Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 1680 – 1695

doi:10.3906/elk-1809-75

Efficient hierarchical temporal segmentation method for facial expression sequences

Jiali BIAN, Xue MEI*, Yu XUE, Liang WU, Yao DING
College of Electrical Engineering and Control Science, Nanjing Tech University, Nanjing, P.R. China

Received: 12.09.2018 • **Accepted/Published Online:** 19.01.2019 • **Final Version:** 15.05.2019

Abstract: Temporal segmentation of facial expression sequences is important to understand and analyze human facial expressions. It is, however, challenging to deal with the complexity of facial muscle movements by finding a suitable metric to distinguish among different expressions and to deal with the uncontrolled environmental factors in the real world. This paper presents a two-step unsupervised segmentation method composed of rough segmentation and fine segmentation stages to compute the optimal segmentation positions in video sequences to facilitate the segmentation of different facial expressions. The proposed method performs localization of facial expression patches to aid in recognition and extraction of specific features. In the rough segmentation stage, facial sequences are segmented into distinct facial behaviors based on the similarity between sequence frames, while similarity between segments is computed to obtain optimal segmentation positions in the fine segmentation stage. The proposed method has been evaluated in experiments using the MMI dataset and real videos. Experiment results compared to other state-of-the-art methods indicate better performance of the proposed method.

Key words: Clustering, temporal segmentation, similarity calculation, facial image analysis

1. Introduction

Much attention has been focused on recognizing and understanding the facial actions in videos in the past three decades and it remains an active area of research in the field of computer vision. Temporal segmentation of facial expression sequences aims to split the expression action units in the video sequences, all in an unsupervised fashion. An expression action unit represents the whole change process of facial actions from the beginning to the end of an expression. Most of the analysis and synthesis methods in the area consider dividing streams of video data into perceived segments. An effective solution in this area can be applied in various areas, including automatic facial image analysis [1], facial expression recognition in video sequences [2], human emotion monitoring [3], automatic collection of large-scale face datasets, and so on.

Temporal segmentation of facial expressions mainly faces the following challenges. It is difficult to find a suitable metric to distinguish among different expressions, and to deal with the complexity of facial muscle movement and the uncontrolled environmental factors in the real world. Currently, the popular methods for solving segmentation problems include the self-similarity matrix (SSM) and sparse subspace representation. These algorithms share some common ideas that find a better representation structure of data to obtain the subsequences in time series.

In this paper, a coarse-to-fine segmentation method is presented to compute the optimal segmentation

*Correspondence: seraph_mx@163.com

positions in video sequences to facilitate the segmentation of different facial expressions. The segmentation approach in the paper has two stages: rough segmentation and fine segmentation. Rough segmentation combines the SSM and spectral clustering (SC) to achieve the rough segmentation. The dynamic time alignment kernel (DTAK) is adopted in the second step to calculate the similarities between different segments. The rough segmentation obtained is then optimized to obtain optimal segmentation results. Using the MMI facial expression dataset, the method shows better segmentation performance compared with other segmentation methods. Additionally, this method is extended to detect nonneutral facial expressions in real-world videos.

This paper is organized as follows. Section 2 reviews previous work on temporal segmentation. Section 3 describes the framework of the segmentation algorithm. Section 4 demonstrates experimental results using the MMI facial expression database and nonneutral expression detection for real-world videos. Section 5 concludes the paper and outlines future work.

2. Related work

As early as 1971, Ekman and Friesen defined facial behavior using six basic expressions: happiness, sadness, anger, fear, surprise, and disgust. A few years later, the famous facial action coding system (FACS) was developed to detect the subtle variability in expressions [4]. Most of the existing algorithms are based on geometric and visual features. According to [5], shape and appearance features invariantly are clustered to some geometric transformations and the clusters are grouped into temporally coherent facial gestures based on the FACS. In [2], the authors extracted facial appearance features from selected facial patches and selected different facial patches for different expression.

Temporal video segmentation aims at splitting a video sequence into homogeneous subsequences to make the properties of each subsequence different. It can be considered as a clustering problem, which has been widely used in human behavior segmentation [6–12] and face clustering [5, 13–18]. One important condition expected of a clustering algorithm is that the intracluster variance should be low. That is, the key to a clustering problem is to obtain a good distance metric for representing the structure of interpersonal dissimilarities and intrapersonal similarities. Therefore, many methods, like SSM and sparse subspace representation, were proposed for solving segmentation problems.

The SSM, which is a variant of the recurrence plots (RPs) [12], is used widely to describe the texture features between sequences. Junejo et al. [8] explored self-similarities of action sequences over time to recognize human behaviors in different perspectives. Similarly, Kruger et al. [6] introduced a method for automated temporal segmentation of human motion data into distinct actions and compositing motion primitives based on self-similar structures in the motion sequence by analyzing SSMs. Xia et al. [10] built a novel depth cuboid similarity feature (DCSF) to describe the local 3D depth cuboid around the STIPs from depth videos (DSTIPs) with an adaptable supporting size.

Sparse subspace representation has received much attention since it was proposed in 2009. It achieves more accurate segmentation by subspace recovery. Sparse subspace clustering (SSC) and low rank representation (LRR) are pioneer works of sparse subspace representation. LRR [19] seeks the lowest rank representation that can represent the given data samples as linear combinations. Correspondingly, the SSC algorithm is used to discover the sparse representation of data, and the optimal representation is sufficiently sparse according to the method. The sparse representation of data can be efficiently computed by convex optimization. Consequently, many improvement methods for SSC and LRR have been proposed. Yin et al. [7] used a hypergraph Laplacian regularizer to improve LRR, and their method showed higher accuracies on face clustering. Cao et al. [13]

proposed a constrained multiview video face clustering (CMVFC) method based on the SSC model and used multiple features and angles to analyze face clustering. Cao et al. [20] researched reducing the influence of face poses, illumination, and noise on face detection and proposed a robust tensor clustering algorithm to reduce the noises in face clustering. Most of these works focused on the structure of space clustering and the robustness to various noises [15].

We propose a coarse-to-fine segmentation method, first presented in [21], to segment the facial expressions in video sequences. In this paper, we extend our previous work and propose two new contributions. First, we use frame similarity in rough segmentation to enhance similarity matrix S to make the rough segmentation results more reliable. Additionally, we compare the method with other face clustering and segmentation methods, and we try to apply it to the detection of nonneutral expressions in real-world videos.

