Online Face Recognition System based on Local Binary Patterns and Facial Landmark Tracking

Marko Linna[1], Juho Kannala[1], and Esa Rahtu[1]

Department of Computer Science and Engineering, University of Oulu, P.O. Box 4500, 90014 University of Oulu, Finland,
marko.linna@student.oulu.fi

Abstract. This paper presents a system for real-time face recognition. The system learns and recognizes faces from a video on the fly and it doesn't need already trained database. The system consists of the following sub-methods: face detection and tracking, face alignment, key frame selection, face description and face matching. The system detects face tracks from a video, which are used in learning and recognition. Facial landmark tracking is utilized to detect changes in facial pose and expression in order to select key frames from a face track. Faces in key frames are represented using local binary patterns (LBP) histograms. These histograms are stored into the database. Nearest neighbor classifier is used in face matching. The system achieved recognition rate of 98.6% in offline test and 95.9% in online test.

1 Introduction

Majority of the state-of-the-art face recognition methods focus on a specific sub-problem and are usually computationally intensive [5, 6, 7, 8]. Therefore it is not straightforward to use them in online face recognition systems, where learning and recognition must happen in real-time while processing a video. They may not be suitable for applications, which require short response times (e.g. human-robot interaction).

In this paper, we present real-time face recognition system for practical application. We don't focus on a specific sub-problem, but consider the whole system. Our method is a complete package including learning, recognition and database update. The method operates in real-time while processing a video. Already trained database is not mandatory, because the method learns on the fly. Possible cases for our method are human-robot interaction, door camera, greeter, home security and home entertainment systems (e.g. integrated on a TV). The software implementation of our method is publicly available.

For real-time video based face recognition the computational lightness, as well as accuracy, is very important. To cope with this, we selected LBP for face descriptor [3] and fast facial landmark tracker [13] for tracking detected faces, omitting the need of computationally heavy face detection for every frame.

Our method detects face tracks from a video. Face tracks are used in recognition and learning. In the recognition, face representations of n frames from the

beginning of the face track are used to find out nearest match from the database. Nearest neighbor classifier is used in matching. Distances between face representations are measured using weighted Chi-square. In the learning, key frames are selected from the face track so that they represent the face in different poses and expressions. Then face representations are extracted from the key frames and stored into the database.

2 Related Work

Face recognition may typically consist of sub-problems like face detection, face alignment, facial landmark tracking, face description and face matching. Most common face detection methods use Haar-like features and AdaBoost [17]. Some face recognition methods focus on alignment [9], [5]. Several methods are introduced for facial landmark tracking [11], [13], [15]. Common methods used in face description are principal component analysis (PCA) [1], linear discriminant analysis (LDA) [2] and local binary patterns (LBP) [3]. Lately deep learning has been studied a lot and methods based on it seem to give most promising results [5, 6, 7].

The face recognition studies usually focus on a specific sub-problem and don't consider the face recognition at a system level. Few works focus on purely real-time or online face recognition systems. Most related to our work is [4], which provides real-time face recognition and online learning. The method increases the separability of the classes by maximizing the inter-class and minimizing the intra-class variations. Only the most representative samples are selected in order to gain robustness against illumination and pose changes. Additionally, to allow for classification of unknown faces, the method finds the relation between the distance of classification and the certainty of that classification. To further improve recognition performance multiple frames are used in classification. The method uses class-weighted average PCA (CAPCA) for face description, while our method uses LBP. The CAPCA uses feature size of 4096, while our LBP descriptor size is 2301. Our descriptor size is roughly half of the CAPCA size, which means that our method is potentially faster in face matching and requires a smaller database. In addition, LBP is known to be more robust against illumination and pose changes than holistic approaches like PCA [3] based methods, so our method is likely to perform better in challenging environments.

It is not trivial to compare our method against state-of-the-art methods, because their software may not be available or it might be very difficult to implement them to work in real-time. Therefore, we don't compare our method to other methods.

