HUMAN-MACHINE INTERFACE USING FACIAL GESTURE RECOGNITION

Zikra Ibrahim Toure

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

December 2017

Toure, Zikra Ibrahim. *Human-Machine Interface Using Facial Gesture Recognition*. Master of Science (Electrical Engineering), December 2017, 60 pp., 4 tables, 36 figures, 66 numbered references.

This master thesis proposes a human-computer interface for individual with limited hand movements that incorporate the use of facial gesture as a means of communication. The system recognizes faces and extracts facial gestures to map them into Morse code that would be translated in English in real time. The system is implemented on a MACBOOK computer using Python software, OpenCV library, and Dlib library. The system is tested by 6 students. Five of the testers were not familiar with Morse code. They performed the experiments in an average of 90 seconds. One of the tester was familiar with Morse code and performed the experiment in 53 seconds. It is concluded that errors occurred due to variations in features of the testers, lighting conditions, and unfamiliarity with the system. Implementing an auto correction and auto prediction system will decrease typing time considerably and make the system more robust.

ACKNOWLEDGEMENT

"In the name of God, the Most Gracious, the Most Merciful."

First and foremost, I would like to thank God. Second, I am sincerely grateful to my advisor Dr. Kamesh Namuduri for all the guidance and support he provided during this Master's thesis research. His passion for his students is shown through his daily efforts in educating successful students.

I would also like to thank my colleague Lokesh Kumar Viswavarapu for his support and guidance in the completion of this research.

I would like to acknowledge Peggy Foster for always keeping a smile on no matter the day and vibrating such an uplifting attitude and energy.

Most importantly, I would like to thank my father, who I love from the bottom of my heart, for sending me across the sea to pursue a better education.

Lastly, I am beyond grateful for the undying love and support I received from my family all throughout this process.

TABLE OF CONTENTS

Page

ACKNOWLEDGEMENT iii

LIST OF TABLES vi

LIST OF FIGURES vii

CHAPTER 1. INTRODUCTION 1
- 1.1 Background 1
- 1.2 Motivation 2
- 1.3 Objectives and Contributions 2
- 1.4 Organization 3

CHAPTER 2. RELATED WORK 4
- 2.1 Facial Recognition Techniques 4
- 2.2 Facial Feature Extraction 5
- 2.3 Active Appearance Models 8
- 2.4 Facial Expression Recognition Systems 9

CHAPTER 3. GESTURE RECOGNITION FOR COMMUNICATION 13
- 3.1 Hand Gesture Recognition for Communication 13
- 3.2 Challenges of Hand Gesture Communication Systems 13
  - 3.2.1 Illumination Condition 14
  - 3.2.2 Background Noise 14
  - 3.2.3 Variety of Implementation 15
  - 3.2.4 System Limitations 15
  - 3.2.5 Availability of Training Data 15
- 3.3 Face Gesture Recognition 16
- 3.4 Challenges to Face Gesture Recognition 17
  - 3.4.1 Human Posture 17
  - 3.4.2 Occlusion 17
  - 3.4.3 Illumination 17

3.4.4 Databases 18
3.4.5 Face Aging 20
3.5 Steps to Facial Expression Recognition 20
3.5.1 Face Detection 20
3.5.2 Facial Feature Extraction 21
3.5.3 Classification 22
3.5.4 A Facial Expression Recognition Example: EIGENFACE Algorithm 22
3.6 Communicating using Facial Gestures 23
3.6.1 Head Motion Controlled Devices 24
3.6.2 Face Motion Controlled Devices 25
3.7 Software 26
3.7.1 Python 26
3.7.2 OpenCV Library 27
3.7.3 Dlib Library 29
3.8 Morse Code 30
3.8.1 Morse Code Representation 30
3.8.2 Morse Code Learning Method 30

CHAPTER 4. FACIAL GESTURE RECOGNITION FOR MORSE CODE DESIGN AND IMPLEMENTATION 32
4.1 Face Detector Training 32
4.2 Feature Extraction 34
4.3 Classification 39
4.4 Experiments and Results 41

CHAPTER 5. CONCLUSIONS 54

REFERENCES 55

LIST OF TABLES

Page

TABLE 4-1: Experimental training and Dlib training comparison. 42

TABLE 4-2: Success and failure for each detector for first trial 48

TABLE 4-3: Success and failure for each detector fro 15th trial 49

TABLE 4-4: Success and failure rate for each tester 52

LIST OF FIGURES

Page

Figure 2.1: Gabor Labelled elastic graph representation of an image 6

Figure 2.2: Depiction of location of model points from [2] 7

Figure 2.3: Depiction of a training shape from [2] 8

Figure 2.4: Proposed algorithm in [15] mixing an AAM with a neural evolution. 9

Figure 2.5: Action Parameters on contoured face from [18] 10

Figure 2.6: Block diagram of the proposed facial expression recognition system in [27]........... 12

Figure 2.7: Block diagram of the multi-modal facial emotion detection in [28]. 12

Figure 4.1: Human-computer interface system block diagram 32

Figure 4.2: Face detection algorithm 33

Figure 4.3: Examples of bounding boxes manually drawn on images from LFW database 33

Figure 4.4: Facial landmark and bounding box on images from iBUG 300-W dataset................ 34

Figure 4.5: Exact location of points of facial landmarks depicted 35

Figure 4.6: Points location on eye... 35

Figure 4.7: Block diagram of eye blink detector algorithm 36

Figure 4.8: Points location on inner line of mouth 37

Figure 4.9: Block diagram of the open mouth algorithm 37

Figure 4.10: Point locations of nose bridge. 38

Figure 4.11: Block diagram of the head shake/node algorithm 39

Figure 4.12: Depiction of eye blink being translated into a dash 40

Figure 4.13: Depiction of an open mouth translated into a dot.. 40

Figure 4.14: Morse code dictionary 41

Figure 4.15: HOG filter from experiment file and HOG filter from DLIB example file................. 41

Figure 4.16: Facial landmark displayed in real time video. 43

Figure 4.17: Graph of the blink detector behavior 43

Figure 4.18: Graph of the open mouth detector behavior... 44

Figure 4.19: Depiction of the mouth state and its MAR... 45

Figure 4.20: Graph of head movement behavior. 45

Figure 4.21: Depiction of head movement detector used in [66] 46

Figure 4.22: Depiction of the system being used to spell out "Hello"... 47

Figure 4.23: Graph depicting each detector's behaviour on first trial 47

Figure 4.24: Graph depicting each detector's behaviour on 15th trial 48

Figure 4.25: Student testing the system 49

Figure 4.26: Morse code for "I WANT WATER" 50

Figure 4.27: Mouth aspect ratio graph 51

Figure 4.28: Eye aspect ratio graph 51

Figure 4.29: Head movement graph 52

CHAPTER 1

INTRODUCTION

1.1 Background

A disability is a state of being where one is not in the capacity of using one or more of his or her bodily functions, which in return substantially impacts his or her daily activities. In most cases, people do not choose to have a disability. It is a state that comes upon us due to genetics, birth complications, or accidents. Any individual who has a disability can attest to the challenges that disabled people face daily due to the lack of resources within society to help in the adjustment.

On top of technological advancement in the domain, many resources have been created and many laws have been passed in the hope of procuring more advantages to people with disabilities. In the United States of America alone, multiple important federal laws exist to provide them rights and protect them from discrimination. In the case of physical impairments, many technologies were developed to aid in the adjustment. Wheelchairs have been around for centuries, and they are still the first solution for people with physical disabilities. The invention of prosthetics has brought new hopes to those that have missing limbs. Many computer software and mobile applications and technologies are on the market today to help people with disabilities, such as vision impairment, speech impairment, or motion disorders. With these technological advancements, not many technologies are out there for people with severe disability or extremely limited muscle movement, such as completely paralyzed individuals.

In the case of limited muscle movements or speech impairment, a lot of research was performed in the area of computer vision to provide sustainable solutions. More particularly, advanced research was performed in the domain of gesture recognition and machine learning to aid individuals in communicating with a human-computer based interface.

1.2 Motivation

Previous works were done in the domain of hand gesture recognition. Notably many computer vision techniques were developed in the aid of hand gesture recognition to aid people with speech impairment to communicate with a human-machine based interface. However, these works are limited as most of them don’t consider the case of lacking voice and muscle movement. Even though some work spent significant efforts in developing methods for hand gestures and facial gestures recognitions, much efforts were not put towards achieving a human-computer based system to aid communications for people lacking voice and muscle movements. Research has focused on hand gesture recognitions for translating American Sign Language into voice and speech and facial gesture recognition for emotions identification and computer control, such as mouse click and web browsing.

1.3 Objectives and Contributions

The goal of this research is to provide a human-computer interface for individuals with limited hand movements that incorporate the use of facial gesture as a means of communication. Notably, this thesis proposes a facial gesture based Morse code system. This thesis contributes by joining multiple facial gesture detector and mapping them into Morse code for a system to translate in real time in English. It makes use of a trained face detector, a facial landmark for feature extraction, an eye-blink detector, an open mouth detector, and a head nod/shake detector to build the system.

1.4 Organization

The paper is organized in the following manner. Chapter 2 goes over some literature research related to the thesis. Chapter 3 presents some background information on gesture recognition for communications. Chapter 4 presents the implementation of the design and the experiments. Chapter 5 concludes the thesis and suggests future work.

CHAPTER 2

RELATED WORK

Facial gesture recognition has for long been a topic of interest in the domain of computer vision. Indeed, several algorithms were invented to account for all the changes inherent to vision, such as variation in luminosity, facial expression, eye wear, hair pieces and hair styles, gender, age, color, skin texture, and shape. Those algorithms were proposed to efficiently locate or identify faces and facial features regardless of the variables mentioned above. Facial gesture recognition has many applications in the domain of law enforcement and human-computer interface.

2.1 Facial Recognition Techniques

Many algorithms were developed for facial recognition. The simplest of them is the correlation algorithm, which is a nearest neighbor classifier in an image space [11]. It uses normalization to give all images zero mean and unit variance. It is equivalent to choosing from the training set the image that best correlates with the sample image. Another algorithm is the Linear subspace algorithm. Three images of the face under 3 linearly independent lighting sources are used to reconstruct the image under any arbitrary lighting direction [12], which makes the algorithm work under a wide variation of lighting conditions.

The EIGENFACES algorithm [9] uses Principal Component Analysis (PCA) to maximize the scatter of all projected samples. The mean of all sample images and the covariant matrix is determined using the formulas below from [9].

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \; ; \; S_T = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$$

where xi is part of a set of sample images.

The covariance matrix tells us how far each sample is from each other. It gives us the scatter of the data. From the covariant matrix, the eigenvectors and eigenvalues are found. Those eigenvectors are the “Eigenfaces”. The largest eigenvectors with the highest associated eigenvalues are kept, and they are used to represent both existing and new faces.

The “Fisherfaces” algorithm [10] is an enhance version of the “Eigenfaces” method to compensate for variation in luminosity. It is better as it uses the Fisher’s Linear Discriminant Analysis to maximize scatter ratio.

2.2 Facial Feature Extraction

There are multiple techniques for facial feature extraction. A classical one is principal component analysis (PCA) [27]. Its goal is to reduce dimension for efficient face indexing.

A combination of techniques based on geometrical shape parametrization, template matching, and dynamic deformation of active contours were presented in [1] to extract eye-brows, eyes, nostrils, mouth, cheeks, and chin features. The authors used geometric representations to extract those features. The cheek areas were represented by a straight line. The chin area was represented by the equation of a parabola. Certain features, such as the eyebrows, cannot be extracted using geometrical shape parametrization due to their irregularity.

A template matching technique using binary blocks as a mask used to find small parts of an image matching a template image is used to extract the eyebrow features of the face. Another matter they tackled in the paper is determining the orientation of the face. They made use of an isosceles triangle made from the eyes and mouth feature of the face. The longest eye to mouth side determines the orientation of the face.

In [2], the author presents an algorithm to automatically extract facial features from a set of digital images using a combination of Gabor Wavelet labeled elastic graph matching [3], [4] and "Eigenfaces" or "Fisherfaces" algorithms [5], [6], [7]. Gabor filtering is a linear bandpass filtering used in image processing for texture analyzing. Small changes in face and skin texture can be approximately represented by Gabor Wavelet functions [13]. The images are first transformed using Gabor filter. The response is sampled at some points and combined into a labeled graph vector, as shown in Figure 2.1 from [13].

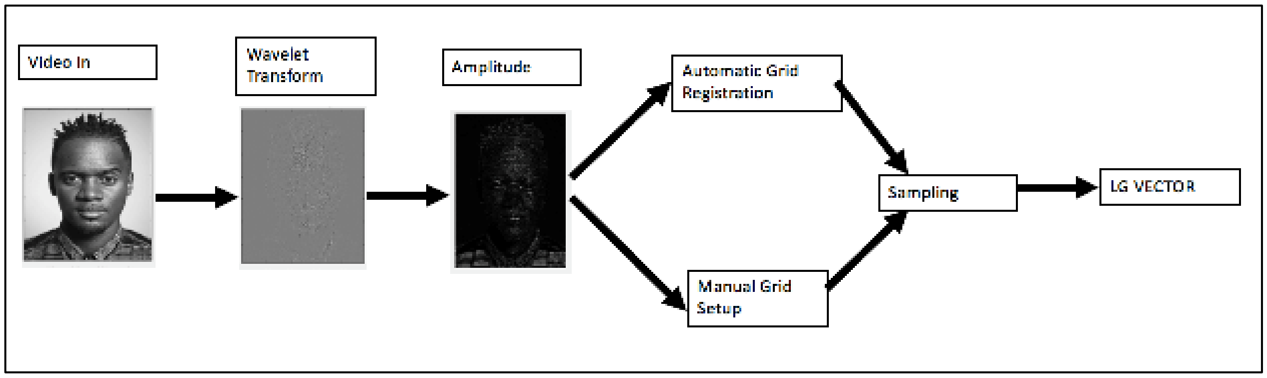


Figure 2.1: Gabor Labelled elastic graph representation of an image

From the LD vector, the "Eigenfaces" algorithm is applied. The mean of all sample images and the covariant matrix is found using the formulas below from [2].

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \, ; \; S_T = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$$

where xi is part of a set of sample images.

The covariance matrix tells us how far each sample is from each other. It gives us the scatter of the data. From the covariant matrix, the eigenvectors and eigenvalues are found. Those are used to construct a linear transform that allows us to go from the space of x to the space of y as described in the equation below from [2].

$$y_i = W^T(x_i - \mu)$$

A discriminant vector is calculated using training images. That vector is used to classify the LG input through projection. The input vector is then classified within the nearest cluster.

Facial feature extraction is a difficult task because of the highly variable components, such as luminosity, pose, and expression. The authors in [8] proposed a compact parametrized facial appearance that take those variables into factor. Their approach has 2 main phases:

1) Modeling: Mathematical expressions are used to generate models of different facial appearances. Each face is represented by 152 points (depicted in Figure 2.2) manually located on each image within the training set. A mean shape is derived from the analysis of a training set. Figure 2.3 depicts a training shape

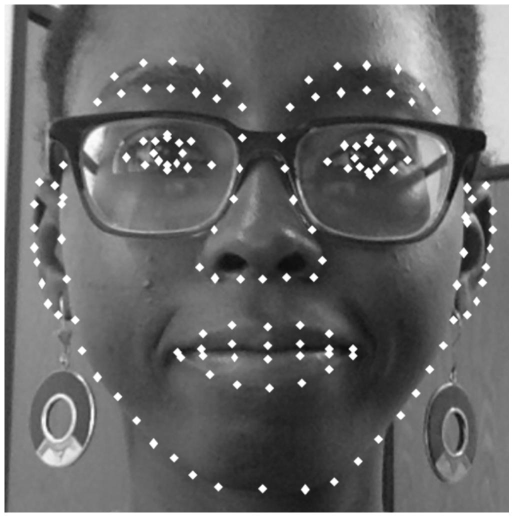


Figure 2.2: Depiction of location of model points from [2]

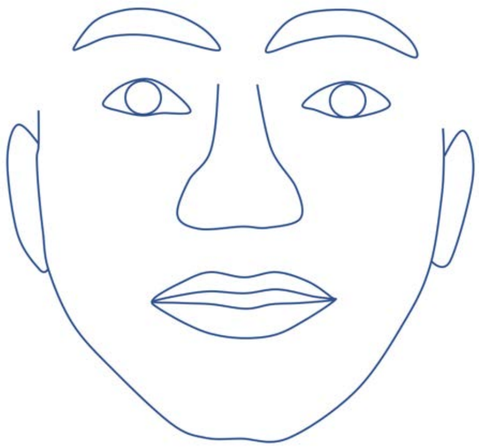


Figure 2.3: Depiction of a training shape from [2]

2) Interpretation: The models are used to interpret the input face images. The shape and grey level models are used to automatically identify facial features. The input image is interpreted in terms of these 2 appearance parameters, and those are used to identify things like gender, expression, and to reconstruct face images.

2.3 Active Appearance Models

Active appearance models or AAM is a method used to match statistical models of objects' appearance to a new image. It was first introduced by G. Edwards, C. Taylor, and T. Cootes in [14]. In [15], the authors used AAMs to extract facial features and used neural network and differential evolution algorithms to classify the emotions. After an AAM was used to obtain a statistical representation of the face, features like mouth, eyes, and eyebrows were extracted. A neural network is trained using a differential evolution to obtain the optimum neural network called a neural evolution. This neural evolution can then distinguish between happy, sad, and neutral. Figure 2.4 described the proposed algorithm in [15].

