High Accuracy Driver Identification and Status Monitoring System Research

Xiaohan Yuan

Glasgow College, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China

2289066y@student.gla.ac.uk

Abstract. In this paper a driver's identity authentication and condition monitoring system will be proposed, which mainly includes two parts: identity authentication and status monitoring. The driver's facial features are extracted by the convolutional neural network, and then the feature dimensions are compressed to reduce calculation. All the vectors composed of these features will be matched with the face database in the local library, and only matched person is able to start the car. After authentication a camera will collect images of the face as well as its surroundings in real time. Later features are extracted, the vectors composed of these features are used to iteratively update the parameters of Bayesian classifier, thereby following the face position in real time, and according to the track of face motion determines whether the driving state is safe.

1. Introduction

The whole system mainly includes: data collection, face detection, face alignment, feature extraction, face recognition, and face tracking. The specific instructions are as follows:

1.Data collection: Use the camera to capture face images and establish face database;

2.Face detection: Use the skin color Gaussian model to detect the face position;

3.Face alignment: Extract Harris corner features of the face images, construct affine transformation matrix according to the feature points to realize face alignment.

4.Features extraction: extracting Harr-like features from aligned face images, and reducing feature dimensions by principal component analysis to generate feature faces;

5.Face recognition: Using regularized sparse representation for occlusion detection and face reconstruction recognition; 6. Face tracking: For the current frame, sample the image rectangles of several targets (positive samples) and background (negative samples), then extract features with Harr-like, reduce dimensions through sparse measurement matrix. Later use the features whose dimensions have been reduced to train the naive Bayes classifier (including the target and background, belongs to binary classification problem). In the next frame, randomly sample multiple rectangular windows around the target position tracked by the frame in the previous frame, reduce the dimension by the same sparse measurement matrix, extract features, and then use the naive Bayes classifier trained by last frame to classify. The rectangular window which has the largest classification score is recognized as our target face window, thus achieving the goal of target tracking from the current frame to the next frame.

2. Authentication

After the driver gets into the car, the camera first collects the driver's images, then we will detect face region from the captured images, align the detected face images, later face features will be extracted. Harris feature which is the key feature of the face, is commonly applied [1] [2]. In view of the prominent performance of convolutional neural networks in face recognition, this paper mainly uses neural networks to extract facial convolution features [3]. Finally, the feature vector is matched with the authenticated face in the local template library. If all the matching results are lower than the preset threshold, it indicates that the user is an unauthorized user, and then a warning sound is issued to prompt the user to leave, meanwhile alarm information as well as intruder's image will be sent over the network to the owner and the police monitoring center. If match result is higher than the threshold, it is an authorized user. After that, perform other password authentication operations (such as saying the name, age, etc. to see if it matches the local database) to further improve the accuracy of identity authentication. At the same time, a history log related to the driver's identity is searched, and if the driver's history log shows that his unsafe driving state is large, he or she is refused to start the car, otherwise the car can be started and the driving monitoring stage is on. The specific authentication process is as follows:

Face Detection. Face detector is used to find the position of the face in the image, and if there is a face, the coordinates of the bounding box containing each face are returned.

Face alignment. The goal of face alignment is to scale and crop the face image using a set of reference points located at fixed positions in the image. This process usually requires the use of a feature point detector to find a set of face feature points. In the simple 2D alignment case, it is to find the best affine transformation that fits the reference point. More complex 3D alignment algorithms can also achieve face positive, which adjusts the face's posture to the front.

Face representation. In the face representation stage, the pixel values of the face image are converted into compact and discernible feature vectors, also known as templates. Ideally, all faces of the same subject should be mapped to similar feature vectors.

Face matching. While the face matching building module, the two templates are compared to obtain a similarity score, which gives the possibility that the two belong to the same subject.

If you enter the driving state monitoring stage, the system will first extract facial features through classical convolutional neural network, then compress the features by random sampling and use the extracted features to update the Bayesian classifier model parameters. Meanwhile Bayesian classifier is used to track the face in real time, extracting the center position of face image, obtaining the motion trajectory map of the image center. Utilize the neural network that trained in advance to classify and recognize the motion trajectory map to determine whether the driver is in a safe driving state. At the same time, the identification and motion trajectory are packaged into a driving log and upload to server to use as the driving history of the driver.

