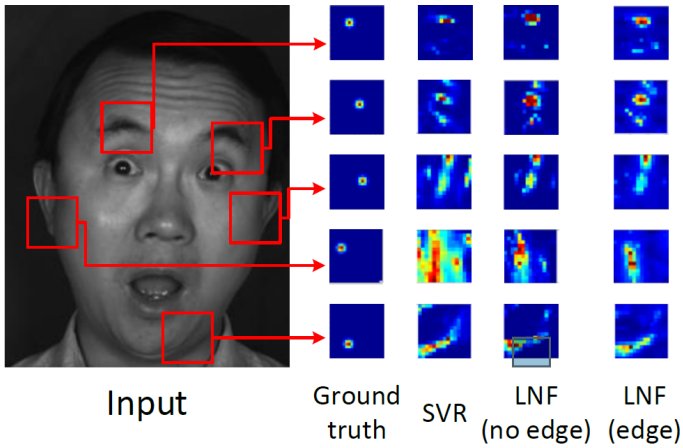


Figure 3.6: Response maps obtained after performing SVM-LR and LNF (with or without edge features) local detectors on areas of interest around different landmark locations. Taken from [9]

As shown in Figure 3.6, the ground-truth response map is the ideal input used in training. There are also response maps resulting from SVM-LR and LNF (with and without edge features) local detectors. The LNF local detector with edge features exploits the spatial features while the counterpart without edge features does not, which means the latter does not have edge feature functions in the potential function. It's clear that response map from linear SVM-LR has more noisiness. In addition, response map from LNF with edge has fewer peaks and is smoother compared to that without edge.

3.1.3 CLM Fitting Method

Finally, a fitting method is needed to minimize the regularization term and misalignment term in Equation 3.1 after the shape model and local detector are decided. Usually, a two-step fitting strategy is employed to minimize the error function regressively.

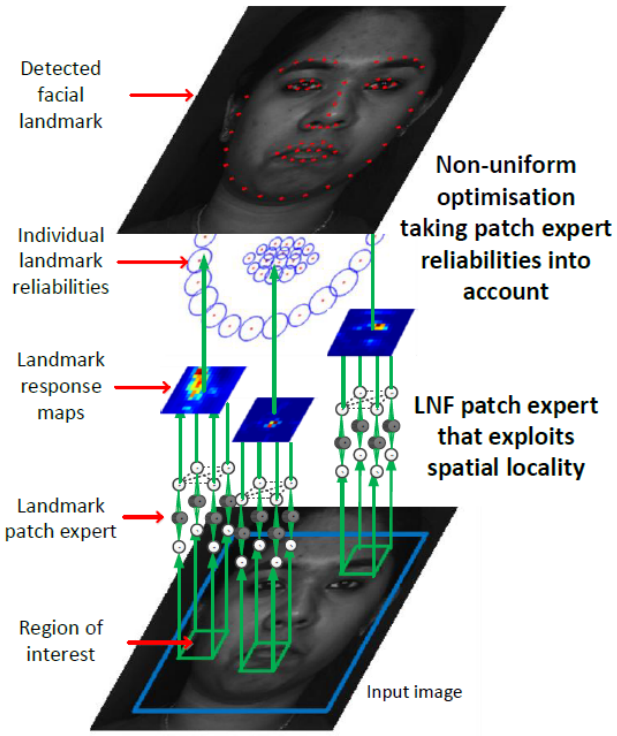


Figure 3.7: Overview of the CLNF model. Taken from [9]

An overview of CLNF, the CLM version chosen in this thesis, is illustrated in Figure 3.7, which uses LNF local detector and NU-RLMS fitting method. Firstly, an exhaustive search using local detector is applied to region of interest around each currently estimated position, which results in response maps on every region of interest $\{p(l_i = 1|\mathbf{x}_i, \mathcal{I})\}_{i=1}^{n}$. Secondly, model parameter $\mathbf{p}$ keeps being updated to minimize regularization factor $\mathcal{R}(\mathbf{p})$, until it converges. For more details about the formula derivation of parameter updating and convergence, please refer to paper [9]. Instead of optimizing on the original response maps directly, some approximation methods are usually performed on the response maps to make the response maps have simpler parametric forms [120]. Saragih et al. proposed an approximation method called Regularized Landmark Mean-Shift (RLMS) in [120], which shows better performance compared to others. However, RLMS treats all local detectors equally while the accuracy of local detectors for different landmark locations varies. Therefore, a Non-uniform Regularised Landmark Mean Shift (NU-RLMS) method is used to allocate different weights to different local detectors which results in more reliabilities [9]. In the realistic environment, video can provide temporal information used by CLM fitting to track the facial landmark. The parameters estimated by previous frames can be utilized as the

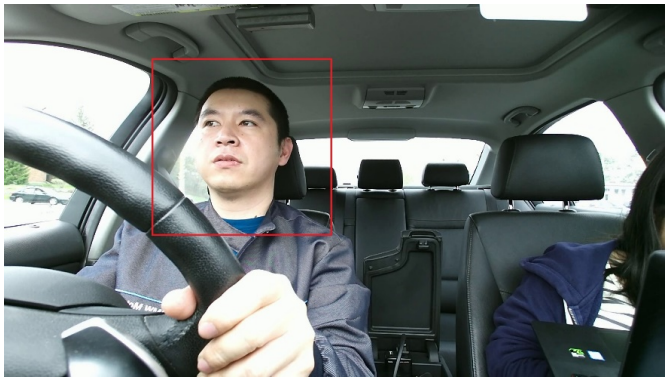
initial estimation of next frame.

At the beginning of CLNF fitting, face detection is employed to provide an initial estimate of facial landmark locations, which gives a bounding box around the face. Since the face detection is a very time-consuming task, it is only applied at the initiation stage in the beginning or every time the CLNF failed to track the facial landmark.

Viola-Jones face detector [142] is utilized to provide a bounding box around the face and initialize the parameters in CLNF model. It is one of the most popular face detectors and faster than many other methods [9]. The implementation of Viola-Jones method has already been included in many image processing libraries, such as OpenCV. The method is carried out by applying a fixed size detection window on an image. The detection window slides along x-axis and y-axis to detect if there is a face in the window. For each detection window, multiple Haar-like features are extracted. There is a corresponding weak classifier for each type of Haar feature. Next, the feature is put into a cascaded classifier to distinguish negative area from the positive area, where negative area means there is no face in this window while positive area means the opposite. The cascaded classifier consists of multiple layers of strong classifiers, and feature from a window is input in these classifiers layer by layer. Each strong classifier is a combination of weak classifiers with different weights. A window is classified as positive if it passed all cascaded classifiers, and it will be discarded if it failed at any step of the cascaded classifier.

3.1.4 CLM optimization

Extracting region of interest (ROI) is a very efficient way to optimize an detection algorithm in both time and computation costs. The task is usually performed in preprocessing stage, and it is often unnecessary to process the whole image. The ROI is usually a rectangular area that contains the most important information. In the realistic driving environment, the video data can provide spatial constraints for facial landmark detection because the driver's face is always in a rectangular area, as shown in Figure 3.8. Therefore, an ROI can be predefined in preprocessing stage of frames, which can reduce false detection rate by narrowing the searching areas. Figure 3.9 illustrates some failed detection without setting ROI. The trees outside the car, and the interesting shapes and patterns on the clothes can all increase the false detection rate.



(a) Color image (1920*1080)

(b) Infrared image (512 * 424)

Figure 3.8: ROI set in driving environment

In addition to the spatial constraints, the video data can also provide temporal constraints, because the movement of head is minor between two image frames. Thus, optimization methods for tracking can be applied to raise detection accuracy.

Figure 3.9: Some failed detection without setting ROI

3.2 Head pose Estimation

Since both 2D facial landmark locations in the image plane and their corresponding 3D locations in object frame are obtained, many methods that solve Perspective-n-Point (PnP) problem can be employed to get the head pose of a person. From previous section, a set of 3D reference points $\boldsymbol{p}_i^{Obj} = (X_i, Y_i, Z_i)^T, i = 1, ..., n, n > 3$ in the object frame, and their corresponding 2D projection points $\boldsymbol{p}_i^{Img} = (u, v, 1)^T, i = 1, ..., n, n > 3$ in image frame are calculated. Additionally, the intrinsic matrix of a camera can be obtained from the camera calibration progress.

The relationship between the 3D points and 2D projection points can be described as the following [126]:

$$\boldsymbol{p}_i^{Img} = f(\boldsymbol{p}_i^{Obj}; \boldsymbol{R}, \boldsymbol{t}) = \boldsymbol{K}[\boldsymbol{R}|\boldsymbol{t}]\boldsymbol{p}_i^{Obj}, \tag{3.27}$$

where $\boldsymbol{K}$ is the intrinsic matrix of camera. $\boldsymbol{R}$ is the rotation matrix and $\boldsymbol{t}$ is the translation. The full form of the equation is shown as follows [126]:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{3.28}$$

The most popular and accurate methods used in pose estimation are iterative optimization methods [88]. The pose estimation can be formulated as optimization problem whose objective function is Equation 3.29 [126],

$$\sum_{i=1}^{n} \left\| \tilde{\boldsymbol{p}}_i^{Img} - \boldsymbol{p}_i^{Img} \right\|^2, \tag{3.29}$$

where $\tilde{\boldsymbol{p}}_i^{Img}$ is the reprojection coordiantes in image plane obtained from the original 3D points and the estimated $\boldsymbol{R}$ and $\boldsymbol{t}$. After linerizing the reprojection error, the ojbective is to minimize the following error iteratively [126]:

$$\sum_{i=1}^{n} \rho(\frac{\partial f}{\partial \boldsymbol{R}} \Delta \boldsymbol{R}) + \frac{\partial f}{\partial \boldsymbol{t}} \Delta \boldsymbol{t} - \boldsymbol{r}_i), \tag{3.30}$$

where $\boldsymbol{r}_i = \tilde{\boldsymbol{p}}_i^{Img} - \boldsymbol{p}_i^{Img}$ is the current residual vector.

Gauss-Newton Method and Levenberg-Marquardt (LM) [84] are two common methods used in the optimization above iteratively. Since Gauss-Newton method needs a good initiation to converge while LM method does not need a good initiation estimation and guarantees the convergence, LM method is prevailing recently [88].

Many libraries have the implementation of PnP problem, such as OpenCV. In OpenCV, the function *solvePnP* can be used to estimate pose using 2D-3D reference points.

3.3 Eye Information Extraction

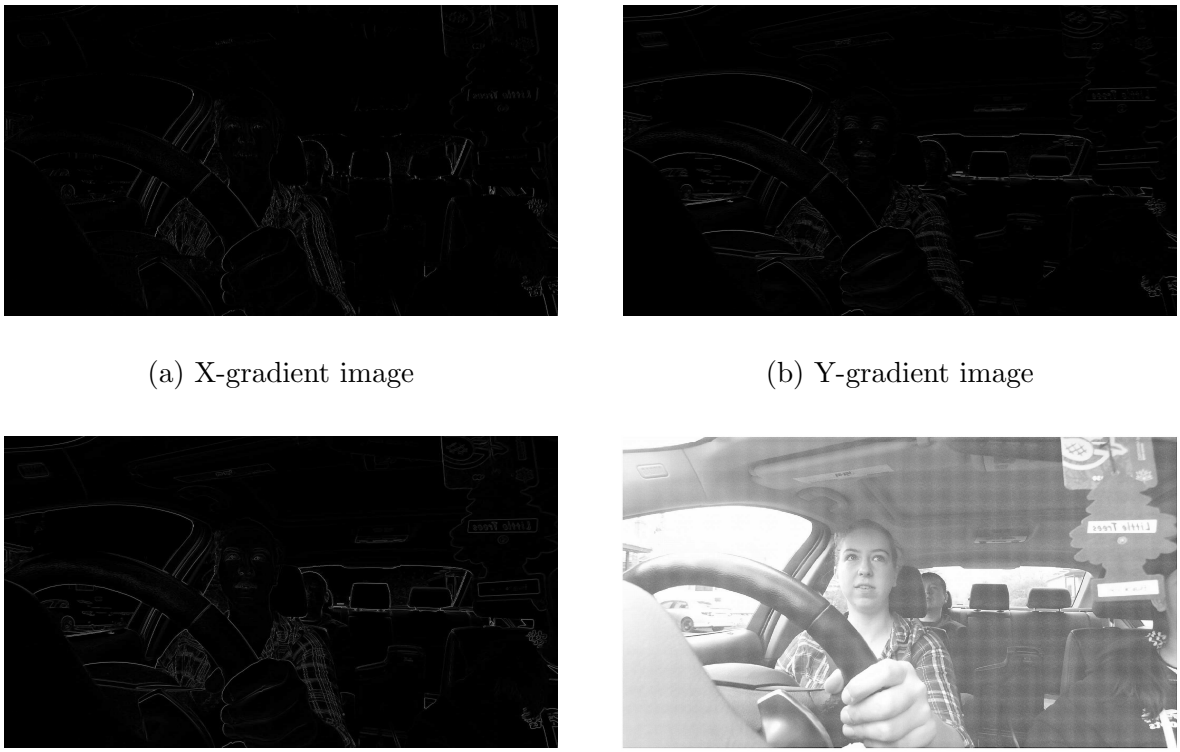
In this section, the methods of eye information extraction are described in detail, composed of iris center detection and upper-eyelid information extraction. These two resources can provide both horizontal and vertical information for gaze estimation.

3.3.1 Iris Center Detection

Iris Center Detection used in this thesis is based on paper [129]. A regression-model-based method is used to refine the estimation of iris center in image $\mathcal{I}$ iteratively. At every iteration, the currently estimated location of iris center is updated as Equation 3.31 [129],

$$\boldsymbol{p}_k = \boldsymbol{p}_{k-1} + \boldsymbol{R}_{k-1} feat(\boldsymbol{p}_{k-1}, \mathcal{I}), \tag{3.31}$$

where $\boldsymbol{p}_{k-1}$ is the previously estimated location, $\boldsymbol{p}_k$ is the currently estimated location, and $feat(\boldsymbol{p}_{k-1}, \mathcal{I})$ is the feature vector extracted from a patch around location $\boldsymbol{p}_{k-1}$ in image $\mathcal{I}$. There exist many image descriptors to extract the feature of a patch image. HOG descriptor [48] is applied in this thesis. $\boldsymbol{S} = (\boldsymbol{R}_1, ..., \boldsymbol{R}_K)$ is a set of regression matrices, which guide the update in each iteration based on the feature vector of previously estimated location. K is the iteration number to be set. An initial estimated position of iris center $\boldsymbol{p}_0$ needs to be set at the beginning of the iris center estimation progress. The iteration refines the estimated location step by step because the successive model reduces the residual errors, shown in Equation 3.36, between ground truth and the estimation .



(a) X-gradient image

(b) Y-gradient image

(c) Gradient image

(d) Image with visualized HOG

Figure 3.10: Some resulting images

The computation of HOG descriptor has three main steps [48]:

Step 1: Compute the gradients for each pixel.

Before obtaining the overall gradients, the gradients along x axis and y axis are calculated respectively, as shown in Equations 3.32 and 3.33. It is equivalent to apply a Sobel filter with filter kernels $[-1, 0, 1]$ and $[-1, 0, 1]^T$ on the image.

$$G_x(x, y) = \mathcal{I}(x + 1, y) - \mathcal{I}(x - 1, y) \tag{3.32}$$

$$G_y(x, y) = \mathcal{I}(x, y + 1) - \mathcal{I}(x, y - 1) \tag{3.33}$$

where G_x and G_y are horizontal and vertical gradients, and $\mathcal{I}(x, y)$ is the intensity in position (x, y).

The gradient of a pixel is described by its magnitude and direction together, which can be achieved from horizontal and vertical gradient, shown in the following equations:

$$Gmag = \sqrt{G_x^2 + G_y^2}, \tag{3.34}$$

$$Gdir = arctan\frac{G_x}{G_y}, \tag{3.35}$$

where $Gmag$ is the magnitude and $Gdir$ is the direction of gradient on a pixel.

