Lip Reading to Text

Mogammat Waleed Deaney

Project presented in partial fulfillment
of the requirements for the degree of
Bachelor of Science (Honours)
at the University of the Western Cape

Supervisor: Prof. IM Venter
Co-supervisor: Mr M Ghaziasgar
Mentor: Kenzo Abrahams
Co-mentor: Nathan de la Cruz

28 March 2014

Declaration

I, Mogammat Waleed Deaney, declare that this project "*Lip Reading to Text*" is my own work, that it has not been submitted before for any degree or assessment at any other university, and that all the sources I have used or quoted have been indicated and acknowledged by means of complete references.

Signature: ..... Date: .....

Mogammat Waleed Deaney.

Abstract

Lipreading also known as speechreading, is a visual way of "listening" to someone. This is done by looking at the speakers face to follow their speech patterns in order to recognise what is being said. This project will attempt to develop a computerized lip-reader. The system will observe the user's lip-movement through a sequence of frames and convert the recognised sound/word to text which will be displayed on a screen. The project will regard the mouth as the region of interest for lip reading. All other features such as movement of the tongue, hand and facial gestures is considered to be beyond the scope of the project. The subject will be asked to articulate certain sounds or letters which need to be correctly identified by the system.

Key words

Visual Speech Recognition

Lip Reading

Local Binary Patterns

Speech Reading

Support Vector Machine

Haar-Like Features

Lip Segmentation

Acknowledgments

I would like to thank my supervisors Prof. IM Venter and Mr M Ghaziasgar for their support and encouragement during my Honours year. Without our weekly meetings, this work would not have been possible. Without their help I would certainly not be where I am today.

Contents

List of Tables

List of Figures

Glossary

LBP	Local Binary Patterns
GUI	Graphical User Interface
SVM	Support Vector Machine
ROI	Region of Interest

Chapter 1

Background

Speech recognition is not purely auditory. A speaker produces visual information which can be used by the listener to interpret what the speaker is saying. This is considered to be part of the speech recognition process. The first reports of visual information being used for speech recognition were contributed by McGurck and MacDonald [McGurk and MacDonald, 1976] [Mark Barnard and Owens, 2010].

This project is an attempt to recognise lip movement as speech and display the result as text. It entails developing an application that uses a web camera, which will be pointed at a person's face, to lip read that person. This will involve recognizing simple vowels or sounds at first. It can later be extended to words and sentences, time permitting. The application will then take the recognized vowel, word or sentence and convert it to text to be displayed on the screen.

Chapter 2

User Requirements

2.1 Introduction

This section describes the problem from the user's point of view. It states the problem domain and the functionality of the program.

2.2 Users view of problem

Lip reading is the process of understanding speech by visually interpreting movement of the mouth. The user needs a system that will computerize the process of lip reading.

2.3 Description of the problem

A system is required that can automatically lip read letters, sounds or words without input from the user. The objective is to create an automated lip reader that will determine what the subject has said and display it on the screen.

Speech recognition translates spoken words into text. It is difficult for the software to identify a particular speaker when another person is speaking at the same time. Another disadvantage is that it does not work well in noisy environments. The sound of particular words can sometimes be similar for e.g "one" and "won". The words can be mistaken for one another as they sound the same but have distinctly different meaning and lip movements. The "lip reader" will address the disadvantages of speech recognition.

2.4 Expectations of the software solution

The software should be able to detect the subject's mouth-movements using a web camera and be able to recognise simple sounds or letters articulated

by the subject. The program should display the recognised sound or letter as text to the user. It should also allow the user to stop and start the program at any given time.

2.5 Not expected of the software solution

The software will not track multiple subjects and is not expected to monitor subjects who are not directly facing the camera. The subject being lip read is required to be as still as possible. The camera should have a clear uninhibited view of the user's face. The lip reader application will not do real-time lip reading.

2.6 Conclusion

The above section describes the user requirements of the system that will allow speech recognition visually. It also describes the scope of the project. In Chapter 3 the designers interpretation of the problem will identify the most beneficial means for the system's implementation.

Chapter 3

Requirements analysis design

3.1 Introduction

In the previous chapter, the user specified the need for a visual speech recognition system. This chapter will explain how the designer will deal with the user's requirements. This is done by breaking down the problem into high level constituent parts and identifying all the relevant details.

