7-17-2013

Automated and Real Time Subtle Facial Feature Tracker for Automatic Emotion Elicitation

Ahmad Ridwan Ibn Sarwar

AUTOMATED AND REAL TIME SUBTLE FACIAL FEATURE TRACKER FOR AUTOMATIC EMOTION ELICITATION

by

Ahmad Ridwan Ibn Sarwar

A Thesis

Submitted in Partial Fulfillment of The

Requirements for the degree of

Master of Science

Major: Computer Science

The University of Memphis

August 2013

ABSTRACT

Sarwar, Ahmad. M.Sc. The University of Memphis. August 2013. Automated and Real Time Subtle Facial Feature Tracker for Automatic Emotion Elicitation. Major Professor: Dr. King-Ip (David) Lin.

This thesis proposed a system for real time detection of facial expressions those are subtle and are exhibited in spontaneous real world settings. The underlying frame work of our system is the open source implementation of Active Appearance Model. Our algorithm operates by grouping the various points provided by AAM into higher level regions constructing and updating a background statistical model of movement in each region, and testing whether current movement in a given region substantially exceeds the expected value of movement in that region (computed from the statistical model).

Movements that exceed the expected value by some threshold and do not appear to be false alarms due to artifacts (e.g., lighting changes) are considered to be valid changes in facial expressions. These changes are expected to be rough indicators of facial activity that can be complemented by contextual driven predictors of emotion that are derived from spontaneous settings.

Table of Contents

Contents **Page**

List of Tables

Table **Page**

List of Figures

Figure **Page**

Chapter 1

Introduction

1.1 Overview

In person-to-person nonverbal interaction, human intelligence enables people to take appropriate actions through perceived emotional experiences. Emotion perception depends on human behavior, which is composed of facial expression, voice, action, and movement of different body parts. In human to human communication, faces are one of the most important attributes that governs the perception of emotional and affective states. Facial behavior can provide information about affective state or cognitive activity just to name a few. And facial behavior is characterized through facial expressions. Although there are six basic facial expressions described by Ekman *et al.* [1], spontaneous behavior of a human face can range up to thousands of intensities over those six basic expressions. Moreover, psychologists are interested in complex emotional states of the human mind rather than the six basic emotions.

Therefore, one of the grand challenges in emotion research is to make artificial agents and machines capable of understanding the mechanisms of how human beings interact with the world and each other. In fact, human-like communication is desirable between man and computer agents. For example, in automated mentoring systems autonomous agents could provide feedback based on assessment and identification of users' emotion and affective states using state of the art facial image analysis. In addition, functional relevance of learning with emotion, affective state, and their interplay can be used to enhance learners' experience as well as the utility of the tutoring systems.

Autonomous analysis and synthesis of facial expressions and emotions are emerging issues in affective computing and in artificial intelligence community. Capturing facial features and measuring their appearances are the core discipline of facial expression recognition research. Though a human observer can easily perceive the changes in facial features quite easily, objective definition of each facial feature is necessary for a machine to perceive automatically.

1.2 Research Challenges

Facial expression recognition has been at the core of interest of behavioral scientists over the last decade and extensive works have been done focusing the issue. Despite the surge of methods and tools, robust and real time recognition of facial expressions remains challenging due to inaccuracies of measurements of subtle facial deformation, pose, and out of plane head movements. Facial Action Coding System (FACS) [1] and its underlying Action Units (AUs) have been widely used for recognizing facial expressions [2], emotions [3] and more widely for affective states [4] .Though AUs are typically able to represent the activities of facial muscles those are significant and intense but are not very responsive to dynamic and fast paced changes of facial expressions specially those are subtle enough.

Another key area of limitation is that state-of-the-art methods in expression recognition are limited to subsets of posed expressions. They lack robustness in recognition of natural and spontaneous facial expressions, modeling blended emotions and are also unsuitable for real time applications.

Although in most recent days some contributions have been made focusing on any one of these issues but still no significant efforts have been made considering all of the issues mentioned above. Like the most influential and popular contribution in recent days is the work by Bartlet *et al.* (2009) [5] where they have developed the Computer Expression Recognition Toolbox (CERT) for automatic extraction of facial expressions from video sequences and has been successfully applied to a collection of natural data set. But CERT relies only on FACS AUs and are not designed to handle subtle expression changes and thus limited to detecting only extreme, intense and slowly moving changes in facial expressions. A number of other recent key contributions are [6] ,[7], [8] and [9]. Approaches followed in all of these reported literature were AU based as well. As subtle changes reflects only in a small number of facial features, dealing with subtle expressions instead of AUs requires any detection algorithm to be able to classify different expressions using these small features set only. Moreover to be able to detect subtle changes by any method or tool that tool has to be highly sensitive to any small variations in facial structure and at the same time should not be susceptible to noises induced by different perturbation sources. As these noises can cause only small variations in output data they could have been otherwise ignored if we were not needed to consider small variations as real expression changes. All these issues make it highly challenging to follow other approaches instead of FACS-AU.

Some other recent contributions those have taken care of the limitations of AUs and thereby followed other alternate approaches are reported in [10], [11] and [12]. In later two a geometric feature based approach have been followed where the former one adapted geometric feature along with feature/motion magnification approaches. Unfortunately although all these approaches appears to be promising in terms of detection method and accuracy as well, all these were evaluated only on posed expression databases and no results on any benchmark natural dataset were obtained. Reasons behind sticking to posed expression data base is due to the fact that spontaneous and natural facial videos are highly susceptible to changes in pose, illumination and other sources of variations regularly encountered in a real world environment. Dealing with real world data is also tricky because of the underlying challenges involved to deal with the out-of-plane head rotation, partial or full facial occlusions or partial zoning out of face from the camera frame as all these perturbations coincide with any real world data.

So because of all the mentioned challenges and difficulties involved a comprehensive facial expression detection mechanism tackling all these issues still remains elusive.

1.3 Proposed Method

A comprehensive emotion elicitation system needs an automatic facial expression recognition method that will be able to overcome all the challenges mentioned in previous section. In this thesis we propose a method and an accompanying software tool for facial expression detection that addresses all these issues with remarkable successes.

In order to make our system sensitive enough to any subtle deformation in facial structure we needed to follow an alternate approach rather than relying on AUs. Keeping this in mind our work focused on utilizing geometric feature based approaches as unlike the AU dependent appearance based models geometric features are able describe the shape variations of each individual components of the face such as mouth or eyebrow. One particular track of geometric approaches that is based on deformable models has become popular in recent years for non-rigid object tracking and is making its way into the field of real-time facial expression recognition. One of the deformable model based approaches which is commonly known as the Active Appearance Model [13], has become very popular for tracking non-rigid objects such as the human face. This approach use Active Appearance Model (AAM) or its derivatives track a dense set of facial points (typically 60-70). AAM works by first building a statistical model based on some training facial images and then fitting this model to previously unseen facial images. After fitting is done successfully, different features of the face and changes in these features according to change of appearance of face can be extracted using AAM. A number of different algorithms have been devised according to the underlying statistical theories for building as well as for fitting the AAM (A. Asthana *et al*. 2009)[11]. Model building is done by building a mesh that actually replicates the shape and appearance of human face. And the goal of fitting is to correctly superimpose this mesh on the facial region of a facial image/video frame and thus this mesh will change its shape and appearance according to the corresponding changes in facial expressions.

However our present focus is on the real time analysis of naturalistic facial behaviors those accompany the expressions of complex and context specific emotions. In particular we are interested in learning-centered emotions such as confusion or frustration considering the recent surge of research in developing automatic mentoring systems and those systems need to assess the participants' affective states in a real time manner.

Previous research has indicated that facial activities around the brow and mouth are particularly diagnostic of confusion [14][15]. So our focus is on detecting changes in those areas. Any subtle changes in these regions detected by our system is considered to be rough indicators of facial activity that will be complemented by contextual driven predictors of emotion that are derived from tutorial events. For example, a change in brow activity immediately following a contradictory statement by an animated agent is expected to be indicative of confusion because of the alignment of both the diagnostic signals from the face and predictions from the stream of tutorial events.

1.3.1 Technical Architecture of the Proposed Method

In our system, we used the mentioned [37] real-time facial feature tracker based on the AAM to extract the shape vector for each video frames. This shape vector is further processed and a compact feature vector, representing the facial features, is obtained. This feature vector is then used for identifying subtle changes in appearance in certain regions of face. Among different shape and appearance features captured and provided by AAMs, we have chosen 68 vertex points and each point is a two dimensional vector consists of x- and y-coordinates, resulting in a raw 136 dimensional feature vector.

These points change their location and relative distance in frame by frame according to the appearance change of different facial regions. Ideally, location of these points are measured from the upper left most point of the frame and the distance values that are provided by the AAM are relative to that point. After extracting the values of these points, we measured the relative distance of each point from the centre most point on the face, which is point 67, located on the nose. For each frame, we have the relative distance from our central nose point to each of the points of each of the regions. So, if we observe a significant change in values of any point in a frame compared to its value that was in the previous frame(s), we can say that those corresponding point(s) has changed their locations and hence we can detect a movement in those points.

1.4 Major Contributions

In this present work we have presented a GUI based software tool for fully automatic and real-time facial expression recognition and also releases it to the research community for free use. The underlying open source tool that has been used as the backbone of our system is the openCV implementation of Active Appearance Model (AAM) (T. Cootes *et al*. 1998) [16]. This software will facilitate researcher to analyze facial structures, movement of facial muscles, tracking of eyes and other significant movements. Researcher can analyze both in frame level and video level. Moreover, user can track their confusions through the GUI. We also describe two sets of benchmark performance data as a resource to assess the efficacy of our system.

More focused goal of our research was to develop a system to detect affect (particularly confusion) by monitoring changes in activity around the brow and mouth. Our system was concurrently evaluated on two separate sets of video data and both of them were captured in real world scenarios. First set of videos were captured during the device breakdown study and the second set were captured during the confusion induction through contradictory assertions study. Brief descriptions of both data sets are given in Chapter 5. of this thesis. One was used to evaluate the system’s performance in detecting facial expression changes in mouth and eye brow region and the later one was used to assess how well these detected facial changes align with the different self reported emotions so that these detections can be considered as diagnostic of these emotions. For our expression detection evaluation task our automated tracking system’s annotated segments of brow and mouth movements were compared to the human annotated segments and the achieved outcome was a 80% hit rate with 2.25 d-prime value for mouth and a 75% hit rate with a 2.11 d-prime value for eye brow. And for our confusion detection task it has been observed that during the contradiction episodes (where induce confusions are likely to be higher) our system captures highly frequent changes in mouth and eye brow regions which is almost 100% higher than the episodes those are neutral and thereby expects no induced confusions.

Key contributions of this thesis are summarized as follows:

1. The facial feature tracker system that has been developed as part of this thesis is able to detect subtle facial changes with notable accuracies.
2. To overcome perturbations and thereby be able to process real-world and naturalistic data our system is integrated with a number of noise detection and reduction strategies.
3. Our system demonstrated satisfactory performance for using its detected facial expressions as diagnostics for complex emotions like confusion.
4. Outperforms other existing works in terms of these three key factors namely in detecting subtle changes, robustness against real world data and correct assertion of complex emotions like confusion.
5. All of the backbone tools and software used for our system are totally open source and are freely available. This was a primary goal of this project so that it will allowed us to release our tool as openly available to research community.
6. Our system can be used with any previously unseen data set without the need of any prior training and still the same performance outcome will be generated as we obtained with our data set.

1.5 Outline

The rest of the thesis is organized as follows:

Chapter 2 explores the available literatures for automatic facial expression recognition systems. Chapter 3 and 4 provide all the technical details of our proposed methods, where chapter 4 is particularly devoted to describe the noise detection reduction processes. Chapter 5 describes data collection and annotation process of emotion elicitation dataset. Chapter 6 and 7 describe our systems performance on the two data sets where in chapter 6 it describes our facial movement detection performance using first data set and in chapter 7 it describes our contextual emotion inference performance using second data set. Chapter 10 concludes the thesis with future plans.

Chapter 2

Literature Review

Facial Expression Recognition and classification of displayed facial expressions in a number of discrete emotion categories has been an active topic in computer science and in behavioral science for decades, with the first landmark work on this area of research being published in 1973 [17]. Many other effective works have been reported since then [18], [19]. A comprehensive survey in this field was first published in 1992 [20] and has been followed by several others [18], [19], [21].

A key source of notable attention is the FERA (Facial Expression Recognition and Analysis) challenge [22] that presented a meta-analysis of the first challenge in automatic facial expression recognition held during the IEEE conference on Face and Gesture recognition 2011. That reported a number of recent key contributions on this area of research.

There are two main research streams those are followed by the facial expression recognition and analysis community. One of them stems from the sign based approaches to facial expression measurement in psychological research [23] and they represent the facial actions in a coded way (like FACS-AU [1]) or through a collection of some landmark points, where the facial actions are abstracted and described by the locations and displacement intensity of these points. And the other stream stems from the judge based approaches which is followed by the psychological research of facial expression measurement as well [23] and in and this approach directly associate specific facial patterns with mental activities, where the facial actions are classified into one of the six basic emotion classes [1] or in few cases into other complex emotions as well.

The former of the two streams is commonly called as *Sign Detection* and the later one is called as *Emotion Recognition.*

2.1 Emotion Recognition

The collection of researches done in this stream can be divided in two groups based on the types of facial features they have used. These are *Appearance based features* or *Geometric based features* [22]. Appearance Features treat the face as one single entity and describes the textures of the face and Geometric Features describes individually the shape of each different component on the face such as mouth or eye brow.

