Automated Facial Expression Recognition Using Local Transitional Pattern (LTP)

By
Tahseen Mohmmad

MASTER OF SCIENCE IN INFORMATION AND COMMUNICATION TECHNOLOGY

Institute of Information and Communication Technology
Bangladesh University of Engineering and Technology
March, 2013

This thesis titled, **"Automated Facial Expression Recognition Using Local Transitional Pattern"** submitted by Tahseen Mohammad, Roll No: M04053116, session 2005-2006 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Master of Science in Information and Communication Technology on the 2 March, 2013.

BOARD OF EXAMINERS

1. Dr. Md. Liakot Ali Professor IICT, BUET, Dhaka	Chairman
2. Dr. Md. Saiful Islam Professor and Director IICT, BUET, Dhaka	Member (Ex-Officio)
3. Dr. Md. Saiful Islam Professor and Director IICT, BUET, Dhaka	Member
3. Dr. M. Abdur Razzak Associate Professor School of Engineering & Computer Science IUB, Dhaka	Member (External)

CANDIDATE'S DECLARATION

It is hereby declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.

Tahseen Mohammad

To my parents Ruhul Amin and Rezaun Nahar

Contents

CHAPTER 2: Fundamentals of Facial Expression Recognition

CHAPTER 3: Implementation of LTP

LIST OF FIGURES

LIST OF TABLES

`

ACKNOWLEDGEMENTS

I will in this occasion, first and foremost express my profound gratitude to the Almighty Allah (SWT) for allowing me to my thesis work successfully. It has been a long time coming and I am grateful Allah had granted me the strength, patience and intellect required to achieve this.

I have had the help of a lot of people without whose contribution this work may ever have come to light. The first person to acknowledge should be my supervisor Dr. Md. Liakot Ali of IICT, BUET for his continued help over the course of this work. His patience, sincere guidance and encouragement had made this work possible. My sincere gratitude and thanks to him.

My thanks to all my teachers here at IICT for helping me prepare for this task. I also have had the pleasure of support from the cordial and always helpful staff of IICT over the years that have made my life at IICT that much easier.

I would also like to acknowledge the works of so many people upon which my own work rests. They had provided me with the inspiration for this work, the required understanding, and the tools necessary to complete it.

Finally, my gratitude must go to my family whose unconditional support had made life challenges easier to meet and reach for success. My special gratitude to Taskeed Jabid, my elder brother, whose work in the similar fields had been a great inspiration and help.

Abstract

Automatic facial expression recognition is a prominent and challenging research interest with usefulness in a variety of fields. It plays an important role in the fields of human computer interaction, data-driven animation etc. Success of most facial image analysis solutions depend on an effective facial feature representation. This thesis presents a novel appearance-based facial feature, the Local Transitional Pattern (LTP). LTP can extract robust facial feature from a face image that gives accurate and reliable recognition performance for expression recognition. The LTP operator applied on a pixel finds the monotonic intensity transition of neighboring pixels at different radii. The micro patterns thus found is enhanced with spatial information by tiling the image and taking histogram of each tile. The final feature vector is a collation of these histograms. This feature vector is then employed to classify expressions with well known machine learning method: Support Vector Machine (SVM). Cohn-Kanade expression database is used to conduct experiments comparing LTP descriptor's performance against other well known appearance based feature descriptors. It shows that LTP descriptor has higher accuracy than LBP and Gabor descriptors and it is also more robust against non monotonic illumination.

CHAPTER 1
Introduction

1.1 Introduction

Facial expression recognition means identifying the emotional state of a person by analyzing one’s facial feature. It has become an important aspect of facial analysis in recent years, just like facial recognition. It has gained a lot of application in the field of Human Computer Interaction (HCI), since facial expression remains one of the most natural and immediate means of communication for humans [1]. It can be highly useful in fields like clinical psychology, criminal investigations etc. It might not be as verbose or explicit a medium such as verbal communication; regardless presents a highly accurate and expressive indication of a person's emotional state and interests.

Facial expression has also come to focus due to the increased practicality of its usage. A whole new generation of devices has come to the market with cameras and significant computational resources to make expression recognition a practical prospect for everyday usage. Most of these devices are also portable making intuitive and intelligent interface a major attraction, further increasing the demand. These portable devices also come with unique challenges due to the constraints they have. The image capture quality limited by the smallish cameras, uncontrolled lighting and angles and the limit on computational capacity adds significant challenges to the problem [2]. It is true that the resources at hand now in such devices are magnitudes higher than just a few years ago but still poses challenges for solving complex computer vision problems.

This thesis is inspired both by the progressive importance of facial expression recognition in devices and the resource constraint challenge they have. Naturally this work focuses on developing a robust facial expression recognition method with low computational overhead.

1.2 Motivation and Contribution

A feature descriptor is what lies at the center of a classification system. Ideally a feature descriptor represents items in the problem space i.e., a facial image in this case, in a manner that is optimal for the desired classification purpose. A significant amount of work has been done on feature descriptors [3-6], but it still remains a significant challenge to solve [7].

A number of literatures have proposed a sequence of images generally by using optical flow analysis [8-11]. The main drawback of these methods is the computational overhead and the need for multiple consistent images thus limiting its real-time performance and robustness.

Still image based facial features are divided into geometric and appearance based feature descriptors. Geometric feature descriptors utilize various geometric information (like position, distance etc.) of different facial components of the face image. Geometric feature descriptors thus require an accurate and reliable identification of different facial components making them a difficult proposition in natural environments [12].

Factors like aging, hair growth, spectacles, scars or lighting, camera angles etc. can significantly affect any facial features. Any image processing approach that deals with the face as a whole has significant chance of being affected by the local changes [5]. These issues are significantly more challenging for geometric facial feature. There also exists algorithms for normalizing environmental impact of images somewhat to help with the accuracy at the cost of further computational overhead [13, 14] which is undesirable to us.

Appearance based feature descriptors models the face images by applying an image filter or filter banks on the whole face or some specific regions [9]. These methods can be further classified into two groups according to how the face is divided for the filters. They are also suitable to work without normalization [15].

Global feature descriptors take the facial image as a whole to generate the feature vector. Local feature descriptors on the other hand apply the filters after splitting the facial image in some predefined regions. Results from these regions are then combined to achieve the final feature vector [10].

One of the most successful appearance based approach for processing human face image is applying Gabor filter banks with different orientation and scale. Zhang et al. [16] is pioneer in this regard. Beside Gabor filter, Principal Component Analysis (PCA) and Independent Component Analysis (ICA) are other popular techniques. Donato et. al. [17] performed a comprehensive analysis of different techniques, including PCA, ICA, local feature analysis, and Gabor wavelet for facial recognition and demonstrated that the best performance can be achieved by ICA and Gabor wavelet.

Recently Ahonen et. al. [18] presented a novel appearance based method where the facial area is equally divided into small regions to extract local facial features. Local Binary Pattern (LBP) feature played the pioneer role for extracting local facial features. Following this method, a number of techniques for extracting facial features have been proposed. The LBP [2] method is computationally efficient and robust to monotonic illumination changes. However, it is sensitive to non-monotonic illumination variation and also shows poor performance in the presence of random noise [19]. This thesis proposes a novel technique to overcome these limitations.

1.3 Objectives

1. To develop a robust appearance based facial feature descriptor method Local Transitional Pattern (LTP) and implement it in software
2. To use the LTP feature descriptor to devise a facial image representation for facial expression recognition system and implement it.
3. To generate standard quality training/test dataset from available benchmark expression database and using our implementation extract facial feature for the dataset.
4. To use our facial feature representation with other techniques to compare recognition performance

1.4 Organization of the Thesis

The chapters 2 through 5 of this write up are organized as follows.

- Chapter 2 describes the background concepts necessary to understand the work presented in this thesis.

- Chapter 3 details the proposed feature descriptor and how the complete recognition system would along with other necessary requirement for the system.

- Chapter 4 describes the actual experimental process to evaluate our proposed feature descriptor.

- Chapter 5 analyzes the experimental result to draw conclusion on the feature descriptor and outline ideas for further work enhancing the descriptor.

