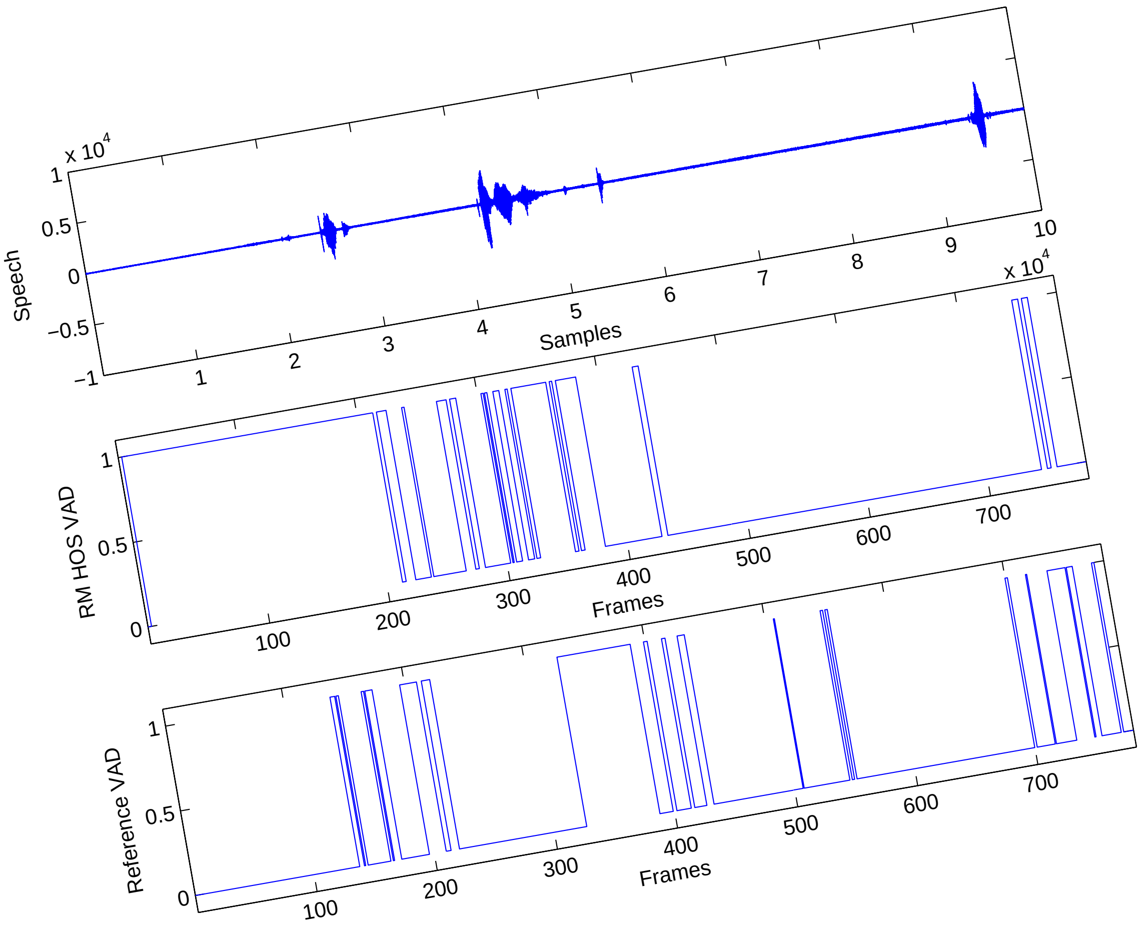
Robust Voice Activity Detection and Noise Reduction Mechanism Using Higher-Order Statistics



Aalborg University
Institute of Electronic Systems
Department of Communication Technology

Project Group 841, 2005

Aalborg University

Institute of Electronic Systems

Fredrik Bajers Vej 7 ▪ DK-9220 Aalborg East Phone +45 96 35 87 00

Title:	Robust Voice Activity Detector and Noise Reduction Mechanism Using Higher-order Statistics
Theme:	Department of Communications Technology
Project Period:	Feburary 2005 to June 2005

Project Group:

841

Group Members:

Michael Yaw Appiah

Raimonda Makrickaite

Milda Gusaite

Sasikanth Munagala

Supervisors:

Per Rubak

Ole Wolf

Publications:	4
Pages:	69
Supplement:	CD-ROM
Finished:	May 2005

Abstract:
This contribution presents a robust algorithm for voice activity detection (VAD) and noise reduction mechanism using combined properties of higher-order statistics (HOS) and an efficient algorithm to estimate the instantaneous Signal-to-Noise Ratio (SNR) of speech signal in a background of acoustic noise. The Rainer Martin's algorithm with HOS is capable of robustly tracking non stationary noise signal. The flat spectral feature of Linear Prediction Coding (LPC) residual results in distinct characteristics for the cumulants in terms of phase, periodicity and harmonic content and yields closed-form expressions for the skewness and kurtosis. The HOS of speech is immune to Gaussian noise and this makes them particularly useful in algorithms designed for low SNR environments. The proposed algorithm uses HOS and smooth power estimate metrics with second-order measures, such as SNR and LPC prediction error, to identify speech and noise frames. A voicing condition for speech frames is derived based on the relation between the skewness, kurtosis of voiced speech and estimate of smooth noise power. The algorithmis presented and its performance is compared to HOS-only based VAD algorithm. The results show that the proposed algorithm has an overall better performance than HOS only, with noticeable improvement in Gaussian-like noises, such as street and garage, and high to low SNR, especially for probability of correctly detecting speech. The proposed algorithm is replicated on DSK C6713.

Preface

This document reports on the work of group 841 in the 8th semester. This report is organized into five chapters. The first chapter provides the introduction, the motivation and the scope of the project. Chapter 2 focuses on the problem analysis; the working of VAD; the different noises and their effect. The last part of the chapter mentions the problem statement. In chapter 3, the algorithms are described i.e., the HOS-VAD algorithm and the Rainer Martin's algorithm for the estimation of SNR. Chapter 4 discusses the implementation of the algorithms on Matlab its conversion to C code and later the implementation on the DSK(DSP Starter Kit). Finally, chapter 5 provides the conclusion and recommendations of the project. All the associated codes can be found in the companion CD. Several appendices were provided at the end of the report as references.

Michael Yaw Appiah

Raimonda Makrickaite

Milda Gusaite

Sasikanth Munagala

Aalborg, June 2005

Acknowledgements

It is not possible for us to list by name of all who helped with useful suggestions; we are grateful to them all. In this project, some lecturers and colleagues have been particularly generous in providing us with ideas, comments, suggestions, and materials.

We especially want to recognize the efforts of our supervisors, Mr. Per Rubak, and Ole Wolf, who influenced many of our ideas. We highly appreciate their generosity in helping with this project.

We commend and acknowledge Aalborg University for creating a very challenging environment for us to do this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In speech communications, noise is fluctuations in and the addition of external factors to the stream of target information (signal) being received at a detector. It may be deliberate as for instance jamming of a radio or video signal, but in most cases it is assumed to be merely undesired interference with intended operations. Many speech processing system users are familiar with the amount of background noise present in loud environments. This is because their hands free instruments amplify environment noise just as much as the conversation that they are trying to follow. Work is ongoing to suppress background noise as much as possible to positively influence the intelligibility of the speech in noisy environments.

Although speech processing in artificially constrained conditions has recently reached high levels of performance, problems still remain in the deployment of speech recognition technology in the real world. One of the problems is the performance degradation of speech detection when they are used in noisy environments such as offices, automobile cabins, streets, and computer rooms. Many reasons account to eliminate or reduce noise from speech signals. However one of the biggest challenges is to avoid removal of speech components in this process. An approach have been considered for robust speech detection in this project.

To develop effective robust speech recognition method, noisy speech uttered in the real world is required and the speech database should contain every possible distortion which could occur in noisy environments. But it is not feasible to collect speech data in various noisy environments. Speech or Voice Activity Detector (VAD), aims to distinguish between speech and several types of acoustic background noise even with low signal-to-noise ratios(SNRs).

In the field of multimedia applications, a VAD permits simultaneous voice and data applications. Similarly, in Universal Mobile Telecommunications Systems (UMTS) [2], it controls and reduces the average bit rate and enhances overall coding quality of speech.

In cellular radio systems (for instance GSM and CDMA systems) based on Discontinuous Transmission (DTX) mode, this facility is essential for enhancing system capacity by reducing co-channel interference and power consumption in portable digital devices [3], [4], [5].

It is very difficult to distinguish between noise and silence, in the presence of background noise, so more efficient and self-sustaining algorithms are needed for speech activity detection and noise reduction in a changing and adverse noise acoustic background. There are different metrics used for speech detection in VAD algorithms, but recently Higher-order statistics (HOS) have shown potential results in a number of signal processing applications, and are of particular value when dealing with a mixture of Gaussian and non-Gaussian processes and system with non-linearity [1].

1.1 Motivation of Project

The project is motivated by the fact that, combination of HOS and an algorithm proposed in [6][7], yields a better, efficient and robust VAD.

The application of Rainer Martin's algorithm with HOS to speech processing and specifically to VAD is primarily triggered by:

1. Observation that the smoothed power estimate of a noisy speech signal exhibits distinct peaks and valleys that is capable of tracking varying noise level during speech activity. Work in this area[1] is based on the idea that peak correspond to speech activity the valleys of smoothed noise is used to obtain the noise power estimates.
2. The algortihm's inherent suppression of additive coloured Gaussian noise and phase preservation properties. It is based on the assumptions that speech has certain HOS properties that are distinct from those of Gaussian noise.

Finally, the implementation and verification of the algorithm using Texas Instruments TMS320C6713 DSP Kit *(DSK)*, is itself a motivation for pursuing the project.

1.2 Scope of the Project

The first part of the project involves analyzing the characteristics of the third and fourth-order cumulants of the LPC residual of speech signals. The flat spectral envelope of this residual results in distinct characteristics for these cumulants in terms of phase, periodicity and harmonic content and yields closed-form expressions for the skewness

[1]Rainer Martin's algorithm

and kurtosis based on harmonic speech model.

The proposed algorithm is tested on variety of noise types like the noise present in the street, car, garage, train at different SNR levels and the performance is compared to the HOS VAD. To quantify performance, the probability of correctly classifying speech and noise frames as well as the probability of false classification are computed by making references to truth marker files in clean speech conditions.

To compute these metrics and generate the noisy speech test cases, a proposed TIA database material (mentioned in E) is used for the evaluation of VAD algorithms.

The second part of the project involves running the two combined efficient algorithms using Texas Instruments (TI) Code Composer Studio(CCS) and then implement the corresponding C program onto the TMS320C6713 DSP Starter Kit (DSK).

Chapter 2

Problem Analysis

This chapter explores and dissects the question to be considered, solved, or answered in this project. How is additive noise (in the form of gaussian noise) corrupted with clean speech suppressed or isolated? This is identified as the main question to be explained in this chapter.

2.1 Voice Activity Detection

2.1.1 Overview

The process of separating conversational speech and silence is called the voice activity detection (VAD). It was first investigated for use on Time Assigned Speech Interpolation (TASI) systems. VAD is an important enabling technology for a variety of speech-based applications including speech recognition, speech encoding, and hands-free telephony. For these purposes, various types of VAD algorithms were proposed that trade off delay, sensitivity, accuracy and computational cost.