3.2 Interpretation and breakdown of the problem

The input of the application will be a live video stream of the subject speaking. The subject will have to face the camera directly and not move his/her head around while the program is running. When the subject speaks he/she will have to articulate well. The web-cam will capture the change of mouth movements when the subject speaks.

In order to accurately identify the change in mouth movements, preprocessing of the the video snippet will have to be done. Some of the preprocessing will be done with OpenCV. OpenCV is an open source computer vision and machine learning software library which allows for a simple-to-use computer vision infrastructure which has over 500 different functions available [Bradski and Kaehler, 2008].

3.3 Complete analysis of the problem

3.3.1 Face and mouth detection

The application will accept video input from a webcam frame by frame. For this project, the mouth is considered to be the region of interest. In order to detect the mouth, the subjects face needs to be located. This is done by implementing OpenCV's face detection library. It uses the Haar-classifier, a tree-

based technique used to detect rigid objects [Bradski and Kaehler, 2008]. The core basis for Haar-classifier object detection is the Haar-like features [Wilson and Fernandez, 2006]. Haar-like features are a contrast in values across a rectangular area of pixels showing both light and dark areas. Both the face and mouth have Haar-like features and therefore can be detected using this algorithm.(see Figure 3.1)

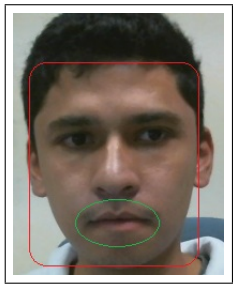

Figure 3.1: Locating the face and mouth

3.3.2 Pre-processing frames

Once the mouth is detected it is segmented from the rest of the frame. The image of the mouth is then grey-scaled in preparation for the next operation. Next the Local Binary Pattern (LBP) operator will be applied on the image. LBP as an image operator is used to extract LBP histogram (LBPH) features for texture description [Yang and Ai, 2007]. This will show the valleys and ridges of the image and could be useful in extracting features from the mouth.

3.3.3 Categorizing frames

A Support Vector Machine (SVM) will be used to categorize the frames. The SVM will be fed images to train and test the system. The frames will be classified as a mouth movement which can be matched to speech. A basic SVM takes a set of input data and predicts for each given input which of the two possible classes forms the output, making it a non-probabilistic binary linear classifier. The decision is made by solving a linearly constrained quadratic programming problem [Osuna et al., 1997].

3.3.4 Displaying the speech as text

The system will allow the user to see what the subject has said on the screen. The result will be displayed as soon as the user finishes the sound/letter.

3.4 Other similar systems

People inherit the process of lip reading subconsciously by interacting with others from a young age, although a skilled lip reader cannot be a 100% accurate as lip reading is not a precise art.

H. Mehrotra, G. Agrawal and M.C. Srivastava created a lip reading system which proposed a solution based on automatic lip contour tracking. Their aim was to recognize characters of the English language after a lip contour is accurately detected on each frame. The system made use of the k-nearest neighbour machine learning technique and was trained on five different speakers with five characters each A,E,I,O and U. Results varied between a 33-73% accuracy proportional to the value of k. [Mehrotra et al., 2009]

3.5 Suggested solution

The solution will detect mouth movement and should be able to recognise simple letters. The solution could later be extended to capture words and sentences leading to a real-time lip reader. The tools needed for the implementation are a web camera and a computer with OpenCV installed.

3.6 Conclusion

This section states how the designer interpreted the problem. A generic framework for a solution was given in order to solve how the “lip reader” would proceed in visually recognising speech. The next chapter will describe how the results will be displayed to the user.

Chapter 4

User interface specification

4.1 Introduction

This chapter will describe the user interface and will explain how the user interacts with the program. This includes displaying how the user can manipulate the system and in turn display the effects of the manipulation.

4.2 Description of the user interface

The application will have a rudimentary Graphical User Interface (GUI), since the application is a prototype. The interface will be minimal, yet simple and intuitive to use. A window containing the video stream of the subject will be created and displayed to the user.