CHAPTER 2

Fundamentals of Facial Expression Recognition

2.1 Introduction

Computer vision is the field working to replicate the human vision system both in its capacity to capture the world visually and more importantly its cognitive ability to analyze it to take decisions based on them. Understanding human faces is a significant sub field of computer vision. There are many aspects of a facial image that can be analyzed and classified. Facial expression analysis has a lot of things common with other facial image analysis. In this chapter we explore concepts that are both general to computer vision or facial image analysis or are unique to facial expression recognition.

2.2 Image Acquisition and Representation

The first step of any computer vision system must of course be the acquisition of the said external environment in the form of an image. Various kind of still and motion image capture devices i.e. camera are naturally used for that. Generally standardized image capture and storage system are used for this part. So the same systems that are used to procure photos or videos for multi media and entertainment generally suffice for computer vision purposes. Interestingly compared to the multimedia purpose, computer vision generally requires quite low end hardware and image capturing capability. Low resolution images are suitable and preferable in computer vision simply due to the cost of processing involved which in general is generally proportional to the dimension of the said image. There are systems that work on more sophisticated techniques dealing with color images, high resolution images and even 3D modeling of the environment though we will not focus on them here.

There are systems that work on a single still image to perform the analysis on; the other being systems that operates on series of images generally extracted from a video of an event. Where some computer vision problem i.e. motion detection inherently must work on a video or a sequence of images, problems such as facial expression analysis can be worked on both from a single image capturing a particular expression of a person or it can be worked on by analyzing the transition of a person's expression say, from neutral to anger in a series of images that capture that transition.

The video or multi image algorithms at the core essentially operates on still images treating each one as a discrete element, comparing one another and detecting or analyzing the changes and transitions between them.

The representation of images for computer vision from the capture and processing are generally different. Images can be stored from the capturing devices in a multitude of formats from the uncompressed bitmap to jpeg or png files. But most image processing algorithms essentially work on 8-bit grayscale images that encodes basic pattern of the image. It is this 8 bit image treated as a two dimension matrix of intensity value that is used for the actual processing or pre processing steps.

2.3 Face Detection and Extraction

We will now focus on the particular subject of facial image analysis. In real life environment images captured would generally contain just the face or it might not even contain a face. Before any processing can be done we must detect the presence of a face in a captured image. Once it's verified that the image do contain a face, all the faces from the image must be extracted from the image in a minimally bound rectangle. The figure below shows the detection and extraction in action.

Figure 2.1: Detecting and extracting a face

Face detection techniques are generally classified in two different groups. The first is the holistic approach that treats the whole face as a single element and the second is the analytic approach where characteristic facial elements are detected ,first from which the whole facial region is identified.

2.3.1 Holistic Face Model

The following section describes a selection of holistic approaches to face detection. A simple explanation of the process is explained along with references to the appropriate works in the following paragraphs.

Point Distribution Model (PDM) by Huang et. al. [20] represents the mean geometry of the human face. The approach involves applying the Canny edge detector first to find two symmetrical vertical edges to estimate the face position. The PDM is then fitted on the image.

The proposed system by Pantic and Rothkrantz [21] works on images of frontal and profile face view. First vertical and horizontal histogram analysis is used to find face boundaries. Then, face contour is obtained by thresholding the image with HSV (Hue, Saturation, Value) color space values.

2.3.2. Analytic Face Model

The analytic face model on the other hand recognizes critical region of the face, the most important generally being the eye due to its uniqueness in face texture. Generally the distances between the critical regions of the face are then used to find out the exact facial image area.

Kobayashi and Hara [22] had found the face position by iris localization. Their approach uses image captured in monochrome mode to find face brightness distribution which is then used for iris localization.

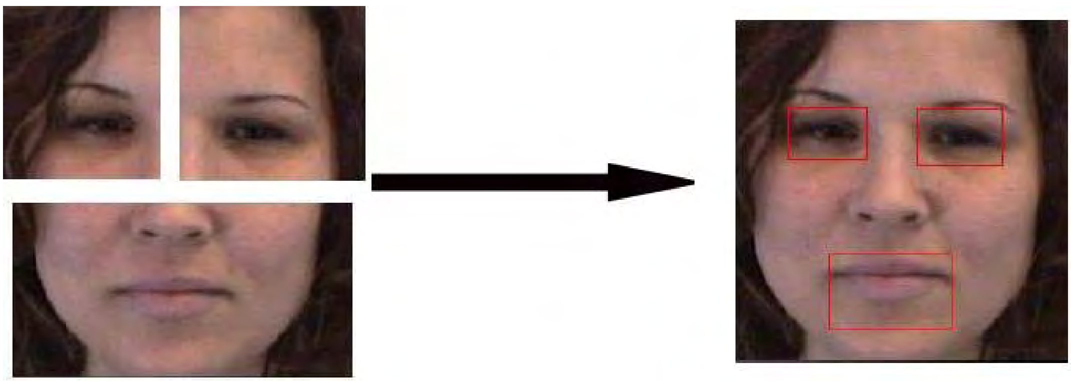

Figure 2.2: Analyzing face by detecting eye and mouth position

Kimura and Yachida [23] used a technique that processes input image with an integral projection algorithm to find position of eye and mouth corners by color and edge information. Face is represented with Potential Net model which is fitted by the position of eyes and mouth.

2.4 Facial Feature Descriptors

Facial images are processed to generate a feature descriptor that is then used by machine learning algorithm to reach at conclusions. There are two approaches to finding feature descriptors. One works on discrete images while the other depends on a sequence of images and their transitions as mentioned earlier.

A significant number of image sequence based approaches utilize optical flow analysis for finding image features. Subsequently different pattern recognition tools are used for recognizing the optical flow patterns associated with particular facial expression [6, 8-10]. One main drawback of this method is the computational burden associated with finding optical flow features. In addition this approach requires acquisition of multiple frames of images to recognize expressions and thus has limitations in real-time performance and robustness.

On the other hand facial feature extraction methods that work on still image are mostly generic working across problems like face or expression recognition. These methods are again generally divided into two categories namely geometric and appearance based feature.

Geometric features utilize various geometric information (like position, distance etc.) of different facial components, which are extracted from single face image to form a feature vector. This feature vector is used as the representation of face geometry and is used for classification. One of the main challenges of geometric feature is to develop a feature extraction method that works well regardless of variations in human subjects and environmental conditions. Moreover, this method needs an accurate and reliable identification of these facial features, which may be difficult to achieve in many real world and real time scenarios.

Appearance based facial image feature extraction techniques from still image is another popular methodology for expression recognition system. The appearance based system models the face images by applying an image filter or filter banks on the whole face or some specific regions of the face to extract changes in facial appearance [11]. This method can be further classified into two groups according to amount of image area it use to generate facial features – global feature and local feature. In global feature extraction process, the whole image is taken into account, but local feature considers only local regions within the given image [7]. One of the most successful appearance based approach for processing human face image is applying Gabor filter banks with different orientation and scale. Zhang et al. [16] is pioneer in this regard and they proposed a Gabor wavelet based facial expression coding system which can maintain a high degree of correlation with the human semantic ratings. Beside Gabor filter, Principal Component Analysis (PCA) is also a widely used method to extract features from face image. Independent Component Analysis (ICA) another subspace representation technique is also becoming popular in this arena. Donato et. al. [17] performed a comprehensive analysis of different techniques, including PCA, ICA, local feature analysis, and Gabor wavelet, to represent images of faces for facial action recognition and demonstrate that the best performance can be achieved by ICA and Gabor wavelet. Recently Ahonen et al [18] presented a novel appearance based method where the facial area is equally divided into small regions to extract local facial features.

Local Binary Pattern (LBP) feature plays the pioneer role for extracting local facial feature. Following this method, a number of techniques for extracting facial features have been proposed. The LBP method is computationally efficient and robust to monotonic illumination changes. However, it is sensitive to non-monotonic illumination variation and also shows poor performance in the presence of random noise [7]. The proposed method Local Transitional Pattern (LTP) described in this paper overcomes some of the drawbacks of Local Binary Pattern (LBP) and is more robust in recognizing facial expression.

2.5 Facial Expressions

There are six facial expressions [12] that we try to recognize in any standard system or seven once we include the neutral expression. The six primary expressions are anger, joy, surprise, disgust, sadness, fear.