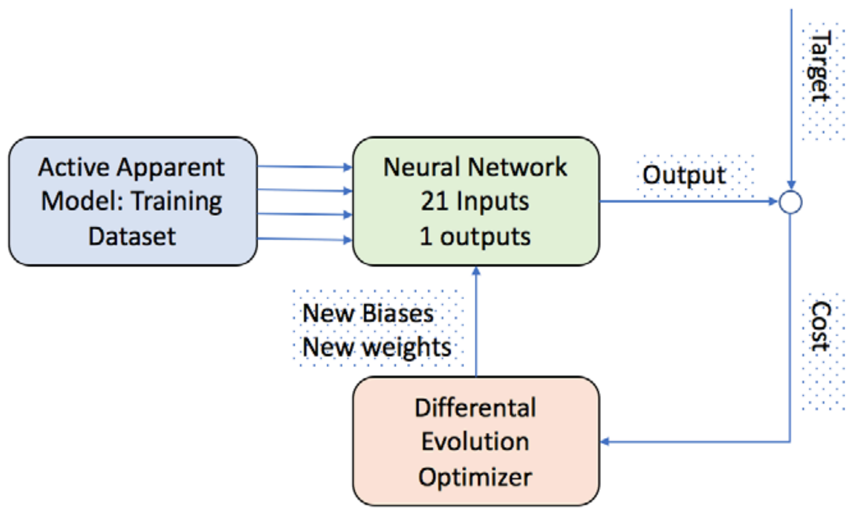

Active Apparent Model: Training Dataset
Neural Network 21 Inputs 1 outputs
Output
Target
Cost
New Biases New weights
Differental Evolution Optimizer

Figure 2.4: Proposed algorithm in [15] mixing an AAM with a neural evolution.

A medical application of AAMs is proposed in [16]. The authors proposed an automated approach to segmentation of left and right cardiac ventricles from magnetic resonance images. They developed a multistage hybrid combination of active shape and appearance models. Although not directly related to facial gesture recognition, this paper is worth mentioning for its innovative contribution to the medical field.

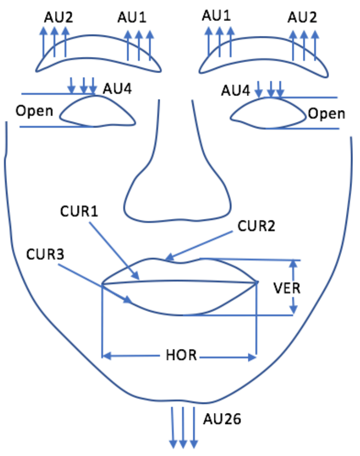
The authors in [17] addressed the issue of computational time in AAM and developed a fast and stable AAM search for 3D face tracking. Their work is different to that in [14] because they deal with 3D- geometry of shape and motion for the first time, and they adopt independent 3D shape and appearance models.

2.4 Facial Expression Recognition Systems

For the last few decades, multiple researchers have spent time developing algorithms and techniques for facial expression detection. Facial expressions are the results of certain facial muscles contraction. As mentioned in [26], facial expression detection and recognition is done in

4 stages. First stage is face detection. Second stage is feature extraction. Third stage is selection of required features, and fourth stage is feature classification.

In [18], the author used model-based feature extraction to find facial features and action parameters for classification to detect facial expressions. Just like in [8], feature points on the training images were manually located. Their model has 2 phases, a training phase given 90 different expressions, where 10 action parameters called APs are used to label the position variations of the designated points on the face, and a recognition phase, where those Aps are extracted, analyzed, and correlated to identify the given facial expression. They accomplished an average correct ratio of about 80% for the 6 facial expressions combined, happy, smile, surprise, sad, anger, and fear. These APs are calculated using facial parameters (FPs) shown in Figure 2.5 below. AP1, AP2, AP3, and AP4 are equivalent to AU-1, AU-2, AU-4, and AU-26. The rest of APs are generated using the difference between FPs from current and last frame.



AP6 = VER(last) – VER(first)
AP7 = HOR(last) – HOR(first)
AP8 = CUR1(last) – CUR1(first)
AP9 = CUR2(last) – CUR2(first)
AP10 = CUR3(last) – CUR3(first)

Figure 2.5: Action Parameters on contoured face from [18]

In [19] the authors provided another spectrum to analyse and recognize facial emotions in real time. In their paper, Facial Emotion Recognition in Real Time, they developed a neural network to classify human emotions. They made use of the Japanese Female Facial Expression database (JAFFE) [3] and the Cohn-Kanade database, both of which provide well-defined facial expressions in a controlled laboratory environment, to develop a model. They also developed and made use of their own database consisting of expressions from 5 other individuals. With the help of the open source software, OpenCV, they developed an application that successfully detects six facial expressions (anger, fear, happy, sad, neutral, and surprise) in real time with an accuracy level of 90% in laboratory condition.

Deep convolutional network features were applied to emotion recognition in [20]. A convolutional network model trained on 1.2 million images as described in [21] was used to extract features from images. The training dataset chosen was the Extended Cohn-Kanade Dataset (CK+) [22]. To do the classification, a Support Vector Machine model was used [23]. The system is implemented on Python with OpenCV [24] library where the Viola Jones detector [25] is implemented.

The authors from [27] proposed a facial expression recognition system as described in Figure 2.6. The eyebrows and outer-lip facial animation parameters (FAPs) were extracted from a source video and fed into a Hidden Markov Model (HMM) for automatic facial recognition. Their experiment first used single stream HMM under different scenarios using the extracted features separately then jointly. They then proposed a multi-stream HMM approach, which achieved a reduction of the recognition error of 44% compared to the single-stream HMM system [27].

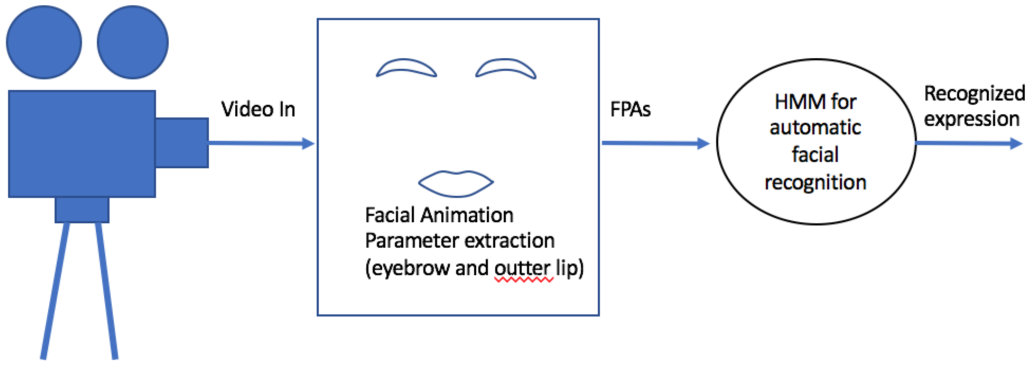

Video In
Facial Animation Parameter extraction (eyebrow and outter lip)
FPAs
HMM for automatic facial recognition
Recognized expression

Figure 2.6: Block diagram of the proposed facial expression recognition system in [27]

The authors in [28] used a multimodal approach for facial emotion recognition. A block diagram of their algorithm is shown in figure 2.7. They basically combined the sound of an emotion and the facial expression of the emotion to provide more accuracy to their system.

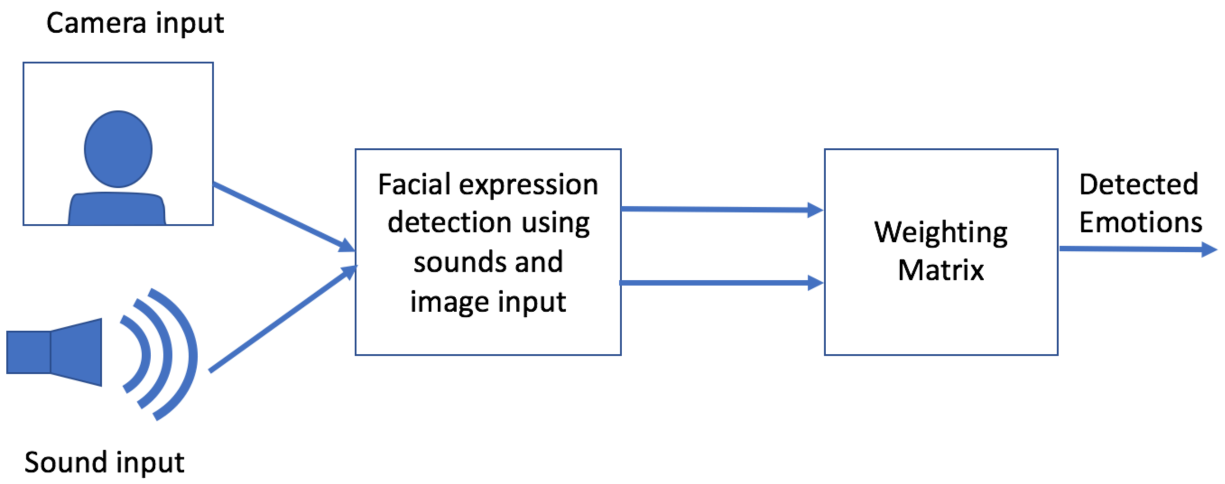


Figure 2.7: Block diagram of the multi-modal facial emotion detection in [28].

CHAPTER 3

GESTURE RECOGNITION FOR COMMUNICATION

3.1 Hand Gesture Recognition for Communication

Many times, people have been trying to come up with revolutionary ideas to help people with the lack of muscle movement to communicate. There has been an increase in the demand for human computer interface to aid with communication. Innovations in this domain are in demand due to the fact that there has not been a single solution capable of solving all issues in the domain of communication in order to serve all types of abled bodies.

Hand gesture recognition technologies are among the first type of human-computer interfaces. Scientists developed algorithms so that people can use simple hand movements to interact with or control a device without the need of touching anything. Their practicality made life somehow easier for humans by eliminating the need to hold or press a device to control another. With the simple use of hand movements, people switch channel, reduce or augment volume of the television without the need of a remote. They can scroll or shift on a tablet with in-air movements. They can play video games online or offline through hand gesture. Also, they can browse online contents.

3.2 Challenges of Hand Gesture Communication Systems

The main goal of any gesture recognition system is to clearly and perfectly translate a human gesture into its intended purpose in rapid and efficient manner. When a person waves at an interface, the interface must be able to recognize the wave and appropriately respond to it without a glitch or vast amount of wait time. The interface must be robust and reliable under all circumstances to deliver the best experience.

Because of these expectations, there are many challenges that hand gesture communication systems face.

3.2.1 Illumination Condition

Humans were granted the gift of eyes. With the help of the eye organ, humans can recognize faces, movements, objects, shapes, and colors under any variation of lighting condition. However, computers and cameras do not have this luxury as they are merely instruments for humans to control. Any change in brightness, saturation, exposure, or contrast impacts the ability of an interface to effectively recognize hand gestures. Most hand gesture recognition algorithms use skin color extraction to firstly identify the location of the hand in a video. When there are fluctuations in this task, it becomes more complicated to accurately detect the hand region.

3.2.2 Background Noise

Background noises in any type of application has made it a challenge to meet the desired goal of said application. During a telephone communication, they almost overshadow the intended voice one is trying to listen to. In photography, they reduce the quality of an image. In television, they alter the reception of video images. Recognizing and tracking gestures in a background noise has always been a difficult task. The accuracy and rate of recognition drops significantly due to them. Usually, background subtraction and edge detection help in eliminating the issue. However, a technique has yet to be perfected to achieve a near perfect accuracy.

3.2.3 Variety of Implementation

Multiple algorithms have been developed for gesture recognitions. With the careful research and planning, some researchers may omit to consider the versatility of their algorithm. They may work on one interface but may not work on another. For example, a gesture recognition system engineered to work for people with pale skin may not work for people with darker skin, and vice versa, or an algorithm may work with certain cameras and not work with others.

3.2.4 System Limitations

Another challenge to gesture recognition is the limitation put on the posture and the gesture itself, such as the right hand needs to be doing the gesture, the palm of the hand must be facing the camera, only the upper body needs to be in the frame, the arm must be straight initially, and many more. Those limitations preclude the system to be robust and versatile.

3.2.5 Availability of Training Data

In identifying sign language for example, an important constraint to accuracy is the amount of training data available. The more training set we get per sign, the more accurate the system is. The existing open source datasets have unfortunately limited amount of training images. For example, the ASLLVD dataset [28] has at least 3000 signs, but they provide only 2 examples per sign. The Purdue ASL dataset [29] is also available with 66 different signs and sentences performed by 15 different individuals. The size of the dataset is considerably small.

3.3 Face Gesture Recognition

When it comes to gesture recognition for communication, people tend to study, do research on, and focus on general body movements such as hand gestures for common actions and sign language. Species that usually use facial gesture to communicate are animals. For example, dogs communicate anger through growling and snarling and showing their teeth. Young chimpanzees communicate submission through displaying their teeth almost like a smile. Humans make a use of facial expression to emphasize the emotion they want to portray. Most studies that have been done in the domain of facial gesture recognition for communication involve recognizing facial expressions or emotions.

Facial muscle actions are direct ways for humans to communicate their expression. They represent visible signals that regulate our social interaction. Identifying facial expressions can be important in the domain of law enforcement, psychological research, visual-speech synthesis, perceptual interface, lip reading, and more. Facial expressions in machine learning is usually done in 3 steps: detecting the face region in a set of images, extracting the facial features, and performing classification. There has been a growing interest in the field of facial gesture recognition among scientists, spanning from computer scientists, engineers, and psychologists. However, an automated interface that accurately recognizes facial expressions as good as humans has yet to be achieved. There are some challenges.

3.4 Challenges to Face Gesture Recognition

3.4.1 Human Posture

Variation in posture can have considerable impact in the appearance of the face. Any yaw, pitch, roll, or head movements introduce a new element to the image that a training set may not be used to. Therefore, there will be less accuracy in the classification step of the facial expression recognition algorithm. However, in [30], they introduced pose correction algorithms to bring a solution to the challenge.

3.4.2 Occlusion

Problems can arise when there are extra elements or the lack of it in the image that is being processed. For example, when a training data set consists of images of people with empty background, it may be difficult to process an input image that includes structuring elements or components. Or, if all images in the training set consist of people and specific structural elements, processing of the input image can also be less accurate.

Components such as moustaches, beard, hat, sunglasses, or anything in the background or foreground of the image introduce occlusion and therefore reduces the robustness of a system that has not considered such issue. R. Singh, M. Vatsa, and A. Noore [31] did an experiment to evaluate what would be the most accurate way to identify expressions regardless of occlusion. They found that texture-based algorithms are the most efficient.

3.4.3 Illumination

Just as in hand gesture recognition, variation in illumination drastically changes the performance of a system in facial expression detection. Variation in lighting introduces shadows or bright areas that might have not been trained in a dataset. A way to deal with the issue would be to just introduce faces under different types and angles of lighting into the training dataset. Another method introduced in [32] would be histogram equalization to normalize the illumination intensity.

3.4.4 Databases

To do the classification step of facial expression recognitions, all algorithms use a dataset of faces that are either homebrewed or acquired from other research. A broad database with a variety of illuminations, facial expressions, occlusion, resolution, and color level is important to produce the best interface. Compared to hand gesture recognition, there are a lot of databases available for the application of face gesture recognition.

- Cohn-Kanade AU Coded Facial Expression database [22]: The database comprises of 100 males and females with age ranging from 18 to 30 in a variety of race and ethnicity. Everyone performed 23 different facial expressions spanning from neutral to emotional expressions.
- The Yale Face database [33]: 15 individuals were used in the development of this database. Each of them performed 11 different facial expressions, such as happy, neutral, sad sleepy, wink, under the constrain of 2 different lighting configurations with and without glasses.
- Japanese Female Facial Expression Database [3]: This database comprises of 10 female Japanese posing 7 different expressions, neutral, sad, happy, surprised, fear, angry, and disgust.

- Face Recognition Technology (FERET) [34]: One of the biggest databases, it used almost 1200 individuals each posing 5 to 11 different expressions acquired in different resolutions under a variety of illumination types.
- Caltech 10,000 web faces [35]: This database is formed with images acquired from Google images. A total of 10,524 faces under different lighting, resolution, and other conditions were acquired.
- Large Age Gap database [36]: This database introduces a variety in age. It contains 3838 images of 1010 celebrities as children and as adults.
- AT&T database of faces [37]: 40 different people pose 10 different expressions each. The images were taken in a variation of lighting, with different facial expressions and details such us open or closed eyes or with or without glasses. All the images had the exact same background without any occlusion.
- BioID Face Database [38]: This database has 1521 gray scale images under the same resolution of 23 different people's frontal view under real-life conditions, such as different lighting or background.
- Caltech faces [39]: This database is of 27 different people giving different facial expression with a variety of lighting and background. It has a total of 450 face images in JPEG format.
- Face Detection Data Set and Benchmark [40]: This database contains shots captured in an uncontrolled environment, i.e. each image includes some sort of occlusion or pose variation. It consists of 2845 images of 5171 faces in both greyscale and color.

- Labeled Face in the Wild [41]: This is another database with images captured in real life. It consists of 5749 individuals of different gender, age, race, and ethnicity posing in 13233 images under different lighting, focus, and saturation in JPEG format mostly in color.

3.4.5 Face Aging

As humans age, their facial features change with time. The shapes and lines on the face are modified. The hairstyle may also be different. In [42], they present a facial expression recognition system considering aging patterns. Their system can recognize face images across the age spectrum. Their algorithm has 2 steps. First the input images from different age groups are synthesized to extract patterns. Then, an aging face verification system is used to interpret whether the input faces are of the same person.

