

UNIVERSITI TEKNOLOGI MALAYSIA

BORANG PENGESAHAN  
LAPORAN AKHIR PENYELIDIKAN

TAJUK PROJEK : Development of Real-Time Embedded System  
with Speech Recognition for Smart House

Saya ZURAIMI BIN YAHYA  
(HURUF BESAR)

Mengaku membenarkan Laporan Akhir Penyelidikan ini disimpan di Perpustakaan Universiti Teknologi Malaysia dengan syarat-syarat kegunaan seperti berikut :

1. Laporan Akhir Penyelidikan ini adalah hakmilik Universiti Teknologi Malaysia.
2. Perpustakaan Universiti Teknologi Malaysia dibenarkan membuat salinan untuk tujuan rujukan sahaja.
3. Perpustakaan dibenarkan membuat penjualan salinan Laporan Akhir Penyelidikan ini bagi kategori TIDAK TERHAD.
4. \* Sila tandakan (/)

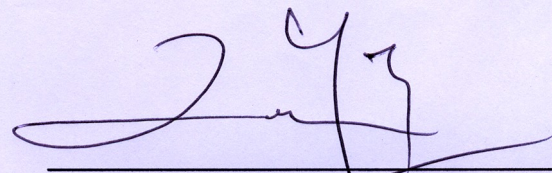
SULIT

(Mengandungi maklumat yang berdarjah keselamatan atau Kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972).

TERHAD

(Mengandungi maklumat TERHAD yang telah ditentukan oleh Organisasi/badan di mana penyelidikan dijalankan).

TIDAK  
TERHAD



TANDATANGAN KETUA PENYELIDIK

**Zuraimi Yahya**

Penasihat Akademik

Fakulti Kejuruteraan Elektrik

Universiti Teknologi Malaysia

81310 UTM Skudai

Nama & Cop Ketua Penyelidik

Tarikh : 23/06/08

CATATAN : \* Jika Laporan Akhir Penyelidikan ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali sebab dan tempoh laporan ini perlu dikelaskan sebagai SULIT dan TERHAD.

<b>Chapter</b>	<b>Title</b>	<b>Page</b>
	<b>List of Figures</b>	<b>III</b>
	<b>List of Tables</b>	<b>V</b>
<b>1.0</b>	<b>Chapter Overview</b>	<b>1</b>
<b>2.0</b>	<b>Introduction</b>	<b>3</b>
2.1	Review of Smart Home System via Voice Command	4
2.2	Overview of Malay Text to Speech	5
2.2.1	Overview of Text To Speech System	5
2.2.2	History of Synthesis Techniques	6
2.2.3	The Human Speech Production System and Speech Wave Generation Methods	8
2.2.3.1	The Human Speech Generation	8
2.2.3.2	The Source-Filter Theory of Speech Production	11
2.2.4	Synthesis techniques	12
2.2.4.1	Format Synthesis	12
2.2.4.2	Articulatory Synthesis	13
2.2.4.3	Concatenative Synthesis	14
2.2.5	Current Commercial and Non-commercial TTS System	18
<b>2.3</b>	<b>Overview of Audio Visual Speech Synthesis</b>	<b>21</b>
2.3.1	History and development of Talking Head	22
2.3.2	Previous work	23
2.3.3	Vismes	23
2.3.4	Visemes formed from phonemes of English	24
2.3.5	Methods to model a photo-realistic face	26
<b>2.4</b>	<b>Speech Recognition</b>	<b>27</b>
2.4.1	Isolated Word Recognition System	27
2.4.2	Feature Extraction	27

<b>Chapter</b>	<b>Title</b>	<b>Page</b>
<b>2.5</b>	<b>HMM models</b>	<b>28</b>
<b>2.6</b>	<b>HMM model training</b>	<b>28</b>
2.7	The HMM Recognizer	31
<b>3.0</b>	<b>Introduction</b>	<b>32</b>
<b>3.1</b>	<b>Malay Text to Speech</b>	<b>32</b>
	3.1.1 Malay TTS Smoothing Engine	32
	3.1.2 Pitch Detector and Marking Module	34
	3.1.3 Synchronize overlap-added method	36
	3.1.4 Result of speech units smoothing	36
<b>3.2</b>	<b>Malay Audio Visual Implementation</b>	<b>39</b>
	3.2.1 Expressions of Malay Talking Head	41
	3.2.2 Visemes groupings of Malay Language	41
	3.2.3 Image database	42
	3.2.4 Duration estimation	45
	3.2.5 Design Malay Talking Head and AV-TTS System	48
	3.2.6 Malay Talking Head module generation	53
<b>3.3</b>	<b>Experimental Evaluation</b>	<b>56</b>
	3.3.1 Task & Database	56
	3.3.2 Experimental Results for Multi speaker SD Task	56
	3.3.3 Experimental Result for SI Task	59
<b>3.4</b>	<b>Hardware design</b>	<b>59</b>
	3.4.1 Wireless Telemetry System	59
	3.4.1.1 Communications between the master and slave control terminal	62
	3.4.1.2 Master control terminal layout	63
	3.4.1.3 Slave or zone control terminal	64
	3.4.1.4 Maximum range test results	66

<b>Chapter</b>	<b>Title</b>	<b>Page</b>
3.4.2	Embedded Video Camera Movement Controller System	67
	3.4.2.1 Design Process	67
	3.4.2.2 Mechanical Design	68
	3.4.2.3 Microcontroller Circuit	69
	3.4.2.4 Program Flow Chart	69
3.4.3	USB Input Output Device	71
	3.4.3.1 Design Process	71
	3.4.3.2 Graphical User Interface (GUI)	73
3.4.4	Ultrasonic Water Level Detector	74
	3.4.4.1 Ultrasonic Water Level Measurement	74
	3.4.4.2 Program Flow Chart	76
	3.4.4.3 Measuring Process Test	77
<b>4.0</b>	<b>Conclusion</b>	<b>78</b>

## LIST OF FIGURES

Figures	Titles	Pages
Figure 1	Main architecture of smart house system	2
Figure 2.1	Simple text-to-speech synthesis procedure	6
Figure 2.2	The Human Speech Production System (Rabiner, 1993)	9
Figure 2.3	Cylindrical tube of varying cross sectional area to represent the vocal tract.	9
Figure 2.4	Parallel and Cascade Configuration of the Formants in Formants Synthesis Method(Lemmetty,1999)	13
Figure 2.5	Block diagrams of Residual-excited Linear Predictive (RELP)	17
Figure 2.6	Pitch modification of a voiced speech segment (Lemmetty, 1999)	18
Figure 2.7	The architecture of Talking Head Driven by Text	22
Figure 2.8	Previous Works of English Talking Head	23
Figure 2.9	Methods to model a photo-realistic face	26
Figure 2.10	Block diagram of MFCC front end	27
Figure 2.11	Feature extraction for DHMM & CDHMM	27
Figure 2.12	Shows the type of HMM. It is a 5 state left-to-right model	28
Figure 2.13	HMM training with BW algorithm	29
Figure 2.14	HMM training with Viterbi algorithm	31
Figure 3.1	MALAY TTS smoothing process	33
Figure 3.2	Pitch marking and overlap of two adjacent joint unit	33
Figure 3.3	The overlapping of two adjacent joint unit	33
Figure 3.4	An example of pitch marks on the short-time speech signal of the vowel /a/ for Malay Language.	34

## LIST OF FIGURES

Figures	Titles	Pages
Figure 3.5	Pitch-mark found on the voice regions of the waveform For the waveform for the word (Sembilan)	35
Figure 3.6	Frame blocking (Wei et al, 2006)	36
Figure 3.7	Concatenation articles for the word “jangan” in Malay	37
Figure 3.8	The artifacts of joint between two-adjacent unit	37
Figure 3.9	The result of concatenative speech through joint smoothing	38
Figure 3.10	output file for “jangan makan nasi.wav”	38
Figure 3.11	Block diagram of design of Malay Talking Head	40
Figure 3.12	Visemes groupings of Malay Talking Head	42
Figure 3.13	Surprise expression’s images for Malay Talking Head	44
Figure 3.14	Codes of MouthPhoneme structure definition	46
Figure 3.15	Codes to define array of Malay phoneme which comprises Structure {phoneme, duration, index of viseme}	46
Figure 3.16	Flow of the Malay Talking Head AV-TTS system design	49
Figure 3.17	Codes of the ‘Speak’ function	59
Figure 3.18	Part of codes of the ‘Sent2Pho’ function	51
Figure 3.19	Codes of the ‘Pho2Head’ function	52
Figure 3.20	Codes of the ‘OnTimer’ function	53
Figure 3.21	Codes of ‘Voice (CString Message)’ function	55
Figure 3.22	Example which imports the Malay Talking Head module	55
Figure 3.23	Project work flow	60
Figure 3.24	Communication between master and slave terminal	62
Figure 3.25	Internal functions of master control terminal	63
Figure 3.26	Block diagram of each zone	64
Figure 3.27	Flow diagram of slave or zone control terminal	65
Figure 3.28	The prototype board for this project	66

Figures	Titles	Pages
Figure 3.29	Project Block Diagram	67
Figure 3.30	Mechanical Design	68
Figure 3.31	Microcontroller Circuit	69
Figure 3.32	Control Program Flow Chart	70
Figure 3.33	Project Flow Chart	72
Figure 3.34	Graphical User Interface	73
Figure 3.35	Ultrasonic Water Level Measurement Model	75
Figure 3.36	Ultrasonic Water Level Detector Flow Chart	76
Figure 3.37	Measuring Process Flow	77

**LIST OF TABLES**

Tables	Titles	Pages
Table 2.1	The comparison between current commercial and non commercial TTS system.	20
Table 2.2	Viseme groupings of consonants I	25
Table 2.3	Viseme groupings of consonants II	25
Table 2.4	Viseme groupings of consonants III	25
Table 2.5	Viseme groupings of consonants IV	26
Table 3.1	Expressions of Malay Talking Head	41
Table 3.2	Viseme duration for every phoneme of Malay Talking Head	45
Table 3.3	DHMM and CDHMM recognizers with different training algorithms	56
Table 3.4	Comparison of DHMM and CDHMM recognizers with different training algorithm on SD task	58
Table 3.5	Comparison of DHMM and CDHMM recognizers for SI tasks	59



## Abstract

The hidden Markov model is used for the acoustic modeling of the speech recognition system. The Continuous Density HMM (CDHMM) which models acoustic observation directly using estimated continuous probability density function (pdf) without VQ, has shown to have higher recognition accuracy than DHMM. In this paper, CDHMM with Multivariate Gaussian Density is used for speaker dependent (SD) and speaker independent (SI) Malay isolated digit recognition and comparison is made with DHMM. The CDHMM was trained by different algorithms- Baum-Welch (BW), Viterbi (VB) (with segmental k-mean estimation) algorithm and combination of BW and VB then comparison is discussed. The training database consisted of 26 speakers with 5 utterances for each Malay digit. In SD task, another 5 utterances each digit from the same speakers are used for testing. Recognition accuracy of CDHMM with BW, VB training and combination of BW and VB is 99.00%, 98.85% and 98.69% respectively while 96.62% for DHMM. The accuracy for BW and Segmental K-Mean training is comparable, but the latter consumed less computational time. In SI task, 40 speakers, different from the training speakers, with each recorded 2 tokens are used for testing. The CDHMM achieves 85.63% accuracy and outperform DHMM with 8.17% improvement.

## Abstrak

Hidden Markov model digunakan untuk pemodelan akustik sistem pengenalan suara. Continuous Density HMM (CDHMM) yang memodelkan pemerhatian akustik secara langsung dengan menggunakan fungsi densiti kebarangkalian berterusan tanpa VQ, telah ditunjukkan mempunyai kejituan yang lebih tinggi daripada DHMM. Dalam kerja ini, CDHMM dengan densiti multivariate Gaussian digunakan untuk pengenalan digit Melayu terasing penutur-bergantung dan penutur-bebas dan perbandingan dibuat dengan DHMM. CDHMM dilatih dengan pelbagai algoritma- Baum-Welch (BW), algoritma Viterbi (VB) (dengan segmental k-mean estimation) dan pergabungan antara BW dan VB, perbandingan dibincangkan. Pangkalan data untuk perlatihan model terdiri daripada 26 penutur dengan 5 penuturan untuk setiap digit Melayu. Dalam tugas SD, 5 penuturan yang lain untuk setiap digit daripada penutur yang sama, digunakan untuk pengujian. Kejituan pengenalan oleh CDHMM dengan latihan BW, VB dan pergabungan antara BW dan VB ialah 99.00%, 98.85% dan 98.69% manakala 96.62% untuk DHMM. Kejituan oleh BW dan latihan segmental K-mean adalah setanding, tetapi segmental K-mean menggunakan masa pengiraan yang lebih kurang. Dalam tugas SI, 40 penutur, lain daripada penutur untuk latihan, merekodkan 2 tanda digit setiap orang, digunakan untuk pengujian. CDHMM mencapai kejituan 85.63% dan melebihi DHMM dengan pembaikan 8.17%.

## CHAPTER I

### INTRODUCTION

#### 1.0 Chapter Overview

Smart house command via voice is a special intelligent house maintenance and security system which utilizes the human physiology as the identity to access or control the device inside the house. The main engine behind it is the speech technologies; consist of speech recognition system, speaker recognition system and Malay text-to-speech system.

In this project, speech and speaker recognition utilize the Hidden Markov Model (HMM) technique for voice command and control of the device inside the house and security door system at the front door or main entrance. The voice pattern of the user is used to train the HMM model for word command and security door access identification. Thus, the system will only allow specific group of users to switch on or off the device and enter the house.

Besides that, the system also utilizes Malay text-to-speech system to response the user or alerts the user when any fault occurs. This feature makes the system more interactive and not only single direction of communication. Apart from that, the smart house also has some special monitoring devices such as camera, motion detector and glass break sensor installed at the house to monitor the house and prevent any illegal intruder.

All the devices are control by centre system using wireless system. If any suspected movement detected, the system will speak to the user using Malay text-to-speech system to alert the owner. Then the owner will give command to the system via voice to take any prevention steps. Keywords: Smart house, speaker recognition, speech recognition, text-to-speech

This is a project that integrated various Malay speech technologies such as Malay speech synthesizer, Malay Speech Recognizer and Talking Head with hardware controller for smart house system. Figure 1 shows the main architecture of smart house system utilizing speech technology. Malay Text to Speech (TTS) system, together with Talking Head serves as a system that provides the output synthetic voice that assists the user in using the system. Meanwhile, the speech recognition engine provides the capability for the system to recognize the input voice command and control the device via voice. With these two engines, user can communicate with the smart house system and controller the house appliance and monitor the house.

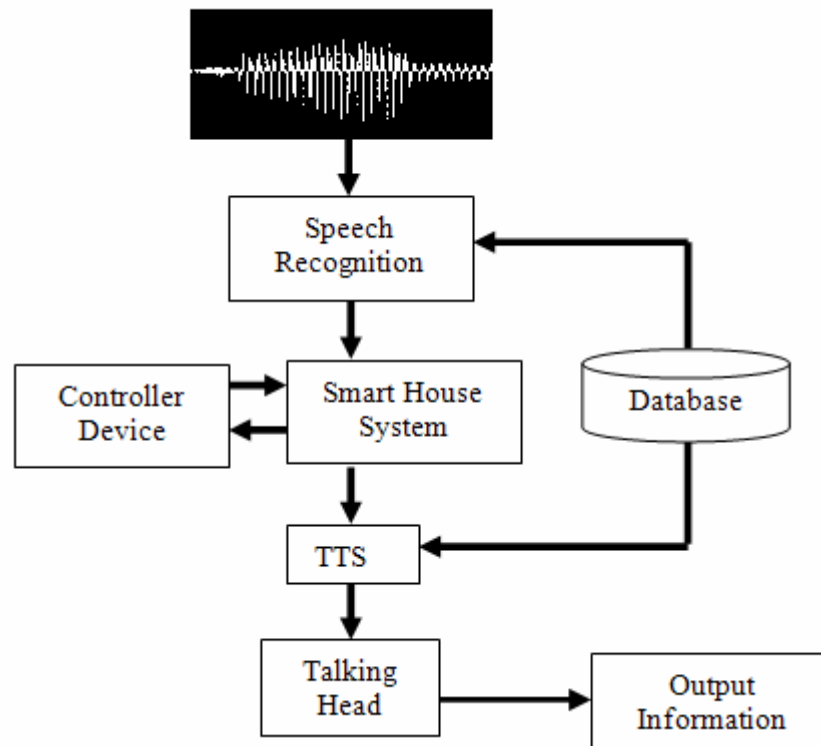


Figure 1: Main architecture of smart house system

## **CHAPTER 2**

### **LITERATURE REVIEW**

#### **2.0 Introduction**

Smart House concept comes from integration of microelectronics and telecommunications to support daily living. It is a broad spectrum of innovations combining fairly simple sensors, controller and futuristic automated homes. Smart House System brings a respect for autonomy and privacy, meeting real needs and acceptance and also correctly chosen functionalities well installed to raise confidence.

Smart House system in the market are the assemble of a few module-systems such as X10 for wireless communication, Home Automation System Integration for control center and remote control for system interface. The total cost of this available system is so expensive and not very intelligent. Users still need to use remote to control the system and set everything like lighting or cooling manually. For security purpose, users need to install another security system module, which will increase the total cost. Combination of all the modules will generate unpredictable reliability. With the advancement of computers and technologies, Smart House systems are being developed to overcome the problem. Current research is focused on the speech recognition and speaker verification for security purpose.

This research application is intended to develop a Smart House system using speech recognition for controlling the system and speaker verification for security. Hidden Markov Model will be use to develop algorithm for speech recognition and speaker verification. It is also intended to develop an intelligent knowledge based

system based on the method developed which is able to detect current environment resulting from common components in the system such as motion sensors, heat sensors, rain sensors, video surveillance and etc. This can be done through implementation artificial intelligent on the system. Experiments and tests have to be done on the test system to develop the knowledge-based systems. This system will be equipped with smoke detector alarm processing, fall detectors for elderly, exit door sensors, normal life pattern follow-up, various types of sensors, communicating with the real-time embedded system board and monitoring in real-time environment. The important property is this system is an easy-to-use two-direction communication facility such as monitoring and controlling using speech recognition, phone and also via internet on-line 24 hours a day. And most of all, it's about Security, Convenience, Energy Saving, and Cost Effectiveness.

Finally, this chapter will review the techniques and algorithm involve in developing smart home system. It will cover review of smart home system, review of Malay Text to Speech and Audio Visual, and Review of Speech Recognition system in Smart Home application.

## **2.1 Review of Smart Home System via Voice Command**

In this chapter will focus on the types of algorithm used in voice command for the smart house system. There are three main types of HMM so called discrete HMM (DHMM), continuous density HMM (CDHMM) and semi-continuous HMM (SCHMM). The CDHMM and SCHMM are still applied in current advanced recognition systems.

There are two type of HMM training algorithms that use maximum likelihood (ML) criterion as model parameter estimation objective: (1) Baum-Welch algorithm, (2) Viterbi/segmentation algorithm (Segmental k-means). Current Malay speech recognizer is based on DHMM. Hong and Sh-Hussain achieved 83.06% recognition accuracy for SI task (with 35 training speakers and 36 testing speakers) using DHMM. This paper proposes the application of CDHMM model in multi-speaker SD and SI isolated Malay digit recognition to reduce the quantization error caused by DHMM. Gaussian mixture density was used due to its effective modeling of inter-speaker acoustic variability. A similar system based on DHMM as baseline for

comparison. The CDHMM was also trained with different training algorithms (Baum-Welch, Viterbi, Baum-Welch with segmental k-mean as model bootstrapping). Results show that systems based on CDHMM achieve better recognition accuracy than the DHMM.

## **2.2 Overview of Malay Text to Speech**

This chapter will focus on the background, history, and current available TTS techniques in the hope of providing sufficient details and ideas for developing Malay TTS. Therefore it will concentrate more on current techniques in speech synthesis that will present the comparison of the advantages and disadvantages of each technique. Then, it will narrow down the discussion toward the synthesis method chosen for Malay TTS system. Ahead from that, it will direct the discussion toward the Festival Speech Synthesis system that has been used for implementing Malay TTS. Besides that, it will also briefly discuss the programming tools and speech processing tools that have been used for Malay TTS. Finally, it will justify why Festival Speech Synthesis (FSS) system has been selected for designing and developing the Malay TTS instead of start from the scratch of using other Speech Synthesis System.

### **2.2.1 Overview of Text To Speech System**

The text-to-speech (TTS) synthesis procedure consists of two main phases (Chris, 1991). The first phase of TTS is text analysis. It transcribed the input text into a phonetic or linguistic representation. The second phase of TTS is the generation of speech waveforms that the acoustic output is produced from this phonetic and prosodic information (Syrdal et al, 1994). The first phase is normally called as high level synthesis and the second phase is called low-level synthesis (Christof, 2002).

According to Lemmetty (1999), a simplified version of TTS system is illustrated in Figure 2.1. From Figure 2.1, the input text would be feed into the text

to linguistic analysis black box. The input text might be data from a word processor, scanned text from books and standard ASCII from e-mail (Thierry, 1993). The text to linguistic analysis block will analyze the input raw text to output the phonetic representation with intonation, duration and stress information (Geoff, 1984). Then the prosody and speech generation block will process the phonetic representation to generate the synthesis speech. There are various kinds of TTS system and a lot of TTS synthesis methods or techniques being explored since the start of introducing the TTS system at the year 1773 (Klatt, 1987). This would be discussed in detail in next sub-section.

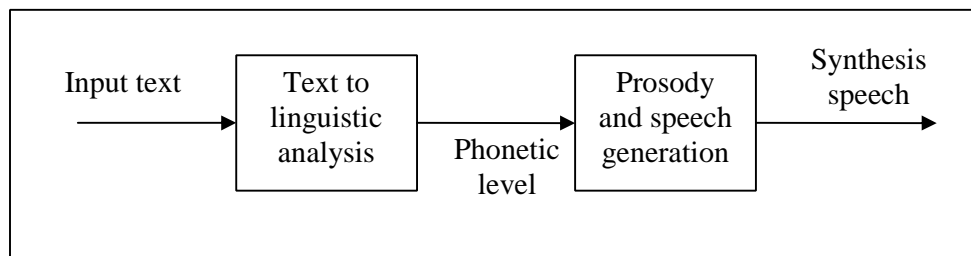


Figure 2.1: Simple text-to-speech synthesis procedure.