Figure 2.3: Primary facial expressions

2.6 Classification – Template Matching

A template matching uses a template for each class of expression images during training to model that particular expression. During the training phase, the histograms of expression images in a given class are averaged to generate the template model *M*. For recognition, a dissimilarity measure is evaluated against each template and the class with the smallest dissimilarity value announces the match for the test expression. Chi square statistics (χ^2) is usually used as the dissimilarity measure as given below:

$$\chi^2 = \sum_{\tau} \frac{\left(S(\tau) - M(\tau)\right)^2}{S(\tau) + M(\tau)} \qquad (2.1)$$

where *S* is the test sample and *M* is the template LTP histogram feature. A weighted χ_w^2 statistics might be used to give more or less importance to particular regions such as eye, nose, and mouth regions. Literature shows that weights are set manually based on observations. In our case, we opted to use the χ^2 statistics for template matching.

$$\chi_w^2 = \sum_{i,\tau} w_i \frac{\left(S_i(\tau) - M_i(\tau)\right)^2}{S_i(\tau) + M_i(\tau)} \tag{2.2}$$

where, w_i is the weight of region R_i .

2.7 Classification – Support Vector Machine

A Support Vector Machine is a special kind of non probabilistic binary classification. It assigns labels to a set of input based on the model it generates during training. It works by finding a maximal hyper plane for the training data and use the hyper plane to classify the actual input in a binary classification according to where it falls against the hyperplane.

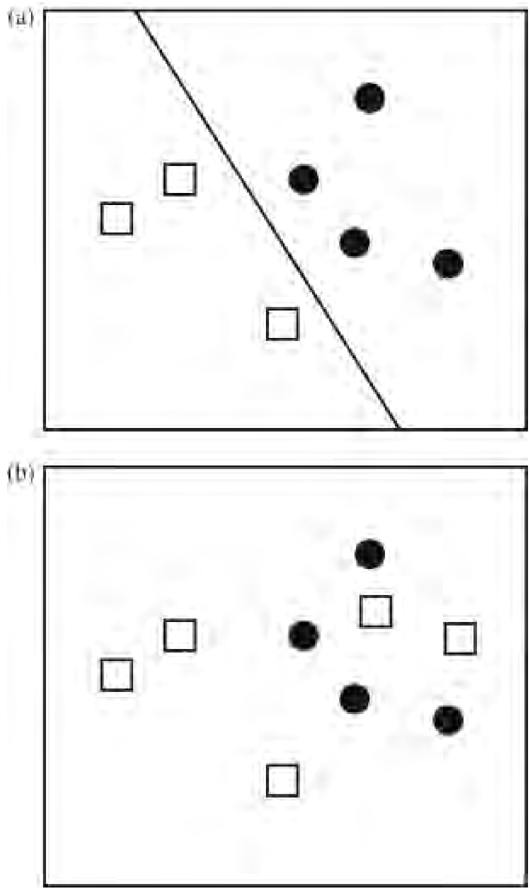

Figure 2.4: a) A linearly classifiable problem, b) A linearly non-classifiable problem

The inputs are represented as vectors hence can be considered as points in n-dimensional space. A set of points in n-dimension can either be linearly classifiable or not as shown in figure 2.4.

Most problems with complex multi dimensional input vectors are rarely linearly separable at least as they are. SVM's primary strength lies in how it handles linearly non-classifiable problem. A problem where the input dimension is M might not be solvable in dimension M. But if the input is transformed in some dimension N where N < M, it might become linearly classifiable. The required dimension N might be significantly high to make the transformation and subsequent search for a maximal hyperplane extremely computationally complex. SVM uses a kernel function that essentially performs the computation in the input space which through the use of kernel function actually is able to do the classification in a much higher dimensional space N where the problem is linearly classifiable [24].

The following figure 2.5 simulates how applying a kernel function can effectively transform an input space into a feature space (the higher dimensional space) and transform a linearly non-separable problem into a separable one.

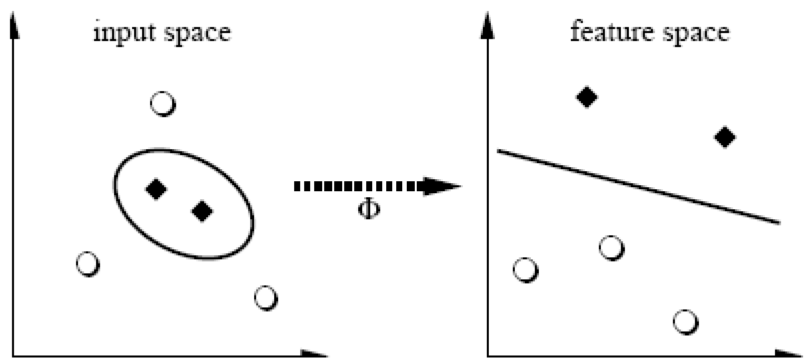

Figure 2.5: Feature space mapping in Support Vector Machine

2.8 Multi-class SVMs

In this work, six expression classes are to be recognized, which is why a multi-class classifier is needed. Multi-class SVMs handle this problem by combining several binary SVMs, using either one-versus-one or one-versus-all as training strategy. In this work, one-versus-one strategy is utilized, where for training purposes one class is considered positive and one other class negative. To get a classification result, a voting strategy is used, where for all pairs of classes the current feature vector is assigned to one of the two classes and finally, the class that receives most votes is considered the correct class. An illustration of one-versus-one multi-class SVMs is displayed in the following figure, figure 2.6.

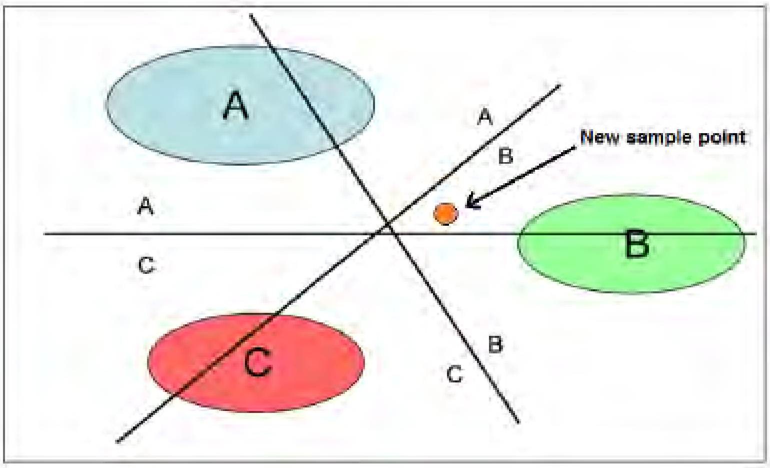

Figure 2.6: Example of a multi-class SVM.

In the above figure for each pair of classes, a separating hyperplane is learned. To assign a new sample to a class, it is classified by all pairs of classes, the votes/wins are counted and the sample is assigned to the class with most votes/wins. The one-versus-one classification in this example happens as follows: the first pair of classes is (A, B), the new sample lies on the 'B-side' of the separating hyperplane and therefore B gets one vote. The second pair of classes is (A, C) and the new sample is classified as A. The last pair is (B, C) classifying the sample as B. Summing up, A has one vote, B has two votes, C has zero votes; therefore, the new sample is classified as B.

2.9 LIBSVM

LIBSVM is a well known SVM classifier that is publicly available and is used in our experiment. We show a list of LIBSVM parameter to help us explain its featuers.

-s svm_type : set type of SVM (default 0)

0 -- C-SVC

1 -- nu-SVC

2 -- one-class SVM

3 -- epsilon-SVR

4 -- nu-SVR

-t kernel_type : set type of kernel function (default 2)

0 -- linear: u'*v

1 -- polynomial: (gamma*u'*v + coef0)^degree

2 -- radial basis function: exp(-gamma*|u-v|^2)

3 -- sigmoid: tanh(gamma*u'*v + coef0)

-d degree : set degree in kernel function (default 3)

-g gamma : set gamma in kernel function (default 1/k)

-r coef0 : set coef0 in kernel function (default 0)

-c cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)

-n nu : set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)

-p epsilon : set the epsilon in loss function of epsilon-SVR (default 0.1)

-m cachesize : set cache memory size in MB (default 100)

-e epsilon : set tolerance of termination criterion (default 0.001)

-h shrinking: whether to use the shrinking heuristics, 0 or 1 (default 1)

-b probability_estimates: whether to train a SVC or SVR model for probability estimates, 0 or 1 (default 0)

-wi weight: set the parameter C of class i to weight*C, for C-SVC (default 1)

35

The k in the -g option means the number of attributes or features in the input data. option -v randomly splits the data into n parts and calculates cross validation accuracy/mean squared error on them. This feature allows for very convenient performance analysis and makes LIBSVM such an effective tool to use.

