Bayesian versus Support Vector Machine based Approaches for Facial Feature Classification in Image Sequences

Rajesh A Patil
NIT Jaipur
Email:rapatil_rtg@yahoo.co.in

Vineet Sahula
NIT Jaipur
Email:sahula@ieee.org

A. S. Mandal
CEERI Pilani
Email:atanu@ceeri.ernet.in

***Abstract*—A method for automatic facial expression recognition in image sequences, is introduced which make use of Candide wire frame model and active appearance algorithm for tracking, and Bayesian classifier for classification. On the first frame of face image sequence, Candide wire frame model is adapted properly. In subsequent frames of image sequence, facial features are tracked using active appearance algorithm. The algorithm adapts Candide wire frame model to the face in each of the frames and tracks the grid in consecutive video frames over time. Last frame of image sequence corresponds to greatest facial expression intensity. The difference of the node coordinates between the first and the greatest facial expression intensity frame, called the geometrical displacement of Candide wire frame nodes is used as an input to a classifier, which classifies facial expression into one of the class such as happy, surprise, sad, anger, disgust and fear. The experimental results show that the proposed method is better in classification correctness in comparison with binary SVM tree classifier.**

***Index Terms*—Feature recognition, Candide wire frame model, feature tracking, Bayesian classifier, SVM**

I. Introduction

The automatic acquisition and analysis of images to obtain desired data for interpreting a scene or controlling an activity is called machine vision. Machine vision is a difficult task - a task that seems relatively trivial to humans is actually complex for computers to perform. Current machine vision research concerns, not only understanding the process of vision, but also designing effective vision systems for various real world applications. Facial expression recognition system is an example of machine vision system.

For a human being facial expression is one of the most powerful, natural and immediate means to communicate their emotions and intentions. Facial expressions can contain a great deal of information, hence the demand of automatically extracting this information has been continuously increasing. Automatic facial expression analysis is an interesting and challenging problem, and impacts important applications in many areas such as human computer interaction, and data driven animation. Various applications using automatic facial expression analysis can be envisaged in the near future, fostering further interest in doing research in different areas, including image understanding, psychological studies, facial nerve grading in medicine, face image compression and synthetic face animation, video indexing, robotics as well as virtual reality [1]. Though much progress has been made, recognizing facial expression with a high accuracy remains a difficult problem due to subtlety, complexity and variability of facial expressions. An effective automatic expression recognition system could take human computer interaction to the next level. Although, facial expression recognition looks simple, it is very difficult because of high variability that can be found in images containing a face. We can see an extremely large variety in lighting conditions, resolution, pose and orientation. In the next section we present a review of major approaches for facial expression recognition.

II. Related work

Good surveys on the research made regarding facial expression recognition in image sequences can be found in [1] and [2]. Facial expression recognition problem in image sequences can be divided into three subproblems face detection, feature extraction and tracking and classification. Before a facial expression can be analyzed, the face must be detected in a scene. Developing a mechanism for extraction of the facial expression information from the observed facial image sequence and then track these features in subsequent frames. Developing a mechanism to classify facial expressions into one of the basic facial expression.

Essa and Pentland [3] make the use of eigenspace and eigenfeatures method to automatically track the face in the scene and extract the positions of the eyes, nose, and mouth. They have applied principal component analysis (PCA) and have used optical flow computation method to obtain motion estimates and error-covariance information. They have generated the spatio-temporal templates for six different expressions two facial actions (smile and raised eyebrows) and four emotional expressions (surprise, sadness, anger, and disgust). Kimura and Yachida [4] utilize a potential net for face representation. The pattern of the deformed net is compared to the pattern extracted from an expressionless face (usually the first frame of the sequence), and the variation in the position of the net nodes is used for further processing. The authors have built an emotion space by applying PCA on six image sequences carrying three expressions anger, happiness, and surprise shown by a single person gradually. The eigenspace spanned by the first three principal components has been used as the emotion space, onto which an input image is projected for a quantified