Figure 3.10 shows the visualized x-gradient, y-gradient and gradient of an image. X-gradient image shows more vertical lines, which provide more information about the intensity change along the x-axis. Y-gradient image shows more horizontal lines, which provide more information about the intensity change along the y-axis. The magnitude of gradient provides both horizontal and vertical information of intensity changes, and they are called edge information.

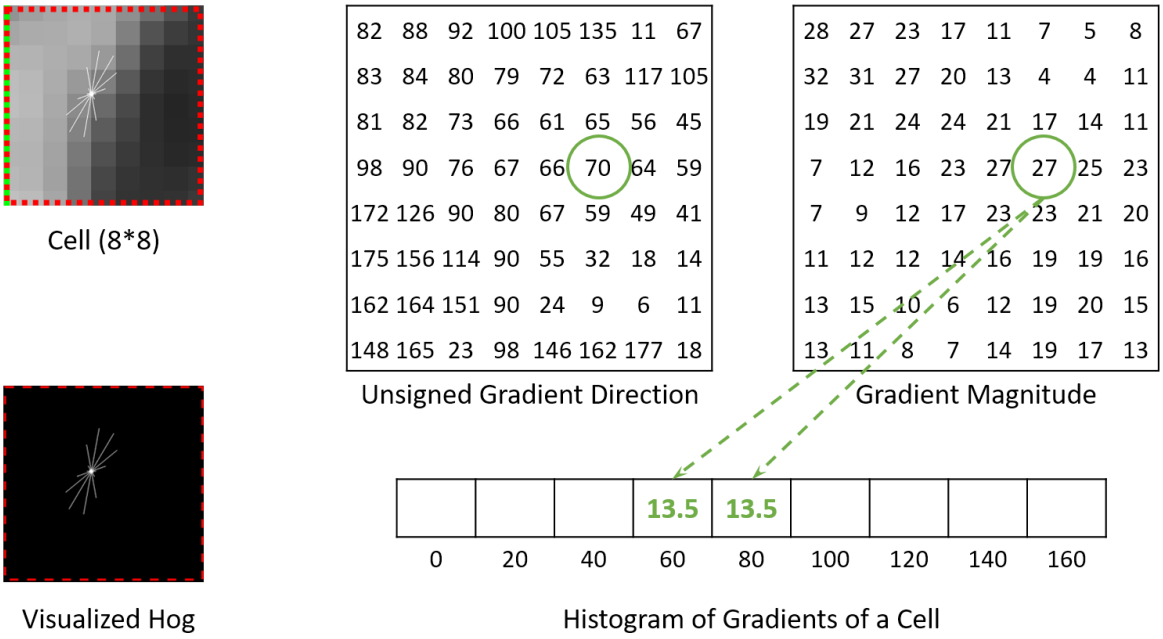


Figure 3.11: HOG computation

Step 2: Calculate the histogram of gradients in a cell.

A cell is usually an 8×8 area. The gradients of pixels in a cell can be divided into nine categories according to its unsigned directions, which are called 9-bins. These 9 bins are corresponding to angles $0, 20, 40, ..., 160$ degrees, respectively. The direction of gradient decides the number of bin and the magnitude decides the amount of value put in the bin. The histogram of gradient for a cell is actually a vector with length 9 that shows the votes of each direction. As shown in Figure 3.11, the green encircled magnitude 27 is in direction 170 degrees that is exactly between bins 160° and 180°. Thus, value 27 is separated equally to vote the two bins because the vote is assigned proportionally according to the direction.

The step of voting is repeated and the 9-bin HOG of a cell can be obtained. The visualized HOG for this cell is a star shape in the cell shown in Figure 3.11. The length of each line represents the histogram, and its direction is perpendicular to the direction of the gradient, which means it looks along the edge.

Step 3: Calculate the HOG feature vector.

In the previous step, the vector representing each cell is calculated. A strategy combining different cells is needed, shown in Figure 3.12. In this thesis, the HOG is obtained from two $40*40$ patches, and each eye has its own patch. A sliding $8*8$ block slides in the patch horizontally and vertically with sliding step $8*8$. Only one $8*8$ cell is contained in a block. The block has the same size as the cell in this case, thus only one cell is used to describe the block. In summary, there are two patches with $5*5$ blocks, and each block has 1 cell. Since the vector for one cell has length 9, the final HOG feature vector is generated by combining vector of each cell. Therefore, the feature vector for two patches has total length $2*5*5*9=450$.

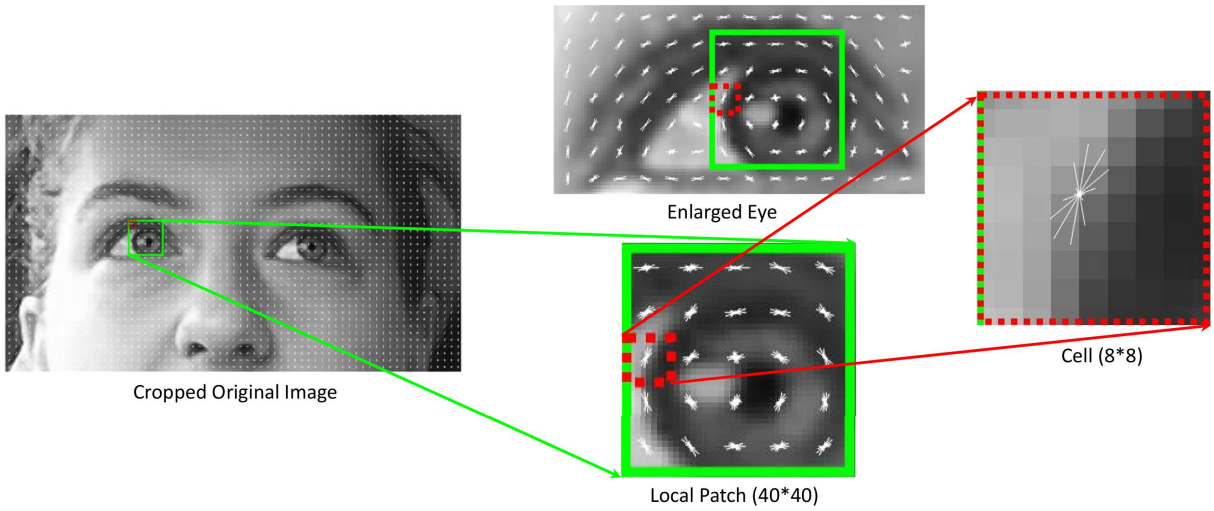


Figure 3.12: HOG feature extraction process

The successive model $\boldsymbol{S}=(\boldsymbol{R}_1,...,\boldsymbol{R}_K)$ is often trained offline and used online. A set of training images $\{\mathcal{I}^i\}_{i=1}^N$ and their corresponding ground-truth iris center locations $\{\boldsymbol{p}_*^i\}_{i=1}^N$ are needed, where N is the amount of training samples. In addition, the initial estimation $\{\boldsymbol{p}_0^i\}_{i=1}^N$ are required. For the training, the residual error, which represents the distance between the ground-truth location and the estimated location, will be minimized in every iteration. Equation 3.36 shown the formulated form of the error (Loss Function) to be minimized at iteration k [129]:

$$L=\sum_i\sum_{\boldsymbol{p}_{k-1}^i}\left\|\boldsymbol{p}_*^i-\boldsymbol{p}_k^i\right\| \tag{3.36}$$

$$=\sum_i\sum_{\boldsymbol{p}_{k-1}^i}\left\|\boldsymbol{p}_*^i-\boldsymbol{p}_{k-1}^i-\boldsymbol{R}_{k-1}feat\left(\boldsymbol{p}_{k-1}^i,\mathcal{I}^i\right)\right\|^2. \tag{3.37}$$

The target becomes:

$$\boldsymbol{R}_{k-1} = \underset{\boldsymbol{R}_{k-1}}{\arg\min} \quad L. \tag{3.38}$$

It is assumed that $\boldsymbol{p}_*^i - \boldsymbol{p}_{k-1}^i = \mathbf{y} = [y_1, y_2]$, and $feat(\boldsymbol{p}_k^i, \mathcal{I}^i) = \mathbf{x} = [1, x_1, ..., x_{n-1}]$. Thus, $\mathbf{y}^T = \boldsymbol{R}_{k-1}\mathbf{x}^T$ according to Equation 3.31. The loss function becomes:

$$L = \sum_i \sum_{\boldsymbol{p}_{k-1}^i} \|\mathbf{y} - \boldsymbol{R}_{k-1}\mathbf{x}\| . \tag{3.39}$$

It is actually a multivariable linear regression model with variable vector $\mathbf{y}$ and length $m = 2$ [135]:

$$\begin{aligned} y_1 &= \beta_{01} + \beta_{11}x_1 + \cdots + \beta_{(n-1)1}x_{n-1} + e_1 \\ y_2 &= \beta_{02} + \beta_{12}x_1 + \cdots + \beta_{(n-1)2}x_{n-1} + e_2 \\ &\cdot \\ &\cdot \\ &\cdot \\ y_m &= \beta_{0m} + \beta_{1m}x_1 + \cdots + \beta_{(n-1)m}x_{n-1} + e_m \end{aligned} \tag{3.40}$$

with $m \times n$ coefficient matrix $\boldsymbol{R}_{k-1}$ and $m = 2$

$$\boldsymbol{R}_{k-1} = \begin{bmatrix} \beta_{01} & \beta_{11} & \cdots & \beta_{(n-1)1} \\ \beta_{02} & \beta_{12} & \cdots & \beta_{(n-1)2} \end{bmatrix} \tag{3.41}$$

where the length of feature vector $\mathbf{x}$ is feature number+1, and the error vector is $\mathbf{e} = (e_1, ..., e_m)$. The least squares estimation can be used to estimate the coefficient matrix of this model as shown below [135]:

$$\hat{\boldsymbol{R}}_{k-1} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}, \tag{3.42}$$

where $\boldsymbol{X} = [\mathbf{x}_{(1)}; ...; \mathbf{x}_{(M)}]^T, \boldsymbol{Y} = [\mathbf{y}_{(1)}; ...; \mathbf{y}_{(M)}]^T$. After K iterations, a sequence of regression matrices $\boldsymbol{S} = (\boldsymbol{R}_1, ..., \boldsymbol{R}_K)$ can be obtained.

Before the training progress, some important preprocessings of the training images are performed, namely Scaling, Rotation, and Histogram Equalization.

1.Scaling: Scale the image to a size, where the distance between both eye centers are equal for every training images, to make sure the size of all eyes is similar. The distance is a fixed length set before the training.

2.Rotation: Rotate the image to ensure the axis through two eye centers is parallel to the x-axis, which makes sure the orientation of all eyes is almost the same.

3.Histogram Equalization: Apply Histogram Equalization only on the eye region (i.e., ROI), which is to enhance the contrast and consequently helps increase feature difference.

Figure 3.13: Diagram of preprocesses of the images

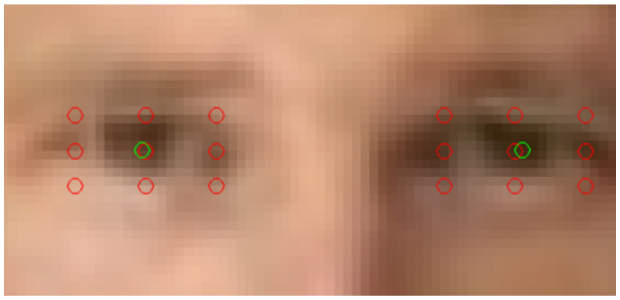


Figure 3.14: Ten initial estimates of iris centers. Taken from [129]

After the detection on the preprocessed image, the estimated iris center coordinates are with regard to the preprocessed image, instead of the original image. De-rotation and de-scaling are needed to achieve the iris center coordinates in the original image.

For the initially estimated iris center positions in training $\{p_0^i\}_{i=1}^N$, multiple different initial estimates are needed to provide the system enough uncertainty to learn a good model. Ten initial estimates are assigned shown in Figure 3.14. The initial estimates consist of eye center, eight locations around the center, and a ground-truth location. The eye center is the middle of two eye corners which are obtained from facial landmark detection in the previous section. Setting the ground truth as initial estimate is to make sure that the regression does not leave the ground truth too far from the first iteration.

For the initially estimated iris center location in testing, or in the real application, eye centers are assigned to the initial estimate. If the iris center detection is applied to the video, and the eye movement between two successive frames are small, which is usually the case, the estimation from the previous frame can be assigned as the initial estimation of the current frame.

To speed up iris detection procedure, an optimization method, such as applying ROI,

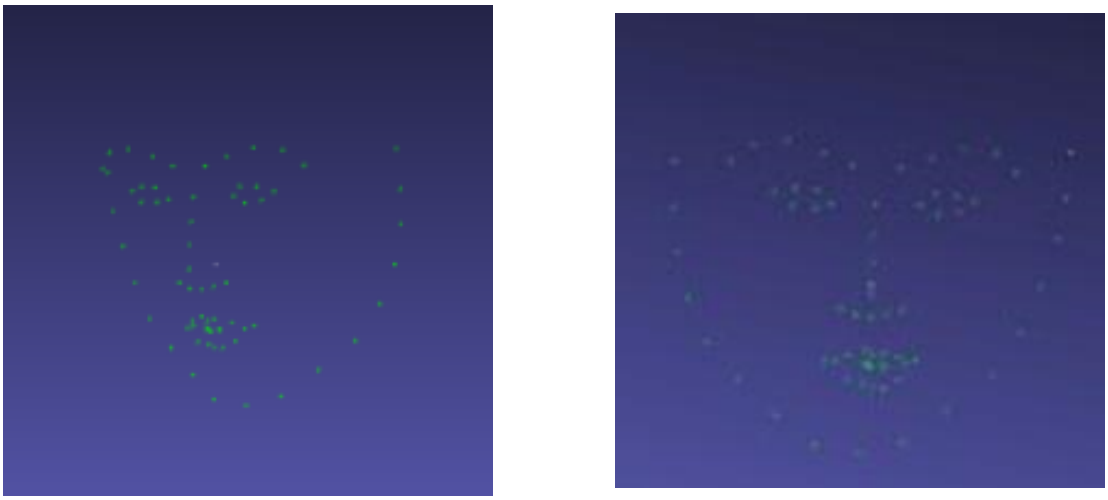
can be utilized. The preprocessings of scaling and rotating are time-consuming tasks. If a smaller region of the image is processed, the time consumption will decrease. Therefore, the area around eye region is cropped from original image before scaling and iris center detection is applied to such ROI.

3.3.2 Upper Eyelid Information extraction

As shown in paper [68], the shapes of human eyelids change with the movement of gaze direction, especially that of upper eyelids and when the gaze move in the up-and-down direction. The shapes of upper eyelids can be modeled as a quadratic function shown in Equation 3.43.

$$y = ax^2 + bx + c \tag{3.43}$$

Only the curvature of the curve is related to the gaze direction. To verify this characteristic, the relationship between the 2nd-order coefficient a and the true up-and-down gaze direction is learned. The verification progress and results will be shown in the next chapter.



(a) Real face orientation

(b) Frontal face orientation

Figure 3.15: Head model in 3D

Before the upper eyelid information is utilized, some preprocessings are performed. As shown in Figure 3.15a, facial landmarks obtained from previous section provide total eight points to describe left and right upper eyelids. However, the shape of original landmarks in the image plane are obtained and reshaped from the projection of real 3D head model. Before eyelids modeling, the 3D head model is rotated to the frontal position first, as shown in Figure 3.15b. Therefore, Equation 3.43 can model the shape of eyelids better.

3.4 Gaze Zone estimation

In this section, gaze zone estimation will be described, consisting of the feature vector extraction and the classification method applied in this thesis.

3.4.1 Feature Extraction

In addition to the head pose, composed of head position and orientation, iris center and upper-eyelid information are utilized in the gaze zone estimation. For iris information, the iris relative ratio (i.e., the relative position of iris between two eye corners) is applied in the classification. Besides, the preprocessing is performed, which rotates the head model to frontal orientation, the same as that in the section of upper-eyelid information extraction. For upper-eyelid information, the 2nd-coefficient of shape model is applied in the classification.