We can run with the default parameters left unchanged and observe basic classification performance. But LIBSVM also gives us the ability to choose different types of kernel which has significant performance but also allows changing of different core optimization parameter of SVM to fine tune the algorithm for the problem at hand.

2.9.1 svm-train and svm-predict

We use two programs svm-train and svm-predict to performa the training and

Usage: svm-train [options] training_set_file [model_file]

Usage: svm-predict [options] test_filemodel_fileoutput_file

We run the following commands. To train a classifier (on the training set) using a RBF kernel (default), and use it for prediction (classification) on the test set:

svm-train.exe -c 10   FeatureVector-7.train.txt   FeatureVector-7.model svm-predict.exe FeatureVector-7.test.txt FeatureVector-7.model FeatureVector-7.output

we change the –c parameter from 0.01 to 10000 (increase by a factor of 10 each time) and study the effect.

We change the –g (gamma) parameter.

If the training set is unbalanced i.e. no. of data in different classes is not same then we try the "–w1 weight" and "–w-1 weight" options to adjust the penalty for misclassification.

CHAPTER 3
Implementation of LTP

3.1 Introduction

Local Transitional Pattern is inspired by the highly successful Local Binary Pattern feature descriptor. We show how LBP feature descriptors are generated and where in the process we make changes in LTP to improve upon accuracy and robustness.

3.2 Local Binary Pattern - LBP

The LBP operator is a well known gray-scale invariant texture primitive. It has gained significant popularity for facial image analysis in different fields for both its computational simplicity and effectiveness. It works by comparing an image pixel against its P-neighbor’s intensity values and converting the result into a pattern code by equation (3.1).

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_p - g_c)2^p, \quad s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{3.1}$$

where g_c denotes the gray value of the center pixel (x_c,y_c) and g_p corresponds to the gray values of P pixels on the circumference of a circle with radius R. The values of neighbors that do not fall exactly on pixels are estimated by bilinear interpolation. In practice, (1) means that the signs of the differential pattern in the neighborhood around a pixel are interpreted as a P-bit binary number, resulting in 2^P possible distinct pattern to represent the texture information for the center pixel g_c.

The process of generating this P-bit binary number is shown in the Figure 3.1.

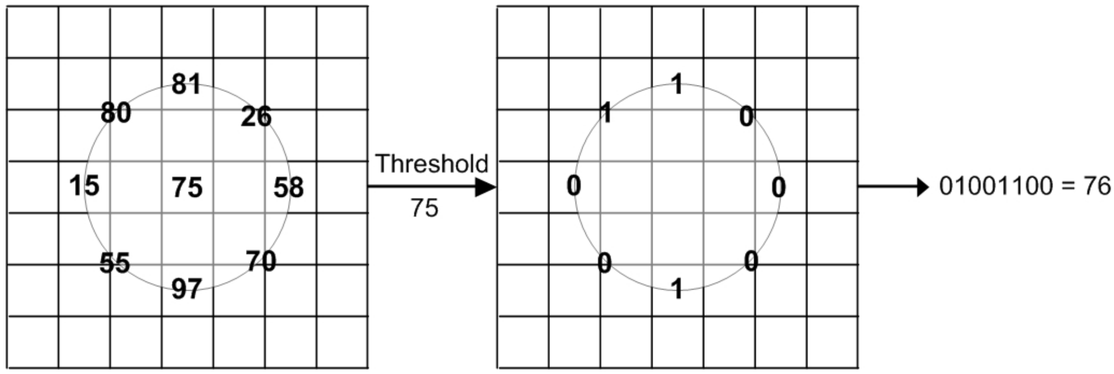

Figure 3.1: The basic LBP operator

One variation of the original LBP, known as uniform LBP, is proposed from the observation that certain LBPs appear more frequently in a significant image area. These patterns are considered uniform because they contain very few transitions from 0 to 1 or 1 to 0 in a circular bit sequence. For example, the patterns 00000000 and 11111111 have zero transitions, 00011000 have two transitions, and 10001101 have four transitions. Shan and others [12] used this variant of the LBP, which has at most two transitions for their facial expression recognition task. Though the LBP shows good recognition accuracy in a constraint environment, it is sensitive to random noise and non-monotonic illumination variation. It is considered that the uniform LBP are the represents edge patterns which results in the better performance.

LBP operator tries to encode the micro-level information of edges, spots and other local features in an image using a simple threshold function among local neighbors as shown in equation (3.1). It could be beneficial to account pixels that are not immediate neighbor but somewhat farther from the center pixel in consideration of even a local pattern given how small a portion a single pixel radius covers. There are variants of LBP that work by changing the radius of the operator. But varying the radius actually excludes the inner most radius if naïve LBP with a different radius is applied. While it is useful to consider pixels farther from the center, it is also important to note that the closest pixels are the most important one still. So what we need an operator that considers the surrounding pixels of multiple radii for a pattern.

3.3 Local Transitional Pattern - LTP

The proposed new operator LTP tries to solve the issue of generating a more informative micropattern. We do so by considering neighbor pixels of multiple radii in a manner where all the pixels are useful in deciding the pattern. We also try using a system that like uniform LBP tries to distinguish edges that would increase useful information content.

It encodes directional micro patterns by considering monotonic intensity changes in one direction and encodes that in a similar fashion to LBP to generate a binary pattern for the center pixel. Given an image I(x,y), the local transitional pattern (LTP) can be computed by

$$LMP_{P,R1,R2}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_{p1} - g_c) \wedge s(g_{p2} - g_{p1}) * 2^p, \; s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \quad (3.2)$$

where g_c denotes the intensity value of the center pixel (x_c, y_c); g_{p1} and g_{p2} corresponds to the intensity values of P equally spaced pixels on the circumference of a circle with radius R_1 and R_2 respectively.