Within the appearance based approaches a recent technique proposed by Zhi *et al*. [6] which the authors called as graph preserving sparse nonnegative matrix factorization (GNSMF) has demonstrated remarkable successes when applied with the problem of six basic emotions recognition. The GSNMF attains occlusion-robustness by transforming high-dimensional facial expression images into a locality-preserving subspace. On the Cohn-Kanade database, it attains a 94.3% recognition rate. On occluded images it scored between 91.4% and 94%, depending on the area of the face that was occluded.

Another recent technique which is a variation of GNSMF and is a based on non-linear non-negative component analysis, a novel method proposed by Zafeiriou and Petrou [24].On the Cohn-Kanade database they attained an average 83.5% recognition rate over the six basic emotions. Littlewort *et al*. [5]

presented a CERT [5] based system where the head orientation prediction and one other CERT output known as Extremes of Displacement, Velocity and Accelaration (EDVA) are computed and then a Multinomial Logistic Regression classifier using these features was used to detect the emotions and they were also measured only the six basic emotions.

Most geometric feature based approaches use Active Appearance Models (AAMs) or derivatives of this technique to track a dense set of landmark facial points (typically 60-70). Locations and displacement of these points are then used to track the shape distortion of facial regions like mouth eye brows and that in turn is used to detect facial expressions. Asthana *et al*. [11] compared different AAM fitting algorithms and evaluated their performance on the Cohn-Kanade database [11]. Another example of a system that uses geometric features to detect emotions is that by Cohn *et al*. [25] where they presented a method utilizing AAM tracking and spectral graph clustering. However, the tracking was limited to the mouth region only. One other landmark achievement by using AAM was done by Sung and Kim where they used AAMs to track facial points in 3D videos [12]. They proposed an improved AAM that enhances the fitting and tracking of conventional AAMs by using multiple cameras to model the 3D shape and motion parameters. Then a Linear Discriminant Analysis (LDA) was used to combine the 3D shape with the 2D appearance output. Although the proposed method provided good results it was evaluated only in a subset (using only 3 of them) of Ekman’s basic emotions.

2.2 Sign Detection

Sign detection or Action Unit detection approaches also follow either of the two avenues (appearance based features or geometric features) of feature selections with a few exceptions those follow a combination of both.

One particular class of appearance based feature that have been extensively used in recent days are dense local appearance descriptors. This method works by computing appearance descriptor for every pixel and then summarizing them by histograms in different sub-regions. Jiang *et al* [24] used this approach with LBP and LPQ [9]. Littlewort *et al*. [5] used the Gabor Wavelet filter as appearance descriptor. And Whitehill and Omlin [26] used Haar-like features backed by AdaBoost. Cohen *et al* [27] uses Gaussian Tree-Augmented Naïve Bayes (TAN) to learn the dependencies among different facial motion features in order to classify facial expressions.

In the geometric feature based approaches Lucey *et al*. [28] used different computer vision techniques where Active Appearance Model (AAM) has been used to extract features those consist of shape and appearance information. Valstar and Pantic [29] made use of 20 landmark facial points those are sparsely distributed over the face. Then different properties of these points such as distances between pair of points or velocity of a point were used to extract different spatio-temporal features of the face.

A mixed approach following both of appearance features and geometric features was utilized by Simon *et al* [8]. It first uses geometric feature based AAM to track different landmark points of the face and those then local appearance descriptors (which is an appearance feature based approach as mentioned earlier) are computed for each of these tracked points. This system was evaluated for 8 AUs on a natural dataset which is famously known as M3 dataset and it achieved an area of 83.75% under ROC curve.

For addressing issues related to real-world data like large data size or infrequent AU occurrences Zhu *et al*. [30] proposed a method for automatic selections of optimal training set and hence reducing the processing load of the large data set. On the natural M3 dataset they achieved a 79.5% area under ROC curve.

2.3 Limitations

- It is clear from the literature that very few efforts have been made to encounter the real world perturbations like out-of-plane head rotations, occlusions or face zoning out and thus application of facial feature tracker in any real world scenario is still remains as challenging endeavor to overcome.
- Due to the mentioned challenges and also due to the large scale feature dependencies (either appearance based or geometric based) of most of the methods mentioned in the literature, processing of natural data set is tricky as natural data is usually accompanied with subtle facial changes which does not makes that many feature variations on the face as it is needed by most of the existing methods.
- Though at [30] some real world issues have been addressed like large data size or infrequent event or AU occurrences, but the proposed

solution was focused on AU detection and not on subtle event detection. But subtle facial behavior is more common in naturalistic settings.

- Even in the FERA challenge [22] it has been seen that most of the existing works are still based on AU detection and thereby limited in detecting intense facial expressions only.
- As in [6], [24] and [25] even though state-of-the art methods and techniques demonstrated promising performance in emotion recognition task, but all of them were evaluated only on six basic emotions or a subset of them [12] and no work with significant performance for complex emotions have been reported.

Chapter 3

The Subtle Facial Feature Tracker

As already been mentioned the goal of this present work was to develop a system to detect affect (particularly confusion) from open source computer vision tools for tracking facial features. This system should be capable of real time analysis of affective states from video sequences captured by a live camera. As requirement of fulfilling this goal we worked towards developing a GUI based software tool using openCV (open Computer Vision) library. The underlying open source tool that we have used is the openCV implementation of Active Appearance Model (AAM) (T. Cootes *et al*. 1998)[16].

This chapter is devoted to provide technical implementation details for each of the steps followed to make our endeavor a reality along with different challenges we encountered at each of the steps and the devised solution to tackle those challenges.

3.1 Technical Description

Among different AAM algorithms and their corresponding implementations we have used the one that is freely available on google code. This implementation employs the ‘Fixed Jacobian Method’ (T. Cootes *et al*.,1998) [16] as well as the ‘Inverse Compositional method’ (S. Baker *et al*., 2001) [31] algorithms. It has been developed using openCV1.0 (Open Computer Vision Library 1.0). Our AAM based system has three major components. These are:

1. Face Detection Component.
2. AAM fitting and Facial Landmark points extraction Component.

3. Facial movement detector Component.

Based on this component architecture a system upper level view is given in figure 3.1. Almost all of our works were devoted in the third component namely to extract facial movements in the target regions by exploiting the spatial properties of extracted landmark points. Subsequent three sections illustrated all these three components one by one.

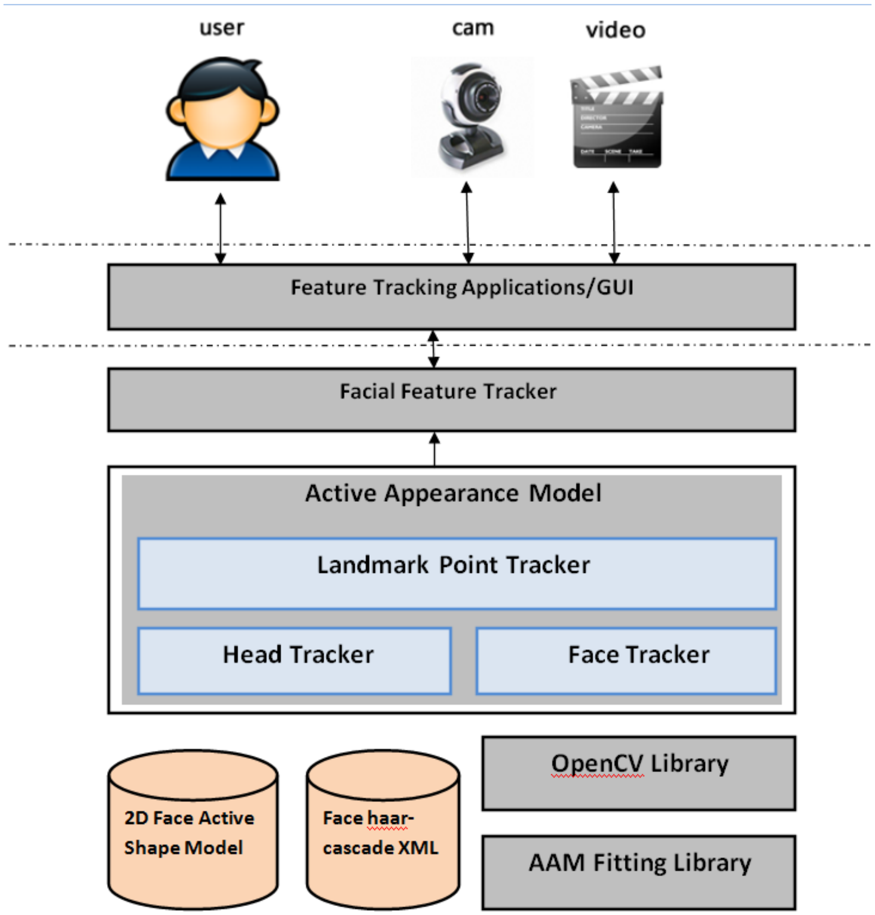


Fig 3.1 Technical Architecture

3.2 Face Detection Component

The very initial challenge that every facial feature tracking system encounters is to detect the location of the face on a given image or video frame with a reasonable accuracy. Among the various face detection methods our system use the well known Viola-Jones face detector (Viola and Jones, 2001)[32] because of its computational efficiency, performance and also because of the availability of an implemented version in OpenCV. This method works by representing any image with their proposed ‘integral image’ representation that in turn is processed by an AdaBoost algorithm to detect the face.

3.3 AAM fitting and landmark point extraction component

AAM is a model based image alignment method that has become the most popular method for image alignment tasks and applications. In facial images or video frames face is aligned with the AAM following the same image alignment procedure. The most important integral part of this alignment process is known as model fitting. This AAM fitting is done by superimposing the underlying mesh on the face.

Once the fitting is done successfully, the resulting mesh can be used to extract different parameters of it including the spatial values of different landmark points. Our AAM algorithm uses 68 landmark points indexed from 0 to 67 where point 0 is located right beside the right eye and point 67 is located on nose tip. All other points are contiguously distributed over different locations of the face as shown in figure 1.2. Spatial properties of these landmark points are then used to detect facial movements in different areas of the face.

3.4 Facial movement detector Component

Our system works by processing the input video frame by frame. It grabs each of the frames one by one, does some initial processing on it and finally fed into the AAM algorithm for fitting. After some iteration of the algorithm and achieving an acceptable accuracy in fitting, finally the mesh is placed on the face. All the different processing that the extracted landmark points undergo to detect facial movements from these points is discussed in the following subsections:

3.4.1 Form the Feature Vector (F_v)

The AAM fitting algorithm outputs spatial properties of the landmark points in the form of a two dimensional vector consisting of 68 points. Each point of this vector has two values, one is for x-dimension and the other is for Y-dimension. We call this vector as ***Feature Vector or F_v***. Each of the points of this vector represents a point of the face in the two dimensional co-ordinate of the screen. So with the output of the algorithm we can know the exact position of these 68 points of the face for each frame. With this information at hand for each of the frames, at any instance of time if we see that a set of points residing in a particular facial region have a significant change in their positions compared to their positions in previous frames we can assert that a movement has been occurred in that region of the face. We represent our *Feature Vector (F_v)* as follows:

$$F_v = [(x_0,y_0), (x_1,y_1), (x_2,y_2), \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.. (x_{63},y_{63})] \qquad (1)$$

3.4.2 Form the Relative Vector (R_v)

In general, if the head moves or the face moves as a whole, then each of the 64 points on the face will give us a change in their positions. If we only take the positions of each point and track their changes than with this head or face movement we will misinterpret that an expression change is being occurred in each of the region of the face. This issue can be resolved if you choose one of the point among all these 64 points as a reference point and track the relative distance of each other points from this reference point instead of tracking the absolute value of each of the points. We have chosen point number 67 as our reference point as this point is located on the tip of the nose which is the center most location on the face. To take the relative pixel values of all other points we subtract the pixel values of this reference point from each of their values for each of the frames. In this way we can track this relative value for each of the points for each of the frames. Now if we see any change in these relative values for any set of points residing in a particular region we certainly know that this change is due to an expression in that region and not because of the movement of the head or whole face. As our Feature Points are two dimensional, these relative distances are measured in the form of Euclidean distances and thus we form a new vector and we call this as ***Relative Vector (R_v).*** Each point of this vector represents the Euclidean distance between our reference point and the corresponding point of the *Feature Vector.* As our reference point in the Feature Vector is ($x_{63,}y_{63}$) and if we represent each point on our *Relative Vector* as d_i (where 0<=i<=67) then our Euclidean distance equation should be:

$$d_i = [(x_{67} - x_i)^2 + (y_{67} - y_i)^2]^{1/2}\ , \quad where\ i = 0,1,2,\ldots\ldots..62 \qquad (2)$$

And so the Relative Vector R_v will be formed as:

$$R_v^T = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ . \\ . \\ . \\ d_{67} \end{bmatrix} = \begin{bmatrix} (x_{67} - x_0)^2 + (y_{67} - y_0)^2 \\ (x_{67} - x_0)^2 + (y_{67} - y_0)^2 \\ (x_{67} - x_0)^2 + (y_{67} - y_0)^2 \\ . \\ . \\ . \\ (x_{67} - x_0)^2 + (y_{67} - y_0)^2 \end{bmatrix}^{1/2} \qquad (3)$$