3. The framework of our approach

In this section, we describe our method that segments the video sequence into different expressions. Figure 1 shows the process of the method. This process is mainly composed of three parts, including feature extraction, rough segmentation, and fine segmentation. For feature extraction, salient facial patches in videos are localized first, and LBP features of these patches are used as the inputs of rough segmentation. In the rough segmentation stage, the SSM is constructed on the basis of features, then combined with k-means and spectral clustering to

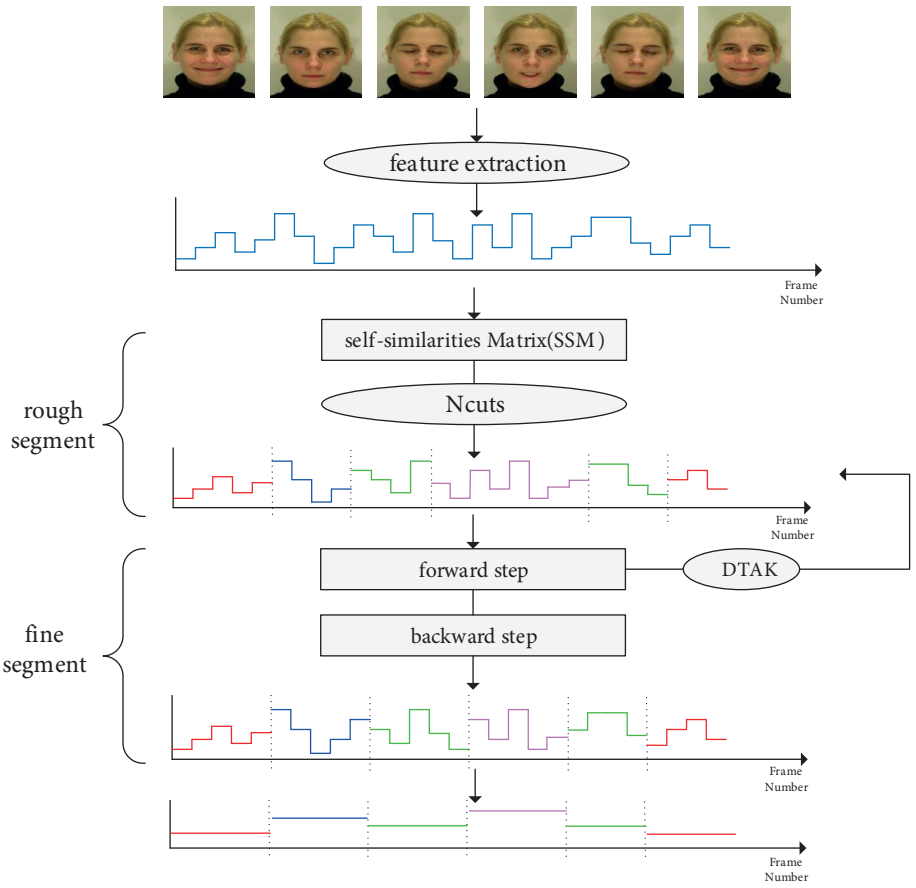

Figure 1. The main processing steps to segment facial expressions in video sequences.

segment the video sequence into distinct facial behaviors. The fine segmentation stage includes a forward step and backward step. The forward step finds the optimal segmentation position by DTAK algorithm, and the final segmentation result is obtained by the backward step.

3.1. Facial feature extraction

Feature extraction plays a vital role in facial image analysis. The changes of facial expressions will result in the expansion and contraction of facial muscles, and the key points in facial images will change too. In this paper, we aim at locating the key feature points on faces and extracting discriminative features around these points, which separate two expressions effectively. The feature extraction process has 5 main steps. Figures 2a–2e show the process to extract salient facial patches.

The first step is to localize face in images by a color-based algorithm [22]. Illumination compensation, with high computational efficiency and recognition accuracy, is used to improve the performance of face recognition in color images. Then the facial image needs to be transformed into a gray-scale image.

In the second step, the Haar cascade is used to localize eyes and nose. For a facial image, the pixel values of the eyes are generally lower than the surrounding pixels, while the pixel values at nose-tip are generally higher than the surrounding pixels. Therefore, it is easy to localize eyes and noses. The Haar cascade can achieve satisfactory results. The Haar classifier returns the rectangular region of the nose and eyes. The midpoint of the rectangular region is computed as the position of eyes and nose.

The third step is to localize the corners of the lips and eyebrows. The coarse region of interest (ROI) for lips and eyebrows is selected by the positions of the eyes and nose. Then the Sobel edge detector is used to detect the corners of lips and eyebrows in these regions.

The fourth step is to localize salient facial patches [2]. The number of salient facial patches is 19 and they are positioned below the eyes, between the eyebrows, and around the nose and mouth corners. These organs reflect more facial features as facial expressions change. The sizes of all facial patches are equal and each one of them is one-ninth of the width of face approximately. Their locations are shown in Figure 2e.

The fifth step is feature extraction. LBP is popular and is used widely for facial features. The LBP pattern is extracted from salient facial patches, which generates a binary number by comparing the center pixel value with 8 neighboring pixel values. Then histograms of each LBP image are computed as features of facial expression.

3.2. Segmentation into distinct facial behaviors

Facial image sequences are given as a collection of n facial expressions, each of which can be represented by a feature vector and correspond to one of k classes. There are two major challenges in framing temporal

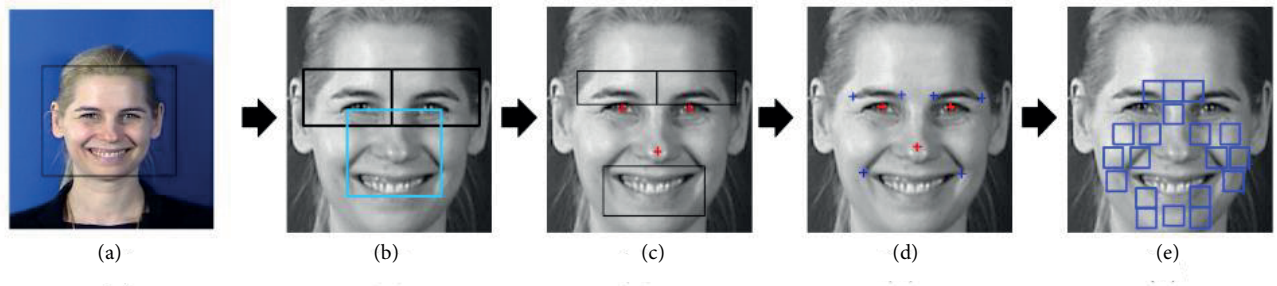

Figure 2. The progress to extract salient facial patches: (a) face detection, (b) coarse ROI selection for eyes and nose, (c) location of the lip and eyebrows, (d) detection of corners of lips and eyebrows, (e) finding salient facial patches.