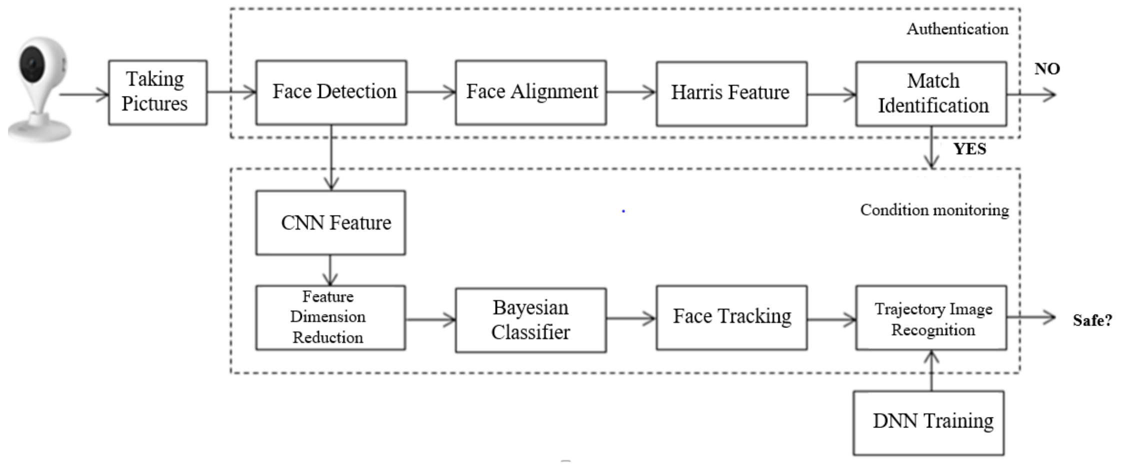

Figure 1. Schematic diagram of system function

3. Face Tracking

Based on the idea of compression tracking [4], the compression tracking method for human face video images is shown in Figure 2. The main tracking steps are as follows:

(a) Model updating: N image frames are sampled around the target position tracked by the t-1 frame, then the feature slices are extracted by convolutional neural network, and feature reduction is performed by random sampling to obtain feature vector for each image, after that update the classifier parameters using the feature vector, as shown in Figure 2(a).

(b) Bayesian classification: these v are classified by Bayesian classifier, and the image slice with the largest classification score is found as the target tracked by the current frame, as shown in Fig. 2(b).

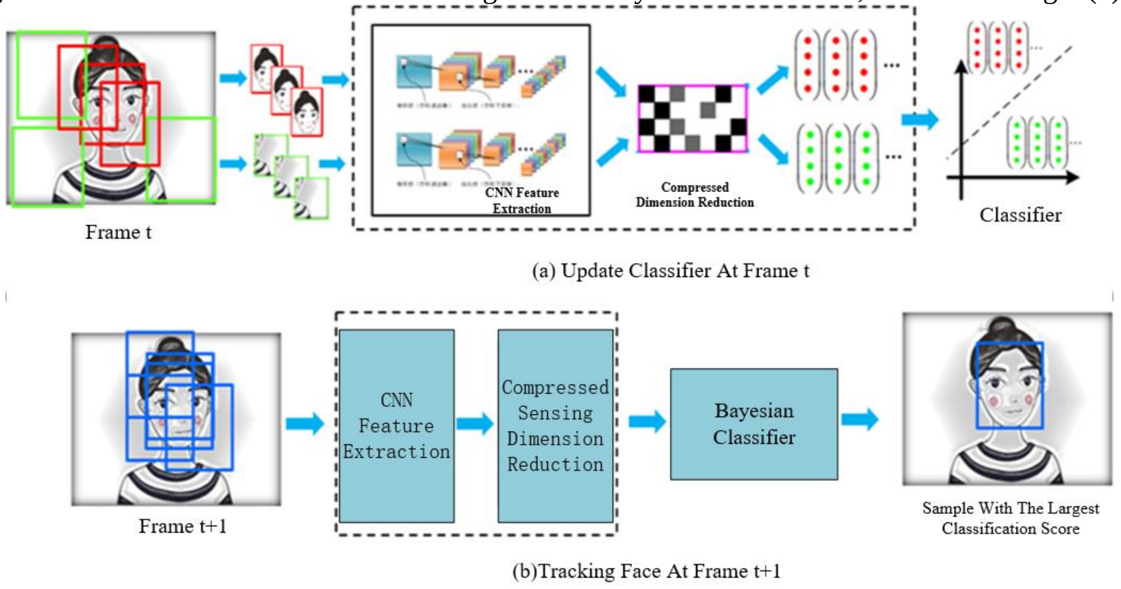
Frame t
CNN Feature Extraction
Compressed Dimension Reduction
Classifier
(a) Update Classifier At Frame t
Frame t+1
CNN Feature Extraction
Compressed Sensing Dimension Reduction
Bayesian Classifier
Sample With The Largest Classification Score
(b)Tracking Face At Frame t+1