978-1-4577-1386-611$26.00©2011 IEEE

emotional classification. Cohn et al. [5] have used a model of facial landmark points localized around the facial features, hand-marked with a mouse device in the first frame of an examined image sequence. In the rest of the frames, a hierarchical optical flow method is used to track the optical flow of 13×13 windows surrounding the landmark points. The displacement vectors, calculated between the initial and the peak frame, represent the facial information used for recognition of the displayed facial expression. Wang et al. [6] utilize 19 facial feature points (FFPs) - seven FFPs to preserve the local topology and 12 FFPs for facial expression recognition. The FFPs are treated as nodes of a labeled graph that are interconnected with links representing the Euclidean distance between the nodes. The initial location of the FFPs in the first frame of an input image sequence is assumed to be known. The FFPs are tracked in the rest of the frames. The correspondence between the FFPs tracked in two consecutive frames is treated as a labeled graph matching problem. The degree of expression change is determined based on the displacement of the FFPs in the consecutive frames. M. Valstar et al. [7] have proposed a system that performs AU recognition using temporal templates as input data. Temporal templates are introduced by Bobick and Davis. These templates are 2D images constructed from image sequences, effectively reducing a 3D spatio-temporal space to a 2D representation. They have used Neural Network as a classifier. Black and Yacoob [8] are using local parametrized models of image motion for facial expression analysis. The motion parameters (e.g., translation and divergence) are used to derive the mid level predicates that describe the motion of the facial features. For each of the six basic emotional expressions, they developed a model represented by a set of rules for detecting the beginning and ending of an expression. The rules are applied to the predicates of the mid level representation.

Seyed Mehdi Lajevardi and Margaret Lech [9] have proposed a method which is fully automatic. They have used Viola Jones method for face detection. For feature extraction they have made use of Log Gabor filters, and as a classifier they have made use of Naive Bayesian (NB) Classifier. Irene kotsia and Pitas [10] have proposed a method which is based on mapping and tracking the facial model Candide onto the video frames. The proposed facial expression recognition system is semi-automatic, in the sense that the user has to manually place some of the Candide grid nodes on face landmarks depicted at the first frame of the image sequence under examination. Using Kanade Lucas Tomasi tracker Caidide grid tracks facial expressions in subsequent frames. The geometrical displacement of node coordinates at the first and the last frame of the facial image sequence, is used as an input to a support vector machine classifier.

In the present work a method is proposed which makes use of Candide wire frame model. Facial feature tracking is done using active face model proposed by J. Ahlberg [11]. Expression classification is performed by Bayesian classifier. Classification is also performed using multiclass SVM and results are compared. Let us consider an image sequence containing face. The face is detected using Viola Jones algorithm [12]. First frame of the sequence gives neutral facial expression and last frame corresponds to fully expressed state. Our approach is automatic fitting of Candide wire frame model on the first frame of the image sequence. The tracking system allows the grid to follow the evolution of the facial expression over time till it reaches its highest intensity, producing the deformed Candide grid at each video frame. We use active face model for tracking. The geometrical displacement of Candide wire frame nodes, defined as the difference of coordinates of each node at the first and the last frame of the facial image sequence, is used as training data for Bayesian classifier. Our framework is different from the one proposed by Irene Kotsia and Pitas [10] who have used pyramidal Kanade Lucas Tomasi tracker [13], which is based on optical flow computation. We make use of Active face model proposed by J. Ahlberg [11] for tracking. As a classifier they have used multiclass SVM and we are using Bayesian classifier. This paper is organized as follows. The system used for facial expression recognition is described in Section III. Results are given in Section IV. We conclude, summarize and discuss limitations in Section V.