3 Method

The usage the method is best understood with a practical example, which follows. The method is running in a greeter robot. The robot is "just born" and doesn't know anybody. Then, a boy comes to say hello to the robot. Robot detects the

face of the boy, then it asks name and the boy answers. The robot starts to memorize the face until the boy leaves. Time passes and several other persons visit the robot. Next day, the same boy comes to greet the robot. Now the robot remembers the face of the boy and says hello with the name of the boy.

Our method consists of *recognition* and *learning*, which both needs face tracks. These face tracks are detected from a video by our method. In the recognition, the beginning of the detected face track is used to find nearest match, meaning that n frames from the beginning of the face track are used in database search. The n is determined by the search time and the video frame rate. If the total frame count in the face track is smaller than the search time multiplied with the video frame rate, the n is the total frame count. Further, the n decreases while the database gets larger, because it takes more time to find nearest match for a single frame. The search and matching is explained in more detail in Section 3.5. Face representation is extracted from each frame used in the search. This representation is in form of LBP histogram and is explained in more detail in Section 3.4. The database search is started immediately when a new face track is detected and it happens in real-time while tracking a face. It lasts for a specified time, after which the person identity is either known or not known (see Fig. 1).

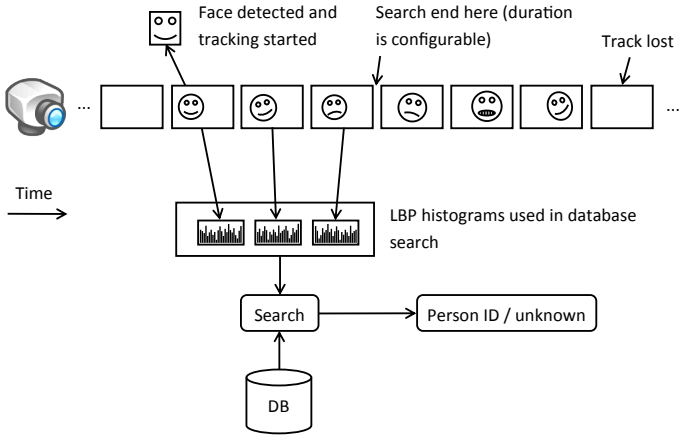

Fig. 1. Recognition: n subsequent frames from the beginning of a track are used in classification. If multiple classes returned, the one with shortest distance is taken.

In the learning, the key frames are selected from the detected face track so that they represent the face in different pose and expression (e.g. person is talking or turning head). The selection of key frames is explained in more detail in Section 3.3. Face representations are extracted from the key frames and stored into the database as LBP histograms. Learning happens in real-time while tracking a face (see Fig. 2).

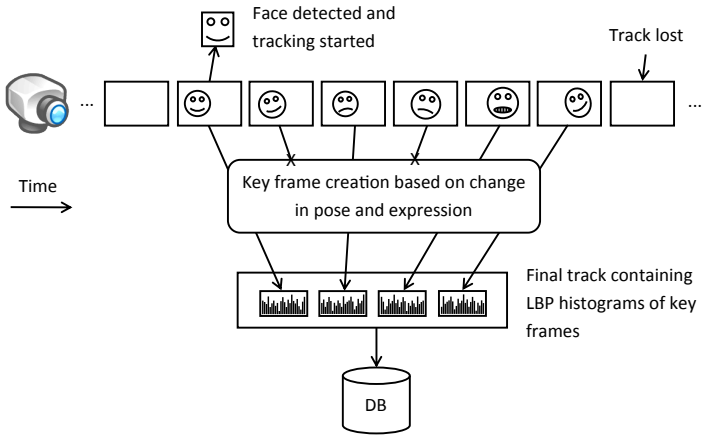

Fig. 2. Learning: face representations of the selected key frames of a track are stored into the database. Selection happens if there is large enough change in pose and expression compared to the last key frame.

The recognition and the learning happens in online fashion. At the beginning the face database is empty, but gets updated while videos are processed and new face tracks detected. If some person is already in database, the LBP histograms extracted from the key frames of the detected face track are added to the person in question, otherwise new person is created. The database consists of entities: *person*, *track* and *histogram*. It can have $0...n$ persons, a person can have $1...m$ tracks and a track can have $1...k$ histograms.

