Expression Recognition from 3D Dynamic Faces using Robust Spatio-temporal Shape Features

Vuong Le, Hao Tang and Thomas S. Huang
Beckman Institute for Advanced Science and Technology
Department of Electrical and Computer Engineering, University of Illinois at Urbana - Champaign
405 North Mathews Avenue, Urbana, IL 61801, USA

***Abstract*—This paper proposes a new method for comparing 3D facial shapes using facial level curves. The pair- and segment-wise distances between the level curves comprise the spatio-temporal features for expression recognition from 3D dynamic faces. The paper further introduces universal background modeling and maximum a posteriori adaptation for hidden Markov models, leading to a decision boundary focus classification algorithm. Both techniques, when combined, yield a high overall recognition accuracy of 92.22% on the BU-4DFE database in our preliminary experiments. Noticeably, our feature extraction method is very efficient, requiring simple preprocessing, and robust to variations of the input data quality.**

I. Introduction

Expression recognition from 3D facial data is a new and interesting problem. Several baseline methods have been introduced and gave very promising results [1] and [2]. In another direction, research has also been done on studying the dynamics of facial expressions in 2D images [3], proving that looking at sequences of face instances can help improve the recognition performance. These initiatives give us the awareness that facial expressions are highly dynamical processes in the 3D space, and therefore observing the state transitions of 3D faces could be a crucial clue to the investigation of the inner state of human subjects.

Recently, this direction is getting more attention with the introduction of appropriate databases such as the BU-4DFE database developed at Binghamton University [4]. It is also inspired by the revolution of inexpensive acquisition devices such as the consumer 3D cameras. One of the challenges for expression recognition using data from these low-end devices is that most of them produce noisy and low resolution depth images. In such kind of condition, expression recognition using traditional methods based on tracked facial feature points can be sensitive to the noise. Moreover, tracking facial feature points by itself would require more computations and thus make the systems harder to satisfy the real-time requirement. This situation shows the urgent need for a robust and efficient feature representation for 3D facial shapes together with a high performance classification algorithm to exploit the features for expression recognition.

In this work, we propose to use a facial level curves based representation of 3D faces for expression recognition. A similar representation was applied to face recognition by Samir and colleagues [5]. To the best of our knowledge, it has not been used for expression recognition. Besides being powerful and robust, one advantage of this representation is that the level curves can be extracted directly from depth images with some simple preprocessing steps.

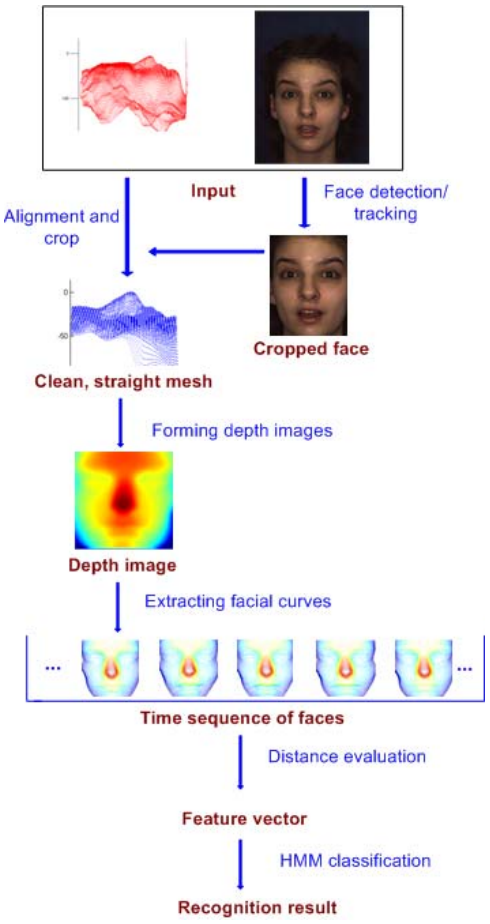

Fig. 1: Framework of expression recognition from 3D dynamic faces using facial level curves

On top of the representation, we introduce a novel method to measure the distances between the corresponding level curves of two 3D facial shapes, which are then used as spatio-temporal features for expression recognition. The method is based on the Chamfer distances of normalized segments partitioned from the level curves by an arclength parameterized function. This feature extraction method does not require feature point localization, and can deal with low resolution

depth images. These characteristics make the method very friendly with low-end acquisition devices and the requirement of fast processing.

We further introduce universal background modeling and maximum a posteriori adaptation for hidden Markov models (HMMs), leading to an HMM-based decision boundary focus classification algorithm. Combined with the proposed feature extraction method, this classification algorithm yields a high overall recognition accuracy of 92.22% on the BU-4DFE database in our preliminary experiments. The overall pipeline of our expression recognition framework is depicted in Fig. 1.

This paper is organized as follows. Section II reviews the related work. The BU-4DFE database used in our experiments is introduced in Section III. Section IV describes the details of the proposed feature extraction method, and Section V the details of the proposed HMM-based decision boundary focus classification algorithm. In Section VI, the experiments and results are presented. Finally, Section VII draws the conclusion and discusses the future work.