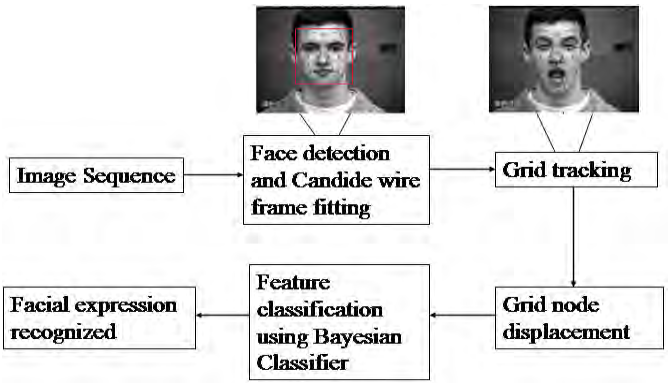

Figure 1. Flow diagram for facial expression recognition system

III. Facial expression recognition system

The proposed framework is composed of three subsystems. First is used for face detection. Second is for Candide grid node coordinate displacement extraction and third is used for grid node displacement classification. Face detection is performed by Viola Jones algorithm. The grid node information extraction is performed by Active face model tracking system, while the grid node information classification is performed by a classifier. The flow diagram of the proposed framework is shown in Figure 1.

A. Face Detection

In our case, as we wish the system to be fully automatic, we have to start by detecting the user's face inside the scene. Although, we seemed it an easy problem at first, we immediately realized that the high variability in the types of faces encountered would make the automatic detection of the face a tricky problem. Many different techniques have been reported in the literature for face detection. We selected Viola

Jones algorithm[12] for face detection. The result of face detection algorithm is shown in Figure 1.

B. Extraction of Candide grid node coordinates

1) Candide wire frame model: The facial wire frame model, we have used in the tracking procedure is the Candide wire frame model [14]. Candide wire frame is a parametrized face mask specifically developed for model-based coding of human faces. A frontal view of the model can be seen in Figure 1, fitted on face image. It has 113 vertices and 184 triangles. The small number of its triangles, allows fast face animation with moderate computing power. The geometry of the model as discussed in [14] can be expressed as in (1).

$$V(\sigma,\alpha) = \overline{V} + \sum_{i=1}^{14} S_i\sigma_i + \sum_{i=1}^{65} A_i\alpha_i \quad (1)$$

Here the resulting vector V contains (x, y, z) coordinates of vertices of the model. $\overline{V}$ is vector containing vertex coordinates of standard model. S_i represents a shape unit. There are 14 shape units, such as head height, mouth width, eyebrows vertical position, eyes width etc. The parameter σ_i is shape parameter. A_i represents animation unit. There are 65 animation units such as lip stretched, nose wrinkle, inner brow raiser, outer brow raiser etc. Whereas α_i is animation parameter. The difference between shape and animation modes is that the shape modes define deformations that differentiate individuals from each other, while the animation modes define deformations that occur due to facial expression. To perform global motion of the model, six more parameters three for rotation, one for scaling, and two for translation are added to formula in (2). Here $R = (\theta_x, \theta_y, \theta_z)$ is rotation matrix, s is scale, and $t = (t_x, t_y)$ is a 2D translation vector. The geometry of the model is thus parametrized by (3).

$$V(R, s, \sigma, \alpha, t) = Rs(\overline{V} + S\sigma + A\alpha) + t \quad (2)$$

$$p = [\theta_x, \theta_y.\theta_z, s, t_x, t_y, \sigma, \alpha]^T \quad (3)$$

Once the model is adapted properly on the first frame, for the subsequent frames only α will change. Our goal is to find the optimal adaptation of the model to the input image i.e. to find *p* that minimizes the distance between the model and the image. The Candide model is adapted to a set of images using different parameters: 3D -rotation, 2D -translation, scale, and action units. We collect those parameters in a vector *p*, which thus parametrizes the geometry of the model. For each image in the training set, the image under the wire frame model is mapped to the model, and the model is then normalized to a standard shape, size, and position, in order to collect a geometrically normalized set of textures. On this set of textures, a PCA has been performed and the eigentextures (geometrically normalized eigenfaces) have been computed as $x = \overline{x} + X\xi$ [11]. Where $\overline{x}$ is mean texture, X is eigen texture and ξ is texture parameter. We can now describe the complete appearance of the model by the geometry parameters p and an *N* dimensional texture parameter vector, where *N* is the number of eigentextures we want to use for synthesizing the model texture. Given an input image and a p, the texture parameters are given by projecting the normalized input image on the eigentextures, and thus p is the only necessary parameter in our case. Geometrical normalization of the face used to obtain its normalized texture removes texture variations caused by its global and local motion and geometrical differences between individuals. We choose to work with 33×40 pixels images which are conveniently small and effective for image warping. Once the model is fitted on the first frame, 113 vertices of the model represents the facial features, which are tracked in subsequent frames. Difference between these vertices coordinates from first and last frame, represents facial feature deviation, which is responsible for facial expression is given to SVM classifier.

