Received August 4, 2019, accepted August 18, 2019, date of publication August 21, 2019, date of current version September 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2936663

Real-Time Driver-Drowsiness Detection System Using Facial Features

WANGHUA DENG[1] **AND RUOXUE WU**[1,2]
[1]Beijing Engineering Research Center for IoT Software and Systems, Beijing University of Technology, Beijing 100124, China
[2]School of Software, Yunnan University, Kunming 650000, China

Corresponding author: Ruoxue Wu (rochelle.wu820@gmail.com)

ABSTRACT The face, an important part of the body, conveys a lot of information. When a driver is in a state of fatigue, the facial expressions, e.g., the frequency of blinking and yawning, are different from those in the normal state. In this paper, we propose a system called DriCare, which detects the drivers' fatigue status, such as yawning, blinking, and duration of eye closure, using video images, without equipping their bodies with devices. Owing to the shortcomings of previous algorithms, we introduce a new face-tracking algorithm to improve the tracking accuracy. Further, we designed a new detection method for facial regions based on 68 key points. Then we use these facial regions to evaluate the drivers' state. By combining the features of the eyes and mouth, DriCare can alert the driver using a fatigue warning. The experimental results showed that DriCare achieved around 92% accuracy.

INDEX TERMS convolutional neural network, fatigue detection, feature location, face tracking.

I. INTRODUCTION

In recent years, an increase in the demand for modern transportation necessitates a faster car-parc growth. At present, the automobile is an essential mode of transportation for people. In 2017, a total of 97 million vehicles were sold globally, which was 0.3% more than that in 2016 [1]. In 2018, the global total estimation of the number of vehicles being used was more than 1 billion [2]. Although the automobile has changed people's lifestyle and improved the convenience of conducting daily activities, it is also associated with numerous negative effects, such as traffic accidents. A report by the National Highway Traffic Safety Administration [3] showed that a total of 7,277,000 traffic accidents occurred in the United States in 2016, resulting in 37,461 deaths and 3,144,000 injuries. In these accidents, fatigue driving caused approximately $20\% - 30\%$ traffic accidents. Thus, fatigued driving is a significant and latent danger in traffic accidents. In recent years, the fatigue-driving-detection system has become a hot research topic. The detection methods are categorized as subjective and objective detection. In the subjective detection method, a driver must participate in the evaluation, which is associated with the driver's subjective perceptions through steps such as self-questioning, evaluation and filling in questionnaires. Then, these data are used to estimate the vehicles being driven by tired drivers, assisting the drivers to plan their schedules accordingly. However, drivers' feedback is not required in the objective detection method as it monitors the driver's physiological state and driving-behavior characteristics in real time [4]. The collected data are used to evaluate the driver's level of fatigue. Furthermore, objective detection is categorized into two: contact and non-contact. Compared with the contact method, non-contact is cheaper and more convenient because the system that not require Computer Vision technology or sophisticate camera allow the use of the device in more cars.

Owing to easy installation and low cost, the non-contact method has been widely used for fatigue-driving detection. For instance, Attention Technologies [5] and SmartEye [6] employ the movement of the driver's eyes and position of the driver's head to determine the level of their fatigue.

In this study, we propose a non-contact method called DriCare to detect the level of the driver's fatigue. Our method employs the use of only the vehicle-mounted camera, making it unnecessary for the driver to carry any on/in-body devices. Our design uses each frame image to analyze and detect the driver's state.

Technically, DriCare addresses three critical challenges. First, as drivers' heights are different, the positions of their faces in the video are different. Then, when the driver is

The associate editor coordinating the review of this article and approving it for publication was Gustavo Olague.

driving, his or her head may be moving; hence, tracking the trajectory of the head in time is important once the position of the head changes. To monitor and warn the driver in real-time, the use of the kernelized correlation filters (KCF) algorithm [7] is preferred based on our system's evaluation.

However, the KCF algorithm only uses a single Felzenszwalb histogram of oriented gradient features [8], which has poor face-tracking accuracy in a complex environment. Moreover, the KCF algorithm uses a manual method to mark the tracked target in the frame. In the case of KCF tracker cannot immediately retrieve the target, and it cannot track the human face once the target leaves the detection area and then returns.

Second, the driver's eyes and mouth play a vital role in tracking. Thus, identifying the key facial features of the driver is important for judging driving fatigue. A previous study [9] proposes a deep convolutional neural network for detecting key points. Though some traditional models [9]–[11] can detect the positions of several facial key points, they cannot determine the regions of the driver's eyes and mouth.

Third, defining the driver's level of drowsiness is crucial for our system. When people are tired, drowsiness is evident on their faces. According to Walter [12], the rate of the driver's eye closure is associated with the degree of drowsiness. Based on this principle, Grace *et al.* [13] proposed PERCLOS (percentage of eyelid closure over the pupil over time) and introduced Copilot to measure the level of the driver's fatigue. Constant yawning is a sign of drowsiness, which may provoke the driver to fall asleep. Li *et al.* [14], Abtahi *et al.* [15], and Fan *et al.* [16] used this feature to estimate the level of the driver's fatigue. However, practically, the driver may have different and complex facial expressions, which may distort the identification of these features.

Our core contributions are as follows:

- First, we propose a new face-tracking algorithm named Multiple Convolutional Neural Networks(CNN)-KCF (MC-KCF), which optimizes KCF algorithm. We combine CNN with the KCF algorithm [17] to improve the performance of the latter in a complex environment, such as low light. Furthermore, we introduce the multitask convolutional neural networks (MTCNN) [18] to compensate for the inability of the KCF algorithm to mark the target in the first frame and prevent losing the target.
- Second, we use CNN to assess the state of the eye. To improve the accuracy of CNN, DriCare measures the angle of an opening eye to determine if the eye is closed. To detect yawning, DriCare assesses the duration of the mouth opening. Besides, DriCare proposes three different criteria to evaluate the degree of the driver's drowsiness: the blinking frequency, duration of the eyes closing, and yawning. If the results surpass the threshold, DriCare will alert the driver of drowsiness.

The remainder of this paper is organized as follows. We review the related research in Section II and present the DriCare overview in Section III. Section IV presents the principle of human face tracking based on the MC-KCF algorithm. In Section V, we present the evaluation method for the driver's degree of drowsiness. We describe the DriCare implementation method and present the results of the experiment in Section VI. In Section VII, presents the conclusions of this study.

II. RELATED WORK

In this section, we categorize the related work into three parts, those related to the visual object tracking algorithm, the facial landmarks recognition algorithm and those to the methods of driver-drowsiness detection.

A. VISUAL OBJECT TRACKING

Visual object tracking is a crucial problem in computer vision. It has a wide range of applications in fields such as human-computer interaction, behavior recognition, robotics, and surveillance. Visual object tracking estimates the target position in each frame of the image sequence, given the initial state of the target in the previous frame. Lucas and Kanade [19] proposed that the tracking of the moving target can be realized using the pixel relationship between adjacent frames of the video sequence and displacement changes of the pixels. However, this algorithm can only detect the medium-sized target that shifts between two frames. With the recent advances of the correlation filter in computer vision [7], [20]–[22], Bolme [20] proposed the Minimum Output Sum of Squared Error (MOSSE) filter, which can produce stable correlation filters to track the object. Although the MOSSE's computational efficiency is high, its algorithm precision is low, and it can only process the gray information of a single channel.