The primary function of a voice activity detector is to provide an indication of speech presence in order to facilitate speech processing as well as possibly provide delimiters for the beginning and end of a speech segment [11]. For a wide range of applications such as digital mobile radio, Digital Simultaneous Voice and Data (DSVD) or speech storage, it is desirable to provide a discontinuous transmission of speech-coding parameters. The advantage can be a lower average power consumption in mobile handsets, or a higher average bit rate for simultaneous services like data transmission or even a higher capacity on storage chips. However, the improvement depends mainly on the percentage of pauses during speech and the reliability of the VAD used to detect these intervals. On one hand, it is advantageous to have a low percentage of speech activity but, on the other hand, clipping of active speech should be avoided to preserve the quality. This is a crucial problem for a VAD algorithm under heavy noise conditions [12].

Voice activity detection is important for speech transmission, enhancement and recognition. The variety and the varying nature of speech and background noise makes it challenging [13]. Earlier algorithms for VAD are based on the Itakura LPC distance measure, energy levels, timing, pitch, and zero crossing rates, cepstral features, adaptive noise modeling of voice signals and the periodicity measure. Unfortunately, these algorithms have some problems for low SNR values, especially when the noise is non-stationary. Consistent accuracy cannot be achieved since most algorithms rely on a threshold level for comparison. This threshold level is often assumed to be fixed or calculated in the silence (voice-inactive) intervals [18]. During the last decade numerous researchers have studied different strategies for detecting speech in noise and the influence of the VAD decision on speech processing systems [19].

2.1.2 VAD Algorithm: The Principle

The basic function of a VAD algorithm is to extract some measured features or quantities from the input signal and to compare these values with thresholds, usually extracted from the characteristics of the noise and speech signals. Voice-active decision is made if the measured values exceed the thresholds. VAD in non-stationary noise requires a time-varying threshold value. This value is usually calculated in the voice-inactive segments [18].

A representative set of recently published VAD methods formulates the decision rule on a frame by frame basis using instantaneous measures between speech and noise [19]. The different measures which are used in VAD methods include spectral slope, correlation coefficients, log likelihood ratio, cepstral, weighted cepstral, and modified distance measures.

A VAD can be decomposed in two steps: the computation of metrics and the application of a classification rule. Independently from the VAD method, the operation is a compromise between having voice detected as noise or noise detected as voice [13]. A VAD operating in a mobile environment must be able to detect speech in the presence of a range of very diverse types of acoustic background noises. In these difficult detection conditions it is vital that a VAD should "fail-safe", indicating "speech detected" when the decision is in doubt so that no clipping is introduced. The biggest difficulty in the detection of speech in this environment is the very low signal-to-noise ratios (SNRs) that are encountered. It is impossible to distinguish between speech and noise using simple level detection techniques when parts of the speech utterance are buried below the noise [20].

Robust voice activity detection algorithms are required, as traditional solutions present a high misclassification rate in the presence of the background noise typical of mobile environments. One important aspect of recent digital cellular systems is the robustness of the speech coding algorithms needed for the channel to be used efficiently. They have to be robust, not only to channel degradation, but also to the background noise typical of mobile environments [21]. The underlying definition of the robustness can be formulated as a *"VAD is robust if it gives decisions close to a reference in quiet as well as in adverse environments"*. There is introduced a new definition claiming that a VAD is robust when it gives similar decisions for clean speech and noisy speech. The robustness can be estimated by taking the VAD's decision on clean speech as a reference and computing error statistics of the same VAD applied on noisy speech. The more robust the VAD, the scarcer the errors [13].

2.1.3 VAD Evaluation

Performance of VAD can be measured in terms of activity and the degree and severity of clipping. In order to evaluate the amount of clipping and how often noise is detected as speech, the VAD output is compared with those of an ideal VAD. The performance of a VAD is evaluated on the basis of the following four traditional parameters [20]:

1. FEC (Front End Clipping): clipping introduced in passing from noise to speech activity;
2. MSC (Mid Speech C1ipping): clipping due to speech misclassified as noise;
3. OVER: noise interpreted as speech due to the VAD flag remaining active in passing from speech activity to noise;
4. NDS (Noise Detected as Speech): noise interpreted as speech within a silence period.

Although the method described above provides useful objective information concerning the performance of a VAD, it only gives an initial estimate with regard to the subjective effect. It is therefore important to carry out subjective tests on the VAD, the main aim of which is to ensure that the clipping perceived is acceptable. This kind of test requires a certain number of listeners to judge recordings containing the processing results of the VAD's being tested. The listeners have to give marks on the following features:

1. Quality.
2. Comprehension difficulty.
3. Audibility of clipping.

These marks, obtained by listening to several speech sequences, are then used to calculate average results for each of the features listed above, thus providing a global estimate of the behavior of the VAD being tested. To conclude, whereas objective methods are very useful in an initial stage to evaluate the quality of a VAD, subjective methods are more significant. As, however, they are more expensive (since they require the participation of a certain number of people for a few days), they are generally only used when a proposal is about to be standardized [21].

One of the primary reason for the use of HOS VAD is to suppress colored noise. The following section describes briefly about various noises.

2.2 Noise

Noise can be defined as the contamination of the desired signal or the unwanted signal. Natural and deliberate noise sources can provide both or either of random interference or patterned interference. Only the latter can be cancelled effectively in analog systems; however, digital systems are usually constructed in such a way that their quantized signals can be reconstructed perfectly, as long as the noise level remains below a defined maximum, which varies from application to application. There are many forms of noise with various frequency characteristics that are classified by "color" [25].

White noise is a signal (or process) with a flat frequency spectrum. In other words, the signal has equal power in any band, at any centre frequency, having a given bandwidth. In practice a signal can be "white" with a flat spectrum over a defined frequency band. A signal that is "white" in the frequency domain must have certain important statistical properties in time. For example, it must have zero autocorrelation with itself over time, except at zero timeshift. The figures (2.2), (2.2) shows that car noise taken for 10000 samples is not white. The periodogram shows that the spectrum is not uniform where as the randomly generated Gaussian noise has a uniform distribution. The power spectral density is the smoothed version of the periodogram.

Noise having a continuous distribution, such as a normal distribution, can be white [26]. Gaussian noise is sometimes misunderstood to be white gaussian noise, but this is not so. Gaussian noise only means noise with pdf[1] of the Gaussian distribution, which says nothing to correlation of the noise in time. Labeling Gaussian noise as white describes the correlation of the noise.

The next most commonly used colored noise is *pink noise*. Its frequency spectrum is not flat, but has equal power in bands that are proportionally wide. Pink noise is perceptually white. That is, the human auditory system perceives approximately equal

[1]Probability Distribution Function

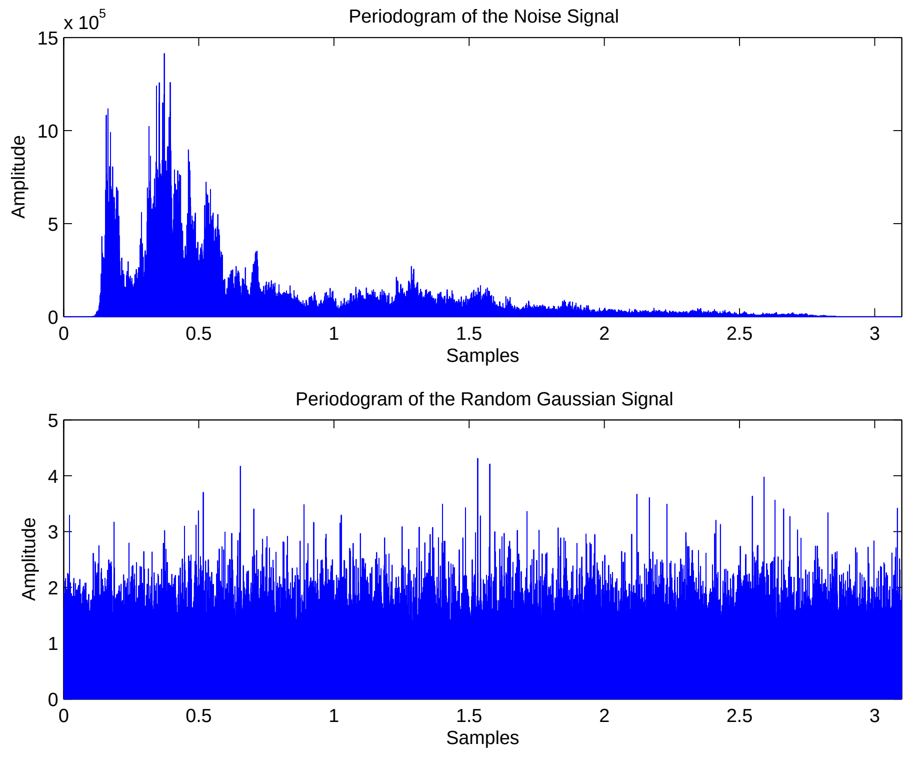


Figure 2.1: Test for Whiteness of Noise in the CAR Noise

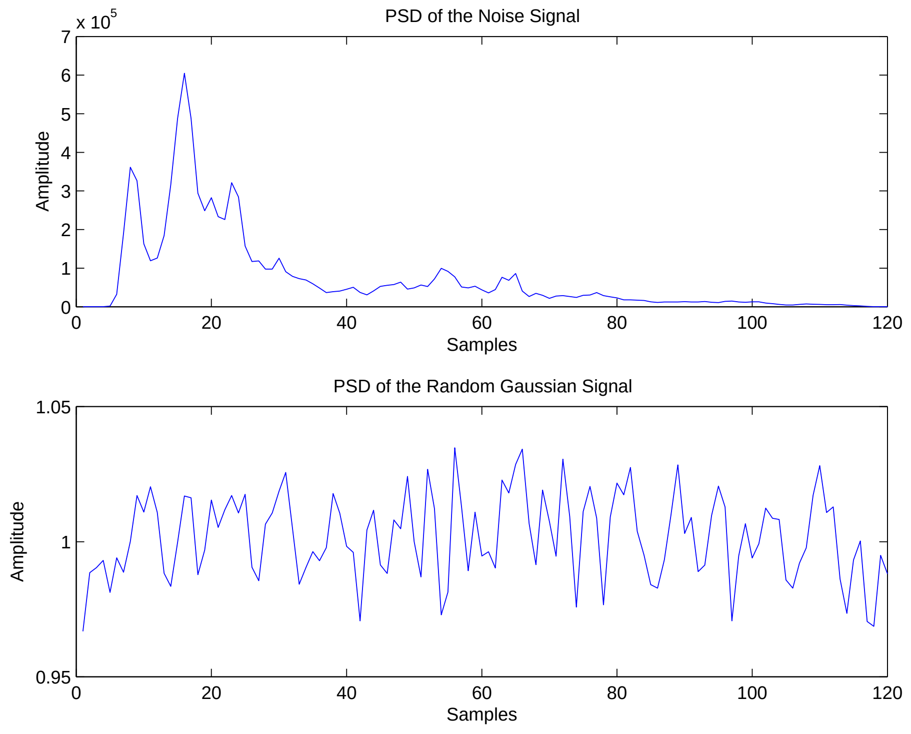


Figure 2.2: Test for Type of Noise is the CAR Noise

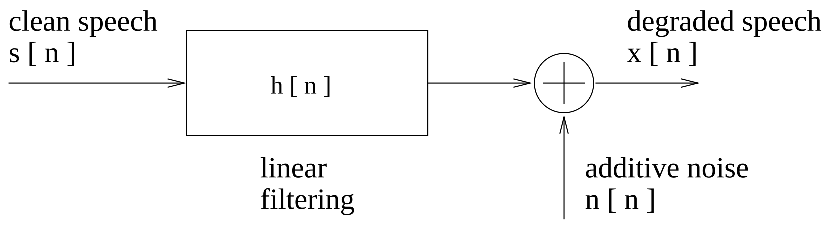


Figure 2.3: Representation of Additive Noise

magnitude in all frequencies. The power density decreases by -3 dB per octave with increase in frequency (density proportional to $1/f$). There are also many "less official" colors of noise such as brown, blue, purple, voilet, grey, red, orange, green and black.