3.4.2 Classification

Multiclass classification method is required in this thesis because multiple gaze zones need to be classified in gaze zone estimation. Multiclass classification method applied in this thesis is RDF [33] since RDF has been successfully and widely utilized in various problems [123], as discussed in Chapter 2.

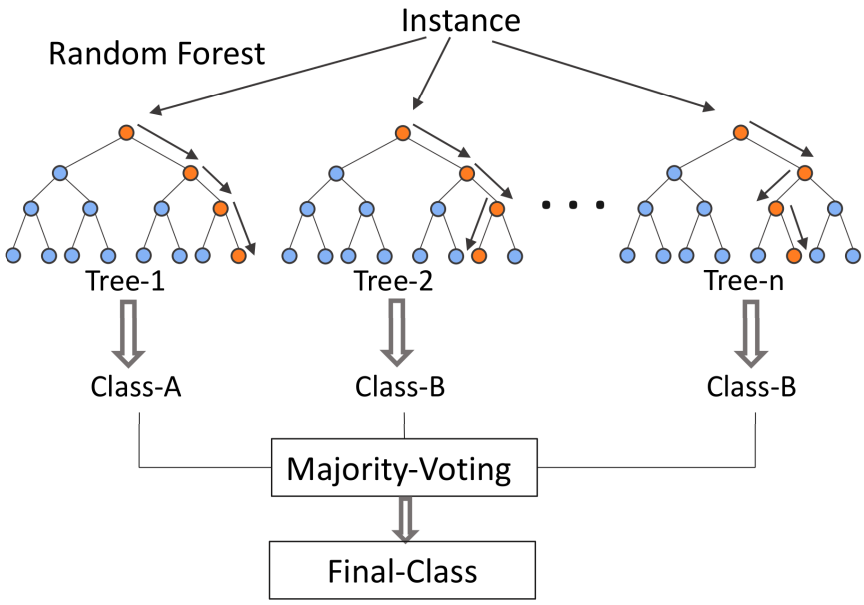


Figure 3.16: RDF classifier procedure [90]

RDF is an ensemble classifier using bagging strategy to aggregate Decision Trees. In addition, strategy of random feature selection is utilized to randomly select features for the training of each tree. Figure 3.16 demonstrates the procedure of the mechanism of RDF

classifier. It is assumed that the training data has N samples and the feature vector has size M. The following steps show the procedure to build a RDF [33]:

Step 1: Randomly select one sample from the total sample set and repeat this operation N times with replacement. Thus, N samples are gotten in the beginning, or at the root of the tree. There is a high possibility that there exist many replicated samples in the selected N samples because the selection is with replacement.

Step 2: At each split node, select m features from M features. Usually, $m = log_2 M$ [33]. The split property is decided by the mechanism used on the m features. The mechanism includes *information gain* and *Gini decrease*.

Step 3: Repeat step 2 until the tree can not split or some criteria is met, such as the maximum tree depth and minimum sample number. Thus one tree is built.

Step 4: Repeat Step 1,2,3 until the forest reaches the expected accuracy or the size of forest reaches the maximum tree number.

The parameters of RDF include maximum tree depth, minimum sample number, regression accuracy and maximum size of the forest.

RDF is an ensemble of Decision Tree, and the split mechanism is based on the Decision Tree algorithm chosen. CART (Classification and Regression Tree) [34], [147] is utilized as the algorithm to build decision trees in a random forest [33]. The algorithm performs differently for continuous and discrete feature vector. CART handling for continuous data will be introduced because continuous feature vector is used in the thesis. The most important component in a decision tree is the split mechanism, which decides the way of choosing the features and dividing the data at each node. CART is a binary tree where each node only splits into two nodes or leaves, which means the dataset is divided into two group of data at each node from the root. The use of binary trees slows down the split procedure and repeated splits are performed on the same attribute. Therefore, many partitions are generated for one attribute. The split rule at each node for is based on *Gini* coefficient, which measures the impurity of a node. The *Gini* coefficient for a split node is calculated from Equation 3.44 [34],

$$Gini(p_k) = \sum_{k=1}^{K} p_k(1 - p_k) = 1 - \sum_{k=1}^{K} p_k^2, \tag{3.44}$$

where p_k is the probability of k^{th} category and K is the total number of possible categories for attribute A.

In case of a binary split, $K = 2$, it is supposed that the probability of one of the category is p, thus the *Gini* coefficient for CART is shown below:

$$Gini(p) = \sum_{k=1}^{2} p_k(1 - p_k) = 2p(1 - p). \tag{3.45}$$

The smaller the *Gini* coefficient, the more purity the split has, and the better the split attribute and split point chosen. For a given sample set D, it is supposed that it needs to

be classified into K categories, and the amount of k^{th} category is C_k. Usually, category's classical probability is calculated based on the dataset. Equation 3.46 [34] gives the *Gini* of the sample set D.

$$Gini(D) = 1 - \sum_{k=1}^{K} (\frac{C_k}{|D|})^2 \tag{3.46}$$

where $|D|$ is the number of samples in the dataset.

For sample set D, if it is split into two parts D_1 and D_2 based on attribute A, the formulated *Gini* of D based on attribute A is shown below [34]:

$$Gini(D, A) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2). \tag{3.47}$$

For an attribute A with continuous value, a discretization scheme is applied to the data. It is supposed that attribute A has m different value in dataset D, and these values are sorted ascendingly in a list $\{a_1, a_2, ..., a_m\}, a_i < a_{i+1}$, for $i = 1, ..., m$. The median of two adjacent value is chosen as the potential split point, and there are $m-1$ candidates. The i^{th} potential split point is $T_i = \frac{a_i + a_{i+1}}{2}$. All $m-1$ $Gini(D, A)$ based on each potential binary split point are calculated. The split point T, which generates smallest *Gini*, is chosen as the split point at the current node, and samples will be classified into two groups based on the comparison between the value of attribute A and T. A sample whose A value is larger than T goes to group 1 while a sample whose A value is smaller than T goes to group 2. The discretization of continuous attribute is implemented from above steps. The stop of splitting on a node also depends on $Gini(D)$ of the current node. A *Gini* threshold is set at the beginning of tree generation.

3.5 Conclusion

In this chapter, the framework proposed for gaze zone estimation is presented in detail, which is composed of Facial Landmark Detection, Head Pose Estimation, Iris Center Detection, Upper Eyelid Information Extraction, and Gaze Zone Estimation. First, CLNF method, a CLM instance that uses LNF local detector and NU-RLMS fitting method, is applied to obtain the facial landmarks in the image plane and the 3D model of the face in the object frame. In addition, extracting region of interest is utilized as an optimization strategy for CLNF facial landmark detection. Second, Levenberg-Marquardt optimization method is used to solve the PnP problem, which estimates the head pose based on the 2D landmark locations in the image plane and their corresponding 3D locations in the object frame. Third, a regression model-based method is employed to obtain the iris center from eye landmarks detected in previous parts. For upper eyelid information extraction, a quadratic function is utilized to model the upper eyelid, and the second-order coefficient of quadratic function is extracted. Finally, the head pose and the eye information are combined to form a feature vector, and RDF classifier is utilized to estimate the current gaze zone of a driver from the feature vector extracted. The experiment results will be discussed in next chapter.

Chapter 4

Experiment Results

This chapter will present the experiment setup, the dataset, and the experiment results for iris center detection, upper eyelid information verification, and gaze zone estimation.

4.1 Experiment Setup and Dataset

This part will introduce the sensor, the experiment setup, the dataset and the software tools used in the experiments. Table 4.1 provides a summary of the experiment setup and dataset.

Names	**Descriptions**
Sensor	Kinect for Windows V2 Sensor
Environment	Realistic Driving Environment
Time	In both Daytime and Nighttime
Data Type	Color Data and Infrared Data
Dataset One	Totally 29,000 images are annotated manually for gaze zone estimation
Dataset Two	Totally 2,100 images are annotated manually for iris center estimation
Software Tools	Kinect Studio

Table 4.1: A summary of the experiment setup and dataset

4.1.1 Sensor

The sensor used in this thesis is a Kinect sensor V2 for Windows. Kinect V2 is the second version of Kinect, which is produced by Microsoft for Windows PC and Xbox One game console. Additionaly, Kinect SDK (Software Development Kit) for Windows is released by Microsoft for developers and researchers, and it is downloadable with a free user license. The SDK provides not only the sensor drivers to access the data streams,

but also some useful functions and code samples in both C++ and C#, which could significantly accelarate the devepolment process for Kinect-based applications in Microsoft Windows. The newest version Kinect SDK v2.0 [1] is applied in this thesis. Figure 4.1 shows the components of Kinect V2, which includes a Color Camera, an Infrared Projector and Receiver, and four Microphone Arrays. These components can provide multiple data sources, such as RGB data, Infrared data, Depth data, and Voice data. Infrared and Depth data is obtained by the Infrared Projector and Receiver.

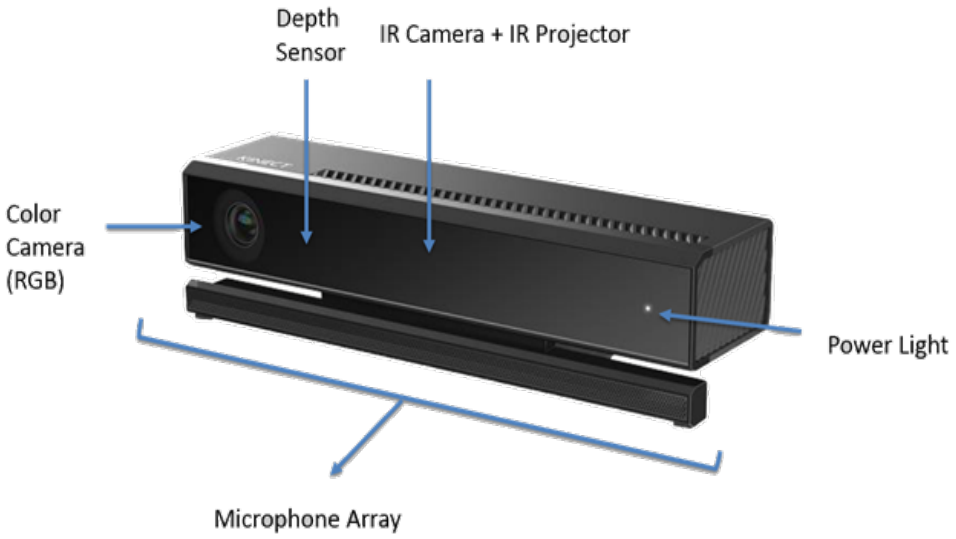


Figure 4.1: Kinect sensor V2 for Windows

As described in the specification table 4.2, the resolution for color and infrared data are 1920×1080 and 512×424 pixels, respectively. The color data is utilized in the daytime while the infrared data is exploited in the nighttime in this thesis. Another feature of the sensor is its total acquisition frame rate which is up to 30 frames per second (fps).

Feature	**Arguments**
Color Camera	1920×1080 at 30 fps
Depth Camera	512×424
Max Depth Distance	~ 4.5 m
Min Depth Distance	50 cm
Horizontal Field of View	70 degrees
Vertical Field of View	60 degrees
Supported OS	Win 8/Win 10

Table 4.2: Specialization of Kinect for Windows V2 sensor

(a) Kinect setup in the car

(b) Kinect setup in the room

Figure 4.2: Kinect Setup

4.1.2 Experiment Setup

For gaze zone estimation and iris center detection, the experiment is carried out in the realistic driving environment in both daytime and nighttime. Three volunteers are involved in the experiments, referred as *Volunteer 1*, *Volunteer 2*, and *Volunteer 3*, respectively. These volunteers come from the different country and have different genders, who are a Brazilian male, a Chinese male, and a Canadian female. The volunteers are required to drive along a designated route, performing left turn, right turn, and parking tasks in Ottawa. The road is uneven and has uphill and downhill sections along the route. The Kinect is attached to the back of the windshield as shown in Figure 4.2a to record the video while the drivers are driving.

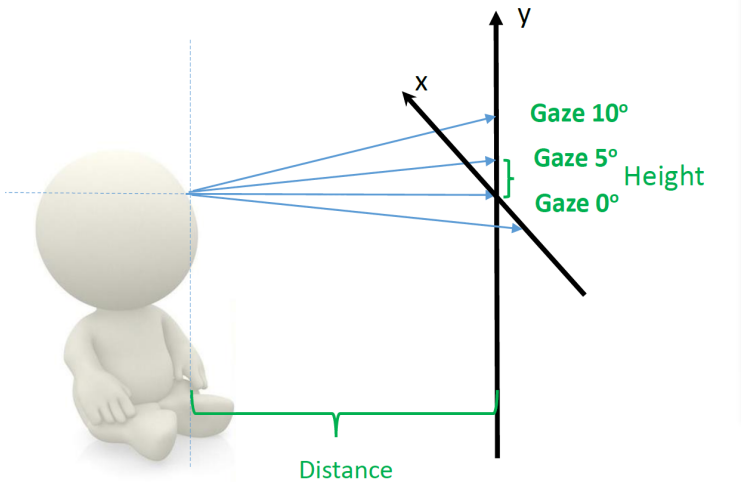


Figure 4.3: Geometric relationship

For Upper-eyelid information verification, the experiment is carried out in the indoor environment, as shown in Figure 4.2b. There exists a geometric relationship between the

gaze direction and the points a person is gazing on the wall. Those gazing points are marked on the wall based on the gaze direction horizontally and vertically, to get the discrete ground-truth gaze direction, shown in Figure 4.3. The horizontal or vertical gaze direction change between two adjacent points is 5°. The volunteer is asked to fix his head to the frontal position, and gaze those points marked on the whiteboard in both horizontal and vertical direction. Kinect is fixed in front of the volunteer to capture the images.

4.1.3 Dataset

For gaze zone estimation, the data is collected from the experiments in the realistic driving environment in both daytime and nighttime involving three volunteers, described in Section 4.1.2, shown in Figure 4.4a. The data is trained and tested for all the volunteers respectively. Totally 29,000 images are annotated manually for gaze zone estimation, 90% and 10% of which are dedicated to training and testing respectively.

For iris center detection, the dataset collected from our experiments described in Section 4.1.2 and the other public dataset are exploited as shown in Figure 4.4. Totally 2,100 images are annotated for iris center estimation from our dataset, 80% and 20% of which are dedicated to training and testing, respectively. The public dataset is called Labeled Faces in the Wild, which is a database for studying face recognition in unconstrained environments and contains more than 13,000 images of faces collected from the web [117]. In this thesis, we randomly select 350 images and make annotations for iris center detection in the wild, which will be a control group.