Based on the correlation filter, Li and Zhu [22] utilized HoG, color-naming features and the scale adaptive scheme to boost object tracking. Danelljan *et al.* [23] used HOG and the discriminative correlation filter to track the object. SAMF and DSST solve the problem of deformation or change in scale when the tracking target is rotating. Further, they solve the problem of the tracker's inability to track object adaptively and the low operation speed. With the development of the deep-learning algorithm, some scholars combine deep learning and the correlation filter to track the mobile target [24]–[28]. Although these algorithms have better precision than the track algorithms based on the correlation filter, their training is time-consuming. Hence, these algorithms cannot track the object in real-time in a real environment. In this study, we propose a MC-KCF algorithm based on the correlation filter and deep learning. This algorithm uses CNN and MTCNN to offset the KCF's limitation and uses the KCF to track objects. Thus, the algorithm can track the driver's face in real-time using our system.

B. FACIAL LANDMARKS RECOGNITION

The purpose of facial key-points recognition is that getting the crucial information about locations of eyebrows, eyes, lips and nose in the face. With the development of deep learning, it is the first time for Sun *et al.* [9] to introduced DCNN based on CNN to detect human facial keypoints. This algorithm only recognizes 5 facial keypoints, albeit its speed is very fast. To get a higher precision for facial key points recognition, Zhou *et al.* [11] employed FACE++ which optimizes DCNN and it can recognize 68 facial keypoints, but this algorithm includes too much of a model and the operation of this algorithm is very complicated. Wu *et al.* [29] proposed Tweaked Convolutional Neural Networks (TCNN) which is based on Gaussian Mixture Model (GMM) to improve different layers of CNN. However, the robustness of TCNN depends on data excessively. Kowalski *et al.* [30] introduced Deep Alignment Network (DAN) to recognize the facial keypoints, which has better performance than other algorithms. Unfortunately, DAN needs vast models and calculation based on complicated functions. So in order to meet the requirement about real time performance, DriCare uses Dlib [31] to recognize facial keypoints.

C. DRIVER DROWSINESS DETECTION

Driver drowsiness detection can be divided into two types: contact approaches [32]–[34] and non-contact approaches [5], [6], [35], [36]. In contact approaches, drivers wear or touch some devices to get physiological parameters for detecting the level of their fatigue. Warwick *et al.* [32] implemented the BioHarness 3 on the driver's body to collect the data and measure the drowsiness. Li *et al.* [33] used a smartwatch to detect driver drowsiness based on electroencephalographic (EEG) signal. Jung *et al.* [34] reformed the steering wheel and set an embedded sensor to monitor the electrocardiogram (ECG) signal of the driver. However, due to the price of contact approaches and installation, there are some limitations which cannot be implemented ubiquitously. The other method employs a tag-free approaches to detect the driver drowsiness, where the measured object does not need to contact the driver. For example, Omidyeganeh *et al.* [35] used the driver's facial appearance captured by the camera to detect the driver drowsiness, but this method is not real-time. Zhang and Hua [37] used fatigue facial expression reorganization based on Local Binary Pattern (LBP) features and Support Vector Machines (SVM) to estimate the driver fatigue, but the complexity of this algorithm is bigger than our algorithm. Moreover, Picot *et al.* [38] proposed a method that uses electrooculogram (EOG) signal and blinking feature for drowsiness detection. Akrout and Mahdi [39] and Oyini Mbouna *et al.* [40] used a fusion system for drowsiness detection based on eye state and head position. Different from these methods, we employ simple formulae and evaluations, which make the results easier to measure.

III. DRICARE OVERVIEW

The proposed system, DriCare, is built using a commercial camera automobile device, a cloud server that processes video data, and a commercial cellphone that stores the result. Figure 1 shows the structure of the DriCare system. While driving, the automobile's camera captures the driver's portrait

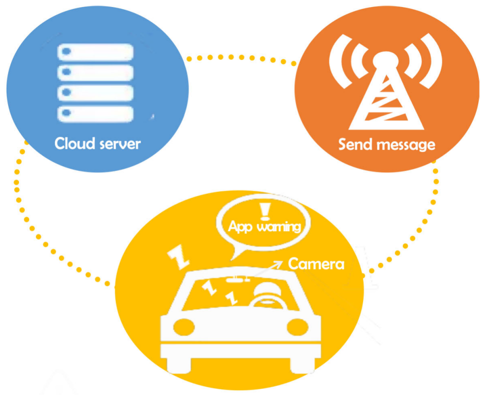


FIGURE 1. The architecture of DriCare.

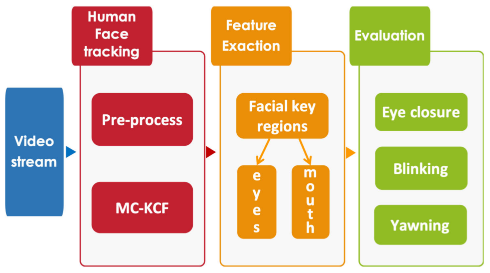


FIGURE 2. System workflow.

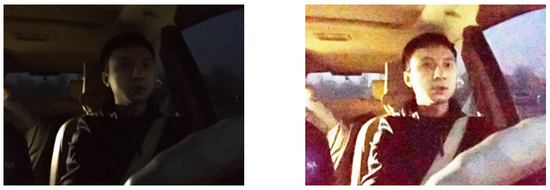
and uploads the video stream to the cloud server in real-time. Then, the cloud server analyzes the video and detects the driver's degree of drowsiness. In this stage, three main parts are analyzed: the driver's face tracking, facial key-region recognition, and driver's fatigue state. To meet the real-time performance of the system, we use the MC-KCF algorithm to track the driver's face and recognize the facial key regions based on key-point detection. Then, the cloud server estimates the driver's state when the states of the eyes and mouth change. The workflow is shown in Figure 2. Finally, the cloud server transmits the result to the driver's cellphone and other apps, through which a warning tone is transmitted if the driver is observed to be drowsy.

IV. DRIVER FACE TRACKING BY MC-KCF

In this section, we illustrate the principle of driver face tracking using DriCare. Owing to the complexity of the real environment, each frame of the video data requires preprocessing to meet the tracking requirements.

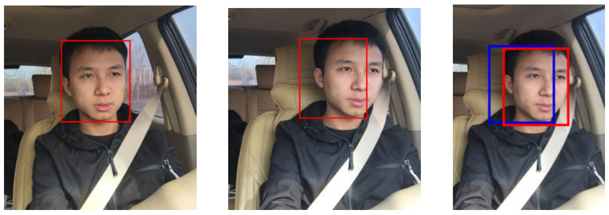
A. PRE-PROCESS

During the detection process, the quality of images is affected and features of the human face become unclear if the illumination intensity within the cab is changed during driving. This usually occurs in case of overcast skylight, rain, and at night. For detection accuracy, we use the illumination enhancement method to preprocess images before tracking the driver's face. Furthermore, we use the histogram equalization (HE) algorithm [41] to improve the brightness of the image frame.



(a) Original image. (b) The image after illumination enhancement.

FIGURE 3. The result of histogram equalization.



(a) The right result of tracking. (b) The tracking window drifting. (c) The tracking result of MC-KCF algorithm

FIGURE 4. The performance of original KCF and MC-KCF algorithm.

To determine whether light enhancement is required for the image frame, DriCare evaluates the brightness of the image. Therefore, we convert the RGB image into a YCbCr image because in the YCbCr color space, Y represents the luminance value. We use Eq. (1) for the mean value of Y M around the driver's face in the image as follows:

$$M = \frac{\sum_i^n L}{n-i} \tag{1}$$

where L denotes the luminance value of each pixel in the YCbCr space, and n and i represent the first and last serial numbers of the driver's facial pixels in the image, respectively. $n-i$ is the total number of driver's facial pixels. If M is lower than the threshold, the image enhances the illumination using the HE algorithm. Otherwise, the image is retained. After counting large samples, we set the threshold to 60. Figure 3 shows the result of the illumination enhancement.