II. Related Work

Shape representation and feature extraction for the analysis of 3D facial data have gained increased attention recently. In this section, we review some existing approaches that use 3D face models for facial expression recognition and some shape representation methods related to ours.

In an intuitive way of analyzing facial expressions, several works, such as [3], [6] and [7], follow the traditional approach of using 3D face models to estimate the movements of the facial feature points. These features are related to the action units (AUs) and their movements control the emotional states of the subject.

In a different way, a dense correspondence frame is built based on a set of predefined landmarks. The arrangement of the corresponded dense point set is used as the feature for classification. Mpiperis and colleagues, in [1], use a correspondence frame built by a subdivision surface model using a set of predefined feature points. The members of the 3D correspondence frame are then used as raw features in a bilinear model which helps separate the identity and expression factors in 3D static data. A tensor-like model is built as PCA subspaces, both among people and across the expressions. With such a relatively simple model, they can afford to utilize the dense features of thousands of dimensions and give impressive recognition results.

A dense correspondence can provide very informative features of the facial shape but will face the curse of dimensionality. Extracting the geometrical shape features at some important locations such as the convex parts, high curvatures areas, and saddle points may reduce the vector size while still keeping the representation robust. One example is the primitive label distribution used in a recent work by Yin et al. [2]. Whilst the feature definitions of the primitive label distribution are intuitively meaningful and shown to work well on studio data, the computation of curvatures in general involves numerical approximation of second derivatives and may be susceptible to observation noise.

Up to the present, most of the 3D face expression recognition works are based on static data. In a pioneer work that exploits dynamic data [8], Sun and Yin extract sophisticated features of geometric labeling and use 2D HMMs as classifiers for recognizing expressions from 3D face model sequences. Their method has led to very promising recognition performance.

In the aspect of 3D facial shape representations, Samir et al., in [9], develop an intrinsic mathematical and statistical framework for analyzing facial shapes for face recognition based on facial level curves. In that framework, the shape representation is similar to ours. However, the distance function used to extract and compare the level curves are significantly different. In Samir et al.'s work, they use the geodesic distances based on an angle function, which are shown to be relatively invariant to expressions [10] and therefore are more suitable for face recognition. Instead, in our work, we propose the localized Chamfer distances which are correlated with expression changes.

III. Database Description

In our experiments, we utilize the BU-4DFE database [4], which captures the dynamics of 3D expressive facial surfaces over a time period. The BU-4DFE database was created at the State University of New York at Binghamton by Yin et al. This database consists of 101 subjects, including 58 females and 43 males of different ethnicity: Asian, Black, Hispanic/Latino, and White. For each subject, there are six 3D face model sequences corresponding to the six fundamental facial expressions (namely anger, disgust, happiness, fear, sadness, and surprise). In each sequence, the face models are captured at the rate of 25 models per second. Each face model in a sequence contains the shape and texture of the subject during an expression period. All expression periods typically last about 4 seconds (approximately 100 frames).

In total, the database contains 606 face model sequences, that is more than 60600 face models. Each face model consists of a cloud of points containing around 35000 vertices and a texture image with a resolution of 1040×1329.

The dynamic characteristics of this database is crucial in describing the intermediate period between the peak of the emotions and the neutral state, which are apparently very important for expression recognition. Our algorithm takes advantage of this property, and the database turns out to be quite appropriate to verify our algorithm.

IV. Facial Level Curves Based Representation of 3D Shapes

Facial level curves are defined to be the planar curves constituted from a facial surface, which are created by extracting the points with the same values of a distance function to a center points [5]. In our approach, the distance function is chosen to be the height of the points in a normalized coordinate system. In this section, the method for extracting facial level curves

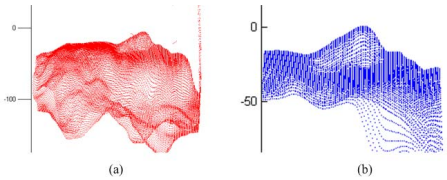

Fig. 2: Facial surface alignment. (a): Raw shape, (b): Aligned shape

and the method for comparing level curves and sets of level curves are described in detail.

A. Data preprocessing

Using the height information as the distance function has the advantages of fast processing time, being convenient with the input of range images and sensitive to expression changes. The most important challenge of using this function is the fact that it is not invariant to viewpoint changes. Therefore, in order to extract useful features with this distance, we need to perform several steps of preprocessing, namely pose normalization and face area extraction, to make sure the features extracted are unbiased and noise is not introduced by different viewpoints. Moreover, normalization will help us reduce the effect of identity variations among different subjects.

The face area and location of the nose and two eyes on a color image are located by the Pittpatt face detection and tracking library [11], [12]. This area is back projected to the 3D model to located the facial points in the 3D space. The outlier points are cut off from the mesh. The clean point cloud is then aligned so that it will be frontal when looking down from the z axis and the nose tip (the highest point in a frontal face) is at the coordinate center. The original and aligned faces are compared in Fig. 2.