segmentation as a clustering problem. One is modeling the temporal variability of facial behavior, and the other is defining a robust metric between facial expressions in video sequences. To address these problems, we combine kernel k-means and Ncut to achieve temporal clustering. Before performing clustering analysis, the similarity of the sequences is calculated by SSMs. The SSM is an effective tool for analyzing system behaviors, and it displays different textures for different time sequences. For a sequence of images $X = x_1, x_2, ..., x_n$, a SSM [12] of X is a square symmetric matrix of size $n \times n$m as indicated in Eq. (1):

$$D = [d_{ij}]_{i,j=1,2,...,n} = \begin{bmatrix} 0 & d_{12} & d_{13} & \cdots & d_{1n} \\ d_{21} & 0 & d_{23} & \cdots & d_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & d_{n3} & \cdots & 0 \end{bmatrix}, \tag{1}$$

where d_{ij} is the Euclidean distance of facial features that are extracted from frames i and j.

The patterns of this matrix and its structures depend on the features and the distance measure. d_{ij} is defined as the Euclidean distance between different facial features that we extract from video sequences. However, it is inaccurate to measure the similarity between facial expression sequences only by computing their Euclidean distance. In order to make the data more separable, the SSM is mapped to high-dimensional space by the kernel function. The kernel trick is a standard method for lifting the points of a dataset to a higher dimensional space, where points are more likely to be separable linearly (assuming that the correct mapping is found). In this paper, we use the Gaussian kernel to achieve self-similarity matrix S, $S = exp(-\frac{D^2}{2\sigma^2})$. D is the Euclidean distance matrix of facial features. Because of the high dimensionality of the expression data, this mapping is more helpful to cluster facial expression in video sequences. The clustering result can then be obtained using Eq. (3).

The degree of similarity between each frame in the video sequence can be achieved by constructing its self-similarity matrix. The diagonal of the SSM is composed of ones, because it shows no dissimilarity by comparing

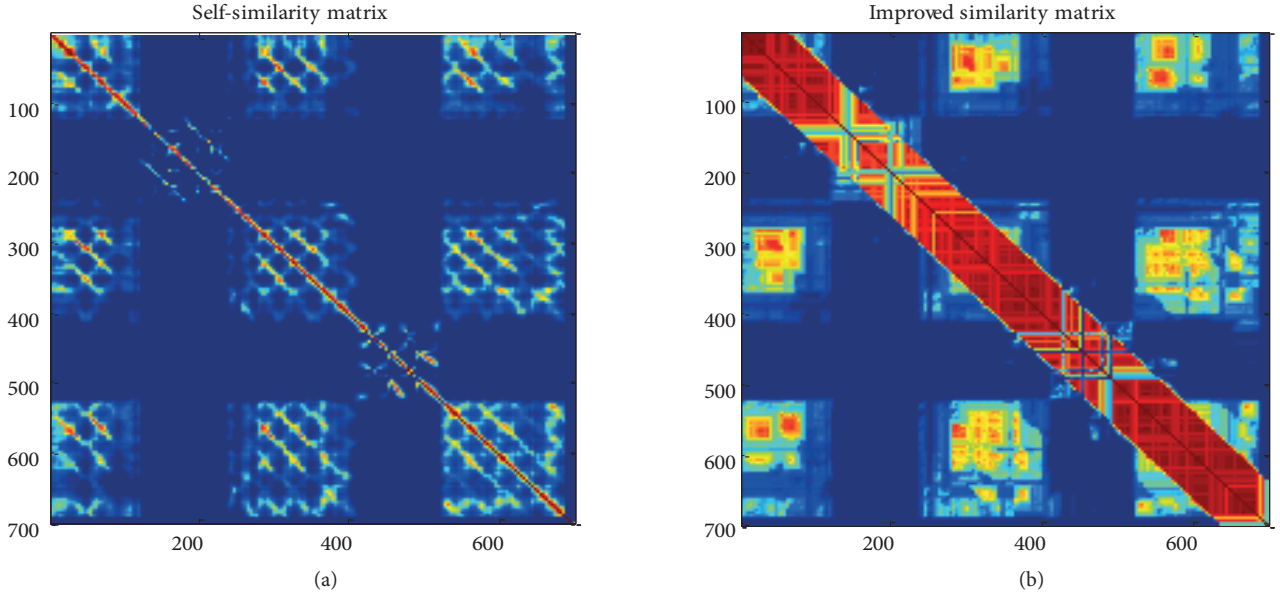

Figure 3. Examples of SSM for different expression sequences: (a) the self-similarity matrix of five different expressions and (b) the improved matrix.

a frame to itself. Different expressions in video sequences can be distinguished easily by the self-similarity matrix. For example, Figure 3a shows a SSM of a facial video, which contains five different expressions of the same man from the MMI database. The similarity values are linearly scaled; red regions show more similarity. The frames with the same expressions are more likely to be together. The diagonal direction of the figure is divided into 5 sections, which represent different segments.

To emphasize the frames that might belong to the same expressions, the SSM can be modified by propagating the similarity between two temporally close frames that share the same cluster. The improved SSM is defined as $\mathbf{S}'$, and it can be achieved by Algorithm 1.

Algorithm 1 Self-similarity matrix improvement

Input: self-similarity matrix S, sequence frame number n, length constraint n_{max}
Output: improved self-similarity matrix $\mathbf{S}'$

for i=1 to n **do**
  for j=i+1 to n **do**
    for w=i to n_{max} **do**
      x=i+w, y=j+w
// compare the similarity of adjacent frames
      temp= $min(s_{ij}, s_{xy})$
//compute the improved similarity matrix
      $s'_{ij} = max(s_{ij}, temp), s'_{xy} = max(s_{xy}, temp)$
      $s'_{iy} = max(s_{iy}, temp), s'_{xj} = max(s_{xj}, temp)$

Here, the value of n_{max} is set in advance. s_{ij} represents the similarity between the ith frame and jth frame. s_{ij}, s_{xy}, s_{iy}, and s_{xj} represent the adjacent frames, which make the similarity among them as consistent as possible.