B. THE PRINCIPLE OF MC-KCF

As previously discussed, the KCF algorithm is based on the FHOG feature [7]. Therefore, in a complex environment and during long-term operation, the tracking window will drift, as shown in Figure 4(b). We propose the MC-KCF algorithm instead of the original KCF algorithm to track a human face. In the MC-KCF algorithm, a new feature comprises the FHOG feature and is based on the KCF algorithm and CNN feature. We will explain the principle of FHOG and CNN feature execution and how these features are integrated.

1) FHOG FEATURE EXACTION

In our algorithm, the FHOG feature [8] is a key factor for human-face tracking. To extract the FHOG feature and for easy calculation, the image is grayed before commencing. Then, we calculate the gradient G and gradient orientation at each pixel α in the image as shown in Eq. (2):

$$\begin{cases} G_x(x,y) = H(x+1,y) - H(x-1,y) \\ G_y(x,y) = H(x,y+1) - H(x,y-1) \\ G = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \\ \alpha = arctan(\dfrac{G_x(x,y)}{G_y(x,y)}) \quad \alpha \in (0°, 360°) \end{cases} \tag{2}$$

where $H(x,y)$, $G_x(x,y)$, and $G_y(x,y)$ represent the pixel, horizontal gradient, and vertical gradient values at (x,y), respectively.

Then, we segment the image into $n \times n$ cells. According to [8], the gradient orientation is categorized into either 9 bins of contrast-sensitive orientations or 18 bins of contrast-insensitive orientations. If any pixel in a cell belongs to the corresponding orientation, the value of the orientation bin increases 1. Finally, each cell has 9-dimensional and 18-dimensional histograms.

The gradient of each cell is related to the internal pixels and the 4 cells around it. After calculating the gradient histogram, we use Eq. (3) and (4) for normalization and truncation.

$$\begin{aligned} N_{a,b}(i,j) = (C(i,j)^2 + C(i+a,j)^2 \\ + C(i+a,j+b)^2 + C(i,j+b)^2)^{\frac{1}{2}} \end{aligned} \tag{3}$$

$$\mathbf{H(i,j)} = \begin{pmatrix} T_\alpha(C(i,j)/N_{-1,-1}(i,j)) \\ T_\alpha(C(i,j)/N_{+1,-1}(i,j)) \\ T_\alpha(C(i,j)/N_{+1,+1}(i,j)) \\ T_\alpha(C(i,j)/N_{-1,+1}(i,j)) \end{pmatrix} \tag{4}$$

In Eq. (3), $C(i,j)$ denotes the 9- or 18-dimensional eigenvector of the cell at (i,j), $N_{a,b}(i,j)$ represents the normalization factor and a, b represent that the number of different normalization factors, $a, b \in \{-1, 1\}$. In Eq. (4), $\mathbf{H(i,j)}$ is a feature vector, and $T_\alpha(x)$ denotes the truncated function. If the value in x is bigger than α, the value is assigned to α.

After normalization and truncation, the 9-dimensional feature vector becomes a 36-dimensional feature vector. The 18-dimensional eigenvector becomes a 72-dimensional feature vector; in total, there are 108-dimensional feature vectors. Then, we arrange this eigenvector with reference to the matrix of 4×27. Finally, we obtain 31-dimensional HOG features, named the FHOG feature, using matrix addition.

2) CNN FEATURE EXACTED BY SQUEEZENET 1.1

SqueezeNet is a small CNN architecture [17] with very fast operation. Figure 5 shows the architecture of SqueezeNet 1.1, which includes a standalone convolution layer (conv1), 3 max-pooling layers, 8 fire modules (Fire2 − 9), a final convolution layer (conv10), and one global average pool layer.

SqueezeNet uses the fire module instead of the traditional convolution layer to reduce the network parameters and improve accuracy. The fire module comprises a squeeze convolution layer of 1×1 filters, feeding into an expand layer with a mix of 1×1 and 3×3 convolution filters, similar to that shown in Figure 6. Three tunable dimensions

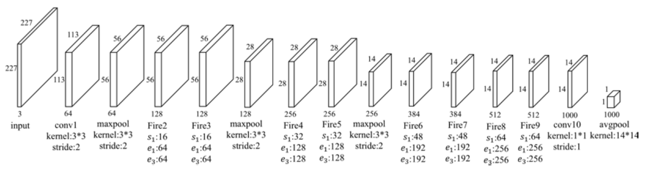


FIGURE 5. The architecture of SqueezeNet 1.1.

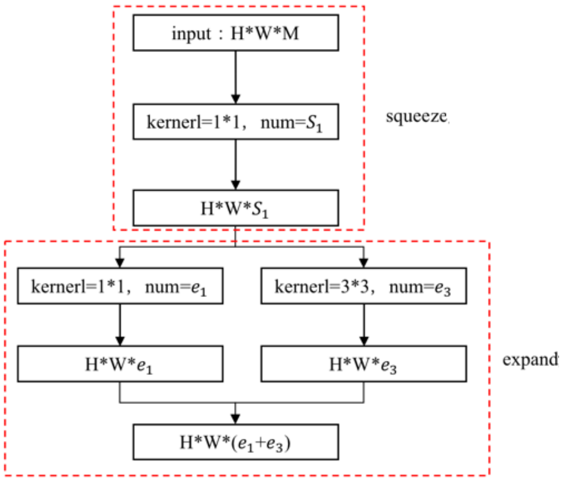


FIGURE 6. The architecture of Fire module.

are S_1, e_1, and e_3. A feature map sized $H \times W \times M$ becomes $H \times M \times S_1$ by squeezing layer processing [17]. Processing by expanding layer [17], we can obtain a feature map sized $H \times W \times (e_1 + e_3)$.

3) MULTI-FEATURE FUSION

To avoid large and redundant CNN features, DriCare uses the CNN feature obtained from the 1×1 convolution filter in the expand layer of Fires 5 and 9 of the SqueezeNet model. After conducting the feature extraction of a $D_1 \times G_1$ original image using the MC-KCF algorithm, we obtain a FHOG feature sized $D_2 \times G_2$ and two CNN features sized $D_3 \times G_3$ and $D_4 \times G_4$. Obviously, the sizes of the three features differ. Therefore, we adjust them so that they have the same size. Thus, the adjustment equation is written as follows:

$$\begin{cases} D = D_a \times \theta \\ G = G_a \times \varphi \end{cases} \tag{5}$$

where D and G denote the standard length and width, respectively. D_a and G_a represent the original length and width of the three features, respectively. θ and φ are the scaling factors.