In the next step of normalization, the face model is scaled down by suitable ratios for all of the three dimensions and stored in a range image. The purpose of this step is to squeeze a face model to make it fit in a fixed-size cube which will help reduce the variation in the geometry of the face. The range image for every face has now the same horizontal and vertical sizes. The depth values are scaled so that the depth of the nose tip has the value 1 and the depth of the eyes have the value 0.4. This will help provide the correspondence of the levels across different surfaces. The depth images are then ready to be fed into the main feature extraction process. A sample result of the preprocessing step is show in Fig. 3(a).

B. Shape representation

In our framework, a facial shape is represented as a collection of planar curves obtained by having a plane parallel to the xy plane moving down along the z axis to cut the aligned surface. At each level, the intersection of the plane and the surface is a planar curve. In the experiments, this process is done by going through various height levels. At the level of height h, we find all the points with a height in the range of $(h-\delta, h+\delta)$, with δ being some predefined constant. This set of points form a band with the thickness of 2δ. Projecting this band to the xy plane and finding a contour through the points, we have a planar facial curve. A sample of the extracted curves of a sample face is shown in Fig. 3(b). The curves are then stored as binary images which has value one at curve points and zero at others.

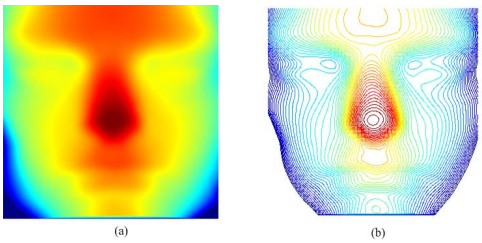

Fig. 3: Range image formation and facial level curves extraction in Matlab's jet colormap. (a): Normalized range image (b): Extracted facial level curves

Based on the level curve representation of facial surfaces, the deformation of the face over time is extracted by comparing the faces in two consecutive frames in the sequence. The method of defining and extracting distances between curves and faces are described in the next section.

C. Localized Chamfer distances for curve and surface comparison

On top of the facial curves based representation method, several ways to estimate the deformation of 3D faces are introduced. In [13], Klassen et al. propose to use geodesics in the preshape space to compare 3D facial curves using direction functions or curvature functions. Using a similar shape representation as the one used in our work, Samir et al. compare two curves by finding the shortest geodesic path between the curves and use the L2 norm of the curves as the distance between the two faces [9].

In our work, with the objectives of building the features which can be extracted efficiently while at the same time exposing the deformation of the curves at local locations instead of the global deformation of the entire closed curves. This inspires us to segregate each facial curve into a set of small segments at corresponded locations and extract the deformation of each segment over time. With that intention, we propose to use the Chamfer distances of curve segments partitioned by a set of arclength boundaries which we call *localized Chamfer distance*. The detail of the partition process will be addressed next.

Given that a planar curve is simple (i.e. with no self crossings) and closed, it can be represented by an *arclength parameterized function* [13]:

$$\alpha(s) : [0, 2\pi] \rightarrow \mathbb{R}^2 \tag{1}$$

.

This function gives the 2D location of the curve points given the angle between the line connecting them to the coordinate center and the Ox axis (i.e. the corresponding arc length of the unit circle).

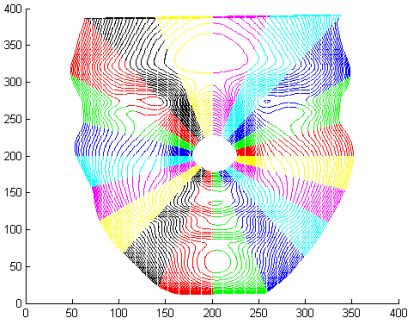

Fig. 4: The partitions of facial curves into bins by the arclength parameterized function. In this case, there are 20 segments made from a curve

The partition of the curve into n curve segments is done by breaking the function into n functions with the supports to be subsets of the main function's support. Each subset corresponds to an equal range of arclength s. The curve segment number i is defined by

$$\alpha_i(s) : [i\frac{2\pi}{n}, (i+1)\frac{2\pi}{n}) \rightarrow \mathbb{R}^2. \quad (2)$$

This segment partitioning process is equivalent to dividing a binary image of a curve into n sectors, with each sector covering an angle of $2\pi/n$ radians. The segments are the part of curves lying in the corresponding sectors. The illustration of this process in the case of 20 bins is depict in Fig. 4.

In our current method, because we have only one center point located at the nose tip, the curves are not always simple and closed, such as the ones drawn in Fig.5. Therefore, the arclength parameterized map is not stricly a function. However, we can still use the map in the same manner to obtain the curve partition with the similar intuitive meaning. The ultimate way to solve this problem is to use many center points and choose the suitable ones for different parts of curves so that they will satisfy the closed and simple requirements. This improvement will be implemented in the future work.