### 2.2.2 History of Synthesis Techniques

Actually Text To Speech system is not a new technology in human history. The research of TTS has been started more than 200 years ago since 1773 (Shuzo and Kazuo, 1985). The early stage of TTS system development was through the mechanism of machine (Lemmetty, 1999). Then it started to improve from machine to electrical device and finally using a computer as the tools for producing speech sound with high quality output (Syrdal et al, 1994). The most significant development or implementation of the techniques in TTS system is through the introduction of computer system as research tools since 1970 (Gold and Morgan, 2000).

The very first Text to Speech system has been developed in 1773, which produced human speech by machine. But the most famous speaking machine was produced in 1778 by Wolfgang von Kempelen (Shuzo and Kazuo, 1985). This “voice



organ” consisted of a resonance chamber, the shape of which was modulated with one hand to produce various vowels (Thierry, 1993). Consonants were produced with the other hand, by controlling the flow of air from the resonance chamber (Jan et al, 1996).

Then, at 1939 Homer Dudley brought the breakthrough in TTS system development through the idea of using an electrical device to produce the speech sound (Lemmetty, 1999). The first electrical device in TTS system was called VODER. Frank Cooper had improved this system in 1950 with his Pattern Playback system in electrical device (Klatt, 1987). The first formant synthesizer, PAT (Parametric Artificial Talker), was introduced by Walter Lawrence in 1953 PAT consisted of three electronic formant resonators connected in parallel (Gold and Morgan, 2000). The input signal was either a buzz or noise. A moving glass slide was used to convert painted patterns into six time functions to control the three formant frequencies, voicing amplitude, fundamental frequency, and noise amplitude (Klatt, 1987). The first Speech Synthesis by Computer has been started in 1970, and has grown tremendously since then (Lemmetty, 1999).

According to Lemmetty (1999) the synthesis methods can be subdivided into three groups:

- Articulatory synthesis, which attempts to model the human speech production system directly.
- Formant synthesis, which models the pole frequencies of speech signal or transfer function of vocal tract based on source-filter-model.
- Concatenative synthesis, which uses different length prerecorded samples derived from natural speech.

The most common used methods in present synthesis systems are formant and concatenative methods (Carlson et al, 2002). Formant synthesis was dominant for a long time, but today the concatenative method has become more and more popular (Jan et al, 1996). The articulatory method is still too complicated for high quality implementations, but may arise as a potential method in the future (Lemmetty, 1999).

### **2.2.3 The Human Speech Production System and Speech Wave Generation Methods**

This section will focus on the background of human speech production system and current speech wave generation methods in hope of providing some brief ideas on how the speech being generated and what methods involved in speech wave generation. It will start from the basic architecture of human speech production. Then it will discuss on how the speech production system is being model mathematically.

#### **2.2.3.1 The Human Speech Generation**

Speech can be described as a highly integrated and complex chain of events leading to a meaningful sequence of sounds (Chris, 1991). Figure 2.2 shows the human speech production system. Features such as height, weight, age and the structure of the vocal chords, nasal, oral cavities, teeth and lips play a major role in the speech production process (Rabiner, 1993).

The modification of the shape of human speech production system such as velum, the jaw, the tongue, and the lips will determine the transfer function of the vocal tract response to an excitation signal (Rabiner, 1993). This transfer function is usually composed of a number of resonances, known as formants, and occasionally of anti-resonances too (Gold and Morgan, 2000). Speech is produced by exciting the resonances and anti-resonances of the vocal tract filter (Parsons, 1987). The voice speech is from the vibration of the vocal cords while unvoiced speech is from turbulent noise created at a constriction somewhere in the vocal tract (Rabiner, 1993).

The vocal tract can be represented as cylindrical tube with varying cross sectional parameters as in Figure 2.3. We define  $f_n$  as the forward volume velocity

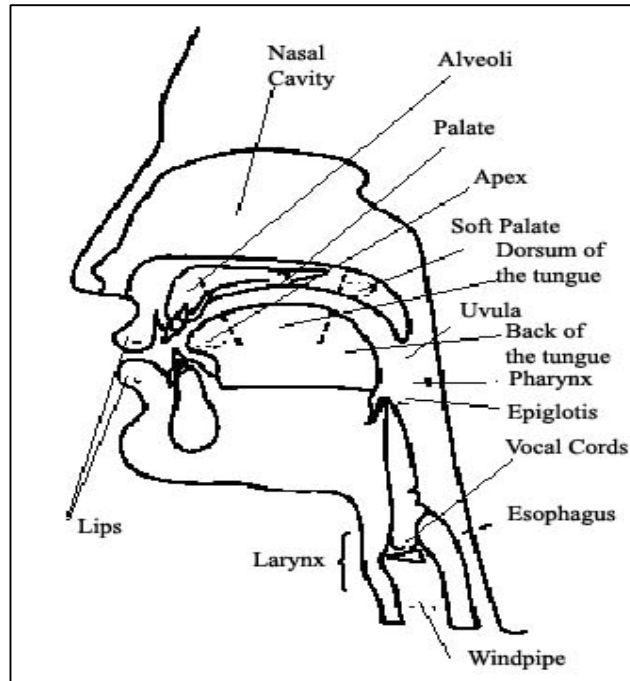


Figure 2.2: The Human Speech Production System (Rabiner, 1993).

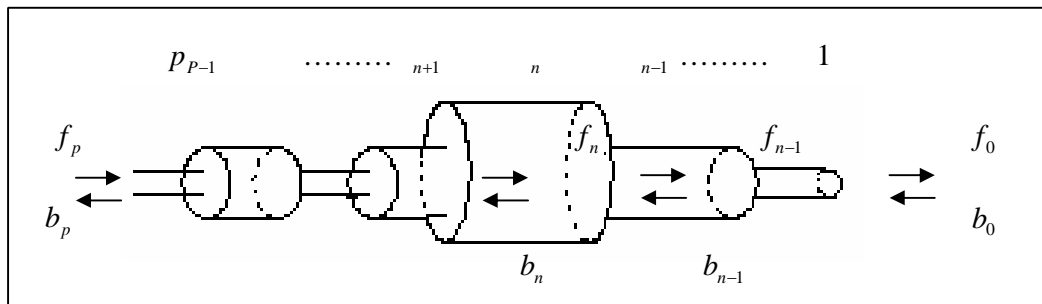


Figure 2.3: Cylindrical tube of varying cross sectional area to represent the vocal tract.

wave, which originates from border  $n-1$  (Parsons, 1987). Between the borders section of  $n$  and  $n-1$  on its left is the acoustics impedance  $R_n$  and is defined as  $\frac{P_0 c}{A_n}$ .

Where  $P_0$  is air density,  $c$  is the pressure waves, and  $A_n$  is the cross-sectional area.

By assuming that the pressure and the velocity are continuous across the junction and solving for  $f_{n-1}$  and  $b_{n-1}$  we get

$$\left| \begin{array}{c} f_{n-1} \\ b_{n-1} \end{array} \right| = \frac{1}{2R_{n-1}} \left| \begin{array}{c} R_n + R_{n-1}R_n - R_{n-1} \\ R_n - R_{n-1}R_n + R_{n-1} \end{array} \right| \left| \begin{array}{c} f_n \\ b_n \end{array} \right| \quad (2.1.1)$$

The reflection coefficient  $k$  is defined as

$$k = \frac{A_n - A_{n-1}}{A_n + A_{n-1}} \quad (2.1.2)$$

Multiplying equation by  $\frac{P_0 c}{A_n - A_{n-1}}$  for both numerator and denominator and simplifying it we have

$$k = \frac{R_{n-1} - R_n}{R_{n-1} + R_n} \quad (2.1.3)$$

If the system is sampled at a rate of  $F_s = \frac{c}{2L}$  then the transformation matrix for the entire code would be.

$$\left| \begin{array}{c} F_{(0)}z \\ B_{(0)}z \end{array} \right| = T \left| \begin{array}{c} F_p(z) \\ B_p(z) \end{array} \right| \quad T = \prod_{i=1}^p T_i \quad (2.1.4)$$

$$H(z) = \frac{G}{1 - \sum_{k=1}^p a_k z^{-k}} \quad (2.1.5)$$

The transfer function of this segmental lossless tube is an all pole model. The resonance frequencies of the models are the poles. According to Parsons (1987) the effect of losses to the vocal tract is minimize. The output of that system is (Geoff, 1984):

$$\sum_{k=1}^p a_k x(n-k) \quad (2.1.6)$$

If error between these two models is zero we can then say that the output  $x(n)$  of the first model is equal to the output of the model.

### 2.2.3.2 The Source-Filter Theory of Speech Production

The source-filter theory of speech production assumes that the excitation source can be considered to be independent from the vocal tract response. Thus, the vocal tract response usually assumed to be linear (Donovan, 1996). The z-transform of the speech signal,  $S(n)$ , can be synthesized as :

$$S(z) = U(z)H(z) \quad (2.2.1)$$

where  $U(z)$  is an approximation to the excitation signal, and  $H(z)$  the transfer function of a digital filter representing the vocal tract response and the radiation characteristic of the lips or nostrils (Rabiner, 1993).

In this equation,  $U(z)$  is often analyzed as  $P(z)G(z)$ , where  $P(z)$  is a pulse train and (or) white noise, and  $G(z)$  (which is only present for voice speech) is the transfer function of the glottal waveform “filter”.

$$U(z) = P(z)G(z) \quad (2.2.2)$$

While  $H(z)$  is often analyzed as  $V(z)R(z)$ , where  $V(z)$  is the transfer function of the vocal tract, and  $R(z)$  the radiation characteristic (Luis, 1997).

$$H(z) = V(z)R(z) \quad (2.2.3)$$

By replacing these two equations (equation 2.2.2 and 2.2.3, the equation for voice speech equation 2.2.1 becomes,

$$S(z) = P(z)G(z)V(z)R(z) \quad (2.2.4)$$

In linear prediction (LP) synthesis, the excitation signal used is often just  $P(z)$ . The LP's coefficients model the combined effects of the shape of the glottal waveform, the vocal tract response, and the radiation characteristic.

## 2.2.4 Synthesis techniques

There are three groups of low level synthesis methods such as formant synthesis, articulatory synthesis and concatenative synthesis methods. This section will compare these three methods to give some brief ideas on choosing the low level synthesis method for Malay TTS system.

### 2.2.4.1 Formant Synthesis

Formant synthesis is a source-filter method of speech production (Parsons, 1987). In formant synthesis, vocal tract filter is constructed from a number of resonances similar to the formants of natural speech (Rabiner et al, 1971). Three formants are required to synthesis intelligible speech for formant synthesis (Rabiner, 1993). Four or five formants are being sufficient to produce high quality speech (Rabiner et al, 1971). Formant is modeled using a two pole resonator that enables both the formant frequency and bandwidth to be specified (Klatt, 1987). According to Luis (1997), there are two methods of combining the formants to make a model of the vocal tract such as:

- Parallel configuration, which the resonators are connected in parallel. Each resonator is preceded by an amplitude control that determines the relative amplitude of a spectral peak for both voiced and voiceless sounds.
- Cascade configuration, which the voiced sounds are synthesized using a set of cascade resonators. The output of a resonator is fed into the input of the next. Amplitudes are automatically corrected as a consequence of cascade function.

Figure 2.4 shows the parallel and cascade configuration of the formants in formants synthesis method. Further details will be discussed in the thesis.

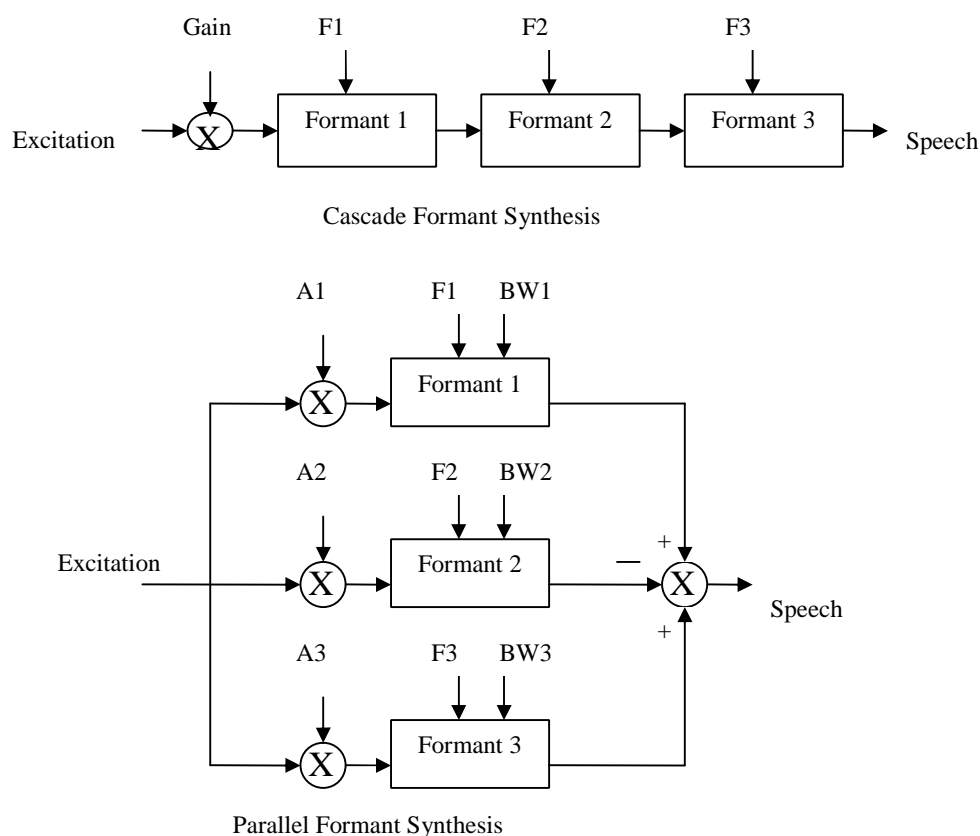


Figure 2.4: Parallel and Cascade Configuration of the Formants in Formants Synthesis Method (Lemmetty, 1999).

#### 2.2.4.2 Articulatory Synthesis

Denovan (1996) denoted that articulatory synthesis is a synthesis method that tries to model the human vocal organs as perfectly as possible, so it is potentially the most satisfying method to produce high-quality synthetic speech. It typically involves models of the human articulators and vocal cords (Johan, 1996). The articulators are usually modeled with a set of area functions between glottis and mouth such as lips, teeth, tongue and jaw while considering the interaction of the articulator mass and the muscle forces (Christine and Robert, 2001). The first articulatory model was based on a table of vocal tract area functions from larynx to

lips for each phonetic segment (Johan, 1996). The data for articulatory model is usually derived from X-ray analysis of natural speech (Johan, 1996).

The disadvantages of articulatory synthesis are the data collected is usually only 2-D while the real vocal tract is naturally 3-D, and the X-ray data do not characterize the masses or degrees of freedom of the articulators (Luis, 1997). Advantages of articulatory synthesis are that the vocal tract models allow accurate modeling of transients due to abrupt area changes (Christine and Robert, 2001).

According to Lemmetty (1999), the articulatory synthesis is quite rarely used in present systems because of two limitations. First, physical measurements of the articulatory size, shape, mass and muscular makeup are difficult to obtain and vary from speaker to speaker. Second, mathematical modeling of the airflow characteristics of the vocal tract and articulator trajectories is computationally intensive compared with other existing methods for speech synthesis (Johan, 1996). But since the analysis methods are developing fast and the computational resources are increasing rapidly, it might be a potential synthesis method in the future (Gold and Morgan, 2000).

#### **2.2.4.3 Concatenative Synthesis**

Concatenative synthesis is a synthesis method that connects pre-recorded natural utterances to produce intelligible and natural sounding synthetic speech (Andersen et al, 1998). It uses speech waveforms or compressed representations (e.g. LPC) that have been stored in database to concatenate the speech during synthesis (Taylor et al, 1998). Concatenation can be actually accomplished by either overlap-adding stored waveforms or by reconstruction using method such as linear prediction or even formant synthesis (Bryan, 1998).

One of the most important aspects in concatenative synthesis is to find correct unit length (Lewis and Mark, 1999). The selection is usually a trade-off between longer and shorter units (Black and Campbell, 1995). With longer units, high naturalness, less concatenation points and good control of co-articulation are



achieved, but the amount of required units and memory is increased. With shorter units, less memory is needed, but the sample collecting and labeling procedures become more difficult and complex (Taylor et al, 1999). The units used for present concatenative synthesis systems are usually words, syllables, demi-syllables, phonemes, diphones, and sometimes even tri-phones (Janet and Sangho, 1999).

There are various methods to accomplish concatenative speech such as Linear Prediction (LP), residual excited linear prediction (RELP), and pitch-synchronous overlap and add (PSOLA).

#### (1) LP

Linear Prediction (LP) synthesis has been used extensively in concatenation systems, since it enables the rapid coding of concatenation units (Macon et al, 1997). The basis of linear prediction theory is the assumption that the current speech sample  $y(n)$  can be predicted as a linear combination of the previous  $P$  samples of speech, plus the small error term  $e(n)$  (Rabiner, 1993). Thus,

$$e(n) = \sum_{i=0}^P a(i)y(n-i) \quad \text{where } a(0) = 1, \quad (2.3.1)$$

from the equation (2.3.1),  $a(i)$  are the linear prediction coefficients, and the  $P$  the linear prediction order. The LP coefficients are found by minimizing the sum of the squared errors over the frame of speech under analysis (Parsons, 1987).

#### (2) RELP

The difference between the predictor and the original signal is referred to as the error signal, also sometimes called the residual error, the LPC residual or residual excited linear prediction (RELP), or the prediction error. RELP synthesis is used in L&H's commercial TTS system (Christine and Robert, 2001). According to Gold and Morgan (2000), the block diagram for RELP is illustrated in Figure 2.5. From Figure 2.5, the processed error signal is used only as excitation (Rowden, 1992).

Rowden (1992) stated that RELP technique allows the residual to be coded by conventional waveform coding techniques such as PCM, DPCM, or DM. He also stated that if the bandwidth of the residual is the same as for the original speech, the coding advantage comes from the reduced variance of the residual, needing fewer bits to achieve a given signal to noise ration. A very natural sounding speech can be produced, although this increase the total bit rate to about 16 kbit/s (Klatt, 1987). Residual-excited LPC has being used in developing various concatenative synthesis systems such as English, German, Spanish, Japanese etc (Macon et al, 1997).

Actually, RELP is a pitch synchronous technique. It required information about where pitch periods occur in the acoustic signal (Macon et al, 1997). Where possible, it is better to record with an electroglottograph (EGG, also known as a laryngograph) at the same time as the voice signal. The EGG records electrical activity in the glottis during speech, which makes it easier to get the pitch moments, and so they can be more precisely found (Taylor et al, 1998). The Edinburgh Speech Tools include a program 'pitchmark' which will process the EGG signal giving a set of pitchmarks. However it is not fully automatic and requires someone to look at the result and make some decisions to change parameters that may improve the result (Alan and Kevin, 2000).

There are few major steps involved to build the RELP database that can be used for diphone RELP synthesis:

Step 1: Labeling the diphone. This step involves the checking of the correct possible of diphone using emu-labelling tools.

Step 2: Extract pitch mark for the diphone.

Step 3: This step uses LPC analysis to extract LPC parameter and LPC residual. This required the pitch marks that are created before the LPC analysis takes place.

Step 4: Group the extracted LPC parameter and LPC residual database in one file.

The extracted LPC parameter and LPC residual can be used later in re-synthesis by passing these two files through a LPC synthesis module as in Figure 2.5.

The detail of LPC analysis and LPC synthesis can be found in Appendix B (Witten, 1982).

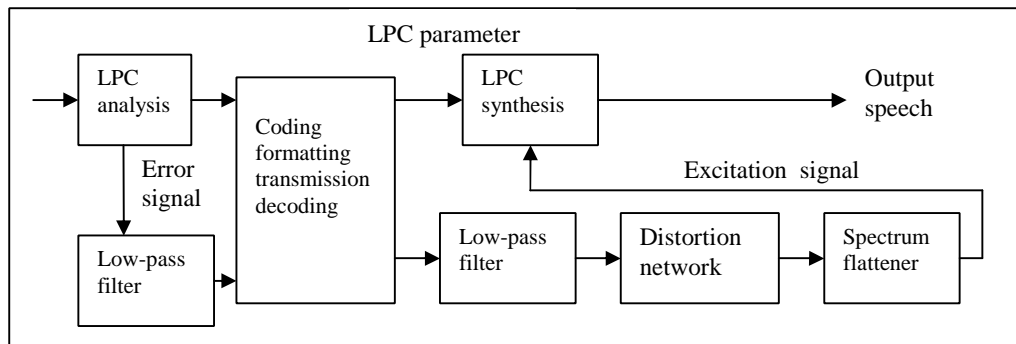


Figure 2.5: Block diagrams of Residual-excited Linear Predictive (REL)

## (2) PSOLA

The pitch-synchronous overlap and add (PSOLA) synthesis method concatenated pitch synchronously (Moulines and Charpentier, 1990). It is actually not a synthesis method itself but allows prerecorded speech samples smoothly concatenated and provides good controlling for pitch and duration, so it is used in some commercial synthesis systems, such as ProVerbe and HADIFIX (Thierry, 1993). The alignment to pitch period permits a variation of pitch frequency by varying the timing repeats for each waveform type. The example of increasing or decreasing the pitch is shown in Figure 2.6. From Figure 2.6, we can see the process of increasing the pitch and decreasing the pitch using PSOLA synthesis method (Lemmetty, 1999).

At first, it computes the pitch marks. Then, window such as Hamming window is erected around the pitch mark, of the integer number of periods for voiced speech. The resultant windowed speech fragments (frags) (Gold and Morgan, 2000). If the segment is properly added, it will produce the original signal. If, now, some frags are selectively removed and the remaining frags are added in an overlap-add way, the result is a sped-up version of the original signal with the same pitch and spectrum as the original (Taylor et al, 1999).