(a) Collected dataset in the daytime and nighttime

(b) Dataset in the wild

Figure 4.4: Dataset

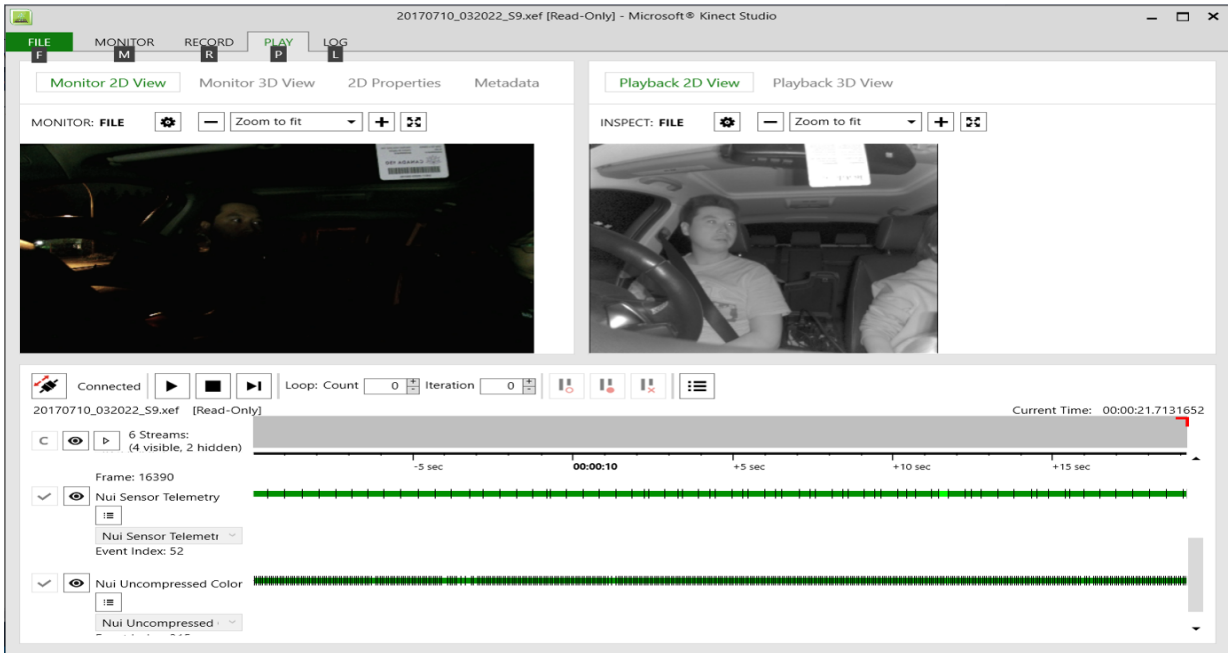


Figure 4.5: Kinect Studio

4.1.4 Software Tools

Another tool provided by Kinect SDK is Kinect Studio [2]. Kinect Studio is a tool that helps the developer to record and playback multiple data streams from a Kinect sensor, which has extension *.xef* (Extended Event File). This tool can help debug functionality, create repeatable scenarios for testing, and analyze the performance of a Kinect-Enabled Application by using it to read and write the *.xef* files. The data can be replayed for multiple times by setting the Iteration Number. Figure 4.5 shows the main window of Kinect Studio. It can also choose the data stream that is needed to be recorded and played. Useless data such as voice data in our case will not be saved if it is not chosen. The application can acquire the data stream from the replayed data from Kinect Studio the same way as it acquires data from a connected Kinect sensor.

4.2 Iris Center Detection

The data obtained from the experiments with three volunteers and the data collected from wild public dataset are utilized for training and testing of iris center detection. Figure 4.6 shows the testing results of iris center detection for color images. Normalized Error is applied to evaluate the results of detection, which is a relative error between the ground-truth position and estimated position in horizontal axis with regard to the length of the eye, represented by equation 4.1,

$$NormalizedError = \frac{|X_{real} - X_{estimate}|}{L_{eye}}, \tag{4.1}$$

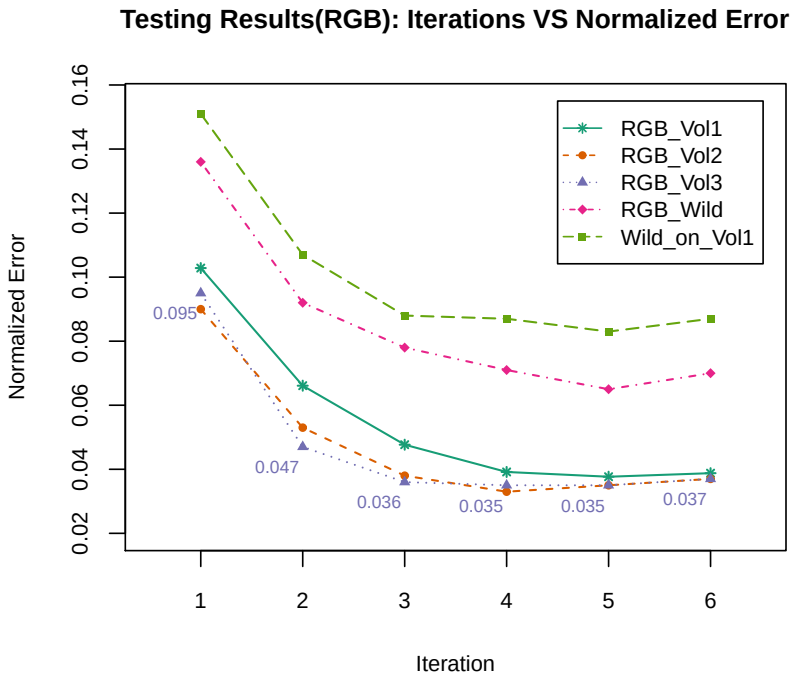


Figure 4.6: Iris center detection results for color data

where X_{real} and $X_{estimate}$ denote the horizontal positions of iris center and L_{eye} is the average of both eye lengths between two corresponding eye corners.

It is obvious that the normalized error decreases with the iteration, which corresponds to the mechanism of the method because the method estimates the iris center iteratively to get the best estimates. *RGB_Vol1, RGB_Vol2, and RGB_Vol3* represent the results of dataset captured from each volunteer, and *RGB_Wild* represents the results of the wild dataset described in section 4.1.3. *Wild_on_Vol1* is the results whose training data is the wild dataset while the testing data is data of *Volunteer 1*. The three line in the bottom are the results gotten from three volunteers, which have smaller normalized error compared with that of *RGB_Wild* and *Wild_on_Vol1*. The best three lines all reach the smallest normalized error at iteration 4, and then the iteration number is set to four in the application. The best result has normalized error 0.035.

The original intensity of infrared data provided by Kinect sensor is represented by 16 bits for each pixel, that ranges from 0 to 2^{16}. Usually, the 8-bit image is enough for detection and can be shown in the screen conveniently. In this case, scaling the 16-bit data down to 8-bit is necessary, which means scale down the pixel value. Different scales result in images with different qualities. Figure 4.7 shows the results of five scale levels for infrared data. The scale value increases from left to right. The right images are lighter than the left ones. After the scaling, gamma correction is utilized to enhance the contrast of the image. The iris center detection method is applied to images with different scale levels for all volunteers. Figures 4.8,4.9, and 4.10 illustrate the results of different scales

Figure 4.7: Different scale results for infrared data

for the volunteers.

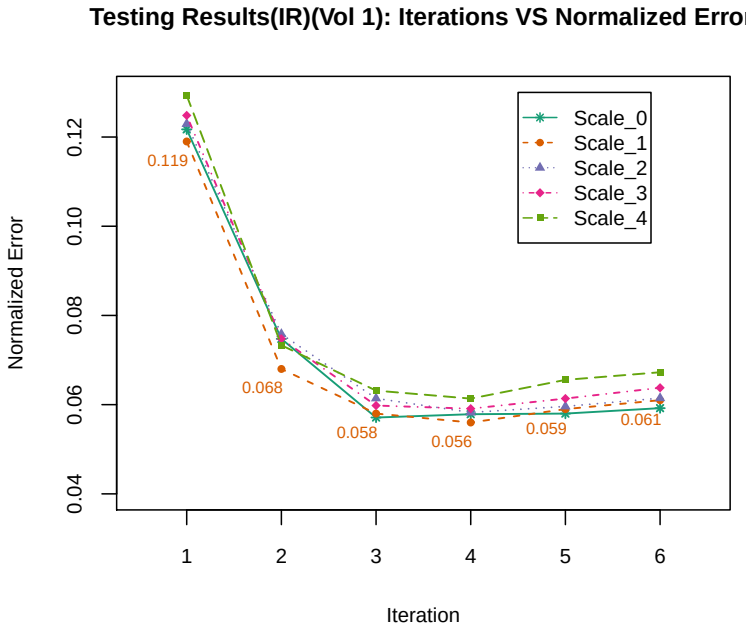


Figure 4.8: Iris center detection results for infrared data with different scales

Figure 4.8 is the results for *Volunteer 1* on five different scale values, the normalized error decreases with the iteration for all scale values, but *Scale_1* get the best accuracy after four iteration, with normalized error 0.056, shown in the bottom line.

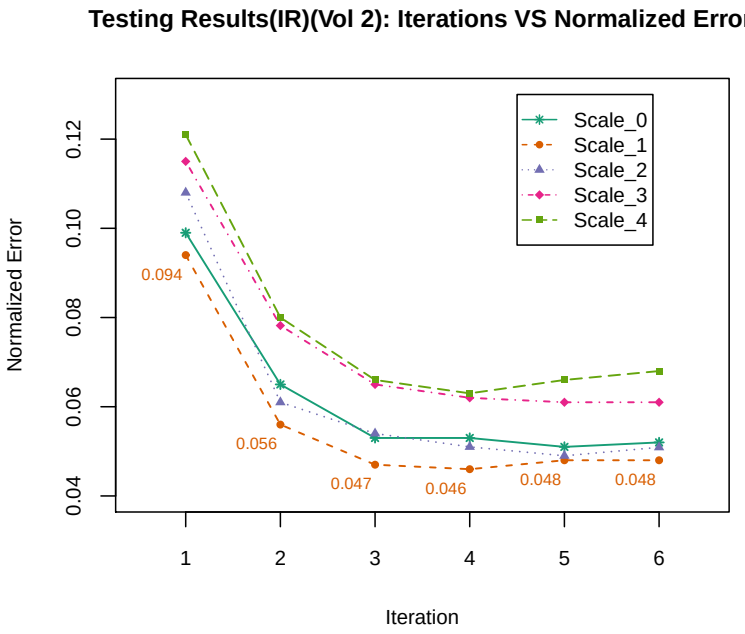


Figure 4.9: Iris center detection results for infrared data with different scales

Figure 4.9 illustrates the results for *Volunteer 2* on five different scale values, the normalized error decreases with the iteration for all scale values, but *Scale_1* get the best accuracy after four iterations, with normalized error 0.046, shown in the bottom line.

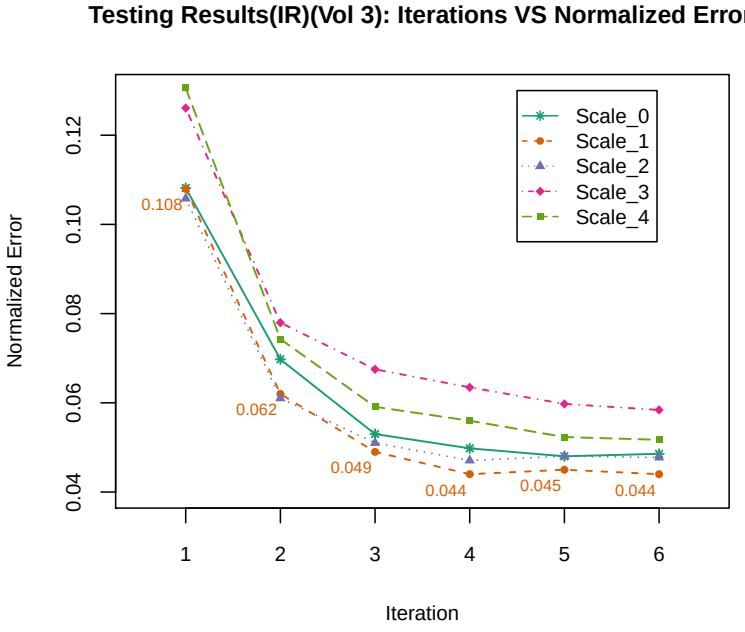


Figure 4.10: Iris center detection results for infrared data with different scales

Figure 4.10 shows the results for *Volunteer 3* on five different scale values, the normalized error decreases with the iteration for all scale values, but *Scale_1* get the best accuracy after four iterations, with normalized error 0.044, shown in the bottom line.

In conclusion, the results of three volunteers all get the best accuracy on *Scale_1* after iteration four with infrared data. The differences among different scale values are great for the last two volunteers compared to *Volunteer 1*. Therefore, *Scale_1* is applied, and the iteration number is set to four in the application.

4.3 Upper Eyelid Information Verification

Data obtained from section 4.1.3 is used to verify the upper eyelid information. The range of vertical direction is from $-40°$ to $25°$ for the original images. The average coefficient between two adjacent gaze points is obtained, which is corresponding to the average gaze direction of the two points. The relationship is learned from the images where the gaze has only vertical changes. Figure 4.11 shows the relationship between gaze directions and 2nd-order coefficient of eyelid curves. We can find the coefficient increase monotonically with the vertical gaze. Then the relationship is applied to the testing images where gaze has both vertical and horizontal changes. The result shows that vertical gaze error with upper-eyelid information is $5.45°$ while the vertical gaze error with only head pose information is $13.57°$. We can see the 2nd-order coefficient of eyelid curve plays an important role in gaze estimation.

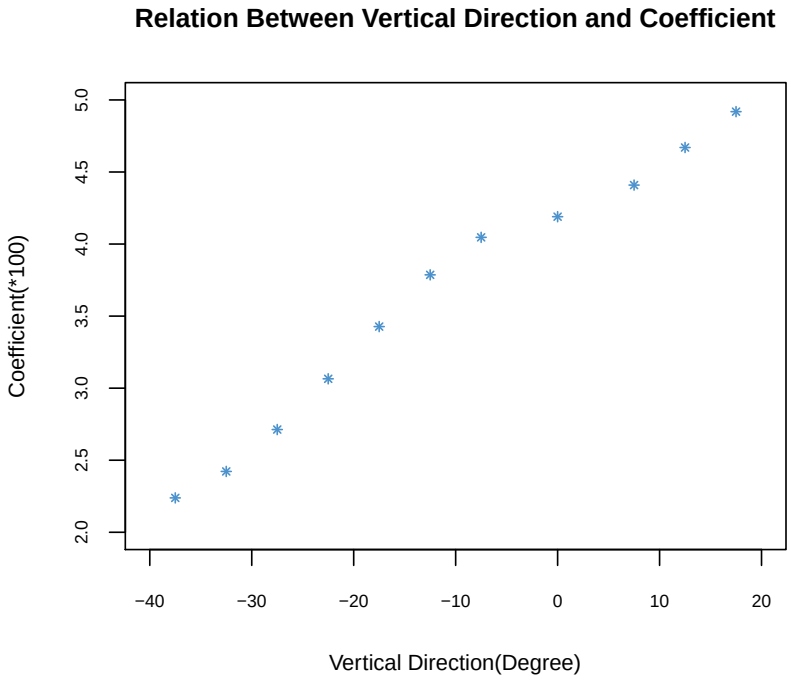


Figure 4.11: Relation between vertical direction and 2nd-order coefficient

4.4 Gaze Zone Estimation

In this thesis, gaze zones are classified into six main areas shown in Figure 4.12, namely: 1. Left Window 2. Right Window 3. Front Windowshield 4. Back Mirror 5. Dashboard, and 6. Center Console.

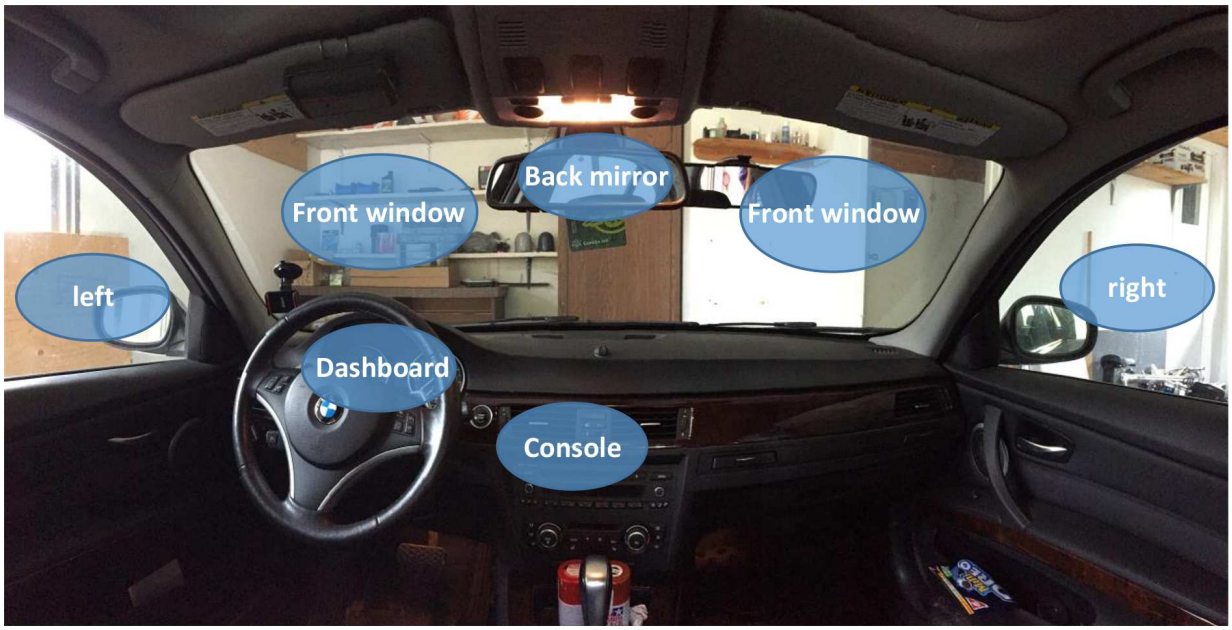


Figure 4.12: The distribution of six gaze zones

4.4.1 Evaluation Metrics and Methods

The gaze zone estimation method is employed to estimate the current gaze zone of the driver based on previous and current images captured by Kinect sensor positioned in the back of the windshield. Three main evaluation metrics are applied in the evaluation of the proposed method.