2) Texture Synthesis: Perform PCA on the training set (stored as 33×40 texture vector) so that we obtain the principal modes of variation, i.e., the eigenfaces. In this case, we collect 32 eigenfaces in a matrix X which could represent 90% of variance. A face vector x can be parametrized as in (4), where $\overline{x}$ is the mean face, and we could synthesize a face image as in (5).

$$\xi = X^T(x - \overline{x}), \quad (4)$$

$$\hat{x} = \overline{x} + XX^T(x - \overline{x}) \quad (5)$$

3) Grid Tracking: Facial tracking means to find optimal adaptation of model to frames in image sequence. It can be obtained by finding the parameter vector p that minimizes the distance between normalized and synthesized faces.

The initial value of p we use is the optimal adaptation to the previous frame. Assuming that the motion from one frame to another is small enough, we reshape the model to $V(p)$ and map the image i (the new frame) onto the model. Then we geometrically normalize the shape and get the resulting image as a vector.

Map the input Image *(i)* on the model. Reshape the model to the standard shape $\overline{V}$ and get the resulting normalized image as a vector as in (6). Then we compute texture parameters from normalized image as in (7). Synthesized texture would be given as in (8). Whereas residual image is calculated as in (9).

$$j(i, p) = j(i, V(p)) \quad (6)$$

$$\xi(i, p) = X^T(j(i, p) - \overline{x}) \quad (7)$$

$$x(i, p) = \overline{x} + XX^T(j(i, p) - \overline{x}) \quad (8)$$

$$r(i, p) = j(i, p) - x(i, p) \quad (9)$$

Summed square error (SSE) is selected as error measure and is given as

$$e = \|r(i, p)\|^2 \quad (10)$$

For good model adaptation residual image and error e is much smaller. Find the update vector $\triangle p$ by multiplying residual image with an update matrix U

$$\triangle p = \mathrm{U}r(p) \quad (11)$$

The new error measure for updated parameter is as follows, if $e_0 < e$ we update $e_0 \rightarrow e$ and $p + \triangle p \rightarrow p$

$$e_0 = \|r(i, p + \triangle p\|^2 \quad (12)$$

if not, try smaller steps. e is recomputed as follows for $k = 1, 2, 3, ...$ if $e_k < e$ we update $e_k \rightarrow e$ and $p + \frac{1}{2^k} \triangle p \rightarrow p$ Iterate the scheme and declare the convergence when $e_k > e$.

$$e_k = \left\| r(i, p + \frac{1}{2^k} \triangle p \right\|^2 \tag{13}$$

4) Update Matrix: Assuming that $r(i, p)$ is linear in p as given in (14). Taylor expanding $r(i, p)$around $p + \triangle p$ results in (15), where we want to find $\triangle p$ that minimizes error e as in (16).

$$\frac{\partial}{\partial p} r(i, p) = G \tag{14}$$

$$r(i, p + \triangle p) = r(i, p) + G \triangle p + O(\triangle p^2) \tag{15}$$

$$e(i, p + \triangle p) = \|r(i, p) + G \triangle p\|^2 \tag{16}$$

Minimizing above equation is least square problem with the following solution, which gives update matrix U as the negative pseudo inverse of the gradient matrix G.

$$\triangle p = -(G^T G)^{-1} G^T r(i, p) \tag{17}$$

$$\mathrm{U} = -(G^T G)^{-1} G^T \tag{18}$$

Gradient matrix *G* is computed from training images in advance, with j^{th} column in *G* is given by (19). The approximation on this column could be given as in (20).