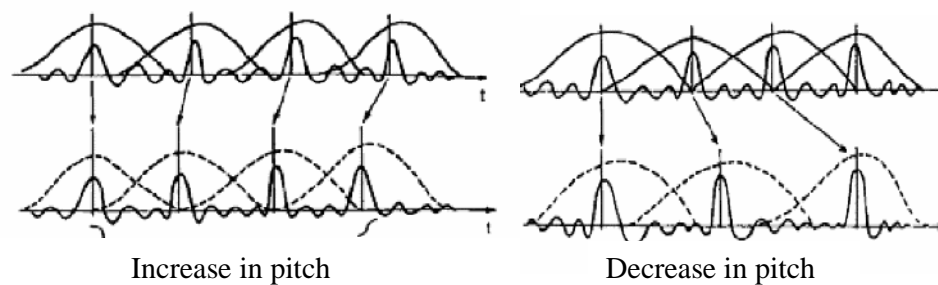


Figure 2.6: Pitch modification of a voiced speech segment (Lemmetty, 1999).

### 2.2.5 Current Commercial and Non-commercial TTS System

This section will provide some brief information about current available commercial or non commercial TTS systems which will provide some ideas on developing Malay TTS system that would discuss in Chapter 3 and 4.

#### (1) Infovox

Infovox was developed in Sweden at the Royal Institute of Technology in 1982. First commercial version of Infovox is Infovox SA-101. It was originally descended from OVE cascade formant synthesizer (Klatt, 1987). Several versions of current system are available for both software and hardware platforms. Infovox 230 is the latest full commercial system which is available for American and British English, Danish, Finnish, French, German, Icelandic, Italian, Norwegian, Spanish, Swedish, and Dutch (Gold and Morgan, 2000).

#### (2) DECTalk

The DECTalk system is originally descended from MITalk and Klattalk (Klatt, 1987). It is now available for American English, German and Spanish and offers nine different voice personalities, four male, four female and one child. It has the capable to say most proper names, e-mail and URL addresses and supports a

customized pronunciation dictionary. It has also punctuation control for pauses, pitch, and stress and the voice control commands may be inserted in a text file for use by DECTalk software applications. The present DECTalk system is based on digital formant synthesis. The speaking rate is adjustable between 75 to 650 words per minute (Gold and Morgan, 2000).

(3) Bell Labs Text-to-Speech

The first full TTS system in Bell Labs was demonstrated in Boston 1972 and released in 1973. It was based on articulatory model developed by Cecil Coker (Klatt, 1987). The concatenative synthesis system in Bell Labs TTS was started by Joseph Olive in mid 1970's (Möbius et al, 1997). Present Bell Labs TTS system is based on concatenation of diphones, context-sensitive allophone units or even of tri-phones. Currently it is available for English, French, Spanish, Italian, German, Russian, Romanian, Chinese, and Japanese (Jan et al, 1996).

(4) Lernout & Hauspies

Lernout & Hauspies (L&H) produces several TTS products with the features which depending on the markets they are used (Lemmetty, 1999). It is available for American English, German, Dutch, Spanish, Italian, and Korean. The architecture is based on concatenation of rather long speech segments, such as diphones, tri-phones, and tetra-phones (Gold and Morgan, 2000).

(5) Whistler

Microsoft Whistler (Whisper Highly Intelligent Stochastic TaLKER) is a trainable speech synthesis system which is under development at Microsoft Research, Richmond, USA (Gold and Morgan, 2000). It is based on concatenative synthesis and the training procedure on Hidden Markov Models (HMM). The Whistler's text analysis component is derived from Lernout & Hauspie's TTS engine. The speech engine supports MS Speech API and requires less than 3 Mb of memory.

## (6) Festival TTS System

The Festival TTS system was developed at Center for Speech Technology Research (CSTR) at the University of Edinburgh by Alan Black and Paul Taylor and in co-operation with CHATR, Japan (Taylor et al, 1998). Currently it is available for American and British English, Spanish, and Welsh (Black and Lenzo, 2001). The system is written in C++ and supports residual excited LPC method (Taylor et al, 1998).

This system is available free for educational, research, and individual use. It is developed for three different aspects. For those who want simply use the system from arbitrary text-to-speech, for people who are developing language systems and wish to include synthesis output, such as different voices, specific phrasing, dialog types and so on, and for those who are developing and testing new synthesis methods (Tan and Sheikh, 2003).

Table 2.1: The comparison between current commercial and non commercial TTS system.

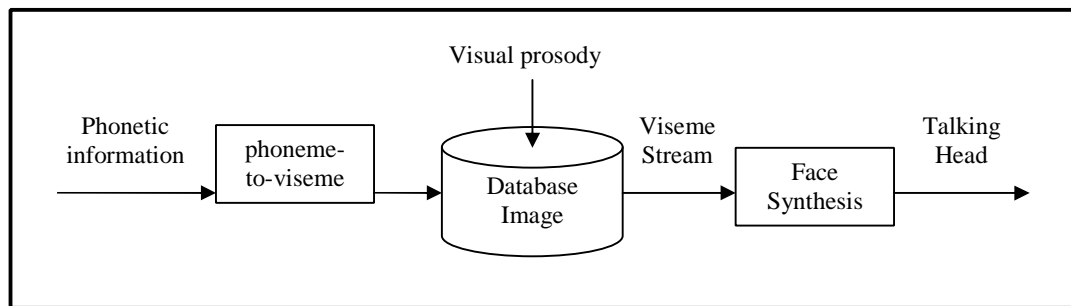
Product	Specification	Advantages/ disadvantages
Infovox 230	<b>Platform:</b> Windows 95/NT. <b>Languages:</b> Finnish, English, Spanish, French. <b>Method:</b> Formant	Available for commercial purpose. Not for research purpose.
Dectalk	<b>Platform:</b> Windows 95/NT. <b>Languages:</b> English, Spanish, German. <b>Method:</b> Formant	Available for commercial purpose. Not for research purpose.
Bell Lab TTS	<b>Platform:</b> Unix, Windows 95. <b>Languages:</b> Finnish, English, Spanish, French, German, Italian. <b>Method:</b> Concatenative	Available for commercial purpose. Not for research purpose.
Lernout and Hauspie	<b>Platform:</b> Windows 95/NT.	Available for commercial purpose. Not for research

	<b>Languages:</b> Finnish, English, Spanish, French, German, Italian. <b>Method:</b> Concatenative	purpose.
Microsoft Whistler	<b>Platform:</b> Windows 95/NT. <b>Languages:</b> English, Spanish, German. <b>Method:</b> Concatenative	Free for research purpose in applying TTS, but not developing new TTS. No detail manual in developing new TTS.
Festival (non commercial)	<b>Platform:</b> Unix, Windows 95/NT. <b>Languages:</b> English, Spanish, German. <b>Method:</b> Concatenative	Free for research purpose in TTS. Manual for development new TTS also include. Support complete TTS modules development.

From Table 2.1, it can be concluded that Festival is the best choice for developing Malay TTS system. All the listed commercial and non commercial TTS are not used for developing new TTS system. Only Festival TTS as non commercial TTS system provides the platform and manual for developing New TTS system. Besides that, a complete module designing manual for new TTS system such as NLP and WP also include in the manual.

### 2.3 Overview of Audio Visual Speech Synthesis

Talking Head is a computer-user interface or communication aid which visual prosody, facial movements and lip synchronization with text are the main issues of it. It can either be in 2D or 3D and in cartoon (graphical) format or human realistic talking face.



**Figure 2.7** The architecture of Talking Head Driven by Text

As mentioned in chapter 1.3, there are three important steps to construct this Talking Head. The first step is model a talking face. Secondly, implement the facial movements to the talking face. And last but not least, implementation of talking head. Figure 1.3 shows the implementation block diagram of Talking Head. First, the phone/phoneme information will be received, and then the system will call out compatible visemes from the database to form a viseme stream. Meanwhile, visual prosody such as smiling and frowning will be added and combine with the viseme stream. This process will run repeatedly and every time there are slightly different between images. Therefore, the result will be seems like this talking head is talking with multi-expressions.

### 2.3.1 History and development of Talking Head

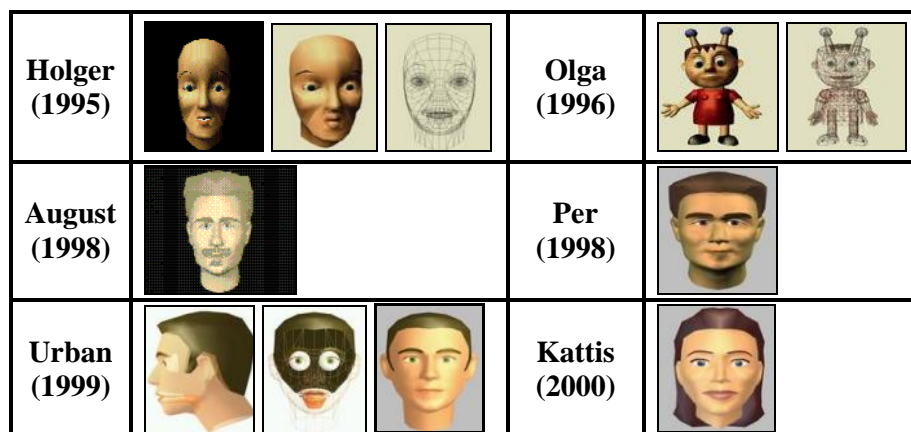
Facial animation and Talking Head has been applied to synthetic speech for about ten years. Most of the present audiovisual speech synthesizers are based on a parametric face model presented by Parke in 1982. The model consisted of a mesh of about 800 polygons that approximated the surface of a human face including the eyes, the eyebrows, the lips, and the teeth. The polygon surface was controlled by using 50 parameters [3]. However, present systems contain a number of modifications to Parke model to improve it and to make it more suitable for synthesized speech. These are usually a set of rules for generating facial control parameter trajectories from phonetic text, and a simple tongue model, which were not included in the original Parke model.



And, Human facial expression has been under investigation for more than one hundred years. The first computer-based modeling and animations were made over 25 years ago. In 1972 Parke introduced the first three-dimensional face model and in 1974 he developed the first version of his famous parametric three-dimensional model [12]. Since the computer capabilities have increased rapidly during last decades, the development of facial animation has been also very fast, and will remain fast in the future when the users are becoming more comfortable with the dialogue situations with machines.

### 2.3.2 Previous work

Figure 3.1 shows some of the previous works of English Talking Head (Malay Talking Head have not formally developed until now). There are still many more projects relevant with English Talking Heads, but just some were pasted.



**Figure 2.8** Previous Works of English Talking Head

### 2.3.3 Visemes

The phoneme is the smallest identifiable unit of speech, using a letter-to-sound correspondence protocol. In a similar fashion, visual attributes are described using the *viseme* (visual phoneme). The term viseme was coined by C. G. Fisher to describe “any individual and contrastive visually perceived unit” [13]. In the English language, there are many acoustic speech sounds that are visually ambiguous. For example, the phonemes /p/, /b/ and /m/ are all articulated in the same manner with the

mouth closed; therefore appearing visually the same. These phonemes are grouped into the same viseme class. The term viseme is not the only term used to describe visual speech. Jeffers and Barley used the term *speechreading movement* to describe any “recognizable visual motor pattern usually common to two or more speech sounds” [14]. Both of these terms, *viseme* and *speechreading movement*, are synonymous and are used interchangeably throughout the literature. Larger units, such as words can also be visually ambiguous. The word “pad”, “bat”, and “man” are all visually indistinguishable and said to be *homophonous*. The term *homophonous* refers to speech sounds or words that appear to be alike on the lips and cannot be distinguished by visual cues alone. The elements of *homophonous* group correspond to the same viseme, but a viseme is not necessarily identical to a *homophonous* group.

#### **2.3.4 Visemes formed from phonemes of English**

Currently there is no viseme system that universally encompasses all phonemes in all visual communication settings [15]. Tables 3.1 through 3.4 list viseme groupings from several published studies. In these studies, the investigators used 16 or more consonants in a nonsense syllable identification task, using /a/ as the vowel component in either a CV (consonant-vowel) or CVC (consonant-vowel-consonant) structure. Visemes were classified using a 75% within-group identification rate; except in the case of Binnie, Jackson, and Montgomery (see Table 3.1) where a 70% identification rate was used. These studies show that /p, b, m/, /f, v/, and /th, dh/ are universally recognized visemes. The other groupings in the tables are not as stable and vary between studies. These variations most likely reflect differences among speakers and poor visibility of the phonemes being grouped. Unfortunately not all speech sounds are visible on the lips. For example, voicing sounds are produced in the vocal tract and do not appear on the lips. Research by Woodward and Barber revealed that only 25-30% of all speech phonemes has uniquely distinguishable patterns [16]. Some took this result to mean that lipreading could improve speech intelligibility by only 25-30%. The personal experiences of Robert Menchel have shown that other visual and context cues improve speech

intelligibility. For example, knowing the topic of a discussion helps pull knowledge of the vocabulary related to the topic [17].

Erber [18]	Binnie, Montgomery, & Jackson [19]	Binnie, Jackson, & Montgomery [20]
/p, b, m/	/p, b, m/	/p, b, m/
/f, v/	/f, v/	/f, v/
/th, dh/	/th, dh/	/th, dh/
/sh, zh/	/sh, zh/	/sh, zh/
/w, r/	/t, d, n, s, z, k, g/	/w/
/n, d, t, s, z/		/r/
/l/		/t, d, s, z/
/k, g/		/l, n/
/h/		/k, g/ (assigned with 68.4%)
		Unassigned: /y/ with 48%

**Table 2.2** Viseme groupings of consonants I

Walden, Erdman, Montgomery, Schwartz, & Prosek [22]	
Pretraining	Post-training
/p, b, m/	/p, b, m/
/f, v/	/f, v/
/th, dh/	/th, dh/
/sh, zh, ch, jh/	/sh, zh, ch, jh/
/w, r/	/w, r/
Unassigned: /t, d, s, z, y, k, n, g, l/	/t, d, s, z, y, k, n, g, l/
Due to 72.6% within-group identification	

**Table 2.3** Viseme groupings of consonants II

Walden, Prosek, Montgomery, Scherr, & Jones [21]	
Pre-Training	Post-Training
/p, b, m/	/p, b, m/
/f, v/	/f, v/
/th, dh/	/th, dh/
/s, z, sh, zh/	/sh, zh/
/w/	/w/
Unassigned:	/r/
/t, d, n, k, g, y, r, l/	/s, z/
Due to 70.9% within-group identification	/l/
	/t, d, n, k, g, y/

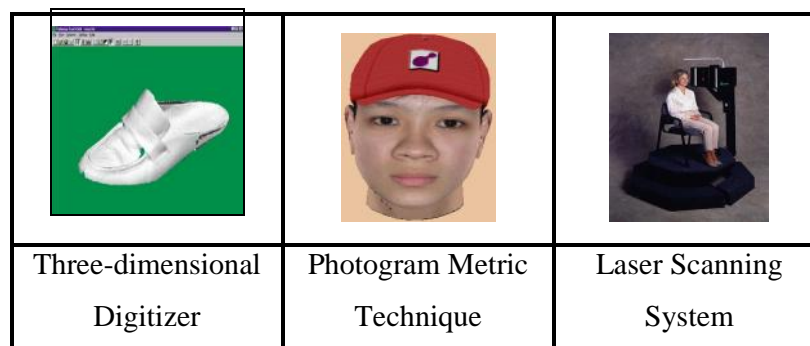
**Table 2.4** Viseme groupings of consonants III

Owens & Blazek [23]	Lesner, Sandridge, & Kricos [24]	Williams, Rutledge, Katsaggelos, & Garstecki [25, 26]
/p, b, m/	/p, b, m/	/p, b, m/
/f, v/	/f, v/	/f, v/
/th, dh/	/th, dh/	/w/
/sh, zh, ch, jh/	/sh, zh, ch, jh/	/l/
/w, r/	/w, r/	/sh, zh/
/k, g, n, l/	/l/	/th, dh/
/h/	/t, d, s, z, n, k, g, y/	/r/
Unassigned: /t, d, s, z, y/		/n, y, d, g, k, s, z, t/
Due to within-group identification less than 75%		

**Table 2.5** Viseme groupings of consonants IV

### 2.3.5 Methods to model a photo-realistic face

To design a Talking Head, first we need to create a talking face. Several different techniques exist for modeling the human head, achieving various degrees of realism and flexibility. There are totally 3 methods to model a photo-realistic talking face, which are three-dimensional digitizer, photogram metric technique and laser scanning system.



**Figure 2.9** Methods to model a photo-realistic face

## 2.4 Speech Recognition

### 2.4.1 Isolated Word Recognition System

The isolated word recognition system consisted of three modules: feature extraction, HMM model training, HMM recognizer. Each of these modules is discussed as below.

### 2.4.2 Feature Extraction

The feature extraction used in the system front end to extract important features from speech signals to reduce the data size before used for training and recognition. Our front end uses Mel-cepstral frequency coefficient extraction (MFCC) [10] block diagram of the front end is shown in Fig. 1.

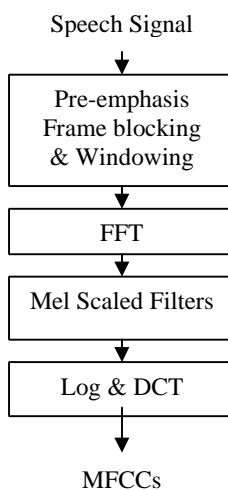


Figure2.10 Block diagram of MFCC front end.

For the DHMM We used 128 orders VQ codebook (a set of 128 codevectors) generated from Linde-Buzo-Gray (LBG) algorithm [11]. However, the MFCC feature vectors were directly used for CDHMM modeling without prior vector quantization. The figure 2 shows the above differences.

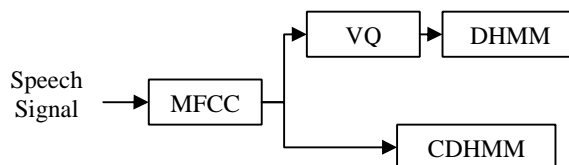
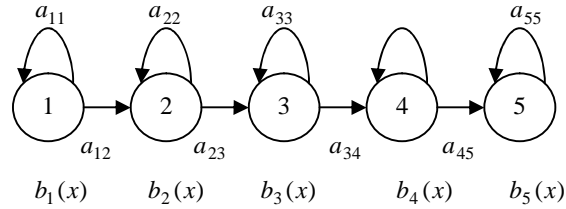


Figure 2.11 Feature extraction for DHMM & CDHMM.

## 2.5 HMM models

Figure 2.12 type of HMM. It is a 5 state left-to-right model.



In CDHMM,  $b(x)$  is represented by a probability density function (pdf), where in this paper we use mixture of Gaussian pdfs as the observation emission probability. The pdf at state  $j$  is given as follows.

$$b_j(x) = \sum_{m=1}^M c_{jm} \mathbf{N}(x, \mu_{jm}, \Sigma_{jm})$$

Where  $\mathbf{N}$  is the normal distribution and  $c$  is mixture weight of each component.

$\mu_{jm}$  is the mean vector and  $\Sigma_{jm}$  is the covariance matrix of the  $m$  mixture component at state  $j$ .

The Gaussian pdf given  $x$ , is given by the following equation. The CDHMM considered in this paper uses multivariate Gaussian mixture densities with diagonal matrices and 4 mixture components.

$$N(x, \mu_{jm}, \Sigma_{jm}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_{jm}|}} e^{-\frac{1}{2}(x - \mu_{jm})^T \Sigma_{jm}^{-1} (x - \mu_{jm})}$$

## 2.6 HMM model training

A word-based HMM acoustic modeling was chosen in this isolated word recognition system. [1]. Baum-Welch (BW) algorithm and Viterbi training/segmentation algorithm can be applied to DHMM and CDHMM. We consider both algorithms in training CDHMM. The CDHMM training based on BW algorithm is shown in Figure 4.

The initial model estimate can be chosen randomly or based on prior knowledge (available model or uniform segmentation). Good initial estimates of parameters of

pdf are essential for rapid and proper convergence of the re-estimation formulas. Uniform segmentation was used to segment each training utterances uniformly into  $N$  segments with each segment  $S_j$ , correspond to each HMM state,  $j$ , then vectors of  $j$ th segment for all training utterance was grouped into cluster,  $C_j$ . Finally each  $C_j$  was used to estimate initial  $b_j$ .

The second step is, for each training utterance, compute  $\alpha_t^k(i)$  and  $\beta_t^k(i)$  using forward backward algorithm [1].

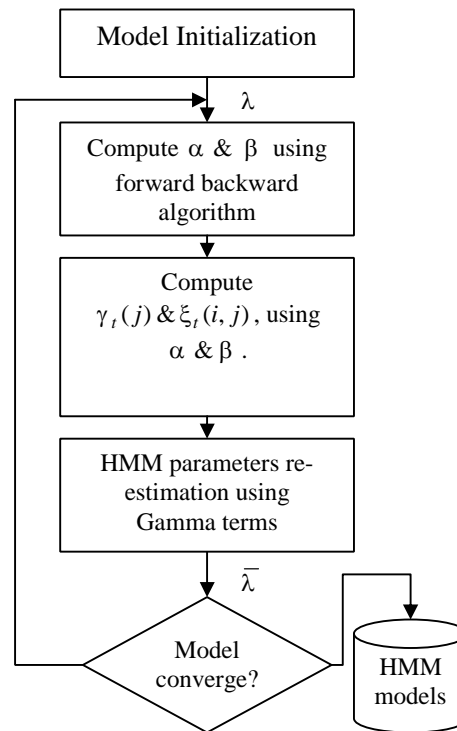


Figure 2.13: HMM training with BW algorithm.

Following step is to calculate the  $\gamma$  and  $\xi$  using  $\alpha$  and  $\beta$  values. The detail computation of  $\gamma$  and  $\xi$  can be found in [1]. Then, the model parameters are re-estimated with the computed gammas terms as following, this set of equation can be found in [1].

$$\bar{c}_{jm} = \frac{\sum_{k=1}^K \sum_{t=1}^T \gamma_t^k(j, m)}{\sum_{k=1}^K \sum_{t=1}^T \sum_{m=1}^M \gamma_t^k(j, m)}$$

(6)

$$\bar{\mu}_{jm} = \frac{\sum_{k=1}^K \sum_{t=1}^T \gamma_t^k(j, m) \cdot x_t^k}{\sum_{k=1}^K \sum_{t=1}^T \gamma_t^k(j, m)}$$

(7)

$$\bar{\Sigma}_{jm} = \frac{\sum_{k=1}^K \sum_{t=1}^T \gamma_t^k(j, m) \cdot (x_t^k - \mu_{jm})(x_t^k - \mu_{jm})^T}{\sum_{k=1}^K \sum_{t=1}^T \gamma_t^k(j, m)}$$

(8)

Finally, an updated model,  $\bar{\lambda}$  is obtained. If the likelihood of the training set has not converged, the old model is replaced by new model  $\bar{\lambda}$  and overall training is repeated.