Metric One: Accuracy for Each Gaze Zone

If the sample number for gaze zone i is $N_i, i = 1, ..., 6$, and N_i^T samples in N_i are correctly estimated, then the accuracy for gaze zone i, A_i is defined as

$$A_i = \frac{N_i^T}{N_i}. \tag{4.2}$$

Metric Two: Weighted Accuracy

Weighted Accuracy takes the percentage of different gaze zone in the dataset into consideration. Weighted Accuracy $A_{weighted}$ is defined by equation 4.3,

$$A_{weighted} = \sum_{i=1}^{6} p_i \times A_i, \tag{4.3}$$

where p_i is the percentage of gaze zone i in the whole samples.

Metric Three: Unweighted Accuracy

Unweigted Accuracy treats all gaze zones with equal importance. The unweighted accuracy $A_{unweighted}$ is donated by the following equation:

$$A_{unweighted} = \frac{1}{6}\sum_{i=1}^{6} A_i, \tag{4.4}$$

which is equivalent to setting $p_i = \frac{1}{6}$ in equation 4.3.

In addition, the randomized ten-fold cross-validation method is applied in the evaluation. It is a validation method used to evaluate the generalization of a model to a dataset scientifically. The selection of training and testing dataset from the whole sample dataset is essential to the results of classification and may affect the accuracy, which will result in an unequal evaluation. Ten-fold cross-validation randomly splits the samples into ten subsets or folds with approximately same size. Every time, one of ten fold is selected as testing dataset while the other nine folds are for training. Repeat the step for ten times and the testing fold is different from each other in ten times. The training and testing set ratio is then 9:1. The final accuracy is denoted by the average accuracy of the results for ten times.

4.4.2 Gaze Zone Estimation Results in Daytime

The gaze zone estimation evaluation is applied to both gaze zone estimation methods with and without eye information in both daytime and nighttime with all the volunteers. The results for daytime and nighttime will be discussed respectively.

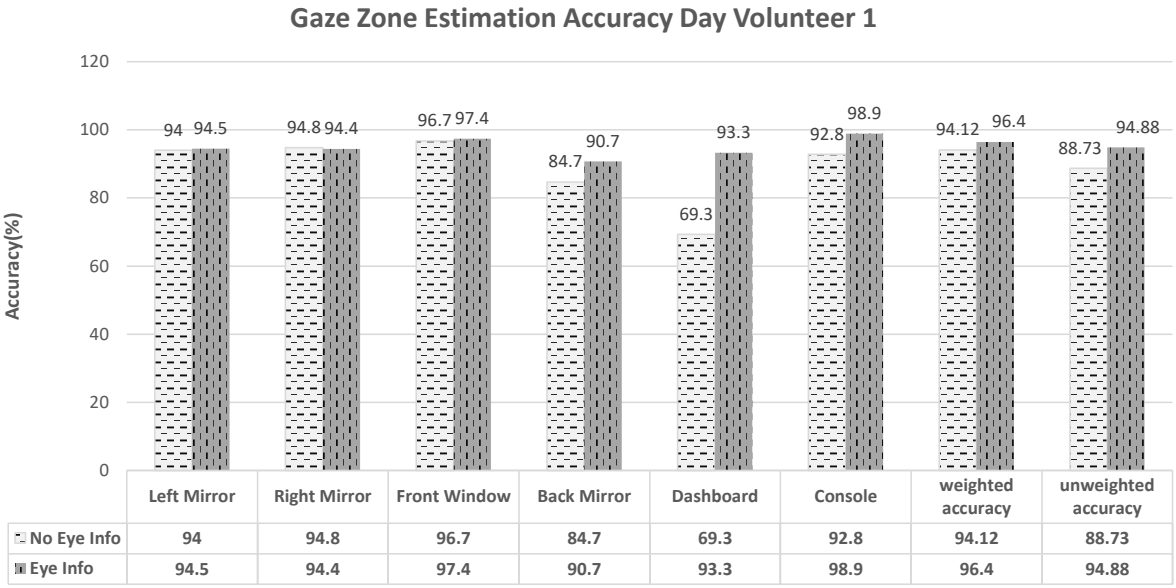


Figure 4.13: Gaze zone estimation accuracy in daytime for Volunteer 1

Figure 4.13 illustrates the gaze zone estimation accuracy in daytime for *Volunteer 1* with and without eye information. The first bar with shadow gray represents the method without eye info, which will be called *Method 1*. The second bar with dark gray represents the method with eye info, which will be called *Method 2*. X-axis describes the evaluation metrics mentioned before. The bar figure presentation will be the same for other volunteers. As shown in the figure, there is a great increase in accuracy for Dashboard when the eye info is used. The accuracy grows from 69.3% to 93.3%, increasing by 24%. The method making use of eye info can recognize zone Dashboard much better than the method, making no use of it, does. The growth of accuracy for zone Console is also great, from 92.8% to 98.9%, which is 6.1%. The weighted recognition accuracy for both methods are 94.12% and 96.40%, respectively, while the unweighted recognition accuracy are 88.73% and 94.88%, increasing by 2.28% and 6.15% when eye info is added. Considering the number of the trees utilized in RDF classifier, the number of trees used by *Method 2* is smaller.

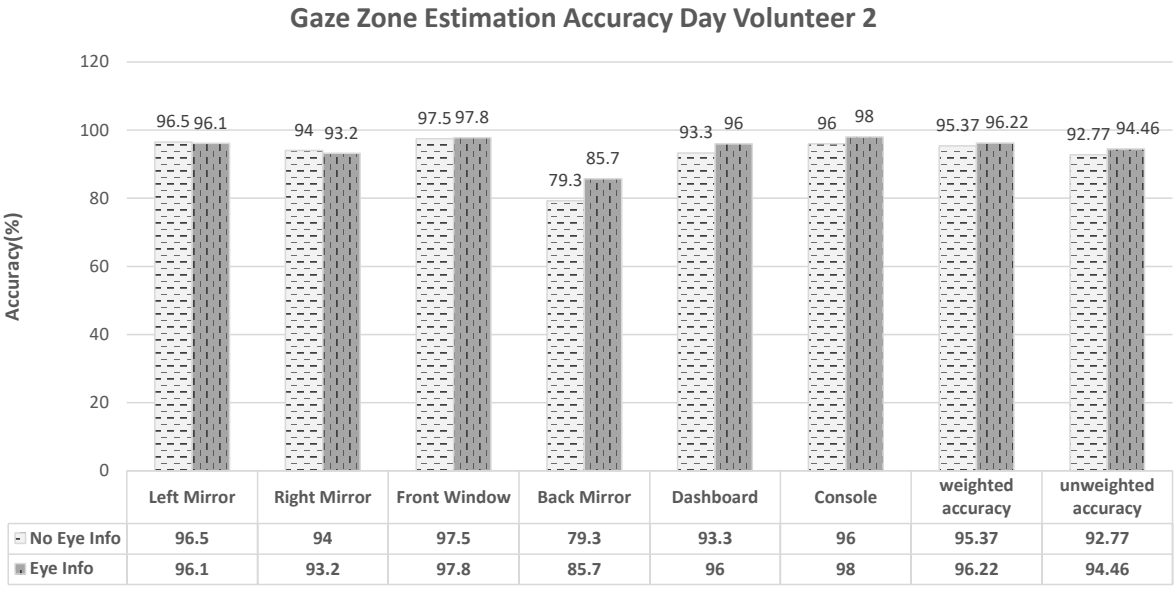


Figure 4.14: Gaze zone estimation accuracy in daytime for Volunteer 2

Figure 4.14 shows the gaze zone estimation accuracy in daytime for *Volunteer 2* with and without eye information. The greatest growth in accuracy is for Back Mirror, from 79.3% to 85.7%, increasing by 6.4%. The overall weighted and unweighted accuracy also increase, to some extent, which reach 96.22% and 94.46%, respectively. Moreover, only an average of 7 trees are utilized in RDF classifier when eye info is added while another method needs 12 trees.

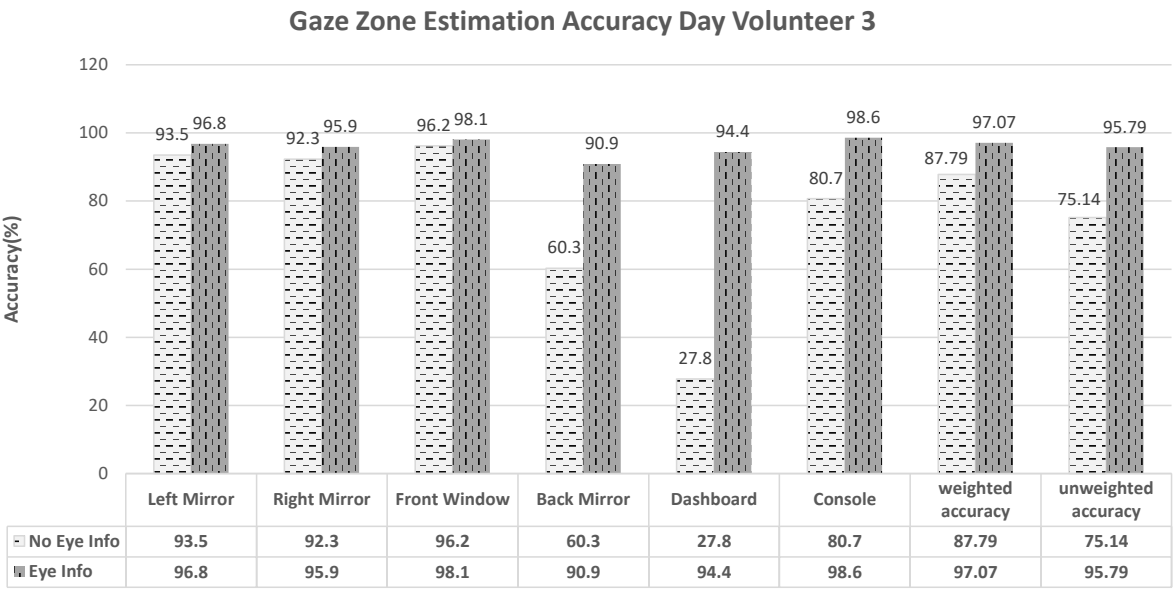
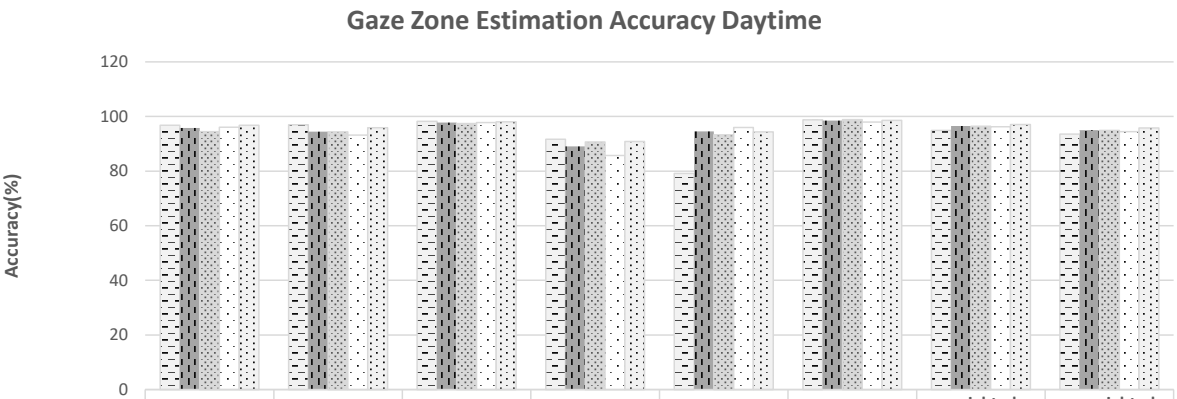


Figure 4.15: Gaze zone estimation accuracy in daytime for Volunteer 3

Figure 4.15 shows the gaze zone estimation accuracy in daytime for *Volunteer 3* with and without eye information. It is obvious that there is a huge increase for zones Back Mirror, Dashboard, and Console. Recognition accuracy increases about 30.6%, 66.6%, and 17.9% for these three gaze zones, respectively. Besides, *Method 2* with eye info has better overall performance compared to *Method 1*, with weighted and unweighted accuracy increasing by 9.28% and 20.65%, respectively. The overall weighted and unweighted accuracy for the *Method 2* reach 97.07% and 95.79%, respectively. The second method also applies fewer trees in RDF classifier.

Conclusion 1: Compared with *Method 1*, *Method 2* achieves better performance in recognition accuracy for all volunteers in the daytime. For some particular gaze zone Back Mirror, Dashboard and Center Console, the improvements of zone accuracy can even reach 66.6%. Three figures show similar improvements, which is influenced by the habit of driving and the way of observing road conditions and in-car instruments. For *Volunteer 3*, the huge increase in three gaze zones may be because that she prefers fixing her head pose, if she can make use of eye movement to reach a gaze zone. She will change her head pose only when the eye movement cannot help her to reach some observation areas. This observation habit makes eye information more critical in the estimation of drivers' gaze zone. Although the improvements differ to some degrees, the second method gains improvements in the overall weighted and unweighted accuracy.



	Left Mirror	Right Mirror	Front Window	Back Mirror	Dashboard	Console	weighted accuracy	unweighted accuracy
Day Reference	96.8	97	98.2	91.7	79.2	98.8	94.9	93.6
Average	95.80	94.50	97.77	89.10	94.57	98.50	96.56	95.04
Day Vol 1	94.5	94.4	97.4	90.7	93.3	98.9	96.4	94.88
Day Vol 2	96.1	93.2	97.8	85.7	96	98	96.22	94.46
Day Vol 3	96.8	95.9	98.1	90.9	94.4	98.6	97.07	95.79

Figure 4.16: Gaze zone estimation accuracy in daytime

Figure 4.16 illustrates the results of our gaze zone estimation method with color data from three volunteers, the average results of three volunteers, and the results of the reference paper. The reference results are taken from paper [129], which also conducts the experiment in the realistic driving environment with the same gaze zone distribution as ours. The sensor used in reference [129] is a color sensor. Compared with method used in the reference, our method shows improvements of performance in accuracy of gaze zone estimation. The overall weighted and unweighted accuracy is 94.9% and 93.6% for the reference while that is 96.56% and 95.04% for our method. Our method achieves about 1.66% and 1.44% increase. Although there is a little decrease for zone Right Mirror, the accuracy for this zone of our method can still reach 94.50%. For zone Dashboard, the accuracy of the reference reaches only 79.2% while that of our method can reach an average of 94.57%, which is an improvement of 15.37%. By the way, the average number of trees utilized in our method is 7 while that used in the reference is 60 for RDF classifier. In summary, the method proposed in this work has better performance than the reference does in the daytime.

4.4.3 Gaze Zone Estimation Results in Nighttime

After presenting the results in the daytime with color data, the following part will illustrate the results in the nighttime with infrared data. The performance for three volunteers are separated to show how eye information improves the performance, and the average results for the volunteers will be compared with that of the reference. The reference is the same as the one utilized in daytime situation, and it still uses color images, instead of the infrared data.