Similar to the structure of the KCF algorithm, in the MC-KCF algorithm, we use each feature to train their classifiers separately using the kernel ridge regression, which is written as follows [7]:

$$\vec{\alpha} = \hat{\vec{\alpha}} = \vec{y}(\mathbf{K} + \lambda \mathbf{I})^{-1} = \frac{\hat{\vec{y}}}{\hat{k^{xx}} + \lambda} \tag{6}$$

where $\mathbf{K}$ is the kernel matrix, $\mathbf{I}$ denotes an identity matrix, $\vec{\alpha}$ represents the vector of coefficients α_i, λ is a hyperparameter,

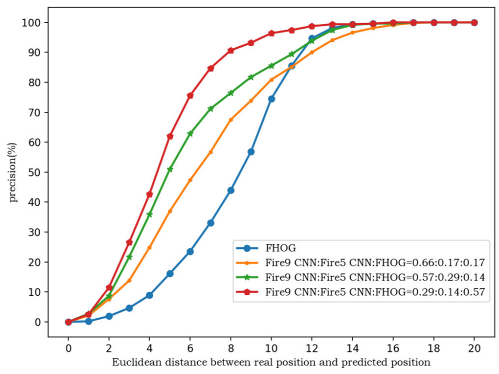


FIGURE 7. The face tracking result in different weights of features.

and $\vec{y}$ is the vector of regression targets. Moreover, a hat $\hat{}$ denotes the DFT of a vector, and k^{xx} is the first row of the kernel matrix $\mathbf{K}$.

After the training, we use each classifier to evaluate the regression function $f(z)$ for every image sample z. The biggest value of $f(z)$ is the forecasted position of the target for each feature. The equation is as follows:

$$f(z) = \mathcal{F}^{-1}(\hat{k^{xz}} \odot \hat{\alpha}) \tag{7}$$

where $\mathcal{F}^{-1}$ denotes the inverse DFT. Thus, we obtain three tracking results. To obtain the final result of the MC-KCF algorithm, we set different weights for the result of three features: δ_1, δ_2, and δ_3. We calculate the entire response value of the MC-KCF algorithm F using the weights and the prediction positions based on FHOG and CNN features. The formula is as follows:

$$F = \delta_1 \times f(z_{fhog}) + \delta_2 \times f(z_{fire5}) + \delta_3 \times f(z_{fire9}) \tag{8}$$

In Eq. (8), the codomain of δ is [0, 1]. When one of $\delta = 0$, the response value of the corresponding feature is not the final result; otherwise, when $\delta = 1$, the response value of the corresponding feature is the entire response value. From the response value, we obtain the position of the driver's face. The different weights of the three features can influence the tracking accuracy. Thus, we calculate 1000 weight ratios of the three features. Figure (7) shows a representative ratio. When the ratio of $\delta_1 : \delta_2 : \delta_3$ is $0.57 : 0.14 : 0.29$, respectively, the performance is optimal. In our system, the ratio is $0.57 : 0.14 : 0.29$.

Therefore, the total dimension of the CNN features is 384, bigger than the dimension of the original FHOG feature, which is a 31-dimensional HOG feature. Since the modified object is small in some frames, we update the model of every N frame to increase the computing speed of the model and improve the real-time performance of the system. We set N value as 3. The whole process is shown in Figure (8).

4) CALIBRATION OF MC-KCF

As discussed above, the original KCF algorithm is unable to automatically obtain the tracking target of the first video frame. Besides the original KCF algorithm, in which the

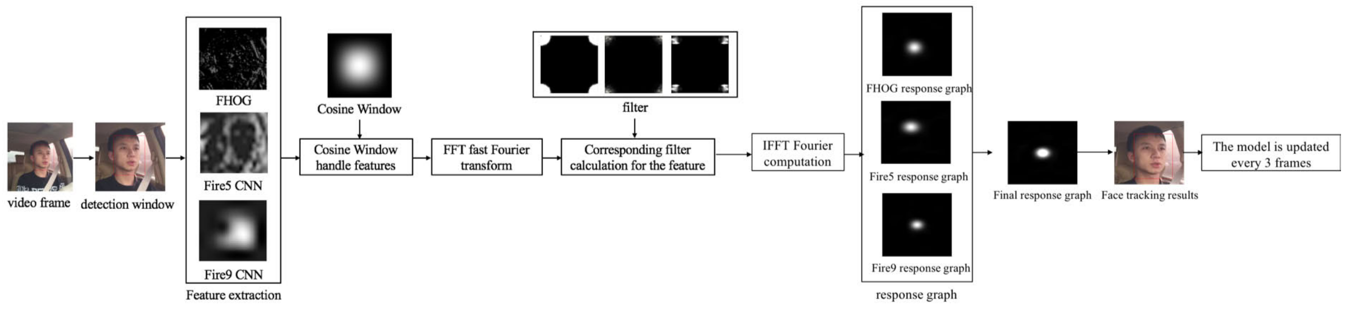


FIGURE 8. **The process of MC-KCF algorithm.**

object goes out of the camera's sight, we align the MC-KCF algorithm in case the algorithm is unable to track the driver's face. In Section III, the MTCNN algorithm uses the bounding box to precisely determine the human face. Thus, we use MTCNN to periodically calibrate the MC-KCF algorithm.

After preprocessing the video frame, the cloud severs will judge whether the current image is the first frame. If it is, the cloud server will use the MTCNN algorithm to locate the human face in the image; otherwise, the cloud sever will continue to judge whether the span of tracking time surpasses $10s$. If the answer is yes, the cloud sever will use the MTCNN algorithm to relocate the human face and reset the tracking time. If the system evaluates that the current image is not the first frame and the duration of the tracking time is less than $10s$, DriCare will use the MC-KCF algorithm to track the driver's face using the result to update the scope of the search for the driver's face for the next frame. We summarize the MC-KCF calibration process in Algorithm 1.

```
Algorithm 1 Calibration of MC-KCF Algorithm
Input: frame of the video $fram$, the span of the tracking
    time $t$, the count number of frames $cnt$, the number of all
    frames$frams$
Output: the result of human face tracking $res\_img$
    Load $fram$, set $t$ is 0 and $cnt$ is 0
    while $t \leqslant 10s$ and $cnt \leqslant frams$ do
        if the current frame is the first frame then
            Use MTCNN algorithm to detect a human face
        else if MC-KCF algorithm cannot detect a human face
        then
            Use MTCNN algorithm to detect a human face
        else
            Use MC-KCF algorithm to detect a human face
            $t++$
            $cnt++$
            Output the result $res\_img$
            Update the scope of the detection
            Read the next frame
        end if
    end while
    if $t == 10s$ then
        Use MTCNN algorithm to align the MC-KCF algorithm
        (use MTCNN algorithm to detect human face) in the
        current frame
        Output the result $res\_img$
        Update the scope of the detection
        $t = 0$
        $cnt++$
    end if
    Read the next frame
```

V. EVALUATION OF THE DRIVER'S FATIGUE STATE

In this section, we discuss the method of analyzing the driver's face via the DriCare system in case of drowsiness. Further, we discuss methods to locate the regions of the eyes and mouth on the driver's face. A change state of the eyes and mouth is a crucial indicator of drowsiness. Additionally, we discuss a new algorithm to detect the driver's fatigue.

A. DETERMINATION OF EYES AND MOUTH REGIONS

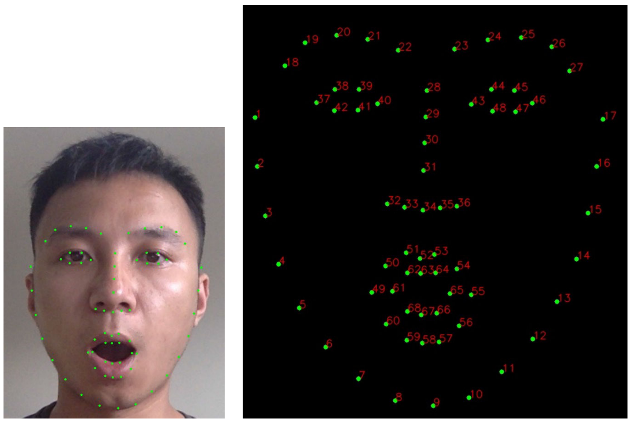
In Section 4, we recognize and track the driver's face in each video frame. Then, we use Dlib [31] to locate 68 facial key points on the driver's face. The result is shown in Figure (9). After obtaining the key points, we set the coordinate of each key point as (x_i, y_i) and use the key points to locate the regions of the eyes and mouth on the driver's face.