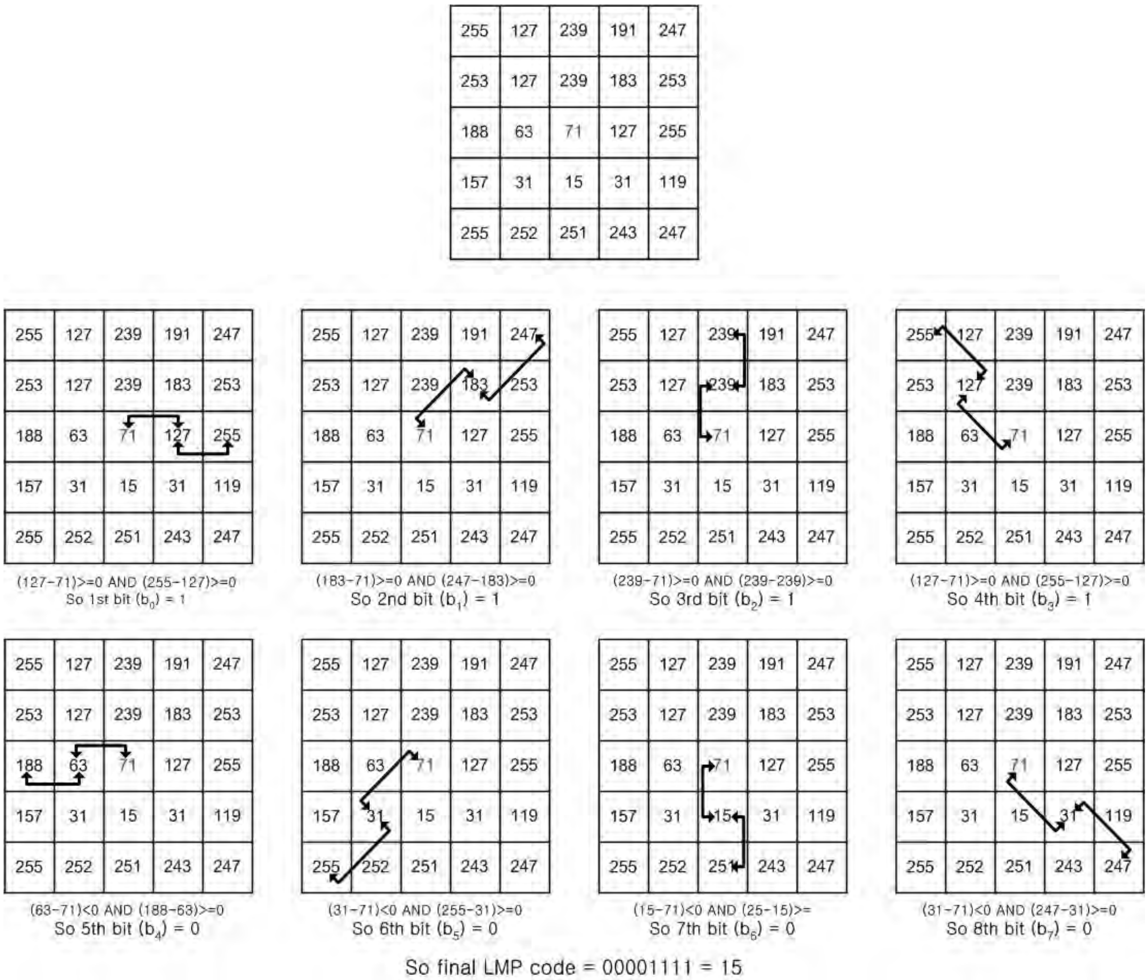

Figure 3.2: Example to obtain the LTP micro-pattern code

The code generation process for a particular pixel is explained with figure 3.2. To describe it in general term the center pixel is considered against neighboring pixels in a directional manner. In each direction unlike LBP we consider not one but two consecutive pixels and compare both of them. We also consider the neighbors for monotonic changes which ensure that the pixel intensity change is more likely to be part of pattern and not a random noise. So the primary goal is finding uniform changes in pixel intensity in particular directions which would natural suggest more useful and important patterns in the given image.

Yet like LBP, LTP only encodes the information of a single pixel against the neighboring pixel. We need some way to combine the pattern of all pixels in the image to generate a useful feature vector that represents the image as a whole which we describe next.

3.4 LTP Histogram

A face image must go through multiple stages before a complete LTP based facial feature can be generated for the face. LTP operator applied on the raw face image using equation (3.2) gives a one to one mapping for each pixel. Rather than using the LTP code directly, we use the histogram of the LTP code to generate our feature vector which is a widely used technique in image analysis. The following equation (3.3) shows a LTP histogram of an image, where the input images I of size M × N is represented by a LTP histogram H using (3). The resultant histogram H is the LTP descriptor of that image.

$$H = \sum_{r=1}^{M}\sum_{c=1}^{N} f\left\{I(r,c)=i\right\}, i = 0,1,\ldots,n-1, \qquad f(A) = \begin{cases} 1 & A\,is\,true \\ 0 & A\,is\ false \end{cases} \tag{3.3}$$

where, n is the number of different pattern produce by LTP operator.

3.5 Concatenated LTP Histogram

However the LTP code itself represents very small micro pattern and this global histogram of an image only gives an idea of the image as a whole. But expressions don’t affect a face as a whole. Changes at different portions are important to identify and compare individually. So it is important to have localized feature representing spatial relationship for good facial analysis [25]. This is achieved by modifying the basic histogram, where the image is divided into g regions R_0, $R_{1,}$ …, R_{g-1} as shown in figure 3.3. A LTP histogram for each of the regions is generated as shown already.

So we get the LTP_i histogram built for each region R_i. The final feature vector is found by collating all the regional LTP_i histogram in sequence yielding the descriptor vector of size gxn. The resulting feature vectors give us information of the local patterns in the form of the regional histograms while giving us a global representation of the image too.

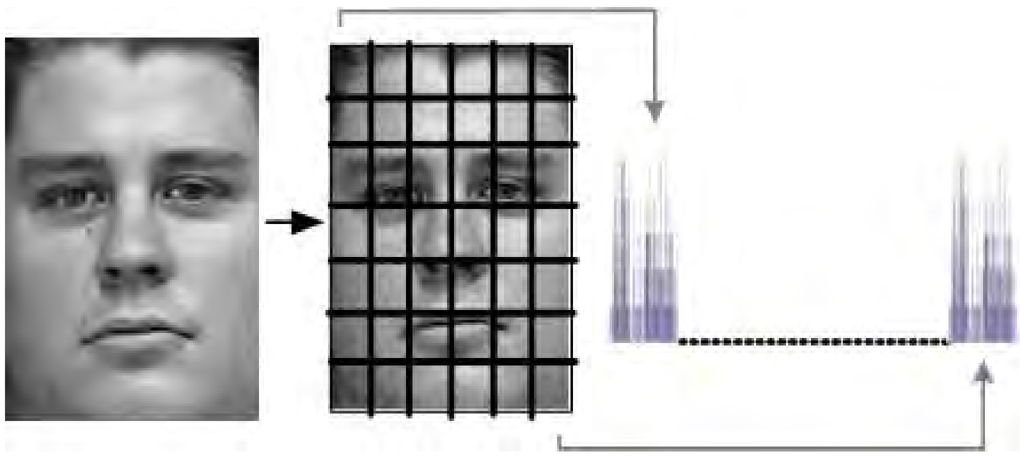

Figure 3.3: Left: Original face image, Right: LTP image

The figure 2.5 shows a facial expression image and how its divided into regions before making a concatenated histogram of the image.

3.6 Data Sets

Most facial expression recognition systems attempt to recognize a set of prototypic emotional expressions like anger, disgust, fear, joy, sadness, and surprise [12]. This 6-class expression set can also be extended as a 7-class expression set by including a neutral expression. In this work, our effort is devoted to recognize both 6-class and 7-class prototypic expressions. The two popular databases for performance evaluation of facial expression recognition system are the CK facial expression database [26] and the JAFFE database [27].

The CK database consists of facial expression images from 100 university students, who at the time of the inclusion were between 18 to 30 years old. 65% of the subjects were female, 15% were African-American and 3% were Asian or Latino. Subjects were instructed to perform a series of facial expression displays starting from neutral or nearly neutral to one of six target prototype emotions. Image sequences from neutral to target display were digitized into to 640x480 or 640x690 pixel arrays of gray-scale frame.

The JAFEE database contains only 213 images of female facial expression from a total of 10 subjects. Each image has a resolution of 256x256 pixels with almost the same number of images for each category of expression. The head in each image is usually in the frontal pose, and the subjects hair was tied back to expose all the expressive zones of her face. Tungsten lights were positioned to create an even illumination on the face.

In our setup we used the CK database for the evaluation of our LTP based facial feature's performance. We selected 408 image sequences from 96 subjects, each of which was labeled as one of six basic emotions. For 6-class prototypic expression recognition, the three most expressive image frames were taken from each sequence that resulted in a total of 1,224 expression images. In order to build the neutral expression set, the first frame (neutral expression) of all 408 sequences was selected to make the 7-class expression dataset consisting of 1,632 expression images.

3.7 Image Preparation

After the image set was created, they were cropped from the original image using the position of two eyes and resized into 150x110 pixels. The provided ground-truth of eye position data of CK database were used for the cropping. For other image databases, an existing eye detection technique with good accuracy was used [28]. The image has been cropped and positioned based on the distance of eye centers as identified.

Once the eyes are detected we use it to extract the relevant part of a face for recognition. Most images would contain a significant portion of background that has no bearing on the expression recognition. Moreover a significant part of the human itself wouldn't be useful for recognition purposes either. The hair, ear and any part of a human below the chin would have no bearing in the expression since the facial muscles do not control them. As a result we would generally need to consider only a portion of the image, in fact we need to reject or discard the rest of the image. This is a very important step in preparing the image.

In this step we center the eyes horizontally with a distance of D and the final width of the image is 2D. This follows that a 0.5D distance acts as boundaries around both eyes. The height of the image is 2.7D with 2D distance between the eye level and the bottom and 0.7D between eye level and top of the head. This is shown in the figure 3.5.