3.5 Steps to Facial Expression Recognition

As discussed previously, there are 3 steps in facial expression recognition. The first step is face detection. The second step is facial feature extraction. The third step is image classification.

3.5.1 Face Detection

For any type of facial recognition system, the very first thing that needs to be done is detecting the face regions in the image, whether there are one or more faces per image. As described above, this may not be an easy task because shots of faces can be taken in various angles. Occlusion and noises are also added challenges. Some authors use feature base detection such as locating the irises and nostrils, the location of lips, or edge detection. Many algorithms exist to aid in detecting faces in computer vision.

- Viola Jones Face Detector: This face detection algorithm is the most popular one. It is often found in IPhone or Android smartphones. It can be regarded as a binary classifier because of the use of Haar features. Haar-like features are obtained by the subtraction of a sub-region of a feature from the remaining region. The Viola Jones algorithm is the first and most widely used algorithm for face detection. It is fast and has a high accuracy rate.

- Local Binary Pattern (LBP) [43]: This is an effective approach for face detection as it uses pixel encoding to describe the image texture features. It is widely used in texture analysis, face recognition, image segmentation, and image retrievals. It is a simpler approach compared to the Viola Jones algorithm, but it is not as accurate.

- Neural Networks [44]: In this type of method, small sections at a time of a frame are scanned to determine the presence or absence of a face. It's a simple algorithm but presents considerably high false positive.

3.5.2 Facial Feature Extraction

After the task of detecting a frontal face in a frame is accomplished, the next step is the extraction of facial expression features. Key elements on the face, such as the mouth, the eyes, the eyebrows, the chin, the cheeks, and the jawline, usually aids the human at recognizing the emotion that is being displayed. These features are extracted because their deformations introduce wrinkles and bulges, which are detectable using image processing. Most of the time, algorithms are trying to detect anger, disgust, fear, happiness, sadness and surprise. Feature extraction algorithms can be either motion-based or deformation-based.

- Optical Flow method: The optical flow method is a motion estimation method. It is a pixel-based approach that has been used to detect muscle actions on the face that trigger deformation of features, such as eyes, mouth, cheeks, and nose. These detected actions are then compared to different facial expressions to find the best match.
- Gabor filter [45]: Gabor filter is a texture analysis method for feature extraction. A two-dimensional Gabor filter is nothing but a Gaussian modulated by a sine wave. Its algorithm is used to select the needed features. The Gabor filter comprises 5 scales and 8 orientations for feature extractions. Computing Gabor filter is rather expensive due to convolution techniques.

3.5.3 Classification

The next step after feature extraction is to identify which emotion is being conveyed. This is done through classification. The classification step can be done in terms of action parameters, such as detecting movements or in terms of key features, such as open mouth and raised eyebrows. Classification can also be as simple as comparing the test image to the training set within a database. One of the most notable techniques for classification in real time is The Facial Action Coding System (FACS) [46]. It uses Action Units to measure a facial expression in real life by breaking it down into individual muscle movement.

3.5.4 A Facial Expression Recognition Example: EIGENFACE Algorithm

The EIGENFACE algorithm is a statistical method used to reduce faces into a smaller quantity of essential components. It uses a mathematic based approach called Principal Component Analysis (PCA). PCA is a statistical approach used to reduce the variables in a dataset by finding its principal components. To implement an EIGENFACE algorithm, a training set N by N face images needs to be created. Each face image in the training set will be converted in vector form. Those vectors are normalized by finding the average face vector (which is depicted by the average features of the training set) and subtracting it from each image vector. Those normalized face vectors are the columns of a matrix A used to calculate a covariance matrix C. The columns of the covariance matrix C are the eigenvectors depicting the principal components of the training dataset. K eigenvectors with the higher eigenvalues are chosen to represent the training images. Each training image then is replaced by a linear combination of the k eigenvectors plus the average face vector that was removed previously. The weights for each linear combination are stored. To determine the emotion conveyed in a test image, it is first converted into a vector image. Its weights are calculated. The Euclidian distance between the test weights and the EIGENFACEs set is computed. The vector that gives the smallest Euclidian distance is used to classify the conveyed emotion in the test image.

3.6 Communicating using Facial Gestures

There have been many developments in the domain of facial gesture recognition, especially for expression recognition. However, it appears the amount of research and studies done around facial gesture for human communication is limited. The need for communication systems with a limited use of muscle movement is existent. There are some people who do not have the ability to use their entire body. People that are extremely paralyzed need to be able to communicate and use day to day devices without much external help.

3.6.1 Head Motion Controlled Devices

A big part of the differently abled society has quadriplegic or nonverbal handicaps from strokes, traumatic brain injuries, or cerebral palsy. They struggle with communicating their thoughts to others. They often use limited motions, such as eye blink, wink, or head nods, or they rely on family and friends to help them out. In recent developments in the domain of computer vision, researchers are creating and implementing new technologies for people to be able to use a computer without the need of a mouse or keyboard or anything that requires the use of hands. Being able to use a computer or any machine without the use of hands would be beneficial for handicapped people who do not have the ability to use their hands as it allows them to communicate their thoughts and needs. These technologies allow them to use the internet, surf on social media, play video games, and access computer controlled technologies.

For people who have retained the ability to move their head, there exists various alternative technologies to a commercial mouse. In [48], the author developed a system that uses special glasses with LED marks to determine position and posture of the glasses through image processing. Those different positions determine the movement of the head. These head movements control the cursor on the screen of a computer, and a switch is used for mouse click. [49] and [50] presents a similar technology where infrared transmitters are mounted on eyeglasses or helmets.

In [47], the authors developed the Camera Mouse, a free program that is currently being used by thousands of people to control a computer with the sole use of the head. In their initial development, they use a first computer to visually track head movements, then they feed that to a second computer that the user is trying to control. Multiple features on the face were tested to determine which feature will be ideal to track. They experimented with the eyes, lips, nose, and a thumb. They concluded that any feature on the face can be selected for tracking as they will give equally good results in general.

Authors in [51] developed a contactless gesture based system that detects and tracks laser speckle pattern. A noncontact laser-based optical motion sensing device is placed by the computer. It projects laser beam on the surface of the skin. It produces speckle pattern. If the reflecting surface moves, the pattern will move proportionally. That feature is used for motion estimation.

3.6.2 Face Motion Controlled Devices

The head is not the only way of communicating for people with disabilities like cerebral palsy. Although the systems described above are widely used, there exist other systems that solely rely on facial muscle motions to control devices such as a computer. In "Communication via Eye Blink" [52], they develop an algorithm to detect eye blinks to measure their durations and interpret them to control a computer. Several techniques exist to detect the eye region.

- Flow field [53]: Gradient directional field describes the pattern of apparent flow of an object. In the eye, the dark aspect of iris over a light background causes a gradient direction field.

Its characteristic is that it is an outward flow field and intersects in a direction opposite of the gradient.

- Color-based technique [54]: This technique uses Bayes decision rule to decide from the color of a pixel if it is part of the sclera of an eye. In the process, the white of the eyes from training images is segmented in greyscale. Then, a distribution is plotted for both sclera and non-sclera region. The likelihood of both is computed, and a decision threshold is calculated using a cost ration function.

Other examples of eye tracking algorithms include horizontal gradient maps of a skin-colored region [55], [56], and pupil detection using infrared lighting [57], [58].

3.7 Software

In this project, a couple of software are used: Python, OpenCV, and Dlib.

3.7.1 Python

Python is a high-level programming language created by Guido Van Rossum. It makes use of white space and indentation to be readable and has a syntax that permits us to write code in fewer lines, not like C or Java. Python supports object-oriented, function, and imperative programming. Its availability on many operating systems made it the perfect choice as the programming language for this project. The core philosophy of Python is summarized in [59] and includes:

- Beautiful is better than ugly
- Explicit is better than implicit

- Simple is better than complex
- Complex is better than complicated
- Readability counts

Python is an extensible programming language that makes it fit to be embedded in applications that need a programmable interface. Python rejects the philosophy "there is more than one way to do it" and was created with the mindset that there should be one obvious way to do it.

Python does things simpler when it comes to programming. With Python, there is no need of curly brackets to identify blocks, and it has fewer syntax exceptions. It also has a larger library providing tools and guidance. The library provides different modules to create graphical user interfaces, to connect to databases, to generate random variables, to do unit testing and many more. Python can be used within web applications. It makes use of libraries such as Numpy to do scientific computing. It has also been embedded in many software solutions that are widely used today. Python has served in artificial intelligence tasks and in security systems. Programming languages such as Ruby, Swift, and Cobra were all influenced by Python and its simplicity.

3.7.2 OpenCV Library

OpenCV (for Open Computer Vision) is a free graphical library, originally developed by Intel, specializing in real-time image processing. The OpenCV library provides many widely varying functionalities to create programs from raw data to basic graphics interfaces. It proposes most of the conventional operations in low-level processing of images:

- Reading, writing and displaying an image

- Calculating the histogram of gray levels or color histograms
- Smoothing, filtering
- Image thresholding (Otsu method, adaptive thresholding)
- Segmentation (related components, GrabCut)
- Mathematical morphology

Some classical algorithms in the field of artificial learning are also available:

- K-means
- AdaBoost and various boosting algorithms
- Artificial neural network
- Large Margin Separator
- Estimator (statistical)
- Decision trees and random forests

Since OpenCV version 2.1, the focus has been on matrices and operations on them. Indeed, the basic structure is the matrix. An image can be considered as a pixel matrix. Thus, all basic operations of the matrices are available:

- The transposed
- Calculation of the determinant
- Inversion
- Multiplication (by a matrix or a scalar)
- Calculation of eigenvalues

It also provides some GUI functions, such as slide cursors, controls associated with mouse events, or text overlay in an image.

3.7.3 Dlib Library

Dlib is a contemporary open source C++ library that performs a diversity of machine learning algorithm, such as data transformation, structured prediction, regression, clustering, and classification [60]. It processes multiple algorithms, such as:

- K-Mean clustering: partitioning n observations in k clusters. In image processing, it is used to segment an object from its background.
- Bayesian network: a model that represents variables and their conditional dependency from each other through the mean of a probabilistic graphical model. It can be used for classification in machine learning and image processing.
- Support vector machine: It is a machine learning algorithm mainly used in classification problems

Dlib is not just for machine learning. It includes the following functionalities:

- Threading: simple threading APIs, threaded functions and objects, a timer object to generate events spaced in time.
- Image processing: Reading and writing common image formats, edge finding, morphological operations, color space conversion, face recognition.
- Numerical algorithms: singular value, transpose, trigonometric functions, transpose, Optimized cutting plane algorithm, quadratic problem solvers.
- Networking: HTTP server, iostream and streambuf objects
- Data compression and integrity algorithm

Dlib has a vast amount of open source documents to aid in research and machine learning application. It works on Windows, Linux, and OS X.

3.8 Morse Code

The Morse alphabet, or Morse code [62], is a code for transmitting text using short and long series of pulses, whether produced by signs, light, sound or gesture. This code is often attributed to Samuel Morse. However, many dispute this, and tend to attribute the paternity of the language to his assistant, Alfred Vail [61]. Morse code is mainly used by the military as a means of transmission, often encrypted, as well as in the civilian sector for certain programs of an automatic nature: aviation, call signs for maritime stations, international transmitters (atomic clocks) or for maritime signaling by certain radar transponders, known as "Morse letters". Morse code is also practiced by scouts (light and sound Morse code), divers or alpinists (light Morse code), by players to solve puzzles, as well as default ring of reception of message for the Nokia mobile phones.

3.8.1 Morse Code Representation

Two symbols, called dot and dash (or "dit" and "dah") are used, and two spacing times, the elementary cut between signals and the space separating the words. The total duration of transmission of a stroke (including the elementary cut between signals) determines the speed at which the message is sent, and is used as a reference rate.

3.8.2 Morse Code Learning Method

There are plenty of popular methods people use to learn Morse code. This section presents some of the most used ones.

- Koch method: It was invented by a German psychologist, Ludwig Koch, in the 1930s. It is one of the methods for fast learning. The Koch method requires a computer or a teacher to listen to the code. By starting immediately with a speed greater than 12 words per minute, it allows to learn to listen to the correct Morse code. It also allows the character recognition by reflex and without reflection phase.

- Farnworth method: In his method, Donald R. "Russ" Farnsworth proposes to use the target speed for learning (for example, start at 20 WPM) but with higher inter-word and inter-letter spaces than required by the target speed. It thus gives more time to the understanding of each sign, while using a high speed from the start for the recognition of signs. Both Farnworth and Koch method can be combined, starting at 20 words / minute, with 2 characters, with triple spaces compared to normal, for example.

- Mnemonic learning: Morse code is easily memorized using short and long codes replaced by syllables. The long (-) code replaced with an "o" syllable. The short code (.) replaced by one of the other vowels.

CHAPTER 4

FACIAL GESTURE RECOGNITION FOR MORSE CODE DESIGN AND IMPLEMENTATION

In this thesis, we present a novel human-computer interface for communication via facial and head movements, notably eye blinks, open and close mouth, and head movements. The system's diagram goes as follow:

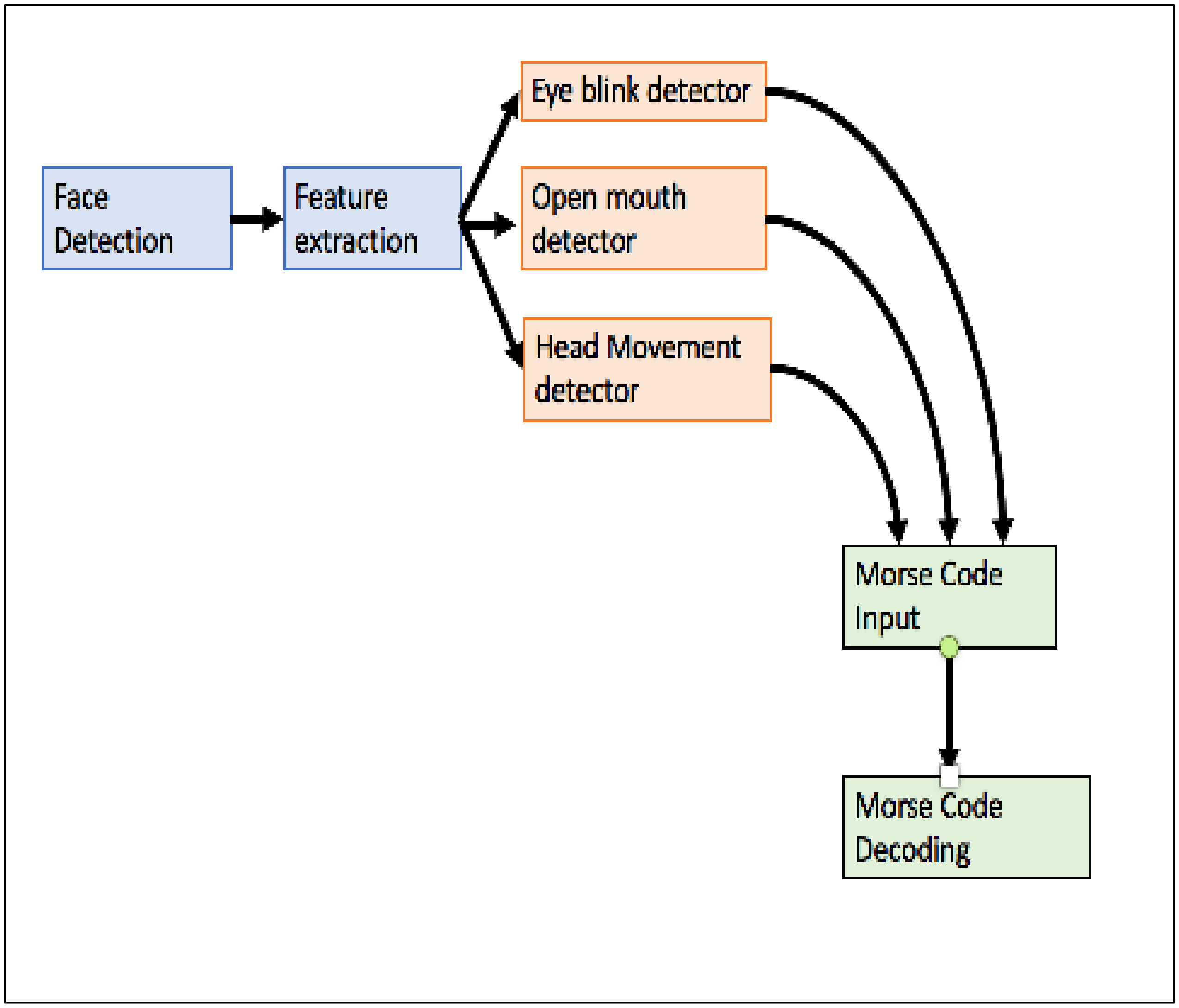


Figure 4.1: Human-computer interface system block diagram

4.1 Face Detector Training

An object detector was created using Dlib. A face detector is trained on the dataset

Labeled Face in the Wild [41]. This dataset was chosen for its wide variation of subjects under different conditions. The training was done on a set of 112 randomly chosen images from LFW database. The trainer is a support vector machine (SVM) that returns the "best fit" hyperplane to categorize the set of images. Rectangles are manually drawn on the training set as shown in Figure 4.3. An XML file containing the images and the coordinates of all the bounding boxes is created using "Imglad" tool kit within Dlib. The XML file is fed to Dlib object trainer, and it returns the detector. A Histogram of Oriented Gradients (HOG) filter as shown in Figure 4.4.