Figure 2. Illustration for video tracking with DNN feature

3.1 Convolution feature extraction

Feature extraction of the classic LeNet network is employed for CNN feature extraction here[5][6][7], as shown in Fig. 3.

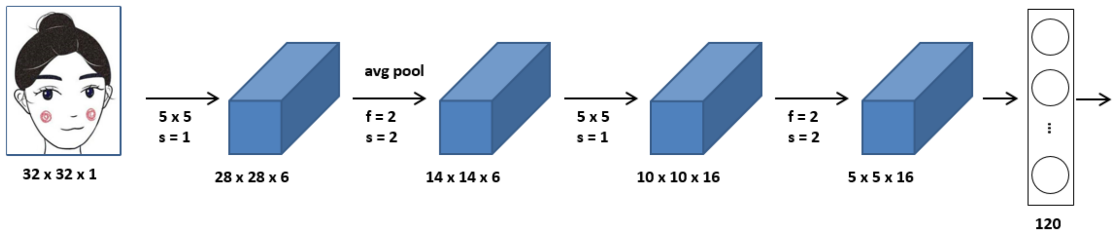

Figure 3. Convolution feature extraction

Layer C1: Convolution

Feature description	Function description
Input image: 32×32 Core size: 5×5 Convolution core type: 6 Output featuremap size: 28 × 28 (32-5+1)=28 Number of neurons: 28×28×6 Trainable parameters: (5×5+1)×6 (5×5=25 unit parameters and one bias parameter for each filter, a total of 6 filters) Number of connections: (5×5+1) ×6×28×28=122304	Perform the first convolution operation on the input image (using 6 convolution kernels with size 5×5) to get 6 C1 feature maps (6 feature maps with size 28×28, 32-5+1=28). Let's take a look at how many parameters are needed. The size of the convolution kernel is 5 × 5. There are 6 × (5 × 5+1)=156 parameters in total, where +1 means that a core has a bias. For convolutional layer C1, each pixel in C1 is connected to 5×5 pixels and 1 bias in the input image, so there are a total of 156×28×28=122304 connections. There are 122,304 connections, but we

	only need to learn 156 parameters, mainly through weight sharing.

Layer C2: Pooling (downsampling)

Feature description	Function description
Input: 28×28 Sampling area: 2×2 Sampling method: 4 inputs are added, multiplied by a trainable parameter, plus a trainable offset. Result by sigmoid Sample type: 6 Output featureMap size: 14×14 (28/2) Number of neurons: 14×14×6 Number of connections: (2×2+1)×6×14×14 The size of each feature map in S2 is 1/4 of the size of the feature map in C1.	The first convolution is followed by a pooling operation, using a 2×2 core for pooling, thus obtaining S2, six 14×14 feature maps (28/2=14). The pooling layer of S2 is to sum the pixels in the 2×2 region of C1 by a weight coefficient plus an offset, and then map the result again. There are also 5×14×14×6=5880 connections.