2.2.1 Additive Noise

There are many sources of acoustic distortion that can degrade the performance of speech recognition systems. For many speech recognition applications the most important source of acoustical distortion is the additive noise [23]. Much research effort in robust speech recognition has been devoted to compensate the effects of additive noise.

If the speech signal $s(k)$ effected by uncorrelated noise $n(k)$ [24], then the observed signal in the frequency domain can be expressed as

$$Y(e^{jw}) = X(e^{jw}) + N(e^{jw}) \tag{2.1}$$

If $s(t)$ is the original clean speech signal, the received speech signal $y(t)$ in time domain can be represented as

$$y(t) = s(t) * h(t) + n(t) = x(t) + n(t) \tag{2.2}$$

where $h(t)$ is the impulse response of channel distortion and $n(t)$ the ambient noise. $(*)$ denotes the convolution operation, and $x(t)$ the noise-free speech as shown in the figure (2.2.1). Typical structural models for adaptation to variability assume that speech is corrupted by a combination of additive noise and linear filtering.

In speech processing, the speech is considered as useful data and all other signals are assumed to be noise. Many algorithms and applications are created to reduce or eliminate noise from signals, such as Voice Activity Detector.

2.3 Choice of HOS

In early VAD algorithms, short-term energy, zero-crossing rate and LPC coefficients were among the common features used for speech detection. Cepstral features, formant shape and least-square periodicity measure are some of the most recent metrics used in VAD designs. G.729B VAD has a set of metrics including the line spectral frequencies(LSF), low band energy, zero-crossing rate and full-band energy.

The short-time energy or spectral energy has been conventionally used as the major feature parameters to distinguish the speech segments from other waveforms. However, these features become less reliable and robust in noisy environments, especially in the presence of non-stationary noise and sound artifacts such as lip smacks, heavy breathing and mouth clicks etc.[14].

HOS has shown good results in a number of signal processing applications and are of particular value when dealing with a mixture of Gaussian and non-Gaussian processes and system nonlinearity. The application of HOS in speech processing is Gaussian suppression and phase preservation properties.

2.4 Problem Statement

The following are some of the problems needed to be solved to satisfy the project goal.

A Matlab & ANSI-C PROGRAM

1. HOS algorithm implementation.
2. Implementation of traditional VAD algorithm in [6]
3. Verification of Algorithm using TIA-Database [8]
4. Convertion and optimization of Matlab code to ANSI-C code.

B DSK IMPLEMENTATION

1. Embedding C code on to TI application specific processor (TMS320C6713) using Code Composer Studio(CCS).

Chapter 3

Design

The chapter discusses the various algorithms needed to design of robust VAD. The skewness and kurtosis as mentioned in appendices (A, F) of the LPC residual of voiced speech is expressed in terms of the number of harmonics M and signal energy. These parameters are greater than zero for any practical value of M which is a function of pitch. The normalized values of skewness and kurtosis are expressed in terms of M. These two metrics can be used to detect voice. The advantage of using the normalized metrics is that they are independent of the signal energy and therefore absolute thresholds are used. The variance of the estimators of the skewness and kurtosis and are normalized to get the unit-variance estimators. The relation between skewness and kurotsis in voiced speech is used to identify the speech frames. This forms the basis for the VAD algorithm using HOS. The appendix B shows the design in which the project was implemented.

3.1 Detection of Noise Frames using HOS

The skewness and kurtosis of Gaussian noise are zero only in a statistical average sense. Generally a finite length frames are used, so the decision that a given frame is noise can only be made in a probabilistic manner with a confidence level that takes into account the variance and distribution of the estimators of the skewness and kurtosis. Given a Gaussian process $g(n)$, the estimators of the second, third and fourth-order moments are

$$M_{kg} = \frac{1}{N}\Sigma_{n=0}^{N-1}[g(n)]^k \tag{3.1}$$

The above equation is for the estimator of $E[\{x(n)\}]^k$ for the values of $k = 2, 3, 4$ and N is the number of frames under consideration. These estimators are unbiased [1]. For the case of white Gaussian noise, their mean and variance may be expressed in terms of the process variance, v_g

$$\begin{aligned} E[M_{3g}] &= 0 \\ E[M_{4g}] &= 3v_g^2 \\ Var[M_{3g}] &= \frac{15v_g^3}{N} \\ Var[M_{4g}] &= \frac{96v_g^4}{N} \end{aligned} \tag{3.2}$$

Thus, the estimator of the skewness $SK = M_{3g}$ is unbiased, with zero mean and known variance. This estimator is the sum of a large number of independent identically distributed (iid) random variables, then by using central limit theorem, the normalized version is given by

$$SK_a = \frac{M_{3g}}{\sqrt{15v_g^3/N}} \tag{3.3}$$

is a Gaussian variable with zero mean and unit variance. Thus given the estimate of the skewness of a frame and the corresponding scaled value denoted by "a", the probability that the frame is Gaussian noise is

$$Prob[Noise] = Prob[|SK_a| \geq a] \tag{3.4}$$

which is equivalent to computing the area under the tail of the Gaussian curve of SK_a graphically. The area under the tail can be evaluated by $erfc(x)$ function[1]. When $a = 0$ the area under the curve is unity, whereas when $a > 0$, $Prob[Noise] = 2/\sqrt{2\pi}\int_a^\infty e^{x^2/2}\,dx$. Thus, $Prob[Noise] = erfc(|a|)$.

A negative skewness is not an indication of noise, while the HOS of speech are positive, since transient segments can have negative HOS. Similarly, the estimator of the kurtosis is first computed from the second and fourth-order moments. To ensure an unbiased estimate, the modified estimator is used

$$KU_U = \left(1 + \frac{2}{N}\right)M_{4g} - 3(M_{2g})^2. \tag{3.5}$$

This estimator is unbiased, with zero mean and known variance. The distribution consists of the difference of two variables, one Gaussian and one chi-square. However an approximation is used here and the estimator is assumed normally distributed.

A unit-variance version of this zero-mean variable is defined as

[1]error function

$$KU_{Ub} = \frac{KU_U}{\sqrt{\frac{3v_g^4}{N}\left(104 + \frac{452}{N} + \frac{596}{N^2}\right)}} \tag{3.6}$$

Therefore, given the value of the estimate of the kurtosis of a given frame and the corresponding scaled value, denoted by "b", the probability of a frame being noise is: $Prob[Noise] = erfc(|b|)$. The probability of a frame being noise using the normalized values of the estimates of skewness & kurtosis and the "erfc" function can be determined.

3.1.1 Necessary Condition for Voicing

The skewness and kurtosis of voiced speech are expressed in terms of energy and number of harmonics and may be used for detecting voiced frames [1]. To eliminate the effect of energy, one may consider the normalized metrics i.e., γ_3 and γ_4, but these metrics become less effective in the presence of noise, for detecting the voiced frames. Therefore, the ratio of the appropriate power of the skewness to that of the kurtosis is considered to eliminate the effect of signal energy, while avoiding the effect of noise.

$$SKR = \frac{skewness^2}{kurtosis^{1.5}} = \frac{9(M-1)^2}{8M\left[\frac{4}{3}M - 4 + \frac{7}{6M}\right]^{1.5}} \tag{3.7}$$

SKR Ratio is independent of signal and is only a function of M where M is the number of harmonics(function of pitch). When Gaussian noise is present, the ratio is undetermined since both operands are zero. But, this zero condition never occurs due to variance of the estimators. The SKR ratio may take on any value, including the range for voiced speech; thus not sufficient enough for detecting voice frames (when transient values in voice speech).

3.2 HOS-Based VAD Algorithm

The sustained unvoiced speech is shown to have Gaussian-like characteristics, it cannot be distinguished from Gaussian noise using HOS [1]. But in reality unvoiced speech occurs at speech transitional boundaries having nonzero HOS. Therefore the VAD detection proposed based on HOS and is formulated as a finite two state machine. The algorithm combines the use of skewness, kurtosis, their normalized versions γ_3 and γ_4, SNR, LPC prediction error, and SKR ratio for distinguishing speech from noise frames.

The following explains the algorithmic steps:

1. **Data Format:**

   Speech sampled at $8kHz$ is used, a tenth-order LPC analysis[2] is performed once every $20ms$, thus generating a $20ms$ residual. VAD is carried out every $10ms$ using the residual and a 20% overlap.

2. **HOS Computations:**

   Every $10ms$ iteration, the estimators for the second, third and fourth-order moments are computed using (3.1) with $N = 100$. An autoregressive scheme is used to smooth the estimates of the moments. From these, the unbiased estimate of the kurtosis (3.5) is deduced. The estimate of the skewness is simply the third-order moment (3.1). Then they are normalized by the signal energy to give

$$\begin{aligned} \gamma_3 &= \frac{SK}{M_{2x}^{1.5}} \\ \gamma_4 &= \frac{KU_u}{M_{2x}^2} \end{aligned} \tag{3.8}$$

3. **Noise and SNR Estimation:**

   The noise power is estimated using frames declared as nonspeech. Moreover, it is assumed that first three frames are nonspeech and are used to initialize the noise power estimate. Whenever a frame is declared as nonspeech, its energy is used to update noise estimate according to an autoregressive averaging

$$v_g(k) = (1-\beta)v_g(k-1) + \beta M_{2X} \tag{3.9}$$

   where k is iteration index;
   M_{2X} is frame energy;
   v_g is estimate of the noise energy;
   β is 0.1*Prob[Noise].

   At every iteration the current estimate of the noise energy is used to compute the SNR of that frame.

$$SNR = Pos\left[\frac{M_{2X}}{v_g} - 1\right] \tag{3.10}$$

   where $Pos[x] = x$ for $x > 0$ and 0 otherwise. In the above equation M_{2X} is the power of the speech corrupted with noise and v_g is the noise energy.Since the residual is low-pass filtered at $2kHz$, the above SNR is applicable to the lower spectrum only. The *total SNR* is computed using the nonfiltered residual and the energy of the full band.

[2]refer to appendix C for details

4. **Probability of Noise-only Frames:**

   Once the skewness and kurtosis are computed, the variance of these estimates are computed using the noise energy v_g, according to (3.3) and (3.6), to yield the zero-mean, unit variance estimates SK_a and KU_{Ua}, respectively. From these two scaled values, the probability of the frame being noise is deduced

$$Prob[Noise] = [erfc(a) + erfc(b)]/2 \qquad (3.11)$$

   where a and b are the computed values of SK_a and KU_{Ub}, respectively.

5. **SKR Ratio:**

   The ratio is computed directly from the non-normalized estimates of the skewness and kurtosis

$$SKR = \frac{[SK]^2}{[KU_U]^{1.5}} \qquad (3.12)$$

6. **LPC Prediction Error:**

   The LPC prediction error is the inverse of the prediction gain and may be computed from the set of the reflection coefficients (r_i) generated by the LPC analysis

$$PE = \Pi_{i=0}^{10}(1 - r_i^2) \qquad (3.13)$$

7. **Speech/Noise State Machine:**

   The VAD algorithm is implemented as a two-state machine as ahown in the figure (3.1). The following operations are carried out in each state.

   (a) **Noise State:** The noise energy is updated according to the $Prob[Noise]$ (3.11). The SKR ratio, the Gaussian likelihood (Probability of noise), the SNR (3.10) and the Probability of error (3.13) values are used to determine whether the frame is speech. The occurrence of the following three conditions triggers a transition:

      i. $Prob[Noise] < T_{Gaus}$ for two consecutive frames.

      ii. SKR in voicing range and ($SNR > T_{SNR1}$ or $PE < T_{PE}$) indicates a voiced frame.

      iii. Total $SNR > T_{SNR2}$ indicates a strong speech frame.