Next discussed is Viterbi/segmentation training. The HMM training based on Viterbi/segmentation training as shown in figure 5. Segmental K-mean procedure [1,2,5] is used to re-estimate the B parameters. The model initialization is the same as of the BW algorithm discussed above. Following procedure is to segment each training observation sequence corresponding to states based on the current model  $\lambda$ . This segmentation is achieved by finding the optimum state sequence  $\bar{q}$ , via Viterbi algorithm, and then backtracking along the optimum path. Each observation sequence is segmented into N segment with each segment  $S_j$ , correspond to each HMM state, j. The result is, for each of the N states, a set of the observations that occur within each state.

Next is the optimization step. Here, a segmental K-means procedure is used to cluster the observation vectors within each state  $S_j$  into a set of M clusters, where



each cluster represents one of the  $M$  mixtures of the  $b_j(x)$  density. In this work, the observation vectors are associated with the mixture component with the highest likelihood and the K-Mean clustering is used in initial uniform segmentation stage. From the clustering, an

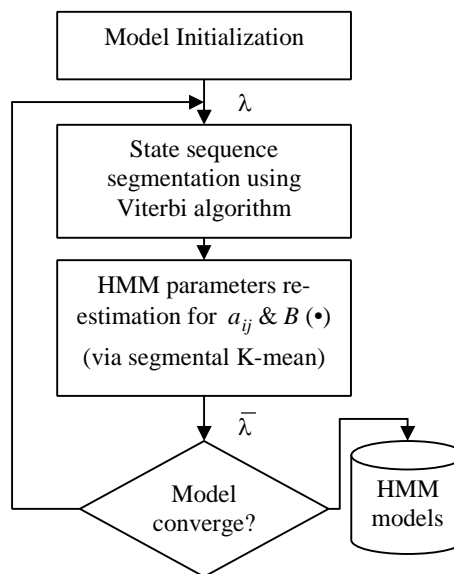


Figure 2.14 HMM training with Viterbi algorithm (modified from [5]).

updated set of parameters, is derived. The  $\bar{a}_{ij}$  obtained from the number of vectors from state  $i$  for which the subsequent vector from state  $j$ , divided by the number of vectors that are in state  $i$ . This work also considers training procedure which combines BW and VB (for bootstrapping model) [1,5].

## 2.7 The HMM Recognizer

Once the HMMs have been trained on each vocabulary word, the recognition strategy [1] is straightforward. The unknown speech signal to be recognized is first analyzed using MFCC feature extraction. Then, for each vocabulary word model, the optimum state sequence is found via the Viterbi algorithm and the log likelihood score for the optimal path is computed. The unknown word assigned to the vocabulary word whose model has the highest likelihood score.

## **CHAPTER 3**

### **RESULT AND DISCUSSION**

#### **3.0 Introduction**

This chapter will review the techniques and algorithm involve in developing smart home system. It will cover review of smart home system, review of Malay Text to Speech and Audio Visual, and Review of Speech Recognition system in Smart Home application.

#### **3.1 Malay Text to Speech**

##### **3.1.1 MALAY TTS Smoothing Engine**

To smooth the concatenation joint of phoneme unit, the Malay TTS smoothing engine which utilizes PSOLA's overlap add function has been designed. Figure 2(a) shows the smoothing process in MALAY TTS engine. The unit sequence to be smoothed will first go through pitch detector in PSOLA module to mark all pitch mark. Then last pitch mark of the first unit and first pitch mark of the second unit of two adjacent units will go through windowing process and then overlapped as illustrated in Figure 2(b) and Figure 2(c). This also provides the advantages to the next process of speech modification (it would not be discussed in detail here) which involve pitch location modification for intonation control.



Figure 3.1: MALAY TTS smoothing process.

The windowing in first unit and next unit as shown in Figure 2(b) will reduce the amplitude of the wave at the end of first unit and the start of the next unit due to windowing function. Then, the two units will be overlap-added together and the amplitude of the artifact at the joint will become insignificant.

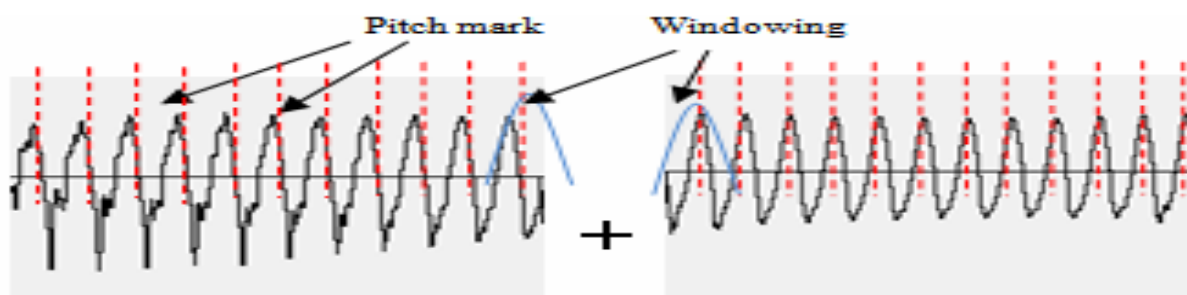


Figure 3.2: Pitch marking and overlap of two adjacent joint unit.

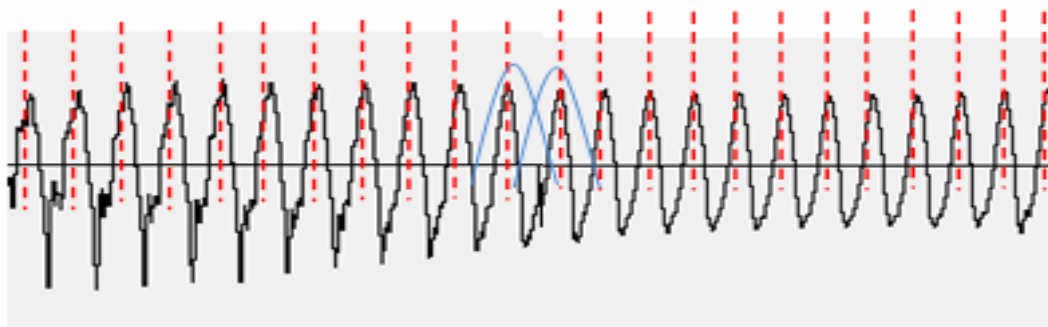
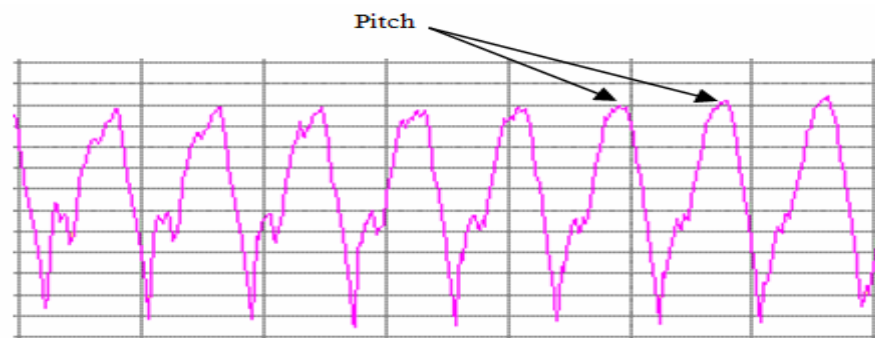


Figure 3.3: The overlapping of two adjacent joint unit.

### 3.1.2 Pitch Detector and Marking Module

The first process in smoothing is to detect the pitch location. The pitch detector in PSOLA has been utilized for this purpose. Pitch synchronous speech synthesis algorithms requires the beginning location of the pitch period (pitch-mark) for every voiced segment prior to speech synthesis. This synthesis technique is pitch synchronous. In other words, it requires information about where pitch-marks occur in the acoustic signal. Pitch-marks are especially important in prosodic modification algorithms that employ PSOLA to change the time and pitch scale of a speech signal. It is said that the exact point of marking the periodicity of the signal has an important effect on the final quality of synthesis (Eric and Francis, 1990).

A pitch-mark is defined as the location of the short-time energy peak of each pitch pulse in a speech signal, in other words, the beginning of a pitch period. This pitch pulse corresponds to the glottal closure instant (GCI). From this definition one can see that an unvoiced speech frame does not have pitch-marks since it has no pitch period (Eric and Francis, 1990). Figure 2(d) shows two pitch-marks on the short-time speech signal of the vowel /a/ for Malay Language (*Bahasa Malaysia*)



**Figure 3.4:** An example of pitch marks on the short-time speech signal of the vowel /a/ for Malay Language.

The pitch marking algorithm is shown as below which is taken from (Dennis and Sheldon, 2006):

### *Pitch Detection Steps*

- Step 1: Finds all the pitch marks in the input file and returns the markings in a matrix*
- Step 2: Determines initial settings for block size and mark*
- Step 3: Grabs the next block to examine*
- Step 4: Finds the high point in the block*
- Step 5: Checks to see if there is really a signal in this block*
- Step 6: Checks to see if there is a pitch mark before the current pitch mark*
- Step 7: Finds largest point in block from beginning to current pitch mark*
- Step 8: Checks to see if high mark is sufficient size to be a pitch mark*
- Step 9: Starts the next block to be examined 50 samples after this block*
- Step 10: Makes sure next block size is of sufficient size*
- Step 11: Plot the marks*

One of the sample speech file that has been pass through pitch marking is shown as in Figure 2(e). But this algorithm still will detect some pitch mark at unvoiced area. Further work is now ongoing to fine tune the algorithm so that it will not detect the wrong pitch mark.

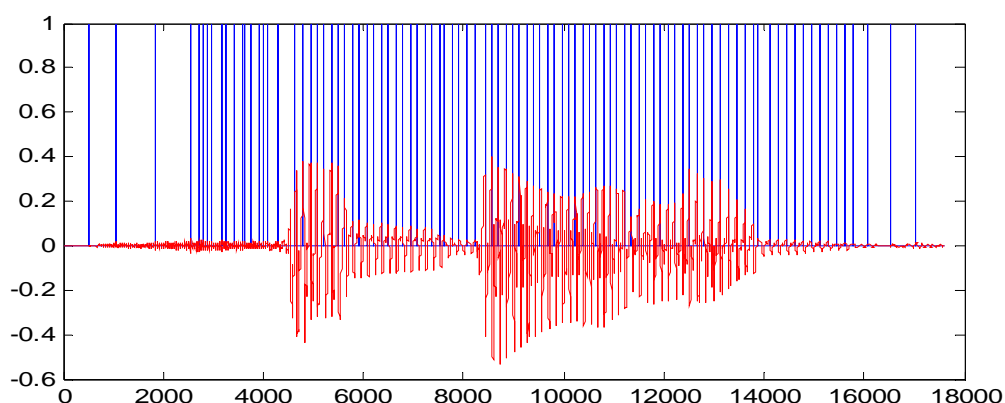


Figure 3.5: Pitch-marks found on the voiced regions of the waveform for the word nine (*Sembilan*).

### 3.1.3 Synchronize overlap-added method

The pre-emphasized speech is separated into short segments called frame. The frame length is set to 20ms (160 samples) to guarantee stationary inside the frame. There is a 10ms (80 samples) overlap between two adjacent frames to ensure stationary between frames, as shown in Figure 2(f) (Wei et al, 2006).

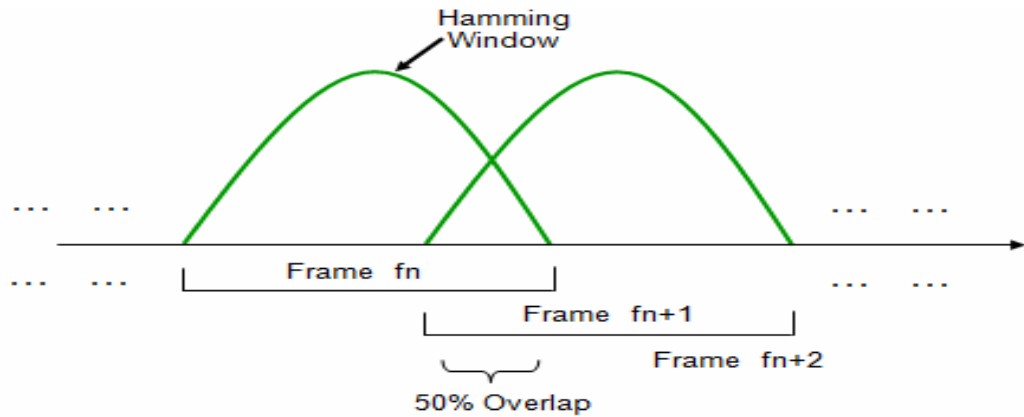


Figure 3.6: Frame blocking (Wei et al, 2006).

A frame can be seen as the result of the speech waveform multiplies a rectangular pulse whose width is equal to the frame length. This will introduce significant high frequency noise at the beginning and end points of the frame because of the sudden changes from zero to signal and from signal to zero. To reduce this edge effect, a 160-points Hamming window is applied to each frame. The mathematical expression of Hamming window is shown in equation (2.1) (Wei et al, 2006).

$$Ham(N) = 0.54 - 0.46 \cos\left(2\pi \frac{n-1}{N-1}\right) \quad (2.1)$$

Where N is equal to 160, the number of points in the one frame, and n is from 1 to N.

### 3.1.4 Result of speech units smoothing

Figure 2(g) shows the concatenation result of the word “jangan” from speech units without applying smoothing technique. It can clearly see that there is a dark black line in the spectrogram at all jointing. Figure 2(h) provides the zooming look of the

signal where we can see the artifacts at joint either in waveform or spectrogram. We can clearly hear the “click” sound at jointing.

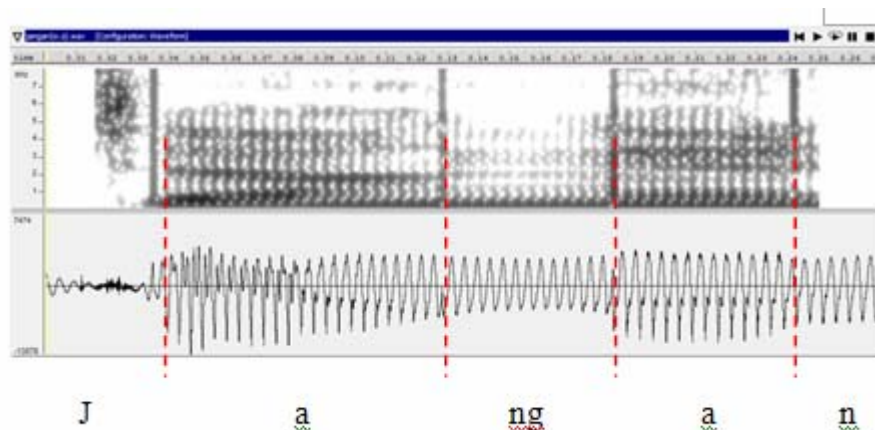


Figure 3.7: Concatenation artifacts for the word “jangan” in Malay.

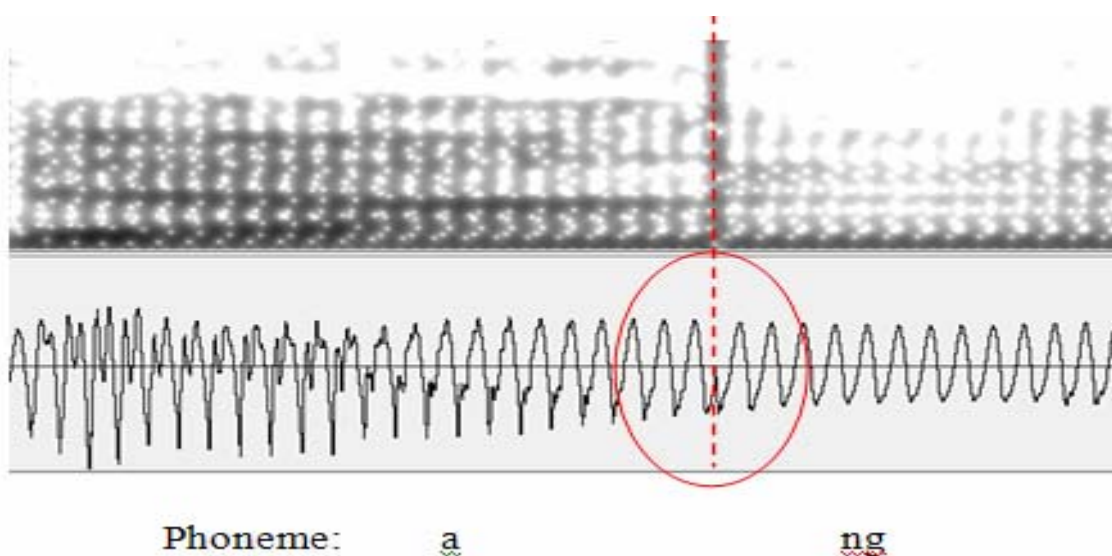


Figure 3.8: The artifacts of joint between two adjacent unit.

When the smoothing technique apply to the joint, the artifacts effect will be remove and it can be clearly see as shown in Figure 2(i). This is because the windowing function will reduce the amplitude of the wavefile at the side of the window. Thus, the amplitude of joint area will become smaller and thus did not producing the artifact sound.

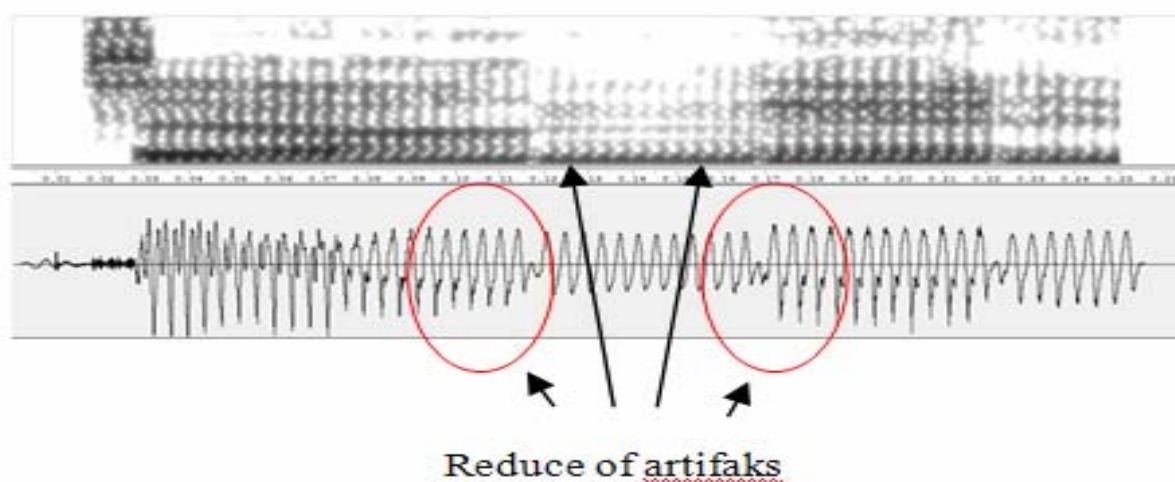


Figure 3.9: The result of concatenative speech through joint smoothing.

The figure shows that the artifacts of the sentence level of unit concatenation been reduced. We can clearly see that the dark black lines have been removed at lower picture.