1) THE REGION OF THE EYES

First, we offer the solution for locating the eyes' regions. From Figure 9, one eye has six key points. However, these points are near the eyeball. By using these points to detect the region of an eye, the region will not include the upper and lower eyelids from the analysis, thereby influencing the result of the subsequent evaluation. Therefore, we use the key points of the eyebrow and nose to define the scope of the eye and eye socket. The equation is as follows:

$$\begin{cases} lex = \dfrac{x_i + x_j}{2} \\ ley = y_m + \dfrac{y_n - y_m}{4} \end{cases} \tag{9}$$

In Eq. (9), x_i and x_j represent the X coordinate of the i_{th} and j_{th} key points, respectively. y_n and y_m represent the Y coordinate of the n_{th} and m_{th} key points, respectively. lex and ley denote



(a) The location of the facial key points. (b) The number of the facial key points.

FIGURE 9. The numbering of facial key points.

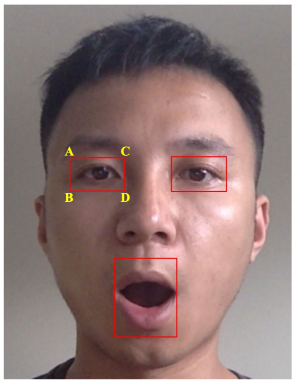


FIGURE 10. The ratio of mouth's width and height in different states.

TABLE 1. The calculation parameters of the position of the two eyes.

Name	Position	i	j	m	n
Left Eye	Upper left vertex of the region	18	37	$\min(y_{18}, y_{22})$	$\min(y_{38}, y_{39})$
	Lower right vertex of the region	22	40	30	$\min(y_{41}, y_{42})$
Right Eye	Upper left vertex of the region	23	43	$\min(y_{23}, y_{27})$	$\min(y_{44}, y_{45})$
	Lower right vertex of the region	27	46	30	$\min(y_{47}, y_{48})$

the vertices' coordinates of the rectangular region of the eye. In our system, according to Fig. 9(b), when i is Point 18, j is Point 37. m is the point number, with the minimum value of Y coordinate between Points 18 and 22. n represents the point number with the minimum value of Y coordinate between Points 38 and 39. As shown in Figure (10), we obtain the A vertex of the left eye.

After we obtain the coordinate of the upper left A and lower right vertices of the region D, we determine the eye socket region on the driver's face based on rectangular symmetry. Table 1 shows the calculation parameters of the position of the two eyes.

B. EVALUATION OF THE DRIVER'S FATIGUE STATE

In this section, we discuss the principle of evaluating the driver's fatigue state. As shown in Figure 2, DriCare uses two factors to evaluate the state of the driver's fatigue: the states

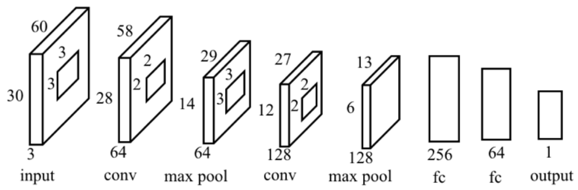


FIGURE 11. The architecture of CNN.

of the eyes and mouth. Unlike other methods, we propose a new assessment for the eyes state to achieve higher accuracy. With CNN, we use the angle of the eyes to evaluate the eye state. Moreover, we use the state of the single eye near the camera to assess the state of the whole eye. Besides, DriCare also measures the state of the mouth to judge if the driver is yawning. After these assessments, DriCare merges these results and evaluates the driver's degree of drowsiness.

1) EYE STATUS RECOGNITION

a: RECOGNITION BASED ON CNN

We build a CNN to recognize the eight layers of the eye state. Figure 11 shows the CNN architecture.

We use two convolutional layers and maximum pooling layers to extract the feature based on the eye region. The features are integrated by two full connection layers. Finally, the results of the output are used to judge if the eye is open. The number of neurons in the output layer is 1, and the activation value is obtained using the sigmoid function as the values are equal to or greater than 0, as shown in Eq. (10).

$$S(x) = \frac{1}{1 + e^{-x}} \tag{10}$$

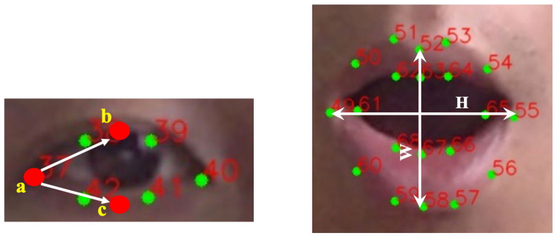
In Eq. 10, the range of the result is in $[0, 1]$. During training, the value of an open eye is 1, representing positive samples, and the value of a closed eye is 0, representing negative samples. A predicted value of greater than 0.5 in the sigmoid activation function output represents the result of open eyes; otherwise, it represents closed eyes.

b: RECOGNITION BASED ON ANGLE

Owing to the CNN drawbacks (the accuracy of the eye closure recognition by CNN is poor), we use the angle of the eye to compensate for the CNN's limitations regarding eye closure recognition. After the CNN validates that the driver's eye is open, we use the angle of the eye to validate the result. A blink is the process of the eye closing and opening. As discussed in the previous section, we identify the eye region using the video frame. As revealed in Fig. 12(a), we use the key points in the eye region to assess the angle of the eye. The equation is as follows:

$$\begin{cases} d_{ij} = \sqrt{(y_j - y_i)^2 + (x_j - x_i)^2} \\ A = (arccos\dfrac{d_{ab}^2 + d_{ac}^2 - d_{bc}^2}{2 \times d_{ab} \times d_{ac}})/\pi \times 180^\circ \end{cases} \tag{11}$$

In Eq. (11), d_ij is the distance between Points i and j. (x_i, y_i) and (x_j, y_j) represent the coordinates of Points i and j in the



(a) The angle of eye opening. (b) The width and height of the mouth.

FIGURE 12. The state recognition for eye and mouth.

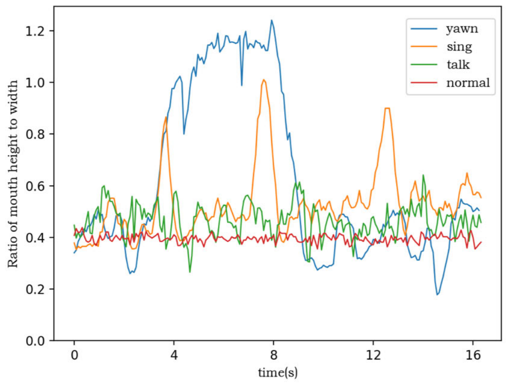


FIGURE 13. The ratio of mouth's width and height in different states.

frame, respectively. In our system, (x_i, y_i) and (x_j, y_j) are the two average coordinate values of two points in the eyelids. A is the angle of the eye. When we obtain the result, if the result is bigger than the threshold, DriCare will consider if the state of the eye is opening, and vice versa. We analyze the large number of samples. In the eye-closed state, the angle of the eye is lower than 20°. Therefore, we set the threshold at 20°.