The comparison between two same-level curves are done by finding the Chamfer distances of the n pairs of curve segments and stack them up to form a distance vector. This vector represents the deformation intensity of the level curve over time in different directions. The distance vectors of the curves are further stacked to represent the deformation of the face. Therefore, at each frame transition we have a raw feature vector of $n \times l$ dimensions where l is the number of curves used. A sample of two curves being compared is depicted in Fig. 5

This representation indirectly implies a reference frame for correspondence. As the faces are now not represented as a set of points, this type of correspondence does not show the displacement of one particular point on the facial skin. Instead, comparing two corresponding curve segments of the same level and same sector from two consecutive frames shows how

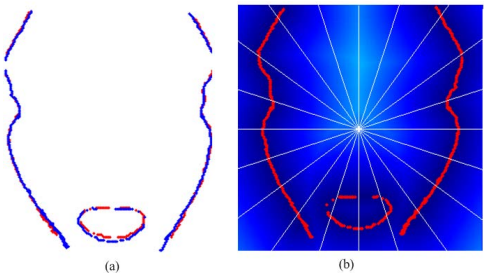

Fig. 5: Comparison between two sample curves. (a): two curves superimposed on each other(b): one curve superimposed on the other's distance transform image. The white lines indicate partition boundaries

the facial shape deforms at that relative location in space over time. Hence, this representation is able to express the shape deformation in the spatio-temporal realm.

In practical experiments, this representation may face the curse of dimensionality, due to the size of the distance vectors. Also, it only represents the deformation at one particular point in time whereas the current bigger context of the dynamics can make the feature more expressive. To alleviate those challenges, we utilize several steps of distance metric learning on the features which are detailed in the next subsection.

D. Distance metric learning

In our framework, we perform context expansion of the feature vectors. For each frame, we form an augmented vector by stacking up the feature vectors of the previous, current and future frames. This augmented vector is called the context expanded feature vector of current frame. Context expansion takes into account the temporal dynamics of the feature vectors, and is demonstrated to be an important stage for expression recognition from 3D dynamic faces. After context expansion, the feature vector has the size of $3 \times n \times l$ which span a space of thousands of dimensions. To reduce the dimensionality we use principal component analysis (PCA), followed by linear discriminant analysis (LDA). After this step, the feature vectors are ready to be used in the main classification algorithm which is described in the next section.

V. Decision Boundary Focus Classification with Hidden Markov Models

In this work, we adopt hidden Markov models (HMMs) [14] for facial expression classification. An HMM is a doubly stochastic process which consists of an underlying discrete random process possessing the Markov property (namely a Markov chain having a finite number of states) and an observed random process generated by a set of probabilistic functions of the underlying Markov chain, one of which is associated with each state of the Markov chain. At a discrete time instant, the HMM is assumed to be at a certain state, and an observation is generated by the probabilistic function associated with that particular state. The underlying Markov chain changes its state at every time instant according to some

state transition probability distribution. Note that within an HMM, it is only the observations that are seen by an observer, who does not have any direct knowledge of the underlying state sequence that generated these observations. HMMs have been proven to be a natural and effective probabilistic models for sequential data such as audio, speech and video.

However, due to their generative nature, HMMs may not perform as well as discriminative models for the classification purpose when there is insufficient training data. To overcome this difficulty, we introduce a strategy based on the maximum a posteriori (MAP) adaptation of a universal background model (UBM) to generate the individual states of the class-dependent HMMs. For reasons that will be clear shortly, this strategy enhances the discriminatory power of the class-dependent HMMs by focusing on the decision boundaries when applied to classification tasks.

A. HMM-based facial expression recognition

Without loss of generality, we assume that a 3D facial expression model sequence is effectively represented by a sequence of observation vectors $O = \mathbf{o}_1\mathbf{o}_2\cdots\mathbf{o}_T$. The joint probability distribution of O is a generative model which could have generated O. Let $p(O|c)$ denote the class-dependent model for facial expression c. The Bayesian or minimum-error-rate classification rule [15] is given by

$$c = \underset{c}{\operatorname{argmax}}\, p(c|O) = \underset{c}{\operatorname{argmax}}\, p(O|c)p(c) \tag{3}$$

where $p(c)$ is the prior probability of facial expression c. In this work, $p(O|c)$ is modeled by an HMM, namely

$$p(O|c) = \sum_{q_1 q_2\cdots q_T} \left[\pi^c_{q_1} b^c_{q_1}(\mathbf{o}_1) \prod_{t=2}^{T} a^c_{q_{t-1}q_t} b^c_{q_t}(\mathbf{o}_t)\right] \tag{4}$$

The above formulas form the basis of HMM-based facial expression recognition.

B. The Baum-Welch learning algorithm

To perform HMM-based facial expression recognition, we need to learn a class-dependent HMM for every facial expression. An HMM is completely determined by its parameters $\lambda = \{A, B, \Pi\}$. Here, A is the state transition probability matrix whose entries, $a_{ij} = P(q_t = S_j|q_{t-1} = S_i)$, $1 \le i, j \le N$, specify the probabilities of transition from state S_i to state S_j at time t. B is the state emission probability matrix whose entries, $b_{jk} = P(o_t = v_k|q_t = S_j)$, $1 \le j \le N, 1 \le k \le M$, specify the probabilities of emitting an observation symbol v_k given that the model is in state S_j at time t. Π is the initial state probability matrix whose entries, $\pi_i = P(q_1 = S_i)$, $1 \le i \le N$, specify the probabilities of the model being initially in state S_i. For the case of continuous observations, the entries of the state emission probability matrix are given by continuous probability density functions, namely $b_j(o_t) = P(o_t|q_t = S_j)$, $1 \le j \le N$. One important class of continuous probability density functions widely used for the state emission densities of the continuous-observation HMM is the Gaussian mixture density functions of the form