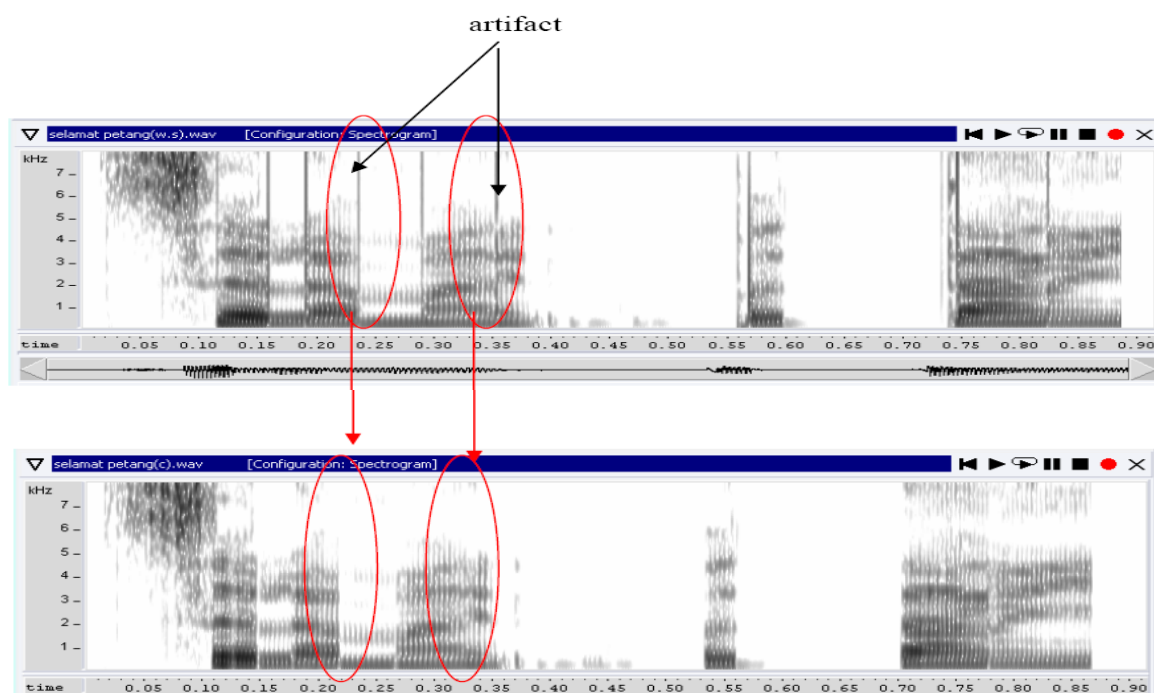


Figure 3.10: output file for “jangan makan nasi.wav”

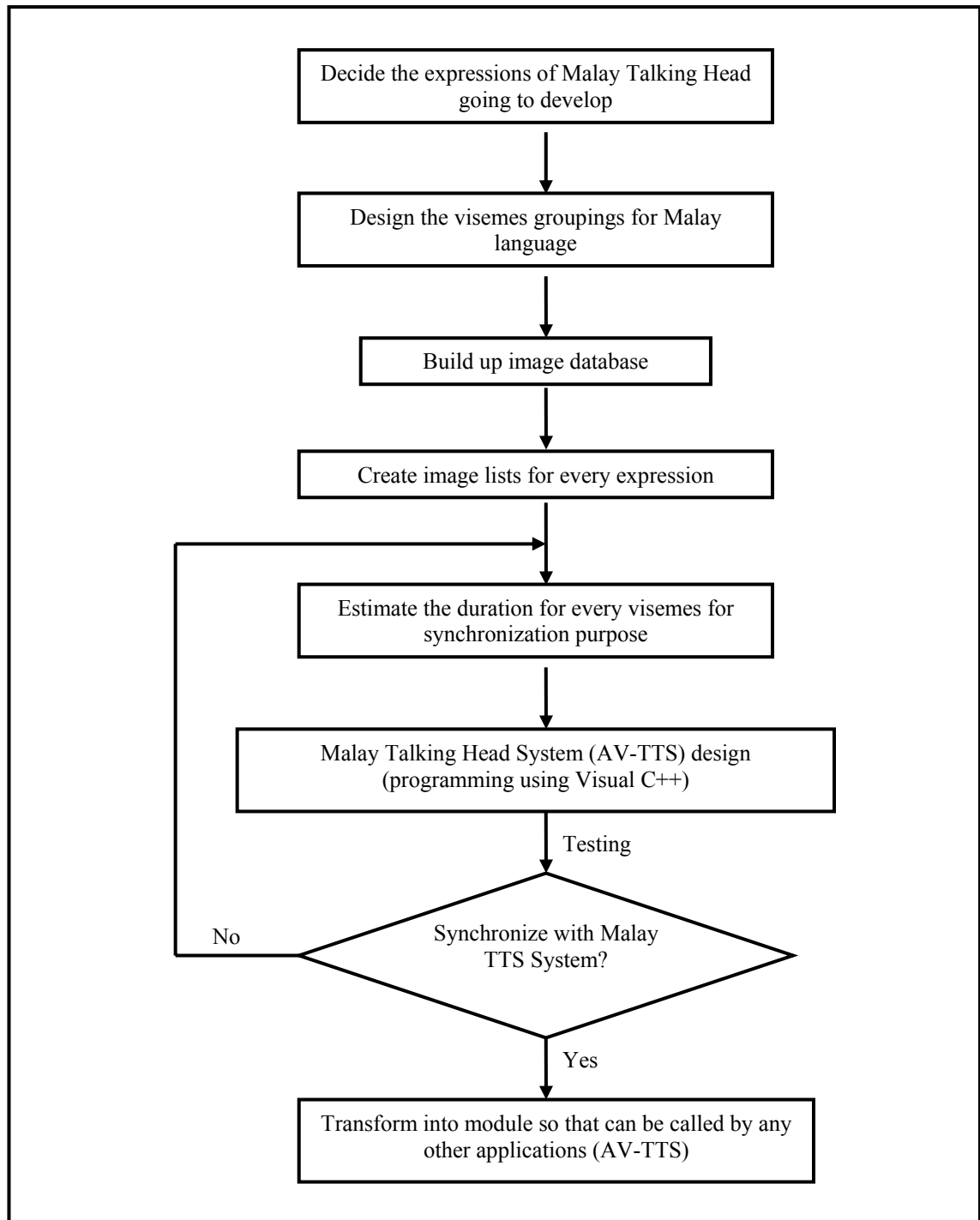


### 3.2 Malay Audio Visual Implementation

Malay TTS system has great potentialities in the market because there is no commercialize Malay TTS system until now. But, only TTS system cannot make it a strong candidate to compete in the market because limited applicable system, not attractive and not intelligent.

Hence, to make Malay TTS system more complete and attractive, Malay Talking Head should be added to form a Malay AV-TTS system which can be widely applied and increase the speech intelligibility significantly. Listener can identify speaker's emotional states easily and also attracted.

Not like English Talking Head which has developed fast and matured now, Malay Talking Head needs to be start from zero, and no software like Microsoft® Agent can be utilized. Hence, in this work, Malay Talking Head development is totally divided into 7 phases as shown in Figure 5.1. Every phase will be presented in details in the following sections. After the Malay Talking Head is finish built up, we need to test it by insert different text inputs to test whether the Talking Head visemes are synchronize with the Malay TTS System. If not, the duration of the relevant viseme will be estimated until it performs correct result. Malay TTS module designed by Mr. Tan Tian Swee is imported to this Malay Talking Head. And Visual C++ is utilized in this design, Malay Talking Head.



**Figure 3.11** Block diagram of design of Malay Talking Head

### 3.2.1 Expressions of Malay Talking Head

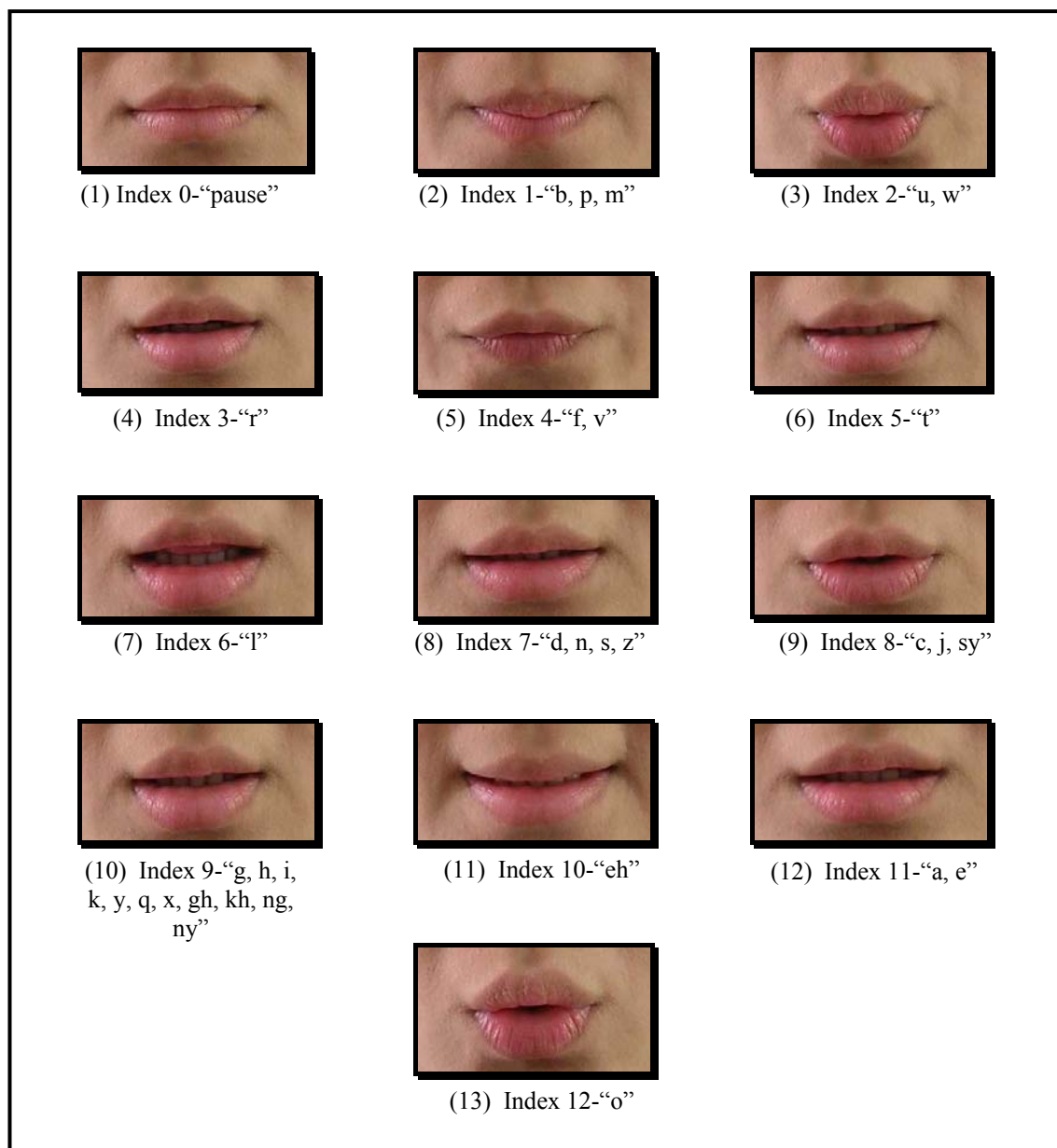
Within scope of this work, there are totally 9 expressions will be developed for Malay Talking Head, which are neutral, happiness, sadness, anger, surprise, boredom, fear, disgust and confuse, which is quite enough to be implemented for any application.

No.	Expressions
1.	Neutral
2.	Happy
3.	Sad
4.	Angry
5.	Surprise
6.	Boredom
7.	Fear
8.	Disgust
9.	Confuse

**Table 3.1** Expressions of Malay Talking Head

### 3.2.2 Visemes groupings of Malay language

Referring to section 3.5.2, Malay language visemes (multiple expressions) design is categorized and listed out in Table 3.7. In this work, these visemes can be re-grouped some more into fewer categories because some of them are very similar in mouth shapes. Thus, after arrangement, visemes of Malay language are grouped into 13 groups which are listed in Figure 5.2. Each expression will need to have an independent set of visemes due to muscle movements are different for every expression. Thus, totally we will have ( 9 x 13 = 117 ) 117 images.



**Figure 3.12** Visemes groupings of Malay Talking Head

### 3.2.3 Image database

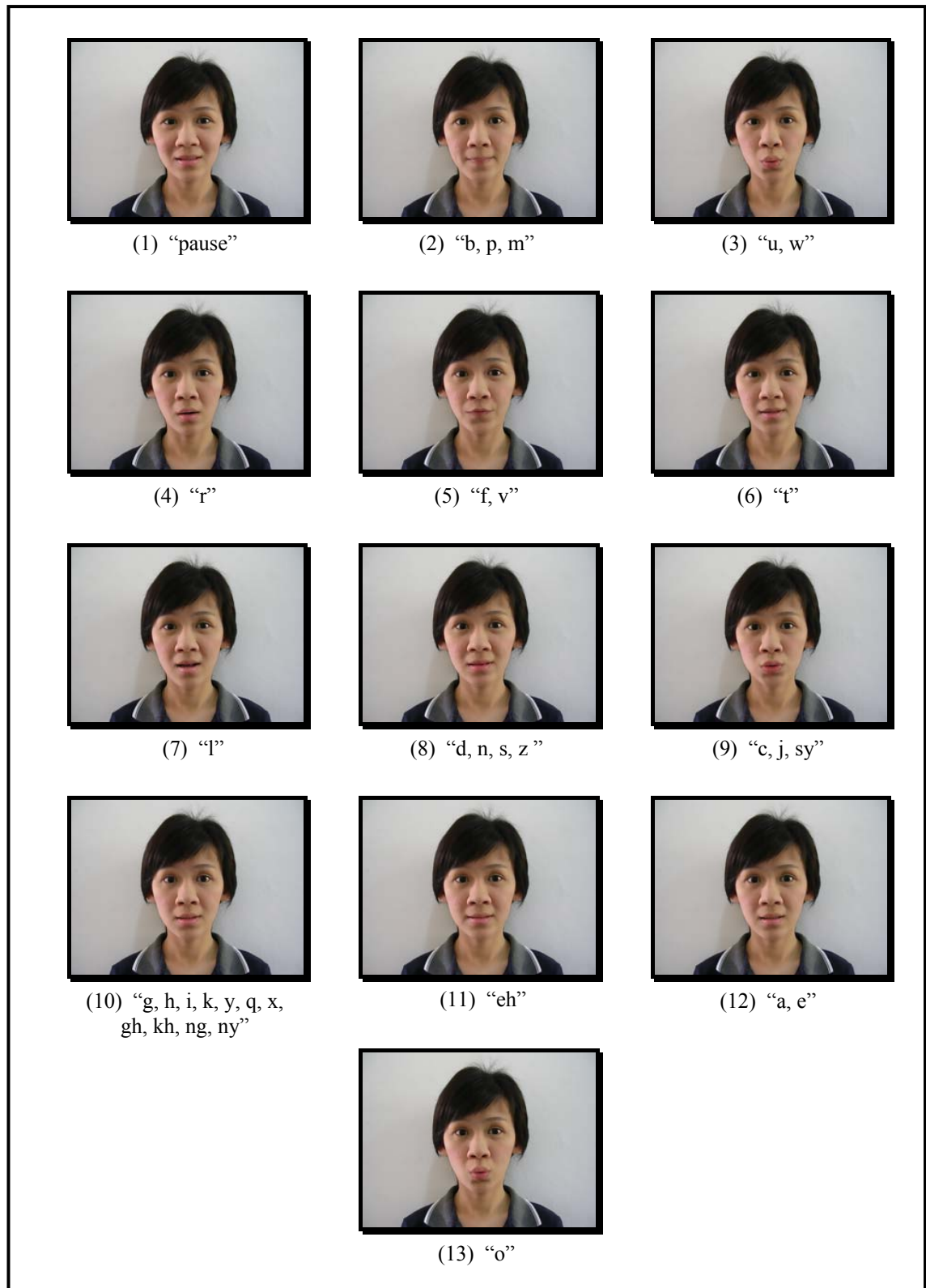
After finished listing out expressions under interest and re-group the visemes of the Malay Talking Head, next should proceed to built up the image database. In this phase, actually the same procedures of image capturing and editing are taken as what we did in English Talking Head. The only difference is no need to convert the color mode to palette (8-bit) because there is no limitation in Malay Talking Head. Directly, the image quality will be improved a lot compare to English Talking Head.

There are totally 13 images (13 visemes) captured for each expression using digital camera (Photogram Metrics Technique). Figure 5.3 shows 13 images taken for surprise expression.

Below are the steps followed in order to build the image database for Malay Talking Head:

1. Decide how many images need to take.
2. Set up tripod; adjust best height, brightness, and position. Use white color background.
3. Start capturing pictures, but remember do not use the camera flash feature so that the light will not focus on the model's face. And also the Talking Head model must idle and maintain consistently her position.
4. Resize all the images to 250x188 pixels in width and height. Then, save in Windows Bitmap format.

**\*\*All these steps must be done in one time because the position of the Talking Head needed to be maintained. If this process is divided into several time, impossible the best quality of Talking Head can be get due to Talking Head's movements are not smooth.**



**Figure 3.13** Surprise expression's images for Malay Talking Head

### 3.2.4 Duration estimation

After built up the image database, proceed to duration estimation process, which is considered as Talking Head NLP rules configuration. This phase is the most important and also the hardest part in the process of developing Malay Talking Head because it decides whether the Malay Talking Head is synchronized with Malay TTS System or fail to do so. This process will be keep repeating in trial-and-error process until the synchronized result is get. Table 5.2 shows the correct duration estimated for every viseme to play (depends on phonemes) after the process stated above.

In this work, the information which comprises {phoneme, duration, index of viseme} of every Malay phoneme is store in a structure – *struct MouthPhoneme*. Then, an array, *mouth* is created and store the information of every phoneme in the format of structure mentioned above. Codes of this process please refer to Figure 5.4 and Figure 5.5.

Phoneme	Duration	Viseme index	Phoneme	Duration	Viseme index
a	8	11	r	7	3
b	8	1	s	8	7
c	12	8	t	9	5
d	6	7	u	7	2
e	8	11	v	8	4
f	9	4	w	11	2
g	8	9	x	9	9
h	8	9	y	9	9
i	7	9	z	10	7
j	9	8	eh	9	10
k	9	9	gh	10	9
l	8	6	kh	10	9
m	7	1	ng	10	9
n	8	7	ny	10	9
o	7	12	sy	10	8
p	10	1	pause	25	0
q	9	9			

**Table 3.2** Viseme duration for every phoneme of Malay Talking Head

```

/*****
/*   Structure of MouthPhoneme                               */
/*   Consists of Phoneme, duration and Index                 */
/*****
typedef struct MouthPhoneme
{
    LPCSTR  sPho;
    int     time;
    int     index;
}TEST;

```

**Figure 3.14** Codes of MouthPhoneme structure definition

```

/*****
/*   Array store structures of MouthPhoneme                 */
/*   Consists of Phoneme, duration and Index                 */
/*****
TEST mouth[] = {
    {"a",8,11},
    {"b",8,1},
    {"c",12,8},
    {"d",6,7},
    {"e",8,11},
    {"f",9,4},
    {"g",8,9},
    {"h",8,9},
    {"i",7,9},
    {"j",9,8},
    {"k",9,9},
    {"l",8,6},
    {"m",7,1},
    {"n",8,7},
    {"o",7,12},
    {"p",10,1},
    {"q",9,9},
    {"r",7,3},
    {"s",8,7},
    {"t",9,5},
    {"u",7,2},
    {"v",8,4},
    {"w",11,2},
    {"x",9,9},
    {"y",9,9},
    {"z",10,7},
    {"eh",9,10},
    {"gh",10,9},
    {"kh",10,9},
    {"ng",10,9},
    {"ny",10,9},
    {"sy",10,8},
    {"pau",25,0},
};

```

**Figure 3.15** Codes to define array of Malay phoneme which comprises structure  
{phoneme duration, index of viseme}

The purpose these structure and array defined is to let the Talking Head system achieve them in order to get the correct sequence of viseme images and



duration of each viseme after break the text input into phonemes and store them in another array. Finally, when the whole sentence / word got all the responding visemes and their durations, then only the images will be played in sequence according to the information it collects before this concurrently with the speech produced by TTS system.

### 3.2.5 Design of Malay Talking Head and AV-TTS System

Figure 5.8 shows the flow of the complete design of Malay Talking Head System. Functions are created and called in the following sequence:

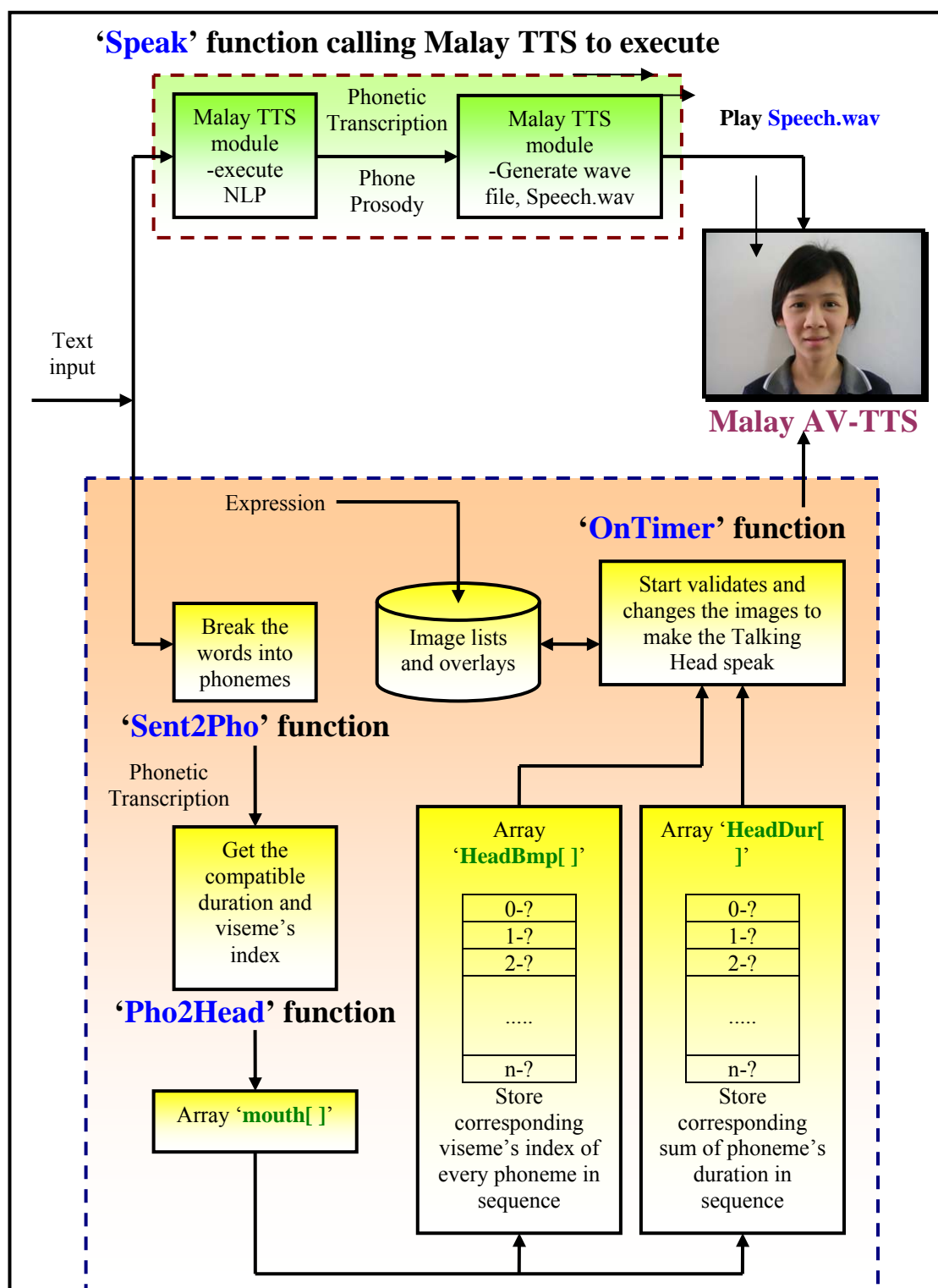
1. Calls '*Speak*' function to generate the wave file.
2. Calls '*Sent2Pho*' function to break the words into phonemes.
3. Calls '*Pho2Head*' function to get the compatible image index (viseme's index) and the corresponding duration of every phoneme.
4. Calls '*OnTimer*' function to play the animations of Talking Head.
5. Plays the '*Speech.wav*'.