Let's consider an example where the system is running on a PC with webcam. Initially the database is empty. Then person X comes to front of the webcam. The system detects the face and starts tracking it. Because the database is empty, new person is created right away into the database. LBP histograms are added to the database while person X moves her/his head and changes expression. Now person X moves away, system loses the face track and stops adding new histograms to the database. So far the system added 3 new histograms. In reality there would be much more, but for simplicity smaller numbers are used. What the database looks like now is described on left in Fig. 3.

Now person Y comes to front of the webcam. The face is detected and database search initialized. The search uses all the available frames (not only key frames) from the beginning of the face track to find close enough match. The search continues for a specific time (or until whole database is gone through), then the classification is done. The *unknown* class is returned and new person is added into the database along with the histograms of the key frames of the detected face track so far. Histograms are added until suddenly person Y changes head pose so that the system loses track of the face. Then she/he watches again directly to webcam and a new track is initialized for the same person. The search is started and now it finds the *person id* of the person Y. New track is added for

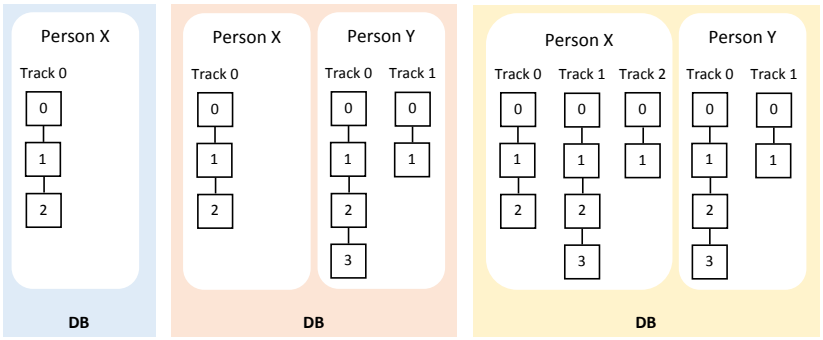

Fig. 3. Database state where squares are LBP histograms of key frames. Left: after adding person X. Center: after adding person Y. Right: after updating person X.

person Y and following histograms are added to it. After person Y moves away, the database looks like what is described in center in Fig. 3.

For next, person X comes again in front of the camera. The search finds *person id* of the person X correctly and the following histograms are added for it. Then the person X looks away for a while and then back to camera, so that the track is lost and new one started. Also this time search found correct *person id* and just adds next histograms to it. After the person X moves away, she/he has 3 tracks in the database. What the database looks at the end, is described on right in Fig. 3.

Our method consists of the following key components: *face detection and tracking*, *face alignment*, *key frame selection*, *face description* and *face descriptor matching*. These all are explained in their own sub-sections below.

3.1 Face Detection and Tracking

We considered two facial feature detection methods for locating and tracking facial landmarks. The first one was using conditional regression forest [11] and the second one was based on updating a discriminative facial deformable model [13]. The software, [12] and [14] respectively, was available for both methods. According to our experiments, the second method turned out to be faster and more robust with big head rotations so it was selected for the final experiments reported in section 4. The software of the selected method is called Chehra. Chehra provides 49 facial landmark points and 3D head-pose estimation when a face is detected. After initial detection, Chehra starts tracking already found landmarks, until they are lost. After losing landmarks, Chehra starts detecting faces again. This detection phase turned out to be computationally very slow compared to the tracking phase. Internally Chehra uses OpenCV's Viola-Jones face detection algorithm [17].

In addition, a third method, using an ensemble of regression trees to estimate face's landmark positions [15] has been recently published. This method

is reported to achieve real-time performance with high quality predictions and could be used in our system, but we have not yet experimented with it.

3.2 Face Alignment

Alignment of the face image and landmarks are used for geometric normalization of the input image. The landmark points of the eyes are used to determine a region of interest (ROI) around the face. For every facial frame, the ROI is found out by first calculating distance between eyes and then multiplying that distance with certain constants to get distances from point between eyes to edges of the ROI, thus giving 4 corner points. The corner points of the ROI are then rotated according to the head-pose (see Fig. 4). The aim is to size ROI to get minimum amount of background pixels, and exclude hair. This way of getting ROI, from eye points only, works well if there are no very big out-of-plane head rotations. From the face ROI, the transformation matrix T is calculated. Then it is used to do perspective transformation for the landmark points and the face image in order to get rid of rotation, scaling and translation.