$$G_j = \frac{\partial}{\partial p_j} r(i, p) \tag{19}$$

$$G_j \approx \frac{r(i, p + hq_j) - r(i, p - hq_j)}{2h} \tag{20}$$

Here, *h* is the step size for perturbation and q_j is a vector with one in j^{th} column and zero in the rest elements. The Candide wire frame model was adapted to every training image in the training set to compute the shape and texture modes. So, a set of corresponding parameter vectors p_n is obtained for a suitable step size to estimate G_j by averaging as where *N* is the number of training images and *K* is the number of steps to perturb the parameter [15].

$$G_j \approx \frac{1}{NK} \sum_{n=1}^{N} \sum_{k=1}^{K} \frac{r(i_n, p_n + khq_j) - r(i_n, p_n - khq_j)}{2h} \tag{21}$$

To fit Candide model on first frame of image sequence, we consider scaling, rotation and translation parameters. For initial fitting rough estimate of scaling and translation parameter is very essential. Our face detection algorithm gives a rectangle enclosing a face. Top left point of this rectangle has coordinates (x, y). Width of rectangle is C. Height of rectangle is D. From this data we get rough estimate of translation parameters (t_x, t_y). The Candide model is fitted manually on 100 images. From manual fitting we have selected rough estimate of scaling parameter as $s = 0.78 * C$. With rough estimation of scaling and translation parameters, Candide model roughly fits on face image. Initially rotation is assumed to be zero. Once the model roughly fits on face image, exact fitting is obtained by active appearance algorithm. Separate update matrix is constructed only for scaling, translation and rotation parameters. For unknown image, when model fits roughly, residual image is computed. Residual image gets multiplied with update matrix to get updated parameter vector $p = [s, t_x, t_y, \theta_x, \theta_y, \theta_z]$. With this new parameter, model gets deformed, again residual image is computed, and the process repeats till model fits exactly, where error e is reduced to minimum. Once the model fits properly on the first frame, in the subsequent frames only animation parameters need to be processed [16].

C. Classification

The classification is performed only on the basis of geometrical information, not taking into consideration any luminance or color information. The geometrical information used is the displacement of one point, defined as the difference between the last and the first frame's coordinates. For every image sequence to be examined, a feature vector is constructed, containing the geometrical displacement of every point taken into consideration.

Let μ be the video database that contains the facial image sequences. It is clustered into six different classes μ_k, $k = 1, ..., 6$, each one representing one of the six basic facial expressions. The geometrical information used for facial expression recognition is the displacement of one node d_{ij}, defined as the difference of the i^{th}grid node coordinates at the first and fully formed expression facial video frame $\mathrm{d}_{ij} = [\triangle x_{ij} \quad \Delta y_{ij}]^T$, where $i = 1, ..., E$ and $j = 1, ..., N$, and $\triangle x_{ij}$, $\triangle y_{ij}$ are the x, y coordinate displacement of the i^{th} node in the j^{th}image respectively. E is the total number of nodes and N is the number of the facial image sequences. This way, for every facial image sequence in the training set, a feature vector $g_j = [d_{1,j} d_{2,j} ..... d_{E,j}]^T$ is created. The vector g_j is called grid deformation feature vector, which contains the geometrical displacement of every grid node. The dimension of vector g_j is $D = 113 \times 2 = 226$ dimensions. Out of 113 nodes only 60 nodes are contributing for facial expressions, so dimension of vector g_j is reduced to $D = 60 \times 2 = 120$. Each grid deformation feature vector g_j belongs to one of the six facial expression classes. The feature vector is used as training data for classifier, that classifies each set of Candide grid node's geometrical displacements to one of the 6 basic facial expressions happy, surprise, sad, anger, fear, disgust.