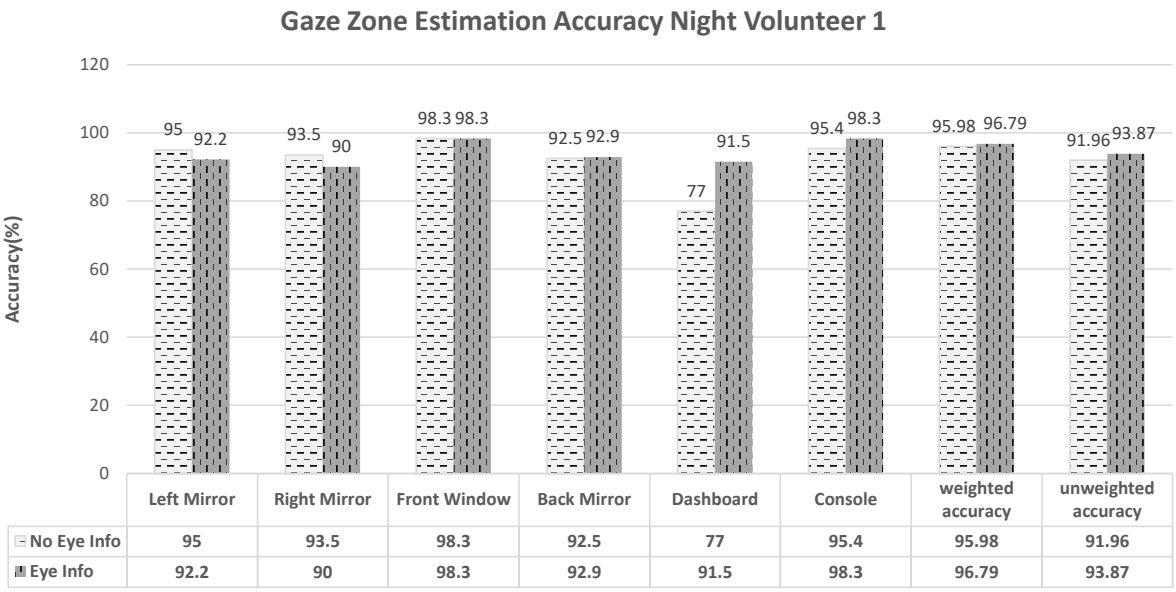


Figure 4.17: Gaze zone estimation accuracy in nighttime for Volunteer 1

Figure 4.17 shows the gaze zone estimation accuracy in the nighttime for *Volunteer 1* with and without eye information. The method with eye info gains great improvement for zone Dashboard with accuracy increasing from 77% to 91.5%, which is an increase of 14.5%. The overall weighted and unweighted accuracy obtain 0.81% and 1.91% increase, respectively. Therefore eye information indeed improves the performance of gaze zone estimation.

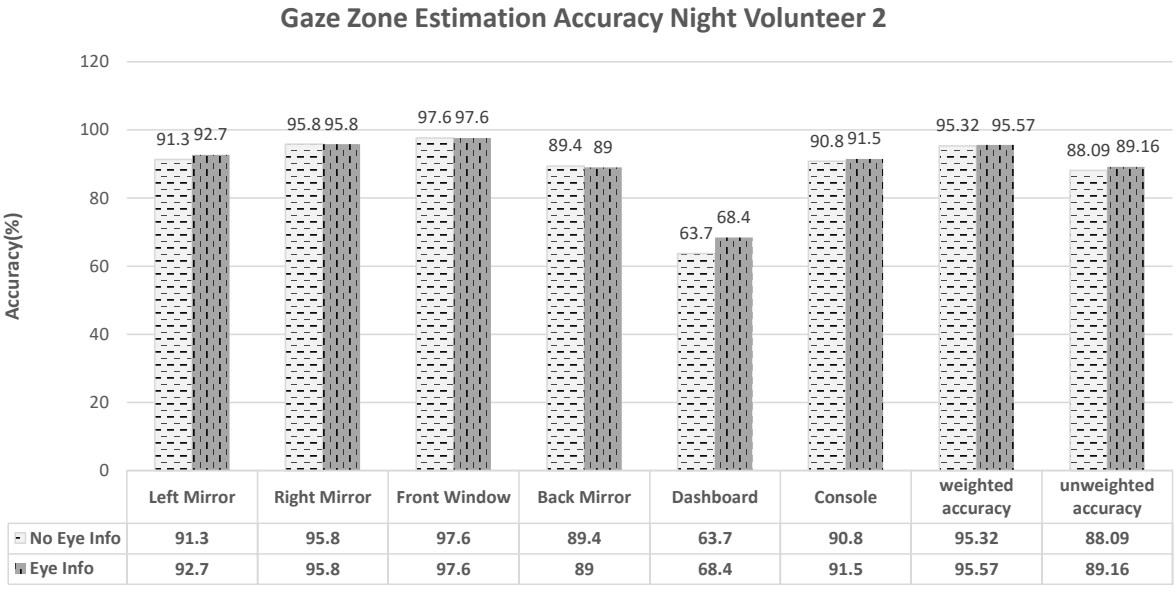


Figure 4.18: Gaze zone estimation accuracy in nighttime for Volunteer 2

Figure 4.18 presents the gaze zone estimation accuracy in the nighttime for *Volunteer*

2 with and without eye information. Although the overall weighted accuracy does not have significant improvement when eye information added, the improvement in recognition accuracy for zone Dashboard is great, with accuracy growing from 63.7% to 68.4%, increasing by 4.7%. In addition, the overall unweighted accuracy achieves an increase of 1.07%.

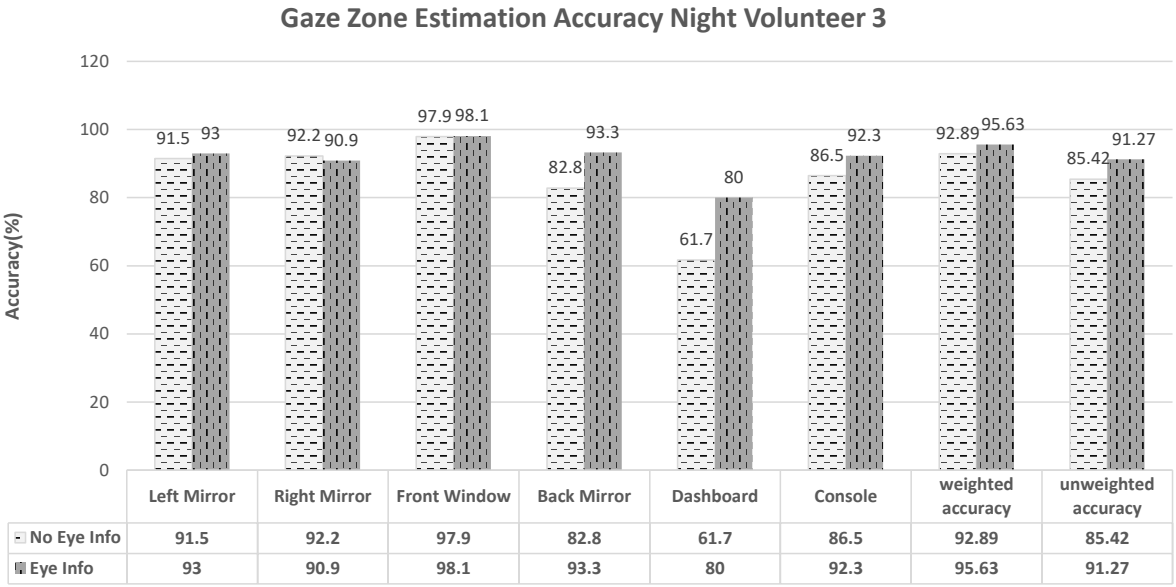
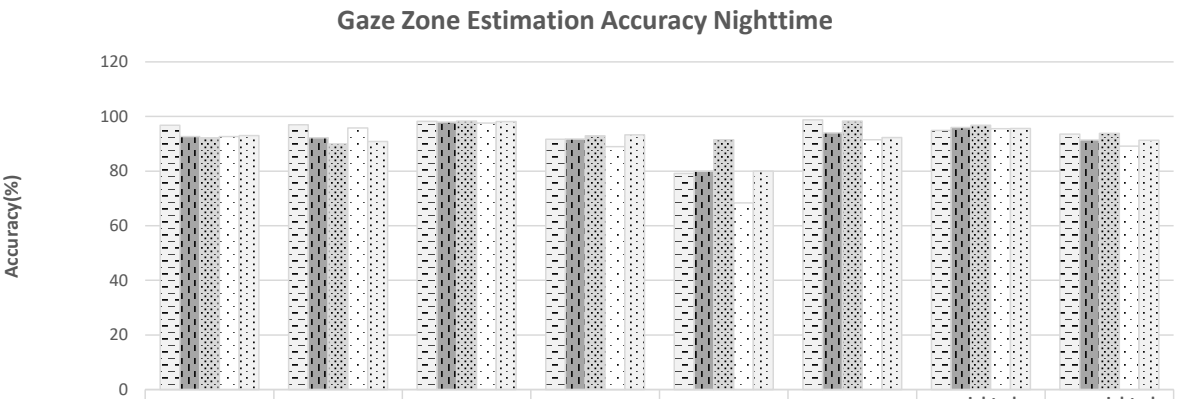


Figure 4.19: Gaze zone estimation accuracy in nighttime for Volunteer 3

Figure 4.19 illustrates the gaze zone estimation accuracy in the nighttime for *Volunteer 3* with and without eye information. As shown in the figure, there are huge improvements in zone accuracy for Back Mirror, Dashboard, and Console. The zone accuracy increases by 10.5%, 18.3%, and 5.8%, respectively. The overall weighted accuracy improves from 92.89% to 95.63%, and the unweighted accuracy grows from 85.42% to 91.27%, increasing by 2.74% and 5.85% respectively.

Conclusion 2: Compared with method without eye info, the method making use of eye information obtains better performance in recognition accuracy for all volunteers in the nighttime. For some particular gaze zone Back Mirror, Dashboard and Center Console, the improvements in zone accuracy can even reach 18.3%. Three figures show similar improvements, which is similar to the results in the daytime, and the reason is analyzed in above parts. The habit of driver's driving behavior and the way of their observation on the driving condition are the principle reasons for the difference. In conclusion, the performance is improved when eye information is exploited in the gaze zone estimation.



	Left Mirror	Right Mirror	Front Window	Back Mirror	Dashboard	Console	weighted accuracy	unweighted accuracy
Day Reference	96.8	97	98.2	91.7	79.2	98.8	94.9	93.6
Average	92.63	92.23	98.00	91.73	79.97	94.03	96.00	91.43
Night Vol 1	92.2	90	98.3	92.9	91.5	98.3	96.79	93.87
Night Vol 2	92.7	95.8	97.6	89	68.4	91.5	95.57	89.16
Night Vol 3	93	90.9	98.1	93.3	80	92.3	95.63	91.27

Figure 4.20: Gaze zone estimation accuracy in nighttime

Figure 4.20 illustrates our gaze zone estimation results with infrared data of three volunteers, the average results of three volunteers, and the results obtained in daytime in the reference [129]. Although the results with infrared data in nighttime decline to some degree compared to that with color data in daytime in the reference, the weighted and unweighted accuracy of the former can also reach 96.0% and 91.43%. The results are acceptable since the quality of the image has an influence on the results. The night environment makes the detection and estimation more difficult, and the infrared data is not able to provide adequate details in comparison with the color data obtained in the daytime.

4.4.4 Gaze Zone Estimation Implementation

The application for gaze zone estimation method proposed is implemented by Visual Studio 2013 IDE, and C++ is utilized as the programming language. The data stream is achieved from Kinect V2, and the results, such as facial landmarks, will be shown in a window. Here are some demo results from the output of the application, which demonstrate how the application work for both infrared data and color data.

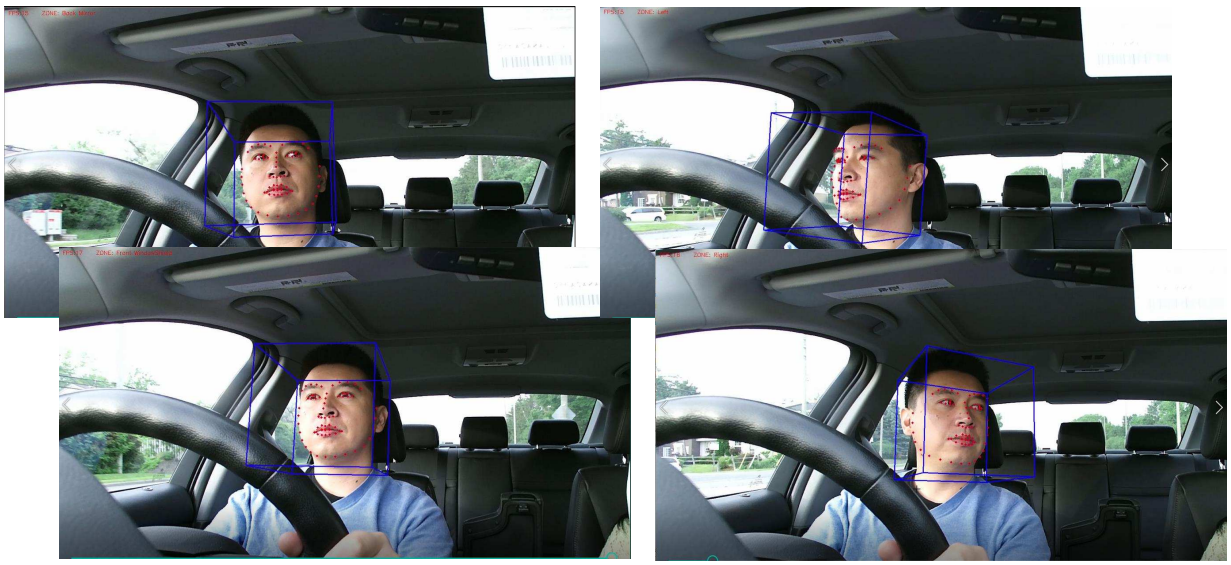


Figure 4.21: Gaze zone estimation demo in daytime

Figure 4.21 are some pictures captured from the results of the application for color data. The detected facial landmarks are shown by the red points, and the current head pose is illustrated by the blue 3D box. At the top of the frame, there are some texts describing the current gaze zone of the driver and the framerate with unit frame/second. For color data, with resolution 1920×1080, the framerate is about 15-17 fps, which is acceptable and near real-time.

Figure 4.22: Gaze zone estimation demo in nighttime

Figure 4.21 demonstrates some captures from the results of the application for infrared

data. There is some information at the top left of the window which demonstrates the current gaze zone of the driver and the framerate with unit frame/second (fps). The framerate can reach 30 fps, because the resolution of infrared data is only 512×424, which is much smaller than that of color data.

4.5 Conclusion

This chapter discussed four main parts of the experiments in detail, namely the experiment setup and dataset, results for iris center detection, upper eyelid information verification, and results for gaze zone estimation. The experiments are carried out in the realistic driving environment in both daytime and nighttime with three volunteers. Kinect for Windows V2 sensor is applied in the experiments, put right behind the windshield. Two datasets are collected, which contain 29,000 and 2,100 annotated images for gaze zone estimation and iris center detection, respectively. The two datasets can be exploited by other researchers in the future. Compared with the results of iris detection in the wild dataset, the detection results trained and tested for individual volunteers show better performance. Compared with gaze zone estimation method with head information only, the method with eye information has a significant increase in accuracy. Finally, the proposed gaze zone estimation framework, whose weighted and unweighted accuracy reaches 96.6% and 95.0%, respectively for daytime, has better performance compared to the reference. For nighttime, the unweighted accuracy can reach 91.4%, and the weighted can reach 96%.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, we introduced a framework for driver's gaze zone estimation composed of five main components: Facial Landmark Detection, Head Pose Estimation, Iris Center Detection, Upper Eyelid Information Extraction, and Gaze Zone Estimation. The experiment is carried out in the realistic driving environment in both daytime and nighttime with three volunteers using Kinect sensor V2 for Windows, which is put at the back of windshield. Experiment results show that the proposed gaze zone estimation framework has better performance compared to that of the reference.