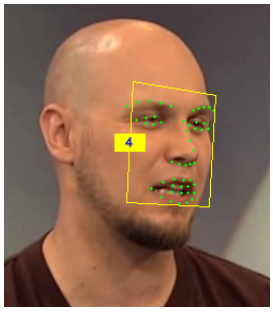

Fig. 4. A frame with facial landmarks and face ROI.

There are three factors in 3D head-pose: *yaw*, *pitch* and *roll*. The roll is in-plane rotation and is easily compensated. The yaw and pitch are out-of-plane and more problematic because the face will be partially occluded when the rotation angle is high. So aligning using only perspective transformation doesn't give good results with highly out-of-plane rotated faces. However, a small rotation is not a problem. Alternative, more complicated, alignment method was presented by Taigman et al. in their paper [5]. They use 3D model of a face for alignment.

Because LBP is not very robust to noise [10], aligned face image is smoothed with median filter using kernel size 3. This blurs the image only a little and gives better results than without smoothing, especially with noisy source images. After smoothing, the face image is converted into grayscale, thus giving the final normalized face image. See Fig. 5 for illustration of the alignment and normalization.

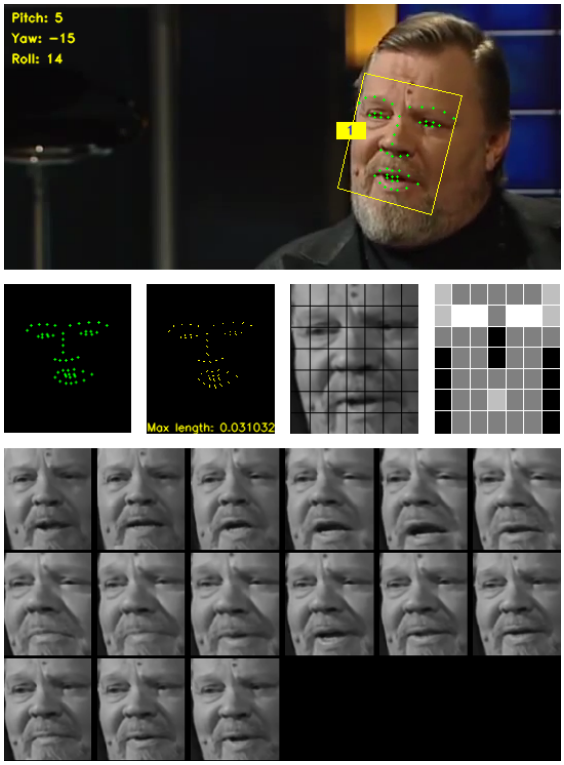

Fig. 5. Normalization and key frame creation. Top: a frame with facial landmarks and face ROI. Center: normalized landmarks, delta vectors of landmarks, normalized face image on 7x7 grid and weight map where black square has weight 0, dark gray 1, light gray 2 and white 4. bottom: Subsequent key frames of a track while expression and pose are changing.

3.3 Key Frame Selection

Our method uses facial landmark tracking to detect changes in facial pose and expression. This is important when selecting the key frames from a face track. The key frames should represent the face in different pose and expression, which can lead to better performance. For each facial frame, a delta vector is calculated for every landmark. The delta vectors show landmark movement between current frame and the last key frame. If at least one delta vector length exceeds a certain limit, a new key frame is selected. See Fig. 5 for illustration of the key frame selection.

3.4 Face Description

We selected LBP for face description, because it has proven to work both fast and accurate in face recognition tasks. The LBP image and histogram is created from

the normalized face image using so called *extended LBP operator* [3]. It defines local neighborhood as a set of sampling points evenly spaced on a circle centered at the pixel to be labeled, allowing any radius R and number of sampling points P.