   (b) **Speech State:**

      The noise likelihood (3.11) along with the values γ_3 and γ_4 (3.8) are used to determine whether the frame is Gaussian. After a hangover period, transition to the noise state occurs if $Prob[Noise] > T_{Gaus}$ and $\gamma_3 < T_{\gamma_3}$ and $\gamma_4 < T_{\gamma_4}$

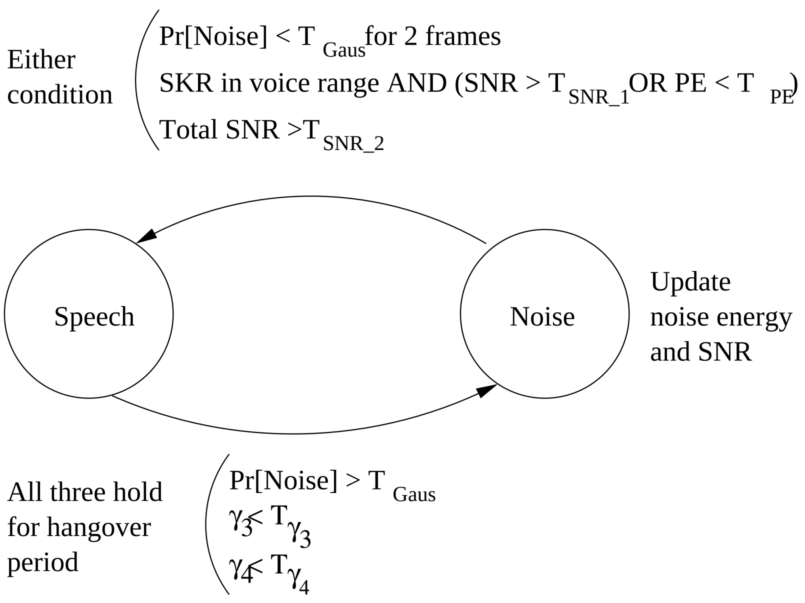

Either condition
Pr[Noise] < T Gaus for 2 frames
SKR in voice range AND (SNR > T SNR_1 OR PE < T PE)
Total SNR >T SNR_2
Speech
Noise
Update noise energy and SNR
All three hold for hangover period
Pr[Noise] > T Gaus
γ3 < Tγ3
γ4 < Tγ4

Figure 3.1: HOS based VAD State Machine

3.3 Estimation of the Instantaneous SNR of Speech Signals

Estimation of the instantaneous SNR is an essential component of speech processing algorithms which are sensitive to varying noise levels [6]. An instantaneous SNR estimate is based on short time power estimates with time constants of integration in the range of $0.02 - 0.1s$. To acquire noise statistics, the conventional approach to SNR estimation employs a VAD to extract the noise only segments of the disturbed speech signal.

The Rainer Martin's algorithm, does not need an explicit speech/nonspeech decision to gather noise statistics and is capable to track varying noise levels during speech activity. The algorithm is based on the observation that the smoothed power estimate of a noisy speech signal exhibits distinct peaks and valleys. While the peaks correspond to speech activity the valleys of the smoothed noise estimate can be used to obtain a noise power estimate. To estimate the noise floor, the algorithm takes the minimum of a smoothed power estimate within a window of finite length.

3.3.1 Algorithmic Description

Assume that the bandlimited and sampled disturbed signal $x(i)$ is sum of a speech signal $s(i)$ and a noise signal $n(i)$, $x(i) = s(i)+n(i)$, where i denotes the time indexand also assuming that $s(i)$ and $n(i)$ are statistically independent, $E\{x^2(i)\} = E\{s^2(i)\}+E\{n^2(i)\}$.

$SNR_x(i)$ denotes the estimated signal-to-noise ratio fo signal $x(i)$ at time i. The algorithm works on a sample basis, i.e. a new output sample $SNR_x(i)$ is computed for each input sample $x(i)$. The computation of $SNR_x(i)$ is based on a noise power estimate $P_n(i)$ which is obtained as the minimum of the smoothed short time power estimate $\tilde{P}_x(i)$ within a window of L samples.

Besides initialization the algorithm is split into three major parts:

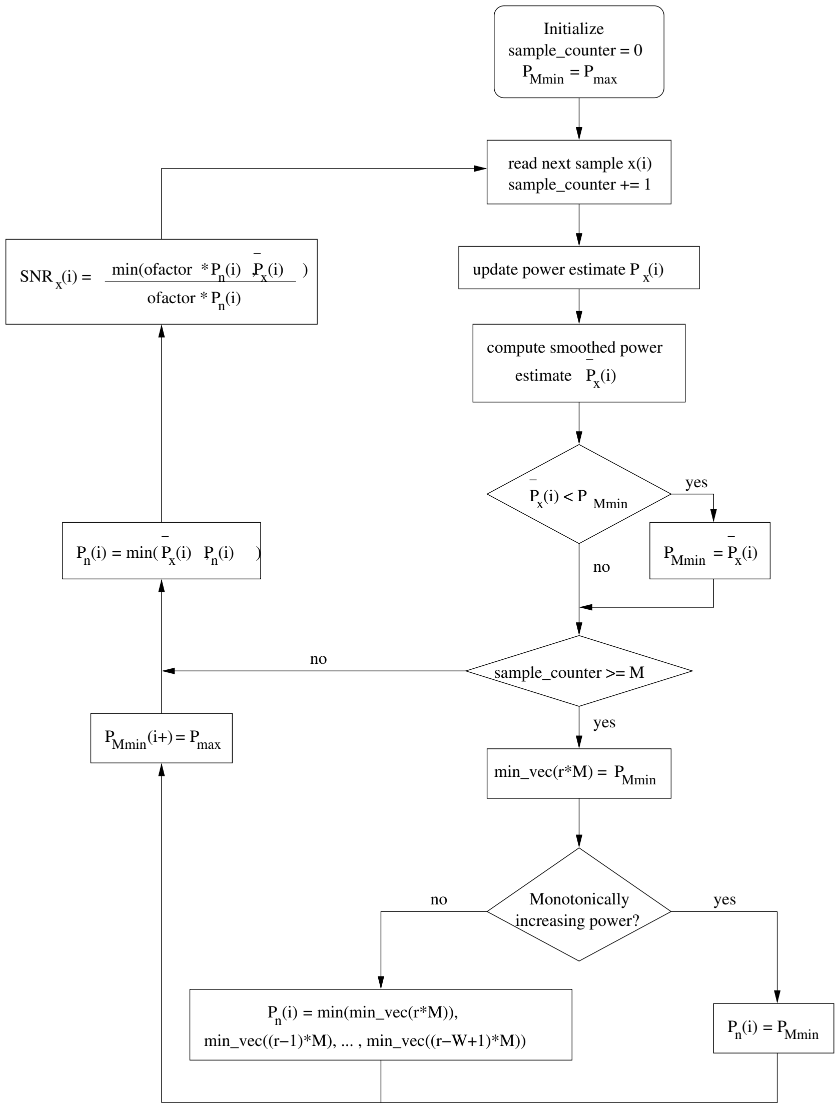

Initialize
sample_counter = 0
$P_{Mmin} = P_{max}$
read next sample x(i)
sample_counter += 1
update power estimate $P_x(i)$
compute smoothed power estimate $\bar{P}_x(i)$
$\bar{P}_x(i) < P_{Mmin}$
yes
no
$P_{Mmin} = \bar{P}_x(i)$
sample_counter >= M
no
yes
min_vec(r*M) = P_{Mmin}
Monotonically increasing power?
no
yes
$P_n(i)$ = min(min_vec(r*M)), min_vec((r−1)*M), ... , min_vec((r−W+1)*M))
$P_n(i) = P_{Mmin}$
$P_{Mmin}(i+) = P_{max}$
$P_n(i) = \min(\bar{P}_x(i)\ P_n(i))$
$SNR_x(i) = \frac{\min(\text{ofactor} * P_n(i)\ \bar{P}_x(i))}{\text{ofactor} * P_n(i)}$

Figure 3.2: The Estimation of the Instantaneous SNR of Speech Signals

1. Computation of a smoothed short time power estimate $\tilde{P}_x(i)$ of signal $x(i)$
2. Computation of the noise power estimate $P_n(i)$
3. Computation of the $SNR_x(i)$

Figure (3.2) shows the overall flow of RM algorithm.

1. **Computation of a smoothed power estimate**
   Computation of the short time signal power $P_x(i)$ and smoothing of the power estimate is done in two steps. The power estimate may be obtained recursively or non-recursively. A sliding rectangular window of length N with $N = 128$ is normally used. Let $\tilde{P}_x(i)$ denote the smoothed short time power estimate at time i. Smoothing of the power estimate is done constant is typically set to values between $\alpha = 0.95 \dots 0.98$. The recursion for $i > N$ is given by (3.14):

$$\begin{aligned} P_x(i) &= P_x(i-1) + x(i) * x(i) - x(i-N) * x(i-N) \\ \tilde{P}_x(i) &= \alpha * \tilde{P}_x(i-1) + (1-\alpha) * P_x(i) \end{aligned} \tag{3.14}$$

   In figure (3.3) the first two parts show the short time signal power and the smoothed power estimate respectively for the case 67 (speech in car noise environment) for 12000 samples.

2. **Noise power estimation**
   The noise power estimate is shown in the third part of the figure (3.3). The noise power estimate is based on the minimum of signal power within a window of L samples. For reasons of complexity and delay the data window of length L is decomposed into W windows of length M such that $M * W = L$.

   The minimum power of the last M samples is found by a sample wise comparison of the actual minimum $P_{min}(i)$ and the smoothed power $\tilde{P}_x(i)$. Whenever M samples are read, i.e. $i = r * M$, the minimum power of the last m samples are stored and the maximum value of $P_{min}(i = r * M)$ is reset: $P_{Mmin}(i = r * M+) = P_{max}$

   Determination of the noise power is estimated by two cases:

   (a) slowly varying noise power,
   (b) rapidly varying noise power.

   If the minimum power of the last W windows with M samples each is monotonically increasing, then a rapid noise power variation decision is made. In this case the noise power estimate equals the power minimum of the last M samples
   $P_n(i) = P_{Mmin}(i = r * M)$.