4.2.1 Input video window

The system will start with a video feed of the subject which is displayed inside a window. The window will open only when the system detects the subject's mouth.(See Figure 4.1)

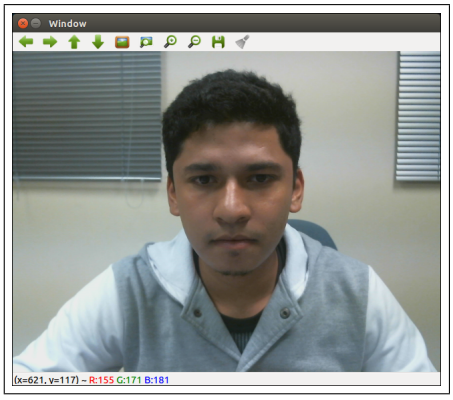

Figure 4.1: Window displaying subject

4.3 How the user interface behaves

The user will interact with the system with mouse clicks, each click will create a new window displaying the next stage of the application. The user will click on the input video window once the subject is ready to speak starting the process as shown in Figure 4.2. Two new windows will subsequently open, one displaying the subjects mouth and the other an image of the local binary patterns. Once the computations are complete a new window will open displaying the result to the user.

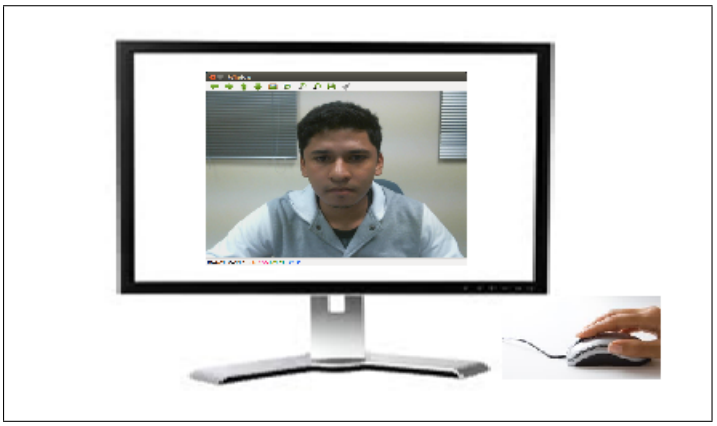

Figure 4.2: User interaction

4.4 Conclusion

This chapter discusses how the user interacts with the system. The user interface is minimal and uses windows to display the output. The aim of the project is to provide results for research purposes, thus less focus is placed on the interface. The next chapter will discuss the high level design of the software.

Chapter 5

High level design

5.1 Introduction

This chapter describes the High Level Design (HLD) of the software solution. This perspective shows a high level abstraction of the automated lip reader to analyse the methodology in the construction of the system. The programming language of choice is C/C++.

5.2 Breakdown of technical solution in subsystems

The technical solution consists of four phases which include input from the video camera, pre-processing, classification and the recognition of the letter/sound (output). These phases form the technical solution of the automated lip reader system. A breakdown of these phases are necessary to form high level constituent parts of the system. Figure 5.1 illustrates the breakdown of the technical solution into subsystems namely, video frames, segmentation of the lip region (ROI), feature extraction, support vector machine, recognition of the letter/sound.

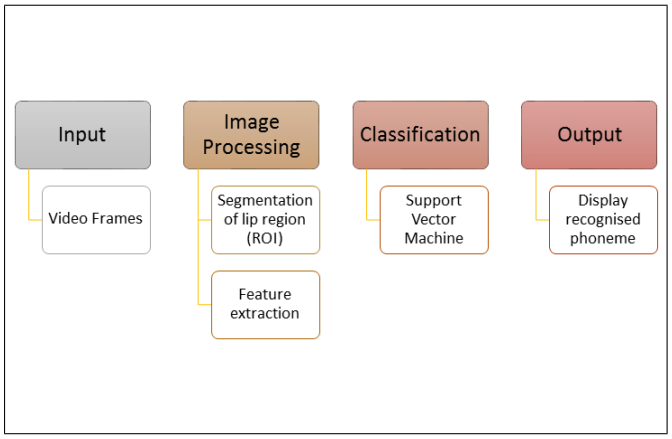

Figure 5.1: Breakdown of technical soution to HLD

5.3 Description of subsystems and operations

Table 5.1 represents important components including and relating to the subsystems and their operations.