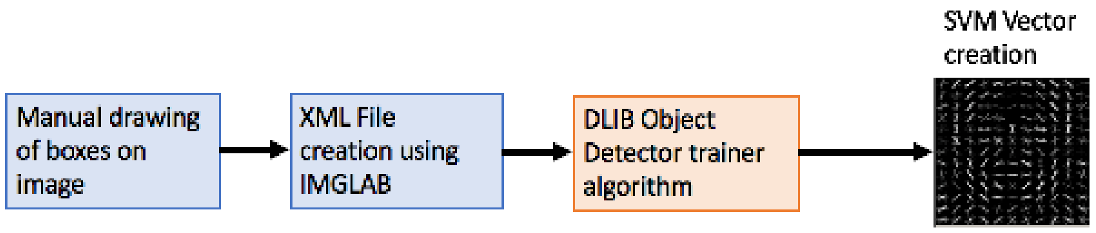

Manual drawing of boxes on image
XML File creation using IMGLAB
DLIB Object Detector trainer algorithm
SVM Vector creation

Figure 4.2: Face detection algorithm

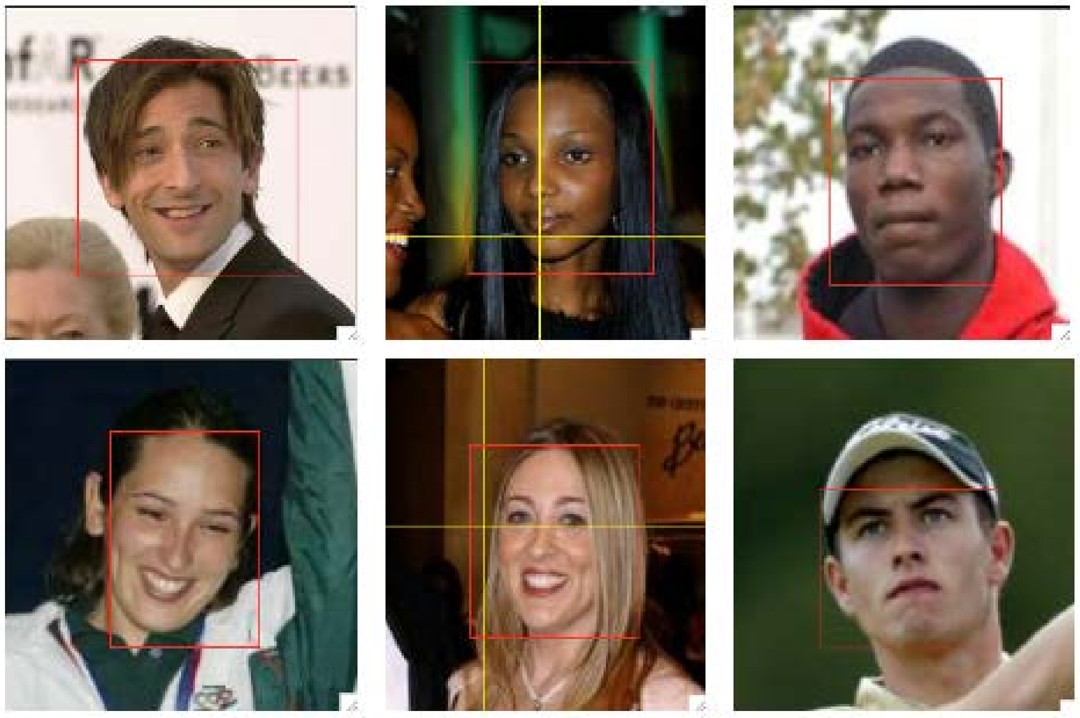


Figure 4.3: Examples of bounding boxes manually drawn on images from LFW database

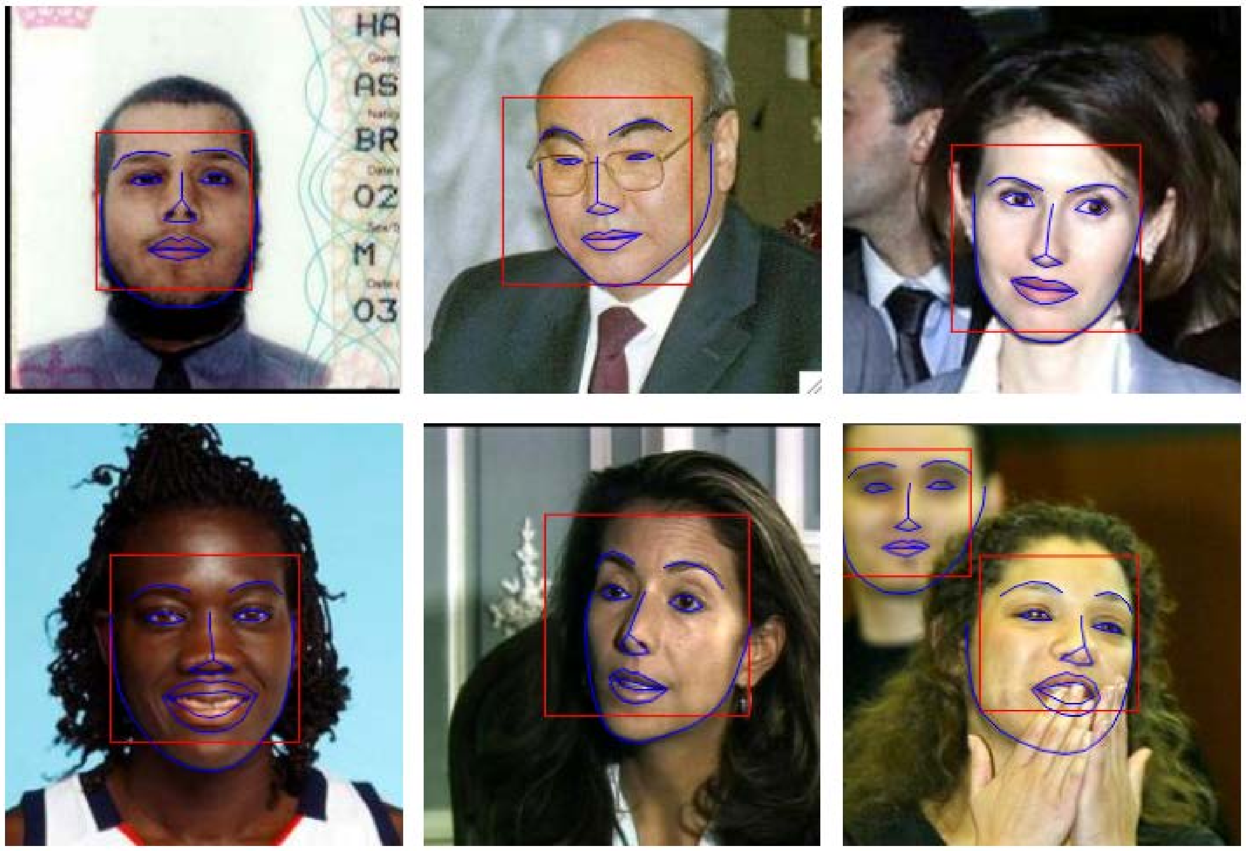


Figure 4.4: Facial landmark and bounding box on images from iBUG 300-W dataset

To detect the facial features, Dlib facial landmark detector was used. It is a pre-trained detector of 68(x, y)- coordinates trained on the iBUG 300-W dataset [63]. Figure 4.3 represents facial landmarks detected using the pre-trained detector from Dlib on images acquired from Labeled Face in the Wild Database.

4.2 Feature Extraction

In the facial movement communication with Morse code, the blink of an eye is mapped as a dash (-), the opening of the mouth is mapped as a dot (.), and a shake is mapped as space to separate letters and words.

To detect blinks, the eyes features must be extracted. The exact coordinates of the eyes are extracted from the facial landmark. In Figure 4.5, the exact location of each point of the facial landmark is depicted.

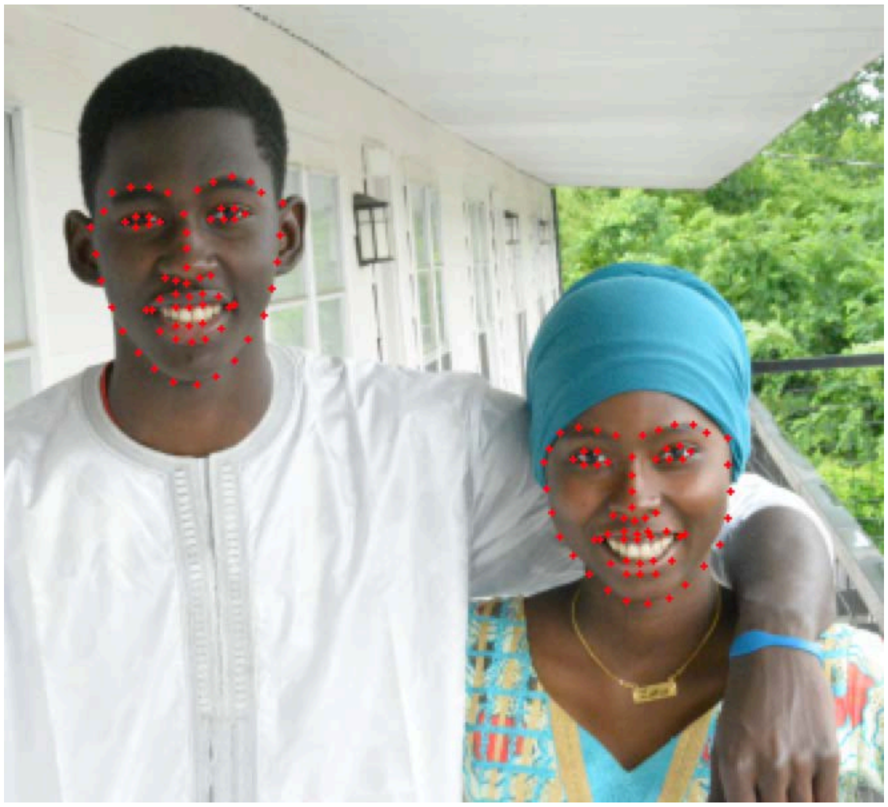


Figure 4.5: Exact location of points of facial landmarks depicted

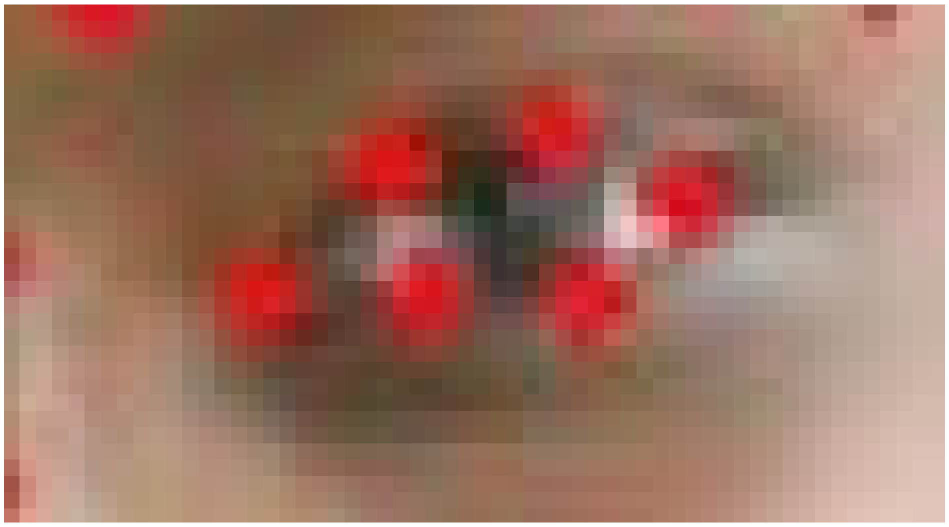


Figure 4.6: Points location on eye.

The block diagram in Figure 4.7 explains the algorithm used to detect an eye blink. It was borrowed from [65]. Each facial feature can be accessed by indexing. The eyes can be accessed through points [36, 42] and [42, 48] for the right and left eye respectively in the facial landmark dictionary inside one of Python's library, "imutils". As we zoom in on the eyes, each of them are represented by 6 coordinates labeled A1, A2, A3, A4, A5. To detect blinks in real time, we used the eye aspect ratio (EAR) equation from [64]. It estimates how open the eye is.

$$EAR = \frac{\|A2 - A6\| + \|A3 - A5\|}{2\|A1 - A4\|}$$

The EAR for each eye is calculated and they are averaged to give the EAR of both eyes blinking. As the eyes close, the distance between the points grow smaller and reach near 0.

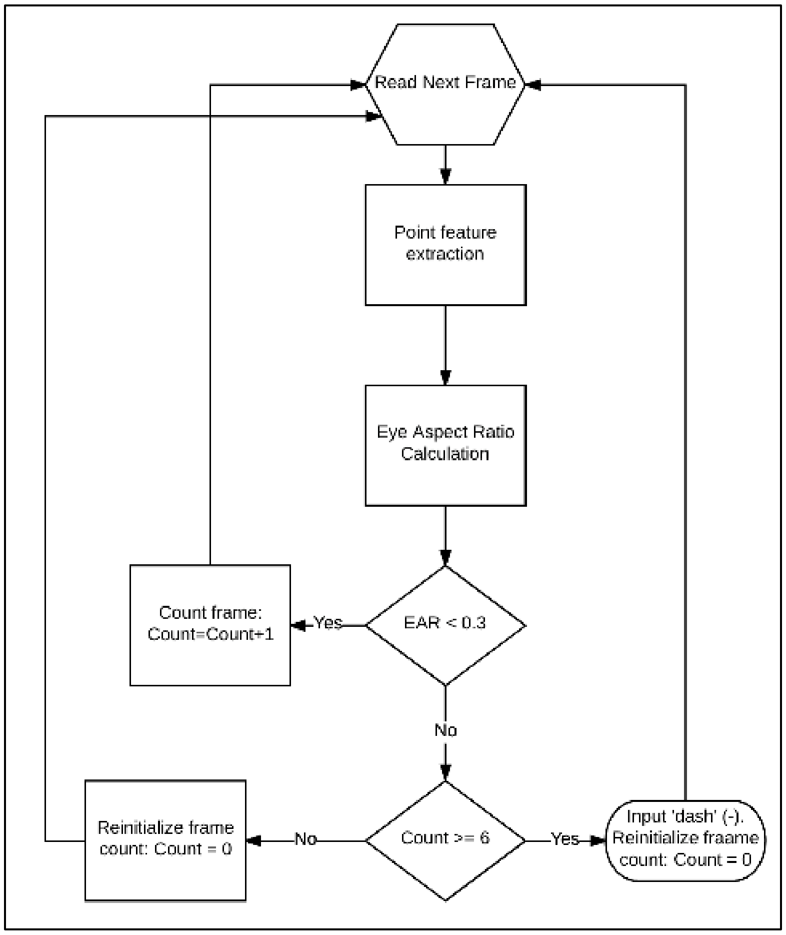

Read Next Frame
Point feature extraction
Eye Aspect Ratio Calculation
EAR < 0.3
Yes
Count frame: Count=Count+1
No
Count >= 6
No
Reinitialize frame count: Count = 0
Yes
Input 'dash' (-). Reinitialize fraame count: Count = 0

Figure 4.7: Block diagram of eye blink detector algorithm

The same concept is applied to the mouth. The mouth can be accessed through points [48, 68]. As depicted on Figure 4.8, the inner line of the mouth is depicted by 8 points, B1, B2, B3, B4, B5, B6, B7, and B8. The mouth aspect ratio (MAR) is determined by the equation below.

$$MAR = \frac{\|B2 - B8\| + \|B3 - B7\| + \|B4 - B6\|}{3\|B1 - A5\|}$$

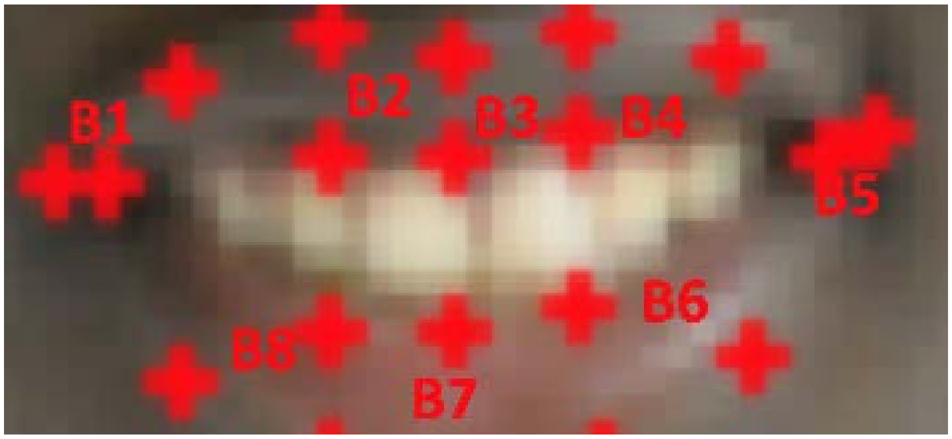


Figure 4.8: Points location on inner line of mouth

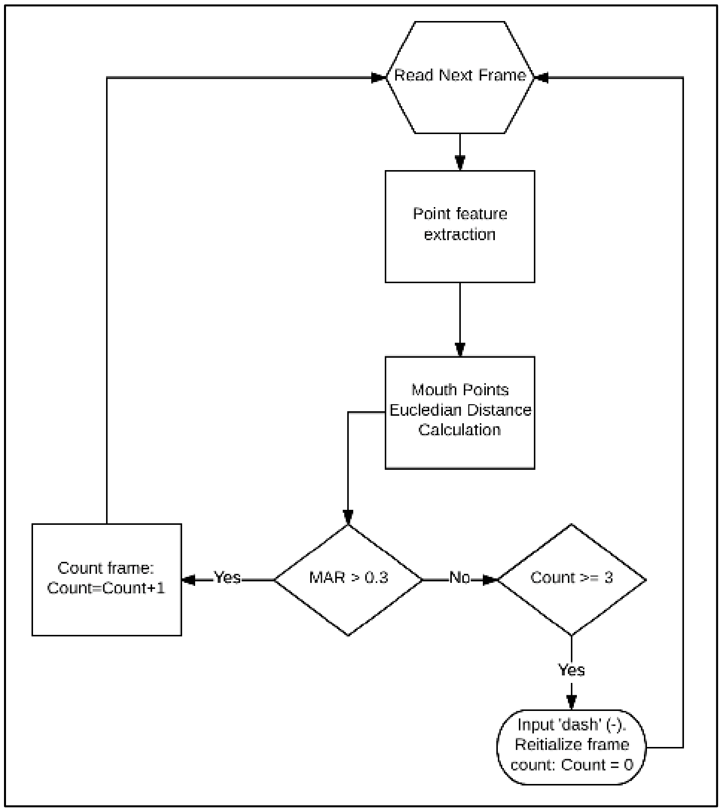

Read Next Frame
Point feature extraction
Mouth Points Eucledian Distance Calculation
MAR > 0.3
Yes
No
Count frame: Count=Count+1
Count >= 3
Yes
Input 'dash' (-). Reitialize frame count: Count = 0

Figure 4.9: Block diagram of the open mouth algorithm

In contrary to eye blink, an open mouth is most of the time voluntary. It would be intuitive to put the threshold at zero. At rest, the MAR is at about 0.05. As the mouth opens and widens, the MAR grows bigger. When the MAR passes a certain threshold, subsequent frames are monitored to keep count of their MAR. If its value is above the threshold value for at least 3 frames (300 milliseconds), then a 'dot' (.) is recorded as input to the Morse code detector.