   In case of non monotonic power the noise power estimate is set to the minimum of the length L window $P_n(i) = P_{Lmin}(i)$. The minimum power of the length L window is easily obtained as the minimum of the last W minimum power estimates:

$$P_{Lmin}(i) = min(P_{Mmin}(i = r * M), P_{Mmin}(i = r * M), \dots P_{Mmin}(i = r * M)) \tag{3.15}$$

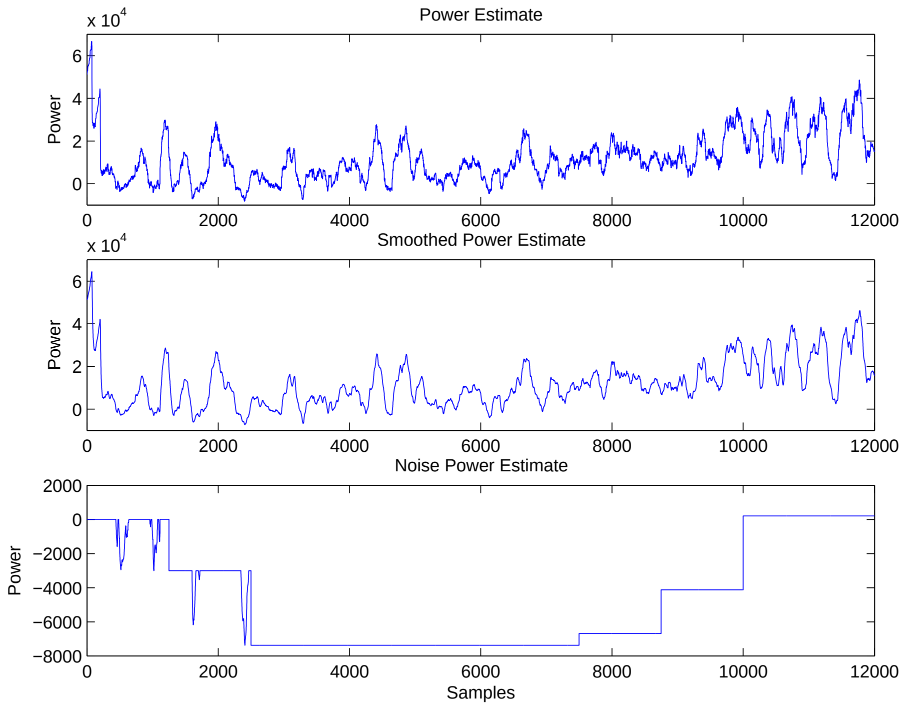


Figure 3.3: Power Estimation using Rainer Martin's Algorithm

If the actual smoothed power is smaller than the estimated noise power $P_n(i)$ the noise power is updated immediately independent of window adjustment: $P_n(i) = min(\tilde{P}_x(i), P_n(i))$.

3. **Computation of SNR**
The estimated SNR is computed on the basis of the estimated minimum noise power $P_n(i)$. A factor $ofactor$ accounts that the minimum power estimate is smaller than the true noise power. The range of $ofactor$ is between 1.3 and 2

$$SNR(i) = 10 * \log_{10}\left(\frac{\tilde{P}_x(i) - min(ofactor * p_n(i), \tilde{P}_x(i))}{ofactor * P_n(i)}\right) \tag{3.16}$$

The window length $L = M * W$ must be large enough to bridge any peak of speech activity, but short enough to follow non stationary noise variations. In case of slowly varying noise power the update of noise estimates is delayed by $L + M$ samples. If a rapid noise power increase is detected this delay is reduced to M samples, thus improving the noise tracking capability of the algorithm.

Table (3.1) shows a simple representation of steps in implementing HOS-VAD. Figures (3.4) and (3.5) shows the system models or the implementation flow for HOS and (RM + HOS) Algorithm respectively with both in Matlab and DSK.

Process	**Input**	**Output**	**Description**
Buffering	Speech signal	Sampled frame	The speech signal is represented in a matrix form resulting in the frames
LPC	Frame	Residual, LPC coefficients	Calculates the LPC coefficients and residual
HOS computations	Residual	Normalized skewness and normalized kurtosis	The second, third and fourth order moments are calculated and hence the skewness and kurtosis
Estimation of noise and SNR	Residual	Estimated noise energy and SNR	Noise energy and the SNR are calculated using the frame energy.
Calculation of probability of noise-only frames	Normalized skewness and kurtosis with respect to Noise energy	Probability of noise only frames	Calculated using the error function of the normalized skewness and kurtosis.
SKR Ratio	Normalized skewness and kurtosis	SKR Ratio	Calculates the SKR ratio
LPC prediction error	Reflection coefficients from the LPC analysis	LPC prediction error	Calculates the LPC prediction error
Noise state	Probability of noise, SKR ratio, SNR and probability of error	Frames considered as noise	Depending on the threshold, decision is made that the frame is noisy.
Speech state	Noise likelihood, normalized skewness and kurtosis	Frames considered as speech	Frame is decided as speech depending on the threshold values.
Calculation of the probabilities	Correctly detected speech frames, correctly detected noise frames, incorrectly classified speech or noise frames	Probability of correctly detected speech, noise frames and probability of false detection	Calculates the probability of correctly detected speech, noise frames and probability of false detection $P_{c_{speech}}$, $P_{c_{noise}}$, P_f.

Table 3.1: Process Description

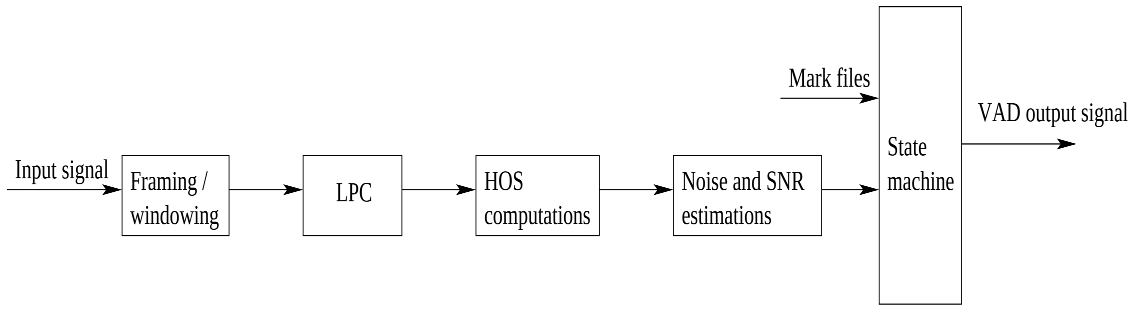


Figure 3.4: System Model of HOS-VAD

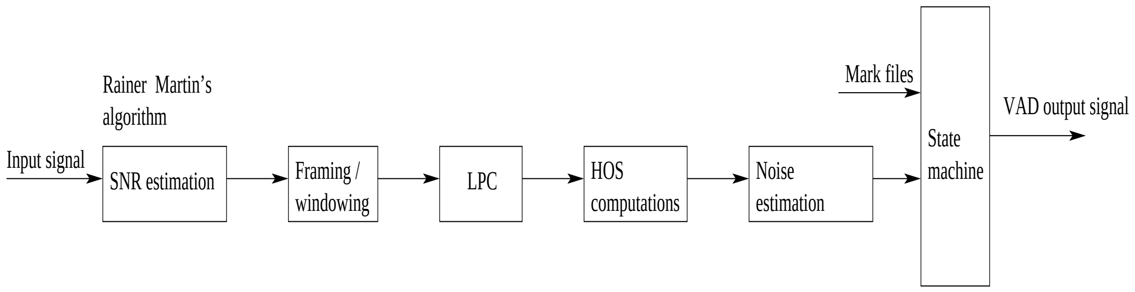


Figure 3.5: System Model using Rainer Martin's Algorithm of HOS-VAD

Chapter 4

Implementation

This chapter describes the implementation of the algorithm mentioned in the previous chapter, which is done in three steps: designing the algorithm and running the simulations on Matlab then converting the Matlab code into C code (Conversion to C code was done in two ways). The resulting C code is then implemented on the Code Composer Studio which is the interface for the TMS320C6713 DSK.

4.1 Matlab Simulation of VAD Algorithm using HOS

Simulations of the system represent the functionality of the individual process mentioned in this chapter. In simulation process it was assumed, that some program takes input signal, frames it in $20ms$ frames and supplies the result for the simulation. During simulation algorithm works in real time and only with one frame at a time. Furthermore, it is assumed that RVAD algorithm using HOS is not only working with the current frame, but also gets coming samples of the next frame.

4.1.1 Input Signals

The system is simulated with different speech signals[1]:

- Noise contaminated signal i.e., the test cases [8]
- Noise free signal or the clean speech signal
- Mark files or the reference signal

The speech signals consist of the 10 different scenarios of which five are male and five are female speakers. There are four noise signals used which are Car, Garage, Train and Street noises. The speech data signals and noise signals are combined in various ratios and result in 80 different test cases. Each case is a different combination of the

[1]Refer to appendix E for details of the source of the speech signals

speech normalization level, the noise type and the SNR. These cases have different SNR levels of 6dB, 12dB, 18dB and ∞. For example, *Case 6* is created combining speech file *m1left1.nom* and noise file *car.nom* added for a *SNR of 6dB.* These different test cases form the noise contaminated speech signals as the input for the algorithm. The reference signal or the mark files were generated for the comparison of the results.

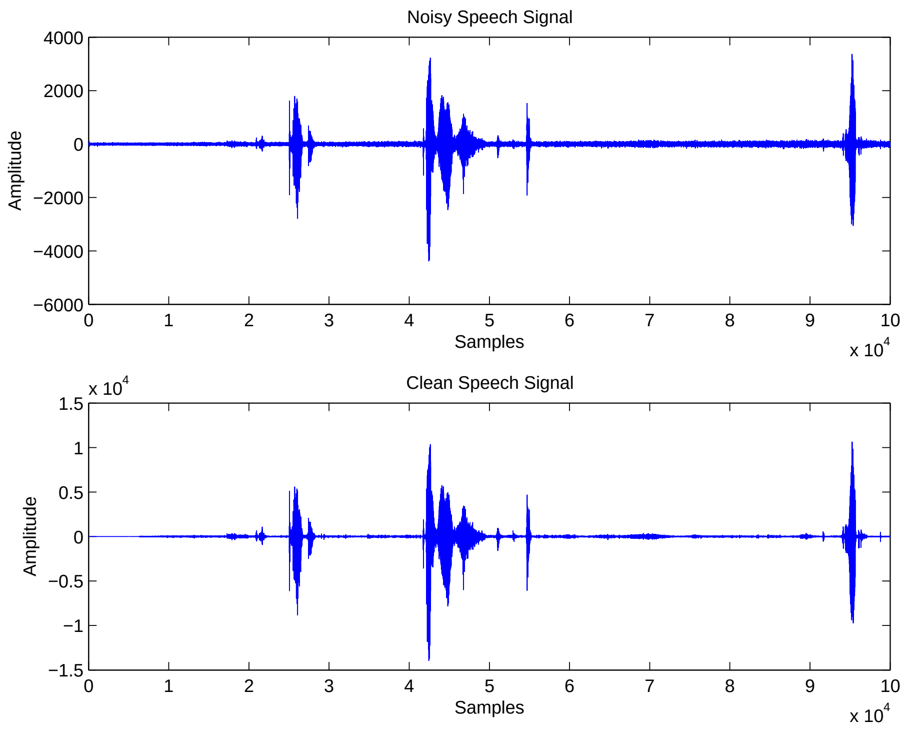


Figure 4.1: Speech Signals with and without Noise

4.1.2 Framing

Frame size is set to $20ms$, since the frame length is considered to be between $10ms$ to $30ms$. If the length is less than $10ms$, it results in roughness and the frame size more than $30ms$, the perceptual quality is decreased.

4.1.3 Windowing

The windowing length determines the portion of the speech signal that is to be selected. The ideal window frequency response has a very narrow main lobe which increases the resolution and decreases the side lobes or frequency leakage. Since an ideal window does not exist practically so a compromise is made depending on the specific application.

Different windows are available such as rectangular, hanning or hamming window. The rectangular window has the highest frequency resolution due to the narrow main lobe and having a large frequency leakage. The large side lobes results in high frequency leakage thus the rectangular windowed speech is noisier. So the rectangular window is not used for spectral analysis of speech. The trapezoidal windows such as hamming and hanning windows are prominent having smaller frequency leakage but with lower resolution. Thus produce a smoother spectrum than the rectangular window. Hanning window is used.

4.1.4 Calculation of the HOS Parameters

In order to detect whether the current frame is speech or non-speech frame, normalized skewness and normalized kurtosis are estimated for the frames. Those values are counted according to equation (3.8). For this reason beforehand second, third and fourth order moments are computed using equation (3.2).