We assess the driver's degree of fatigue from three perspectives based on the angle of the eye: (1) the proportion of the number of closed-eye frames to the total number of frames in 1 minute, (2) continuous time of eye closure, and (3) frequency of blinking. According to [42], [43], when the driver is awake, the proportion of closed-eye frames is less than 30%. Moreover, the driver's closure time for a single eye is shorter when he or she is awake, so when the driver's single eye closure time exceeds $2s$, the driver is considered fatigued. Besides, when people are awake, they blink an average of 10 to 15 times per minute. However, when people are mildly tired, the number of blinks increase; in case of severe fatigue, the number of blinks will be lower because the eyes are closed most of the time. To detect fatigue based on the frequency of blinking, it is necessary to count the blinking frequency of the eyes within 1 minute. If the blinking frequency is greater than 25 times/min or lower than 5 times/min, fatigue is indicated.

2) MOUTH STATUS RECOGNITION

For the detection of fatigued driving, the features of the mouth are important because when a driver is drowsy, continuous yawns will occur. Therefore, DriCare uses these features to measure the accuracy of evaluation. In Section 4, we obtain some key points in the mouth to calculate the ratio of the mouth's width and height. The equation is rewritten as follows:

$$\begin{cases} H = \sqrt{(y_r - y_e)^2 + (x_r - x_e)^2} \\ W = \sqrt{(y_u - y_v)^+ (x_u - x_v)^2} \\ f = \dfrac{H}{W} \end{cases} \tag{12}$$

In Eq. (12), f is the ratio of the mouth's width and height in one image frame. H represents the height of the mouth, and W denotes the width of the mouth. (x_r, y_r) is the coordinate of the vermilion tubercle, and (x_e, y_e) is the lowest point in the lower low lip. (x_u, y_u) and (x_v, y_v) are two coordinates of the angulus oris. If f is larger than the threshold, DriCare will consider that the driver is opening the mouth, and vice versa. According to the measurement for a large number of samples, in our paper, we set the threshold to 0.6.

In practice, the opening of the mouth may resemble other situations, such as singing, speaking, and laughing. These phenomena present the same results. To reduce errors, we draw the diagram of curves using the width-height ratio of the mouth obtained in each frame as shown in Figure (13). From this illustration, when the driver is yawning, the mouth will open continuously for a longer time, and the wave peaks are wider. Otherwise, when the driver is speaking, the mouth opens continuously for a shorter time, and the wave peaks are narrower. Hence, we use Eq. (13) to calculate the duration time ratio R of the opening mouth, which can discriminate actions such as yawning and speaking. The equation is written as follows:

$$R = \frac{n}{m} \times 100\% \tag{13}$$

where m represents the number of frames for some time. n is the number of the frame, and f exceeds the threshold. According to [44], [45], we know that the whole yawning process lasts for $7s$ in general, and the video frames, the f of which is higher than the threshold, are approximately $3 \sim 4s$. Therefore, we set R to 50%. When judging whether yawning occurs, we count the number of frames the ratio of mouth-height to -width of which is higher than the threshold of $7s$. To determine the proportion of these frames, the total number of detections should be greater than 50%. If this is established, we consider the driver to be yawning. We count the number of yawns in one minute and if the number of yawns is more than two times per minute, the driver is said to be drowsy.

However, according to [46], we can know that when driver is a condition of boring or tedious activities that increased yawning frequency. Hence, in order to eliminate the error, we set weights for these features separately, and then we account the total of weight, if the total of weight is higher than the threshold, DriCare will consider the driver is drowsy.

We summarize the entire detection process for DriCare in Algorithm 2.

Algorithm 2 Fatigue Detection Algorithm for DriCare

```
Input: frames of the video
Output: Evaluation of the degree of driver fatigue
  Load the frames of video
  Assess the states of the eye and mouth
  Calculate $r$ the ratio of the frame of eye closure in 1 minute
  and $t$ a duration time of eye closure.
  Calculate $b$ the frequency of blinking and $y$ the number of
  yawning in 1 minute.
  if $r > 30\%$ then
      $W_r = 1$
  end if
  if $t > 2s$ and is not yawning then
      $W_t = 1$
  end if
  if $b > 25$ or $b < 5$ then
      $W_b = 1$
  end if'
  if $y \geqslant 2$ then
      $W_y = 1$
  end if
  Calculate $T$ the total value of these weight. ($T = W_r + W_t + W_b + W_y$)
  if $T \geqslant 2$ then
      The driver is drowsy
  else
      The driver is awake
  end if
```

VI. EXPERIMENTS

A. DATASETS AND SETUP

Figure 14(a) shows a prototype of DriCare comprising a commercial Qihu 360 camera and an Intel Core i7 CPU Macbook Laptop at 2.5 GHZ and 16 GB memory to simulate the cloud server. Figure 14(b) shows the system interface.

We use 10 volunteers to collect the video data captured by the vehicle camera. Each volunteer simulates the drowsy and clear driving states. Each video is 1-h long. For the evaluation of drowsiness, we use the CelebA [47], YawDD [48] dataset and volunteer video data to assess the performance of DriCare.

We used Python 3.6, OpenCV 3.4, Tensorflow 1.8, and Tkinter 8.6 to build the software environment required for our experiments.

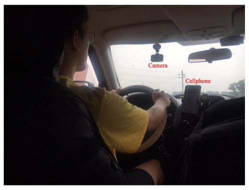
B. EXPERIMENTAL EVALUATION

We tested the DriCare performance and compared them with other methods in the same condition.

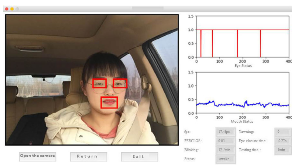
1) PERFORMANCE OF MC-KCF

The Euclidean distance between the predicted and real values of the target border is used to evaluate the performance of the tracking algorithms. We compare the MC-KCF algorithm with the other tracking algorithms using different scenarios. The main test scenarios are fast motion, target disappears in the field of vision, and target rotation. The average test results in each scenario are counted as the final experimental results, as shown in Figure15(a).

From Figure15(a), the MC-KCF algorithm demonstrates the best tracking accuracy. In a complex environment, the accuracy of MC-KCF is nearly 90%. Face tracking in the driving environment is simpler than in other environments because the driver's face moves less and the speed is average. Moreover, the face will be visible in the field of vision. Figure 15(b) shows the results of the MC-KCF test performance and other tracking algorithms, revealing that the MC-KCF algorithm produces the best performance, with the

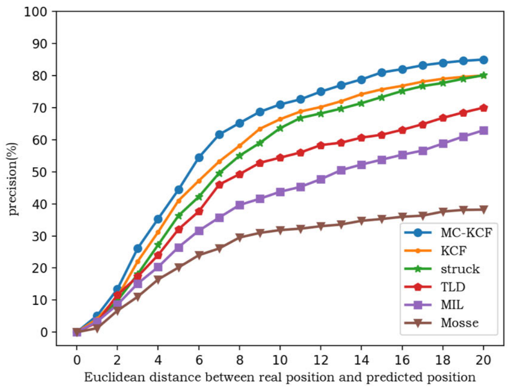


(a) The hardware of the implemented system.

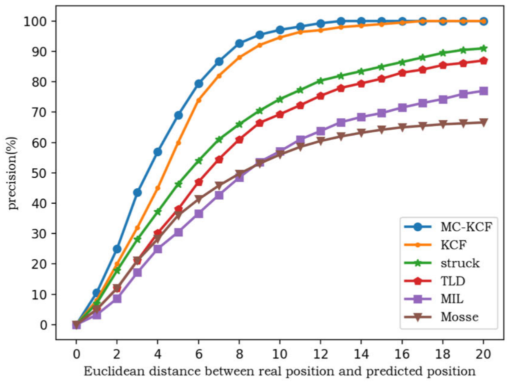


(b) The main interface of the system.