And finally our vector R_v will be:

$$R_v = [d_0 \quad d_1 \quad d_2 \quad \cdots \quad d_{67}] \qquad (4)$$

3.4.3 Divide the points in zones and take average value for each zone

After having these relative distances for all of the points, we are now interested to detect if these distances have a significant changes in their values compared to the previous frames. Now instead of treating each point individually and tracking the changes in their values we are interested to treat the points as a whole those are closely placed in a facial region of our interest. This facilitates us finding deformation/changes in appearances in different facial regions of our interest like mouth, eye brow, chin etc. We have divided all 68 points in the following regions: Mouth (points 48-66), Eye-brow (points 21-26 & 14-20), Eye-lid (27-31 & 32-36), Chin (0-8) and the boundary region of the face (2-12). We take the mean of the values (relative distance from the central points as mentioned earlier) of all the points in a region and if we detect any change in this mean value compared to the mean of previous frame(s) we consider an appearance change in that region. This way we can say if a movement has been occurred in mouth or eye brow etc. With this process the initial 68 dimensional *Relative Vector* gets mapped to a lower dimensional vector where the number of dimension is only equal to the number of regions of our interest. We name this low dimensional feature vector as ***Mapped Vector (MP$_v$)***. In addition to finding appearance changes in different area of interest this zoning process also works as a filter for the noise that our system encounters due to lighting and other unavoidable issues which we will discuss later. We form this *Mapped Vector* by multiplying our *Relative Vector (R$_v$)* with the *Area Matrix* and this *Area Matrix* is a matrix where each column of it represents a transposed *Area Vector.* Each of the area of our interest (between which we are dividing our points) has its own *Area Vector* which is a 68 point vector where each of the points those represent that area has a value of 1 and all the other points has a value of 0. Like for mouth the *Area Vector* will have a 1 for the points 48 to 66 and all other points of its vector will be 0. So if our *Area Matrix* is a (m×n) dimensional vector then m will be equal to 68 and n will be equal to the number of areas of our interest and which is 5 in our case. So our *Area Matrix* will be like this:

$$A_M = \begin{array}{r|ccccc|} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 8 & 1 & 1 & 0 & 0 & 0 \\ 9 & 0 & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 12 & 0 & 1 & 0 & 0 & 0 \\ 13 & 0 & 0 & 0 & 0 & 0 \\ 14 & 0 & 0 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 26 & 0 & 0 & 1 & 0 & 0 \\ 27 & 0 & 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 36 & 0 & 0 & 0 & 1 & 0 \\ 37 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 47 & 0 & 0 & 0 & 0 & 0 \\ 48 & 0 & 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 66 & 0 & 0 & 0 & 0 & 1 \\ 67 & 0 & 0 & 0 & 0 & 0 \end{array} \tag{5}$$

And so the ***Mapped Vector*** will be formed by using the following formula:

$$\mathrm{MP_v} = \begin{bmatrix} d_0 & d_1 & \dots & \dots & d_{67} \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (6)$$

$$\mathrm{MP_v} = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & a_4 \end{bmatrix} \times \begin{bmatrix} \frac{1}{9} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{11} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{13} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{10} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{19} \end{bmatrix} \qquad (7)$$

3.4.4 Build model of each zone

This is obvious that different people have different shape, appearance and structures of faces. This results in different ranges of values for the relative distances of points. To encounter this issue our system builds a model for each users using a predefined numbers of initial frames of the video/web cam session. This predefined number is termed as ***Model Length***. The system stores the Area Mapped feature vectors for each of the incoming frames in a collection until the collection is filled with the number of frames equal to Model Length. When the collection is filled the system calculates the mean value of all the Area Mapped feature vectors stored in the collection and makes a new vector storing the mean values for each of the areas. In this way all the stored Area Mapped feature vectors are now converted into a single vector where each dimension of the vector represents the mean value of the averaged relative distances of points residing in an area of our interest over all of the frames in the model. We call this vector ***Model Vector***. So, this Model Vector gives us an estimate of the facial structure of the participant as from this we can know what should be the ideal distances of all of the regions from our reference point when the face is in its neutral state (there is no movement in the face). So from this estimate at hand if we see a large deviation of the distance of any region from the reference point in compared to its distance in the Model Vector we can certainly infer that a movement has been occurred in that region.

$$(\mathrm{M_V})^T = \frac{1}{N}\sum_{i=0}^{N-1}(\mathrm{A_i})^T \tag{8}$$

$$(\mathrm{M_V})^{\mathrm{T}} = \frac{1}{\mathrm{N}}\left(\begin{bmatrix} a_{0_0} \\ a_{0_1} \\ a_{0_2} \\ a_{0_3} \\ a_{0_4} \end{bmatrix} + \begin{bmatrix} a_{1_0} \\ a_{1_1} \\ a_{1_2} \\ a_{1_3} \\ a_{1_4} \end{bmatrix} + \begin{bmatrix} a_{2_0} \\ a_{2_1} \\ a_{2_2} \\ a_{2_3} \\ a_{2_4} \end{bmatrix} + \cdots\cdots\cdots + \begin{bmatrix} a_{(N-1)_0} \\ a_{(N-1)_1} \\ a_{(N-1)_2} \\ a_{(N-1)_3} \\ a_{(N-1)_4} \end{bmatrix}\right) \tag{9}$$

3.4.5 Calculate z-score and Detect Event

It should be noted that the unit of our measurement of all the values or distances that we are talking about is the number of pixels on the computer screen. As the unit is very small even with the small changes in facial appearance we get significant changes in pixel values. Again as the intensity of facial movement can ranges from quite high to very subtle resulting in a highly fluctuating or varied values of pixel distances, we need to take into consideration of the variance or standard deviation of the values of the points. So, to consider a change from the Model Vector as a significant one, it is not always sufficient to rely simply on the difference from the mean values of each zone those are stored in the Model Vector. Hence we have considered z-scores of the averaged values of the points residing in a zone of the current frame to detect a movement. And in no doubt these z-scores are calculated against the values stored in our Model Vector. If the z-score of a region of the current frame (which is calculated by subtracting the mean values of the model frames from the value of the current frame and dividing by the standard deviation value of the model frames) is higher than a pre-determined threshold value, we consider a movement in that region in that frame.

3.4.6 Dynamic Model rather than a static one

It is quite intuitive that participants' body posture as well as the head position will change from time to time while using the system. Some sources of impediments can also be there like the participant can keep their head tilted in forward or backward direction or can keep their hand over the face and so on. All these changes impose constraints on the fitting process of the underlying AAM algorithm and the mesh doesn't always get placed properly. As a result the distances between points those we are interested with (distance from reference point to all other points) get drifted to some extent from their ideal values even when there is no facial movements at all.

We have to abide by these sorts of impediments as our goal is to make our system robust enough to detect facial expressions on natural dataset, we can't impose any restrictions on the posture or any gesture of the users as that will thwart the natural work flow with the task they are involved in.

But the good think is even with all these impediments the mesh still changes its shape and appearance according to the corresponding changes in facial movements in case of real changes in the face. But now in order to detect these changes correctly we have to adjust our baseline values. And this baseline should take into consideration the fact that the distances those we are having in neutral state are not ideal. And this can only be done if we rebuild our model using the frames in which these drift from ideal is being occurred.

To address all these issues we have proposed that the Model we have built using the first few initial frames should be a dynamic one rather than a static one. It is done by simply keeping a dynamically moving window of frames by deleting the very first frame that was added in the collection and adding the current frame at the end of the collection. Each time the collection is updated by this add and delete process the Model Vector is re-calculated so that it can reflect the newly added frame in the mean. Benefits of having a dynamically moving model are two folds. Firstly, it makes the system able to cope up with the dynamics of body postures and head orientations by tweaking the model and making it adaptive. This is possible as the model is now being filled out with the *Mapped Feature Vectors* of the frames in which these changes have occurred.

Ideally the steps those have been described so far would be enough to have our system detects the facial features correctly. But due a number of perturbations the system experiences it fails to exhibit its ideal performances. And so it is needed to exercise a number of noise detection and elimination steps to make our system robust against all sources of perturbations. Next chapter is dedicated to describe all these noise reduction steps in detail.

Chapter 4

Noise Reduction

Noise or false alarm detection and reduction and making our tool robust against these noises was one of the most significant part of our research as our tool were subject to a number of different types of noises induced from different sources. It is obvious that in our system the term noise or false alarm indicates the fact that our system is falsely detecting an event (a movement in any region) when no real movement is taking place in that region.

As said earlier, noise can be induced from a number of different sources resulting different types of noises each having unique natures and characteristics. To address this issue we needed to device different detection mechanisms and filtering techniques to tackle down each different type of noises. These are the primitive sources of noises that our system encounters:

1. Noise induced by fitting inaccuracy of the underlying AAM algorithm due to different external and environmental artifacts like inappropriate lighting or rapid changes in lighting etc.
2. Noise induced by rapid and random head movement of the user.
3. Rapid changes in body postures.
4. Face occlusions caused by some natural gestures of the users like ‘hand over face’ gestures.
5. Fitting error caused by inaccurate face orientation like tilted face etc.

Noise or false alarms induced by each of these sources, their natures and characteristics along with the steps we have taken to detect and reduce them are described in the subsequent sections of this chapter.

4.1 Noises induced by environmental artifacts

This is the primary source of noises that our system encounters. Now, it has been observed that, the mesh itself moves a lot even when there is not any real movement in the face, and this noisy movement of the mesh gives different ranges of values of the point depending on its own movement.

But this movement of the mesh is unavoidable due to fitting error of the mesh that is caused by some limitations of its implementation and the underlying algorithms that this implementation uses. But we had to abide by this noisy nature of this implementation because this is the best implementation of AAM that is open source and can be used freely. As we intend to rely only on open source tools in our project, we have chosen this as this is the best one of them.

So, keeping this noisy nature of the mesh in mind, we have applied some filtering methods to discriminate real movements from those that are caused by the noisy output of the mesh.

After closely observing the output of our system It has been identified that, a distinguishing feature of this type of noise is that events caused by these errors lasts only for one to two frames whereas a true movement in any regions usually cause a resonance in at least 5-6 frames. Though these 5-6 frames may not be consecutive, but they resides so close that if we find them in some consecutive number of frames, we can determine this as a true movement. Observing this behavior of this kind of noises, three approaches have been taken to filter them out. These are in the following subsections.

4.1.1 Averaging the feature Vector

When comparing with the Model Vector in order to calculate the z score and find an event, instead of taking a single Mapped Vector associated with a single frame we can take a number of consecutive frames and average their Mapped Vector. Like the equation 10:

$$\begin{bmatrix} \mathrm{X}_0 \\ \mathrm{X}_1 \\ \mathrm{X}_2 \\ . \\ . \\ . \\ \mathrm{X}_{67} \end{bmatrix} = \frac{1}{N} \begin{bmatrix} \mathrm{X}_{0_0} \\ \mathrm{X}_{0_1} \\ \mathrm{X}_{0_2} \\ . \\ . \\ . \\ \mathrm{X}_{0_{67}} \end{bmatrix} + \begin{bmatrix} \mathrm{X}_{1_0} \\ \mathrm{X}_{1_1} \\ \mathrm{X}_{1_2} \\ . \\ . \\ . \\ \mathrm{X}_{1_{67}} \end{bmatrix} + \ldots\ldots\ldots\ldots + \begin{bmatrix} \mathrm{X}_{(\mathrm{N}-1)_0} \\ \mathrm{X}_{(\mathrm{N}-1)_1} \\ \mathrm{X}_{(\mathrm{N}-1)_2} \\ . \\ . \\ . \\ \mathrm{X}_{(\mathrm{N}-1)_{67}} \end{bmatrix} \qquad (10)$$

Now if we consider this averaged vector and calculate the z scores based on this vector it will average out the noise pulses (as said earlier noises appear only in one or two frames like a pulse). This will happen as the neighboring frames those are not having these noisy pulse will compensate for the high values in the noisy frames.

4.1.2 Look ahead for detecting events

This approach is directly induced from the distinguishing feature of these noises mentioned earlier that true events have their impact in at least 5-6 (may not consecutive) frames whereas noises have their impact in only 1-2 frames. Keeping this finding in mind we have decided that rather than deciding right away that a frame is in event whenever its z-score is higher than the threshold we should instead raise a flag that an event may have occurred and take a look at a predefined number of consecutive frames following that frame. If we find that at least a predefined number of frames in those predefined number of consecutive frames have mapped vectors giving z-score higher than the threshold then we consider that an event has occurred. And if we unable to find mapped vectors with z-score higher than the threshold at least in those predefined number of frames we ignore the events as noise.

So this needed us to fix those two predefined numbers or parameters for filtering true detections from noise. One is the consecutive number of frames that we shall consider to find the bunch of frames those are with movement which we call as ***allFrameInEvent*****.** And the other is the number of frames with movement that we shall consider as a bunch which we call as ***minEventFrameInEvent***.

4.1.3 Wait before inserting *Mapped Vector* into the model

As we are filtering out events those have a short duration considering them as noise it should be make certain that no real event into this small number of frames and are always observed at least a minimum number of frames so that it can be distinguished from noises. Although all real events actually have a long enough duration that allows us to easily filter them out, but the way our system works it may fail to detect a real event in all the frames the event has occurred and this lead us to wrongly classify a real event as a noise one and filter that out.

The underlying reason for this is the way we are updating our Model window. It should be noted that we are constantly updating our Model by inserting each current frame at the end of it and discarding the oldest frame stored in it. This is done in order to keep the model tolerant against the constantly changing head and body posture of the subjects as mentioned earlier.

But this imposes the risk that whenever a real event occurs and the event spans through a number of frames and as we are constantly pushing each incoming frame into the model, the model can get saturated with some initial frames (especially if the initial frames have large pulses in them) of the event. As a result the *Model Vector* will be bumped up to a higher value even before the later frames of the event have been compared with the model and hence all these later frames' *Mapped Vector* will have a z-score below our threshold and will not be considered as event frames.

So the overall result of this is that only some initial frames of the event will be detected as event frames by our system and will be ignored as a legitimate event as we are filtering out any event considering them as noises if they spans through a small number of frames only.