First of all, this Talking Head system will take the user text input and call '*Speak*' function which sends the text string to Malay TTS module to execute the natural language processing and generate the corresponding waveform. Then, the generated speech waveform will save as '*Speech.wav*'. Part of the codes of this '*Speak*' function is pasted in Figure 5.9.

While for the Malay Talking Head part, user's text input will be broken into phonemes by '*Sent2Pho*' function (Figure 5.10). Then, '*Pho2Head*' function (Figure 5.11) is called to get the compatible duration of every phoneme and image index (viseme's index). These information will be stored in two arrays, which are *HeadDur[ ]* and *HeadBmp[ ]*. *HeadDur[ ]* stores the sum of the duration of phonemes and last element will be the total duration of the text inputs. While *HeadBmp[ ]* stores the image index (viseme's index) of every phoneme. At the end, the Talking Head's images will change based on the index stored in this *HeadBmp[ ]* array.

Then, '*OnTimer*' function is executed (Figure 5.12). This function actually animate the Talking Head based on the sample-based approach, which means concatenate all the corresponding images in the correct order. It will refer to *HeadDur[ ]* and *HeadBmp[ ]* arrays. Image database also achieved to display the images based on the viseme's index. Only the specified expression's image list will be called.

Finally, 'Speech.wav' is played concurrently with the Talking Head animations. Hence, a complete text-driven photo-realistic Malay Talking Head with multiple expressions is done and form a Malay AV-TTS system.



**Figure 3.16** Flow of the Malay Talking Head and AV-TTS system design

```

/*****
/*
/*  Speak Function
/*  Calls Festival to generate the wave file and save as
/*  speech.wav
/*
/*
/*****
void CTalkingHeadDlg::Speak(CString SpeakStr)
{
FILE * pFile;
pFile = fopen ("myfile.tts","w");

fprintf(pFile, "(voice_utm_ml_hel_diphone)\n");
fprintf(pFile, "(Parameter.set \'Duration_Stretch 0.95)\n");
fprintf(pFile, "(voice_utm_ml_helmi_diphone)\n");
fprintf(pFile, "(set! utt1 (Utterance Text\ \"%s\"))\n", SpeakStr);
fprintf(pFile, "(utt.synth utt1)\n");

fprintf (pFile, "(utt.wave.rescale utt1 2.00)\n");
fprintf (pFile, "(utt.save.wave utt1 \"Speech.wav\")\n");
UpdateData(FALSE);

fprintf (pFile, "(exit)\n");
fclose (pFile);
UpdateData(FALSE);
WinExec("\C:\\Festival\\festival\\bin\\festival.exe" --pipe
        myfile.tts", SW_HIDE);

UpdateData(FALSE);
}

```

**Figure 3.17** Codes of the *'Speak'* function

```

/*****
/*  Sent2Pho Function                                     */
/*  Seperate the sentence into phoneme and calls Pho2Head */
/*  function to find the compatible talking head images.  */
/*****
void CTalkingHeadDlg::Sent2Pho(CString inputString)
{
    //note: need to consider the empty space...
    CString tmpString, nextString; //for temp use
    int length,i; //length of string

    //get the input word length
    length = inputString.GetLength (); //get the length
    totalPho =0;
    HeadDur[0]=0;
    HeadBmp[0]=0;
    HeadDur[1]=5; //pau
    HeadBmp[1]=0;

    //break the word to phoneme
    for (i=0; i<length; i++)
    {
        //get the phoneme from location 0 to end
        tmpString = inputString.GetAt(i);
        //insert the empty space

        /*****
        /*      " " empty                                     */
        /*****
        if (((i+1)<length) && ((tmpString == " ") ||
        (tmpString == ",") || (tmpString == ".") ||
        (tmpString == "!") || (tmpString == "?")))
        {          //do nothing          }

        /*****
        /*      " gh "                                       */
        /*****
        else if ((i+1)<length && (tmpString == "g" ||
        tmpString == "G"))
        {
            totalPho++;
            nextString= inputString.GetAt(i+1);

            if (nextString == "h" || nextString == "H")
            {
                Pho2Head(tmpString + nextString);
                i++;
            }
            else
                Pho2Head(tmpString);
        }
    }
}
CONTINUE.....

```

**Figure 3.18** Part of codes of the '*Sent2Pho*' function

```

/*****
/*
/*  Pho2Head Function
/*
/*  Based on the parameter (phoneme) sent to this function,
/*  finds the compatible images.
/*
/*
/*****

void CTalkingHeadDlg::Pho2Head(CString inputPho)
{
//total phoneme: 33
//checking and access the duration
int i;
int tempDur=0, tempBmp=0;

for(i=0;i<33;i++)
{
    if (inputPho == mouth[i].sPho )
    {
        tempDur= mouth[i].time ;
        tempBmp= mouth[i].index ;
        HeadDur[totalPho] = HeadDur[totalPho-1] +tempDur;
        HeadBmp[totalPho] = tempBmp;
    }
}
UpdateData(FALSE);
}

```

**Figure 3.19** Codes of the 'Pho2Head' function

```

/*****
/*
/*   OnTimer Function
/*
/*
/*****
void CTalkingHeadDlg::OnTimer(UINT nIDEvent)
{
if(nIDEvent==10)
{
    nTIncrement++;
    UpdateData(FALSE);

    //first frame: is set to 0
    if (nTIncrement ==1)
    {
        m_iMouthBmp= 0;//pau
        InvalidateRect(m_cMouthRect, false);}

        //then increment by setting the things:
        //HeadDur is a array [100]
        if(nTIncrement< HeadDur[totalPho] && increPho < totalPho)
        {
            //here set the increment to increment the head pattern
            //use iSetTime to set the time for first phone and follow
            if (nTIncrement== HeadDur [increPho] )
            {
                m_iMouthBmp= HeadBmp [increPho];
                InvalidateRect(m_cMouthRect, false);
                increPho++;
            }
        }

        else if (nTIncrement == HeadDur[totalPho])
        {
            KillTimer(10);
            m_iMouthBmp= HeadBmp [increPho];
            InvalidateRect(m_cMouthRect, false);
            nTIncrement=0;
            UpdateData(FALSE);
        }
    }
}
}
}

```

**Figure 3.20** Codes of the ‘*OnTimer*’ function

### 3.2.6 Malay Talking Head module generation

After the Malay Text-driven Talking Head managed to develop, we should transform it into module so that can be easily applied into any other applications. Actually the codes are still the same, but we just delete all the GUI interfaces and their event handler, such as Edit box, buttons and etc, and only left the Picture control which validates the images. Then, add a new function, ‘*Voice(CString*

*Message*)' (Figure 5.13) which accept text as inputs and will process exactly as what Malay Talking Head done when speak button is clicked.

Then every time if want to import this Malay Talking Head, just need to: -

1. Copy and import the **MalayTalkingHead.h**, **MalayTalkingHead.cpp**, **na\_play.exe** and the **image resources** to your project's folder and add them to your application.
2. Then, create a dialog named **IDD\_TALKINGHEAD\_DIALOG** with only 1 picture control named **IDC\_MOUTH\_IMG** with size 167x116, so that the Talking Head images can be previewed.
3. Next, include your application header file, such as "ImportModule.h" to it.  

```
#include "ImportModule.h"
```
4. To make it Talking Head works for you, just create an object of this MalayTalkingHead class → CTalkingHeadMalay and use codes as showed below:

```
#include "TalkingHeadMalay.h"
CTalkingHeadMalay head;

head.m_strExpre="Happy";
head.Voice("Selamat berjaya");
```

\*\*The MalayTalkingHead.h and MalayTalkingHead.cpp are saved as digital copy!

Figure 5.14 shows a program (ImportModule) developed and imported this Malay Talking Head to it. Codes can be obtained from the digital copy.



```

void CTalkingHeadMalay::Voice(CString Message)
{
    //Call Speak function to generate the Wave file using
    //Festival
    Speak(Message);
    //Break the words into phoneme and get the compatible
    //duration and index (Pho2Head)
    Sent2Pho (Message);

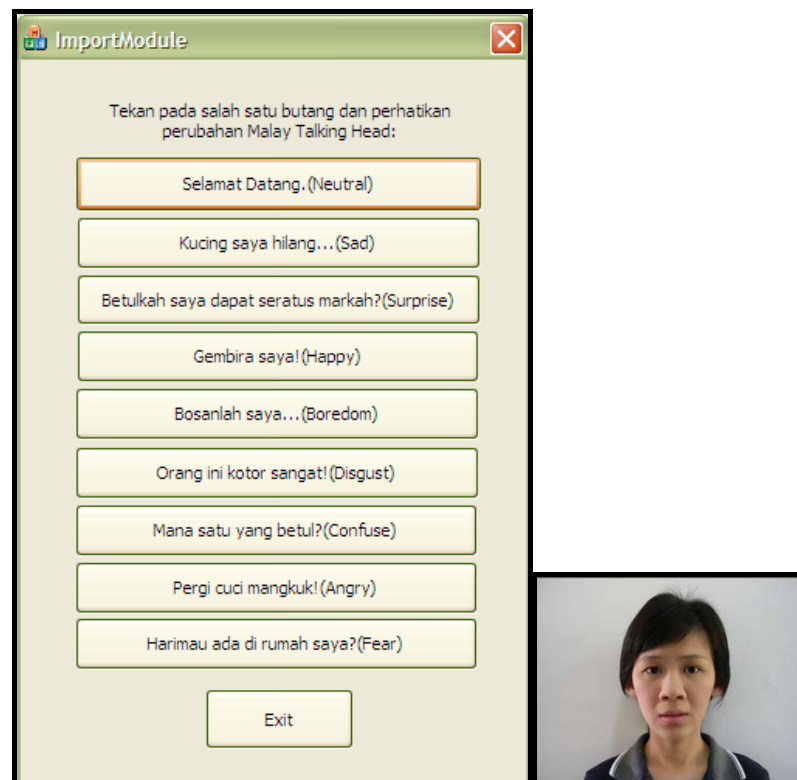
    nTIncrement=0;
    increPho=1;

    int duration;
    //Talking Head total display duration
    duration = HeadDur[totalPho];
    Sleep (duration*12);
    //Simply delay to wait for Festival to finish execution
    SetTimer(10,1,NULL);

    OnTimer(10);
    UpdateData(FALSE);
    WinExec( "\\na_play\\" Speech1.wav", SW_HIDE);
    UpdateData(FALSE);
}

```

**Figure3.21** Codes of 'Voice(CString Message)' function



**Figure 3.22** Example which imports the Malay Talking Head module

### 3.3 Experimental Evaluation

#### 3.3.1 Task & Database

The developed isolated word recognition system is evaluated on the task of multi speaker speaker-dependent (SD) and speaker-independent (SI) Malay isolated digit recognition. A training set consisting of 130 occurrences of each Malay digit, 0-9 by 26 speakers (i.e, 5 occurrences of each digit per speaker) was used. For multi speaker SD testing, separate set of 130 occurrences of each digit by the same 26 speakers used. For SI task, the testing set consists of 40 speakers (different from the speaker in the training phase), with each speaker recorded 2 token per digit. 12 order MFCC was used for feature extraction throughout. The pdf of CDHMM used was a mixture of 4 Gaussian densities. 5 state left-to-right word models were used.

#### 3.3.2 Experimental Results for Multi speaker SD Task.

Experiment has been carried out for speaker dependent isolated Malay digit recognition to investigate the difference between the DHMM and CDHMM recognizers in the Malay digit domain. Different algorithms were used in training the CDHMM to see its effect on recognition performance (recognition rate). The four isolated word recognizers to be evaluated are shown in Table 1:

Table 3.3: DHMM and CDHMM recognizers with different training algorithms.

Recognizers	Specification
VQ/DHMM/BW	DHMM recognizer with VQ (128 order codebook) and BW training algorithm.
CDHMM/BW	CDHMM recognizer with BW training algorithm.
CDHMM/VB	CDHMM recognizer with Viterbi training algorithm
CDHMM/VB+BW	CDHMM recognizer with combination of Viterbi & BW training algorithm

The recognition tests results are given in Table 2. The result shows that the recognition accuracy of the CDHMM recognizers with different training algorithm is

better than the DHMM recognizer. This is because the DHMM models speech signals that has been vector quantized which cause loss of information in the subsequent modeling. While CDHMM directly model the continuous acoustic space without VQ, this maintains the information to be modeled.

On the other hand, the result shows that not much improvement in recognition accuracy for CDHMM compared to DHMM recognizer. It may be due the ability of discrete model to approximate any density function. The discrete model is considerably sufficient to Table 2

Table 3.4: Comparison of DHMM and CDHMM recognizers with different training algorithm on SD task.

Type of recognizer	Word accuracy (%)
VQ/DHMM/BW	96.62
CDHMM/BW	99.00
CDHMM/VB	98.85
CDHMM/VB+BW	98.69

model the small acoustic variability in speaker dependent recognition in this study. Beside that, the considerably sufficient training token for each digit also assure the accurate and reliable estimate for discrete model. We expect in speaker independent recognition which impose large acoustic variability due to different speakers, the recognition accuracy improvement for CDHMM recognizer compared to its DHMM counterpart will be greater.

While the recognition accuracy of CDHMM recognizers with different training algorithm is comparable, this implies that both BW and VB result in well behaved solutions. But the CDHMM trained with BW algorithm yield slightly better recognition rate than CDHMM trained with Viterbi algorithm. This may be due to the simple average estimation of mean and covariance. But the Viterbi algorithm consumes less computational time and reduces numerical complexity compared to BW algorithm. The recognition rate of CDHMM recognizer trained with combination of VB and BW is the intermediate between CDHMM recognizer trained with BW and VB. This implies that combination of VB and BW yields no better performance than the single BW algorithm.

### 3.3.3 Experimental Results for SI Task.

Experiment has been carried out for speaker independent isolated Malay digit recognition to investigate the difference between the DHMM and CDHMM recognizers in the Malay digit domain. The CDHMM models were trained by Viterbi training. For DHMM, 128-order codebook was used and BW is used for model training.

Table 3 shows the comparison between CDHMM and DHMM in term of SI recognition accuracy. The CDHMM with mixture densities is outperform DHMM, with

Table 3.5: Comparison of DHMM and CDHMM recognizers for SI tasks.

Type of recognizer	Word accuracy (%)
VQ/DHMM/BW	78.63
CDHMM/VB	85.63

improvement of 8.17%. This is may be due that the Gaussian mixture densities of CDHMM is more capable of predicting large acoustic variability even though for the inter-speaker acoustic features that not in the training set. Whereas the distribution of DHMM is prone to over fit the training data and lack of generalization for acoustic feature for speakers which is out of training set.

## 3.4 Hardware design

In the previous section discuss about software and its application however in this section will discuss about the hardware design. These hardware designs are the combination of few hardware modules to form a complete system.

### 3.4.1 Wireless Telemetry system

This module is to build a wireless interconnection between security control center & remote sensor. Beside that, the stop and wait ARQ (Automatic Repeat Request) is implemented because this protocol uses the simple stop and wait acknowledgment (ACK) schemes. The monitoring system in this design is very simple. It is possible to identify each zone by scanning through address at each zone. And because of all the sensors are connected to a microcontroller in each zone, so the

sensors status will be send through wireless RF to PC/Server and display the sensors status at the working GUI. It is important to have a central or master control terminal that runs the main program and control and monitor the devices or sensor from each zone. Finally, the number or costs of installation for stand-alone zone controller can be varying according to the space or level of floor in a building or house.

The hardware consists of two parts, which are a central or master control terminal and slave control terminal. Develop a GUI using visual C++ program to monitor the sensors status and develop a wireless communication protocol between the master control terminal and slave control terminal. Besides that, in this design PIC (Microchip PIC 18F4550 with USB function) will be use to connect between the sensors and the RF module (transmitter and receiver) for master control terminal and slave control terminal.

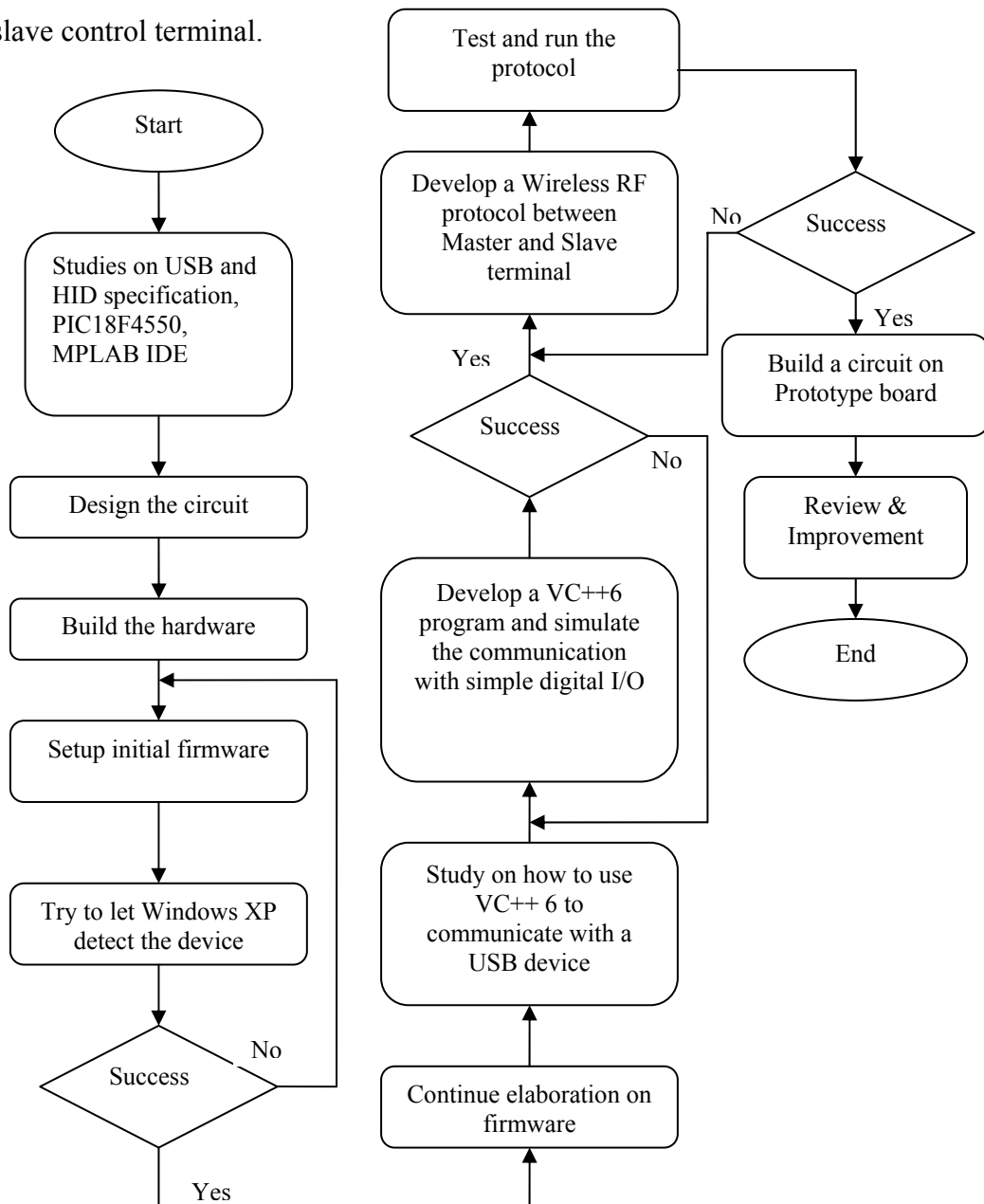


Figure 3.23 Project work flow

The hardware consists of three separate circuits. These three separate circuits are building on three different prototype boards. These circuits are master control terminal and two slave or zones control terminal. Each of these circuits containing a PIC18 F4550 microcontroller, seven LED's, two DIP switches, one Tx module, one Rx module, one USB connector and two voltage regulator +5v and +12v. The figure bellow is the block diagram of this project.

The hardware of monitoring sensor status consists of three separate circuit boards. One circuit board for master control terminal and two circuit boards for slave control terminals. Three of these separate circuit boards are based on PIC18F4550 microcontroller, a fully featured Universal Serial Bus communications that is compliant with the USB specification Revision 2.0.

The RF (Radio Frequency) link between the master control terminal and slave control terminal was design using a transmitter module and receiver module. Both of these modules are operate at 315MHz. The RF module comprises of a PT2272 controller along with an RF transmitter/receiver. The PT2272 is a remote control decoder that the author paired with PT2262.

The PT2262 is a remote control encoder that utilizes CMOS technology. It encodes data and address into a serial coded waveform suitable for RF modulation. The author choose these two chips because they have a maximum of 12 bits of tri-state address pins providing up to 531,441 (or  $3^{12}$ ) address codes; thereby, drastically reducing any code collision and unauthorized code scanning possibilities. Both are low power consuming chips that have very high noise immunity.