FIGURE 14. Experimental environment and interface of the system.



(a) In the complex environment.



(b) In the dricab environment.

FIGURE 15. The accuracy of face tracking in different environment.

TABLE 2. Comparison of other performances.

Method	The size of the video frame	The size of the human face	Accuracy	Frames Per Second
MTCNN	1280 × 720	240 × 300	93.2%	3fps
KCF			91%	188fps
DSST			85%	6fps
Struck			76%	8fps
KCF+CNN			93%	26fps
MC-KCF			95%	25fps

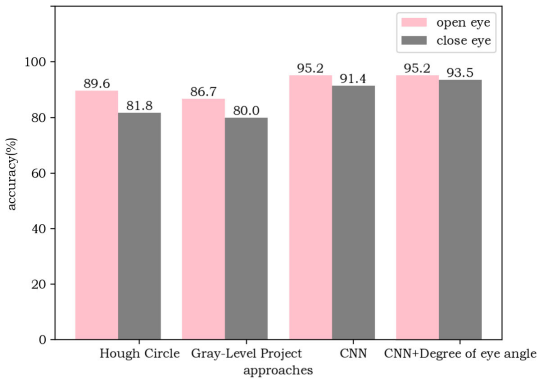


FIGURE 16. The comparison of eye state recognition method.

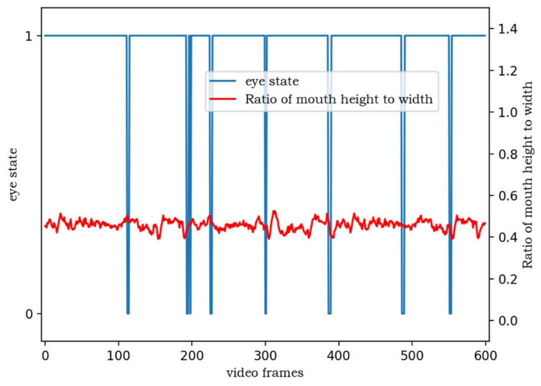
accuracy reaching approximately 95%., when the Euclidean distance within 20px.

As shown in Table 2, we further compare the different methods in terms of speed. Although the KCF algorithm offers the highest speed, its accuracy is the worse than those of MC-KCF and KCF + CNN. The MC-KCF algorithm has the best accuracy for face tracking; its accuracy is nearly 20% more than that of the Struck algorithm; however, its speed is slightly lower than that of KCF. The MC-KCF algorithm can process 25 video frames per second, which meets the requirement of our system. Thus, we consider that the MC-KCF algorithm performs better and offers the practical requirements for speed and accuracy.

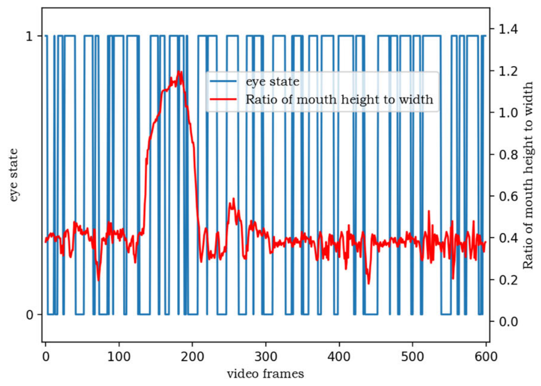
2) PERFORMANCE OF DETECTION METHODS

For testing the performance of our evaluation algorithm, we compare our method for evaluating the state of the eye with other methods. Figure 16 shows the result, indicating that the angle of eye opening is 95.2%, which is the highest among the evaluated methods. Additionally, the closed-eye recognition is the highest, at 93.5%. The success rate of identifying a closed eye is significantly improved by our method; it is 10% more than HoughCircle.

Figures 17(a) and (b) show that the recognition result of the states of the eye and mouth during drowsiness and otherwise. The horizontal axis represents the number of video frames; and the left vertical axis represents the opening of the eyes, wherein 1 represents the eye opening and 0 represents the eye closing. The right vertical axis represents the ratio of the height and width of the mouth. The experimental results show that when the driver is awake, the blinking frequency and eye-closing time are low. However, when the driver is



(a) In the awake state.



(b) In the drowsiness state.

FIGURE 17. The recognition result of eye and mouth in different state.

TABLE 3. The performance in different environments.

Number	The driving environment	Detection rate	Frames per second
1	Bright & Glasses	92%	18fps
2	Bright & No glasses	92.6%	18fps
3	Darkness & Glasses	91.1%	16fps
4	Darkness & No glasses	91.6%	16fps

tired, the blinking frequency and eye-closing time are high, and sometimes, the driver will be yawning.

3) PERFORMANCE OF DRICARE

To test the performance of our system, we measure the system in different experimental environments. The result is shown in Table 3.

From Table 3, our system provides the best accuracy when the cab is bright and the driver wears no glasses. If the driver wears glasses and the driving environment is slightly dim, the accuracy of fatigue driving is reduced. Regardless of the environmental condition, the average accuracy of our method is approximately 92%. However, the average processing speed is 18fps when the environment is bright. When the environment is dark, the speed is 16fps.

For now, there are not an image-based public driver drowsiness recognition dataset can be used to estimated the efficiency of our method. Therefore, we cannot compare the effectiveness of DriCare with other methods [37]–[39] due

TABLE 4. Comparison of results between DriCare and other state-of-the-art methods.

Research	Methodology	Accuracy, %
Zhang [37]	boost-LBP + SVM	85.9
Picot [38]	blinking feature + EOG	82.1
Akrout [39]	blinking + pose estimation	90.2
DriCare	MC-KCF + blinking + yawning	93.6

to different datasets. So we compare our method with other methods which are obtained from a video-based dataset. The result are shown in Table 4.

Table 4 reveals that compared with existing methods such as Zhang and Hua [37], Picot *et al.* [38] and Akrout and Mahdi [39], the average accuracy of DriCare is better than other methods, especially, the accuracy of DriCare is 11% more than Picot *et al.* [38]. Thus, DriCare can meet our requirements in terms of the estimation accuracy.

VII. CONCLUSION

We propose a novel system for evaluating the driver's level of fatigue based on face tracking and facial key point detection. We design a new algorithm and propose the MC-KCF algorithm to track the driver's face using CNN and MTCNN to improve the original KCF algorithm. We define the facial regions of detection based on facial key points. Moreover, we introduce a new evaluation method for drowsiness based on the states of the eyes and mouth. Therefore, DriCare is almost a real-time system as it has a high operation speed. From the experimental results, DriCare is applicable to different circumstances and can offer stable performance.

REFERENCES

[1] International Organization of Motor Vehicle Manufacturers. (2018). *Provisional Registrations or Sales of New Vehicles*. [Online]. Available: http://www.oica.net/wp-content/uploads/

[2] Wards Intelligence. (2018). *World Vehicles in Operation by Country, 2013–2017*. [Online]. Available: http://subscribers.wardsintelligence.com/data-browse-world

[3] National Highway Traffic Safety Administration. (2018). *Traffic Safety Facts 2016*. [Online]. Available: https://crashstats.nhtsa.dot.gov

[4] G. Borghini, L. Astolfi, G. Vecchiato, D. Mattia, and F. Babiloni, "Measuring neurophysiological signals in aircraft pilots and car drivers for the assessment of mental workload, fatigue and drowsiness," *Neurosci. Biobehav. Rev.*, vol. 44, pp. 58–75, Jul. 2014.