Layer C3: Convolution

Feature description	Function description
Input: all 6 or several feature map combinations in S2 Convolution kernel size: 5×5 Convolution core type: 16 Output feature Map size: 10×10 (14-5+1)=10 Each feature map in C3 is connected to all 6 or several feature maps in S2, indicating that the feature map of this layer is a different combination of the feature maps extracted from the previous layer. One way to exist is that the first six feature maps of C3 are input with three adjacent feature map subsets in S2. The next six feature maps are input with four adjacent feature map subsets in S2. Then three of the four feature map subsets that are not adjacent are input. The last one takes all the feature maps in S2 as inputs. Then: the training parameters: 6× (3×5×5+1)+6× (4×5×5+1)+3× (4×5×5+1)+1× (6×5×5+1)=1516 Number of connections: 10×10×1516=151600	After the first pooling, it is the second convolution. The output of the second convolution is C3, 16 10x10 feature maps, and the convolution kernel size is 5×5. S2 has 6 14×14 feature maps. Here, 16 feature maps are calculated by special combination of the feature maps of S2.

Layer C4: Pooling

Feature description	Function description
Input: 10×10 Sampling area: 2×2 Sampling method: 4 inputs are added, multiplied by a trainable parameter, plus a trainable offset. Result by sigmoid Sampling type: 16 Output feature Map size: 5×5 (10 / 2) Number of neurons: 5×5×16=400 Number of connections: 16×(2×2+1)×5×5=2000 The size of each feature map in S4 is 1/4 of the size of the feature map in C3.	S4 is the pooling layer, the window size is still 2×2, a total of 16 feature maps, and 16 10×10 graphs of the C3 layer are respectively pooled in 2×2 to obtain 16 5×5 feature maps. There are 5×5×5×16=2000 connections. The connection is similar to the S2 layer.

Layer C5: Convolution

Feature description	Function description

Input: all 16 unit feature maps of the S4 layer (all connected to s4) Convolution kernel size: 5×5 Convolution core type: 120 Output feature map size: 1×1 (5-5+1) Trainable parameters / connection: 120×(16×5×5+1)=48120	The C5 layer is a convolutional layer. Since the size of the 16 maps of S4 layer is 5×5, which is the same as the size of the convolution kernel, the size of the map formed after convolution is 1×1. This produces 120 convolution results. Each is connected to the 16 graphs of the previous layer. So there are (5 × 5 × 16+1) × 120 = 48120 parameters, and there are also 48120 connections.

As shown in the process of convolution feature extraction above, after 5 layers of feature extraction, 120 features would be obtained for an image with a size of 32×32, and then random sampling of features can be carried out to further compress the feature dimension.

3.2 Feature Compression

It can be known from the sparse perception theory that a low-dimensional compressed subspace can be obtained by projecting the original image feature space through a very sparse measurement matrix that satisfies the RIP condition. The low-dimensional compressed subspace can well preserve the information of the high-dimensional image feature space. Therefore, features of the foreground target and the background are extracted by sparse measurement matrix, used as positive and negative samples for online studying to update the classifier. Then the naive Bayes classifier is applied to classify the target image to be tested of the next frame image.

An n×m random matrix R which is able to transforms the x (m-dimensional) of a high-dimensional image space into a low-dimensional space v (n-dimensional), where n is much smaller than m, the mathematical expression is: v = R×x .

In the most ideal case, it is of apparently expected that the low-dimensional v can completely retain the information of the high-dimensional x, or the distance relationship of the samples x in the original space, so that it is meaningful to classify in the low-dimensional space.

Johnson-Lindenstrauss inference shows that an appropriate high-dimensional subspace (which needs to be smaller than the original spatial dimension) can be randomly selected, the distance between two points in the original space is projected into this subspace, which can retain this distance relationship with high probability. (K+1 measurements are sufficient to accurately recover the K-sparse signal in the N-dimensional space). Baraniuk proved that a random matrix that satisfies the Johnson-Lindenstrauss inference also satisfies the restricted isometry property (RIP) condition in compressed sensing theory. Thus when the random matrix R satisfies Johnson-Lindenstrauss inference then if x is compressible (or namely sparse), we can recover x from v with great possibility by minimizing the error.

As indicated by Ref. [4], a typical measurement matrix that satisfies the RIP condition is random Gaussian matrix with matrix elements satisfying N(0,1) distribution. However, if the dimension of m is relatively large, the matrix is still dense, and its operation as well as storage consumption is relatively large. A very sparse random measurement matrix can used [4], whose matrix elements are defined as:

$$r_{ij} = \sqrt{s} \times \begin{cases} 1 & \text{with probability } \dfrac{1}{2s} \\ 0 & \text{with probability } 1-\dfrac{1}{s} \\ -1 & \text{with probability } \dfrac{1}{2s} \end{cases} \tag{1}$$

In this paper, s=m/4, each row of the matrix R only needs to calculate the value of c (less than 4) elements. So its computational complexity is O(c*n). In addition, we only need to store the non-zero elements of R, so there is very little storage space required.