### 3.4.1.1 Communications between the master and slave control terminal

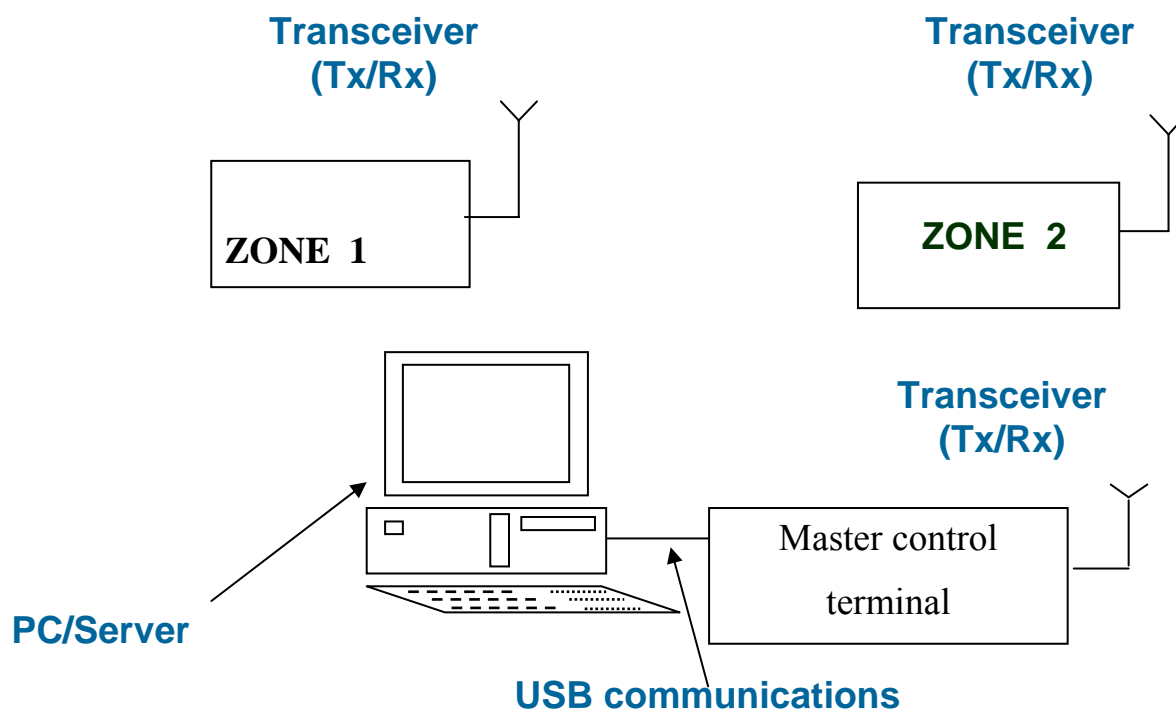


Figure 3.24: Communication between master and slave terminal



### 3.4.1.2 Master control terminal layout.

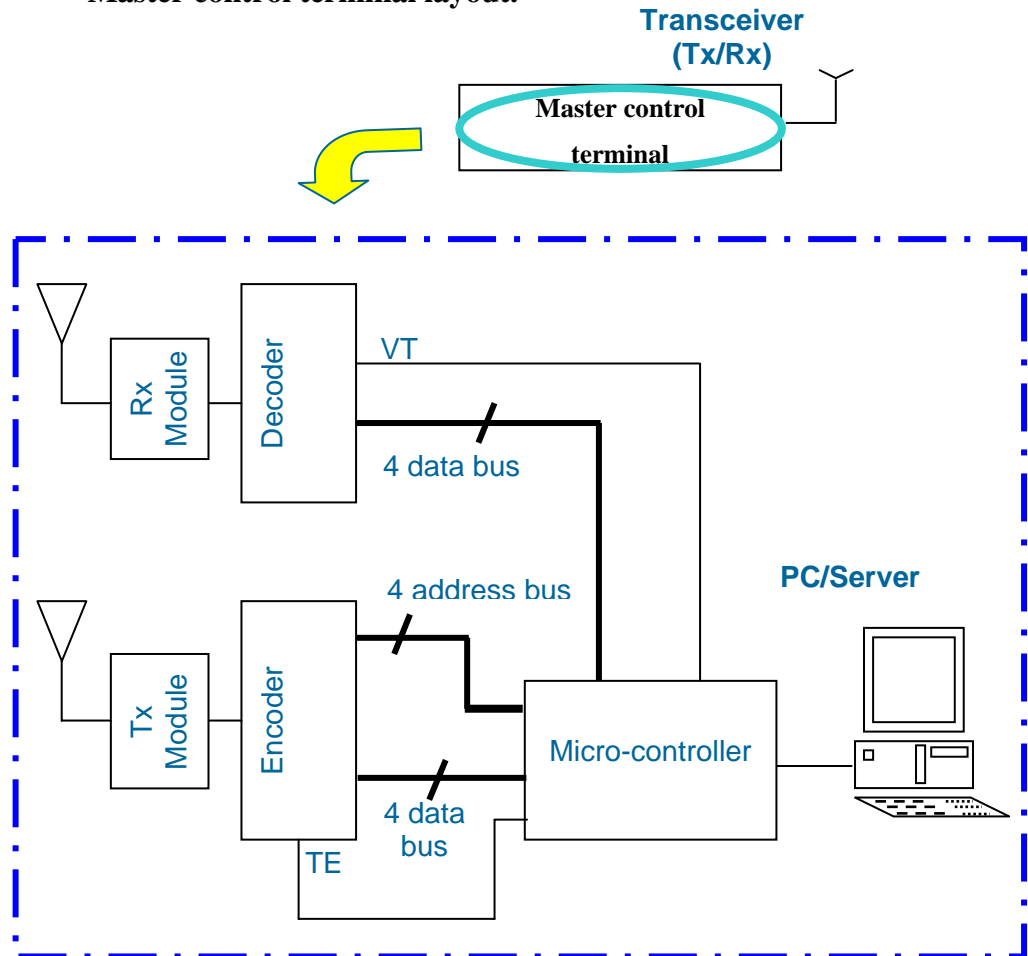


Figure 3.25: Internal functions of master control terminal

There are 5 combinations parts in the master control terminal (as shown in figure 3.27). First is a receiver module to receive signal that send from USB controller. Second is a transmitter module to transmit the acknowledgement signal to USB controller. Third is a decoder to decode the sending addressed or to recognize it own address for in a particular zone. Fourth is an encoder to encoder the sending signals with it own address. Fifth is a micro-controller to identify the received signal and sending signal from different zones. Finally, the microcontroller is connected to PC/Server through USB communication the sensors status at each zone will be display at the GUI.

### 3.4.1.3 Slave or zone control terminal

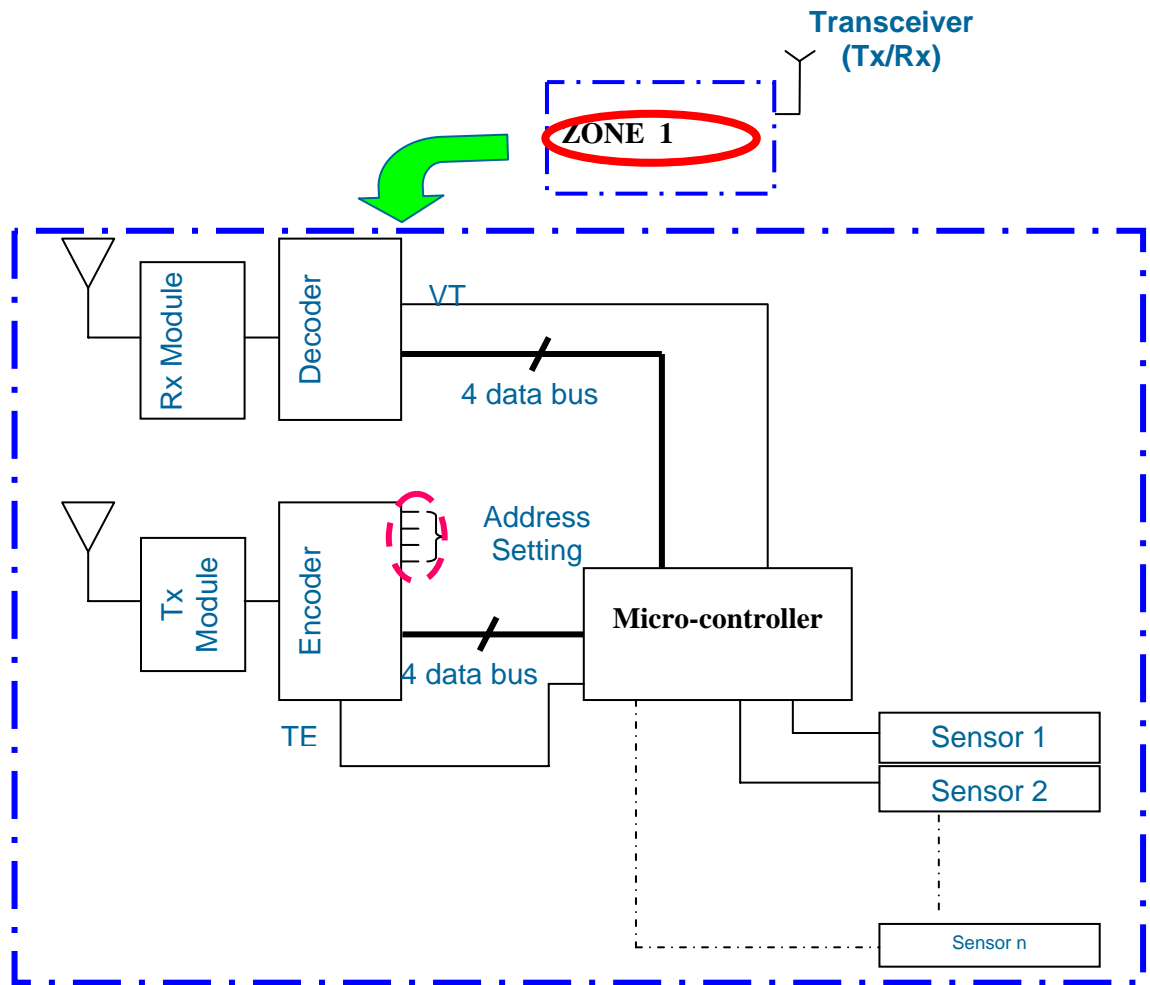


Figure 3.26: Block diagram of each zone

There are 5 combinations parts in the stand-alone zone circuit (as shown in Figure 3.28). First is the receiver module to receive signal that send from USB controller. Second is a transmitter module to transmit the acknowledgement signal to USB controller. Third is a decoder to decode the sending addressed or to recognize it own address for a particular zone. Fourth is an encoder to encoder the sending signals with it own address. Fifth is a micro-controller to identify the received and sending signal of the zone itself and also monitoring the sensors status in the zone itself.

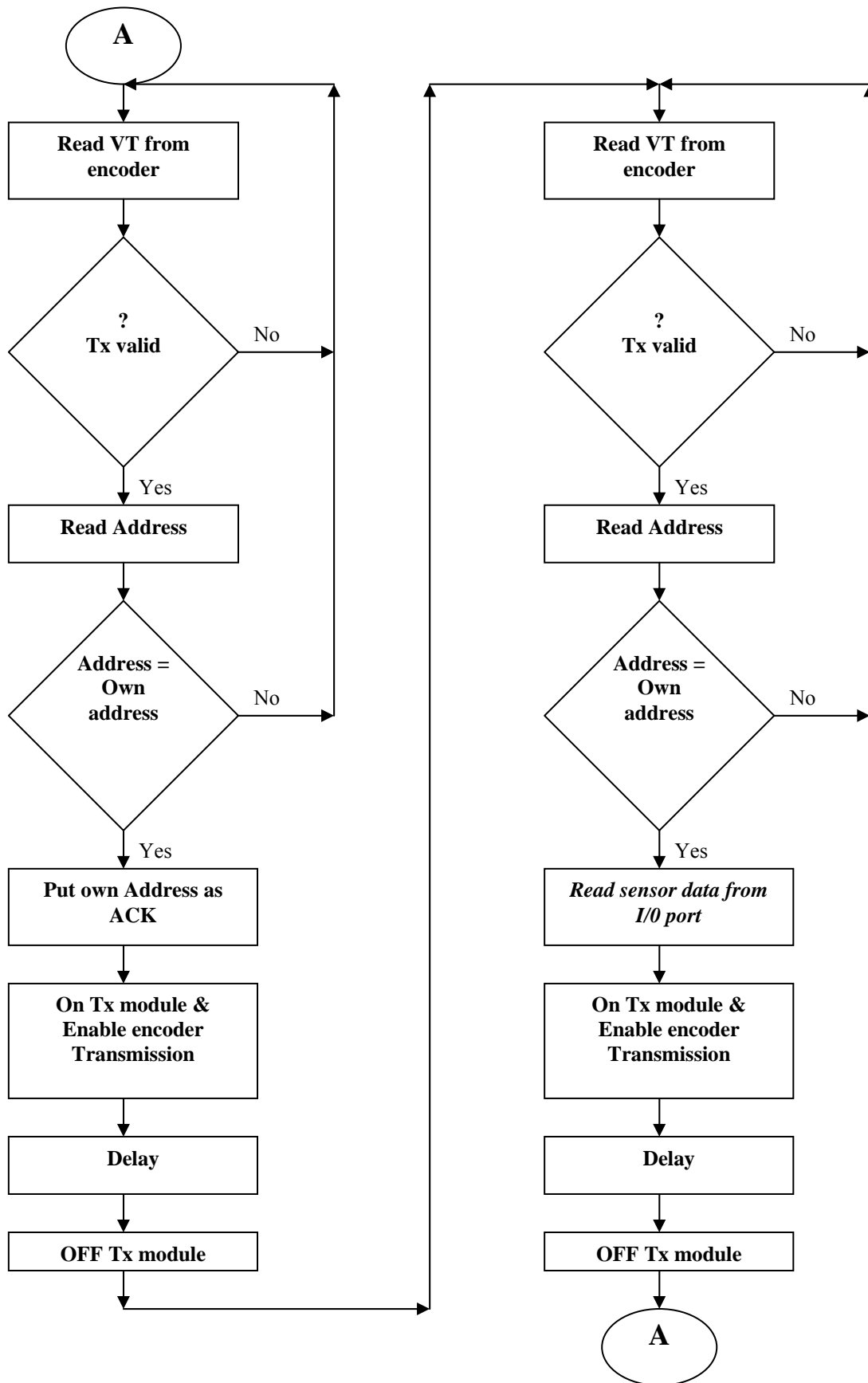


Figure 3.27: Flow diagram of slave or zone control terminal

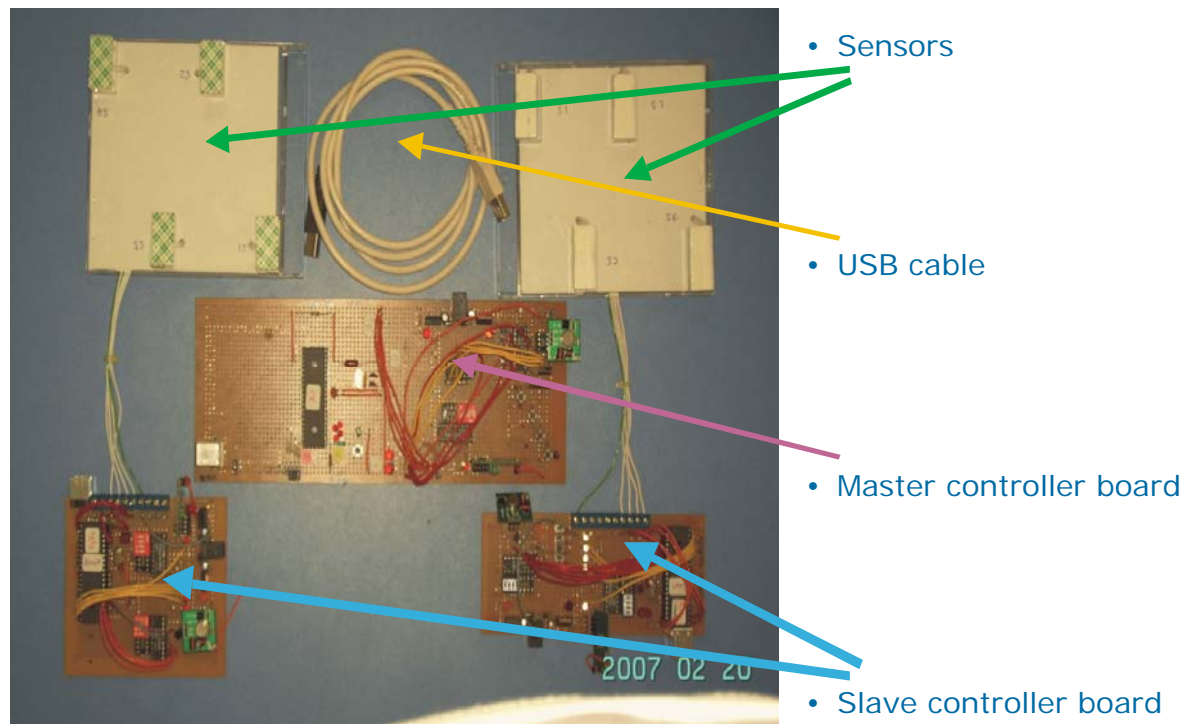


Figure 3.28: The prototype board for this project

The figure 3.30 shows the complete working prototype boards for this project. It consists of master controller board, slave controller board, USB cable and sensors.

#### 3.4.1.4 Maximum range test results

The testing range (estimated) for this project is as following: The distance is in diameter.

Open space: around 90 to 100 meter (outsides the Center Bio- Medical Engineering)

Close space: around 60 to 70 meter (insides the Center Bio-Medical Engineering)

### 3.4.2 EMBEDDED VIDEO CAMERA MOVEMENT CONTROLLER SYSTEM

This module is build to ease and comforts for all those who are using a video camera in their modern life style. The idea for this module has already existed and builds on their own way of techniques and skills. This module was developing to archive the objectives that have been set, which is allowing an extra feature in now day's video camera technologies. The results is suggested to be used with a current webcam or CCTV video camera for a personal use and the results will make a commonly video camera turns out to be an automatic tracking capture device. The process development approach is by a simple top-down solution like a waterfall model used in software engineering and using a microcontroller unit for process control, infrared pyroelectric sensors, and servo motors

#### 3.4.2.1 Design Process

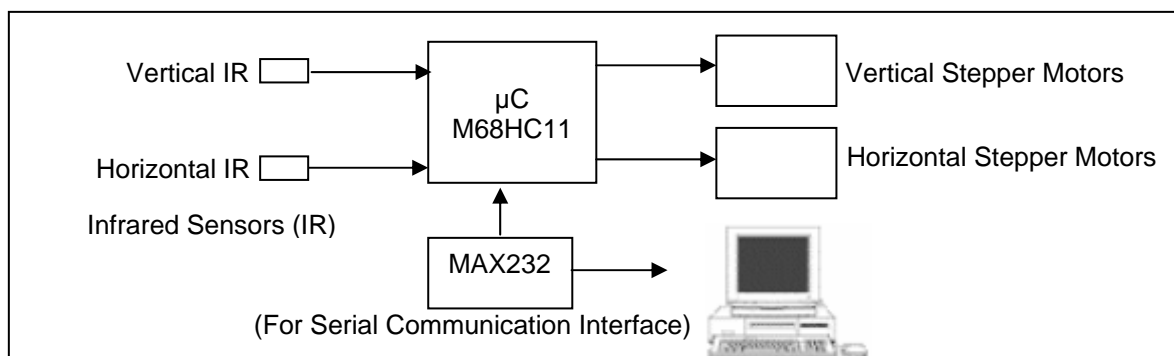


Figure 3.29: Block Diagram

### 3.4.2.2 Mechanical Design

Mechanical design is consist of positioning servo motor and joint forming a strong and simple stand for video camera. This design will make the stand move the mounted video camera in horizontal movement and vertical movement. Both movement in horizontal and vertical when combined shall make the mounted video camera on the stand act like a human eye because it will cover all view in front of it and however it would not be as flexible as human neck.

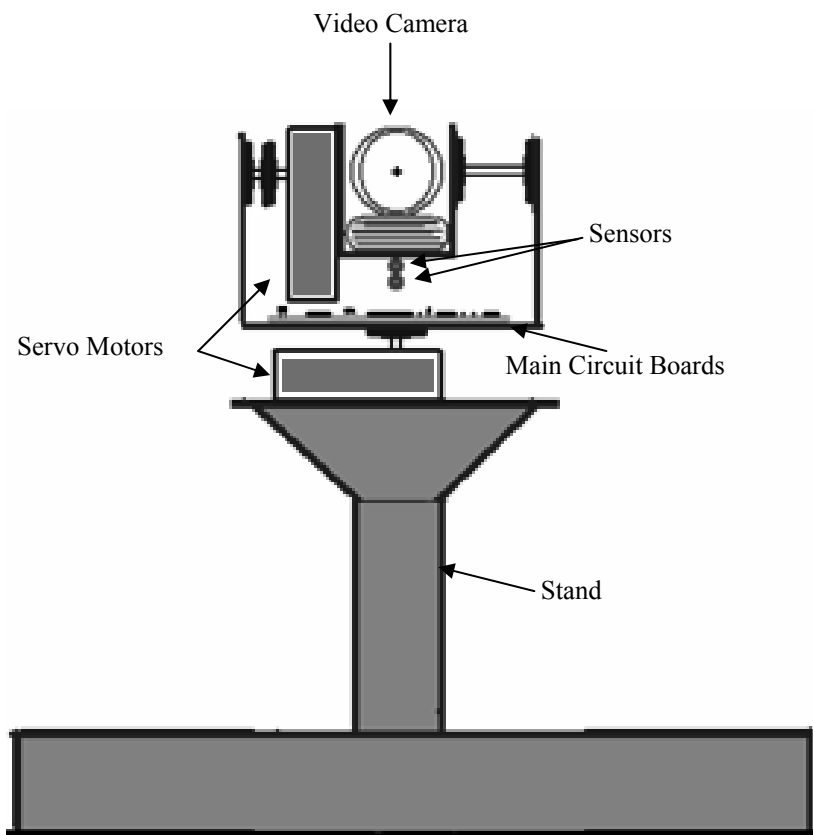


Figure 3.30: Mechanical Design

### 3.4.2.3 Microcontroller Circuit

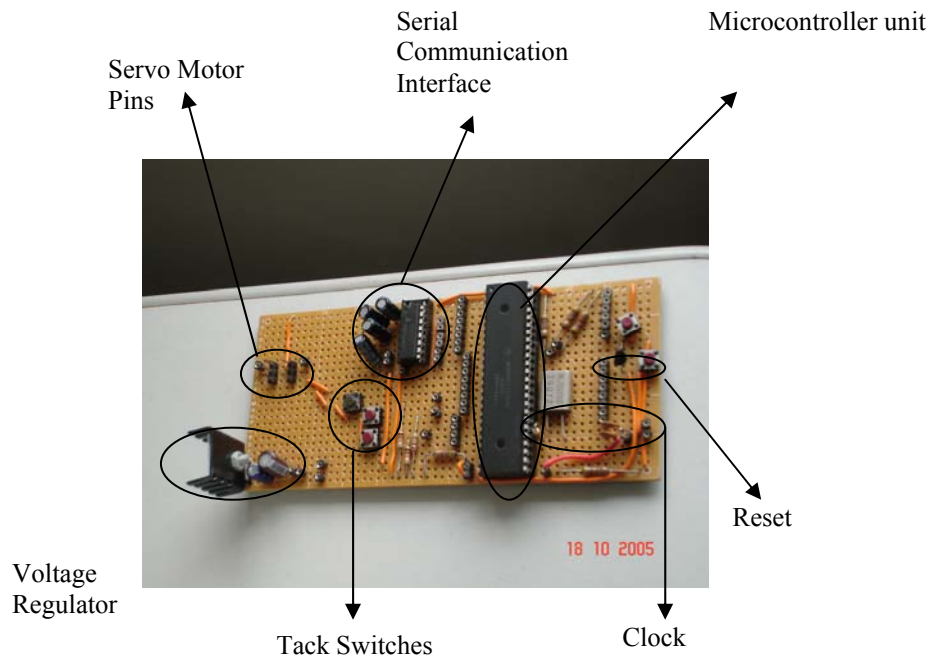


Figure 3.31: Microcontroller Circuit

### 3.4.2.4 Program flow chart

This project has three major components. There are three main components which are controller part, mechanical part, sensor part.