Chapter 1 described the background of ADASs, our motivation and contribution. Driver's distraction and drowsiness are two major causes of fatal and injury crashes. Therefore, driver's distraction and drowsiness detection have become two main subjects, which researchers are focusing on, in ADAS design. Head pose and eye-gaze direction are two strong indicators of a driver's gaze and the current focus of attention. We proposed a new driver's gaze zone estimation framework which combines head and eye information to estimate the gaze zone of the driver in both daytime and nighttime.

Chapter 2 made a literature review composed of five sections, namely video-based gaze estimation, head pose detection, iris center detection, facial landmark detection, and classification techniques. The naturalistic driving environment always presents unique challenges to video-based gaze estimation and tracking systems. In an automobile, the continuously changed lighting condition, resulting in heavy shadows and high illumination variability, and unstable capture of data, caused by jolts, all present more difficulties to gaze estimation task. Moreover, performing in real time or near-real time is another consideration for gaze estimation in an ADAS. Therefore, the results tested in the stationary laboratory with stable illumination have higher proficiency than those in the wild. However, these methods may work poorly in the driving environment. Unique approaches are needed to design for ADASs to overcome these unique difficulties.

Chapter 3 presented the proposed framework for gaze zone estimation in detail following the framework. Firstly, CLNF facial landmark detection method, a CLM instance that uses

LNF local detector and NU-RLMS fitting method, is applied to obtain the facial landmarks in the image plane and the 3D model of the face in the object frame. Extracting region of interest is utilized as an optimization strategy for CLNF facial landmark detection. Secondly, Levenberg-Marquardt optimization method is used to solve the PnP problem, which estimates the head pose from the 2D landmark locations in the image plane and their corresponding 3D locations in the object frame. Next, a regression model-based method is employed to obtain the iris center from eye landmarks detected in the previous part. For upper eyelid information extraction, a quadratic function is utilized to model the upper eyelid, and the second-order coefficient is extracted. Finally, the head pose and the eye information are combined to form a feature vector, and RDF classifier is utilized to estimate the current gaze zone of the driver from the feature vector extracted.

Chapter 4 described the setup of experiment, the dataset, and the experiment results composed of three main parts, namely results of iris center detection, upper eyelid information verification, and gaze zone estimation. Two datasets are collected, which contain 29,000 and 2,100 annotated images for gaze zone estimation and iris center detection, respectively. The two datasets can be exploited by other researchers in the future. Compared with the results of iris detection in the wild dataset, the detection results trained and tested for individual volunteers show better performance. Compared with gaze zone estimation method with head information only, the method with eye information has a significant increase in accuracy. Finally, the proposed gaze zone estimation framework, whose weighted and unweighted accuracy reaches 96.6% and 95.0%, respectively for daytime, has better performance compared to the reference. For nighttime, the unweighted accuracy can reach 91.4%, and the weighted can reach 96%, which also meets the expectation.

5.2 Future Work

Although the framework proposed can achieve good performance for driver's gaze zone estimation, some future work can also be considered for the improvements and extension of our work.

As the quality of the image provided by the sensor has influence on the speed and estimation accuracy of the gaze zone estimation method, the choice of the sensor also plays an important role. The higher the resolution of the images, the better the estimation is, but the slower the processing speed is. A good choice of the sensor can obtain the expected speed and estimation accuracy. The choice of sensor can be taken into consideration in the experiment setup. In addition, the performance of the method may be affected by different environment, such as different weather and different road conditions. More different environment can be considered in the collection of dataset and evaluation of the method. In our experiments, only the situation without sunglass is considered. In the situation where the driver wears a sunglass, the eye information will be occluded by the sunglass and can not be detected. Therefore, a sunglass detection method can be applied before gaze zone estimation to detect whether the driver is wearing a sunglass. If a driver wears a sunglass, only head information will be employed for gaze estimation [136].

The detected facial landmarks and eye information can further be utilized in drowsiness detection analysis, such as eye state analysis, eye blinking analysis using upper-eyelid, mouth and yawning analysis using contour of mouth [76]. In addition, gaze zone estimation can be combined with vehicle parameters, such as steering wheel movement, lane keeping, acceleration pedal movement, and braking, to detect whether a driver is looking at the right area in the right scene. The system will give alerts to the driver when danger situation may occur because of driver's distraction or drowsiness. The dangerous alerts can also be broadcast to near drivers through vehicle networks like Vehicle Ad-Hoc Networks ([5],[47],[113],[27],[140],[21],[25],[54],[26],[4],[30],[28],[55],[17],[150],[18]) to inform them of the dangerous state of this driver. When getting such alerts, those drives will be more careful about this car and then reduce the probability of traffic accidents. Gaze zone estimation can also be assocaited with other detection, such as pedestrian detection or animal detection, to detect whether the driver notices these information. Therefore, the camera used for driver monitoring can be combined with other sensors and integrated in wireless sensor networks ([23],[49],[141],[24],[22],[7],[138],[139],[108],[29],[137],[16],[118],[20],[19]). The combination of these sensors and detection can improve the functions of an ADAS.

References

[1] Kinect for windows sdk 2.0. [Online]. Available: https://www.microsoft.com/en-us/download/details.aspx?id=44561. Accessed on: March, 2018.

[2] Kinect studio. [Online]. Available: https://msdn.microsoft.com/en-us/library/dn785306.aspx. Accessed on: March, 2018.

[3] Opencv library. [Online]. Available: https://opencv.org/. Accessed on: March, 2018.

[4] Kaouther Abrougui, Azzedine Boukerche, and Richard Werner Nelem Pazzi. Design and evaluation of context-aware and location-based service discovery protocols for vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):717–735, 2011.

[5] Osama Abumansoor and Azzedine Boukerche. A secure cooperative approach for nonline-of-sight location verification in vanet. *IEEE Transactions on Vehicular Technology*, 61(1):275–285, 2012.

[6] Mohamed Aly. Survey on multiclass classification methods. *Neural Networks*, pages 1–9, 2005.

[7] Thanasis Antoniou, Ioannis Chatzigiannakis, George Mylonas, Sotiris Nikoletseas, and Azzedine Boukerche. A new energy efficient and fault-tolerant protocol for data propagation in smart dust networks using varying transmission range. In *Proceedings of ANSS'04*, page 43. IEEE Computer Society, 2004.

[8] Nicholas Ayache. *Vision stéréoscopique et perception multisensorielle*. Inter-Editions, 1989.

[9] Tadas Baltrušaitis. *Automatic facial expression analysis*. PhD thesis, University of Cambridge, 2014.

[10] Tadas Baltrušaitis, Peter Robinson, and Louis-Philippe Morency. 3d constrained local model for rigid and non-rigid facial tracking. In *Proceedings of CVPR'12*, pages 2610–2617. IEEE, 2012.

[11] Tadas Baltrusaitis, Peter Robinson, and Louis-Philippe Morency. Constrained local neural fields for robust facial landmark detection in the wild. In *Proceedings of ICCVW'13*, pages 354–361. IEEE, 2013.

[12] Shumeet Baluja and Dean Pomerleau. Non-intrusive gaze tracking using artificial neural networks. In *Proceedings of NIPS'94*, pages 753–760. Morgan-Kaufmann, 1994.

[13] Stephen D Bay. Combining nearest neighbor classifiers through multiple feature subsets. In *Proceedings of ICML'98*, volume 98, pages 37–45. Citeseer, 1998.

[14] Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[15] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of SIGGRAPH'99*, pages 187–194. ACM Press/Addison-Wesley Publishing Co., 1999.

[16] Azzedine Boukerche, Regina B Araujo, and Leandro Villas. A wireless actor and sensor networks qos-aware routing protocol for the emergency preparedness class of applications. In *Proceedings of LCN'06*, pages 832–839. IEEE, 2006.

[17] Azzedine Boukerche and Amir Darehshoorzadeh. Opportunistic routing in wireless networks: Models, algorithms, and classifications. *ACM Computing Surveys*, 47(2):1–32, 2015.

[18] Azzedine Boukerche, Sajal K Das, and Alessandro Fabbri. Swimnet: a scalable parallel simulation testbed for wireless and mobile networks. *Wireless Networks*, 7(5):467–486, 2001.

[19] Azzedine Boukerche and Xin Fei. A coverage-preserving scheme for wireless sensor network with irregular sensing range. *Ad hoc networks*, 5(8):1303–1316, 2007.

[20] Azzedine Boukerche, Xin Fei, and Regina B Araujo. An optimal coverage-preserving scheme for wireless sensor networks based on local information exchange. *Computer Communications*, 30(14-15):2708–2720, 2007.

[21] Azzedine Boukerche, Sungbum Hong, and Tom Jacob. An efficient synchronization scheme of multimedia streams in wireless and mobile systems. *IEEE transactions on Parallel and Distributed Systems*, 13(9):911–923, 2002.

[22] Azzedine Boukerche, Anahit Martirosyan, and Richard Pazzi. An inter-cluster communication based energy aware and fault tolerant protocol for wireless sensor networks. *Mobile Networks and Applications*, 13(6):614–626, 2008.

[23] Azzedine Boukerche, Horacio ABF Oliveira, Eduardo F Nakamura, and Antonio AF Loureiro. Localization systems for wireless sensor networks. *IEEE wireless Communications*, 14(6), 2007.

[24] Azzedine Boukerche, Richard Werner N Pazzi, and Regina B Araujo. Hpeq a hierarchical periodic, event-driven and query-based wireless sensor network protocol. In *Proceedings of LCN'05*, pages 560–567. IEEE, 2005.

[25] Azzedine Boukerche and Yonglin Ren. A security management scheme using a novel computational reputation model for wireless and mobile ad hoc networks. In *Proceedings of PE-WASUN'08*, pages 88–95. ACM, 2008.

[26] Azzedine Boukerche, Cristiano Rezende, and Richard W Pazzi. Improving neighbor localization in vehicular ad hoc networks to avoid overhead from periodic messages. In *Proceedings of GLOBECOM'09*, pages 1–6. IEEE, 2009.

[27] Azzedine Boukerche and Steve Rogers. Gps query optimization in mobile and wireless networks. In *Proceedings of ISCC'01*, pages 198–203. IEEE, 2001.

[28] Azzedine Boukerche and Samer Samarah. A novel algorithm for mining association rules in wireless ad hoc sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 19(7):865–877, 2008.

[29] Azzedine Boukerche and Damla Turgut. Secure time synchronization protocols for wireless sensor networks. *IEEE Wireless Communications*, 14(5), 2007.

[30] Azzedine Boukerche, Anis Zarrad, and Regina B Araujo. A cross-layer approach-based gnutella for collaborative virtual environments over mobile ad hoc networks. *IEEE Transactions on Parallel & Distributed Systems*, (7):911–924, 2009.

[31] Martin Breidt, Heinrich H Biilthoff, and Cristóbal Curio. Robust semantic analysis by synthesis of 3d facial motion. In *Proceedings of FG'11*, pages 713–719. IEEE, 2011.

[32] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[33] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[34] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. Wadsworth, 1984.

[35] Xavier LC Brolly and Jeffrey B Mulligan. Implicit calibration of a remote gaze tracker. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on*, pages 134–134. IEEE, 2004.

[36] Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. Face alignment by explicit shape regression. *International Journal of Computer Vision*, 107(2):177–190, 2014.

[37] Jixu Chen and Qiang Ji. 3d gaze estimation with a single camera without ir illumination. In *Proceedings of ICPR'08*, pages 1–4. IEEE, 2008.

[38] HR Chennamma and Xiaohui Yuan. A survey on eye-gaze tracking techniques. *Indian Journal of Computer Science and Engineering*, 4(5):388–393, 2013.

[39] Carlo Colombo and Alberto Del Bimbo. Real-time head tracking from the deformation of eye contours using a piecewise affine camera. *Pattern Recognition Letters*, 20(7):721–730, 1999.

[40] Tim F Cootes, Mircea C Ionita, Claudia Lindner, and Patrick Sauer. Robust and accurate shape model fitting using random forest regression voting. In *Proceedings of ECCV'12*, pages 278–291. Springer, 2012.

[41] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[42] Timothy F Cootes and Christopher J Taylor. Active shape models'smart snakes'. In *Proceedings of BMVC'92*, pages 266–275. Springer, 1992.

[43] Timothy F Cootes, Christopher J Taylor, et al. Statistical models of appearance for computer vision, 2004.

[44] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[45] Canada Safe Council. Driver fatigue: Falling asleep at the wheel. [Online]. Available: https://canadasafetycouncil.org/safety-canada-online/article/driver-fatigue-falling-asleep-wheel, 2009. Accessed on: March, 2018.

[46] David Cristinacce and Timothy F Cootes. Feature detection and tracking with constrained local models. In *Proceedings of BMVC'06*, volume 1, page 3, 2006.

[47] Felipe Cunha, Leandro Villas, Azzedine Boukerche, Guilherme Maia, Aline Viana, Raquel AF Mini, and Antonio AF Loureiro. Data communication in vanets: Protocols, applications and challenges. *Ad Hoc Networks*, 44:90–103, 2016.

[48] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of CVPR'05*, volume 1, pages 886–893. IEEE, 2005.

[49] Horacio Antonio Braga Fernandes De Oliveira, Azzedine Boukerche, Eduardo Freire Nakamura, and Antonio Alfredo Ferreira Loureiro. An efficient directed localization recursion protocol for wireless sensor networks. *IEEE Transactions on Computers*, 14(5):677–691, 2008.

[50] Daniel F Dementhon and Larry S Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1):123–141, 1995.

[51] Ramón Díaz-Uriarte and Sara Alvarez De Andres. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7(1):3, 2006.

[52] Thomas G Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[53] Andrew T Duchowski. Eye tracking methodology. *Theory and Practice*, 328, 2007.

[54] Mourad Elhadef, Azzedine Boukerche, and Hisham Elkadiki. Performance analysis of a distributed comparison-based self-diagnosis protocol for wireless ad-hoc networks. In *Proceedings of MSWiM'06*, pages 165–172. ACM, 2006.

[55] Mourad Elhadef, Azzedine Boukerche, and Hisham Elkadiki. A distributed fault identification protocol for wireless and mobile ad hoc networks. *Journal of Parallel and Distributed Computing*, 68(3):321–335, 2008.

[56] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in Computer Vision*, pages 726–740. Elsevier, 1987.

[57] Pia M Forsman, Bryan J Vila, Robert A Short, Christopher G Mott, and Hans PA Van Dongen. Efficient driver drowsiness detection at moderate levels of drowsiness. *Accident Analysis & Prevention*, 50:341–350, 2013.

[58] Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *Proceedings of ICML'96*, pages 148–156. Morgan Kaufmann, 1996.

[59] Jerome Friedman. Another approach to polychotomous classification. Technical report, Technical report, Department of Statistics, Stanford University, 1996.

[60] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000.

[61] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

[62] Xinbo Gao, Ya Su, Xuelong Li, and Dacheng Tao. A review of active appearance models. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(2):145–158, 2010.

[63] Er Manoram Vats1and Er Anil Garg. Detection and security system for drowsy driver by using artificial neural network technique. *International Journal of Applied Science and Advance Technology*, 1(1):39–43, 2012.

[64] Leon Gu and Takeo Kanade. A generative shape regularization model for robust face alignment. In *Proceedings of ECCV'08*, pages 413–426. Springer, 2008.

[65] Elias Daniel Guestrin and Moshe Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *IEEE Transactions on Biomedical Engineering*, 53(6):1124–1133, 2006.

[66] Zhibo Guo, Huajun Liu, Qiong Wang, and Jingyu Yang. A fast algorithm face detection and head pose estimation for driver assistant system. In *Proceedings of ICSP'06*, volume 3. IEEE, 2006.

[67] Peter W Hallinan. Recognizing human eyes. In *Proceedings of SPIE'91*, volume 1570, pages 214–227. International Society for Optics and Photonics, 1991.

[68] Sang Yoon Han, Insung Hwang, Sang Hwa Lee, and Nam Ik Cho. Gaze estimation using 3-d eyeball model and eyelid shapes. In *Proceedings of APSIPA'16*, pages 1–4. IEEE, 2016.