Therefore, compression based feature compression is shown in Figure 4 below. Using a nxm sparse matrix, the x (m-dimensional) of a high-dimensional image space can be transformed into a low-dimensional space v (n-dimensional). The mathematical expression is: v = R x ;. The blue arrow indicates an element in a non-zero element perceptual x from a row of the measurement matrix R, equivalent to a square window filter and a grey-scale convolution of a fixed position of the input image.

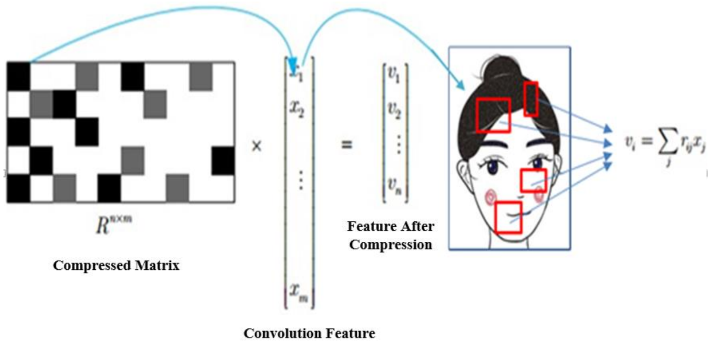

Figure 4. Feature compression

4. Bayesian Classifier

For each sample z (m-dimensional vector), its low-dimensional representation is v (n-dimensional vector, n is much smaller than m). It is assumed that the elements in v are independently distributed. It can be modelled by the Naive Bayes classifier.

$$H(\mathbf{v}) = \log\left(\frac{\prod_{i=1}^{n} p(v_i|y=1)\,p(y=1)}{\prod_{i=1}^{n} p(v_i|y=0)\,p(y=1)}\right) = \sum_{i=1}^{n} \log\left(\frac{p(v_i|y=1)}{p(v_i|y=0)}\right) \tag{2}$$

Here y ∈ {0, 1} represents the sample label, y = 0 represents the negative sample, and y = 1 represents the positive sample, assuming that the prior probabilities of the two classes are equal. p (y = 1) = p (y = 0) = 0.5. Diaconis and Freedman proved that random projections of high-dimensional random vectors are almost Gaussian. Therefore, we assume that the conditional probabilities p(vi|y=1) and p(vi|y=0) in the classifier H(v) also belong to the Gaussian distribution and can be described by four parameters:

$$\left(\mu_i^1, \sigma_i^1, \mu_i^0, \sigma_i^0\right),\ p(v_i|y=1) \sim N\left(\mu_i^1, \sigma_i^1\right),\ p(v_i|y=0) \sim N\left(\mu_i^0, \sigma_i^0\right) \tag{3}$$

The above four parameters are incrementally updated as the following:

$$\mu_i^1 \leftarrow \lambda\mu_i^1 + (1-\lambda)\mu^1$$

$$\sigma_i^1 \leftarrow \sqrt{\lambda\left(\sigma_i^1\right)^2 + (1-\lambda)\left(\sigma^1\right)^2 + \lambda(1-\lambda)\left(\mu_i^1 - \mu^1\right)^2} \tag{4}$$

where $\lambda > 0$ is the study parameter, representing the rate of update.

For the first iteration, there is:

$$\mu^1 = \frac{1}{n}\sum_{k=0|y=1}^{n-1} v_i(k),\ \sigma^1 = \sqrt{\frac{1}{n}\sum_{k=0|y=1}^{n-1}\left(v_i(k) - \mu^1\right)^2} \tag{5}$$

5. Status Monitoring

Status monitoring can be trained and identified using classical support vector machines (SVM) or neural network for binary classification. This paper uses LeNet network for training and recognition. Its feature extraction structure is shown in Section 2.1. The difference is that a fully connected layer and a softmax layer are used for classification. The number of output nodes for the fully connected layer and softmax is 2, which means safety and unsafety, respectively.