Last component is the controller part which is using M68HC11E1 the Motorola microcontroller. This microcontroller has been used widely in many applications as a control unit system. This microcontroller unit is selected by considering basic knowledge that already learned. The M68HC11 is an advance 8 bit microcontroller with significant ability and internal peripheral.

There are many advantages in using microcontroller for this project rather than using microprocessor. The microcontroller acts like microcomputer because all of basic component for microprocessor such as EEPROM, ROM, RAM, and analogue digital converter are already embedded in this system.

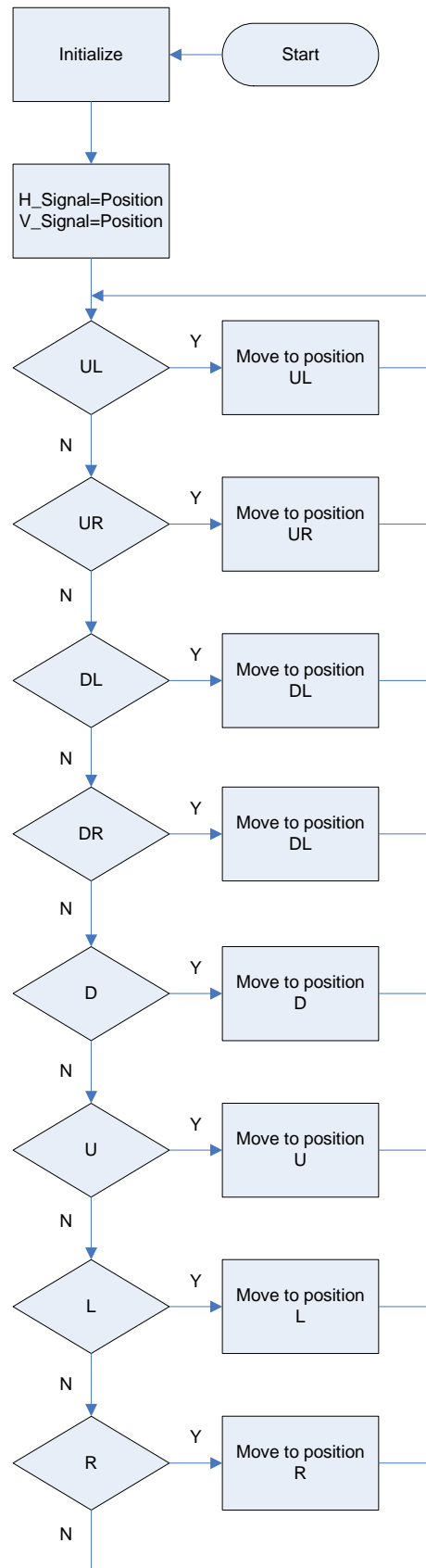


Figure 3.32: Control Program Flow Chart



### **3.4.3 USB Input Output Device**

This module is about a USB (Universal Serial Bus) device with parallel I/O and analog input. The USB device will be connected to the host system (PC) via the USB bus interface. This module was divided into three main subsystems- hardware, host system and firmware. The hardware of the USB device has been developed. It equipped with two type of input; analog to digital (ADC) input and digital input and the output are displayed through LED. The USB device uses PIC16C745 as it microcontroller. The PIC16C745 microcontroller is programmed by using the Hi- Lo system programmer to allow the host system (PC) to communicate with the analog and digital peripheral. The USB device with parallel I/O and analog input which has been developed can send the data to host system (PC) and display the output using the GUI.

#### **3.4.3.1 Design Process**

This module is done in 3 stages. The first stage is conducting preliminary studies on the relevant subjects regarding the project such as USB specification and functionality, USB protocol, PIC16C745/765 microcontroller, PIC programming language and so on. Upon completing this stage, the second stage is to build the hardware. Hardware connections are done and tested to check whether the circuits are correct. The final stage is to setup the firmware by using MPLAB IDE. When the source code is completed, it will be tested and run. Trouble- shooting and reprogramming are done to identify and solve the problem when the USB device is malfunction. The overall project flow chart is shown on figure 3.33.

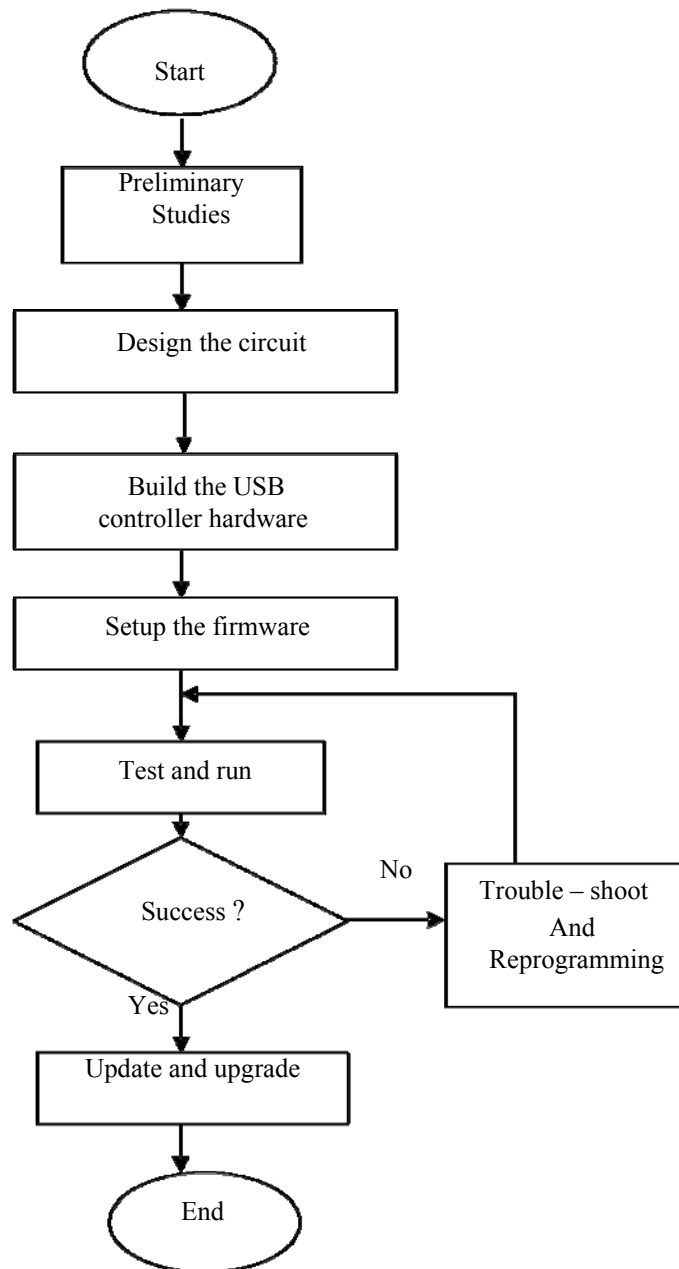


Figure 3.33: Project flow chart

### 3.4.3.2 Graphical User Interface (GUI)

The USB device with parallel I/O and analog input can use to send the data to host system (PC) and display the output using the GUI. It can also receive the data from host system. There are two control states for the host system; host control and board control. For the host control state, the USB device will receive the data from host system and display the output on the respectively I/O.

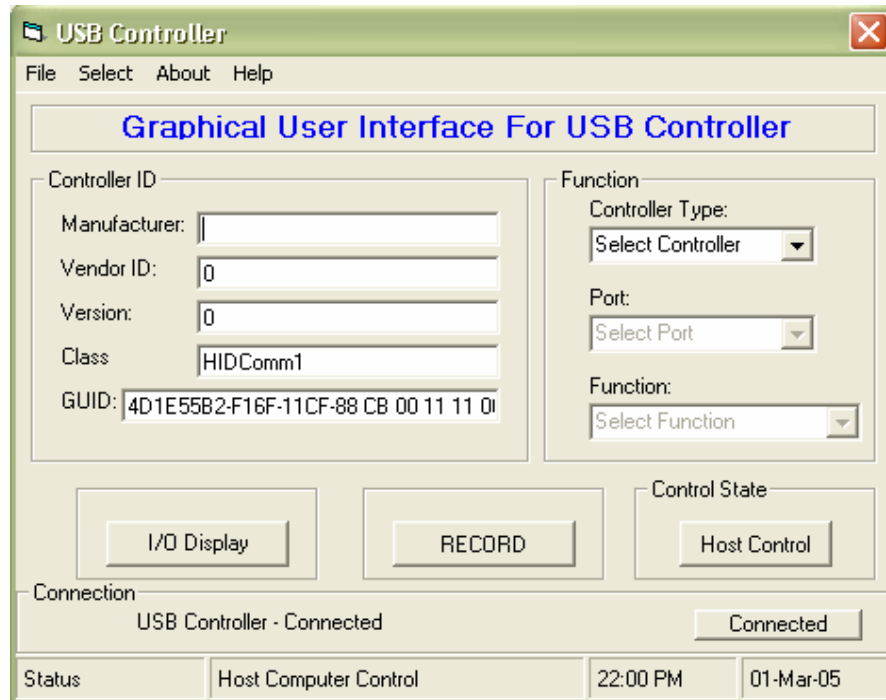


Figure 3.34: Graphical user interface

### 3.4.4 Ultrasonic Water Level Detector

This module is to develop an Ultrasonic Tank Water Level Detector with a high accuracy of water level reading in real-time mode. The ready made ultrasonic detectors have always faced the problem of inaccurate water level reading and the inability of addressing various critical factors that affect the ultrasonic waves. These problems can be solved by using the response delay value and error conversion data in PIC programming. The distance from ultrasonic sensor to water surface can be obtained by measuring the time interval between the transmitted and received signal. Then, this time interval is converted to a distance measurement using the speed of sound in air calculation. The level of the water tank can be measured by subtracting this distance from the exact height of the tank and displayed on the 7 segment LED. In the end of the project, an ultrasonic tank water level detector with a high precision of water level reading was successfully created.

#### 3.4.4.1 Ultrasonic Water Level Measurement

Ultrasonic water level measurement is indirect measurement that involves conversion the measurement of other physical quantity into level quantity. It works on the principle of sending a sound wave from a peizo-electric transducer to the contents of the tank.

Ultrasonic sensors transmit a series of cone shape sound waves through the air. These sounds pulses reflected off the water surface and are in turn received by the sensor, which measures the time interval between the transmitted and received signal. Then, the microcontroller converts this time interval into a distance measurement using the speed of sound in air. The level of the water tank can be measured by subtracting this distance from an exact height of the tank. Its successful measurement depends on reflection from process material in a straight line back to transducer. Figure 3.35 show ultrasonic water level measurement model. The speed of sound in air varies with ambient temperature. The formula is:

$$v = 331.4 + 0.6 * T_c \quad \text{where} \quad T_c \text{ in degree Celsius } (^{\circ}\text{C})$$

$v = \text{speed of sound in air}$

So, when the air temperature goes high, the speed of sound also goes high. It could cause wrong water level reading because the formula below is use to calculate the distance of ultrasonic sensor to water surface:

$$d = v * t \quad \text{where}$$

$v$  = speed of sound in air

$t$  = time interval between transmitting  
and receiving

$d$  = distance of ultrasonic sensor to water surface

Therefore, ultrasonic water level measurement must use the conversion data to composite the change of air temperature continuously.

To determine water level (h):

$$h = H - d \quad \text{where} \quad \begin{array}{l} H = \text{tank height} \\ h = \text{water level} \end{array}$$

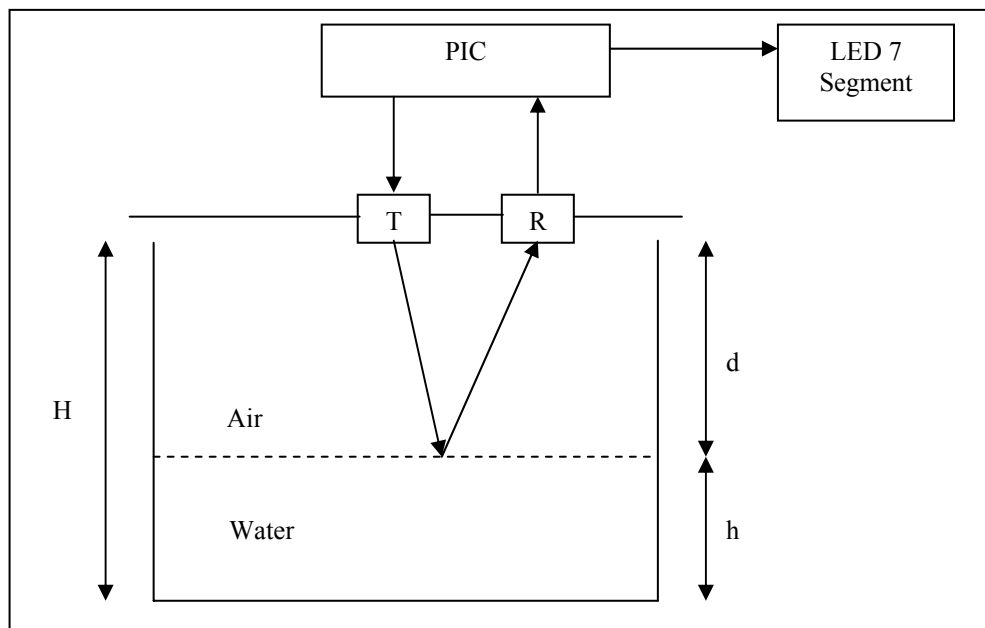


Figure 3.35: Ultrasonic Water Level Measurement Model

### 3.4.4.2 Program flow Chart

This ultrasonic water level detector detects a reflected wave from the object after sending out an ultrasonic pulse. By measuring the time which returns after emitting a sound wave, a distance to the water surface is measured and the water level can be obtained by subtract the actual tank high with the measure distance. Ultrasonic Water Level Detector flow chart is shown in Figure 3.36.

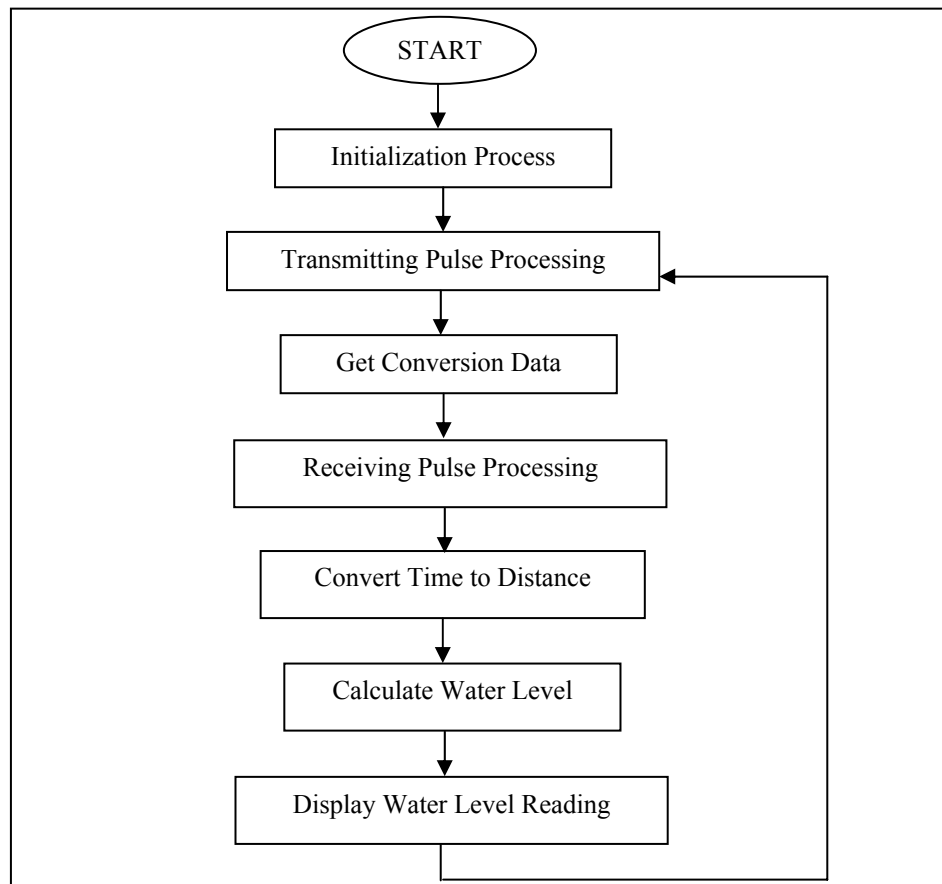


Figure 3.36: Ultrasonic Water Level Detector Flow Chart

### 3.4.4.3 Measuring Process Test

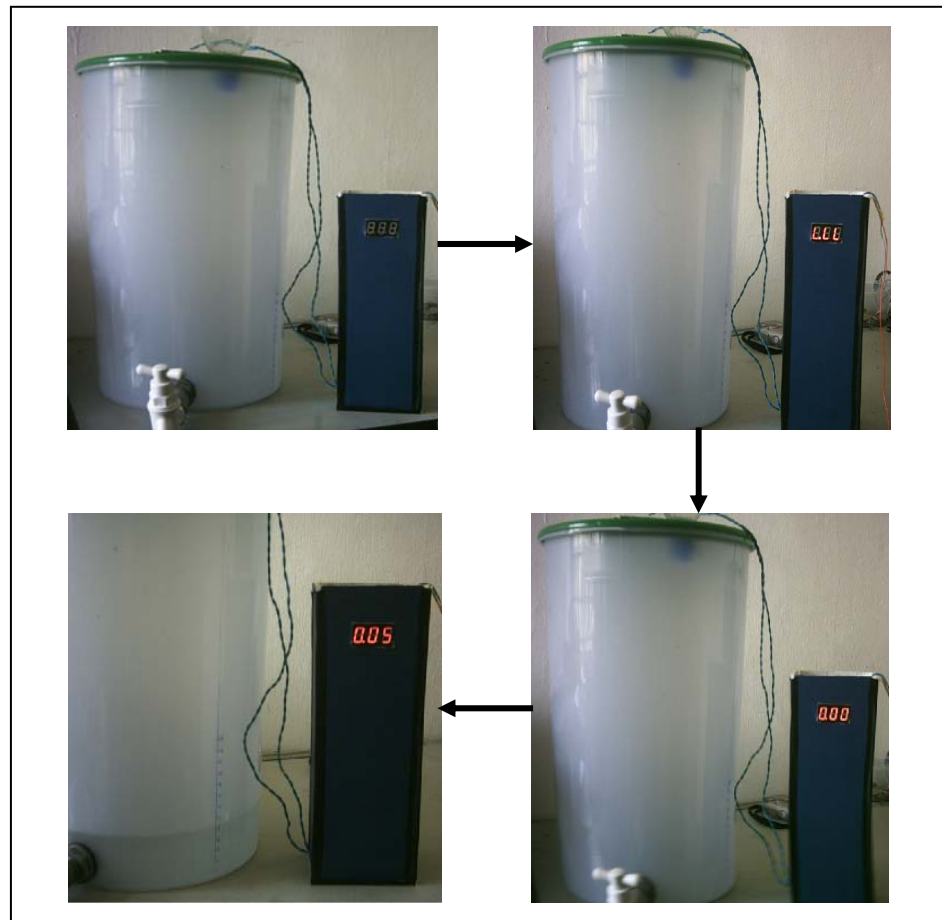


Figure 3.37: Measuring Process Flow

The measuring process is conducted in order to prove the efficiency of this device. It shows in Figure 5.2 above. In the first picture, the device is set up with a replica tank. Then, the power is turned ON. Picture 2 shows the device finishing the initializing process with displaying **L . L . L** and measuring the tank level. After that, the device starts to measure the current water level. In Picture 3, the display shows (0.00m) **0 . 0 0** reading which means that there is no water in the tank. Finally, when the water is filling the tank, the display reading is changing in real-time mode. This process is executed repeatedly. It is shown in Picture 4.

## **Chapter 4**

### **Conclusion**

#### **4.1 Conclusion**

The Malay isolated word recognition system has been described. In the domain of speaker-dependent Malay isolated digit recognition, the result shows that the recognition accuracy of the CDHMM with Gaussian mixture density is measurably higher than that of the DHMM, with the best recognition accuracy of 99% for CDHMM/BW. In speaker independent recognition, accuracy of 85.63% is obtained using CDHMM/VB, Besides, in SI task which imposes large acoustic variability due to different speakers, the accuracy improvement for CDHMM recognizer compared to its DHMM counterpart is greater, which is 8.17%. This supports that the CDHMM able to eliminate the quantization problem of DHMM and thus increase the recognition accuracy. The BW and VB algorithm of CDHMM training is both perform comparably well but the latter less computational time and reduce numerical difficulty.

Future work will extend the Malay speech recognition research from isolated word small vocabulary to large vocabulary continuous speech recognition system. For large vocabulary system, sub-word unit model such as phoneme will be used because of the large occurrences of phoneme in the training set enable the reliable model estimation. The sub-word models will be concatenated together to form a word model. Language modeling will also be incorporated to impose constraint to the word transitions and limit the search to the most probable word sequences. The large vocabulary system will enable a larger command words to the smart house, besides user can give commands in a spoken sentences in a continuous form.

Robustness of the system is another important issue when the system implemented in the real time situation. Noise and channel mismatch will cause degradation in recognition performance. Methods have to be investigated to compensate



the channel mismatch. Another application of speech technology into the smart house is the speaker recognition as a biometric solution to enhance the home security.

In hardware design, there are few modules that combine to form a complete system. Each module are function according to it need and the combination of all these modules are control by one developed graphic user interface. The complete system is depending on the performance of each module that integrated to one another.