$$\begin{aligned} b_j(o_t) &= \sum_{k=1}^{M} c_{jk} N(o_t|\mu_{jk}, \Sigma_{jk}) \\ & 1 \le j \le N, 1 \le k \le M \end{aligned} \tag{5}$$

where M is the number of Gaussian components, c_{jk} is the k^{th} mixture weight, and $N(o_t|\mu_{jk}, \Sigma_{jk})$ is a multivariate Gaussian density function with mean vector μ_{jk} and covariance matrix Σ_{jk}. Since $b_j(o_t)$ is a valid probability density function, the following constraints must hold:

$$c_{jk} \ge 0, \quad 1 \le j \le N, \quad 1 \le k \le M \tag{6}$$

$$\sum_{k=1}^{M} c_{jk} = 1, \quad 1 \le j \le N \tag{7}$$

The Baum-Welch algorithm [14] for learning the parameters of an HMM, λ, given an observation sequence, $O = o_1 o_2 \cdots o_T$, is an iterative re-estimation procedure that increases the log likelihood $P(O|\lambda)$ monotonically. It is based on an efficient algorithm known as the forward-backward algorithm. In the forward algorithm, a "forward" variable, $\alpha_t(i)$, is defined as the probability of the partial observation sequence up to time t and state S_i being occupied at time t

$$\begin{aligned} \alpha_t(i) &= P(o_1 o_2 \cdots o_t, q_t = S_i|\lambda) \\ & 1 \le i \le N, 1 \le t \le T \end{aligned} \tag{8}$$

The following iterative formulas are used to compute the α's efficiently:

$$\alpha_1(i) = \pi_i b_i(o_1), \quad i = 1, 2, \cdots, N \tag{9}$$

$$\begin{aligned} \alpha_t(j) &= \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(o_t) \\ & j = 1, 2, \cdots, N, t = 2, 3, \cdots, T \end{aligned} \tag{10}$$

In the backward algorithm, a "backward" variable, $\beta_t(i)$, is defined as the probability of the partial observation sequence from time $t+1$ to T given that the state S_i is occupied at time t

$$\begin{aligned} \beta_t(i) &= P(o_{t+1} o_{t+2} \cdots o_T|q_t = S_i, \lambda) \\ & 1 \le i \le N, 1 \le t \le T \end{aligned} \tag{11}$$

Likewise, the following iterative formulas are used to efficiently compute the β's:

$$\beta_T(j) = 1, \quad j = 1, 2, \cdots, N \tag{12}$$

$$\begin{aligned} \beta_t(i) &= \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \\ & i = 1, 2, \cdots, N, t = T-1, T-2, \cdots, 1 \end{aligned} \tag{13}$$

Starting with random initialization (or initialization with a smarter scheme) of the model parameters λ, the Baum-Welch algorithm proceeds as follows:

(1) Re-estimation of the new model parameters $\hat{\lambda}$:

$$\hat{\pi}_i = \frac{\alpha_1(i)\beta_1(i)}{\sum_{j=1}^{N}\alpha_1(j)\beta_1(j)} \quad (14)$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{t=1}^{T-1}\alpha_t(i)\beta_t(i)} \quad (15)$$

$$\hat{c}_{jk} = \frac{\sum_{t=1}^{T}\gamma_t(j,k)}{\sum_{t=1}^{T}\sum_{k=1}^{M}\gamma_t(j,k)} \quad (16)$$

$$\hat{\mu}_{jk} = \frac{\sum_{t=1}^{T}\gamma_t(j,k)o_t}{\sum_{t=1}^{T}\gamma_t(j,k)} \quad (17)$$

$$\hat{\Sigma}_{jk} = \frac{\sum_{t=1}^{T}\gamma_t(j,k)(o_t-\hat{\mu}_{jk})(o_t-\hat{\mu}_{jk})^T}{\sum_{t=1}^{T}\gamma_t(j,k)} \quad (18)$$

where $1 \le i,j \le N, 1 \le k \le M$, and

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)}\frac{c_{jk}N(o_t|\mu_{jk},\Sigma_{jk})}{\sum_{m=1}^{M}c_{jm}N(o_t|\mu_{jm},\Sigma_{jm})} \quad (19)$$

(2) Test of convergence: If $\|\hat{\lambda}-\lambda\|_2 < \theta$ (a threshold), the Baum-Welch algorithm converges. Otherwise, set $\lambda = \hat{\lambda}$ and go to step (1).

Note that in the above Baum-Welch re-estimation formulas, $\gamma_t(j,k)$ can be viewed as the posterior probability that the observation o_t was generated from state S_j and accounted for by the k^{th} component of the Gaussian mixture density of state S_j, given the current model parameter λ.