1) Bayesian Classifier: Bayesian learning methods are relevant to machine learning for two different reasons. First, Bayesian learning algorithms that calculate explicit probabilities for hypotheses, such as the naive Bayes classifier, are among the most practical approaches to certain types of learning problems. Michie et al.[17] provide a detailed study comparing the naive Bayes classifier to other learning algorithms, including decision tree and neural network algorithms. These researchers show that the naive Bayes classifier is competitive with these other learning algorithms in many cases and that in some cases it outperforms these other methods. The second reason is that they provide a useful perspective for understanding many learning algorithms that do not explicitly manipulate probabilities [18]. One practical difficulty in applying Bayesian methods is that they typically

require initial knowledge of many probabilities. When these probabilities are not known in advance they are often estimated based on background knowledge, previously available data, and assumptions about the form of the underlying distributions. The classification decision is made using the following formula, where $C_i = 1, ..., 6$ is class label and f_i is feature related to each sample. Using following formula, a test grid deformation feature vector is classified to one of the six facial expressions. Detailed description about bayesian classifier equations can be found in [18].

$$C = \text{argmax}\{P(C_i) \prod P(f_i \mid C_i)\} \tag{22}$$

2) Multiclass SVM (MC-SVM): Support vector machines (SVMs) are a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. Basically, SVMs maximize the hyper plane margin between different classes. They map input space into a high dimension linearly separable feature space. This mapping does not affect the training time because of the implicit dot product and the application of the kernel function. In principle the SVM technique finds the hyper plane from the number of candidate-hyper planes, which has the maximum margin. The margin is enhanced by support vectors, which are lying on the boundary of a class. The SVM takes a data set that contains samples from two classes (labeled -1 and +1), and constructs separating hyperplanes between them. The separating hyperplane that best separates the two classes is called the maximum-margin hyperplane and forms the decision boundary for classification. The data samples that lie at the boundary of each class and determine how the maximum-margin hyperplane is formed, are called support vectors (SVs). The SVs are obtained during the training phase and are then used for classifying new (unknown) data. When two data classes are not linearly separable, a kernel function is used to project data to a higher dimensional space (feature space), where linear classification is possible. This is known as the kernel trick and allows an SVM to solve nonlinear problems. An input is classified using decision function, where $\alpha_i$is weight of support vector, $y_i$is class label of support vector, $\vec{s_i}$is support vector, $\vec{x}$ is input vector. $K(\vec{x}.\vec{s_i})$ is kernel function, and b is bias value.

$$D(\vec{x}) = sign(\sum_{i=1}^{m} \alpha_i y_i K(\vec{x}.\vec{s_i}) + b) \tag{23}$$

It classifies data into two classes. Classifier i, j is trained using all patterns from class *i* as positive instances, and all patterns from class *j* as negative instances.

a) Binary SVM tree multi-class SVM classifier*:* To solve multiclass problem binary SVM is used in a hierarchical structure, called binary SVM tree. In binary SVM tree data set is divided into two subsets from root to the leaf until every subset consists of only one class. Only *C* -1 binary classifiers are constructed. Where *C* is number of classes. The steps in constructing binary tree are as follows [19]. Compute class centers using $m_i = \frac{1}{n}\sum x_t$ where m_i is mean of class *i.* x_t is training samples of i^{th} class. Then compute distance between classes using $D_{ij} = m_i - m_j$. After that compute radius of hyper-sphere in feature space using (24) and then compute class similarity using (25).

$$R_i = \max_{t = 1.....I_j} \parallel x_t - m_i \parallel \tag{24}$$

$$\text{similarity}(i, j) = \frac{R_i^2 + R_j^2}{\parallel m_i - m_j \parallel^2} \tag{25}$$

Select the two classes with biggest similarity to train SVM and unit these two classes. Compute new united class center and repeat the procedure. Construct the most upper SVM of binary tree when the number of the class equals to 2.

b) One against One multi-class SVM classifier*:* In this approach $C(C-1)/2$ classifiers are constructed. Where *C* is the number of classes. Classifier i, j is trained using all patterns from class *i* as positive instances, and all patterns from class *j* as negative instances and disregarding the rest. To combine obtained classifiers a simple voting scheme is used. When classifying a new instance each one of the base classifier casts a vote for one of the two classes used in its training. The class that gets maximum vote will be declared as class of new instance.