In addition, we selected another extension, *uniform patterns*. A local binary pattern is called uniform if it contains at most two bitwise transitions from 0 to 1 or vice versa when the bit pattern is considered circular. Basically this means that a uniform pattern has no transitions or precisely two transitions. When computing LBP histograms, uniform patterns are used so that the histogram has a separate bin for every uniform pattern, and one extra bin for all non-uniform patterns. So there is $P(P-1)+2$ bins for uniform patterns and 1 shared bin for all non-uniform patterns. The added two uniform pattern bins are cases when there are no transitions at all. E.g. 00000000 and 11111111, if P is 8. With uniform patterns, the histogram size is reduced considerably and it can still detect important features.

The face image, from which the LBP histogram is created, is divided into 7×7 regions (see Fig. 5) using region size 18×21 in pixels. The $\mathrm{LBP}^{u2}_{8,2}$ operator is used separately to each region, which are combined to form the final LBP histogram. According to [3] this is a good trade-off between recognition performance and feature vector length. The $\mathrm{LBP}^{u2}_{8,2}$ operator has 8 sampling points, radius of 2 and it use uniform patterns. Thus there are $8(8-1)+3=59$ bins for a single region.

3.5 Face Descriptor Matching

Our method uses nearest neighbor classifier in LBP histogram matching. To measure distances between LBP histograms, weighted Chi-square (χ^2) is used. It is defined as

$$\chi^2_w(S,M)=\sum_{i,j} w_j \frac{(S_{i,j}-M_{i,j})^2}{S_{i,j}+M_{i,j}} \tag{1}$$

where S and M are histograms to be compared, indices i and j refer to ith bin in histogram corresponding to the jth region and w_j is the weight for region j. We selected weights based on the experiments made by Ahonen et al. [3]. Their method emphasize eye, eyebrow, mouth and temple regions and ignore parts of cheeks and bridge of the nose.

Classification is done for n frames in a detected face track. For each frame, the method starts to calculated distances between the LBP histogram in the frame and histograms in the database, consecutively. The order is such that subsequent histograms are from different person. Distances are calculated until under some histogram the distance is below the *distance threshold* or all histograms are gone through and all distances are above the threshold. In the first case, classification to the person owning the histogram in question happens. Otherwise classification to unknown class happens. If different frames results in different classes, the one with the shortest distance is selected.

4 Experiments

4.1 Dataset

We tested the method on the videos of the Honda/UCSD video database [16]. The dataset contains 20 persons and each person (except one) has at least two videos. Videos are recorded indoor at 15 frames per second. All the videos contain significant in-plane and out-of-plane head rotations. In addition, some videos contain changes in expression and partial occlusions. Videos are separated into training and testing videos.

4.2 Offline test

In the offline test, we used training videos of the dataset to create training database. The training database contains 20 persons, 96 tracks and 2888 histograms (about 144 histograms/person). For testing we used testing videos of the dataset. During testing the database was not updated so every face track was tested against the same database. All frames from a face track were used in testing. In testing videos, there were 278 detected face tracks. The average frame count of the detected face tracks were 47.

The results with three different distance thresholds 0.57, 0.27 and 0.42 are presented in Fig. 6 and Fig. 7 as confusion matrices. Recognition rates with different thresholds were 62.2%, 64.0% and 98.6% respectively. Recognition rate was calculated by dividing all the correct classifications (values in diagonal of a confusion matrix) with the total number of the tested tracks, which is 278 in this case.

4.3 Online test

In the online test, the database was initially empty and was updated every time when new face track were tested. Every later track was tested against a bigger database. Both training and testing videos of the dataset were used in testing. In videos, there were 389 detected face tracks. The average number of frames used in database search were 17 for every detected face track. The search time was 1.5 seconds and distance threshold 0.37 was used. A little stricter threshold 0.37 was selected (0.42 in offline test), because it performed better in online test. Two things were tested. First, what is systems ability to detect new persons. Second, does the system recognize already trained persons.