To detect a head nod or a head shake, a center coordinate on the face was chosen. The top point of the nose landmark as depicted in Figure 4.10, was judged best to use because of its center location. It is depicted as point 28 in the pre-trained facial landmark detector.

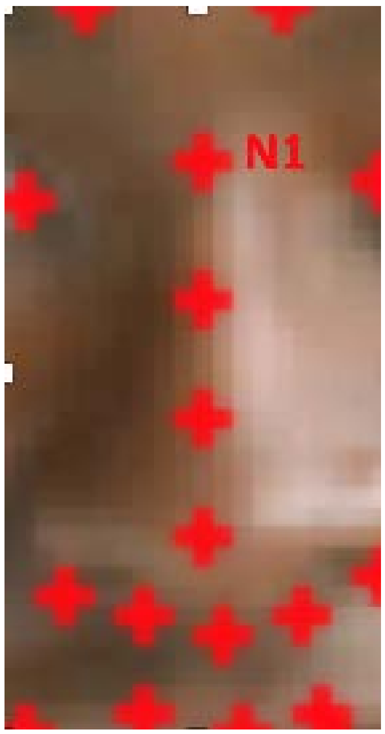


Figure 4.10: Point locations of nose bridge.

The x and y coordinates of point N1 are extracted from the facial landmark. The head nod/shake detector was tested to find the threshold for the Morse code detector. After testing, the conclusion was that an intentional nod or a head shake takes at least 5 frames (500 milliseconds). It was also concluded that during the head movement, a displacement of at least 10 pixels is made in either directions. For every 5 new frames, N1's coordinates are compared to that of the old 5 frames to detect a head shake or nod. A nod is detected by a change in the y direction bigger than 10 pixels. A shake is depicted by a change in the x direction bigger than 10 pixels. Figure 4.11 depicts the block diagram of the head shake detector.

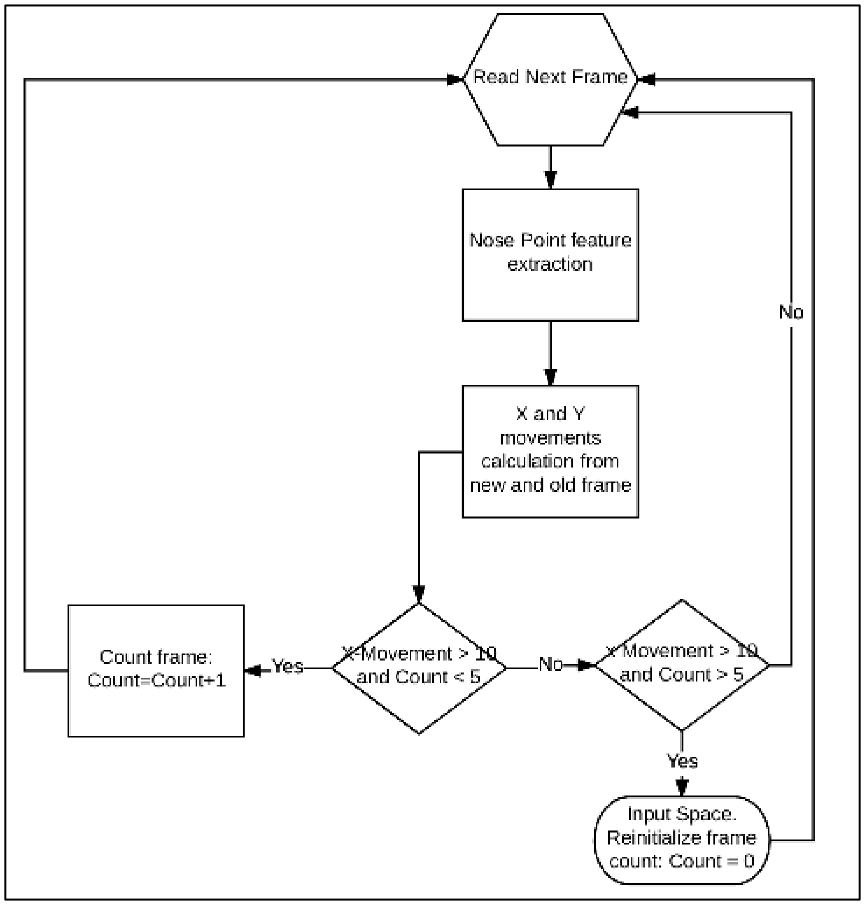

Read Next Frame
Nose Point feature extraction
X and Y movements calculation from new and old frame
No
Count frame: Count=Count+1
Yes
X-Movement > 10 and Count < 5
No
x Movement > 10 and Count > 5
Yes
Input Space. Reinitialize frame count: Count = 0

Figure 4.11: Block diagram of the head shake/node algorithm

4.3 Classification

As discussed previously, each extracted movement will be mapped to a Morse symbol. An eye blink corresponds to a dash (-), as depicted in Figure 4.12. An open mouth corresponds to a dot (.), as shown in Figure 4.13. A head shake corresponds to a space to separate letters and words.

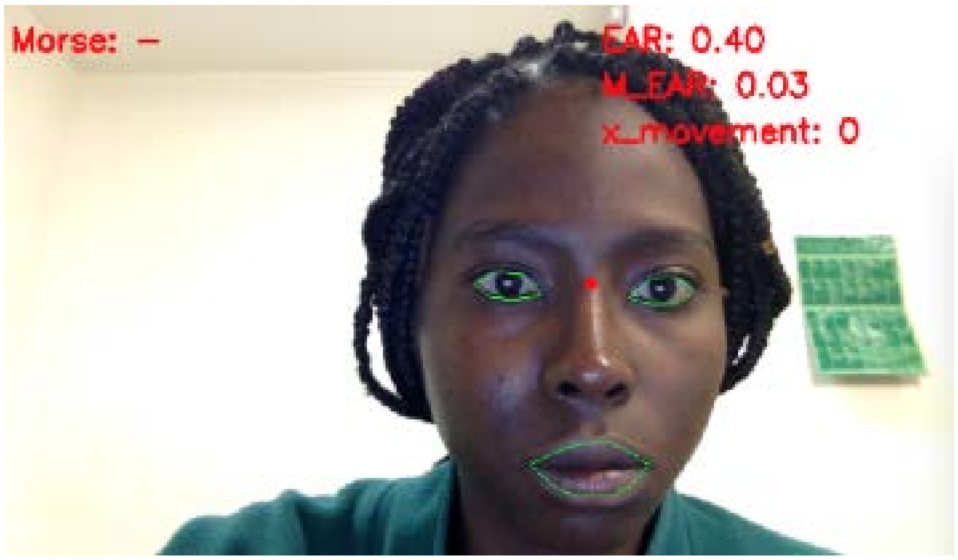


Figure 4.12: Depiction of eye blink being translated into a dash

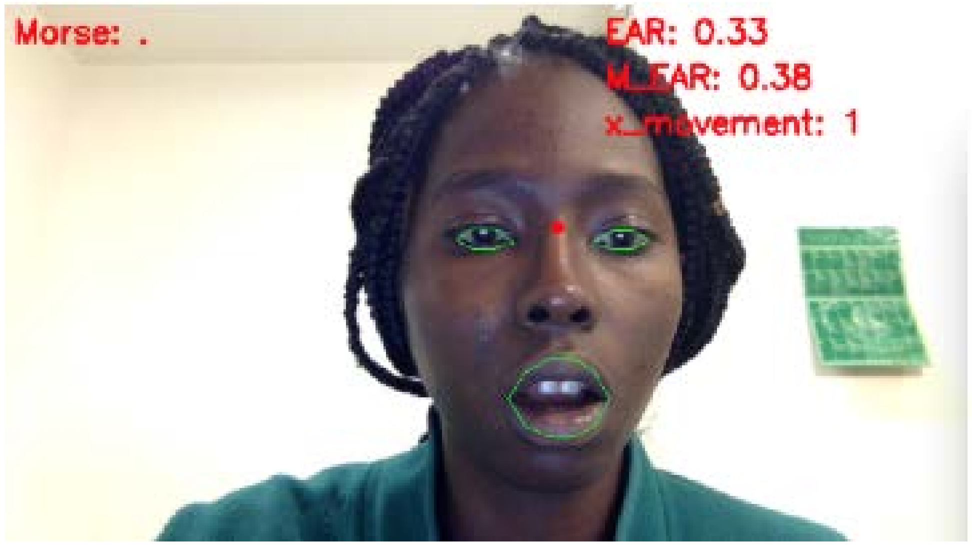
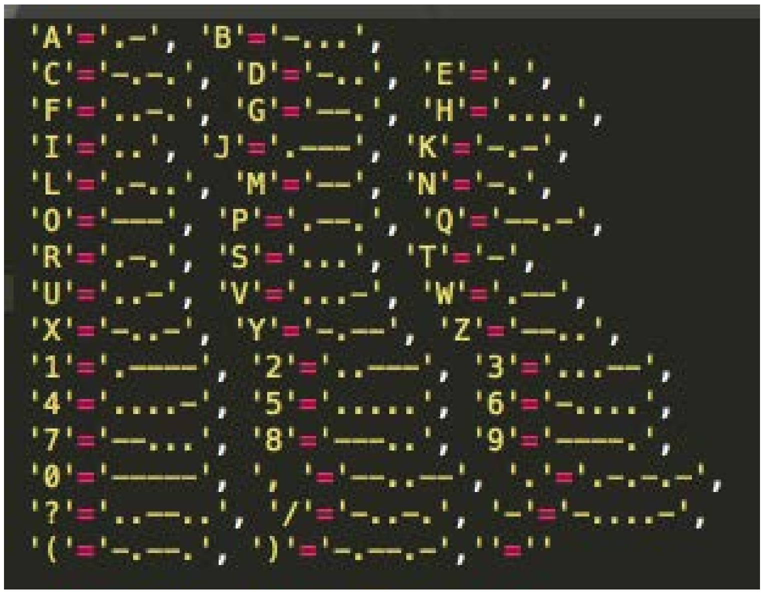


Figure 4.13: Depiction of an open mouth translated into a dot

Using these two symbols, the user can spell out words with the help of the following dictionary of Morse code.

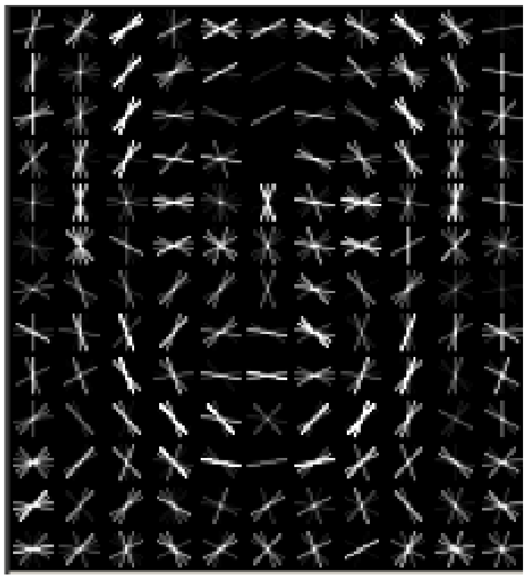


```
'A'='.-', 'B'='-...',
'C'='-.-.', 'D'='-..', 'E'='.',
'F'='..-.', 'G'='--.', 'H'='....',
'I'='..', 'J'='.---', 'K'='-.-',
'L'='.-..', 'M'='--', 'N'='-.',
'O'='---', 'P'='.--.', 'Q'='--.-',
'R'='.-.', 'S'='...', 'T'='-',
'U'='..-', 'V'='...-', 'W'='.--',
'X'='-..-', 'Y'='-.--', 'Z'='--..',
'1'='.----', '2'='..---', '3'='...--',
'4'='....-', '5'='.....', '6'='-....',
'7'='--...', '8'='---..', '9'='----.',
'0'='-----', ', '='--..--', '.'='.-.-.-',
'?'='..--..', '/'='-..-.', '-'='-....-',
'('='-.--.', ')'='-.--.-',''=''
```

Figure 4.14: Morse code dictionary

4.4 Experiments and Results

In the experiment, the Histogram of Oriented Gradient filter for face detection was first produced using Dlib object detection trainer. Figure 4.15 below shows the result of the trainer compared to that of Dlib's example file.



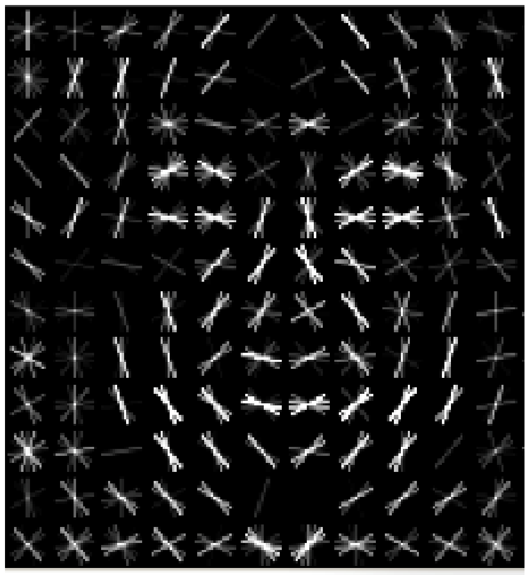


Figure 4.15: HOG filter from experiment file and HOG filter from DLIB example file

The final image is a HOG filter that resembles a face. Evidently, the trainer processed all sub-windows of every image and added up the rate of changes between a pixel and its neighborhood. We are left with general feature description of the item enclosed in the bounding box we introduced. At first, it looks like the experiment had poor results compared to that of the DLIB example file. However, Table 4.1 shows that both trainers achieved equal accuracy when the detectors are tested on both the training set of data and a testing set of images they were not trained on. What makes the HOG filter from the DLIB example file stands out better is the fact that it only used 4 images, whereas the experimental XML file had over 100 images to account for a broader variety of facial position.

TABLE 4-1: Experimental training and Dlib training comparison.

	Experiment Training set			Dlib Training set		
	Precision	Recall	Average	Precision	Recall	Average
Training Accuracy	1	1	1	1	1	1
Testing Accuracy	1	1	1	1	1	1

After face detection, the facial landmark of the user is detected using DLIB internal facial landmark detector. The facial landmark of the face in real time is depicted in Figure 4.16. To achieve that, the coordinates of the landmarks are extracted and used to draw contour on the detected face.

After facial landmark detection, the key features of the face are extracted as coordinates. Those are used to test the eye-blink detector, the open mouth detector, and the head nod/shake detector. Figure 4.17 depicts the behavior the blink detector for Morse code.