Figure 3b is the improved SSM of Figure 3a, and it shows that the frames with similar expressions are linked together after the propagation. The improved matrix shares the same similarity within a certain length constraint. Close frames tend to show higher similarity, and they are more likely to be in the same class. The effect of spectral clustering can be determined by the quality of the SSM, and the cluster can be achieved by the area around the diagonal of the matrix from Figure 3b.

Next, the SC and SSM are combined for rough segmentation. Spectral clustering evolves from the concepts of graph theory and has been used widely in clustering. It constructs a weighted graph, $\mathbf{A} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$, where $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2, ...\mathbf{V}_n]$ represents the point set, $\mathbf{E}$ represents the set of edges, and w_{ij} represents the weight between the ith point and jth point. The goal of SC is to find a partition of the graph that minimizes a particular cost function. A popular cost function is given in Eq. (2):

$$cut(\mathbf{A}) = \sum_{v_i \in R, v_j \in Q-R} w_{ij}, \tag{2}$$

where v_i denotes the ith node of Graph A, Q represents all nodes, and R is a subset of the nodes. The work in [23] shows the relation of kernel k-means and Ncuts. This relation can be described as in Eq. (3):

$$E(\mathbf{M}, \mathbf{G}) = \left\| (\phi(\mathbf{B}) - \mathbf{M}\mathbf{G}^T)\mathbf{W}_c \right\|_F^2, \tag{3}$$

where $\mathbf{G} \in R^{n \times c}$ and $\mathbf{M} \in R^{d \times c}$. $\mathbf{G}$ is a binary indicator matrix and subject to $\mathbf{G}l_c = l_n$, and $\sum_j g_{ij} = 1$.

c denotes the number of classes and n is the number of samples. The columns of $\mathbf{B} \in R^{d\times n}$ contain the original data points, and the columns of $\mathbf{M}$ represent the cluster centroids. d is the dimension of the data. $\phi(\mathbf{B})$ represents the mapping relation of $\mathbf{B}$ to higher dimensional space. $\mathbf{W}_c = diag(\Gamma^T \Gamma l_n)^{-1/2}$ represents the weight between the data after mapping. The clustering results can be achieved by minimizing Eq. (3), which is the same as maximizing Eq. (4):

$$\mathbf{G}, s = E(\mathbf{Z}) \propto tr((\mathbf{Z}\mathbf{Z}^T)^{-1}\mathbf{Z}\mathbf{W}_c\mathbf{S}'\mathbf{W}_c\mathbf{Z}^T), \tag{4}$$

where $\mathbf{S}'$ is the improved self-similarity matrix. $\mathbf{W}_c = diag(\mathbf{S}'l_n)^{-1/2}$, $\mathbf{Z} = \mathbf{C}^T\mathbf{W}_c$, $\propto$ refers to "proportional to", $\mathbf{G} \in \{0,1\}^{n\times c}$ is the segment cluster indicator matrix and stores the class of each frame in the video sequence, and s contains the beginning and end of each segmentation. In the rough segmentation, video sequences are segmented into distinct facial behaviors.

3.3. Subdividing distinct facial behavior into different expressions

Fine segmentation is mainly discussed in this section. This step aims at optimizing the rough segmentation result and determining a more suitable segmentation position. The main process is creating a new small segmentation X and calculating its similarity with the rough segmentation. Next, the rough segmentation result is updated by the class and starting position of this small segmentation. Therefore, a metric is needed to describe the similarities between segments. DTAK was originally proposed to modify the expression of a SVM kernel, introducing the DTW distance between two sequences in the feature space [24]. It can be applied to compute the distance between segments with different lengths. In this paper, DTAK is used to measure the similarity between segmentations.

Assume there are two sequences, $X = [x_1, ...x_{n_x}]$ and $Y = [y_1, ...y_{n_y}]$. In the case that the lengths of two sequences are equal, $n_x = n_y$, the inner product of X and Y, can be obtained directly. However, once their lengths are not same, the inner product cannot be calculated directly. In DTAK [25], the computation of the inner product can be solved effectively by dynamic programming. The recursive formula in dynamic programming is given in Eq. (5):

$$U_{ij} = max \begin{cases} U_{i-1,j} + k_{ij} \\ U_{i-1,j-1} + 2k_{ij}, \tau(X,Y) = \dfrac{U_{n_x n_y}}{n_x + n_y} \\ U_{i,j-1} + k_{ij} \end{cases}. \tag{5}$$

Here, U is initialized at the upper left, i.e. $u_{11} = 2k_{11}$. $\mathbf{k}$ is the kernel matrix. Its definition is $k_{ij} = exp(-\frac{\|x_i - y_j\|^2}{2\sigma^2})$. DTW can find the optimal distance to minimize the cumulative matrix. Unlike DTW, DTAK finds the optimal path by maximizing cumulative similarity. It is noteworthy that DTAK is a semipositive definite kernel under certain conditions.

For instance, we can consider two short sequences, X=[1,2,1,2] and Y=[1,1,2,2]. Construction of matrix $\mathbf{U}$ starts from the upper-left corner and fills along the rows and columns gradually. Figure 4a shows the procedure to build matrix $\mathbf{U}$. Figure 4b illustrates the binary matrix k where the value of $\sigma \to 0$. The final value of DTAK, $\gamma(X,Y) = 7/8$, is obtained by dividing the bottom right by the sum of the sequence lengths.

In addition to improving the segmentation performance further, dynamic programming combined with DTAK can optimize the rough segmentation by calculating the similarity between segments. For a sequence X of length n, the number of all possible fine segmentations is exponential, and the fine segmentation stage has

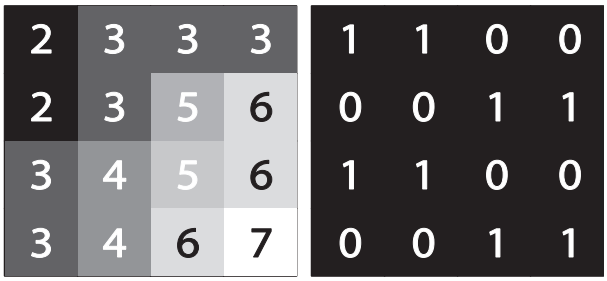

Figure 4. Computation of DTAK: (a) example to calculate U with DTAK and (b) kernel matrix K.