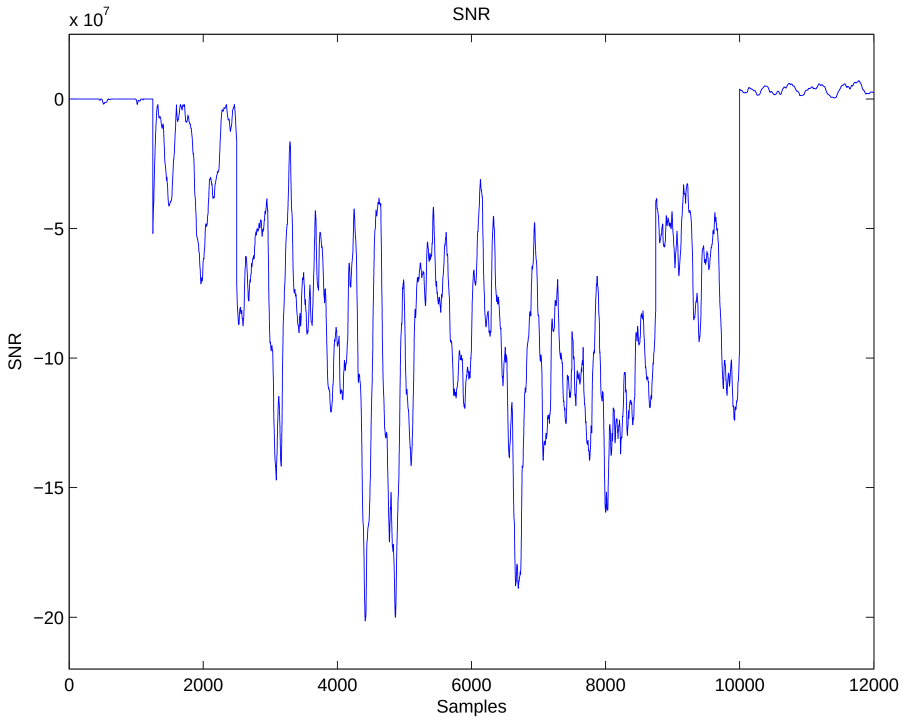


Figure 4.2: SNR Estimation using Rainer Martin's Algorithm

4.1.5 Calculation of SNR

Calculation of signal-to-noise ratio is performed using the current estimate of noise energy as in equation (3.10). Noise power is required for estimating noise energy. If the noise power declares the current frame as non speech then the noise energy is computed using equation (3.9). Otherwise, noise energy is left unchanged from the previous frame. After calculation of SNR for each frame, "total SNR" metric is updated.

HOS, Rainer Martin's algorithm mentioned in (3.3) for estimation of SNR is applied to improve VAD algorithm. But SNR estimation using Rainer Martin's algorithm is done before LPC as the SNR estimation is utilized sample by sample basis. The SNR estimation for the first 12000 samples of *case 67* is shown in the figure (4.2).

4.1.6 State Machine

After estimation of normalized skewness, normalized kurtosis, SNR, noise probability, LPC prediction error and SKR, the algorithm decides whether frame is speech or non-speech. The decision is made using a state machine model, which has two states: noise and speech state. The current state depends on the previous frame.

If the state machine is in noise state, the verification is done based on whether the current frame is still noise or not. The decision is made according to the values of noise probability, prediction error, SKR, SNR and total SNR for the current frame in comparison to the appropriate thresholds set.

If state machine is in speech state, decision is based on whether the current frame is still speech or not. The decision is made according to the results of noise probability, normalized skewness and normalized kurtosis for the current frame in comparison to the appropriate thresholds.