As a remedy of this we have decided to put a buffer between the model and currently arriving frames. This buffer works as a temporary storage of *Mapped Vectors* before they are inserted into the model. Whenever a new frame comes its *Mapped Vector* is inserted at the end of this buffer and the vector that was at the beginning of this buffer is inserted at the end of the model. We call this buffer as ***bfrModelBuffer***. So if the current frame is X_i and the length of our *bfrModelBuffer is* n we insert X_i at the end of this buffer and remove X_{i-n} from this buffer (which is at the beginning of it) and insert this X_{i-n} at the end of the model. In this way our model will be always N number of frames behind our current frame. And thus it is highly unlikely that the model gets saturated with the initial high pulse frames of an event. Because these initial high pulse frames are now being residing in this buffer instead of the model and as a result whenever the later frames of the event has arrived the model has not yet been update with these pulses and thus the *Model Vector* is still having a lower value.

4.2 Noises Induced by rapid head movement of the user

As our tool should be robust against any natural data set we cannot impose any restriction on the posture or their dynamic movement that naturally arises during their interaction with our tool. And if the duration of the interaction is substantially long it is more likely that the users will change their body posture from time to time. And that is the case for most of our studies where the duration of each session is one hour or more. And head is the most dominant part that users tend to move a lot while interacting with any computer tool.

As our system is capable to deal with natural data set it is of no surprise that it is robust against different head positions and can fit the mesh almost full accuracy no matter how the head is positioned e.g. tilted forward or backward or right or left. But as it needs some times for the AAM fitting algorithm to fit the mesh, if the movement of the head is very fast it may not be able to fit it during the interim period when the head is on the move and can only fit it correctly once the head get settled in a position. So it is certain that our tool needs to dynamically detect if the head is on the move and react accordingly.

For this detection we rely on the absolute values of the *Feature Vector* that we obtained from the mesh at the very first step of our process. Points of the

Feature Vector give us the absolute position of the 68 points of the face on the computer screen. We were making this value relative to our reference point (which is the central nose point) for ease of our detection of subtle movement in the facial regions. But if we want to detect the movement of the face as a whole instead of subtle deformation in any specific facial region it is helpful to use the absolute values rather than the relative ones. This strategy has been followed for detecting head movement of the user. We considered the absolute value of the central nose point which is the ($X_{67,}Y_{67}$) point of the *Feature Vector.* We have been detecting subtle movement using the relative values namely by calculating the z-score based on the model and tracking if the z-score is above or below some threshold.

This exact same strategy have been followed for detecting head movement but this time using the absolute values and using only one single point on the face which is the central nose point as mentioned earlier. Our goal is to constantly track the absolute value of this point in each of the *Feature Vectors* and see if we observe any abrupt change in this value in any of them. This change is measured using the z-score as we have done with different facial regions as well.

For facial regions we created the *Mapped Vector* from the *Feature Vector* by dividing the points in different regions and each point of the *Mapped Vector* represents the average value of a region. We treated the central nose point as a different region in order to ease our process. So the *Mapped Vector* contains one extra point that represents a region consisting only of this one single point and as the *Model Vector* is just an averaged vector of different *Mapped Vectors,* the *Model Vector* contains a point representing this region as well. And it is obvious that this point of the *Model Vector* contains the averaged absolute value of this central nose point and not the relative ones like the other regions.

So in order to track any abrupt changes in the value of this point in any given frame we just need to calculate the z-score of the absolute value of this point against that corresponding region in the *Model Vector.* And if we see that this z-score is above some pre specified threshold we consider an abrupt change in that point. Now as we have been tracking absolute values for this point, an abrupt change in this point means the nose have changed its position significantly from its prior location. And as the nose tip is a static point on the face, means we can't move this point unless we move whole of our face/head, a movement in the nose tip means the whole face/head has been moved.

This way with the bumping up of the z-score we can detect that the participants head has been moved and if the changes is too high and this change happens in a very short duration of time that implies that the head has moved so fast and it moved significantly. As we know that AAM algorithm is unable to fit the mesh correctly when the head is moving in this fast pace, so if we take output from the mesh during these times it is of no doubt that those values will be incorrect giving us a lot of noises. And when someone is moving their head it is highly unlikely that there will be any legitimate facial expressions. Keeping this in mind and to reduce noise we have decided to discard all the frames those involve these fast head movements.

So whenever we detect a significant and rapid head movement in any frame by observing its z-score value we discard a predefined number of frames following that frame and whatever events we detect in those frames we consider them as noise. This process needs us to set two more predefined numbers. One is the threshold for z-score of that central nose point above which we will consider a head movement has occurred and we call this as ***HeadThreshold****.* And the other is the number of frames those needed to be ignored whenever a head movement is detected and we call this as ***IgnoreCount****.*

4.3 Noise Induced by partial face occlusions

As mentioned earlier, in order to keep the natural workflow of the users we can’t impose any restrictions on their movement or any posture(s) or gestures(s) they want to make. And to make our system robust enough to act accordingly against any noises or missing of real events those may caused by these natural movements of the user.

Two particular form of posture are seen very frequent among all the users and both of them significantly thwart the performance of the detection algorithm. One of them is commonly known as the hand over face gesture and that is when someone puts their hand over the face occluding some parts of the face with it [33]. And the other one is tilted face means the face is not oriented in a way so that it is parallel to the computer screen and rather it is tilted towards the screen.

Though AAM is trained with a variety of faces with different poses for each of them but its’ not trained with partially occluded faces. As stated, being trained with a variety of faces and poses still makes it able to fit the mesh even on the faces which are partially occluded, but the inaccuracies may causes it to get converged to a shape that is either squeezed or expanded too much in some frames. With these converged shapes the values for different regions changes drastically giving a high z-score and make our system wrongly detect that an event has occurred. And clearly all these are noises. So this converging of the mesh causes a significant number of noises to be created during these gestures of the users. But luckily as this is not a constant error and observed only in some of the frames during the gesture and not in all the frames we can still detect events during these periods if we can filter out the frames those are having noises.

It has been observed that though for real events the z-scores are above the threshold (so that we can detect them) they are not too high and mostly remain in the ranges of 2.00 to 4.00. But as the mesh displaced significantly when it gets converged to a squeezed or expanded shape, it is quite intuitive that the position changes of different points of the face will be large enough so that it can be easily distinguished from the real events. And so the z-scores for different regions for the frames giving these noises will be much higher than the z-scores we usually get for the real events. This feature makes us able to distinguish the noises during these gestures from the real events.

To make our system automatically distinguish these noises using the feature mentioned we just need to set an upper limit for the threshold and consider that frames having z-scores above this upper limit contains noises and the frames having z-scores above our previously set value but below this upper limit contains real events. We call this upper limit of threshold that we need to set for this as ***UpThreshold***. So, now our event function will be:

$$E(z) = \begin{cases} 0, & z < thr \\ 1, thr \leq z < UpThr & \\ 0, & z \geq UpThr \end{cases} \qquad (11)$$

Chapter 5

Data Collection

Continuous Emotion Recognition is to be robust enough to capture real life scenarios. Real life scenarios contain spontaneous facial behavior. Modeling spontaneous facial behavior is the key challenges in analyzing emotions. Lack of spontaneous facial data is one of the most important reasons for slow progress in this area. Spontaneous facial behavior includes various combinations of facial expressions that can be different from combinations in posed expressions. In deliberate expression data-set such as Cohn-Kanade dataset [34], subjects are asked to display certain facial expressions. Moreover, they are instructed to display single or combination of AUs. In the situation where subjects are asked to display facial behavior in these ways, spontaneous facial behavior is rare.

In addition, the temporal evolution of natural facial expressions is different from those in prototypic and posed expressions. The reason is that in spontaneous behavior, facial expressions are more complex and transitions between expressions do not have to involve intermediate neutral state.

To cope with the real situation, recognition systems must be able to model spontaneous facial behavior. Moreover we are interested to model complex and learning centered emotions (confusion, frustration) rather than basic emotions and measure our system's performance response with the facial expressions those accompany with the stated emotions.

However, spontaneous facial expression data is rare compared to posed expressions. Although a very few data sets are available those are recorded in natural settings but no such previous natural video data sets were recorded in a learning context. Unavailability of spontaneous facial behavior data in natural environment and in learning context derives us to collect and annotate data from the video data sets those were recorded during two previous research study conducted by our research group.

Both of these researches were conducted in natural settings and all the recorded videos were fully spontaneous with no prior instructions and no constraints on the natural movement were imposed on the participants. First set of videos were recorded when the participants were involved in effortful problem solving tasks which is expected to induce learning centered emotions like confusion, frustration or boredom. The second set was recorded while the participants were experiencing cognitive disequilibrium that was induced by contradicting assertions made by two animated agents on some scientific topics and this is expected that they will exhibit confusions as a result of this disequilibrium.

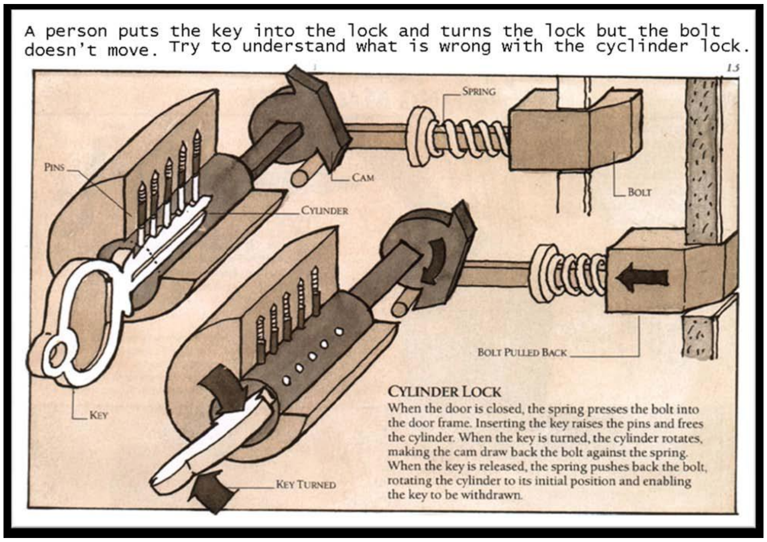
Details descriptions of each of the data sets are given below:

5.1 Inducing Confusion with Breakdown Scenarios

This study investigated the role of *cognitive disequilibrium* during complex learning and reasoning in a task [35], [36]. Complex learning and reasoning occur in effortful problem solving tasks as well as other tasks that require a person to comprehend difficult technical materials, to solve difficult problems, and to make difficult decision.

5.1.1 Data Collection Context

In two experiments, participants read four illustrated texts on everyday devices: a cylinder lock, an electric bell, a car temperature gauge, and a toaster. Descriptions of the device mechanisms along with illustrations were extracted from a book of illustrated texts titled *The Way Things Work* (Macaulay, 1988). The illustrated texts contained sections in printed text, visual diagrams of the components of the device, labels of major components, and directional arrows that convey motion or temporal changes. A breakdown scenario was prepared for each of the four devices. The breakdown scenario consisted of one or two sentences that identified physical symptoms of a device malfunction [35], [36]. For example, in the case of the cylinder lock, the breakdown scenario was *“A person puts the key into the lock and turns the lock but the bolt doesn’t move”*. Image of breakdown scenario for a cylinder lock is given in figure 5.1



A person puts the key into the lock and turns the lock but the bolt doesn't move. Try to understand what is wrong with the cyclinder lock.

CYLINDER LOCK

When the door is closed, the spring presses the bolt into the door frame. Inserting the key raises the pins and frees the cylinder. When the key is turned, the cylinder rotates, making the cam draw back the bolt against the spring. When the key is released, the spring pushes back the bolt, rotating the cylinder to its initial position and enabling the key to be withdrawn.

Fig 5.1 Breakdown Scenario

The plausible faults are the following: The cam is broken, the rod that the cam is hooked over is broken, or the intersection between the cam and its connecting rod could be broken, slipping, or not connected in some way.
The hypothesis was that these device breakdowns would induce cognitive disequilibrium in the minds of the participants. We tested this hypothesis in two experiments.

5.1.2 Data Collection Method

The experiment had a within-subjects design in which participants studied all four devices in four phases (for each device). In phase 1, participants read an illustrated text of a device for 1.5 minutes. They were then presented with either a breakdown scenario for the device (experimental condition) or the same illustrated text without a breakdown scenario (control condition) for another 1.5 minutes (phase 2). Next, they were given 30 seconds to recall all the components of the device in order of importance (phase 3). Finally, in phase 4, they completed a three-item affect rating scale that asked them to self-report their levels of confusion, engagement, and frustration.

These four phases were repeated for all four devices (two for experimental group and two for the control group). The assignment of devices to conditions and the presentation order of devices and conditions were counterbalanced across participants with a Latin Square.

5.1.3 Data Capture and Extraction

Once all four phases are over, participants then viewed videos of their faces and computer screens and provided *offline* continuous assessments (sampled 1 Hz) of their confusion levels using a *confusion dial*. The dial is a software program that allows participants to provide continuous confusion assessments on a scale from 1 (not confused) to 10 (very confused) while watching videos of their faces that were recorded while they studied each device (phase 2).

There were 52 participants in Experiment 1 and 46 participants in a subsequent replication (Experiment 2). Videos of participants' faces were recorded during phases 1 and 2 via a web-cam that was integrated into the computer monitor. There was a difference between the two experiments in the device presentation time, which was increased to two minutes per device in Experiment 2 (see phases 1 and 2 above).

As each of the participants were presented with each four of the devices and videos were recorded for first two phases for each of the devices a total of 8 videos were recorded for each participants. So a total of 52*8 = 416 videos with 1.5 minutes length for each were collected during experiment 1. And a total of 46*8 = 368 videos were collected in experiment 2 where half of them were 1.5 minutes long and half of them were 2 minutes long.