high calculation complexity. Thus, we use a DP-based algorithm to search the optimal fine segmentation results in polynomial time. The rough segmentation is optimized at each iteration using Eq. (6):

$$\mathbf{G}^{new}, s^{new} = \underset{\mathbf{G},s}{argmin}\, J(\mathbf{G}, s) = \underset{\mathbf{G},s}{argmin} \sum_{c=1}^{k} \sum_{j=1}^{m} g_{ci} dist^2(X_{[i,v]}, z_c), \tag{6}$$

where $X_{[i,v]}$ represents a segmentation. g_{ci} is a class indicator, and when $X_{[i,v]}$ belong to class c, g_{ci} is 1. m represents the number of segmentations. $\mathbf{G}^{new} \in \mathbf{R}^{m \times k}$ is the indicator matrix of fine segmentation. $\mathbf{s}^{new}$ contains the beginning and end of each fine segmentation. z_c represents the center of class c and $dist^2(X_{[i,v]}, z_c)$ is the distance computation between $X_{[i,v]}$ and z_c, as demonstrated in Eq. (7):

$$dist^2(X_{[i,v]}, z_c) = \tau(X_{[i,v]}, X_{[i,v]}) - \frac{2}{n_c} \sum_{j=1}^{m} \tau(X_{[i,v]}, Y_j) + \frac{1}{n_c^2} \sum_{j1,j2=1}^{m} \tau(Y_{j1}, Y_{j2}), \tag{7}$$

where τ represents the value of the DTAK between segments. Y_i is the rough segmentation results. $n_c = \sum_{j=1}^{n} g_{cj}$ is the number of rough segmentations whose class is c, and $\sum_{j1,j2=1}^{m} \tau(Y_{j1}, Y_{j2})$ represents the sum of DTAK values between each group of rough segmentation. To obtain $J(\mathbf{G}, s)$ that satisfies the principle of optimality, the relationship needs to be leveraged further between $\mathbf{G}$ and s:

$$J(v) = \underset{v-n_{max}<i\leq v}{min} (J(i-1) + \underset{\mathbf{G},s}{min} \sum_{c=1}^{k} \sum_{j=1}^{m} g_{ci} dist^2(X_{[i,v]}, z_c)), \tag{8}$$

where n_{max} is the length constraint. Eq. (8) implies that only when the segmentations on both sides of $X_{[v-n_{max},i-1]}$ and $X_{[i,v]}$ are optimal and their sum is minimal will the subsequence get the optimal decomposition. Fine segmentation (followed by rough segmentation) is obtained by forward and backward steps.

In the forward step, we create a new segmentation $X_{[i,v]}$ from the beginning (v=1) to the end (v=n) of the video sequence, where i represents the head position of the new segmentation. Then, for each $i \in [v - n_{max}, v]$, the purpose of optimization is to find the minimal J by Eq. (8). Those processes are repeated until J converges. In the last iteration, the head position i and label g for the segment $X_{[i,v]}$ are stored.

During the backward step, the fine segmentation is obtained according to the head position i and label g. The algorithm begins from the end of the sequence and cuts off the segment whose head position is i. New indicator matrix $\mathbf{G}$ is calculated by label g. This operation is repeated until new $\mathbf{G}$ and s are obtained.

4. Experiments and discussion

4.1. Experiments on the MMI face database

To verify the effectiveness of our algorithm, the MMI face database is used to carry out the experiments [26]. The MMI face database consists of over 2900 videos and high-resolution still images of 75 subjects. The static facial-expression images are all true color (24-bit) images, which are the size of 720×576 pixels. Video sequences of faces in frontal and in profile views display various facial expressions of emotion. All video sequences have been recorded at a rate of 24 frames/s using a standard PAL camera. The database is freely available from the scientific community [27]. Figure 5 shows examples of MMI face database images; the first row is static images and the second row is frames of video sequences.

Figure 5. Examples of MMI face database images. First row: Static frontal view images. Second row: Apex frames of dual view video sequences.

4.1.1. Experimental results and analysis

In the experiment, we select 10 subjects, including 5 males and 5 females. For each subject, 4 different expressions are combined into a long video sequence, and three of these expressions appear two times. The 10 sequences are used to test whether the algorithm is effective. Each sequence combines 4 natural expressions roughly, such as anger, happiness, surprise, and sadness, and contains hundreds of frames that range from 400 to 800. n_{max} is 60, σ is set to be the average distance from the 2 percent closest neighbors, and the expression number is chosen according to the actual sample number in the experiment. By concatenating the video into a long sequence, ground-truth labels can be generated. Accuracy is used to measure the effect of segmentation in this paper. The accuracy is calculated based on a confusion matrix, which is derived from the match between the clustering results obtained from the algorithms (SSC, LRR, LRLRR, rough segmentation, and fine segmentation) and the ground-truth labels. This evaluation metric is widely employed in the current clustering methods.

For video sequences, LBP histogram features are extracted from salient facial patches. The proposed method is compared with some other clustering methods, including SSC [14], LRR [19], and LRLRR [7]. We choose default parameters in the contrast methods, and the parameter value of the k nearest neighbor is 15 in

LRLRR. Figure 6 shows the segmentation results. Different expressions are marked with different colors; that is, each square represents one expression. The first row is manual labeling (ground truth), the second row is the result of SSC, the third row is the result of LRR, the fourth row is the result of LRLRR, the fifth row is the result of the rough segmentation, and the sixth row is the result of the fine segmentation.

For the construction of sequences a–j, we try to choose similar expressions from different people in the MMI database. Table 1 lists the segmenting accuracy of SSC, LRR, LRLRR, rough segmentation, and fine segmentation, and the average segmentation results are 56.77%, 61.11%, 69.78%, 75.96%, and 85.09%, respectively. As can be seen in Figure 6, the facial expression clustering results of LRR and SSC are similar, while LRLRR performs better than LRR and SSC. The proposed method can achieve better clustering results. The advantage of our method lies in the analysis of the difference between the expression behaviors through the SSM, and the segmentation results are optimized further by the DTAK algorithm. However, LRR and SSC use the sparse representation coefficients of high-dimensional data to construct the similarity matrix, which has good performance on high noise data, while it shows poor clustering results for different expressions of the same person.

Table 1. Segmentation results of different samples on MMI data.