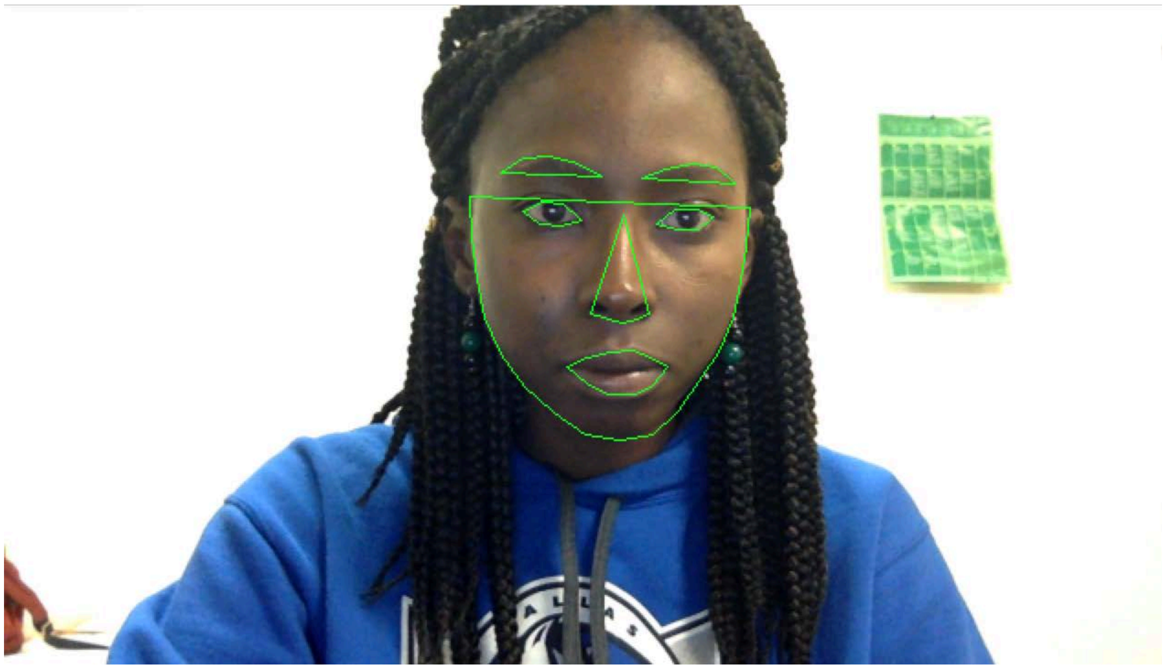


Figure 4.16: Facial landmark displayed in real time video.

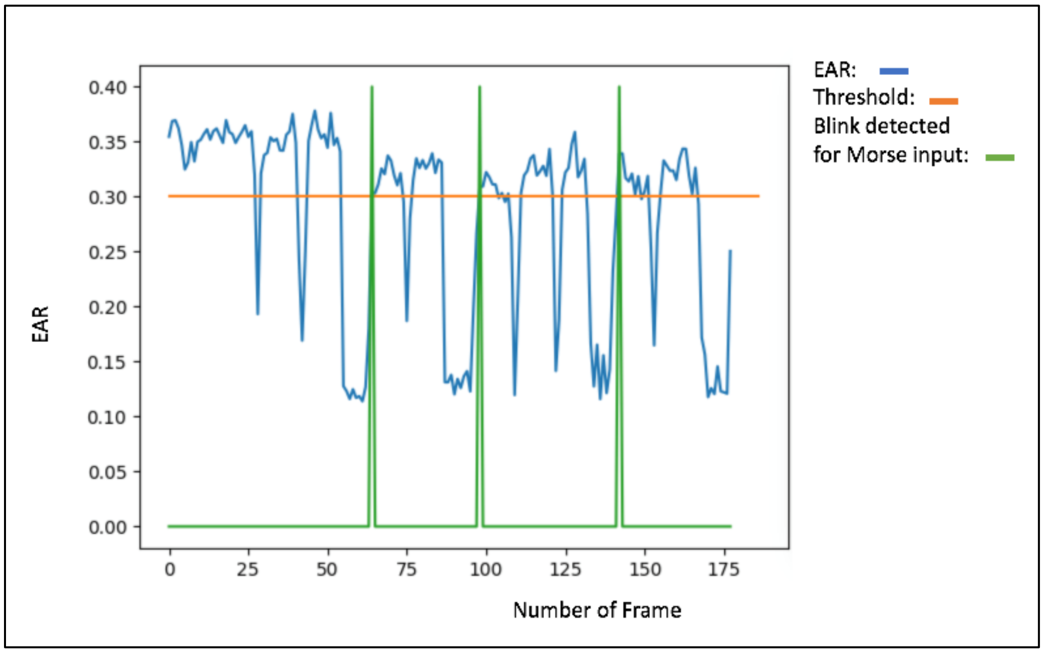


Figure 4.17: Graph of the blink detector behaviour

The blink detector was first tested to decide on an optimum threshold for the EAR. A threshold of 0.3 was chosen at first. As can be seen in Figure 4.17, The EAR seems to stay in the surrounding of 0.35 when eyes are open, but that gradually decreases to 0.3. After testing on different subjects, the threshold was changed to 0.25 to accommodate for smaller than average eyes. People usually blink for about 100 to 400 milliseconds. A threshold of about 6 frames was chosen to detect an intentional blink for the Morse code detector. 6 frames correspond to about 600 milliseconds. The graph shows that unintentional blinks lasting less that 6 frames are ignored and not considered as an input to the Morse code detector. Only blinks lasting for at least 6 frames (600 milliseconds) are considered. The inputs to the Morse code detector are depicted by the green high peaks.

The open mouth detector was also tested separately for optimization. The graph below depicts the behavior of the open mouth detector.

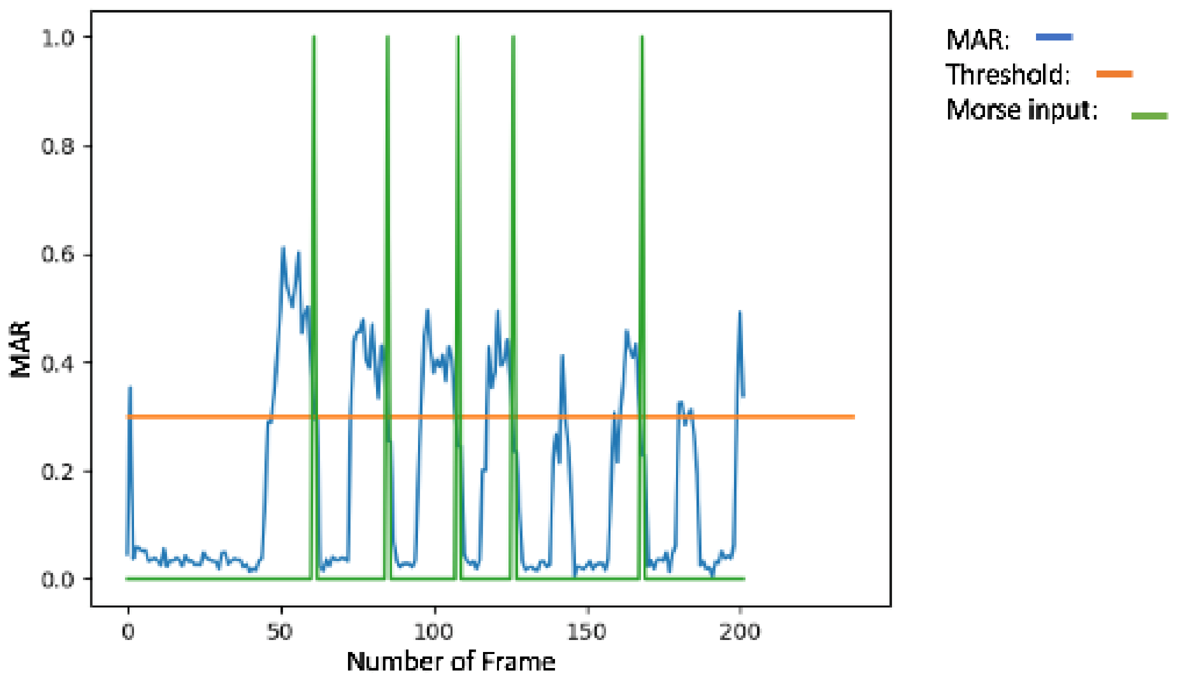


Figure 4.18: Graph of the open mouth detector behaviour

Most people have their mouth close at rest. An MAR threshold of 0 would be the obvious choice here. However, because some people may have their mouth slightly open at rest, a small MAR of 0.3 is chosen as a threshold. Evidently, when the mouth is slightly open at rest, it corresponds to an average MAR of 0.2. Figure 4.19 depicts the mouth states when it's closed and when it's slightly open at rest.

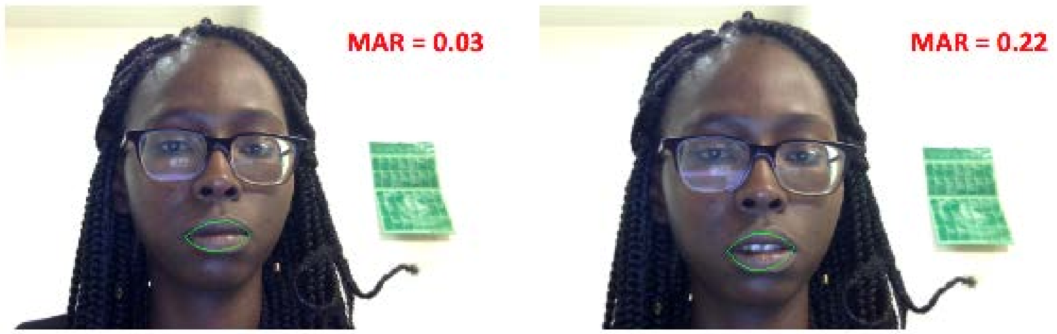


Figure 4.19: Depiction of the mouth state and its MAR

When performing a head nod or a head shake, one displaces their head in one direction and brings it back to the middle. This phenomenon is taken into account when producing the algorithm for head shake. As shown in the graph below, a head shake is counted only if there has been a change of position at least twice. Therefore, we can see on Figure 4.20 that the green graph peaks once for every 2 or 3 peaks of the blue graph.

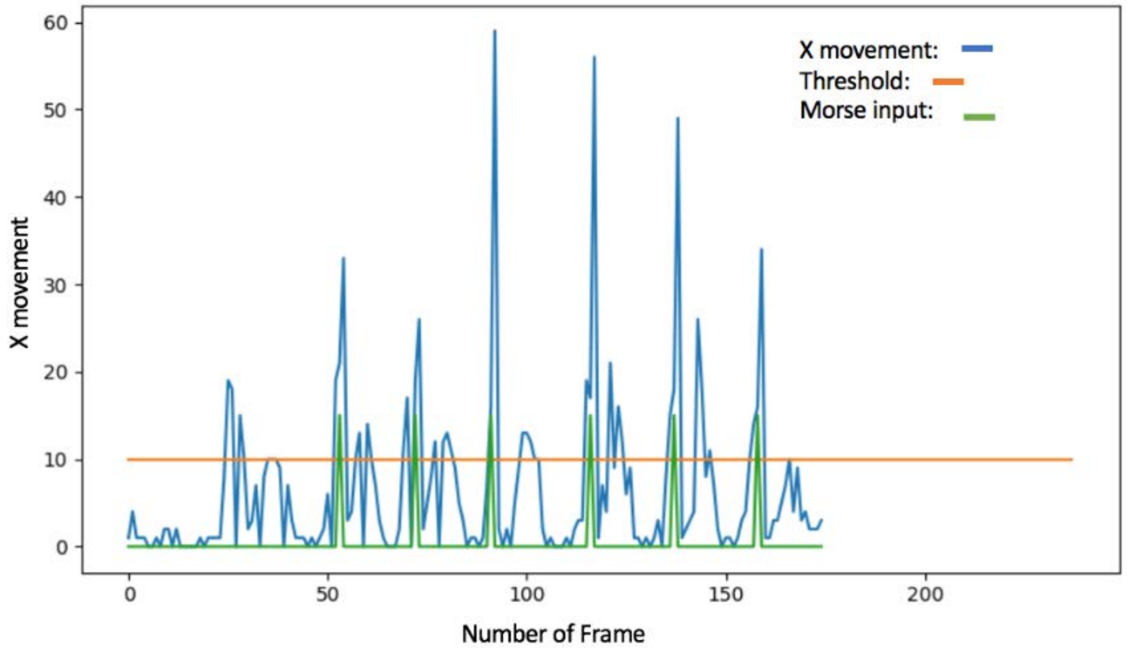


Figure 4.20: Graph of head movement behavior.

The method for head movement detection is fast and reliable, compared to the method presented in [66], in which the Haar-cascade method and OpenCV is used to detect the face region, and some mathematics equation was used to detect the center of the face region. As can be seen from FIGURE 4.21 below, the method in [66] detects head movements. However, when there is a brusque movement, the center point of the face is shifted, and the algorithm is not as reliable. The method in this thesis uses DLIB facial landmark, and the center coordinate of the face always stays in place.

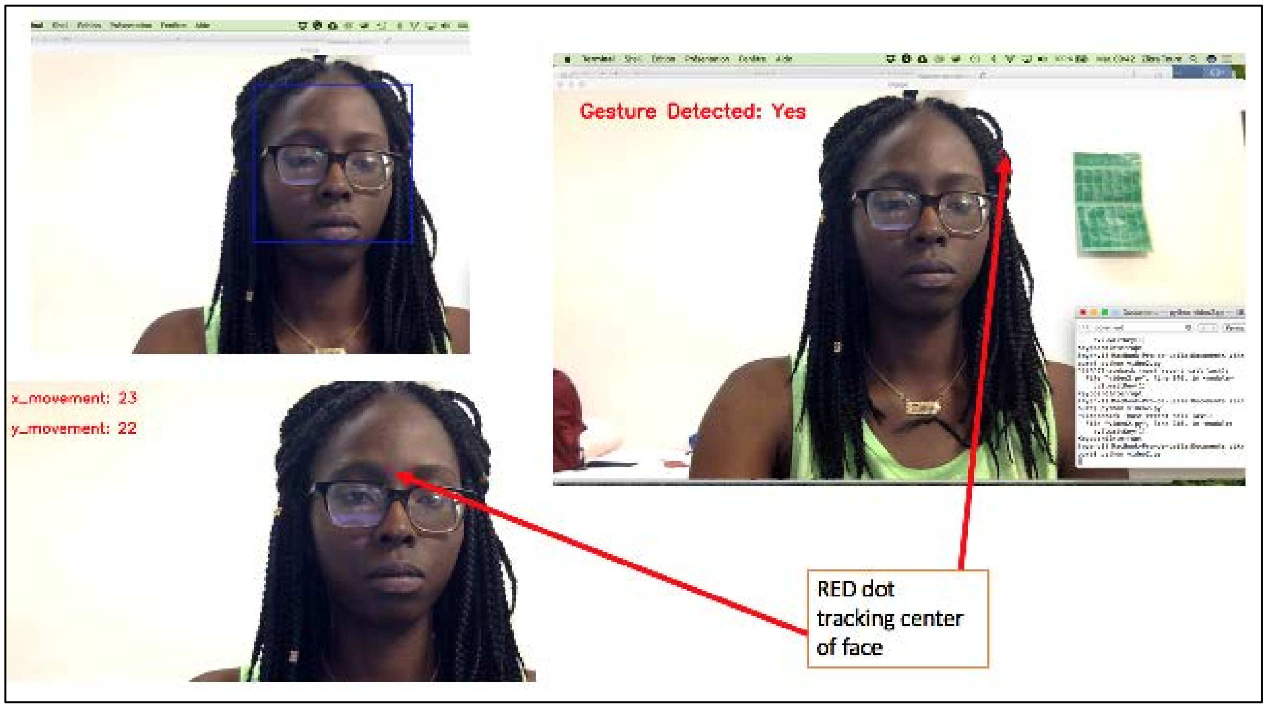


Figure 4.21: Depiction of head movement detector used in [66]

After testing of the individual detectors, the overall system was put under test in a control environment with effective lighting and a control distance between the user and the computer.

A distance no smaller than 25 inches was put between the user and the computer. In a first experiment, the system was used to spell out generic words, such as “Hello”.

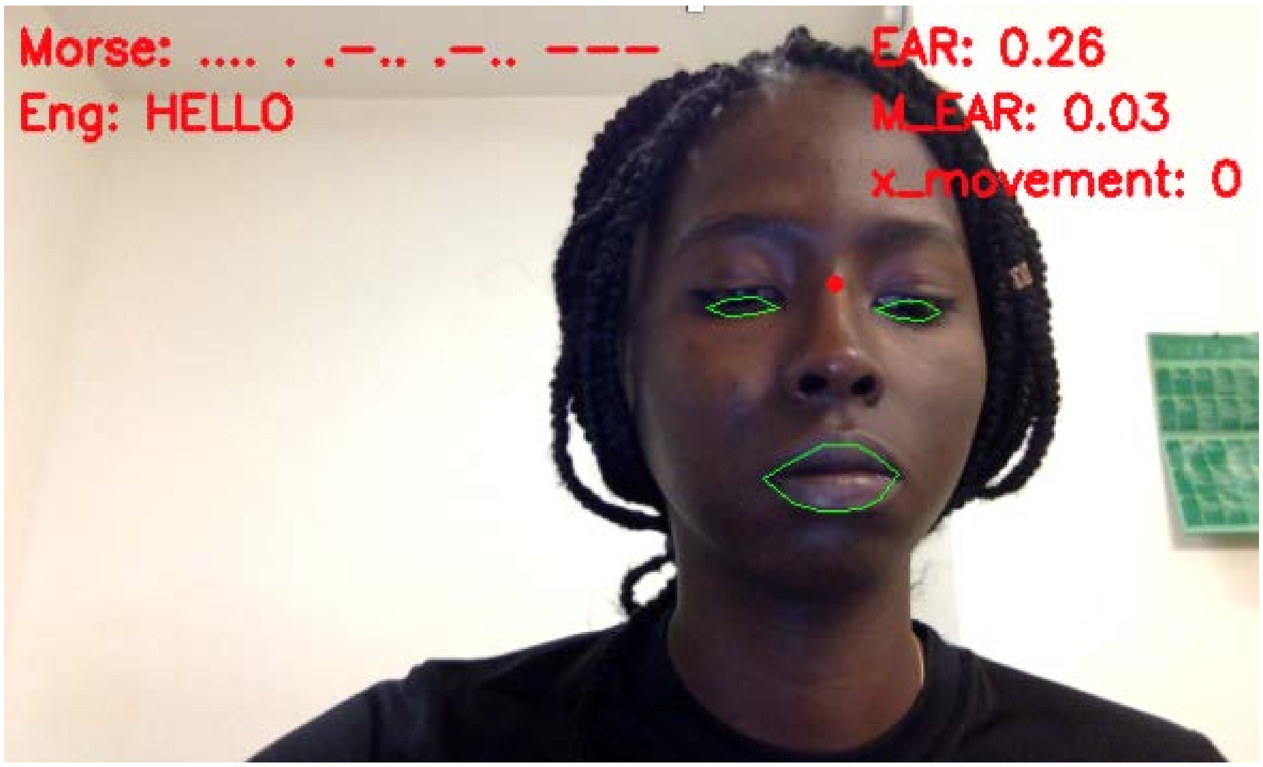


Figure 4.22: Depiction of the system being used to spell out "Hello"

The graph below depicts the changes in EAR in orange, MAR in green, and x-movement in blue. The red circles on the graph tell when a dot in taken as an input. The blue short sticks tell when a dash is taken as an input. The event of space is marked on the graph.