5.1.4 Data Annotation

Each of the videos of our data set was 1.5 to 2 minutes long as mentioned earlier. All of these videos were manually annotated by two human raters individually. When watching through each of the videos if at any moment the rater observed any subtle changes in mouth or eye brow region or in both of the regions he/she paused the video and recorded the frame number the change was initiated and then resumed the playing until the video reached at the frame when the change faded away and then paused again. The rater than recorded the region (Mouth, Brow or both Mouth and brow) where the change was occurred along with the frame numbers of the initiation and offset point of it. A voluntary comment field was also provided for them so that they could keep notes on something they thought significant enough to be recorded. Some sample records are shown below:

10	17	Mouth	
17	19	Mouth-Eyebrow	
74	75	Mouth ;	Head is tilted and resting on the hand
82	85	Eyebrow;	mouth is occluded by hand

Here for the last two entries we have seen that some comments have been recorded.

5.2 Inducing Confusion by Contradictory Assertions

This study was conducted based on the hypothesis that contradictions between animated pedagogical agents playing the roles of tutor and student would induce cognitive disequilibrium and confusion in the minds of the human learner. This study was conducted to experimentally induce confusion in the minds of learners with a *contradictory information* manipulation.

5.2.1 Data Collection Context

This desired contradiction is achieved by having the tutor agent and the student agent staging a disagreement on an idea and eventually inviting the human to intervene (note that student agent refers to an animated conversational agent; the actual learner is referred to as human learner or human). The contradiction is expected to trigger conflict and force the human to reflect, deliberate, and decide which opinion has more scientific merit. The contradictions were introduced during trialogues as the animated agents and the humans attempted to identify flaws in research studies. Some studies had subtle flaws while others were flawless. This made the flaw detection task quite challenging. Each problem included a description of a research study and humans were required to critique the study.

5.2.2 Data Collection Method

There were four contradictory information conditions in this study. In the *True-True* condition, the tutor agent presented a correct fact and the student agent agreed with the tutor; this is the no contradiction control. In the *True-False* condition, the tutor presented a correct fact and the student agent disagreed by providing an incorrect fact. In contrast, it was the student agent who provided the correct fact and the tutor agent who disagreed with an incorrect fact in the *False-True* condition. Finally, in the *False-False* condition, the tutor agent provided an incorrect fact and the student agent agreed with this incorrect fact. The human learner was asked to intervene after each contradiction and there were four opportunities for contradictions in each problem. It should be noted that misleading information from these manipulations was always eventually corrected before learners completed their interactions. Moreover the learners were fully debriefed at the end of the experiment.

The experiments involved a within-subjects design, so participants were exposed to all four manipulations. Next they read a short text introducing the eight critical thinking concepts (e.g., control group, replication and construct validity) that were to be covered in the learning session. Participants and the animated agents next engaged in trialogues where they attempted to find flaws in eight studies (one concept was covered in each study and there were two studies for each condition). Half the studies had flaw while the others used good methodologies.

Assignment of condition to problem and ordering of conditions and problems was counterbalanced across participants with a Latin square. Participants completed a multiple choice posttest after the tutorial session.

5.2.3 Data Capture and Extraction

Participants judged their emotions immediately after they completed the posttest via a retrospective affect judgment procedure. The procedure began by synchronizing and displaying the videos of the participants' face and screen that were captured during the interaction. The videos paused at critical junctures in the tutorial session (immediately after contradictions, when participants had to chime in with their opinion, etc); participants were required to provide affect judgments at these points. They could also provide judgments on their emotions at any time during the session by pausing the videos manually. Participants were provided with a list of the affective states and definitions. The list included confusion, flow/engagement, boredom, frustration, curious, anxious, surprise, and neutral.

There were 32 participants in the first experiment and 64 participants in the second experiment. Unlike our breakdown study in this study only one video was recorded per participant and that single video captures all the phases the participant went through. So each video were almost 1 hour long and there was a total of 32+64= 96 videos.

Chapter 6

Experimental Results for Facial Event Detection

This chapter illustrates the efficacy and the robustness of our system in predicting facial movements in mouth and eye brow regions. Our breakdown study data set was used to quantify the performance of the proposed system. Since most of the state-of-the-art facial tracking system focus on AU detection or are based on posed data set or detects only basic emotions recognition, it is difficult to compare the reported results with our proposed work.

Subsequent two sections of this chapter are devoted to two different aspects of this evaluation. Section 6.1 asses the facial event detection accuracy of the system and section 6.2 assesses the generalization possibility of the proposed system.

6.1 Event Detection Accuracy

For performance evaluation of our systems subtle event detection accuracy each of the 93 videos was fed into our system one by one and the output was recorded as well. Our system runs each of the videos and analyzes it frame by frame to detect if there is any changes in mouth or eye brow regions and for each of the detections it records the frame number the change first initiated and the number where it faded away. When it is done with one video it then converts the frame number into second number by dividing it by the FPS of the videos (which is 12 in our case). In this way we can know for each of the seconds whether there were any eye or mouth movements during that second or the face was in neutral status during that second.

The final output that the system thus produces contains one line for each of the seconds of each of the videos. And each of the line has four fields in it and these are the name of the video file followed by a number that indicates for which second this entry is written and then two other fields those have a 1 or 0 in them. These 1 or 0 in the third and fourth field signifies if there was a mouth or brow movement detected during the second being considered. The third field is for the mouth and a ‘1’ indicates that a mouth movement has been detected during that second and a ‘0’ indicates otherwise. Similarly a ‘1’ in the fourth field indicates there was a detected brow movement during that second and a ‘0’ indicates that the brow was neutral during that second.

Now to measure the accuracy of our system it is needed to compare these detected events with the annotated ones and check their level of agreements in the occurrences of these events. But our annotation was done in frame levels and our system’s output is in second level. To make a meaningful comparison between the two we need to represent both of them in the same scale of measurement. Hence we have converted our annotated events to second level by dividing each of the event occurring frame numbers by the FPS and merging the ones those reside in the same second. And finally we represent the annotated events in the same way as we have represented our detected events, meaning enlisting each second of each video in one single line with four fields in each of the line and the fields are video name, second number, a ‘1’/’0’ for the mouth event and a ‘1’/’0’ for brow event. Once the annotated events and detected events are represented in the same way it is now an easy task to compare them. It is obvious that the annotated event set is our ground truth and we have to compare our detected events with the annotated ones and check how much off they are. To do this we have checked each of the seconds in our annotated events file and checked the corresponding seconds in the detected events file to see if the events match or not. For each of the seconds if we see a ‘1’ for mouth event in the annotated file and a ‘1’ in the detected file as well we consider this as a hit and if instead we see a ‘0’ in the detected file we consider this as a miss. Alternatively if we see a ‘0’ for mouth event in the annotated file and a ‘0’ in the detected file as well we consider it as a true negative and if instead we see a ‘1’ in the detected file we consider it as a false alarm. We followed the same reasoning for the eye brow as well.

As we have 93 videos and each of them are 120 seconds (2 minutes) long that means we have 93*120 = 11160 data points for our analysis where each of these points give us a hit, a miss, a false alarm or a true negative for mouth and a hit, miss, false alarm or true negative for eye brow as well. We constructed our confusion matrix out of all these 11160 data points and it is shown below in Table 6.1 for mouth and in Table 6.2 for eye brow.

Table 6.1: Confusion Matrix for Mouth

		Annotated	
		Event (1)	No Event (0)
Detected	Event (1)	894	849
	No Event (0)	222	9195

Table 6.2: Confusion Matrix for Eye Brow

		Annotated	
		Event (1)	No Event (0)
Detected	Event (1)	89	773
	No Event (0)	29	10269

We calculated our performance measurements from these confusion matrixes and these are namely hit rate, false alarm rate, dprime, recall, precision and F-measure. Values of all these measurements along with the formulas for them are summarized in Table 6.3 for mouth and eye brow.

Table 6.3: Performance Metrics values

	Formula	Mouth	Eye Brow
Hit Rate	$\frac{TP}{TP+FN}$	0.80	0.75
False Alarm Rate	$\frac{FP}{TN+FP}$	0.08	0.07
d Prime	*Z(Hit Rate)-Z(False Alarm Rate)*	2.25	2.15
Recall	$\frac{TP}{TP+FN}$	0.80	0.75
Precision	$\frac{TP}{TP+FP}$	0.51	0.07
F-Measure	$\frac{2*Precision*Recall}{Precision+Recall}$	0.62	0.12

We detected 894 out of the 1116 valid mouth movements and 89 out of the 118 valid brow movements, thereby yielding hit rates of .80 and .75 for the brow and mouth, respectively. We have also calculated the false alarm rate and the corresponding d-prime value to test whether we are having this accuracy out of chance. We have found that our false alarm rate is only 0.08 yielding a d-prime of 2.25. And for eye-brow our false alarm rate and d primes are 0.07 and 2.15 respectively. We consider having hit rates of 80% and 75% as where the d primes are in the ranges of 2.15 to 2.25 as promising enough to use our tool in subtle facial expression detection applications.

6.2 Assessment of Generalization

In the system description section we have seen that our system needs a number of different parameters to be set with predefined values in most of the steps of its workflow namely Model Length or different thresholds etc. Initially we set these values manually based on our observation and domain knowledge. These includes analyzing the facial videos of the participants and our tool's response to them, participant's natural gesture and posture trends while using the computer tool and tracking how our system response changes with that, measuring the amount of deformation the mesh experiences during real facial movements and also in case of noises and overall by statistical analysis of our systems output data that it generated while it was fed in with our video data set.

It is of no doubt that all these tasks are rigorously time consuming and require a lot of domain knowledge and most importantly they are data set dependent meaning that these set values will only work for the data set we have used and if new data set has to be used all these works needed to be redone to find a new set of values. These limitations undermine the possibility that our tool can be widely used with other data sets and systems. Therefore to make the system a better fit for future applications and widely acceptable to facial expression recognition community it is needed that we should find out a unique set of values for our parameters that can be used universally and with any data set and still gives the same performance. So that it can be universally used without any such prior training or parameter lookup process.

Table 6.4 : Description of Parameters

1	Model Length	Number of frames in the model
2	Buffer Length	Number of frames to wait before entering them in the model
3	Number of Grouped Frames	Number of frames to group them together and check there zscore values agains model. This was done to reduce noise by taking the mean zscore of a number of frames instead of taking a single frame.
4	X	Number of contigious frames to check if a predefined number of frames (Y) withing these (X) frames are in events to consider the event as a legitimate one.
5	Y	Minimum number of frames those must be in event between (X) number of contigious frames to consider an event a legitimate one.
6	Mouth / Eye Brow Threshold	If zscore is above this value, consider an event has occurred.
7	Upper Limit for threshold	If the zscore is above the value of threshold+upper limit, then consider the zscore as too high and don't consider it as a legitimate event.
8	Head Threshold	If the zscore value of the cener nose point is above this threshold then consider a head movement is occurred and discard any events in the neighbouring frames.
9	Frames to ignore in head movement	If there is a head movement in a frame this number of frames should be discarded following that frame (having head movement) due to extensive noise in them.

For that we performed a onetime calibration on our system that was used to find out that unique set of values for the parameters. To do this we first divided our video data set into two subsets. A details description of our data set is given in chapter 5. In order to keep the equality of two sub sets videos were clustered among them in such a way that each sub sets are counterbalanced in terms of the number of videos or subjects as listed below:

1. Number of Subjects and Videos.
2. Number of Males and Females.
3. Number of subjects with bright skin color.
4. Number of subjects with dark skin color.
5. Number of seconds having real (annotated) mouth movements.
6. Number of seconds having real (annotated) brow movements.

We named the two sub sets as set A and set B respectively. Table …. shows the breakdown of all these numbers as they were distributed among the two sets:

Table 6.5 : Distribution of Videos

	VIDEOS	**SUBJECTS**	**MALE**	**FEMALE**	**Bright Skin Color**	**Dark Skin Color**	**Mouth Event Seconds**	**Brow Event Seconds**
SET A	50	19	3	16	13	6	547	62
SET B	43	19	4	15	13	6	569	56

We then consider one subset of videos as training set and other as the testing set. Initially set A was considered as training set and set B as testing set. We then fed the test set videos in our system to find out the best values for each of the parameters following an iterative approach. At each of the iteration we ran our system through all the videos of the test set and track its performance for a wide range of values for each of the parameters and tune up the values accordingly. Based on this tuning the next iteration begins with a new range (possibly reducing the width of the range) of values for the parameters. We continue this iteration until we get a unique set of values for the parameters that we think gives us the best performance.

This performance is measured in terms of hit rate, false alarm rate, dprime and also in true positive, false negative, f-measure metrics. For mouth these values were hit rate = 0.76, false alarm rate = 0.08, dprime = 2.13, recall = 0.76, precision = 0.50 and F-measure = 0.60. And for Brow the values were hit rate = 0.74, false alarm rate = 0.07, dprime = 2.11, recall = 0.74, precision = 0.10 and F-measure = 0.17.

Once after some iteration we are settled with a unique value for each of the parameters, we then run our system through all the videos of the test set. But this time using the unique parameter values those we obtained by our iterations on the training set. Our idea was that the best performing parameter values thus obtained can be considered as viable ones only if it can be proved that they have similar impact on the system’s performance with any previously unseen data set.

As these values were obtained using only the videos of our training data set, all the videos of our test data set are unseen to the system. So if we can get similar performance responses from our system using these parameter values with the test data set as well we can assert that these parameter values should behave uniquely for any data set without any prior training.

Performance measures those we obtained with the 43 videos of our test data set are: for mouth hit rate = 0.79, false alarm rate = 0.08, dprime = 2.19, recall = 0.79, precision = 0.55 and F-measure = 0.64. And for Brow the values were hit rate = 0.63, false alarm rate = 0.09, dprime = 1.67, recall = 0.63, precision = 0.07 and F-measure = 0.13. So all the measurement are almost same as those we obtained for our training data set. This confirms that the values those we set for the parameters after our tuning iterations phase behaves identically for any data set and hence can be considered as universally applicable.