Table 5.1: Description of

COMPONENT	DESCRIPTION
OpenCV	OpenCV is an open source computer vision library. It provides functions and data structures that allow for the processing of images[Bradski and Kaehler, 2008].
Haar-like Features	Features used in object detection. The idea was extended by Viola and Jones. It is capable of processing images extremely rapidly and achieving high detection rates[Paul Viola, 2001].
Region of Interest (ROI)	A rectangular area which is segmented from the original image to form the ROI. The ROI is isolated from the original image to be further processed upon. The system regards the mouth area of the subject as the ROI.
Feature Extraction	A manipulation of an image to represent interesting parts as a compact feature vector. An example of feature extraction is the Local Binary Pattern operator.
Local Binary Pattern (LBP)	The local binary pattern is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighbourhood of each pixel and considers the result as a binary number[Matti Pietikinen, 2011].
Histogram	A representation of the colour distribution within an image. This representation is sent to the SVM for classification.
Classification	A process in which data is grouped according to specific parameters. The process is completed with the use of a SVM.
Support Vector Machine (SVM)	A SVM is used to recognize patterns associated with the intensity of pixels. Data is sent to the SVM and is used to classify which class the image pixels belong. The extracted histogram data is sent to the SVM for training and testing the system.

5.4 Interaction between subsystems

Figure 5.2 represents the relationship between subsystems. Each subsequent process can begin only if its predecessor is complete. The interaction begins

once the user sits in front of the camera. This will provide input frames for the system. Once the subject has been detected and is ready to speak, the system will start. The lips are segmented from the frames providing a ROI and the features are extracted. The extracted features are sent to the SVM for classification. The final step involves displaying the result to the user as a sound/letter.

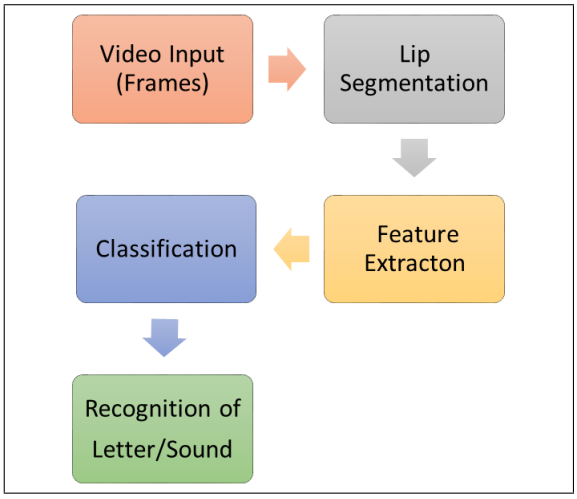

Figure 5.2: Relationship between subsystems

5.5 Conclusion

This chapter illustrates the high level design of the visual speech recognition system. It describes the concepts and provides a visual representation of the system giving a general idea of how the system should behave. The subsystems will be discussed in further detail in the next chapter which will provide the low level design.

Chapter 6

Low level design

6.1 Introduction

In this chapter the Low Level Design (LLD) of the software is described. The LLD will breakdown the HLD into smaller constituent parts allowing for more detail when describing the system.

6.2 Breakdown of high level design

Figure 6.1 illustrates the breakdown of the HLD into the necessary elements required to complete the subsystem.

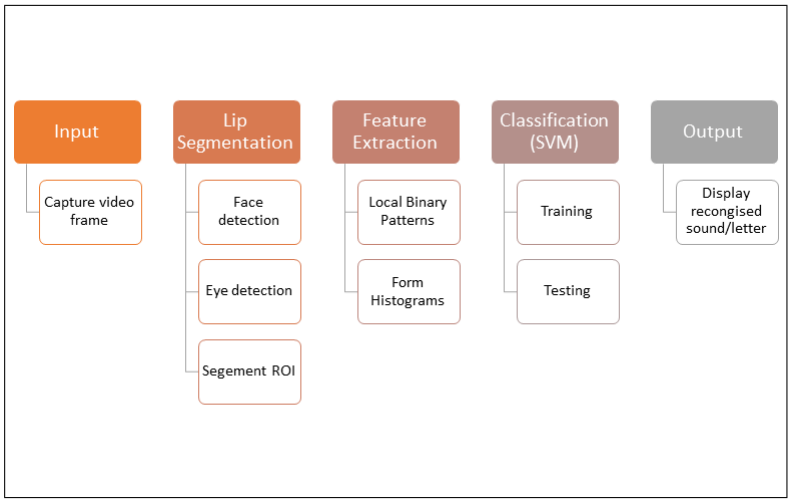

Figure 6.1: Breakdown of HLD to LLD

6.2.1 Capture video frames

Figure 4.1 depicts how video frames are displayed to the user. The method cvCaptureFromCAM() provides the computer access to the web camera and displays the image frame by frame forming the video.