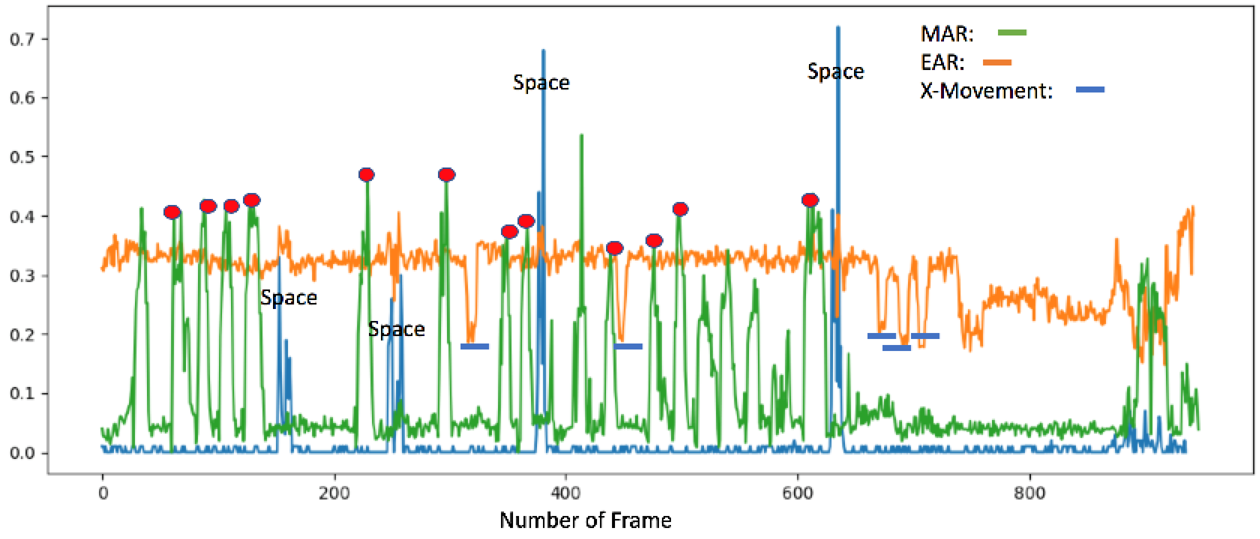


Figure 4.23: Graph depicting each detector's behaviour on first trial

As can be seen on the graph, the spell out of the word "Hello" took about 712 frames which is roughly 71.2 seconds. The eye blink detector gave no false positive of true negative. The head shake showed similar results. The mouth open detector, however, gave 5 true negatives out of 17 trials during the spelling of the word. Table 4.1 shows the results of the experiment in terms of success rate and failure rate.

After multiple testing for the user to get familiar with the system, another try was recorded. Figure 4.23 below describes the behavior of the detectors for the 15th try at the spelling of the word "Hello". The system performed much better, as can be seen on Table 4-3. There were fewer errors and spelling time reduced to 28 seconds.

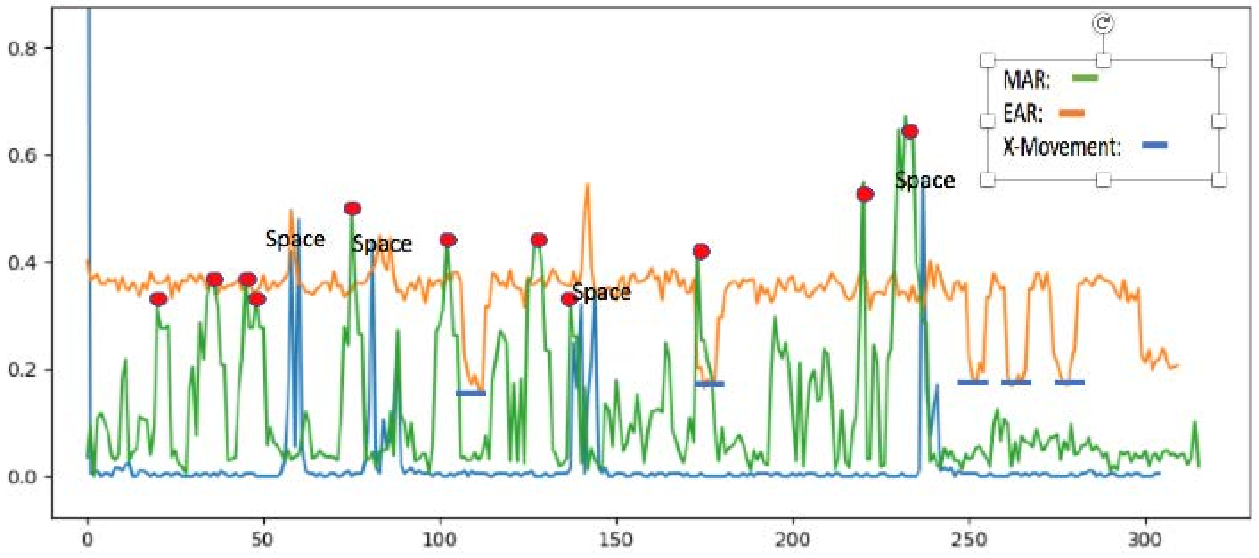


Figure 4.24: Graph depicting each detector's behaviour on 15th trial

TABLE 4-2: Success and failure for each detector for first trial

Detector	Number of tries	Success	Failure	Time
EAR	5	5	0	72 second
MAR	15	11	2	
X movement	4	0	0	

TABLE 4-3: Success and failure for each detector fro 15th trial

Detector	Number of tries	Success	Failure	Time
EAR	5	5	0	28 second
MAR	13	11	2	
X movement	4	0	0	

In a second experiments, 6 students were invited to test the system by spelling common phrase. At the beginning of the experiment, each student was explained how the system works, and they were given trial times to familiarize themselves with the system. Each student used the system to spell out the phrase "I WANT WATER".

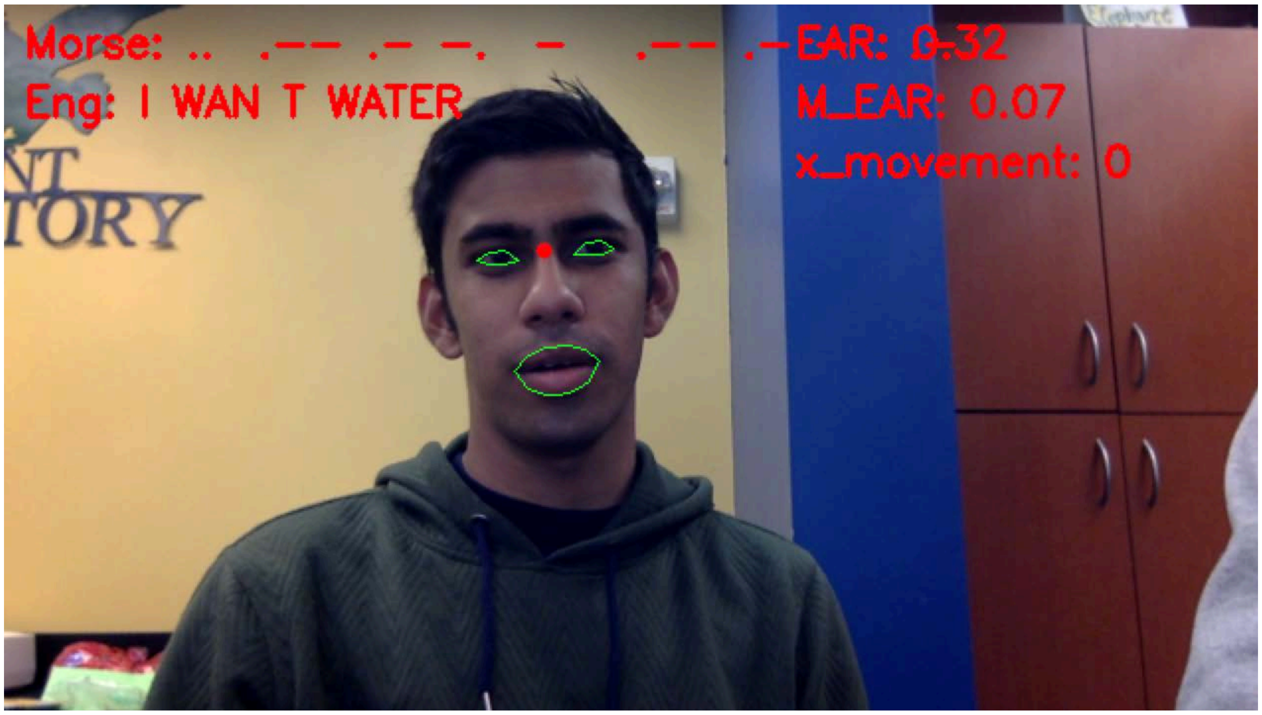


Figure 4.25: Student testing the system

The Morse code for "I WANT WATER" is depicted in Figure 4.25. The blue lines depict a space. The image was shown to the students for them to spell out the phrase. Also, for every Morse code, an oral description of the movement was given during testing in order to prevent errors due to unfamiliarity with Morse code. In the case in Figure 4.24, spelling took an overall time of roughly 980 frames (98 seconds).

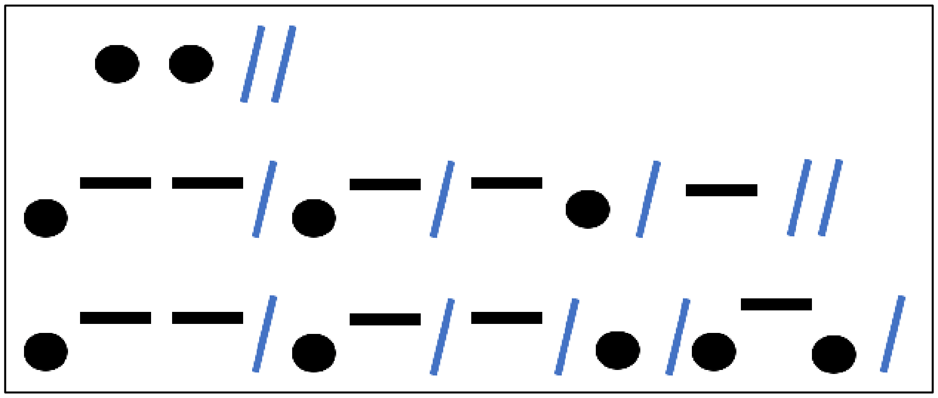


Figure 4.26: Morse code for "I WANT WATER"

The Morse code contains 10 dots, 10 dashes, and 12 spaces. As can be seen on the graphs below, the detector detected exactly 10 dots. It failed to detect two open mouths, for the student did not sustain the MAR value for 3 consecutive frames. It gave zero false positives. For the blink detector, it can be seen on the graph in Figure 4.27 that the Morse code detector had 11 dash inputs. The last dash input was not intentional. It was produced at the end of the spelling of the phrase.

The head movement detector produced the most false-positive. As depicted by the orange line in Figure 4.28, 14 space where recorded throughout the testing. 2 head spaces were recorded instead of one after spelling “N” in “WANT”. Therefore, it is seen in Figure 4.25 that “T” is separated from the word “WANT”. An extra space is again recorded after spelling out “T”, even though the head shake movement was used only twice. The last 3 spaces recorded were introduced at the end of the spelling of the phrase and were not intentional.