The new person detection rate was 100%. It was calculated by considering only the first tracks from the persons, which were not already trained. There were 21 of these tracks (note there were extra unknown person in testing videos in the background resulting a face track creation). Then it was find out how many of these track resulted in unknown class. That value was divided with 21, giving the new person detection rate. The recognition rate was 95.9%. It was calculated by considering all the face tracks excluding the first track from each person. The number of these tracks were $389 - 21 = 368$. By dividing the

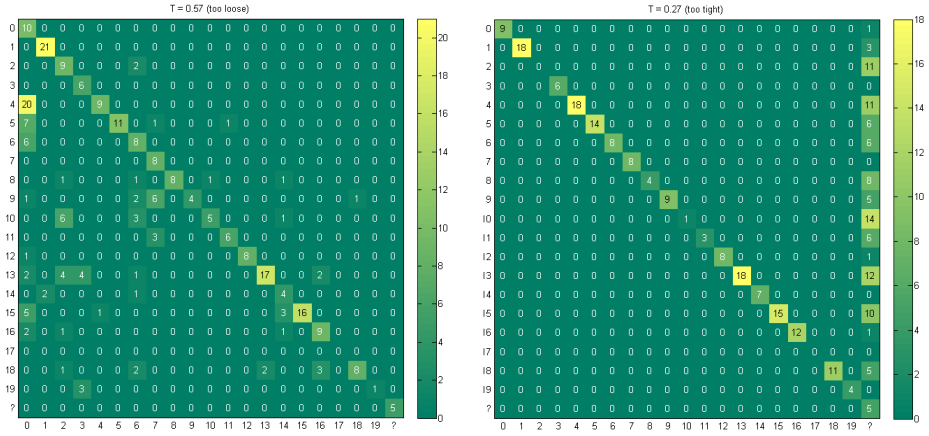

Fig. 6. Confusion matrices showing offline test results. In y-axis are actual persons and in x-axis recognized persons. Left: threshold 0.57 (too loose). The first trained person is emphasized because it is searched first and the search stops when the distance is below the threshold. Right: threshold 0.27 (too tight). The unknown class is emphasized, because there are less histograms whose distance is below the threshold.

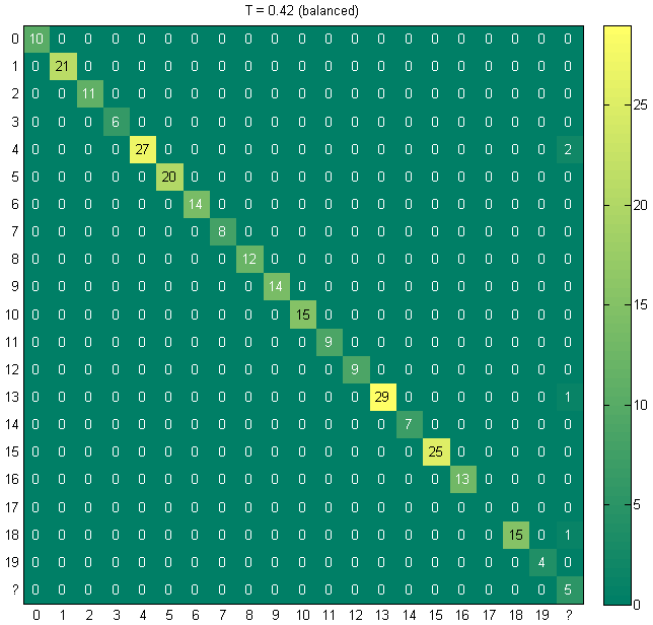

Fig. 7. Confusion matrix showing offline results. In y-axis are actual persons and in x-axis recognized persons. Threshold 0.42. Based on recognition rate, this was the best choice.

number of correct recognitions with this track count, the recognition rate was calculated. The number of false recognitions (classification to wrong/unknown class) in the tracks were 15. The number of the correct recognitions were then $368 - 15 = 353$. It can be seen that the recognition rate 95.9% is a bit worse compared to 98.6% in offline test. This is due to fact that the accuracy is better the more frames from the detected face track were used in the search. In online test this was 17 and in offline test 47. It can be figured out that longer search time results in better recognition rate.