6.2.2 Face and eye detection

For rapid object detection the system will implement the Viola and Jones object classifier. This component requires a greyscaled frame as input and returns the vector of the object specified. The classifier uses haar-like features to detect the face. Haar-like features represent different intensities of greyscale between two or more adjacent rectangles. The face is detected using face_ cascade.detectMultiScale(). As a result the face is isolated from the background (See Figure 6.2).

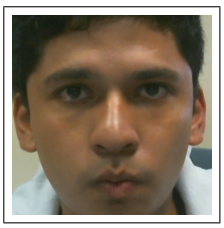

Figure 6.2: Isolated face

The eyes are detected using the eye_ cascade.detectMultiScale(). This is used on the isolated image of the face in order to decrease computing and accurately detect the eyes (See Figure 6.3).

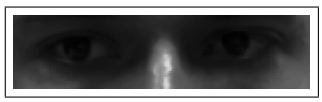

Figure 6.3: Isolated eyes

6.2.3 Segmentation of ROI

The distance between the eyes are used to segment the mouth. It provides the appropriate width and an accurate position along the x-axis for the lips to be segmented. An approximation of the distance is calculated by dividing the eyes into four equal quadrants and using the bounds to locate the lips. The left bound of quadrant two and the right bound of quadrant three provide the distance and x-positions for the lips. The bottom third of the face provides the appropriate y-position of the face. The bounded regions are illustrated in Figure 6.4.

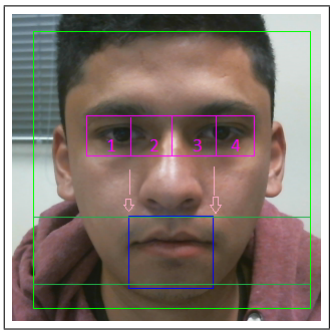

Figure 6.4: Boundries used to isolate the mouth

6.2.4 Local binary patterns

The local binary pattern is an image operator which transforms an image into an array or image of integer labels describing small-scale appearance of the image. The basic LBP operator works in a 3x3 pixel block of an image. The pixels in the block are thresholded by its center pixel value, multiplied by powers of two and then summed to obtain a label for the center pixel[Matti Pietikinen, 2011]. Figure 6.5 displays the LBP procedure pictorially.

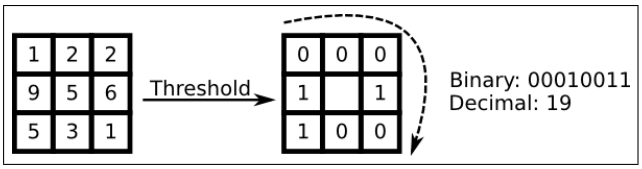

Figure 6.5: LBP calculation from binary to decimal

The features can be extended by changing the neighbourhood values of the LBP where P is the number of sampling points and R the radius of a circle around the center pixel. Figure 6.6 illustrates the different $LBP_{P,R}$ neighbourhoods.

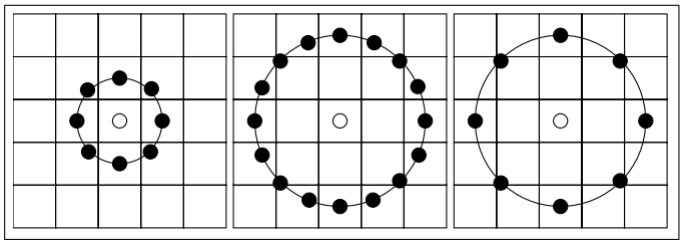

Figure 6.6: Representations of $LBP_{P,R}$

The leftmost image represents a $LBP_{8,1}$ neighbourhood , the center im-

age a $LBP_{16,2}$ neighbourhood and the rightmost image a $LBP_{8,2}$ neighbourhood. The $LBP_{8,2}$ will be applied to the greyscaled image of the mouth resulting in the grey texture image which will be used to form histograms (See Figure 6.7).