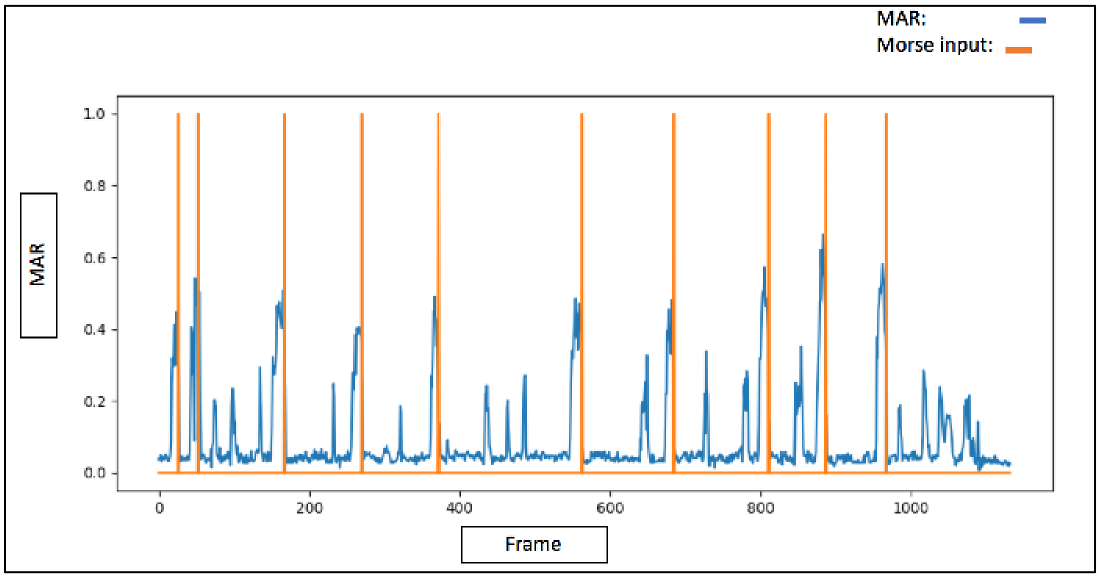


Figure 4.27: Mouth aspect ratio graph

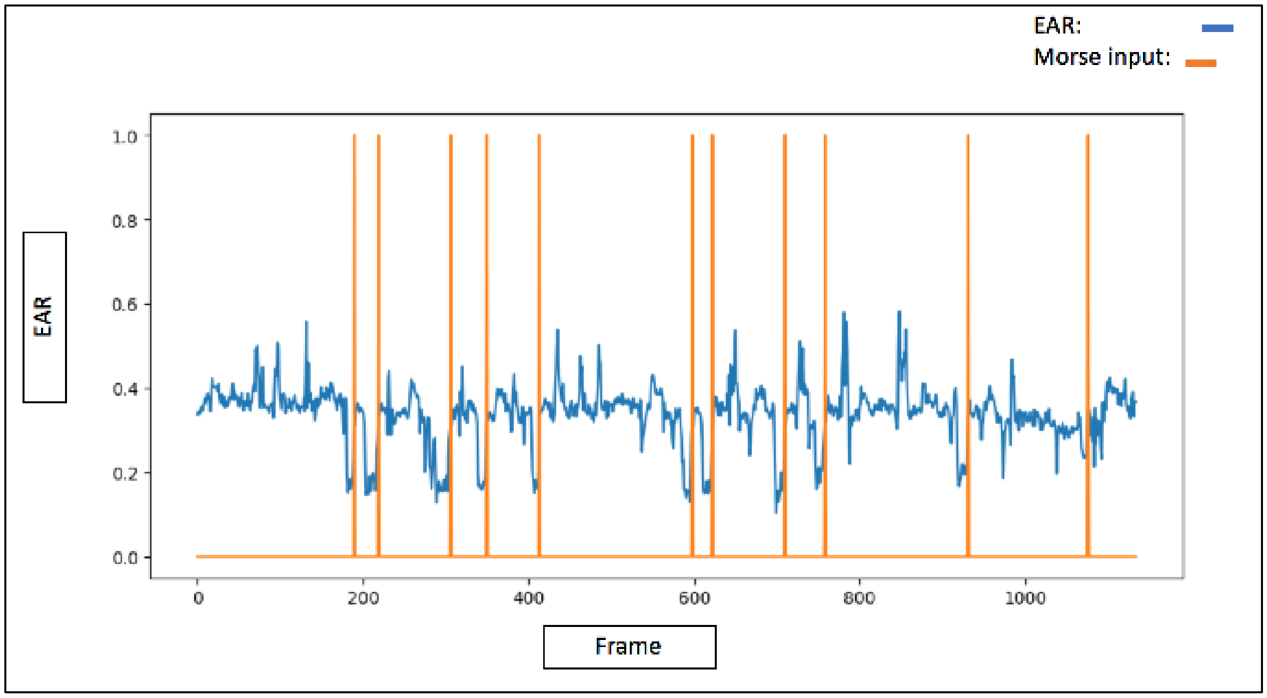


Figure 4.28: Eye aspect ratio graph

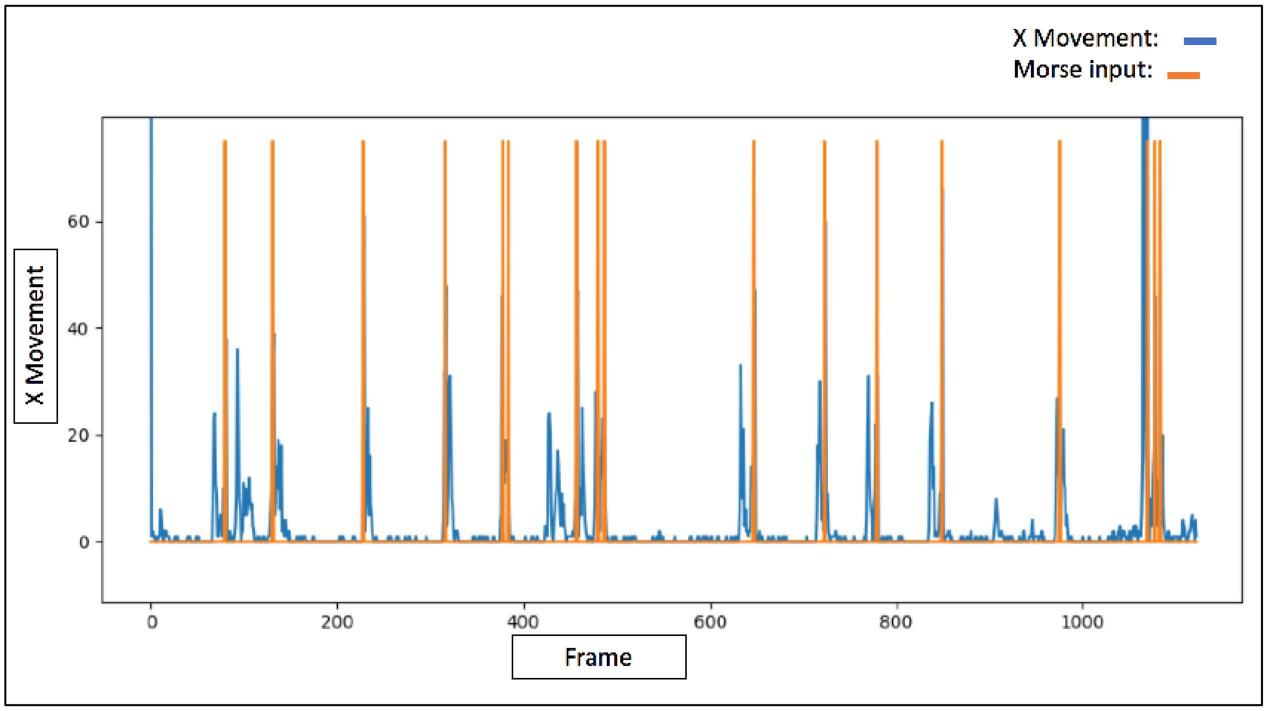


Figure 4.29: Head movement graph

TABLE 4-4: Success and failure rate for each tester

Tester	Detector	Number of tries	Detected	False Detection	Success rate	False positive rate	Total error	Total Time
Tester 1	MAR	11	10	0	91%	0%	9%	92 sec
	EAR	10	10	0	100%	0%	0%	
	X-Movement	11	10	0	91%	0%	9%	
Tester 2	MAR	12	10	0	83%	0%	17%	96 sec
	EAR	10	10	0	100%	0%	0%	
	X-Movement	14	14	2	85%	16%	30%	
Tester 3	MAR	12	10	0	83%	0%	17%	53 sec
	EAR	11	10	0	91%	0%	9%	
	X-Movement	12	12	0	100%	0%	0%	
Tester 4	MAR	11	10	0	91%	0%	9%	87 sec
	EAR	10	10	0	100%	0%	0%	
	X-Movement	10	10	0	100%	0%	0%	
Tester 5	MAR	10	10	0	100%	0%	0%	90 sec
	EAR	10	10	0	100%	0%	0%	
	X-Movement	13	12	1	92%	8.3%	16%	
Tester 6	MAR	11	10	0	91%	0%	9%	85 sec
	EAR	10	10	0	100%	0%	0%	
	X-Movement	12	12	0	100%	0%	0%	

Table 4.3 shows the experiment results for each tester. Each student was given a trial time to get familiar with the system. Afterward, they were told to perform a certain gesture to match the Morse code accordingly. The average time to spell the phrase for people that do not know Morse code at all was 90 second. One of the tester, who was a bit more familiar with Morse code, took 53 second to complete the task.

The head movement detector was the only detector that gave false-positives. It would detect one head shake as two at time. The Mouth detector gave an average error rate of 10%. The eye-blink detector performed the best. It gave a 9% error rate for only one tester out of 6. Finally, the head movement detector gave an average error rate of 10%.

CHAPTER 5

CONCLUSIONS

The world of technology is ever evolving and ever changing. The need for new technologies will never stop as people find new problems to solve or make new discoveries. People with disabilities faces challenges every day, and developing new systems to aid them in fitting in is important.

In this thesis, we proposed a facial gesture detection system for Morse code communications. An eye-blink detector, an open mouth detector, and a head nod/shake detector are combined to permit communications via Morse code. In Chapter 4, we present a head nod detector using facial landmark. This method show better recognition results than the proposed haar-cascade method in [66] with fewer errors and false positive detections.

During the experiment, it was noticed that it took an average of 90 seconds to spell short phrases, such as “I WANT WATER” at the first try. This long period of time is expected for a first try as none of the test subjects were not familiar with Morse code. One of the testers was familiar with Morse code and took 53 seconds to spell the phrase. Constant training on the system would be required to achieve optimum results. Just like typing on a keyboard, typing speed will increase with constant usage of the system.

In future work, the system can be incorporated with an auto correction or auto prediction system to complete words and sentences that are being spelled out. The system is intended for people with severe muscle movement limitations. Errors are bound to occur especially on the first tests. An auto corrector and auto predictor would be able to decrease errors in sentences and decrease typing time considerably.

REFERENCES

[1] A. Nikolaidis and I. Pitas, “Facial feature extraction and pose determination,” Pattern Recogn., vol. 33, pp. 1783–1791, 2000.

[2] M. J. Lyons, J. Budynek, and S. Akamatsu, “Automatic classification of single facial images” IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, no. 12, pp. 1357–1362, Dec. 1999.

[3] M. Lades, J.C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R.P. Wurtz, and W. Konen, “Distortion Invariant Object Recognition in the Dynamic Link Architecture,” IEEE Trans. Computers, vol. 42, no. 3, pp. 300- 311, Mar. 1993.

[4] L. Wiskott, J. Fellous, N. Kruèger, and C. von der Malsburg, “Face Recognition and Gender Determination” Proc. Int'l Workshop Automatic Face and Gesture Recognition, M. Bichsel, ed., pp. 92-97, 1995.

[5] M. Turk and A. Pentland, “Eigenfaces for Recognition,” J. Cognitive Neuroscience, vol. 3, no. 1, pp. 71-86, 1991.

[6] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, “Eigenfaces vs. Fisherface: Recognition Using Class Specific Linear Projection,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 711-720, July 1997.

[7] K. Etemad and R. Chellappa, “Discriminant Analysis for Recognition of Human Face Images,” J. Optical Soc. Am. A, vol. 14, no. 8, pp. 1,724-1,733, 1997.

[8] A. Lanitis, C. J. Taylor, and T. F. Cootes, “Automatic interpretation and coding of face images using flexible models,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 7, pp. 743–756, Jul. 1997.

[9] M. Turk and A. Pentland, “Eigenfaces for recognition,” J. Cogn. Neurosci., vol. 3, pp. 71–86, 1991.

[10] D.J. Kriegman P.N. Belhumeur, J.P. Hespanha, “Eigenfaces vs. fisherfaces: Recognition using class specific linear projection,” IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1997), no. 7, 711–720.

[11] R. Brunelli T. Poggio "Face Recognition: Features vs. Templates" IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 15 no. 10 pp. 1 042-1 053 Oct. 1993.

[12] B.K.P. Horn Computer Vision. Cambridge MA: 1986.

[13] J. G. Daugman, “Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters,” J. Opt. Soc. Amer. A, vol. 2, pp. 1160–1169, 1985.

[14] G. Edwards, C. Taylor, and T. Cootes. “Interpreting face images using active appearance models. In Automatic Face and Gesture Recognition.” Proc. Third IEEE International Conference on, pp 300–305, 1998.

[15] J. G. Bueno, M. G. Fierro, and L. Moreno, “Facial Gesture Recognition using Active Appearance Models based on Neural Evolution,” LBR Highlights pp. 133-134 March 2012

[16] S. C. Mitchell, B. P. F. Lelieveldt, R. J. van der Geest, H. G. Bosch, J. H. C. Reiber, and M. Sonka, “Multistage Hybrid Active Appearance Model Matching: Segmentation of Left and Right Ventricles in Cardiac MR Images”, EEE Trans. Med. Imaging, no. 5, 2001.

[17] F. Dornaika and J. Ahlberg, "Fast and reliable active appearance model search for 3-D Face Tracking," IEEE Trans. Systems, Man, and Cybernetics, Part B, vol. 34, no. 4, 2004, pp. 1838-1853.

[18] C. Huang* and Y. Huang, “Facial Expression Recognition Using Model-Based Feature Extraction and Action Parameters Classification,” Journal of Visual Communication and Image Representation Vol. 8, No. 3, September, pp. 278–290, 1997.

[19] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, “Coding facial expressions with gabor wavelets,” in Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on, pp. 200–205, Apr 1998.

[20] S. Ouellet, “Real-time emotion recognition for gaming using deep convolutional network features.,” arXiv:1408.3750, 2014.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks.,” in NIPS, vol. 1, p.4, 2012.

[22] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, andI. Matthews, “The extended cohn-kanade dataset (ck+): A completedataset for action unit and emotion-specified expression,” in ComputerVision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on, pp. 94–101, IEEE, 2010.

[23] R. A. Khan, A. Meyer, H. Konik, and S. Bouakaz, “Human visioninspired framework for facial expressions recognition,” in Image Processing (ICIP), 2012 19th IEEE International Conference on, pp. 2593–2596, IEEE, 2012.

[24] G. Bradski, “The opencv library,” Dr. Dobbs Journal of Software Tools, 2000.

[25] P. Viola and M. 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, vol. 1, pp. I–511, IEEE, 2001.

[26] G. N. Matre, S. K. Shah, "Facial Expression Detection", in Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on, pp. 1-3, IEEE, Dec 2013.

[27] L. Sirovich and M. Kirby, "Low Dimensional Procedure for Characterization of Human Faces" J Optical Soc. Am., vol. 4, pp. 519-524, 1987.

[28] Athitsos et al., "The American Sign Language Lexicon Video Dataset.", IEEE Workshop on Computer Vision and Pattern Recognition for Human Communicative Behavior Analysis, IEEE, June 2008

[29] Purdue ASL Database for automatic recognition of American Sign Language [Internet]. Available from: http://www2.ece.ohio-state.edu/~aleix/ASLdatabase.htm [Accessed 10/13/2017]

[30] S. Nash, M. Rhodes, J. I. Olszewska. iFR: Interactively-pose-corrected face recognition. In: Proceedings of the INSTICC International Conference on Bio-Inspired Systems and Signal Processing (BIOSIGNALS). 2016. pp. 106–112.

[31] R. Singh, M. Vatsa, A. Noore. Recognizing face images with disguise variations. In: Delac K., Grgic M., Bartlett M. S.; editors. Recent Advances in Face Recognition. Vienna: InTech; 2008. pp. 149–160.

[32] S. Shan, W. Gao., B. Cao, D. Zhao. Illumination normalization for robust face recognition against varying lighting conditions. In: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition (AFGR). 2003. pp. 157–164.

[33] Yale Face Database. Yale University. Connecticut, USA [Internet]. 2001. Available from: http://vision.ucsd.edu/yale_face_dataset_original/yalefaces.zip [Accessed: 2016-07-07]

[34] Caltech Faces. A Public Dataset for Face Recognition. CalTech University. USA [Internet]. 1999. Available from: http://www.vision.caltech.edu/html-files/archive.html [Accessed: 2016-07-07]

[35] Caltech 10000 Web Faces. Human Faces Collected from Google Image Search. CalTech University. Pasadena, California, USA [Internet]. 2005. Available from: www.vision.caltech.edu/Image_Datasets/Caltect_10K_WebFaces [Accessed: 2016-07-07]

[36] S. Bianco, "Large Age-Gap Face Verification by Feature Injection in Deep Networks" In Pattern Recognition Letters, volume 90, pp. 36-42, 2017.

[37] ORL. Face Database. AT&T Laboratories Cambridge [Internet]. 1994. Available from: www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html [Accessed: 2016-07-07]

[38] Jesorsky O., Kirchberg K., Frischholz R. Face detection using the Hausdorff distance. In Bigun J., Smeraldi F., editors. Audio and Video based Person Authentication. LNCS Springer. 2001. pp. 90–95.

[39] Caltech Faces. A Public Dataset for Face Recognition. CalTech University. USA [Internet]. 1999. Available from: http://www.vision.caltech.edu/html-files/archive.html [Accessed: 2016-07-07]

[40] FDDB. A Benchmark for Face Detection in Unconstrained Settings. Jain V., Learned-Miller E. Technical Report. UM-CS-2010-009. University of Massachusetts, Amherst. USA [Internet]. 2010. Available from: http://vis-www.cs.umass.edu/fddb/ [Accessed: 2016-07-07]

[41] Learned-Miller E., Huang G.B., Roy-Chowdhury A., Li H., Hua G. "Labeled Faces in the Wild: A Survey. In: Advances in Face Detection and Facial Image Analysis. Springer. 2016. pp. 189–248.

[42] L. Liu, C. Xiong, H. Zhang, Z. Niu, M. Wang, S. Yan, "Deep aging face verification with large gaps", IEEE Trans. Multimedia, vol. 18, no. 1, pp. 64-75, Jan. 2016.

[43] Ahonen T, Hadid A, Pietika¨inen M,_"Face recognition with local binary patterns". In Proceeding of European conference on computer vision (ECCV2004), LNCS 3021, pp 469–481,2004

[44] V. Gupta, D. Sharma, "A Study of various Face Detection Methods" International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 5, May 2014

[45] M. Lyons ; S. Akamatsu ; M. Kamachi ; J. Gyoba, "Coding facial expressions with Gabor wavelets" Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference, August 2002

[46] P.Ekman and W.V.Friesen, Facial Action Coding System (FACS): Manual, Palo Alto: Consulting Psychologists Press, 1978.

[47] M. Betke, J. Gips and P. Fleming, "The Camera Mouse: visual tracking of body features to provide computer access for people with severe disabilities," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 10, no. 1, pp. 1-10, March 2002.

[48] O. Takami, N. Irie, C. Kang, T. Ishimatsu, and T. Ochiai, "Computer interface to use head movement for handicapped people," in Proc. IEEE TENCON'96, Digital Signal Processing Applications, vol. 1, 1996, pp. 468–472.

[49] D. G. Evans, R. Drew, and P. Blenkhorn, “Controlling mouse pointer position using an infrared head-operated joystick,” IEEE Trans. Rehab. Eng., vol. 8, no. 1, pp. 107–117, 2000.

[50] Y. L. Chen, F. T. Tang, W. H. Chang, M. K. Wong, Y. Y. Shih, and T. S. Kuo, “The new design of an infrared-controlled human-computer in- terface for the disabled,” IEEE Trans. Rehab. Eng., vol. 7, pp. 474–481, Dec. 1999.

[51] R. B. Reilly and M. J. O'Malley, "Adaptive noncontact gesture-based system for augmentative communication," in IEEE Transactions on Rehabilitation Engineering, vol. 7, no. 2, pp. 174-182, Jun 1999.

[52] K. Grauman, M. Betke, J. Gips and G. R. Bradski, "Communication via eye blinks - detection and duration analysis in real time," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 2001, pp. I-1010-I-1017 vol.1.

[53] R. Kothari, J. Mitchell, "Detection of eye locations in unconstrained visual images", IEEE Int. Conf. Image Processing, pp. 519-522, 1996

[54] M. Betke, W.J. Mullally, J. Magee, "Active detection of eye scleras in real time", IEEE Workshop on Human Modeling Analysis and Synthesis, 2000.

[55] S. Singh, N. Papanikolopoulos, Vision-based detection of driver fatigue, 2001.

[56] S. Singh, N. Papanikolopoulos, Vision-based detection of driver fatigue, 2001.

[57] A. Haro, M. Flickner, I. Essa, "Detecting and tracking eyes by using their physiological properties dynamics and appearance", CVPR, 2000.

[58] C.H. Morimoto, M. Flickner, "Real-time multiple face detection using active illumination", IEEE Face & Gesture Recognition, pp. 8-13, 2000.

[59] T. Peters (19 August 2004). "PEP 20 – The Zen of Python". Python Enhancement Proposals. Python Software Foundation. Retrieved 24 November 2008.

[60] DLib: Library for Machine Learning [Internet]. 2014. Available from: https://www.kdnuggets.com/2014/06/dlib-library-machine-learning.html [Accessed: 2017-10-15]

[61] F. L. Pope, “The American Inventors of the Telegraph, with Special References to the Services of Alfred Vail,” Century Illustrated Magazine, no 35 (avril 1888), p. 924–945

[62] Morse code [Internet]. Available from: https://en.wikipedia.org/wiki/Morse_code. [Accessed: 2017-10-18]

[63] Facial Point Annotation. Available from: https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/ [Accessed: 2017-10-10]

[64] T. Soukupov́ a, “Eye-Blink Detection Using Facial Landmarks”, CENTER FOR MACHINE PERCEPTION, p. 11, May 26, 2016.

[65] Eye Blink Detection with OpenCV, Python, and dlib [Internet]. Available from: https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/ [Accessed: 2017-09-01]

[66] Head Nod Detector [Internet]. Available from: https://gist.github.com/smeschke/e59a9f5a40f0b0ed73305d34695d916b [Accessed: 2017-09-02]