C. Universal background modeling and maximum a posteriori adaptation

The Baum-Welch algorithm is a maximum likelihood learning technique for learning the model parameters of an HMM given a set of training observation sequences. Given a sufficient amount of training data, the Baum-Welch algorithm can be used to learn high-quality HMMs. However, in situations where there is insufficient training data available for learning an HMM, the Baum-Welch algorithm is likely to lead to a poorly estimated model. This is known as the over-fitting problem, which is a general property of maximum likelihood estimation techniques. To overcome this difficulty, we introduce universal background modeling. Universal background modeling was originally proposed for biometric verification systems which use a universal background model (UBM) to represent the general and person-independent feature characteristics to be compared against a model of person-specific feature characteristics when making an accept or reject decision. For example, in a speaker verification system, the UBM is a speaker-independent Gaussian mixture model (GMM) trained with speech samples that come from a large set of speakers, which is used when training the speaker-specific model by acting as the prior model in MAP parameter estimation [16]. Under the context of HMMs, the UBM is obtained as follows. We first pool the separate training data for learning the individual HMMs together to form an aggregated training data set. This aggregated training data set is normally large enough that we can assume that it is likely to capture sufficient variations in the population of the data. We then use the Baum-Welch algorithm to learn a single global HMM based on the aggregated training data set. We call this single global HMM learned on the aggregated training data set the UBM, as this single global HMM is supposed to represent the probability distribution of the observations drawn from the population of the data. The UBM is in general a well-trained and robust model, from which we can derive particular individual HMMs specific to small amounts of training data using the MAP adaptation technique for HMMs, as described in detail next. An HMM that is adapted from the well-trained UBM with a small amount of training data is proven to be far more robust than an HMM that is learned directly with the same small amount of training data using the Baum-Welch algorithm.

The rational behind universal background modeling and adaptation is as follows: In the Bayesian or minimum error classification rule, an optimal decision boundary is formed by the posterior probability distributions of two classes, which may be computed from the class-dependent likelihood probability distributions of the two classes, respectively. There may be a lot of fine structures in the class-dependent likelihood probability distribution of either class, and normally we require a lot of training data to learn these fine structures. However, as far as the classification problem is concerned, only the regions of the class-dependent likelihood probability distributions near the decision boundary are important. The fine structures of the class-dependent likelihood probability distributions which are away from the decision boundary are of no use. Therefore, it is a waste of the precious training data to try to learn these fine structures all over, and more disastrously, the fine structures (both near and away from the decision boundary) will never be properly learned if the available training data is insufficient. The introduction of universal background modeling allows us to learn the fine structures irrelevant to the classification problem using a large aggregated training data set, and focus on the regions near the decision boundary by learning the fine structures within these regions using small amounts of training data.

The concept of universal background modeling is related to the more elegant Bayesian learning theory [17]. In Bayesian learning, a prior probability distribution is imposed on the model parameters, which will be adjusted as more and more evidence is present. The UBM may be considered as a prior model corresponding to the prior probability distribution of the model parameters. Bayesian learning is a powerful learning paradigm which has many advantages over maximum likelihood learning. However, it is computationally very expensive. Universal background modeling serves as a good trade-off point between full Bayesian learning and maximum likelihood learning.

In the Baum-Welch algorithm, when we re-estimate the parameters of the Gaussian mixture state emission densities, $\hat{c}_{jk}, \hat{\mu}_{jk}, \hat{\Sigma}_{jk}, 1 \le j \le N, 1 \le k \le M$, instead of starting with randomly initialized parameters, we start with a UBM with parameters $\bar{c}_{jk}, \bar{\mu}_{jk}, \bar{\Sigma}_{jk}, 1 \le j \le N, 1 \le k \le M$. In the following, we will drop the index ranges to avoid cluttering the

equations by assuming that $1 \leq j \leq N, 1 \leq k \leq M, 1 \leq t \leq T$. For each observation vector o_t, we compute the posterior probability of o_t being generated by state S_j and Gaussian component k of the UBM

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \frac{\bar{c}_{jk}N(o_t|\bar{\mu}_{jk},\bar{\Sigma}_{jk})}{\sum_{m=1}^{M}\bar{c}_{jm}N(o_t|\bar{\mu}_{jm},\bar{\Sigma}_{jm})} \quad (20)$$

Based on these posterior probabilities, we compute the data sufficient statistics

$$n(j,k) = \sum_{t=1}^{T}\gamma_t(j,k) \quad (21)$$

$$\mu(j,k) = \frac{1}{n(j,k)}\sum_{t=1}^{T}\gamma_t(j,k)o_t \quad (22)$$

$$S(j,k) = \frac{1}{n(j,k)}\sum_{t=1}^{T}\gamma_t(j,k)o_t o_t^T \quad (23)$$

where $n(j,k)$ can be interpreted as the (fractional) number of observation vectors for which the state S_j and Gaussian component k of the UBM are responsible. Notice that the model sufficient statistics given by the UBM are

$$\tilde{n}(j,k) = T\bar{c}_{jk} \quad (24)$$

$$\tilde{\mu}(j,k) = \bar{\mu}_{jk} \quad (25)$$

$$\tilde{S}(j,k) = \bar{\Sigma}_{jk} + \bar{\mu}_{jk}\bar{\mu}_{jk}^T \quad (26)$$