In order to make our assertions more concrete we decided to repeat our calibration process but this time by swapping our training and test data set. So now for our second calibration process set A was considered as test set and set B was considered as training set.

Then the whole calibration process was repeated and obviously the tuning iteration was performed on set B and the performance measurement was as follows: for mouth, hit rate = 0.85, false alarm rate = 0.07, dprime = 2.55, recall = 0.85, precision = 0.62 and F-measure = 0.71. And for Brow the values were hit rate = 0.75, false alarm rate = 0.10, dprime = 1.88, recall = 0.75, precision = 0.07 and F-measure = 0.12. And once we ran the system through the videos of set A (which is our test set now) using the tuned parameter values the measurements are : for mouth, hit rate = 0.75, false alarm rate = 0.10, dprime = 1.94, recall =

0.75, precision = 0.42 and F-measure = 0.54. And for Brow the values were hit rate = 0.69, false alarm rate = 0.10, dprime = 1.74, recall = 0.69, precision = 0.06 and F-measure = 0.11.

So again with the swapped data set we have seen that our system gives the same performance with training and testing data set when we use the tuned parameter values. This strongly confirms the validity of our method for deducing universal values for the parameters and the validity of the values as well.

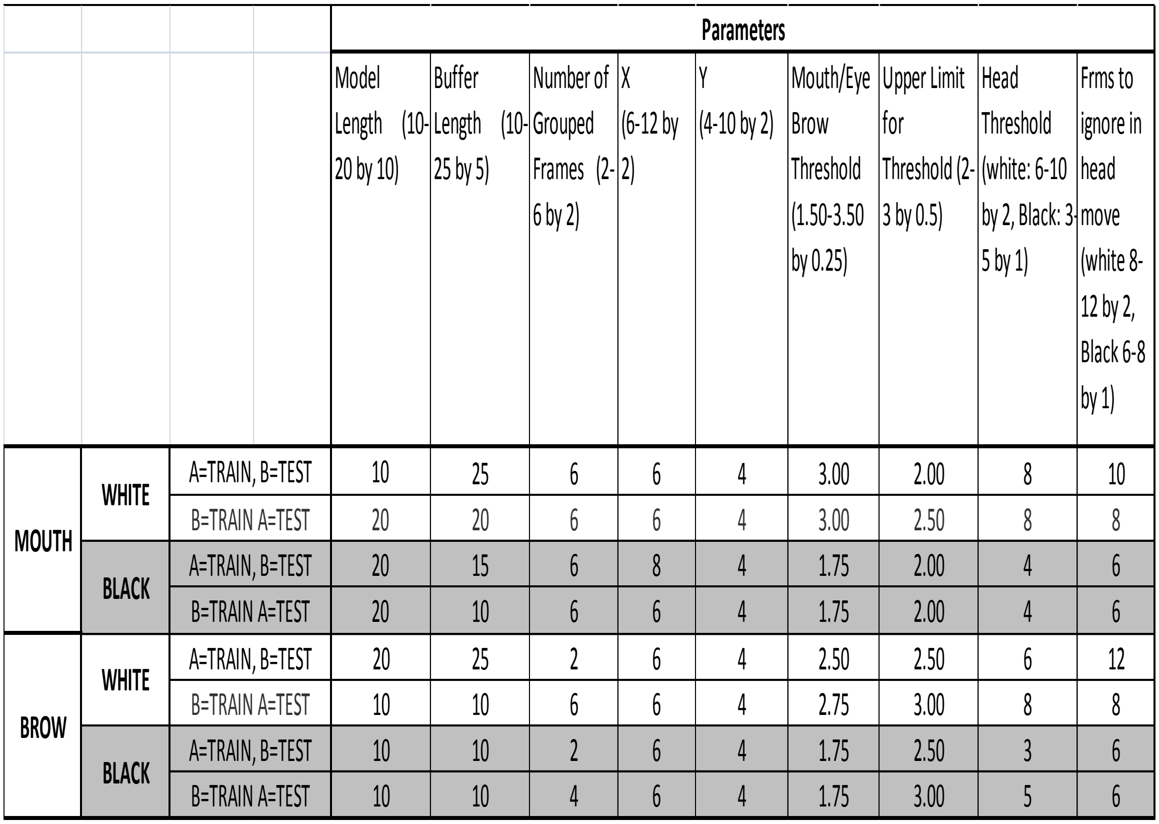
A summary of all these performance metrics for both our calibrations are shown in Table 6.6:

Table 6.6 : Performance of Training and Testing

			Hit Rate	**False Alarm**	**dPrime**	**Recall**	**Precision**	**F-measure**
MOUTH	A= TRAIN B=TEST	Train Set	0.76	0.08	2.13	0.76	0.50	0.60
		Test Set	0.79	0.08	2.19	0.79	0.55	0.64
	B=TRAIN A=TEST	Train Set	0.85	0.07	2.55	0.85	0.62	0.71
		Test Set	0.75	0.10	1.94	0.75	0.42	0.54
BROW	A= TRAIN B=TEST	Train Set	0.74	0.07	2.11	0.74	0.10	0.17
		Test Set	0.63	0.09	1.67	0.63	0.07	0.13
	B=TRAIN A=TEST	Train Set	0.75	0.10	1.88	0.75	0.07	0.12
		Test Set	0.69	0.10	1.74	0.69	0.06	0.11

In addition to the uniformity in output performances with both training and testing sets, another significant finding is the tuned parameter values those were generated during our first round of calibration are exactly the same or very close to the parameter values generated during our second round. A summary of all these parameter values for both of our calibration rounds are given in Table 6.7:

Table 6.7: Tuned values for the Parameters



			Parameters								
			Model Length (10-20 by 10)	Buffer Length (10-25 by 5)	Number of Grouped Frames (2-6 by 2)	X (6-12 by 2)	Y (4-10 by 2)	Mouth/Eye Brow Threshold (1.50-3.50 by 0.25)	Upper Limit for Threshold (2-3 by 0.5)	Head Threshold (white: 6-10 by 2, Black: 3-5 by 1)	Frms to ignore in head move (white 8-12 by 2, Black 6-8 by 1)
MOUTH	**WHITE**	A=TRAIN, B=TEST	10	25	6	6	4	3.00	2.00	8	10
		B=TRAIN A=TEST	20	20	6	6	4	3.00	2.50	8	8
	BLACK	A=TRAIN, B=TEST	20	15	6	8	4	1.75	2.00	4	6
		B=TRAIN A=TEST	20	10	6	6	4	1.75	2.00	4	6
BROW	**WHITE**	A=TRAIN, B=TEST	20	25	2	6	4	2.50	2.50	6	12
		B=TRAIN A=TEST	10	10	6	6	4	2.75	3.00	8	8
	BLACK	A=TRAIN, B=TEST	10	10	2	6	4	1.75	2.50	3	6
		B=TRAIN A=TEST	10	10	4	6	4	1.75	3.00	5	6

These similarities in parameter values serve as a profound foundation for our claim that our system along with these parameter values can be used with any facial video data sets without the need of any prior training and it will still be able to detect subtle facial expressions with the same accuracy level as it has detected with our data set.

Chapter 7

Experimental Results for Emotion Recognition

Empirical analyses using the mentioned datasets consisting of varying degrees of complexities and variability were used to illustrate the utility of the proposed approach. In this chapter the utility has been assessed in terms of the efficacy of our system in diagnosing user's learning centered emotion namely confusion through the different facial event sets that it captures.

We conducted our analysis focusing this issue by using both of our data sets. Section 7.1 is devoted to represent different analysis results those were obtained by running our system on the breakdown study data set and aligning its detected event sets to the self reported emotions of the participants. And in section 7.2 we presented the analysis with our induced emotion through contradiction data set where we investigated how well our system's detected events are able to predict the participants' confusion those were induced by cognitive disequilibrium.

7.1 Analysis with Self Reported Emotions

We conducted our study on our breakdown data set and the idea was to check if the facial events those our system is detecting exhibit any relationship to the self reported emotions of the participants. Different approaches those are followed to analyze these relations are given in the following sub sections:

7.1.1 Correlation Study

As the participants made retrospective affect judgments by an affect dial while watching their own facial vides, we have taken an approach of finding the correlations between those rated values with the number of seconds each video have with any mouth or brow events. As all of our videos were not of same length, instead of taking the summation of affect ratings and number of seconds with events we took the average of those numbers by simply dividing the summation and number of seconds with events by total length (in seconds) of the video. Then the correlation was calculated using those averaged numbers. In addition to finding correlation between affect ratings and number of seconds with brow or mouth events individually we also considered the correlation with the seconds those have both of mouth and brow events or have either of the two.

We have a couple of parameter combinations those we generated in our calibration process. Our correlation study was conducted by each of those parameter value sets. For all the pairs correlation with mouth events are in the range of 0.20 to 0.29, with brow events the range is from 0.18 to 0.29 , and with the seconds having both mouth and brow events this range is from 0.09 to 0.23. While with the annotated events these correlations are 0.12, 0.11 and 0.20 for mouth, brow and for mouth & brow respectively. These numbers are in tabular format in Table 7.1.

So, although the correlations are positive but there is not that much strong correlations (lies between 0.2-0.3 in roughly). But correlations with the annotated events are not that much strong either. So, though the correlation values of our system are small but they are almost same as the annotated ones. Specifically for one case, correlation with the seconds having both mouth and brow events is exactly same for our system and for annotated ones (this case is in bold in the table).

Table 7.1: Correlation Values

	Correlation With Detected Events				**Correlation With Annoatated Events**			
	With Seconds Having **Mouth** Events	With Seconds having **Brow** Events	With seconds having **Mouth AND Brow** Events	With seconds having **Mouth OR Brow** Events	With Seconds Having **Mouth** Events	With Seconds having **Brow** Events	With seconds having **Mouth AND Brow** Events	With seconds having **Mouth OR Brow** Events
Mouth Params of set A. Brow Params of set A	0.2	0.24	0.13	0.26	0.12	0.11	0.2	0.11
Mouth Params of set A Brow Params of set B	0.2	0.23	0.09	0.25	0.12	0.11	0.2	0.11
Mouth Params of set B Brow Params of set A	0.29	0.24	0.17	0.3	0.12	0.11	0.2	0.11
Mouth Params of set B Brow Paramsof set B	**0.29**	**0.23**	**0.2**	**0.3**	**0.12**	**0.11**	**0.2**	**0.11**
Mouth Params of set A have used as common params for mouth and brow	0.2	0.18	0.13	0.24	0.12	0.11	0.2	0.11

	Correlation With Detected Events				Correlation With Annoatated Events			
	With Seconds having **Mouth** Events	With Seconds having **Brow** Events	With Seconds having **Mouth AND Brow** Events	With Seconds having **Mouth OR Brow** Events	With Seconds having **Mouth** Events	With Seconds having **Brow** Events	With Seconds having **Mouth AND Brow** Events	With Seconds having **Mouth OR Brow** Events
Mouth Params of set B have used as common params for mouth and brow	0.29	0.29	0.23	0.35	0.12	0.11	0.2	0.11
Brow Params of set A have used as common params for mouth and brow	0.29	0.24	0.22	0.32	0.12	0.11	0.2	0.11
Brow Params of set B have used as common params for mouth and brow	0.25	0.23	0.18	0.32	0.12	0.11	0.2	0.11

7.1.2 Analysis with window of seconds

Instead of taking only a single second associated with a mouth or brow changes, we considered a window of neighboring seconds for any mouth/brow changes. The average value of ratings for all these neighboring seconds was considered as the rating for the second having the mouth/brow event. Neighboring means taking some seconds before the second where the event took place and taking some seconds after that. Then we took the length of this window as a variable and ran through for different lengths to find the length with best performance. Taking a window of 15 neighboring seconds gave us the best performance. Two types of analysis were done following this approach:

1. Recall/Precision analysis for detecting a subject is confused or not.
2. Comparing the ratings associated with the seconds with events (this means ratings when there is an event) with those not with events.

Findings for both of the approaches are summarized below:

7.1.2.1 Recall/Precision analysis

A hit is considered if the confusion rating is above a threshold (so that we can take the subject as confused) in a second and we have an event at that second. A miss is considered if the rating is above the threshold but there is no event in that second. A false alarm is considered if the rating is below the threshold (that means not confused) but there is an event. This was done with all four area combinations (mouth, brow, mouth AND brow, mouth OR brow) and for different threshold values. For all the thresholds best performance was obtained considering “mouth OR brow” events. Summary of the results obtained with different threshold values given below:

a. Taking the threshold as 6:
b. 6 is above the midpoint (which is 5) of the rated affect values. So we considered someone as confused if his rating is above or equal to 6 and not confused if the rating is below 6. Results are summarized in Table 7.2.