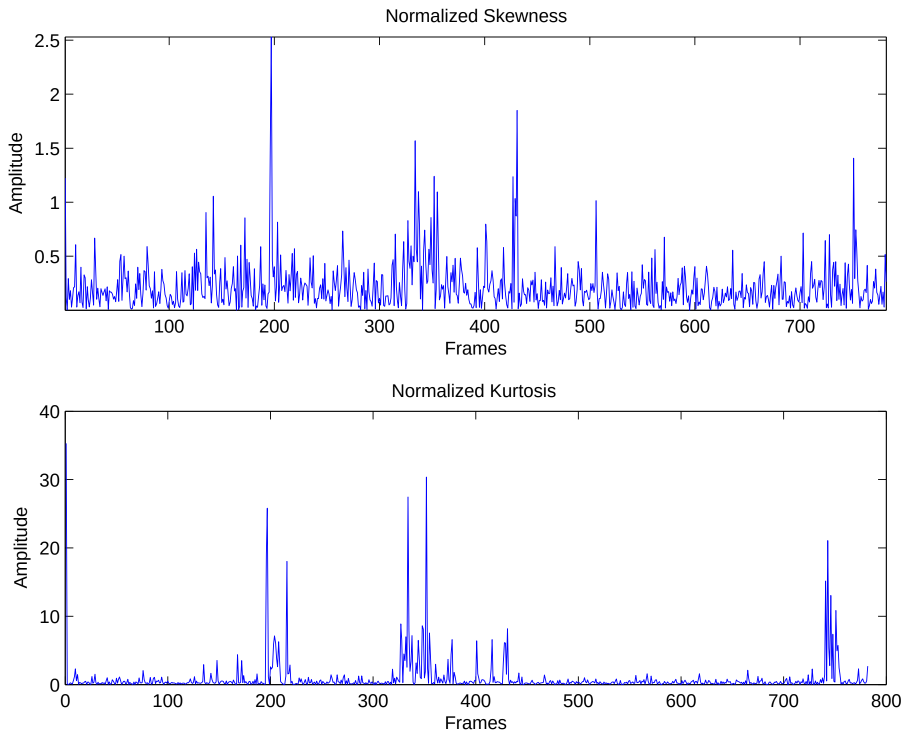


Figure 4.3: Normalized Skewness and Kurtosis

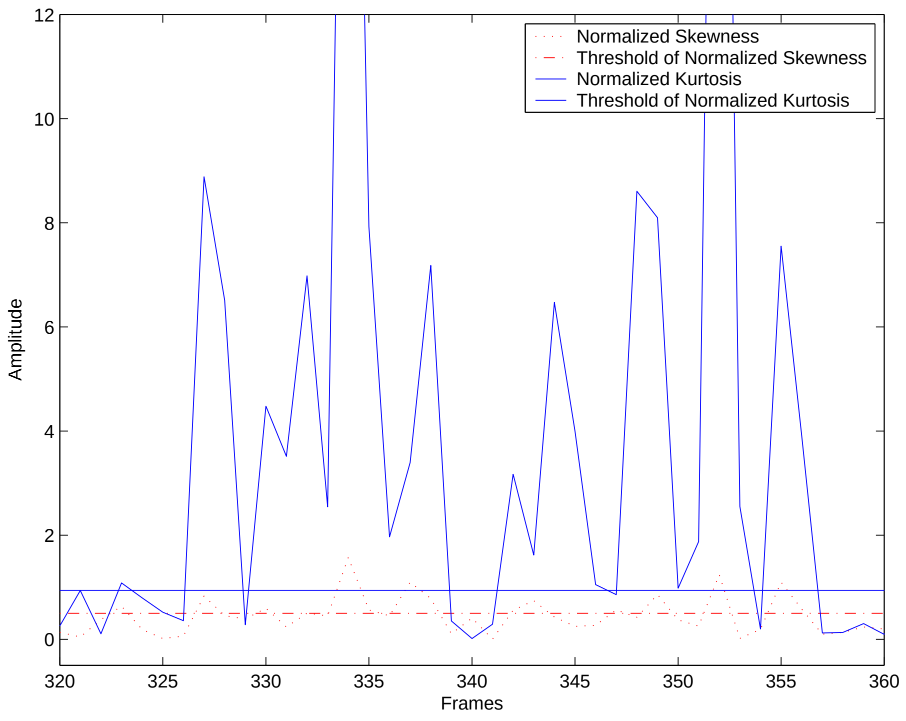


Figure 4.4: Normalized Skewness and Kurtosis between Frames 320 to 360

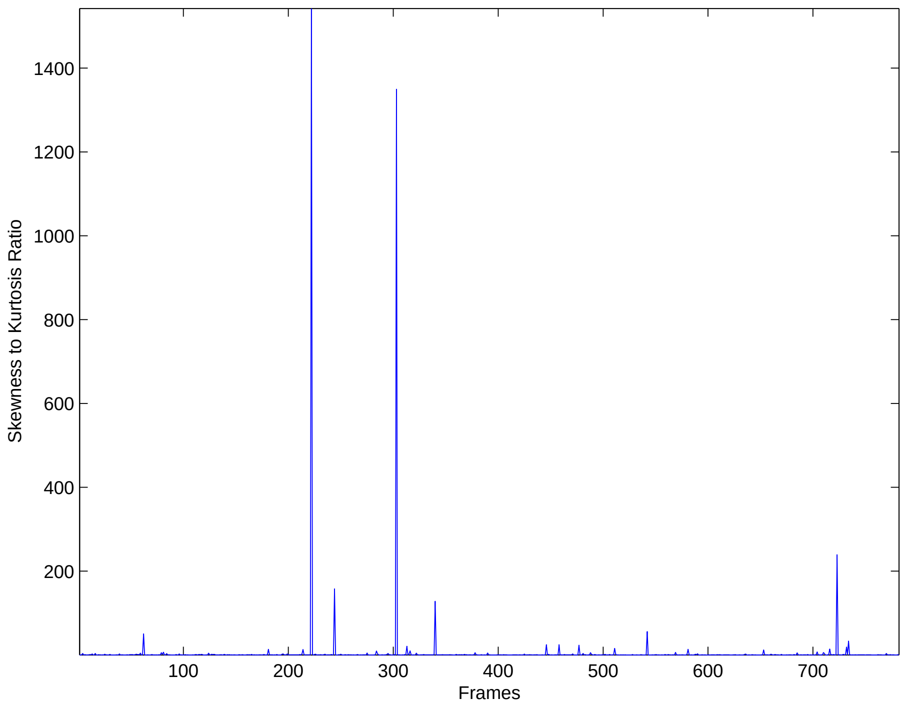


Figure 4.5: SKR Ratio

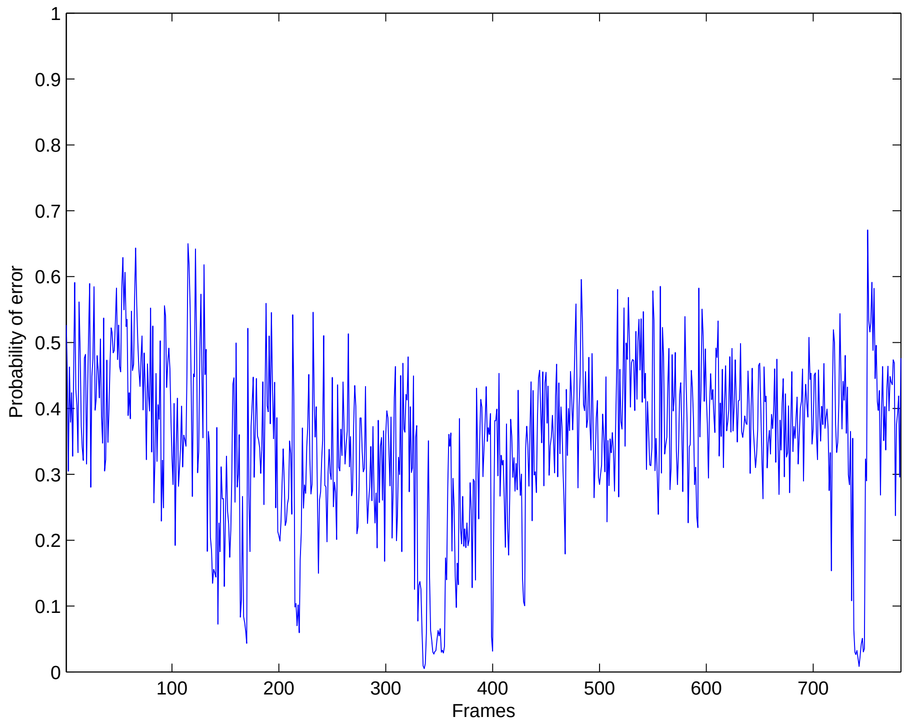


Figure 4.6: Prediction Error

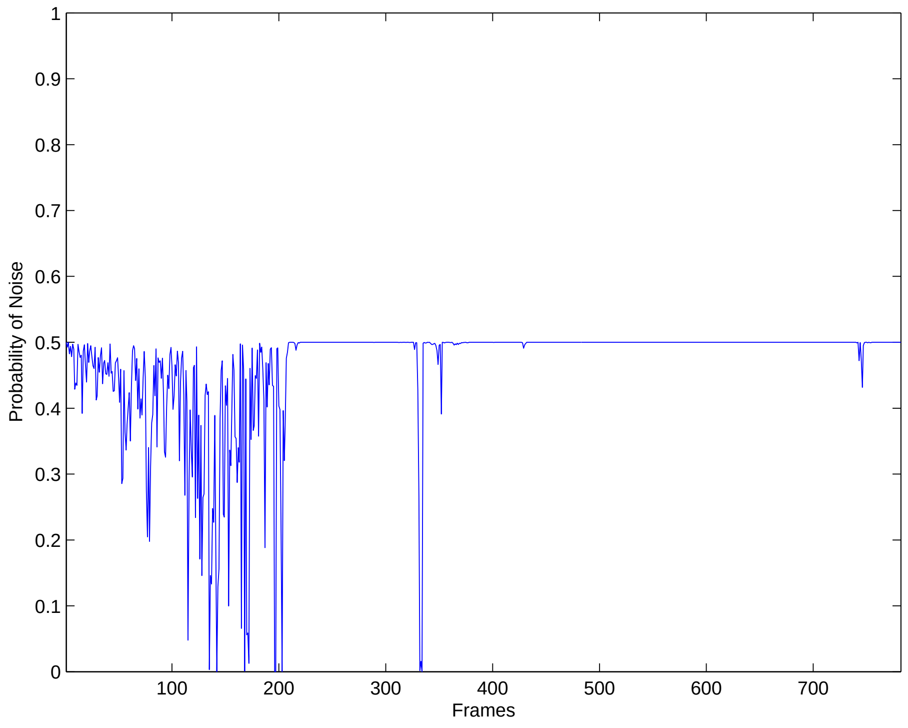


Figure 4.7: Probability of Noise

4.2 Analysis of the results

The results of algorithm are presented in plots of the computed parameters (figures (4.1) - (4.10)). The input signal for computations is 100000 samples of *case 67* which is combination of female speech and car noise as shown in the figure (4.1).

In this case, the highest peaks in the plots of normalized skewness and normalized kurtosis figure (4.3) show speech frames. Combining the results of normalized skewness and normalized kurtosis the decision is made whether the frame is speech or noise. For example, the figure (4.4) shows 40 frames of the signal. The signal frames are noise below 0.5 of skewness and 0.94 of kurtosis amplitude respectively. Frames from 339 to 341 are considered noise because the skewness and kurtosis are below the thresholds. Furthermore, the frames from 354 to 357 frames are considered as speech, but if the values are less than the thresholds then the frame is decided as definitely noise. The frames considered as speech are decided based not only on the skewness and kurtosis, but also on other thresholds.

Skewness to kurtosis ratio plot shown in the figure (4.5) is one of the parameters which helps in detecting the speech frame. The high peaks in SKR plot means non gaussian noise, contrary to normalized skewness and normalized kurtosis plots. Analyzing the plot, shows that speech frames belong to particular range of amplitude values.

The plots of prediction error and probability of noise depicted in the figures (4.6), (4.7) respectively show the statistical information about frames. The higher the value of prediction error is, the more likely the frame is noise. The lower the probability of noise the higher the possibility that frame is speech. Figure (4.8) shows the histogram of frame-by-frame values of the normalized kurtosis generated for 6250 frames of LPC residual signal. Another histogram is generated for the normalized kurtosis for the randomly generated Gaussian noise before LPC filtering. These histograms show the difference in the fourth-order statistics between speech and Gaussian noise. It shows that the speech utterance contains silence periods when kurtosis is zero as shown in the figure.

Based on all of the above parameters, the effectiveness of algorithm is evaluated and three performance metrics are computed. $P_cSpeech$ is probability of correctly detecting speech frames, computed as the ratio of correct speech detections to the total number of hand-labeled speech frames. P_cNoise is probability of correctly detecting noise frames, computed as the ratio of correct noise detections to the total number of hand-labeled noise frames. P_f is probability of false detection, computed as the ratio of incorrectly classified speech or noise frames to the total number of frames. Table 4.1 shows evaluation of car noise signal with SNR $0dB$. As there is no speech in this signal, therefore $P_cSpeech$ is not counted.

In addition, reference files for speech signal were made comparing results of algorithms

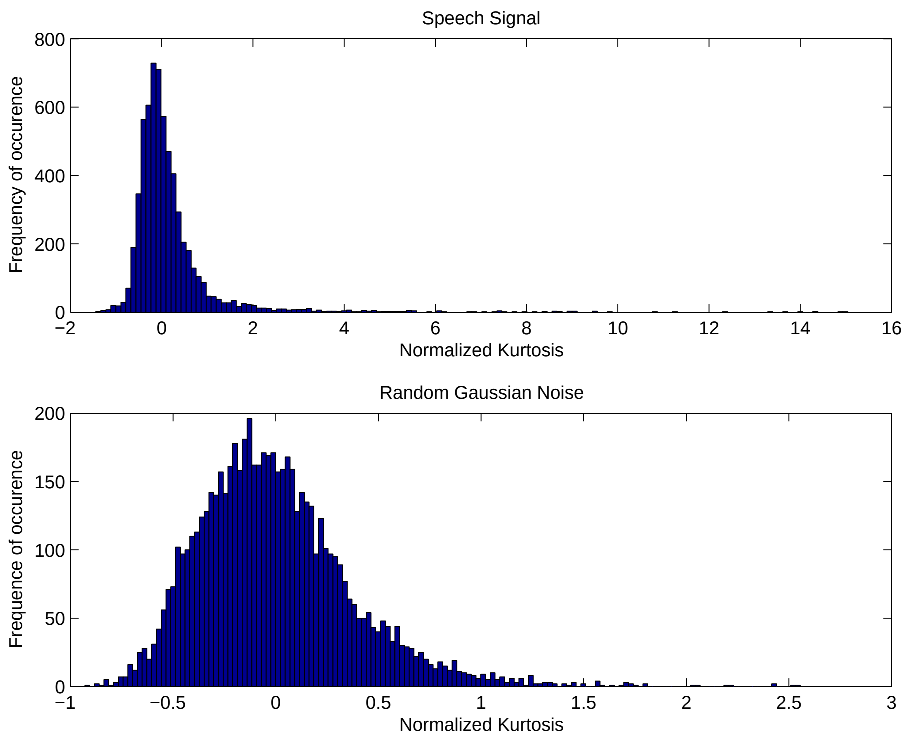


Figure 4.8: Histograms of Normalized Kurtosis of the LPC Residual (Speech versus Gaussian)

with true VAD. Decision whether frame is speech or non-speech, is made according to frame energy. If it is above threshold then frame is speech otherwise frame is silence. In reference plot speech is considered as one and silence has zero. Figures (4.9)and (4.10) show the plots of HOS-VAD and RM + HOS VAD algorithms respectively. Both algorithms were compared to each other to evaluate its effectiveness.

Different Noise environments such as street, garage, car and noise with different SNR levels were used and its corresponding $P_cSpeech$, P_cNoise, P_f were calculated as shown in the table (4.2).

As mentioned earlier, P_cs and P_f are calculated based on the thresholds set for the detection whether the speech frame is speech or not. The thresholds vary for different noise environments and even for the different SNR levels. The thresholds were fixed and could not be made adaptive because adaptive thresholds did not give expected results. The main focus was set on the probability of detecting the frames as speech to be high because speech detected as noise is not acceptable.

It can be inferred from the table that the overall performance of RM-HOS VAD is better than that of HOS-VAD. For example, RM-HOS produced more acceptable results for CAR noise for $18dB$, the $P_cSpeech$ is as high as 99% as compared to that of 97% for HOS VAD. Similarly the metrics for the other noise environments can be analyzed from the table (4.2)

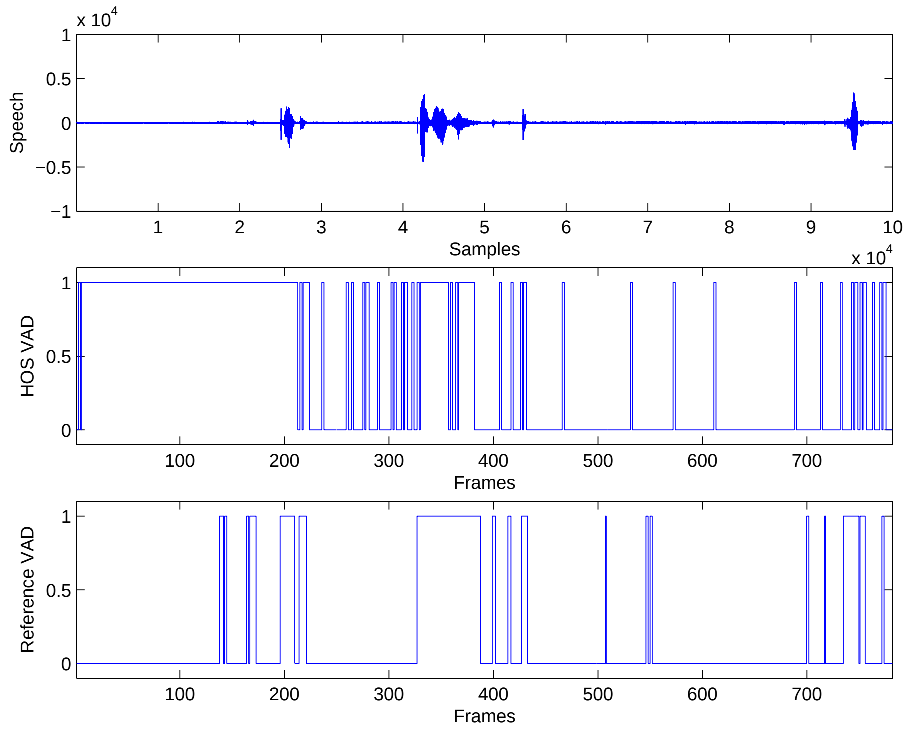


Figure 4.9: Comparison with the Reference VAD

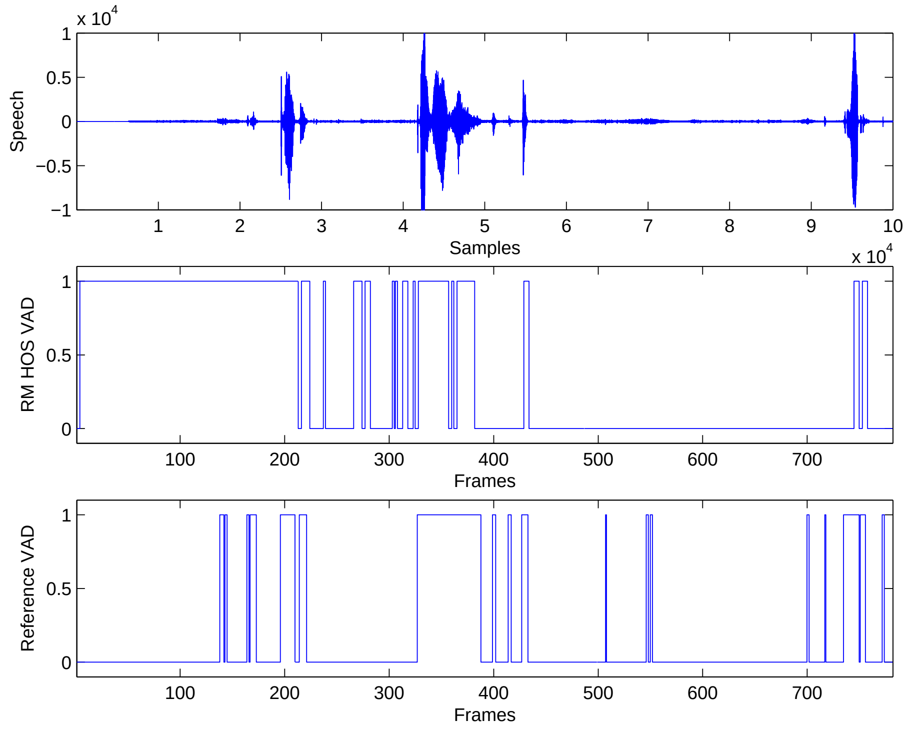


Figure 4.10: Comparison with the Reference VAD using Rainer Martin's Algorithm

Noise Environment		Pc Speech (%)	Pc Noise (%)	Pf (%)
Type	SNR	RM HOS VAD	RM HOS VAD	RM HOS VAD
Car	0 dB	∞	91.2668	8.7332