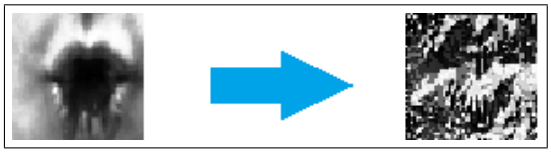

Figure 6.7: Transformation of segmented mouth from Greyscale to LBP

6.2.5 Form histograms

The LBP texture features produced by the mouth are used to form histograms. It produces a representation of the different intensities of the greyscale values associated with the the LBP image. The bin size is relative to the value of the neighbours in the LBP operator. The $LBP_{8,1}$ consists of 8 neighbourhood pixels which result in a total of $2^8 = 256$ different labels (See Figure 6.8). The

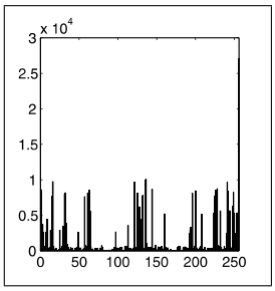

Figure 6.8: Histogram representation of the LBP image

histogram is calculated by dividing the LBP image in smaller quadrants called window sizes. A histogram is calculated for each window. The histograms are then concatenated to form the feature histogram which is sent to the SVM for further processing.

6.2.6 Training and testing

This phase includes training the system to recognise specific sounds/letters using the histograms produced from the LBP image. Each sound/letter is given a unique label representing which sound/letter was spoken. Different

subjects will be used to train the system on specific sounds/letters. The resulting data is then grouped together.

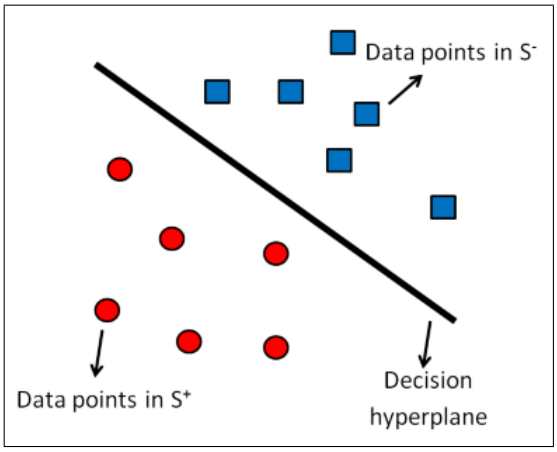

Figure 6.9: Data separated by a linear decision hyperplane

Once the training phase is complete, a decision boundary is calculated. The boundary will separate various spoken sounds/letters. Figure 6.9 represents a decision hyperplane which separates the groups of data. Once training is complete, it is tested with a subject that the system has not seen before. The resultant LBP histogram of the subject is sent to the SVM and it computes to which group it belongs. The result is a label which holds information to which group of sound/letter the image belongs to. The accuracy of the system is the critical. It is calculated by how many letters/sounds it can recognise correctly with using different subjects.

6.2.7 Displaying recognised sound/letter

The label is converted back to the recognised sound/letter and displayed to the user as text. The system will display what it thinks the sound/letter is. The success of the system is determined by its accuracy.

6.3 Conclusion

This chapter describes and details all the relevant information needed to complete the system. It describes the system in an algorithmic fashion by detailing each step required to complete each constituent part of the system.

Bibliography

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV*. O'Reilly Media, first edition.

Mark Barnard, E.-J. H. and Owens, R. (2010). Lip tracking using pattern matching snakes. In *Asian Conference on Computer Vision - ACCV*.

Matti Pietikinen, Guoying Zhao, A. H. T. A. (2011). *Computer Vison Using Local Binary Patterns*, volume 40. Springer.

McGurk, H. and MacDonald, J. (1976). Hearing lips and seeing voices. *Nature*, 264:746–748.

Mehrotra, H., Agrawal, G., and Srivastava, M. (2009). Automatic lip contour tracking and visual character recognition for computerized lip reading. *International Journal of Computer Science*, 3(4).

Osuna, E., Freund, R., and Girosi, F. (1997). Training support vector machines: an application to face detection. In *Computer Vision and Pattern Recognition, Proceedings., IEEE Computer Society Conference*, pages 130–136.

Paul Viola, M. J. (2001). 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*, volume 1, pages 511–518. IEEE.

Wilson, P. I. and Fernandez, D. J. (2006). Facial feature detection using haar classifiers. *Journal of Computing Sciences in Colleges*, 21(4):127–133.

Yang, Z. and Ai, H. (2007). Demographic classification with local binary patterns. In *ICB'07 Proceedings of the 2007 international conference on Advances in Biometrics*, pages 464–473.

Appendix A

Term Plan

GOAL	Due Date
Research • Learn how to use OpenCV	End of Term 1
• Accurately locate mouth and extract features	End of Term 2
Implementation • Train the system to recognize a sounds or letters • Optimize image for better recognition	End of Term 3
Test and Evaluate • Add more training and testing data	End of Term4

Figure A.1: Term Plan