[69] Dan Witzner Hansen and Qiang Ji. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):478–500, 2010.

[70] Antonio Haro, Myron Flickner, and Irfan Essa. Detecting and tracking eyes by using their physiological properties, dynamics, and appearance. In *Proceedings of CVPR'00*, volume 1, pages 163–168. IEEE, 2000.

[71] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In *Proceedings of NIPS'98*, pages 507–513, 1998.

[72] A Geert Heidema, Jolanda MA Boer, Nico Nagelkerke, Edwin CM Mariman, Edith JM Feskens, et al. The challenge for genetic epidemiologists: how to analyze large numbers of snps in relation to complex diseases. *BMC Genetics*, 7(1):23, 2006.

[73] Chen Huang, Xiaoqing Ding, and Chi Fang. Pose robust face tracking by combining view-based aams and temporal filters. *Computer Vision and Image Understanding*, 116(7):777–792, 2012.

[74] LáSzló A Jeni, András Lőrincz, Tamás Nagy, Zsolt Palotai, Judit Sebők, Zoltán Szabó, and Dániel Takács. 3d shape estimation in video sequences provides high precision evaluation of facial expressions. *Image and Vision Computing*, 30(10):785–795, 2012.

[75] Li Jianfeng and Li Shigang. Eye-model-based gaze estimation by rgb-d camera. In *Proceedings of CVPRW'14*, pages 592–596. IEEE, 2014.

[76] Sinan Kaplan, Mehmet Amac Guvensan, Ali Gokhan Yavuz, and Yasin Karalurt. Driver behavior analysis for safe driving: a survey. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):3017–3032, 2015.

[77] Kyung-Nam Kim and RS Ramakrishna. Vision-based eye-gaze tracking for human computer interface. In *Proceedings of SMC'99*, volume 2, pages 324–329. IEEE, 1999.

[78] Sheila G Klauer, Thomas A Dingus, Vicki L Neale, Jeremy D Sudweeks, David J Ramsey, et al. The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data. 2006.

[79] Chih-Chuan Lai, Yu-Ting Chen, Kuan-Wen Chen, Shen-Chi Chen, Sheng-Wen Shih, and Yi-Ping Hung. Appearance-based gaze tracking with free head movement. In *Proceedings of ICPR'14*, pages 1869–1873. IEEE, 2014.

[80] Kin-Man Lam and Hong Yan. Locating and extracting the eye in human face images. *Pattern Recognition*, 29(5):771–779, 1996.

[81] Stephen RH Langton, Helen Honeyman, and Emma Tessler. The influence of head contour and nose angle on the perception of eye-gaze direction. *Perception and Psychophysics*, 66(5):752–771, 2004.

[82] Sung Joo Lee, Jaeik Jo, Ho Gi Jung, Kang Ryoung Park, and Jaihie Kim. Real-time gaze estimator based on driver's head orientation for forward collision warning system. *IEEE Transactions on Intelligent Transportation Systems*, 12(1):254–267, 2011.

[83] Artem A Lenskiy and Jong-Soo Lee. Drivers eye blinking detection using novel color and texture segmentation algorithms. *International Journal of Control, Automation and Systems*, 10(2):317–327, 2012.

[84] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

[85] Dongheng Li, David Winfield, and Derrick J Parkhurst. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. In *Proceedings of CVPRW'05*, pages 79–79. IEEE, 2005.

[86] Ming-ai Li, Cheng Zhang, and Jin-Fu Yang. An eeg-based method for detecting drowsy driving state. In *Proceedings of FSKD'10*, volume 5, pages 2164–2167. IEEE, 2010.

[87] Xiaoming Liu. Generic face alignment using boosted appearance model. In *Proceedings of CVPR'07*, pages 1–8. IEEE, 2007.

[88] C-P Lu, Gregory D Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):610–622, 2000.

[89] Ping Luo, Xiaogang Wang, and Xiaoou Tang. Hierarchical face parsing via deep learning. In *Proceedings of CVPR'12*, pages 2480–2487. IEEE, 2012.

[90] Milad Malekipirbazari and Vural Aksakalli. Risk assessment in social lending via random forests. *Expert Systems with Applications*, 42(10):4621–4631, 2015.

[91] Sujitha Martin, Ashish Tawari, Erik Murphy-Chutorian, Shinko Y Cheng, and Mohan Trivedi. On the design and evaluation of robust head pose for visual user interfaces: Algorithms, databases, and comparisons. In *Proceedings of AutomotiveUI'12*, pages 149–154. ACM, 2012.

[92] John Merchant, Richard Morrissette, and James L Porterfield. Remote measurement of eye direction allowing subject motion over one cubic foot of space. *IEEE Transactions on Biomedical Engineering*, 21(4):309–317, 1974.

[93] Baback Moghaddam and Alex Pentland. Probabilistic visual learning for object detection. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 786–793. IEEE, 1995.

[94] Hiroshi Murase and Shree K Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 14(1):5–24, 1995.

[95] Erik Murphy and Mohan Manubhai Trivedi. Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):300–311, 2010.

[96] Erik Murphy-Chutorian and Mohan M Trivedi. 3d tracking and dynamic analysis of human head movements and attentional targets. In *Proceedings of ICDSC'08*, pages 1–8. IEEE, 2008.

[97] Erik Murphy-Chutorian and Mohan Manubhai Trivedi. Hyhope: Hybrid head orientation and position estimation for vision-based driver head tracking. In *Proceedings of IV'08*, pages 512–517. IEEE, 2008.

[98] Erik Murphy-Chutorian and Mohan Manubhai Trivedi. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):607–626, 2009.

[99] Sameer A Nene and Shree K Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.

[100] NHTSA. Overview of the national highway traffic safety administrations driver distraction program. [Online]. Available: http://www.nhtsa.gov/staticfiles/nti/distracted_driving/pdf/811299.pdf, 2010. Accessed on: March, 2018.

[101] NHTSA. Traffic safety fact crash stats: Drowsy driving. [Online]. Available: www-nrd.nhtsa.dot.gov/pubs/811449.pdf, 2011. Accessed on: March, 2018.

[102] NHTSA. Traffic safety facts: Distracted driving 2012. [Online]. Available: http://www-nrd.nhtsa.dot.gov/Pubs/812012.pdf, 2014. Accessed on: March, 2018.

[103] NHTSA. Traffic safety facts: Distracted driving 2014. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812260, 2016. Accessed on: March, 2018.

[104] Kevin Nickels and Seth Hutchinson. Estimating uncertainty in ssd-based feature tracking. *Image and Vision Computing*, 20(1):47–58, 2002.

[105] Mark Nitzberg and Takahiro Shiota. Nonlinear image smoothing with edge and corner enhancement. Technical report, Technical Report, Harvard Robotics Laboratory, 1990.

[106] Mark Nixon. Eye spacing measurement for facial recognition. In *Proceedings of SPIE'85*, volume 575, pages 279–285. International Society for Optics and Photonics, 1985.

[107] Kenichi Ohue, Yukinori Yamada, Shigeyasu Uozumi, Setsuo Tokoro, Akira Hattori, and Takeshi Hayashi. Development of a new pre-crash safety system. Technical report, SAE Technical Paper, 2006.

[108] Horacio ABF Oliveira, Eduardo F Nakamura, Antonio AF Loureiro, and Azzedine Boukerche. Error analysis of localization systems for sensor networks. In *Proceedings of GIS'05*, pages 71–78. ACM, 2005.

[109] Rebecca L Olson, Richard J Hanowski, Jeffrey S Hickman, and Joseph Bocanegra. Driver distraction in commercial vehicle operations. Technical report, United States. Federal Motor Carrier Safety Administration, 2009.

[110] Antonio Peréz, M Luisa Córdoba, A Garcia, Rafael Méndez, ML Munoz, José Luis Pedraza, and F Sanchez. A precise eye-gaze detection and tracking system. In *Proceedings of WSCG'03*, number 7, pages 105–108. UNION Agency, 2003.

[111] Jochen Pohl, Wolfgang Birk, and Lena Westervall. A driver-distraction-based lane-keeping assistance system. volume 221, pages 541–552. IMECHE, 2007.

[112] J Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann, 1993.

[113] Cristiano G Rezende, Azzedine Boukerche, Heitor S Ramos, and Antonio AF Loureiro. A reactive and scalable unicast solution for video streaming over vanets. *IEEE Transactions on Computers*, 64(3):614–626, 2015.

[114] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5(Jan):101–141, 2004.

[115] Irina Rish. An empirical study of the naive bayes classifier. In *Proceedings on IJCAI'01*, 2001.

[116] Rueda-Domingo. The influence of passengers on the risk of the driver causing a car collision in spain: Analysis of collisions from 1990 to 1999. *Accident Analysis and Prevention*, 36(3):481–489, 2004.

[117] Christos Sagonas, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *Proceedings of ICCVW'13*, pages 397–403. IEEE, 2013.

[118] Samer Samarah, Muhannad Al-Hajri, and Azzedine Boukerche. A predictive energy-efficient technique to support object-tracking sensor networks. *IEEE Transactions on Vehicular Technology*, 60(2):656–663, 2011.

[119] Mandalapu Saradadevi and Preeti Bajaj. Driver fatigue detection using mouth and yawning analysis. *International Journal of Computer Science and Network Security*, (6):183–188, 2008.

[120] Jason M Saragih, Simon Lucey, and Jeffrey F Cohn. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, 91(2):200–215, 2011.

[121] S. V. Sheela and P. A. Vijaya. Mapping functions in gaze tracking. *International Journal of Computer Applications*, 26(3):36–42, 2011.

[122] Neville A Stanton and Paul M Salmon. Human error taxonomies applied to driving: A generic driver error taxonomy and its implications for intelligent transport systems. *Safety Science*, 47(2):227–237, 2009.

[123] Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8(1):25, 2007.

[124] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. Learning-by-synthesis for appearance-based 3d gaze estimation. In *Proceedings of CVPR'14*, pages 1821–1828. IEEE, 2014.

[125] Chao Sun, Jian Hua Li, Yang Song, and Lai Jin. Real-time driver fatigue detection based on eye state recognition. In *Proceedings of Applied Mechanics and Materials'14*, volume 457, pages 944–952. Trans Tech Publications, 2014.

[126] Richard Szeliski. *Computer vision: algorithms and applications*. Springer, 2010.

[127] Simon Baker Takahiro Ishikawa and Takeo Kanade Iain Matthews. Passive driver gaze tracking with active appearance models. In *Proceedings of ITS Japan'04*, 2004.

[128] Kar-Han Tan, David J Kriegman, and Narendra Ahuja. Appearance-based eye gaze estimation. In *Proceedings of WACV'02*, pages 191–195. IEEE, 2002.

[129] Ashish Tawari, Kuo Hao Chen, and Mohan M Trivedi. Where is the driver looking: Analysis of head, eye and iris for robust gaze zone estimation. In *Proceedings of ITSC'14*, pages 988–994. IEEE, 2014.

[130] Ashish Tawari, Sujitha Martin, and Mohan Manubhai Trivedi. Continuous head movement estimator for driver assistance: Issues, algorithms, and on-road evaluations. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):818–830, 2014.

[131] Michael E Tipping and Christopher M Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.

[132] Akira Tomono, Muneo Iida, and Yukio Kobayashi. A tv camera system which extracts feature points for non-contact eye movement detection. In *Proceedings of SPIE'90*, volume 1194, pages 2–12. International Society for Optics and Photonics, 1990.

[133] Philip A Tresadern, Mircea C Ionita, and Timothy F Cootes. Real-time facial feature tracking on a mobile device. *International Journal of Computer Vision*, 96(3):280–289, 2012.

[134] Michel Valstar, Brais Martinez, Xavier Binefa, and Maja Pantic. Facial point detection using boosted regression and graph models. In *Proceedings of CVPR'10*, pages 2729–2736. IEEE, 2010.

[135] Sara Van de Geer. Least squares estimation with complexity penalties. *Mathematical Methods of Statistics*, 10(3):355–374, 2001.

[136] Francisco Vicente, Zehua Huang, Xuehan Xiong, Fernando De la Torre, Wende Zhang, and Dan Levi. Driver gaze tracking and eyes off the road detection system. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):2014–2027, 2015.

[137] Leandro Villas, Azzedine Boukerche, Regina Borges De Araujo, and Antonio AF Loureiro. Highly dynamic routing protocol for data aggregation in sensor networks. In *Proceedings of ISCC'10*, pages 496–502. IEEE, 2010.

[138] Leandro A Villas, Azzedine Boukerche, Daniel L Guidoni, Horacio ABF De Oliveira, Regina Borges De Araujo, and Antonio AF Loureiro. An energy-aware spatio-temporal correlation mechanism to perform efficient data collection in wireless sensor networks. *Computer Communications*, 36(9):1054–1066, 2013.

[139] Leandro A Villas, Daniel L Guidoni, Regina B Araújo, Azzedine Boukerche, and Antonio AF Loureiro. A scalable and dynamic data aggregation aware routing protocol for wireless sensor networks. In *Proceedings of MSWIM'10*, pages 110–117. ACM, 2010.

[140] Leandro Aparecido Villas, Azzedine Boukerche, Guilherme Maia, Richard Werner Pazzi, and Antonio AF Loureiro. Drive: An efficient and robust data dissemination protocol for highway and urban vehicular ad hoc networks. *Computer Networks*, 75:381–394, 2014.

[141] Leandro Aparecido Villas, Azzedine Boukerche, Heitor Soares Ramos, Horacio AB Fernandes de Oliveira, Regina Borges de Araujo, and Antonio Alfredo Ferreira Loureiro. Drina: A lightweight and reliable routing approach for in-network aggregation in wireless sensor networks. *IEEE Transactions on Computers*, 62(4):676–689, 2013.

[142] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of CVPR'01*, volume 1, pages I–I. IEEE, 2001.

[143] Nannan Wang, Xinbo Gao, Dacheng Tao, and Xuelong Li. Facial feature point detection: A comprehensive survey. *arXiv:1410.1037*, 2014.

[144] Yang Wang, Simon Lucey, and Jeffrey F Cohn. Enforcing convexity for improved alignment with constrained local models. In *Proceedings of CVPR'08*, pages 1–8. IEEE, 2008.

[145] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[146] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. Learning an appearance-based gaze estimator from one million synthesised images. In *Proceedings of ETRA'16*, pages 131–138. ACM, 2016.

[147] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, S Yu Philip, et al. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2008.

[148] Alan L Yuille, Peter W Hallinan, and David S Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.

[149] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Appearance-based gaze estimation in the wild. In *Proceedings of CVPR'15*, pages 4511–4520, 2015.

[150] Zhenxia Zhang, Richard W Pazzi, and Azzedine Boukerche. A mobility management scheme for wireless mesh networks based on a hybrid routing protocol. *Computer Networks*, 54(4):558–572, 2010.

[151] Yefeng Zheng, Xiang Sean Zhou, Bogdan Georgescu, Shaohua Kevin Zhou, and Dorin Comaniciu. Example based non-rigid shape detection. In *Proceedings of ECCV'06*, pages 423–436. Springer, 2006.

[152] Danjie Zhu, Steven T Moore, and Theodore Raphan. Robust pupil center detection using a curvature algorithm. *Computer Methods and Programs in Biomedicine*, 59(3):145–157, 1999.

[153] Zhiwei Zhu and Qiang Ji. 3d face pose tracking from an uncalibrated monocular camera. In *Proceedings of ICPR'04*, volume 4, pages 400–403. IEEE, 2004.

[154] Zhiwei Zhu and Qiang Ji. Eye and gaze tracking for interactive graphic display. *Machine Vision and Applications*, 15(3):139–148, 2004.

[155] Zhiwei Zhu, Qiang Ji, and Kristin P Bennett. Nonlinear eye gaze mapping function estimation via support vector regression. In *Proceedings of ICPR'06*, volume 1, pages 1132–1135. IEEE, 2006.

[156] Zhiwei Zhu, Qiang Ji, Kikuo Fujimura, and Kuangchih Lee. Combining kalman filtering and mean shift for real time eye tracking under active ir illumination. In *Proceedings of ICPR'02*, volume 4, pages 318–321. IEEE, 2002.