The size of the database was 8623 histograms (about 80 megabytes) at the end of the test. The database can grow even bigger and the method would still perform well, but the search strategy have to be improved for very large databases or search will get inefficient. Tests were ran on 6 years old PC with Intel Core i7 920 2.66 GHz processor and 6 gigabytes memory. Results from the both tests are summarized in the Table 1.

Table 1. Test results

Test name	Database updated	Tracks tested	Average frame count	Recognition rate	New person detection rate
Offline test	No	278	47	98.6%	-
Online test	Yes	389	17	95.9%	100%

5 Conclusion

We introduced online face recognition method based on LBP and facial landmark tracking. The method is a system including real-time face recognition and learning. The method detects face tracks from a video. In learning, key frames are selected from a face track by using facial landmark tracking so that they represent the face in different pose and expression. Faces in key frames are represented by using LBP histograms. The database is updated with LBP histograms on the fly while processing videos. Already trained database is not required. The method uses nearest neighbor classifier in LBP histogram matching.

Our method is computationally efficient and achieved good results. These originate from the use of LBP and fast facial landmark tracking, which omits the need to detect faces from every frame. We carried out two tests on the Honda/UCSD video database: *offline test* and *online test*. They resulted in recognition rates of 98.6% and 95.9% respectively. In addition, the software implementation of our method is publicly available.

References

1. Turk, M., Pentland, A.: Eigenfaces for Recognition. In: Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86 (2006)

2. Belhumeur, P.N., Hespanha, J.P., Kriegman, D.: Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 711–720 (1997)
3. Ahonen, T., Hadid, A., Pietikäinen, M.: Face Description with Local Binary Patterns: Application to Face Recognition. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, 2037–2041 (2006)
4. Gaisser, F., Rudinac, M., Jonker, P.P., Tax, D.: Online Face Recognition and Learning for Cognitive Robots. In: 2013 16th International Conference on Advanced Robotics (ICAR), pp. 1–9 (2013)
5. Taigman, Y., Yang, M., Ranzato, M.A., Wolf, L.: DeepFace: Closing the Gap to Human-Level Performance in Face Verication. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1701–1708 (2014)
6. Sun, Y., Wang, X., Tang, X.: Deeply learned face representations are sparse, selective, and robust. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2892–2900 (2015)
7. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A Unied Embedding for Face Recognition and Clustering. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 815–823 (2015)
8. Lu, C., Tang, X.: Surpassing Human-Level Face Verification Performance on LFW with GaussianFace. In: Proc. 29th AAAI Conference on Artificial Intelligence (AAAI), pp. 3811–3819 (2015)
9. Wagner, A., Wright, J., Ganesh, A., Zhou, Z., Mobahi, H., Ma, Y.: Toward a Practical Face Recognition System: Robust Alignment and Illumination by Sparse Representation. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 2, pp. 372–386 (2012)
10. Chen, J., Kellokumpu, V., Zhao, G., Pietikäinen, M.: RLBP: Robust Local Binary Pattern. In: Proc. British Machine Vision Conference (BMVC), pp. 1–10 (2013)
11. Dantone, M., Gall, J., Fanelli, G., Van Gool, L.: Real-time Facial Feature Detection using Conditional Regression Forests. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2578–2585 (2012)
12. Facial Feature Detection (Matthias Dantone, Juergen Gall, Gabriele Fanelli, Luc van Gool), `http://www.dantone.me/projects-2/facial-feature-detection/`
13. Asthana, A., Zafeiriou, S., Cheng, S., Pantic, M.: Incremental Face Alignment in the Wild. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1859–1866 (2014)
14. Chehra (Akshay Asthana, Stefanos Zafeiriou), `https://sites.google.com/site/chehrahome/home/`
15. Kazemi, V., Sullivan, J.: One Millisecond Face Alignment with an Ensemble of Regression Trees. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1867–1874 (2014)
16. Lee, K.C., Ho, J., Yang, M.H., Kriegman, D.: Visual Tracking and Recognition Using Probabilistic Appearance Manifolds. Computer Vision and Image Understanding, vol. 99, no. 3, pp. 303–331 (2005)
17. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, 511–518 (2001)