The MAP adaptation technique generates a new set of sufficient statistics by interpolating the data and model sufficient statistics, namely

$$\hat{n}(j,k) = \rho^{(1)}n(j,k) + (1-\rho^{(1)})\tilde{n}(j,k) \quad (27)$$

$$\hat{\mu}(j,k) = \rho^{(2)}\mu(j,k) + (1-\rho^{(2)})\tilde{\mu}(j,k) \quad (28)$$

$$\hat{S}(j,k) = \rho^{(3)}S(j,k) + (1-\rho^{(3)})\tilde{S}(j,k) \quad (29)$$

where the interpolation coefficients, $\rho^{(1)}, \rho^{(2)}, \rho^{(3)}$, are smartly adaptive to the amount of available training data according to the following empirical formula

$$\rho^{(l)} = \frac{n(j,k)}{n(j,k)+r^{(l)}}, \quad l = 1,2,3 \quad (30)$$

with $r^{(l)}$ being a tunable constant specified by the user.

The new set of sufficient statistics is now used for re-estimating the model parameters

$$\hat{c}_{jk} = \hat{n}(j,k)/T \quad (31)$$

$$\hat{\mu}_{jk} = \hat{\mu}(j,k) \quad (32)$$

$$\hat{\Sigma}_{jk} = \hat{S}(j,k) - \hat{\mu}_{jk}\hat{\mu}_{jk}^T \quad (33)$$

We call the above algorithm the UBM adapted Baum-Welch (UBM-BW) algorithm, in which the re-estimation formulas for the Gaussian mixture state emission densities are replaced by the above MAP adaptation formulas. Note that in the second iteration of the algorithm, the newly adapted Gaussian mixture state emission densities will replace the UBM in the re-estimation formulas. From then on, the estimated Gaussian mixture densities at a current iteration will serve as a prior model for the next iteration. The UBM-BW algorithm is summarized in Algorithm 1.

Algorithm 1 The UBM adapted Baum-Welch algorithm

1: Input: the UBM $\bar{b}_j(o_t)$, training data $O = o_1 o_2 \cdots o_T$.
2: Initialization: Π, A.
3: Set $b_j(o_t) = \bar{b}_j(o_t), \quad 1 \leq j \leq N$.
4: Repeatedly perform parameter re-estimation using Equations (14), (15), and (20)-(33) until convergence.
5: Output: final parameter estimates Π, A, B.

It is worth mentioning that in Equation (30), when the number of observation vectors for which state S_j and Gaussian component k are responsible is small, i.e., $n(j,k) \approx 0$, $\rho^{(l)}$ approaches $1/r^{(l)}$. In this case, the model sufficient statistics will dominate the new sufficient statistics, and hence the new model parameters will remain close to those of the prior model (e.g. the UBM). When the number of observation vectors for which state S_j and Gaussian component k are responsible is large, i.e., $n(j,k) \to \infty$, $\rho^{(l)}$ approaches 1. In this case, the model sufficient statistics will vanish from the interpolation formulas, and the new sufficient statistics consist of only the data sufficient statistics. This is exactly the case of the original Baum-Welch algorithm.

VI. Experiments

To demonstrate the effectiveness of our proposed methods, we perform expression recognition from 3D dynamic faces based on the data of 60 subjects from the BU-4DFE database. Due to the time constraint, we choose to conduct preliminary experiments on three most commonly seen expressions: happiness, sadness, and surprise.

All the 3D face models are preprocessed and stored as depth images of size 200×200 pixels, which are similar to the output of consumer depth cameras. The depth dimension is scaled to the range $[0, 1]$. As the lengths of the sequences are not always the same, we choose to keep 100 frames in each sequence. The feature vectors of sequences with lengths greater than 100 are trimmed at both sides whilst the shorter ones are zero padded.

In the facial level curve extraction phase, the step size between the curve levels is chosen to be 0.02 and the band size is 0.02. This means that all the points are used and distributed into one of 50 bands, ranging from 0 to 1 in depth. The bands are flattened out and stored as binary images. In the binary images, the number of points in the bands are different and sometimes the contours are discontinuous. We did some more morphological operations such as thinning and bridging to correct these issues.

On these binary images, we extract the distances between pairs of same-level curves at consecutive frames using the Chamfer distance of curve segments partitioned by arclength parameterized function. The number of sectors are chosen to be 20, and we compare the curves of 30 highest levels. After stacking up the distances, we have raw feature vectors of 600

dimensions. After context expansion, the feature vector dimension becomes 1800. We then perform PCA and LDA on the context expanded feature vectors. At the PCA step, we keep 100 first principal components. At the following LDA step, because of the small number of classes, we propose a special treatment: we divide every sequence into 5 subsequences of 20 frames each. The first subsequences of all sequences are labeled as 1, the last ones of all are labeled as 2, and all other subsequences are labeled based on the class labels of their parent sequence (e.g. 1-1,1-2,1-3). With this strategy, we expand the number of classes from 3 to 11. Using these finer artificial classes, LDA reduces the dimension of the feature vectors to 10. These 10D features vectors are now ready to be used for classification.