Table 7.2: Metrics Values for Windowed Frames (Threshold 6)

	Detected			Annotated		
	Recall	Precision	F-Msr	Recall	Precision	F-Msr
Mouth	0.59589	0.387816	0.469847	0.567035	0.308716	0.399778
Brow	0.694653	0.365569	0.479039	0.179476	0.297086	0.223769
MouthANDBrow	0.308168	0.458845	0.368707	0.095708	0.27599	0.142129
MouthORBrow	0.798291	0.359244	0.495504	0.622766	0.321952	0.424467

c. Threshold as 4.5:

During our analysis of comparing ratings with and without event, we found that for all the cases (all window size, for all area, detected or annotated) the average ratings associated with an event window is always greater than 4.5 and average ratings associated with non event seconds is always less than 4.5. This finding made us believe that taking a threshold of 4.5 will improve our result, which is totally reflected what we found and the results are summarized in Table 7.3

Table 7.3: Metrics Values for Windowed Frames (Threshold 4.5)

	Detected			Annotated		
	Recall	Precision	F-Msr	Recall	Precision	F-Msr
Mouth	0.509341	0.513125	0.511226	0.583293	0.523186	0.551607
Brow	0.596684	0.510004	0.549949	0.183841	0.540156	0.274319
MouthANDBrow	0.217116	0.548526	0.311096	0.09561	0.485149	0.159739
MouthORBrow	0.738424	0.50117	0.597092	0.690686	0.519783	0.59317

d. With dynamic threshold:

It has been observed that there is a difference in the way each subject rate themselves. Like, some subjects rating ranges from only 0 to 2, on the other hand some rated only in the range of 6-10. So instead of taking a fixed threshold for all subjects we considered the average value for a subject's rating as his threshold. That is when the ratings of a participant is above the average rating value of that same participant we consider him/her to be confused, and when it is below his average rating value, we consider him/her as not confused. Thinking in this way gives us a little improvement as reflected in Table 7.4.

Table 7.4: Metrics Values for Windowed Frames (Dynamic Threshold)

	Detected			Annotated		
	Recall	Precision	F-Msr	Recall	Precision	F-Msr
Mouth	0.508204	0.590639	0.546329	0.526192	0.528338	0.527263
Brow	0.612206	0.602126	0.607124	0.160187	0.535892	0.246647
MouthANDBrow	0.203637	0.591523	0.302973	0.076874	0.449257	0.131284
MouthORBrow	0.753568	0.591441	0.662733	0.710602	0.542478	0.615262

7.1.2.2 Comparing the ratings associated with the seconds with events with those not with events

And for our second analysis (comparing ratings of event window with those without event) it was found that for all cases average ratings associated with events is always greater than average ratings of seconds associated with non events. This implies that it is always true that whenever there is a change in the mouth or eye brow, someone is more confused compared to the cases whenever there is no changes in face. The result is reflected for both our detected and annotated events. For example, results for different window lengths are listed in Table 7.5.

Table 7.5: Comparison between Seconds with events to Seconds with no events

		Detected		Annotated	
Window Length	Area	With Events	Without Events	With Events	Without Events
1	Mouth	5.15	4.44	4.60	4.51
	Brow	4.92	4.46	5.05	4.52
	MouthANDBw	5.05	4.51	5.41	4.52
	MouthORBrow	5.03	4.38	4.60	4.52
5	Mouth	5.11	4.36	4.77	4.47
	Brow	4.95	4.36	5.20	4.51
	MouthANDBrw	5.09	4.48	5.48	4.52
	MouthORBrow	5.02	4.26	4.78	4.47
10	Mouth	5.02	4.27	4.78	4.38
	Brow	4.83	4.24	5.14	4.49
	MouthANDBrw	5.07	4.42	5.36	4.51
	MouthORBrow	4.88	4.13	4.78	4.37

7.2 Analysis with Emotions Induced by Contradiction

Goal of this analysis was to asses if the facial event sets those our system detects can be considered as diagnostic of confusion. For this we have compared our system’s response during the cognitive disequilibrium episodes with the neutral episodes. And as mentioned, this cognitive disequilibrium was induced by contradictory assertions made by two animated agents. And the contradictory episodes are the TF, FT and FF conditions and the neutral one is TT condition. Below is the summary of key findings of this study:

1. There is a significant difference in the amount of Mouth and Brow movement when the Agents (Teacher or Student) were making contradictory assertions. As expected, when False (F) assertions were made (when we expect subject will get confused) much more Mouth and Brow movements were recorded in compared to the episodes when True (T) assertions were made (when we expect subject will NOT get confused).
2. As expected, this difference is not observed for Non contradictory (P-TT , P-FF, N-TT and N-FF) conditions. That is when the agents are making same assertions then there is not that much difference in Mouth or Brow movements.
3. For N-FF condition, that is when there is no problem in the concept and both the agents are making False assertions about it, the amount of movement in both Mouth and Brow is much more higher for all types of episodes (Assertion, Pause and Poll) in comparison to other N conditions (N-TT,N-TF,N-FT). This is also a good result thinking that when there is no problem in the concept and both of the agents are asserting falsely that there is some problem with it, this should make the subject much confused.

Table 7.6: Detected facial events comparison for all contradictory conditions

			Mouth	**Brow**	**Mouth OR Brow**	**Mouth AND Brow**
P	TT	**Asserta**	**10.97**	**12.96**	20.03	3.81
		Pause	13.79	14.94	21.84	5.75
		Assertb	**12.71**	**10.48**	20.13	2.44
		Poll	16.86	11.88	22.61	4.98
	TF	**Asserta**	**13.03**	**10.96**	21.12	2.7
		Pause	18.39	17.24	27.58	6.9
		Assertb	**21.43**	**20.28**	32.02	9.37
		Poll	17.25	8.81	23.38	2.3
	FT	**Asserta**	**15.96**	**16.82**	22.04	2.81
		Pause	15	20.69	26.44	6.9
		Assertb	**7.13**	**11.80**	20.45	3.76
		Poll	19.55	14.18	26.44	6.9
	FF	**Asserta**	**12.3**	**8.55**	17.49	3.36
		Pause	17.24	9.2	21.84	4.6
		Assertb	**13.25**	**11.82**	22.45	2.63
		Poll	11.5	12.27	22.61	0.77
N	TT	**Asserta**	**12.79**	**10.87**	19.87	3.62
		Pause	12.64	9.19	19.54	2.3
		Assertb	**11.73**	**9.97**	18.32	3.33
		Poll	12.65	9.2	18.78	3.07
	TF	**Asserta**	**10.44**	**8.43**	20.17	2.26
		Pause	13.79	13.79	25.29	2.30
		Assertb	**16.10**	**13.17**	20.95	3.48
		Poll	18.01	16.87	26.83	6.13
	FT	**Asserta**	**12.58**	**16.65**	23.11	6.12
		Pause	11.49	11.5	19.54	3.45
		Assertb	**12.47**	**10.27**	19.19	3.56
		Poll	12.26	13.8	22.23	3.07
	FF	**Asserta**	**20.86**	**20.72**	32.84	8.48
		Pause	26.45	26.44	39.08	11.5
		Assertb	**18.61**	**19.77**	31.13	6.73
		Poll	16.86	21.08	30.66	6.13

Graphical representations of our findings are provided in bar charts. Bar chart in Figure 7.1 and 7.2 shows the percentage of time there was a Mouth or Brow movement during AssertA (Teaching Agent's assertion) and AssertB

(Student Agent's assertion) episodes for all of TT,TF,FT and FF conditions of P(with Problems) types of concepts. Figure 7.1 is for mouth and 7.2 is for eye brow.

And Figure 7.3 and 7.4 shows the same findings for N (with No Problems) types of concepts. These charts shows our 1st two findings, that is for contradictory assertions, there is a significant difference in Mouth and Brow movements.

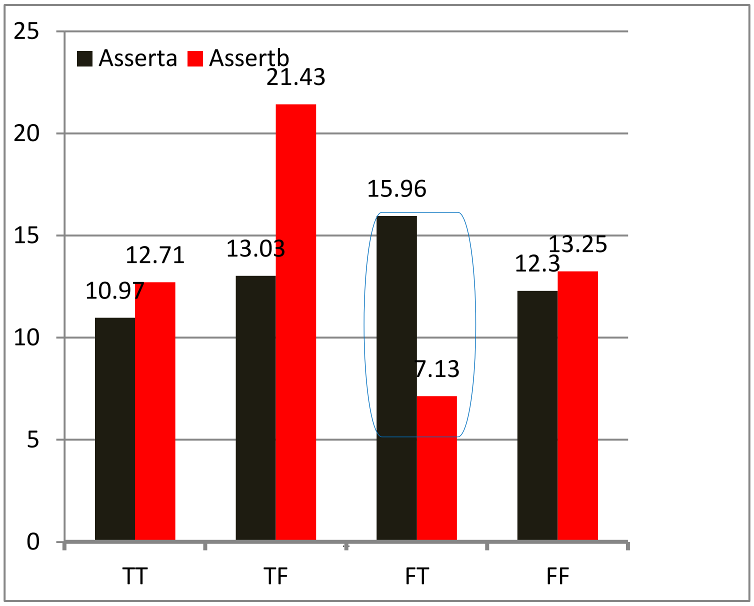


Fig 7.1 Mouth in P

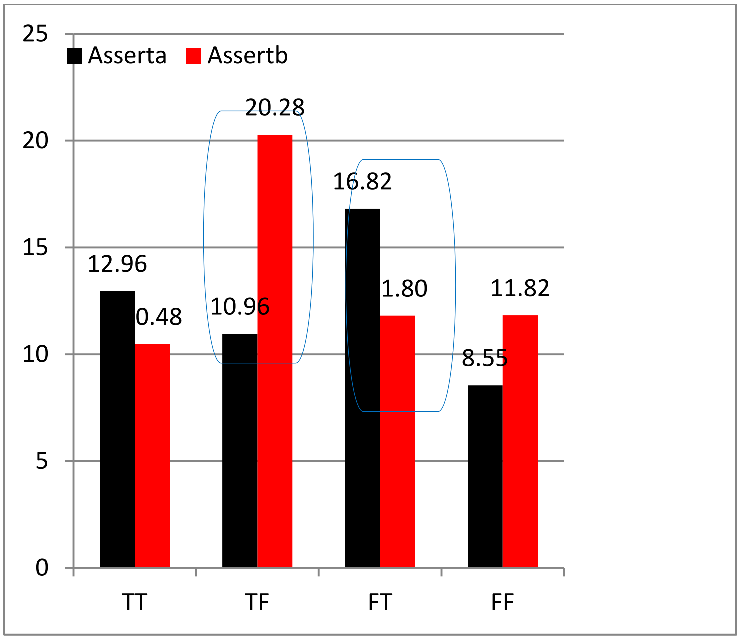


Fig 7.2 Brow in P

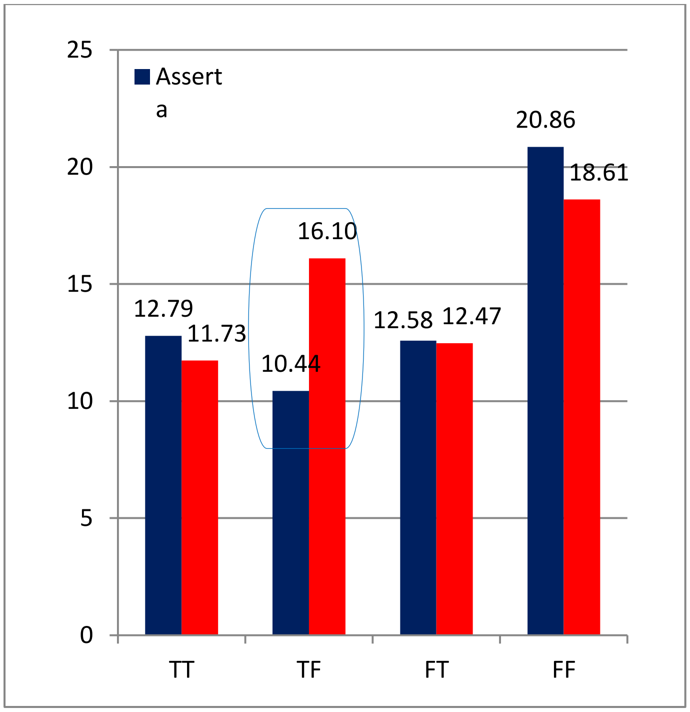


Fig 7.3 Mouth in NP

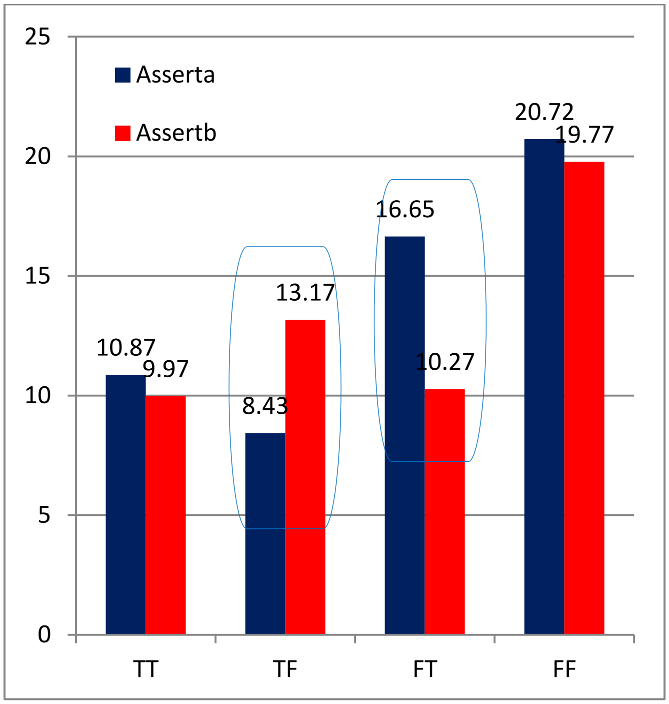


Fig 7.4 Brow in NP

Figure 7.5 and 7.6 show the percentage of time there was a Mouth and Brow movement respectively for TT,TF,FT and FF conditions of No Problem (N) type of concepts. These charts shows our 3rd finding where it is clear that for FF condition there was a much higher rate of Mouth and Brow movement for all types (Assert, Pause, Poll) of episode.