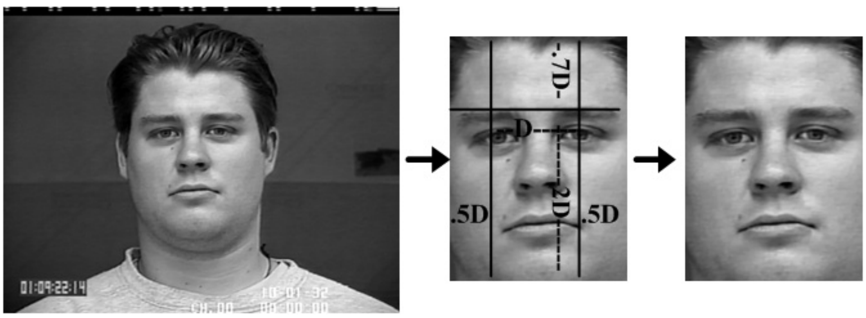

Figure 3.4: Cropping of the original face by eye position

As can be seen from figure 3.4 that cropped thusly we are only considering the bare minimum of the subject’s face that would contain the facial expression, removing as much as redundant data as possible.

There was no farther alignment i.e., the alignment of the mouth was not perform. No correction for illumination factors were done either, since LTP is generally more robust to illumination changes.

The images thus generated are then used in generating the LTP feature vector for the dataset. These generated facial features are then used for both training and recognition at the classification phase.

CHAPTER 4

Results and Discussion

4.1 Introduction

This chapter describes the results achieved from the proposed technique and the highlights on the result. Here, each 150x110pixel facial image is divided into 100 (10x10) regions. To support this parameter values, we also provide an empirical analysis in determining the optimal parameter values. We then explore the results against competing facial features and evaluate its performance. Finally we present the achieved expression recognition rate at low resolution images.

4.2 Basic Performance Analysis

In this section we evaluate different configurations possible for using the LTP feature descriptor for facial expression recognition. We use it to both determine optimal configuration for a LTP based facial expression recognition system but also show LTP's robustness in most scenarios. All descriptors are evaluated against the CK facial expression database [26].

We start with template matching, it being the simplest of classification algorithm. Basic template matching with LTP descriptor achieved a recognition rate of 89.2% and 85.6% for 6-class and 7-class expression recognition problem. In Table 4.1 a comparative results are provided in contrast to LBP and Gabor features and it shows that LTP has significant accuracy advantage over the other two feature descriptor.

Table 4.1: Recognition performance with template matching

Feature Descriptor	6-class Recognition (%)	7-class Recognition (%)
Gabor [29]	83.7 ± 4.5	78.9 ± 4.8
LBP [12]	84.5 ± 5.2	79.1 ± 4.6
LTP	89.2 ± 4.1	85.6 ± 5.8

The next of performance comparison shows LTP feature descriptor's performance under SVM classifier. We perform the classifications using SVM with different kernels. The comparative performances achieved with SVM based on different features are shown in Table 4.2 and 4.3 for both six and seven class classification. It is observed that, LTP representation performs more stably and robustly than both LBP and Gabor representations once again.

Table 4.2: 6-class expression recognition: SVM with different kernels

	Kernels		
Feature Descriptor	Linear (%)	Polynomial (%)	RBF (%)
Gabor [29]	89.4 ± 3.0	89.4 ± 3.0	89.8 ± 3.1
LBP [12]	91.5 ± 3.1	91.5 ± 3.1	92.6 ± 2.9
LTP	95.2 ± 1.2	95.2 ± 1.2	96.7 ± 0.9

Table 4.3: 7-class expression recognition: SVM with different kernels

	Kernels		
Feature Descriptor	Linear (%)	Polynomial (%)	RBF (%)
Gabor [29]	86.6 ± 4.1	86.6 ± 4.1	86.8 ± 3.6
LBP [12]	88.1 ± 3.8	88.1 ± 3.8	88.9 ± 3.5
LTP	92.5 ± 1.8	92.5 ± 1.8	93.1 ± 1.6

So far the performance comparison has shown the overall performance of LTP feature vector over a wide variety of face images comprising of all expressions. The following set of comparative studies shows LTP feature descriptor's accuracy on each of the separate expression to give us a better idea of its performance characteristic.

The confusion matrix for 6-class and 7-class expression recognition with template matching is given in Table 4.4 and 4.5, respectively. It is observed that with the inclusion of neutral expression in the 7-class recognition problem, the accuracy of other six expressions get lower as more facial expression samples are confused as neutral expression.

Table 4.4: Confusion matrix of 6-class expression by template matching

	Anger (%)	Disgust (%)	Fear (%)	Joy (%)	Sad (%)	Surprise (%)
Anger	**81.2**	8.7	0.0	0.5	3.4	6.3
Disgust	10.6	**84.1**	1.5	1.5	2.3	0.0

Fear	12.3	3.6	**67.2**	6.7	5.6	4.6
Joy	4.2	4.2	1.9	**87.5**	0.0	2.3
Sad	25.3	0.5	1.1	0.0	**68.3**	4.8
Surprise	8.3	0.0	3.8	0.0	1.3	**86.7**

Table 4.5: Confusion matrix of 7-class expression using template matching

	Anger (%)	Disgust (%)	Fear (%)	Joy (%)	Sad (%)	Surprise (%)	Neutral (%)
Anger	**70.0**	8.7	0.0	0.8	0.8	5.5	14.3
Disgust	3.8	**82.2**	1.3	0.6	2.2	0.0	9.8
Fear	9.7	3.4	**66.9**	6.3	2.9	4.0	6.9
Joy	2.4	1.0	1.7	**85.1**	0.0	2.7	7.1
Sad	10.0	0.4	1.7	0.0	**65.2**	3.3	19.5
Surprise	2.0	0.0	1.4	4.0	0.7	**84.8**	6.1
Neutral	12.2	0.0	2.0	0.5	0.5	4.0	**80.7**

Tables 4.4 and 4.5 shows the varied effectiveness of LTP in regards to different facial expressions. It shows how the addition of neutral expression has a non linear effect on different expression. The tables also show that LTP is highly effective in recognizing Disgust, Joy, Surprise and Neutral expressions.

4.3 Determining Optimal Parameters

This section covers the experimental results of tuning some of the more common parameters that might affect performance and the selection of optimal parameters.

We have already explained how regional histograms allow us to preserve local patterns and improve accuracy. It is only natural that the number of regions and their boundaries would affect recognition performance. There can be many possible ways to divide an image into regions including different sizes of regions in a single image.

We use for the purpose of this work only a fix size of region to split the image. But we need to find the optimal number of regions needed to find the optimal number of regions on the facial images which essentially also determines the dimension of each region and where the region boundaries would split the image. The commonly used numbers of regions are 3x3, 5x5, 7x6, 7x7, 9x8, 10x10 etc. In our experiment, we evaluate for four different cases: 3x3, 5x5, 10x10 and 15x11. Table 4.6 lists the effect of different number of regions on the recognition performance. With small number of regions, the expression recognition rate is low (below 83%). As we increase the number of regions, the recognition performance starts to increase as the descriptor feature starts to incorporate more local and spatial relationship information. But at a certain point, too many regions incorporate unnecessary local information that might again degrade the performance. From our observation shown in table 4.6, 7x6 numbers of regions gives a good tradeoff between recognition performance and feature vector length.

Table 4.6: Recognition performance by number of regions using LTP + SVM (RBF)

	6-Class Expression (%)	7-Class Expression (%)
g = 3x3	83.1	80.3
g = 5x5	95.6	90.4
g = 10x10	96.2	93.1
g = 15x11	94.7	91.1

Therefore, we conclude that g=10x10 are the optimal parameter values in the proposed LTP descriptor for representing facial expression images.

We have already shown in the previous section that SVM performs significantly better than template matching for classification. But SVM allows for using of different kernel functions. Kernel functions have significant performance impact in SVM based classification and determining optimal kernel function and parameter is a significant step in any SVM based classification experiment.