IV. Experimental results

The Cohn-Kanade data base [20] is used for constructing the update matrix as well as for SVM training. Viola Jones algorithm is successfully used for face detection in a scene. Candide wire frame model is fitted on the first frame and tracked in subsequent frame using active appearance algorithm which is implemented in MATLAB. For texture synthesis 40 images of different persons with different expressions are considered as training images. Candide model is manually adapted to these images, these images are then geometrically normalized (33 x 40 pixels) to standard shape, and then PCA is applied on them. For constructing update matrix we have selected 7 animation units. Theses are upper lip raiser, jaw drop, lip stretcher, eyebrow lowerer, eyebrow raiser, lip corner depressor, and nose wrinkler. Candide model is manually adapted to 40 training images, and then all the parameters have been perturbed one by one in steps of 0.01 in the range [-0.2, 0.2] to collect residual images. Number of steps we have selected is K=20, and number of images we have selected is N=40. Then the update matrix is computed. On each frame update matrix is multiplied with residual image to get updated parameter vector. Bayesian classifier is implemented in MATLAB. Image sequences from Cohn-Kanade database is used for training. From the database 25 image sequences of each class are selected for training. In case of binary tree SVM the accuracy is 61% as shown in Table I, while for one against one SVM accuracy is 79.5% as shown in Table II. For Bayesian classifier accuracy obtained is 74% as shown in Table III. Fear makes confusion with sad, anger and happy, while disgust makes confusion with sad and anger. Table IV gives comparison. Work is going on to improve classification accuracy for more number of expressions.

V. Conclusions

The active shape model and Bayesian classifier can be successfully used for facial expression recognition. Overall

accuracy of Bayesian classifier is 74% when six expressions are considered, while for binary tree SVM it is only 61%, and for one against one SVM it is 79.5%. Lot of precomputations are required in case of SVM to form a tree. For surprise, sad and disgust expressions confusion percentage is more in SVM. The tracking system used, is based on active shape model, which uses active appearance algorithm. Face is detected from a scene in first frame of image sequence and Candide wire frame model is automatically fitted on it. As the facial expression changes in subsequent frames, model gets deformed in shape. Difference between Candide grid node coordinates of first and last frame is used as training data. Only geometrical information is given to classifier, no texture information is given to classifier. The system is fully automatic.

Table I
BINARY TREE MC-SVM: CONFUSION MATRICES AND ACCURACY

Expression	*Happy*	*Surprise*	*Sad*	*Anger*	*Disgust*	*Fear*
Happy	69%	13%	0	0	0	10%
Surprise	8%	59%	0	0	0	0
Sad	0	0	55%	8%	0	10%
Anger	8	0	33%	84%	50%	10%
Disgust	0	0	0	0	37%	0
Fear	15%	28%	12%	8%	13%	70%

Table II
ONE VERSUS ONE MC-SVM: CONFUSION MATRICES AND ACCURACY

Expression	*Happy*	*Surprise*	*Sad*	*Anger*	*Disgust*	*Fear*
Happy	69%	0	11%	0	0	10%
Surprise	8%	88%	0	0	0	0
Sad	0	0	78%	0	0	10%
Anger	8	0	0	92%	15%	0
Disgust	0	6%	0	0	70%	0
Fear	15%	6%	11%	8%	15%	80%

Table III
BAYESIAN CLASSIFIER: CONFUSION MATRICES AND ACCURACY

Expression	*Happy*	*Surprise*	*Sad*	*Anger*	*Disgust*	*Fear*
Happy	64%	0	0	0	0	10%
Surprise	8%	88%	0	0	0	0
Sad	8%	0	78%	8%	12%	10%
Anger	0	0	11%	77%	20%	10%
Disgust	8%	0	11%	7%	68%	0
Fear	12%	12%	0	8%	0	70%

Table IV
COMPARISON OF THE METHODS

Scheme	***Accuracy***	***No. of classifiers/ Efforts in computation***
Binary SVM tree	61%	C -1 i.e 5 classifiers/ precomputation to form tree
One against one SVM	79.5%	$C(C-1)/2$ i.e. 15 classifiers
Bayesian classifier	74%	Computation of probabilities w.r.t each attribute

Manual fitting of Candide wire frame model on first frame of image sequence is not required. The method is applicable for color images also. It recognizes only six basic facial expressions. Nevertheless, this is unrealistic since it is not at all certain that all facial expressions able to be displayed on the face can be classified under these basic emotion categories. It should be modified for more number of expressions. We are working on modifying this frame work for more number of expressions Work is still going on to adopt this frame work to track features in faces with head rotations. Limitations of this method are rigorous training is required for constructing the update matrix, which is time consuming process.