Sequence number	Length	SSC	LRR	LRLRR	Rough seg.	Fine seg.
a	659	70.1%	74.3%	62.2%	83.1%	93.6%
b	721	55.3%	65.4%	66.7%	70.8%	77.2%
c	572	46.1%	46.1%	75.1%	56.0%	87.6%
d	717	72.5%	71.5%	71.9%	72.8%	75.0%
e	504	55.7%	43.7%	43.7%	83.9%	91.7%
f	619	46.0%	46.0%	66.0%	75.9%	83.9%
g	771	55.1%	55.1%	67.9%	77.5%	88.1%
h	777	68.4%	68.9%	92.0%	81.4%	82.2%
i	741	54.1%	69.2%	63.6%	77.2%	84.5%
j	622	44.4%	70.9%	88.7%	81.0%	87.1%

The key to temporal segmentation is that the metric between the same expressions should be low, and that between different expressions should be large. The advantages of our method are reflected in the following aspects. First, the fine segmentation can make the further readjustment of the subsequence class, which is misjudged by rough segmentation. This is reflected in (a), (c), (e), (f), (g), and (j) in Figure 6. Meanwhile, most experimental results show that the fine segmentation can adjust the boundaries of original rough segmentation. However, the method also has some shortcomings, which can be seen in (b), (d), (h), and (i) in Figure 6, as part of the subsequence is identified into other classes mistakenly. A possible reason for the segmentation failure is that some expressions, like anger and happiness, show relatively high similarity in the local region. In general, expression subsequences in video sequences can be segmented correctly.

4.2. Nonneutral expression detection

In this section, we try to use our method to detect the nonneutral expressions in video sequences that we shot. In our experiments, when a person's mouth and eye expressions do not change, the expression is defined as a

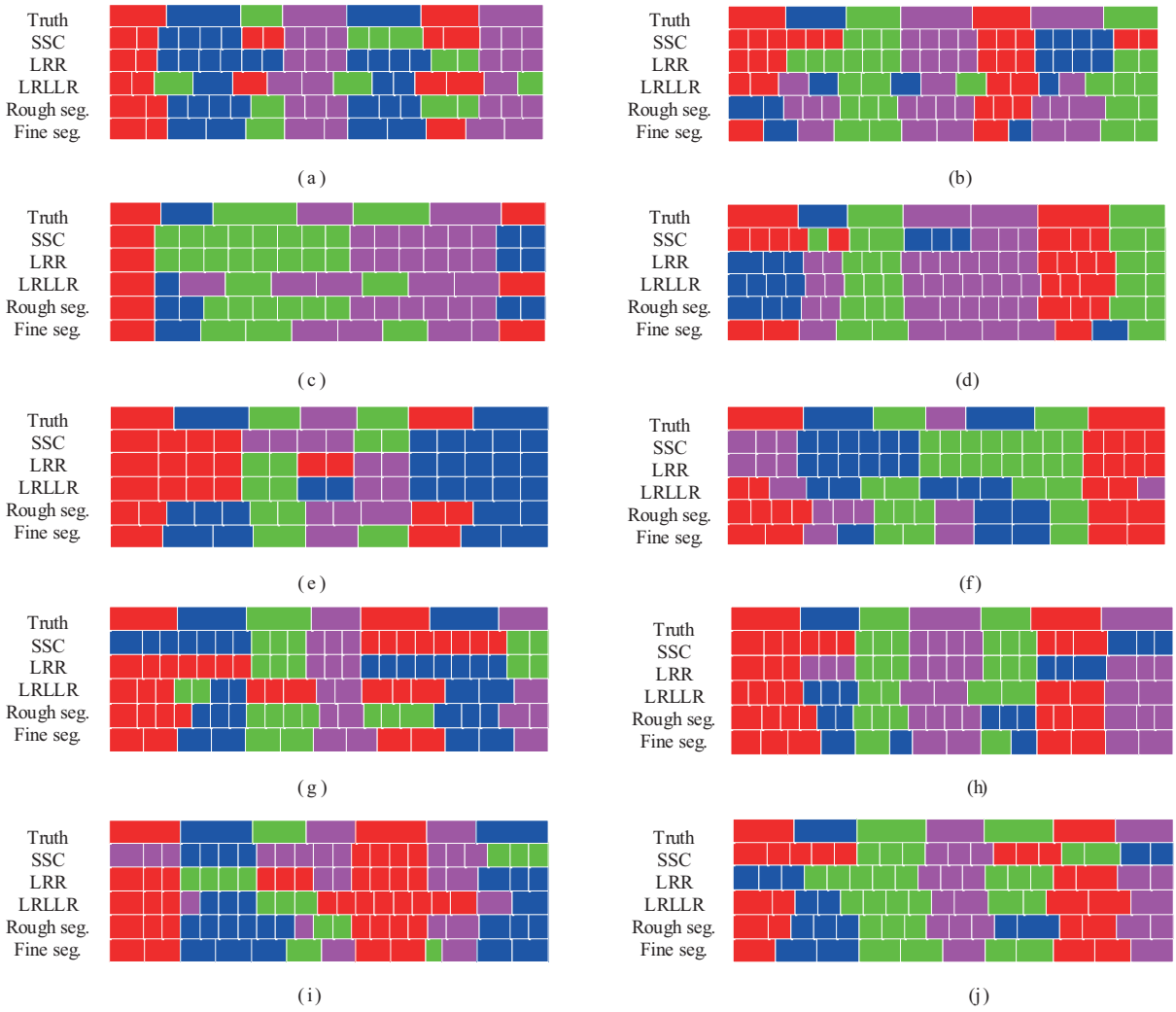

Figure 6. Segmentation results of different temporal clustering methods on MMI data. First row: Manual labeling (ground truth). Second row: SSC. Third row: LRR. Fourth row: LRLRR. Fifth row: Rough segmentation. Sixth row: Fine segmentation. a–j represent the accuracy of different expression sequences, respectively.

neutral expression, and conversely, it is defined as a nonneutral expression. In other words, expressionlessness is deemed to be neutral expression, and otherwise the opposite. To further illustrate the segmentation performance of our method, real videos that contain a different number of expressions are detected in the experiment. Four volunteers were invited to shoot four groups of videos. Each group consists of four short videos, which are about 10 s, 30 s, 40 s, and 60 s in length, corresponding to one, two, three, and four nonneutral expressions. In order to ensure the effectiveness of the experiment, the angle, lighting, and background of shooting for each person were consistent. Nonneutral expressions were roughly divided into happiness, surprise, anger, and sadness in the experiment.

We repeated the same experimental steps in Section 4.1.1 to segment the video sequence. The accuracy of rough segmentation and fine segmentation is demonstrated in Table 2 and Table 3. Figures 7a–7p show the algorithm's segmentation effect on videos with different numbers of expressions. Red indicates a neutral expression, and other colors represent nonneutral expressions like surprise, sadness, and so on. The first rows

Figure 7. Segmentation results of different numbers of expressions: (a–d), (e–h), (i–l), and (m–p) represent different numbers of expressions of the different people, respectively.

represent the manual label, the second rows represent the result of rough segmentation, and the third rows represent the accuracy of fine segmentation. When the number of nonneutral expressions in the video is relatively small, such as only one or two, our method shows satisfactory results. However, with the increase of nonneutral expression number in a video, the performance of segmentation decreases. This phenomenon is caused because it is more difficult to find a suitable metric to distinguish these expressions when the number of nonneutral expressions is higher. According to the accuracy rates of segmentation shown in Table 2 and Table 3, the segmentation accuracy decreases with the increase of expression number in the video sequence.