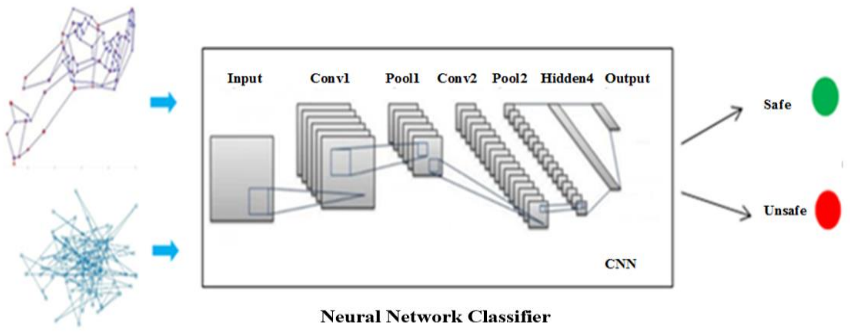

Figure 5. Schematic diagram for driving state monitoring

Layer C6: Fully-Connection

Feature description	Function description
Input: c5 120-dimensional vector Calculation method: Calculate the dot product between the input vector and the weight vector, plus an offset, and the result is output by the sigmoid function. Trainable parameters: 84×(120+1)=10164	The 6th layer is a fully connected layer. The F6 layer has 84 nodes, corresponding to a 7×12 bitmap, with -1 for white and 1 for black, so that the black and white of the bitmap for each symbol corresponds to an encoding. The training parameters and the number of connections for this layer are (120+1)×84 = 10164.

Layer C7: Softmax

Feature description	Function description
Input: c5 120-dimensional vector Calculation method: Calculate the dot product between the input vector and the weight vector, plus an offset, and the result is output by the sigmoid function. Trainable parameters: 2×(84+1)=190	The 7th layer is a fully connected layer. The F7 layer has 2 nodes, which correspond to two states, safe and unsafe. The training parameters and the number of connections for this layer are (84 + 1)×2 = 190.

6. Demo Description

Based on literature [8] document code, the convolutional neural network feature extractor is added, and the driver-based identity authentication and status monitoring demonstration interface is constructed as shown in Fig. 6. The video monitoring area on the left side can monitor driver's driving status in real time. The middle interface is the driver's face detection as well as face recognition matching area. And the right side is the control panel.

The entire demo is controlled by five steps in the Command panel: Step 1, loading the video, simulating the camera to capture the video image in real time; Step 2, face detection, detecting the driver's face from the video image, where Threshold is the decision threshold of the face detection process. It is determined as a face only when the feature operation result is greater than Threshold, and the detection result is displayed on the Detected Face panel; step 3, face recognition, via matching process of the detected face and the face in the local template library, through face alignment, feature extraction and matching process as shown in Figure 1 to complete face recognition and identity authentication. Only the face that can match the local database enables driver to start driving. In the process of face recognition, detected face may be distorted or shield, the detected face is then restored by the reconstruction algorithm and displayed on the Recovered Face panel. Step 4, face tracking, through the processing shown in the second section, namely: convolution Feature extraction, feature compression, Bayesian classification and other operations that complete face tracking, meanwhile recording the face region centre in real time to analyse follow-up motion trajectory; Step 5, motion trajectory analysis, click “Status Analysis” on the “Command” panel to enter the neural network shown in Figure 5 for safety state analysis. By analysing the human face motion trajectory, the system can show driver’s driving status. If it’s not safe, issue a warning message and record the driver's driving behaviour in the local log as part of the subsequent authentication decision conditions, only those with good driving records can start the car after being recognized by his or her face. Figure 7 shows the driver's face motion trajectory. The motion trajectory map is sent to the state judgment system shown in Figure 5. It should be judged as unsafe driving. At this time, the “Unsafety?” button on the panel will be highlighted. , showing unsafe driving.