REFERENCES

[1] B. Fasel and J. Luettin, "Automatic facial expression analysis: A survey," *Pattern Recognition*, vol. 36, no. 1, pp. 259–275, 2003.

[2] M. Pantic and L. J. M. Rothkrantz, "Automatic analysis of facial expressions: The state of the art," vol. 22, pp. 1424–1445, Dec. 2000.

[3] I. A. Essa and A. P. Pentland, "Coding, analysis, interpretation, and recognition of facial expressionsl," vol. 19, pp. 757–763, July 1997.

[4] S. Kimura and M. Yachida, "Facial expression recognition and its degree estimation," in *Proc. Computer Vision and Pattern Recognition*, pp. 295–300, 1997.

[5] J. F. Cohn, A. J. Ziochower, J. J. lien, and T. kanade, "Feature point tracking by optical flow discriminates subtle differences in facial expression," in *Proc. Int'l Conf. Automatic Face and Gesture Recognition*, pp. 396–401, 1998.

[6] M. Wang, Y. Iwai, and M. Yachindai, "Expression recognition from time-sequential facial images by use of expression change model," in *Proc. Int'l Conf. Automatic Face and Gesture Recognition*, pp. 324–329, 1998.

[7] M. Valstar, I. Patras, and M. Pantic, "Facial action unit recognition using temporal templates," in *Robot and Human Interactive Communication, 2004. ROMAN 2004. 13th IEEE International Workshop on*, pp. 253–258, 2004.

[8] M. J. Black and Y. Yacoob, "Recognizing facial expressions in image sequences using local parameterized models of image motion," in *Int'l J. Computer Vision*, vol. 25, pp. 23–48, 1997.

[9] S. Lajevardi and M. Lech, "Facial expression recognition from image sequences using optimized feature selection," in *Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference*, pp. 1–6, 2008.

[10] I. Kotsia and I. Pitas, "Facial expression recognition in image sequences using geometric deformation features and support vector machines," *IEEE Transactions on Image Processing*, vol. 16, pp. 172–187, Jan. 2007.

[11] J. Ahlberg, "Fast image warping for active models," Tech. Rep. Report No. LiTH-ISY-R-2355, Dept. of EE, Linkoping University, 2001.

[12] P. Viola and M. Jones, "Robust real-time object detection," tech. rep., February 2001.

[13] J. Y. Bouguet, "Pyramidal implementation of the lucas-kanade feature tracker," tech. rep., Intel Corporation, Microprocessor Research Labs, 1999.

[14] J. Ahlberg, "Candide-3 an updated parameterized face.," Tech. Rep. Report No. LiTH-ISY-R-2326, Dept. of EE, Linkoping University, 2001.

[15] J. Ahlberg, "An active model for facial feature tracking," *EURASIP Journal on Applied Signal processing*, pp. 566–571, 2001.

[16] R. Patil, V. Sahula, and A. S. Mandal, "Features classification using support vector machine for a facial expression recognition system," in *Springer Machine Vision and Applications.*, Submitted.

[17] D. Michie, D. Spiegelhalter, and C. C. Taylor, *Machine Learning, neural and statistical classification*. New York Ellis Horwoodl, 1994.

[18] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[19] L. Cheng, J. Zhang, J. Yang, and J. Ma, "An improved hierarchical multi-class support vector machine with binary tree architecture," in *International Conference on Internet Computing in Science and Engineering, 2008. ICICSE '08.*, pp. 106 –109, Jan. 2008.

[20] T. Kanade, J. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," in *Proc.IEEE Int. Conf. Face and Gesture Recognition*, pp. 46–53, March 2000.