A facial expression unit starts from a neutral expression, reaches the peak, and finally returns to the neutral expression. Therefore, it is hard to determine the boundary of expression in real videos, which leads to the fact that the range of expression is inaccurate in fine segmentation in Figure 7. Besides, the neutral expression between two expressions is easy to be ignored when the interval between them is too short. For instance, in Figure 7h and Figure 7k, the interval between the first nonneutral expression and the second nonneutral expression is shorter than others, and the neutral expression between them is misclassified. On the whole, most expressions can be divided efficiently. The algorithm can detect the nonneutral expressions that appear in the video sequence and can roughly determine where they are in the video sequence.

Table 2. Temporal segmentation results of different samples (rough segmentation).

Expression number	Accuracy of rough segmentation				Average
	1	2	3	4	
1	93.8%	59.3%	62.3%	92.9%	77.1%
2	55.7%	78.3%	55.8%	56.7%	61.6%
3	55.4%	66.3%	62.3%	69.9%	63.5%
4	54.9%	42.8%	57.3%	68.6%	55.9%

Table 3. Temporal segmentation results of different samples (fine segmentation).

Expression number	Accuracy of fine segmentation				Average
	1	2	3	4	
1	95.0%	93.3%	83.9%	94.2%	91.6%
2	88.2%	86.3%	86.0%	90.1%	87.7%
3	69.3%	85.4%	82.4%	84.5%	80.4%
4	74.8%	75.8%	74.4%	74.7%	74.9%

4.3. Complexity analysis and discussion

Good segmentation tends to have large within-class connectivity and low between-class connectivity. For different expressions in the video sequence, our method can distinguish them correctly and has relatively good segmentation results. Given a facial sequence with n frames, we need $O(n^2 n_{max})$ to compute the improved SSM. Thus, the time complexity of the rough segmentation stage is $O(n^2 n_{max})$, where n_{max} is the length constraint. In the forward step in fine segmentation, the algorithm takes $O(nn_{max})$ to search the optimal segmentation position. The time complexity of the fine segmentation stage is $O(nn_{max})$. The overall time complexity is $O(n^2 n_{max})$. In general, the computational cost of our method is relatively high, especially when the video sequences are long. The accuracy of segmentation decreases as the length of the time series increases. When the facial video sequence is too long or its frame rate is too high, we can reduce the length of the sequence by clustering and merging adjacent frames of the same class in pretreatment. It is observed that the time cost is feasible in many applications, where temporal segmentation can be performed offline.

n_{max} controls the range of the improved similarity matrix and the search range of the optimal position in the fine segmentation stage. Its value not only influences the complexity of calculation, but also affects the accuracy of segmentation. Video samples of (a), (c), (e), (f), (g), and (h) in Section 4.1.1 are chosen for the experiment by changing their nmax values and observing the change of accuracy. The experimental results are shown below.

In Figure 8, the accuracy is higher when the n_{max} value is around 60. If the value is set too high or low, the accuracy is not desirable. When the value is too high, the new segmentation X created in the fine segmentation stage may contain two classes, which results in the inaccurate calculation of the similarity between X and rough segmentation. When the value is too low, the performance of the improved SSM is poor, and the search range of the algorithm is small. This may result in the final segmentation results not being optimal. Therefore, n_{max} is set as 60 in experiments.

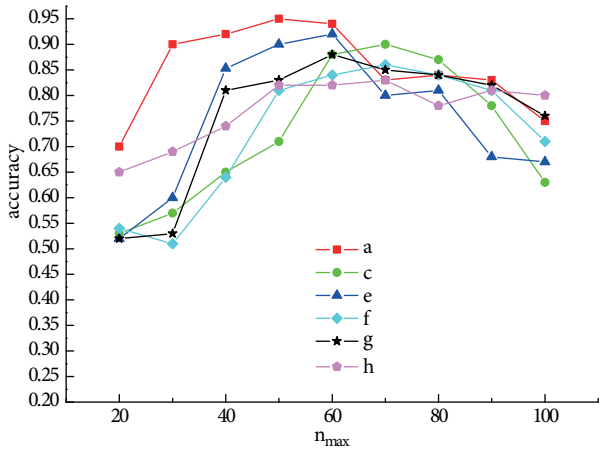

Figure 8. Fine segmentation results with different n_{max} values.

5. Conclusion and future work

In this paper, we propose an effective method for temporal segmentation of facial expressions in video sequences. In accordance with the feature of facial feature points in video sequences, the SSM is calculated. Then two-layer segmentation of facial video sequences is realized by combining SC and the DTAK algorithm. The efficient segmentation framework exhibits high accuracy in the relevant experiments of facial expression video sequences. Meanwhile, the experimental result indicates that some expressions are easy to be segmented, such as happiness and surprise. Some expressions like sadness and anger still have deviations, and these deviations are not only because of the sample quality we collect, but also the method of expression feature extraction and the high similarity between some different facial expressions. The same expression may be divided into different classes according to the experimental results, and this situation can be addressed in the face recognition step by defining the subsequence as input and obtaining the final expression type. The method can find a better measure to distinguish the face differences and can judge the locations of abnormal expressions in the expression sequences.

In the future, we will focus on improving our method as follows. First, the clustering algorithm splits n data points into k groups. Thus, it depends on the knowledge of the number of clusters or number of the expressions present in a video clip. In practical situations, the actual expression number may not be known beforehand. The next important work is to obtain the expression number that exists in a video automatically. Secondly, the calculation of the similarity between two frames leads to higher computational complexity and lower accuracy of segmentation if the video sequences are too long. At the same time, some misclassifications occur in video sequences with high similarity. In order to increase the robustness of this segmentation, our next task is to improve the efficiency and accuracy of segmentation.

Acknowledgment

The research was supported by the open-ended fund of the Advanced Innovation Center for Intelligent Robots and Systems (2018IRS20) and the Research and Innovation Project for College Graduates of Jiangsu Province (KYCX17_0925).