Table 4.7: Confusion matrix of 6-class expression using LTP + SVM (RBF)

	Anger (%)	Disgust(%)	Fear(%)	Joy %)	Sad (%)	Surprise (%)
Anger	**93.8**	2.5	0.5	0.0	2.8	0.5
Disgust	0.0	**97.2**	2.8	0.0	0.0	0.0
Fear	1.0	0.0	**96.5**	2.0	0.0	0.5
Joy	0.0	0.5	0.5	**98.0**	1.0	0.0
Sad	1.6	0.0	0.0	0.0	**97.5**	1.0
Surprise	0.0	0.0	2.0	0.0	0.0	**98.0**

Table 4.8: Confusion matrix of 7-class expression using LTP + SVM (RBF)

	Anger	Disgust	Fear	Joy (%)	Sad	Surprise	Neutral

	(%)	(%)	(%)		(%)	(%)	(%)
Anger	**79.4**	0.5	0.5	0.0	2.3	0.9	16.5
Disgust	2.2	**92.8**	0.7	0.0	0.0	0.0	4.4
Fear	1.5	0.0	**93.1**	0.0	0.0	0.5	4.9
Joy	0.0	0.0	0.4	**99.6**	0.0	0.0	0.0
Sad	1.6	0.5	0.0	0.0	**92.0**	0.0	5.9
Surprise	0.8	0.0	0.0	0.0	0.0	**98.7**	0.4
Neutral	6.0	0.5	0.5	0.0	2.1	0.5	**90.3**

Tables 4.7 and 4.8 show the confusion matrix using SVM with RBF kernel. Comparing these results with template matching we find that SVM with RBF performs significantly better across the board against template matching.

4.4 Evaluation at different resolution

In environments like smart meeting, visual surveillance, old-home monitoring, only low resolution video input is available Deriving action units from such facial images are critical problems. In this section, we explore the recognition performance on low resolution images with LTP descriptor. Four different resolutions (150x110, 75x55, 48x36, 37x27) of face images based on Cohn-Kanade dataset were studied. The low resolution images were formed by down-sampling the original images. All face images were divided into 100 (10x10) number of regions for building the LTP descriptor. To compare with the methods based on LBP and Gabor features, we conducted similar experiments on the 6-class prototypic expression recognition using SVM with RBF kernel. Table 4.9 lists the recognition results with LBP, Gabor and the proposed LTP feature. The proposed LTP based facial representation has obtained improved recognition performance than existing methods and also have lower feature dimension.

Table 4.9: Low resolution images using LTP + SVM (RBF)

Resolution	150x110	75x55	48x36	32x27
Feature				
Gabor [29]	89.8 ± 3.1	89.2 ± 3.0	86.4 ± 3.3	83.0 ± 4.3
LBP [12]	92.6 ± 2.9	89.9 ± 3.1	87.3 ± 3.4	84.3 ± 4.1
LTP	96.7 ± 0.9	95.6 ± 1.7	93.6 ± 2.0	90.3 ± 2.2

With low resolution images, it is difficult to extract geometric feature, therefore, appearance based methods seems to be a good alternative. Our analysis with LTP feature demonstrates that the proposed descriptor performs robustly and stably over a range of expressions, even with low resolution facial images.

CHAPTER 5
Conclusion

5.1 Summary

This thesis started out with the goal of exploring the field of facial expression recognition, its challenges, presenting a new and robust facial feature vector and devising its performance in facial expression recognition.

In chapter 1, we have shown that facial expression recognition has become a vital area of research with important practical consequence. We have also explored the numerous challenges of facial image analysis in general and facial expression recognition in particular. We have also explained our motivation and proposed a new facial feature descriptor to address some of the problem of the existing feature descriptors.

In chapter 2, we have explored the fields of facial image and expression image and the different background concepts and techniques required for the understanding of this research work.

In chapter 3, we presented our new proposed feature descriptor and the detail steps necessary to make a proper facial feature vector using the descriptor. We also explained the facial image dataset that we have used and the required steps for processing it for the experiments to follow.

In chapter 4, we have detailed the experimental process in determining the performance characteristic of our proposed feature descriptor and feature vector. We explored how the different optimal parameters were determined and their impact on the final performance. We have also shown the feature descriptor's robustness in low resolution images.

Finally, the conclusion we can draw from the experimental results in chapter 4, that we have achieved the result that we set out as the goal for this research. We have proposed a new appearance based feature descriptor and have shown that it is both effective and efficient descriptor. We have also shown that it is more robust than some of the most well known existing feature descriptor. The discriminative power of the LTP descriptor mainly lies in the successful integration of multiple radius of neighboring pixel in the micro patterns and with spatially encoded histogram.

Moreover, the simplicity of the system accords its use in real world consumer devices. Its robustness also ensures that it can be used quite successfully in terms of accuracy.

5.2 Future Work

LTP feature descriptor in its current form works fairly well. But this research has not explored some of the well known optimization techniques that are generally found useful in facial feature descriptor and representation. We suggest exploring the following ideas to further improve the performance characteristic of LTP.

1. The complete set of features generated by the concatenated histogram has been used. It has been shown in other similar feature descriptor that dimensionality reduction actually helps with accuracy since non critical information are removed and only significant features are compared against one another [16, 19]. AdaBoost and PCA is two technique that can be used for that purpose

2. We have divided the image into uniform size regions. Regions that are variable in size and are carefully placed so as to segment along critical features of the face might significantly boost the performance. It is an important possibility to explore.

3. We have explored the performance of template matching which is found to be significantly less performing than SVM. But we can also try weighted template matching for classification and analyze its impact on performance.

REFERENCES

1. Tian,Y. L., Brown, L., Hampapur, A., Pankanti, S., Senior, A. W., and Bolle, R. M., Real World Real-time Automatic Recognition of Facial Expressions," *Proc. IEEE workshop on Performance Evaluation of Tracking and Surveillance*, Australia, 2003.

2. Shan, C., Gong, S., and McOwan, P.W., "Robust Facial Expression Recognition Using Local Binary Patterns," Proc. IEEE Int'l Conf. Image Processing, pp. 914-917, 2005.

3. Hwang, M. C., Ha, L. T., Kim, N. H., and Park, C. S., "Person identification system for future digital TV with intelligence," *IEEE Transactions on Consumer Electronics*, vol. 53, no. 1, pp. 218–226, Feb. 2007.

4. Pantic, M., and Rothkrantz, L. J. M., "Automatic analysis of facial expressions: the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1424–1445, Dec 2000.

5. Fasel, B., and Luettin, J., "Automatic facial expression analysis: a survey," *Pattern Recognition*, vol. 36, pp. 259–275, Jan 2003.

6. Zhao, G., Pietikainen, M., "Boosted multi-resolution spatiotemporal descriptors for facial expression recognition," *Pattern Recognition Letters*, vol. 30, no. 12. pp. 1117-1127, Sep. 2009.

7. Jabid, T., Kabir, M. H., and Chae, O. S., "Local Directional Pattern (LDP) for Face Recognition," *IEEE International Conference on Consumer Electronics*, January 2010.

8. Tian, Y., Kanade, T., and Cohn, J.F., “Facial Expression Analysis,” *Handbook of Face Recognition*, Springer, Oct 2003.

9. Goneid, A., el Kaliouby, R., Facial feature analysis of spontaneous facial expression. In: Proceedings of the 10th International AI Applications Conference, 2002.

10. Liu, X., Chen, T., Vijaya Kumar, B.V.K., on modeling variations for face authentication, In: Proceeding of the International Conference on Automatic Face and Gesture Recognition, 2002.

11. Lien, J.J., Kanade, T., Cohn, J.F., Li, C.C., Detection, tracking and classification of action units in facial expression. Journal of Robotics and Autonomous Systems 31 (3), 131–146, 2000.

12. Shan, C., Gong, S., and McOwan, P. W., “Facial expression recognition based on Local Binary Patterns: A comprehensive study,” *Image and Vision Computing*, vol. 27, pp. 803–816, 2009.

13. Essa, I., and Pentland, A., "Coding, Analysis Interpretation,Recognition of Facial Expressions", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.19, No. 7, p. 757-763, July 1997.

14. Pentland, A. P., Moghaddam, B., Starner, T., and Turk, M. A., “View-based and modular eigenspaces for face recognition”, in Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition, Seattle, WA, pp. 84-91, 1994.

15. M.J. Black and Y. Yacoob, Recognizing facial expressions in image sequences using local parameterized models of image motion, Int. Journal of Computer Vision, 25(1), , 23-48, 1997.

16. Zhang, Z., Lyons, M.J., Schuster, M., Akamatsu, S., “Comparison between geometry-based and Gabor-wavelets-based facial expression recognition using multi-layer perceptron,” *Proc. IEEE International Conference on Automatic Face & Gesture Recognition (FG)*, pp. 454-459, 1998.