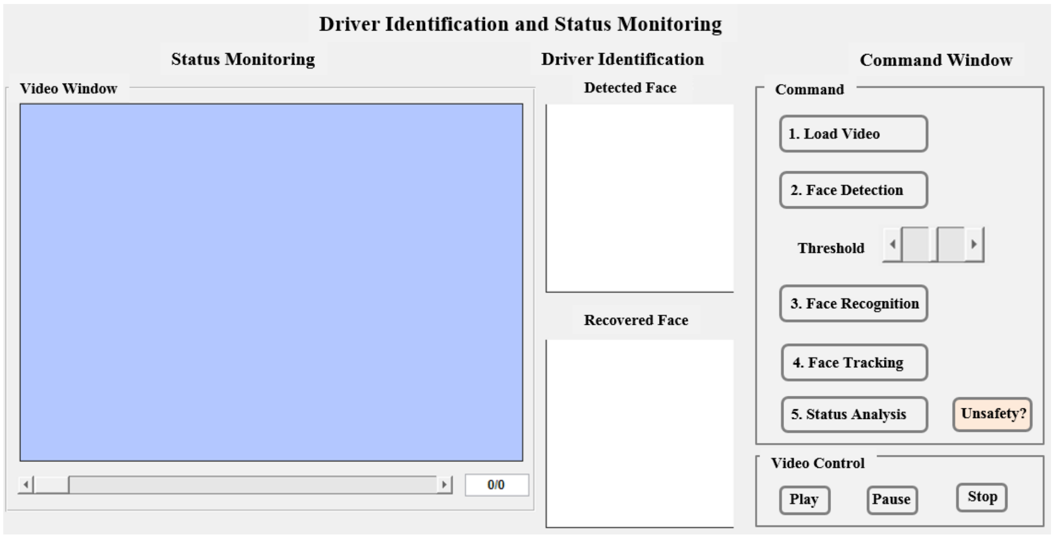

Figure 6. Demo of Driver identification and status monitoring

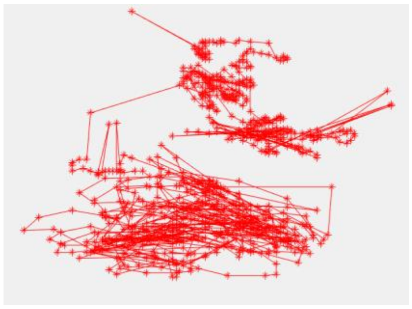

Figure 7. Face trajectory during driving.

7. Conclusion

Based on the existing video compression tracking method, this paper draws on the application method of deep learning in face recognition, extracts the face depth feature by convolutional neural network and then uses the compression tracking method for face tracking. This method is applied to the driver identity authentication and driving state monitoring scenarios so that the compression tracking and convolutional neural networks are well combined and applied to the actual scene. However, in the state monitoring, only two-dimensional motion trajectory is utilized, and lack time dimension. Therefore, the next step will be to train the driving state in deep learning from four dimensions (up, down, left, right, before and after, time) to achieve better condition monitoring.

References

[1] Chen Yongfei, Liu Xinming. (2008) Eye detection in color face image based on skin and Haar feature. Computer Engineering and Applications[J]. 44(33):174-176.
[2] Chen Shigang, Ma Xiaohu. (2011) Face detection based on Multi-Gaussian skin color segmentation and Haar-like intensity features[J]. 27(3):30-34.
[3] Trigueros, Daniel Sáez, Meng L, Hartnett M. (2018) Face Recognition: From Traditional to Deep Learning Methods[J].
[4] Zhang K, Zhang L, Yang M H. (2012) Real-time compressive tracking[C]. European Conference on Computer Vision. Springer Berlin Heidelberg. pp.864-877.
[5] Hong S, Wu Q, Xie H, et al. (2016) A Novel Coupled Template for Face Recognition Based on a Convolutional Neutral Network[C]. Sixth International Conference on Intelligent Systems Design & Engineering Applications. IEEE.
[6] El-Sawy A, El-Bakry H, Loey M. (2016) CNN for Handwritten Arabic Digits Recognition Based on LeNet-5[J].
[7] Yong L I, Xiao-Zhu L, Meng-Ying J. (2018) Facial Expression Recognition with Cross-connect LeNet-5 Network[J]. Acta Automatica Sinica.
[8] Wang Wenfeng, Li Daxiang, Wang Dong. (2018) Fundamental and Practice of Face Recognition[M]. Electronic Industry Press.