We randomly partition the 60 subjects into 10 sets, each containing 6 subjects. The experiment results are based on 10-fold cross validation, where at each round, 9 of the 10 folds (54 subjects) are used for training while the rest (6 subjects) are used for test. The recognition results of 10 rounds are then average to give a statistically significant performance measure of the algorithm. At each round, we first train a UBM, having 5 states and 16 Gaussian mixtures, with all the data for the 54 training subjects, regardless of the expressions. Once the UBM is trained, we adapt the UBM to the data for the 54 training subjects specific to the expressions, and generate three expression-dependent HMMs, one for each of the three expressions, namely happiness, sadness, and surprise. The Bayesian or minimum error rate classifier based on the adapted expression-dependent HMMs, as described in section V, is then used for expression recognition.

The results of our preliminary experiments show that the overall recognition accuracy is as high as 92.22%, with the highest performance obtained for the happiness expression: 95.00%. The confusion matrix is shown in table I. Note that our experiment results are very preliminary, as we have not had time to explore all the design choices. Nonetheless, these preliminary results clearly demonstrate the effectiveness of our proposed feature extraction and classification methods for expression recognition from 3D dynamic faces.

TABLE I: Confusion matrix.

	Happy	Sad	Surprise
Happy	0.95	0.0333	0.0167
Sad	0.0167	0.9167	0.0667
Surprise	0	0.1	0.9

VII. Conclusion and discussion

In this paper, we propose to use a facial level curves based representation of 3D facial shapes for expression recognition from 3D dynamic faces. We introduce a novel method for measuring the deformation of shapes in 3D face models. This method allows us to efficiently extract robust spatio-temporal local features. These features, when combined with an HMM-based decision boundary focus classification algorithm as a result of universal background modeling and maximum a posteriori adaptation, can yield very high performance on low resolution depth images.

With these promising preliminary results, our future work will be to quantitatively evaluate the robustness of the proposed features to noisy data captured from low-end consumer devices and to implement the benchmark test for the computation cost. These tasks will give us more clues to improve the algorithms. Besides, we will explore the idea of using many partition center points instead of a single one to assure that all facial level curves are simple in some coordinate system. We will investigate other distance functions such as the ones based on shape geodesics. The algorithms can also be parallelized and therefore are possible to be further sped up using multi-core processors to cater the needs of large-scale applications.

References

[1] I. Mpiperis, S. Malassiotis, and M. Strintzis, "Bilinear elastically deformable models with application to 3d face and facial expression recognition," in *FG'08*, 2008, pp. 1–8.

[2] J. Wang, L. Yin, X. Wei, and Y. Sun, "3d facial expression recognition based on primitive surface feature distribution," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR2006)*, 2006, pp. 1399–1406.

[3] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang, "A survey of affect recognition methods: Audio, visual, and spontaneous expressions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, January 2009.

[4] L. Yin, X. Chen, Y. Sun, T. Worm, and M. Reale, "A high-resolution 3d dynamic facial expression database," in *The 8th International Conference on Automatic Face and Gesture Recognition (FGR08)*, Sep. 2008, pp. 17–19.

[5] C. Samir, A. Srivastava, and M. Daoudi, "Three-dimensional face recognition using shapes of facial curves," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, pp. 1858–1863, November 2006.

[6] H. Tang and T. Huang, "3d facial expression recognition based on automatically selected features," in *3DFace08*, 2008, pp. 1–8.

[7] H. Tang and T. Huang, "3d facial expression recognition based on properties of line segments connecting facial feature points," in *FG'08*, 2008, pp. 1–6.

[8] Y. Sun and L. Yin, "Facial expression recognition based on 3d dynamic range model sequences," in *10th European Conference on Computer Vision (ECCV 2008)*, Oct. 2008, pp. 58–71.

[9] C. Samir, A. Srivastava, M. Daoudi, and E. Klassen, "An intrinsic framework for analysis of facial surfaces," *Int. J. Comput. Vision*, vol. 82, pp. 80–95, April 2009

[10] E. M. Bronstein, M. M. Bronstein, and R. Kimmel, "Three-dimensional face recognition," *International Journal of Computer Vision*, vol. 64, pp. 5–30, 2005.

[11] H. Schneiderman, "Feature-centric evaluation for efficient cascaded object detection." in *CVPR (2)'04*, 2004, pp. 29–36.

[12] H. Schneiderman, "Learning a restricted bayesian network for object detection." in *CVPR (2)'04*, 2004, pp. 639–646.

[13] E. Klassen, A. Srivastava, W. Mio, and S. H. Joshi, "Analysis of planar shapes using geodesic paths on shape spaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 26, no. 3, pp. 372–383, Mar. 2004.

[14] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, vol. 77, no. 2, 1989, pp. 257–286.

[15] R. Duda, P. Hart, , and D. Stork, "Pattern classification," in *Wiley-Interscience*, 2001.

[16] D. A. Reynolds, "Universal background models," in *Encyclopedia of Biometric Recognition Secaucus, NJ: Springer-Verlag*, 2008, pp. 31–36.

[17] C. Bishop, "Pattern recognition and machine learning," in *Secaucus, NJ: Springer-Verlag*, 2006.