17. Donato, G., Bartlett, M. S., Hagar, J. C., Ekman, P., and Sejnowski, T. J., “Classifying Facial Actions,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 974-989, 1999.

18. Hadid, A., Pietikainen, M., and Ahonen, T., “A discriminative feature space for detecting and recognizing faces,” in Proceedings of the Computer Vision and Pattern Recognition, vol. 2, pp. 797–804, 2004.

19. Jabid, T., Kabir, M. H., Chae, O., “Facial Expression Recognition using Local Directional Pattern (LDP), in Proceedings of the International Conference on Image Processing (ICIP), pp. 1605-1608, September, 2010.

20. Huang, C.L., and Huang, Y.M., "Facial Expression Recognition Using Model-Based Feature Extraction and Action Parameters Classification," J. Visual Comm. and Image Representation, Vol. 8, No. 3, p. 278-290, 1997.

21. Pantic, M., and Rothkrantz, L., "Expert System for Automatic Analysis of Facial Expression", Image and Vision Computing J., Vol. 18, No. 11, p. 881-905, 2000.

22. Kobayashi, H., and Hara, F., "Facial Interaction between Animated 3D Face Robot and Human Beings," Proc. Int'l Conf. Systems, Man, Cybernetics, p. 3,732-3,737, 1997.

23. Kimura, S. and Yachida, M., "Facial Expression Recognition and Its Degree Estimation", Proc. Computer Vision and Pattern Recognition, p. 295-300, 1997.

24. Cortes, C., and Vapnik, V., “Support-vector networks,” *Machine Learning*, vol. 20 pp. 273–297, Nov 1995.

25. Gundimada, S., and Asari, V. K., “Facial Recognition Using Multisensor Images Based on Localized Kernel Eigen Spaces,” *IEEE Trans. on Image Processing*, vol. 18, no. 6, pp. 1314 - 1325, 2009.

26. T. Kanade, J. Cohn, and Y. Tian, “Comprehensive Database for Facial Expression Analysis,” IEEE Int. Conf. Autom. Face Gesture Recog., pp. 46-53, Mar. 2000.

27. Lyons, M. J., Budynek, J., and Akamatsu, S., “Automatic Classification of Single Facial Images,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, no. 12, pp. 1357-1362, 1999.

28. Niu, Z., Shan, S., Yan, S., Chen, X., and Gao, W., “2d cascaded adaboost for eye localization,” in International Conference on Pattern Recognition, pp. 1216–1219, 2006.

29. Valstar, M., Pantic, M., and Patras, I.,29. M.S. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, J. Movellan, Recognizing facial expression: machine learning and application to spontaneous behavior, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2, pp. 568 – 573, 2005.

Outcome

1. Mohammad, T. and Ali M. L., “Robust facial expression recognition based on Local Monotonic Pattern (LMP)”, in International Conference on Computer and Information Technology (ICCIT), 2011, pp. 572 - 576

APPENDIX

The following code generates the LTP feature based expression images as described in the thesis.

```
function ExpressionClassification()

[ProbeImgNameList Type]=textread('CohonExp6.txt', '%s %d');

fid = fopen('LTP-6.txt', 'wt');

fclose(fid);



for i = 1 : size(ProbeImgNameList,1)

  ProbeImgName = ProbeImgNameList(i);

  ProbeImgName{1} = strcat('SixExpressionLast3Images',

                   '/',ProbeImgName{1});

  ProbeImg = imread(ProbeImgName{1}, 'jpg');

  Probe_LTP = GenerateLTP(ProbeImg);

  fid = fopen('LTP-6.txt', 'at');

  fprintf(fid, '%d ', Type(i) + 1);

  fclose(fid);



  HistogramLTP(Probe_LTP, 15, 11, 256, 256);

  if( mod(i,100)==0 )

        i

  end
```

```
end
```

The following code generates the LTP code for each of the pixel of a given image.

```
function [DstImg] = GenerateLTP(SrcImg)
  dr = [0 -1 -1 -1  0  1  1  1];      %East
  dc = [1  1  0 -1 -1 -1  0  1];      %North East

  [Ysize Xsize]=size(SrcImg);
  DstImg=zeros(Ysize, Xsize,'uint8');

  AngleMagnitudeMatrix=zeros(1,8);

  for i=3:Ysize-2
    for j=3:Xsize-2
      bitValue = 1;
      LTP_Code = 0;
      for a = 1 : 8
        nr1 = i + dr(a);
        nc1 = j + dc(a);

        nr2 = i + dr(a)*2;
        nc2 = j + dc(a)*2;
```

```
        diff1 = int16(SrcImg(nr1, nc1)) - int16(SrcImg(i, j));
        diff2 = int16(SrcImg(nr2, nc2)) - int16(SrcImg(i, j));

        if ( (diff1 < 0 && diff2 < diff1)
             || (diff1 > 0 && diff2 > diff1))
          LTP_Code = LTP_Code + bitValue;
        end
        bitValue = bitValue * 2;
      end
      DstImg(i,j) = uint8(LTP_Code);
    end
  end
```

The following code generates the LTP histogram based feature for a given image.

```
function result = HistogramLTP(Face1, RowNo, ColNo, ImagePixel,
                               BinSize)

  [maxRow maxCol] = size(Face1);
  rowSize = maxRow / RowNo;
  colSize = maxCol / ColNo;
```

```
  Histogram1=zeros(1,RowNo*ColNo*BinSize,'double');


  index=1;
  for y=1:RowNo
    for x=1:ColNo
      [counts bin]=imhist(Face1( (y-1)*rowSize+1 : y*rowSize,
                         (x-1)*colSize+1 : x*colSize  ));
    for sc = 0 : BinSize - 1
      Histogram1(1,index + sc)=counts(sc+1);
    end


    index=index+BinSize;
  end
end


fid = fopen('LTP-6.txt', 'at');


for i = 1 : RowNo * ColNo * BinSize
  fprintf(fid, '%d:%d ', i, Histogram1(i));
end
fprintf(fid, '\n');
fclose(fid);
```

The following code uses the ground eye truth method to crops just the face out of the expression image from the database and resize them appropriately.

```
[AllImgName, LEX, LEY, REX, REY, NX, NY, MX, MY]=textread('dupI-groundTruth.txt', '%s %d %d %d %d %d %d %d %d');


for i = 1 : size(LEX,1)

  fname = AllImgName(i);

  L_Eye_X = LEX(i);
  L_Eye_Y = LEY(i);
  R_Eye_X = REX(i);
  R_Eye_Y = REY(i);
  Nose_X = NX(i);
  Nose_Y = NY(i);
  Mouth_X = MX(i);
  Mouth_Y = MY(i);


  folderName = fname{1}(1:5);
  saveFileName = fname{1};


  fname{1} = strcat(folderName,'/',fname{1});

  pos = strfind(fname, '.');
  fname{1}(pos{1}+1) = 'p';
  fname{1}(pos{1}+2) = 'p';
  fname{1}(pos{1}+3) = 'm';

  img = imread(fname{1},'ppm');


  minX = R_Eye_X - (L_Eye_X - R_Eye_X) / 2;
```

```
    maxX = L_Eye_X + (L_Eye_X - R_Eye_X) / 2;
    minY = (L_Eye_Y + R_Eye_Y) / 2 - (Mouth_Y - Nose_Y);
    maxY = Mouth_Y + (Mouth_Y - Nose_Y);


    dimX = maxX - minX + 1;
    dimY = maxY - minY + 1;


    NewDimX = ceil(dimX/6) * 6;
    NewDimY = ceil(dimX/6) * 7;


    changeX = NewDimX - dimX;
    minX = minX - changeX / 2;


    changeY = NewDimY - dimY;
    minY = minY - changeY / 2;
    cropImg = imcrop(img, [minX, minY, NewDimX - 1, NewDimY - 1]);


    fname{1} = saveFileName;
    pos = strfind(fname, '.');
    fname{1}(pos{1}+1) = 'j';
    fname{1}(pos{1}+2) = 'p';
    fname{1}(pos{1}+3) = 'g';
    imwrite(cropImg, fname{1}, 'jpg');
  end
```