[5] Attention Technologies. (1999). *S.a.m.g-3-Steering Attention Monitor*. [Online]. Available: https://www.zzzzalert.com

[6] Smart Eye. (2018). *Smarteye*. [Online]. Available:https://smarteye.se/

[7] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[9] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3476–3483.

[10] S. Ren, X. Cao, Y. Wei, and J. Sun, "Face alignment at 3000 FPS via regressing local binary features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1685–1692.

[11] E. Zhou, H. Fan, Z. Cao, Y. Jiang, and Q. Yin, "Extensive facial landmark localization with coarse-to-fine convolutional network cascade," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2013, pp. 386–391.

[12] W. Walter, "Overview of research on driver drowsiness definition and driver drowsiness detection," in *Proc. Int. Tech. Conf. Enhanced Saf. Vehicles*, 1995, pp. 462–468.

[13] R. Grace, V. E. Byrne, D. M. Bierman, J.-M. Legrand, D. Gricourt, B. K. Davi, J. J. Staszewski, and B. Carnahan, "A drowsy driver detection system for heavy vehicles," in *Proc. 17th AIAA/IEEE/SAE. Digit. Avionics Syst. Conf. (DASC)*, vol. 2, Oct. 1998, pp. I36-1–I36-8.

[14] L. Li, Y. Chen, and Z. Li, "Yawning detection for monitoring driver fatigue based on two cameras," in *Proc. 12th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2009, pp. 1–6.

[15] S. Abtahi, B. Hariri, and S. Shirmohammadi, "Driver drowsiness monitoring based on yawning detection," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf.*, May 2011, pp. 1–4.

[16] X. Fan, B. Yin, and Y. Sun, "Yawning detection for monitoring driver fatigue," in *Proc. Int. Conf. Mach. Learn. Cybern.*, vol. 2, Aug. 2007, pp. 664–668.

[17] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size," in *Proc. ICLR*, 2017, pp. 1–13.

[18] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, Oct. 2016.

[19] D. B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. 7th Int. Joint Conf. Artif. Intell.*, San Francisco, CA, USA, vol. 2, 1981, pp. 674–679.

[20] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2544–2550.

[21] F. J. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. Eur. Conf. Comput. Vis.*, Berlin, Germany: Springer-Verlag, 2012, pp. 702–715.

[22] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *Proc. Eur. Conf. Comput. Vis.*, L. Agapito, M. M. Bronstein, and C. Rother, Eds. Cham, Switzerland: Springer, 2015, pp. 254–265.

[23] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1561–1575, Aug. 2017.

[24] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, New York, NY, USA: Curran Associates Inc., vol. 1, 2013, pp. 809–817.

[25] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4310–4318.

[26] C. Ma, J. Huang, X. Yang, and M. Yang, "Robust visual tracking via hierarchical convolutional features," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.

[27] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)*, Dec. 2015, pp. 621–629.

[28] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5000–5008.

[29] Y. Wu, T. Hassner, K. Kim, G. Medioni, and P. Natarajan, "Facial landmark detection with tweaked convolutional neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 3067–3074, Dec. 2018.

[30] M. Kowalski, J. Naruniec, and T. Trzcinski, "Deep alignment network: A convolutional neural network for robust face alignment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 2034–2043.

[31] D. E. King, "Dlib-ml: A machine learning toolkit," *J. Mach. Learn. Res.*, vol. 10, pp. 1755–1758, Jul. 2009.

[32] B. Warwick, N. Symons, X. Chen, and K. Xiong, "Detecting driver drowsiness using wireless wearables," in *Proc. 12th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Oct. 2015, pp. 585–588.

[33] G. Li, B.-L. Lee, and W.-Y. Chung, "Smartwatch-based wearable EEG system for driver drowsiness detection," *IEEE Sensors J.*, vol. 15, no. 12, pp. 7169–7180, Dec. 2015.

[34] S.-J. Jung, H.-S. Shin, and W.-Y. Chung, "Driver fatigue and drowsiness monitoring system with embedded electrocardiogram sensor on steering wheel," *IET Intell. Transp. Syst.*, vol. 8, no. 1, pp. 43–50, 2014.

[35] M. Omidyeganeh, A. Javadtalab, and S. Shirmohammadi, "Intelligent driver drowsiness detection through fusion of yawning and eye closure," in *Proc. IEEE Int. Conf. Virtual Environ., Hum.-Comput. Interfaces Meas. Syst.*, Sep. 2011, pp. 1–6.

[36] A. Dasgupta, D. Rahman, and A. Routray, "A smartphone-based drowsiness detection and warning system for automotive drivers," *IEEE Trans. Intell. Transp. Syst.*, to be published.

[37] Y. Zhang and C. Hua, "Driver fatigue recognition based on facial expression analysis using local binary patterns," *Optik*, vol. 126, no. 23, pp. 4501–4505, Dec. 2015.

[38] A. Picot, S. Charbonnier, A. Caplier, and N.-S. Vu, "Using retina modelling to characterize blinking: Comparison between EOG and video analysis," *Mach. Vis. Appl.*, vol. 23, no. 6, pp. 1195–1208, Nov. 2012.

[39] B. Akrout and W. Mahdi, "Spatio-temporal features for the automatic control of driver drowsiness state and lack of concentration," *Mach. Vis. Appl.*, vol. 26, no. 1, pp. 1–13, Jan. 2015.

[40] R. O. Mbouna, S. G. Kong, and M.-G. Chun, "Visual analysis of eye state and head pose for driver alertness monitoring," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1462–1469, Sep. 2013,

[41] S. M. Kamel and L. Guan, "Histogram equalization utilizing spatial correlation for image enhancement," *Proc. SPIE*, vol. 1199, pp. 712–723, Nov. 1989.

[42] S. Kaplan, M. A. Guvensan, A. G. Yavuz, and Y. Karalurt, "Driver behavior analysis for safe driving: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3017–3032, Dec. 2015.

[43] R. K. Satzoda and M. M. Trivedi, "Drive analysis using vehicle dynamics and vision-based lane semantics," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 9–18, Feb. 2015.

[44] J. Barbizet, "Yawning," *J. Neurol., Neurosurg. Psychiatry*, vol. 21, no. 3, pp. 203–209, Aug. 1958.

[45] J. R. Eguibar, C. A. Uribe, C. Cortes, A. Bautista, and A. C. Gallup, "Yawning reduces facial temperature in the high-yawning subline of sprague-dawley rats," *BMC Neurosci.*, vol. 18, no. 1, p. 3, Jan. 2017.

[46] A. Franzen, S. Mader, and F. Winter, "Contagious yawning, empathy, and their relation to prosocial behavior," *J. Exp. Psychol., Gen.*, vol. 147, no. 12, pp. 1950–1958, Dec. 2018.

[47] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3730–3738.

[48] S. Abtahi, M. Omidyeganeh, S. Shirmohammadi, and B. Hariri, "YawDD: A yawning detection dataset," in *Proc. 5th ACM Multimedia Syst. Conf.*, New York, NY, USA, Mar. 2014, pp. 24–28.

WANGHUA DENG was born in 1994. He received the B.S. degree in software engineering from the Beijing University of Technology, China, in 2016, where he is currently pursuing the M.S. degree. His research interests include machine learning and deep learning.

RUOXUE WU was born in 1988. She received the B.E. degree in software engineering from Northwest University, China, in 2012, and the M.S. degree in software engineering from Yunnan University, China, in 2016. Her research interests include deep learning and data visualization.