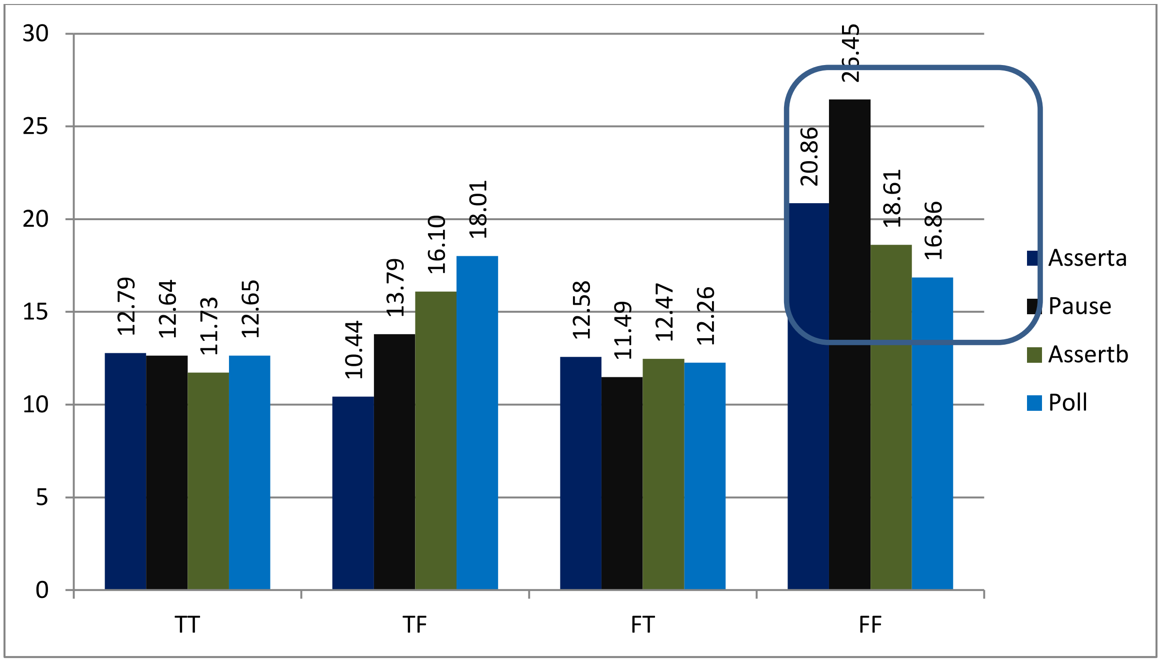


Fig 7.5 Mouth in all episodes

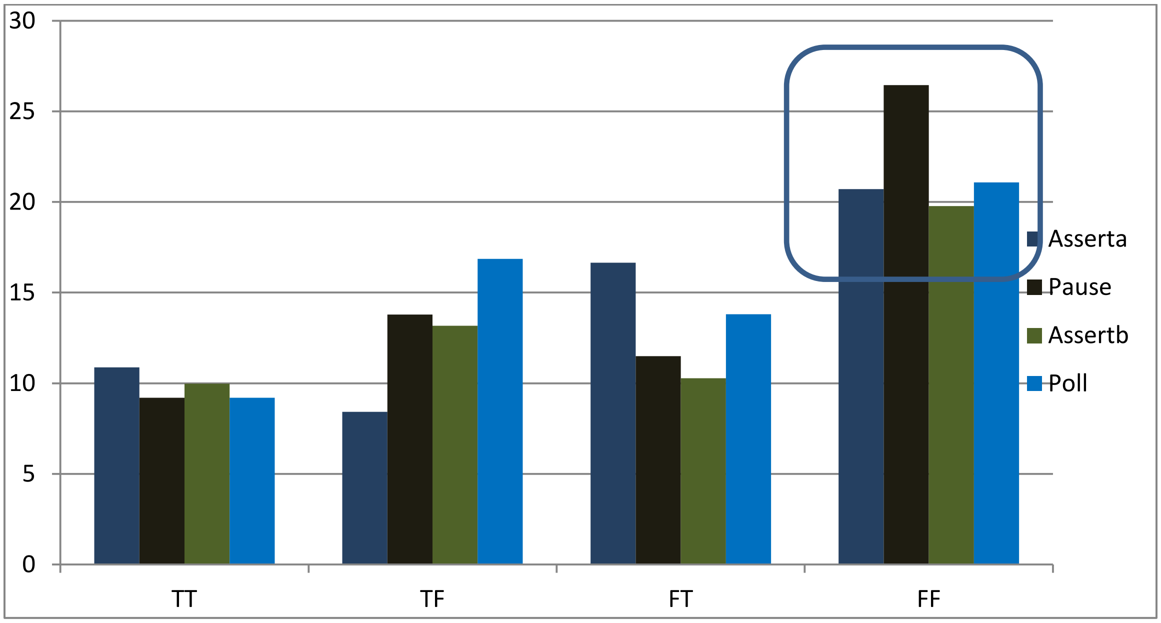


Fig 7.6 Brow in all episodes

All these analysis clearly states that the utility of our system in terms of the efficacy of diagnosing user’s learning centered emotion namely confusion through the different facial event sets that it captures is exhibiting promising successes.

Chapter 10

Conclusion

This thesis presented a theoretical framework and an accompanying open source implementation of a robust facial expression detection system that is capable of detecting subtle changes in mouth and eye brow regions in spontaneous and real world facial videos in real time. Proposed system was evaluated using two separate set of spontaneous video data for assessing its accuracy in determining facial events and accompanying affective states as well. Although our findings are promising there is room for improvement. Next steps should be to consider the issue with the AAM that AAMs are not able to deal with out-of-plain rotations of the face. In order to obtain a better capability in dealing with out-of-plain rotations the AAM could be extended to a 3-dimensional AAM. Other area of improvement is to make our system robust enough to deal with partial face occlusion as now we are ignoring any facial expressions those are occurring during facial occlusions. Our future goal is to integrate this facial feature based confusion detector into any application that relies on facial feature based emotion elicitation preferably to some auto mentoring systems.

References

[1] Paul Ekman and Wallace V Friesen, “Facial action coding system: A technique for the measurement of facial movement.” palo alto, *CA: Consulting Psychologists Press. Ellsworth, PC, & Smith, CA(1988). From appraisal to emotion: Differences among unpleasant feelings. Motivation and Emotion*, vol. 12, pp. 271-302, 1978.

[2] Yan Tong, Wenhui Liao, and Qiang Ji, “Facial action unit recognition by exploiting their dynamic and semantic relationships,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29 no. 10, pp. 1683–1699, 2007.

[3] Mohammed Yeasin, Baptiste Bullot, and Rajeev Sharma, “Recognition of facial expressions and measurement of levels of interest from video,” *Multimedia, IEEE Transactions on*, vol. 8, no.3, pp. 500–508, 2006.

[4] Rosalind W Picard, “Affective computing: challenges,”*International Journal of Human-Computer Studies*, vol. 59, no. 1, pp. 55–64, 2003.

[5] Gwen Littlewort, Jacob Whitehill, Tingfan Wu, Ian Fasel, Mark Frank Javier Movellan, and Marian Bartlett, “The computer expression recognition toolbox (cert),” in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on.* IEEE, 2011, pp. 298–305.

[6] Ruicong Zhi, Markus Flierl, Qiuqi Ruan, and W Bastiaan Kleijn, “Graph-preserving sparse nonnegative matrix factorization with application to facial expression recognition,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 41, no. 1, pp. 38–52, 2011.

[7] Stefanos Zafeiriou and Maria Petrou, “Nonlinear non-negative component analysis algorithms,” *Image Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 1050–1066, 2010.

[8] Tomas Simon, Minh Hoai Nguyen, Fernando De La Torre, and Jeffrey F Cohn, “Action unit detection with segment-based svms,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2737–2744.

[9] Bihan Jiang, Michel François Valstar, and Maja Pantic, “Action unit detection using sparse appearance descriptors in space-time video volumes,” in *Automatic Face & Gesture Recognition and Workshop (FG 2011), 2011 IEEE International Conference on*. IEEE, 2011,

pp. 314–321.

[10] Sungsoo Park and Daijin Kim, “Subtle facial expression recognition using motion magnification,” *Pattern Recognition Letters*, vol. 30, no. 7, pp. 708–716, 2009.

[11] Akshay Asthana, Jason Saragih, Michael Wagner, and Roland Goecke, “Evaluating aam fitting methods for facial expression recognition,” in *Affective Computing and Intelligent Interaction and Workshops*, *2009. ACII 2009. 3rd International Conference on*. IEEE, 2009, pp. 1–8.

[12] Jaewon Sung and Daijin Kim, “Real-time facial expression recognition using staam and layered gda classifier,” *Image and Vision Computing,* vol. 27, no. 9, pp. 1313–1325, 2009.

[13] Gareth J Edwards, Christopher J Taylor, and Timothy F Cootes “Interpreting face images using active appearance models,” in *Automatic Face and Gesture Recognition, 1998. Proceedings.Third IEEE International Conference on*. IEEE, 1998, pp. 300–305.

[14] Scotty D Craig, Sidney D’Mello, Amy Witherspoon, and Art Graesser, “Emote aloud during learning with auto tutor: Applying the facial action coding system to cognitive–affective states during learning,” *Cognition and Emotion*, vol. 22, no. 5, pp. 777–788, 2008.

[15] BT McDaniel, SK DMello, BG King, Patrick Chipman, Kristy Tapp, and AC Graesser, “Facial features for affective state detection in learning environments,” in *Proceedings of the 29th Annual Cognitive Science Society*, 2007, pp. 467–472.

[16] Timothy F Cootes, Gareth J Edwards, and Christopher J Taylor, “Active appearance models,” in *Computer VisionECCV98*, pp. 484 498. Springer, 1998.

[17] Takeo Kanade,“Picture processing system by computer complex and recognition of human faces,” 1974.

[18] Maja Pantic and Leon J. M. Rothkrantz, “Automatic analysis of facial expressions: The state of the art,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 12, pp. 1424–1445, 2000.

[19] Zhihong Zeng, Maja Pantic, Glenn I Roisman, and Thomas S Huang, “A survey of affect recognition methods: Audio, visual, and

spontaneous expressions,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 1, pp. 39–58, 2009.

[20] Ashok Samal and Prasana A Iyengar, “Automatic recognition and analysis of human faces and facial expressions: A survey,” *Pattern recognition*, vol. 25, no. 1, pp. 65–77, 1992.

[21] Beat Fasel and Juergen Luettin, “Automatic facial expression analysis: a survey,” *Pattern Recognition*, vol. 36, no. 1, pp. 259–275, 2003.

[22] Michel Francois Valstar, Marc Mehu, Bihan Jiang, Maja Pantic, and Klaus Scherer, “Meta-analysis of the first facial expression recognition challenge,” *Systems, Man, and Cybernetics, Part B: Cybernetics IEEE Transactions on*, vol. 42, no. 4, pp. 966–979, 2012.

[23] Jeffrey F Cohn, “Foundations of human computing: facial expression and emotion,” in *Proceedings of the 8th international conference on Multimodal interfaces*. ACM, 2006, pp. 233–238.

[24] Stefanos Zafeiriou and Maria Petrou, “Nonlinear non-negative component analysis algorithms,” *Image Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 1050–1066, 2010

[25] Fernando De la Torre, Joan Campoy, Zara Ambadar, and Jeff F Conn, “Temporal segmentation of facial behavior,” in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[26] Jacob Whitehill and Christian W Omlin, “Haar features for facs au recognition,” in *Automatic Face and Gesture Recognition 2006. FGR 2006. 7th International Conference on*. IEEE, 2006, pp. 5–pp.

[27] Ira Cohen, Nicu Sebe, Ashutosh Garg, Lawrence S Chen, and Thomas S Huang, “Facial expression recognition from video sequences: temporal and static modeling,” *Computer Vision and Image Understanding,* vol. 91, no.1, pp. 160–187, 2003.

[28] Simon Lucey, Ahmed Bilal Ashraf, and Jeffrey Cohn, “Investigating spontaneous facial action recognition through aam representations of the face,” *Face recognition*, pp. 275–286, 2007.

[29] Michel F Valstar and Maja Pantic, “Fully automatic recognition of the temporal phases of facial actions,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 42, no. 1, pp. 28–

43, 2012.

[30] Yunfeng Zhu, Fernando De la Torre, Jeffrey F Cohn, and Yu-Jin Zhang, “Dynamic cascades with bidirectional bootstrapping for action unit detection in spontaneous facial behavior,” *Affective Computing, IEEE Transactions on*, vol. 2, no. 2, pp. 79–91, 2011.

[31] Simon Baker and Iain Matthews, “Equivalence and efficiency of image alignment algorithms,” in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on.* IEEE, 2001, vol.1, pp. I–1090.

[32] Paul Viola and Michael Jones, “Rapid object detection using a boosted cascade of simple features,” in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. IEEE, 2001, vol. 1, pp. I–511.

[33] Rana El Kaliouby and Peter Robinson, “Real-time inference of complex mental states from facial expressions and head gestures,” in *Real-time vision for human-computer interaction*, pp. 181–200. Springer, 2005.

[34] Takeo Kanade, Jeffrey F Cohn, and Yingli Tian, “Comprehensive database for facial expression analysis,” in *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on.* IEEE, 2000, pp. 46–53.

[35] Arthur C Graesser, Shulan Lu, Brent A Olde, Elisa Cooper-Pye, and Shannon Whitten, “Question asking and eye tracking during cognitive disequilibrium: Comprehending illustrated texts on devices when the devices break down,” *Memory & Cognition*, vol. 33, no. 7, pp. 1235–1247, 2005.

[36] Arthur C Graesser and Brent A Olde, “How does one know whether a person understands a device? the quality of the questions the person asks when the device breaks down.,” *Journal of Educational Psychology*, vol. 95, no. 3, pp. 524, 2003.

[37] Yao Wei, “Research on facial expression recognition and synthesis,” *Master Thesis, Department of Computer Science and Technology, Nanjing*, 2009.