References

[1] Cai X, Nie F, Huan, H, Kamangar F. Heterogeneous image feature integration via multi-modal spectral clustering. In: IEEE 2011 International Conference on Computer Vision and Pattern Recognition; Colorado Springs, CO, USA; 2011. pp. 1977-1984.

[2] Happy SL, Routray A. Automatic facial expression recognition using features of salient facial patches. IEEE Transactions on Affective Computing 2015; 6 (1): 1-12. doi: 10.1109/TAFFC.2014.2386334

[3] Lan Z, Sourina O, Wang L, Liu Y. Real-time EEG-based emotion monitoring using stable features. Visual Computer 2015; 32 (3): 347-358. doi: 10.1007/s00371-015-1183-y

[4] Ekman P, Friesen W. Facial Action Coding System: A Technique for the Measurement of Facial Movement. Palo Alto, CA, USA: Consulting Psychologists Press, 1978.

[5] Fernando DLT, Campoy J, Ambadar Z, Coln JF. Temporal segmentation of facial behavior. In: IEEE 2007 International Conference on Computer Vision; Rio de Janeiro, Brazil; 2007. pp. 1-8.

[6] Kruger B, Vogele A, Willig T, Yao A, Klein R et al. Efficient unsupervised temporal segmentation of motion data. IEEE Transactions on Multimedia 2017; 19 (4): 797-812. doi: 10.1104/pp.79.3.699

[7] Yin M, Gao J, Lin Z. Laplacian regularized low-rank representation and its applications. IEEE Transactions on Pattern Analysis & Machine Intelligence 2016; 38 (3): 504-517. doi: 10.1109/TPAMI.2015.2462360

[8] Junejo IN, Dexter E, Laptev I, Pérez P. View-independent action recognition from temporal self-similarities. IEEE Transactions on Pattern Analysis & Machine Intelligence 2011; 33 (1): 172-185. doi: 10.1109/TPAMI.2010.68

[9] Lu G, Kudo M, Toyama J. Temporal segmentation and assignment of successive actions in a long-term video. Pattern Recognition Letters 2013; 34 (15): 1936-1944. doi: 10.1016/j.patrec.2012.10.023

[10] Xia L, Aggarwal JK. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In: IEEE 2013 Conference on Computer Vision and Pattern Recognition; Portland, OR, USA; 2013. pp. 2834-2841.

[11] Zhou F, Torre FDL, Hodgins JK. Hierarchical aligned cluster analysis for temporal clustering of human motion. IEEE Transactions on Pattern Analysis & Machine Intelligence 2016; 35 (3): 582-596. doi: 10.1109/tpami.2012.137

[12] Manfred B, Jacob AF, Lorenz-Peter S. Self-similarity matrix based slow-time feature extraction for human target in high-resolution radar. International Journal of Microwave & Wireless Technologies 2014; 6 (3-4): 423-434. doi: 10.1017/S1759078714000087

[13] Cao X, Zhang C, Zhou C, Fu H, Foroosh H. Constrained multi-view video face clustering. IEEE Transactions on Image Processing 2015; 24 (11): 4381-4393. doi: 10.1109/TIP.2015.2463223

[14] Elhamifar E, Vidal R. Sparse subspace clustering. In: IEEE 2009 Conference on Computer Vision and Pattern Recognition; Miami, FL, USA; 2009. pp. 20-25.

[15] Feng J, Lin Z, Xu H, Yan S. Robust subspace segmentation with block-diagonal prior. In: IEEE 2014 Conference on Computer Vision and Pattern Recognition; Columbus, OH, USA; 2014. pp. 3818-3825.

[16] Wright J, Ganesh A, Zhou Z, Wagner A, Ma Y. Demo: Robust face recognition via sparse representation. In: IEEE 2008 International Conference on Automatic Face & Gesture Recognition; Amsterdam, the Netherlands; 2008. pp. 1-2.

[17] Chrysouli C, Vretos N, Pitas I. Face clustering in videos based on spectral clustering techniques. In: IEEE 2008 Asian Conference on Pattern Recognition; Beijing, China; 2011. pp. 130-134.

[18] Wu B, Zhang Y, Hu BG, Ji Q. Constrained clustering and its application to face clustering in videos. In IEEE 2013 Conference on Computer Vision and Pattern Recognition; Portland, OR, USA; 2013. pp. 3507-3514.

[19] Liu G, Lin Z, Yan S, Sun J, Yu Y et al. Robust recovery of subspace structures by low-rank representation. IEEE Transactions on Pattern Analysis & Machine Intelligence 2013; 35(1): 171-184. doi: 10.1109/TPAMI.2012.88

[20] Cao X, Wei X, Han Y, Lin D. Robust face clustering via tensor decomposition. IEEE Transactions on Cybernetics 2015; 45 (11): 2546-2557. doi: 10.1109/TCYB.2014.2376938

[21] Xue Y, Mei X, Bian JL, Wu L, Ding Y. Temporal segmentation of facial expressions in video sequences. In: IEEE 2017 Chinese Control Conference; Dalian, China; 2017. pp. 10789-10794.

[22] Pai YT, Ruan SJ, Shie MC, Liu YC. A simple and accurate color face detection algorithm in complex background. In: IEEE 2006 International Conference on Multimedia and Expo; Toronto, Canada; 2006. pp. 1545-1548.

[23] Torre FDL. A least-squares framework for component analysis. IEEE Transactions on Pattern Analysis & Machine Intelligence 2012; 34 (6): 1041-1055. doi: 10.1109/tpami.2011.184

[24] Principi E, Squartini S, Piazza F. Power normalized cepstral coefficients based supervectors and i-vectors for small vocabulary speech recognition. In: IEEE 2014 International Joint Conference on Neural Networks; Beijing, China; 2014. pp. 3562-3568.

[25] Stathopoulos V, Zamora-Gutierrez V, Jones K, Girolami M. Bat call identification with Gaussian process multinomial profit regression and a dynamic time warping kernel. In 2014 International Conference on Artificial Intelligence and Statistics; Reykjavik, Iceland; 2014. pp. 913-921.

[26] Fang H, Parthaláin NM, Aubrey AJ, Tam GKL, Borgo R et al. Facial expression recognition in dynamic sequences: an integrated approach. Pattern Recognition 2014; 47 (3): 1271-1281. doi: 10.1016/j.patcog.2013.09.023

[27] Valstar M, Pantic M. Induced disgust, happiness and surprise: An addition to the MMI facial expression database. In: Proceedings of the International Workshop on Emotion Corpora for Research on Emotion & Affect; 2010. pp. 1-6.