Table 4.1: 0 dB SNR for CAR Noise

4.3 C Coding

In this project, the conversion of Matlab code to C code was performed in two different techniques. At first, there was attempt to convert Matlab code directly to C using Matlab compiler. Later there was written the plain C code.

4.3.1 Using Matlab for conversion

To write a DSP compatible ANSI C code, Matlab Compiler and Matlab C++ were used. Below are the steps by which Matlab code is converted to C code.

- Installing libraries, which are needed during compilation. To do this »mbuild -setup command is used.
- Copying M-files of algorithm to a working directory.
- Converting M-files to C using Matlab compiler. To do this *mcc command* is used. Generated C code can be compiled by any ANSI C compiler. Wrapper files can also be generated to interface betweeen converted code and executable type.

Not all Matlab M-files can be converted to C using Matlab compiler. There are these restrictions:

- M-files containing scripts
- M-files that use objects
- M-files that use Matlab commands input or eval
- M-files that use Matlab command exist with 2 input arguments
- M-files that load files

The RM+HOS C code (converted with Matlab) has the following functions:

Noise Environment		Pc Speech (%)		Pc Noise (%)		Pf (%)	
Type	SNR	RM HOS VAD	HOS VAD	RM HOS VAD	HOS VAD	RM HOS VAD	HOS VAD
Street	18 dB	95.73766	95.02624	98.61536	95.27732	19.33330	19.94640
Street	12 dB	96.41824	94.81300	93.52636	97.22586	24.49698	24.24436
Street	6 dB	95.90054	96.27360	89.79290	94.24034	26.80592	29.57412
Garage	18 dB	92.62478	90.89854	93.81738	96.50380	16.34816	17.41288
Garage	12 dB	96.76172	91.22174	98.18864	96.31108	21.21978	21.28208
Garage	6 dB	97.89894	92.55860	96.54834	98.21720	25.47020	29.88722
Car	18 dB	99.35094	97.49856	98.45012	93.31110	21.46160	23.55938
Car	12 dB	94.00594	90.20506	96.38050	96.81272	21.76148	21.59166
Car	6 dB	95.47264	93.12092	96.97286	94.75996	26.49182	27.59700
Train	18 dB	93.01504	94.18450	94.08644	97.72330	18.26398	19.86844
Train	12 dB	97.90404	86.78462	97.32174	98.22220	22.33514	23.31566
Train	6 dB	94.83100	92.81294	92.21040	94.00732	26.79338	29.6381

Table 4.2: $P_c's$ and $P_f's$ for the HOS based VAD

- C code consists of the following highest level functions/headers
  rt_hos_vad_mainhg.c : main function for evaluation purpose.
  rt_hos_vad.c : real time RM+HOS function.
  rt_hos_vad.h : real time RM+HOS header.

- The following files contain global definitions, constants and other related subroutines:
  hanning.c : Hanning window function
  lpc.c : General linear predictive implementation function
  poly2rc.c : onversion of lpc coefficients to reflection coeff. function
  levinson_mex_interface.c : Levinson-durbin solution function
  buffer_mex_interface.c : buffering signal vector function

Most C files have their respective header files. All other subroutines and definitions can be found in companion CD attached to this report.

4.3.2 C code of the algorithms

Due to the problems encountered with C code converted from Matlab, new HOS and RM+HOS ANSI-C code were written for the implementation on DSK. These programs use input file, which contains 16-bit data, stored in high-byte/low-byte word format.

HOS program contains the following basic functions:

- *HanningWindow:* This function generates Hanning window of the set size. Different from Matlab C code, the Hanning window here is without zero padding.
- *SignalFraming:* Function forms current frame of the set size.
- *LPC:* This Linear Predictive Model function uses AutoCorrelation and Levinson-Recursion functions. It generates linear prediction and reflection coefficients.
- *AllPoleFilter:* Function generates residue from current frame and linear prediction coefficients.
- *HOSCompute:* Function for Higher Order Statistics.
- *GetVgSNR:* It estimates noise energy and signal-to-noise ratio.
- *GetSKR:* Function estimates skewness-to-kurtosis ratio.
- *PredictionError:* It computes LPC prediction error using reflection coefficients.
- *StateMachine:* This function implements two-state machine used for deciding whether a frame is speech or noise.

RM+HOS program consists of such basic functions:

- *HanningWindow:* this function generates Hanning window of the set size. Differently from Matlab code this function is used without zero padding.
- *SignalFraming:* function forms current frame of the set size.
- *LPC:* Linear Predictive Model function which uses AutoCorrelation and Levinson-Recursion functions. In result it generates linear prediction and reflection coefficients.
- *AllPoleFilter:* function generates residue from current frame and linear prediction coefficients.
- *HOSCompute:* function for Higher Order Statistics.
- *RainerMartin:* function estimates signal-to-noise ratio using to Rainer Martin's algorithm to estimate the instantaneous SNR of speech signal.
- *GetSKR:* function computes skewness-to-kurtosis ratio.
- *PredictionError:* it estimates LPC prediction error using reflection coefficients.
- *StateMachine:* this is the function, which implements two-state machine for deciding whether frame is speech or noise.

Both programs give results by printing them to command window.
The full code of both programs can be found in a CD, attached to this report.

4.4 Implementing the C Code onto the DSK Board

4.4.1 Objective

This section describes the implementing of the ANSI-C code on TI C6713 DSK[2]. The C code files were created as explained in section 4.3.1. Several related C files are added to the code studio composer (CCS) environment. Related dependences to each C files are linked accordingly. A suitable library setting (for e.g in the program used *rts*6700.*lib*) is also linked to the whole application project. Further, optimum configuration settings are triggered and the program is built or compiled to check for any syntax errors. The program is then loaded into the C6713 DSK Kit and executed.

[2]Refer to appendix **??** for the DSK description

4.4.2 The Code Composer Studio

1. **Algorithm Test on DSK**
   The algorithm test aims to confirm the operation of the Matlab generated ANSI-C code on the DSK. The input signal (clean speech corrupted with noise) is windowed to frames. The parameters of interest are the output of the state machine model. The output from the DSK (in the form of displaying variables, flashing LED or sound production) should confirm with the output obtained from Matlab simulation.

2. **Configuration**
   One of the schemes[3] employed in configuration of CCS, prior to the test involved using a strictly non C mode. This permitted some level of tolerance during program execution to avoid unforeseen 'low-level' error. Other schemes involves:

   - Setting the RTDX [4] mode to Simulator.
   - Using far calls and data memory models
   - Using far RTS [5] calls
   - Deactivation of the of the assembly language

   The schemes were modified as shown in the figures (4.11) and (4.13).

3. **Problems during Algorithm Testing**
   It was found that the program successfully loads onto the C6713 DSK, but could not run. One reason for that was: Trouble running target processor:
   "Memory map error: write access by default....". One of the possible solutions is to increase the size of the memory or use an external memory. Figure (4.12) shows that the memory settings were extended to 0x00800000HEX. Even these attempts and others failed to solve the problem. This problem should be considered for future investigation.

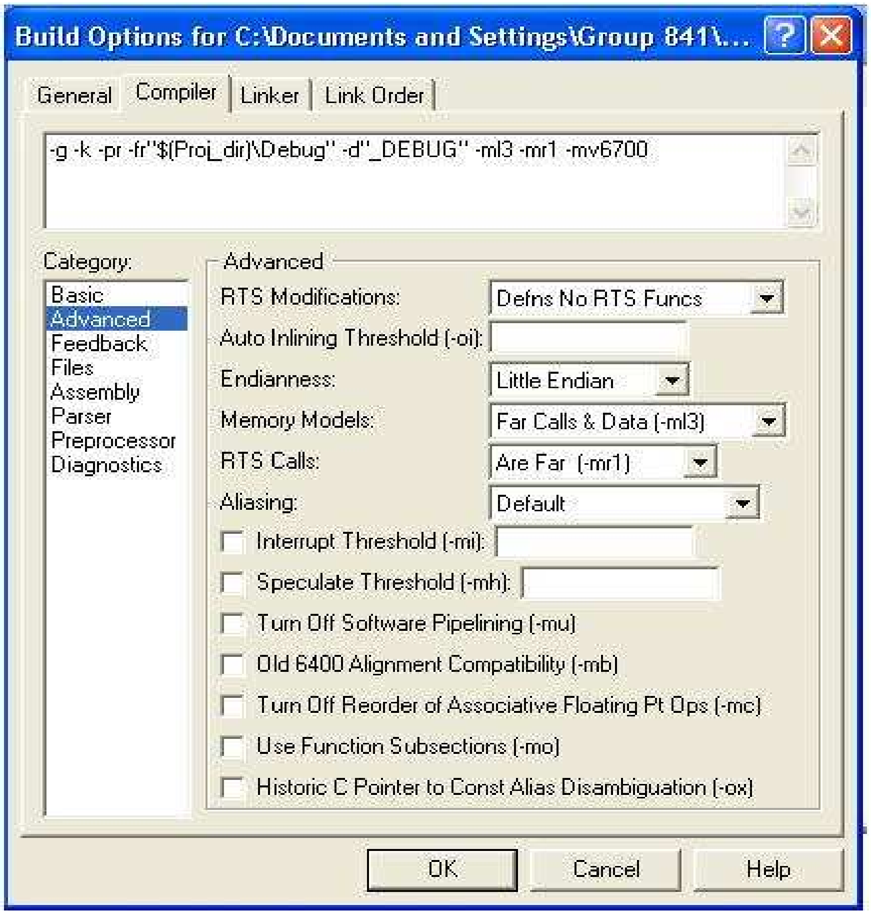
Since the implementation of Matlab converted code failed, the decision was made to write algorithms in C code manually.

4.4.3 C Code on DSK

The new HOS-VAD C code was used for the implementation onto the DSK board. In this case, several changes were made in the code:

- HOS-VAD program on DSK does not use any input data. The signal of 400 samples is used as program's global variable. The input of data from file or microphone could not be implemented on DSK.

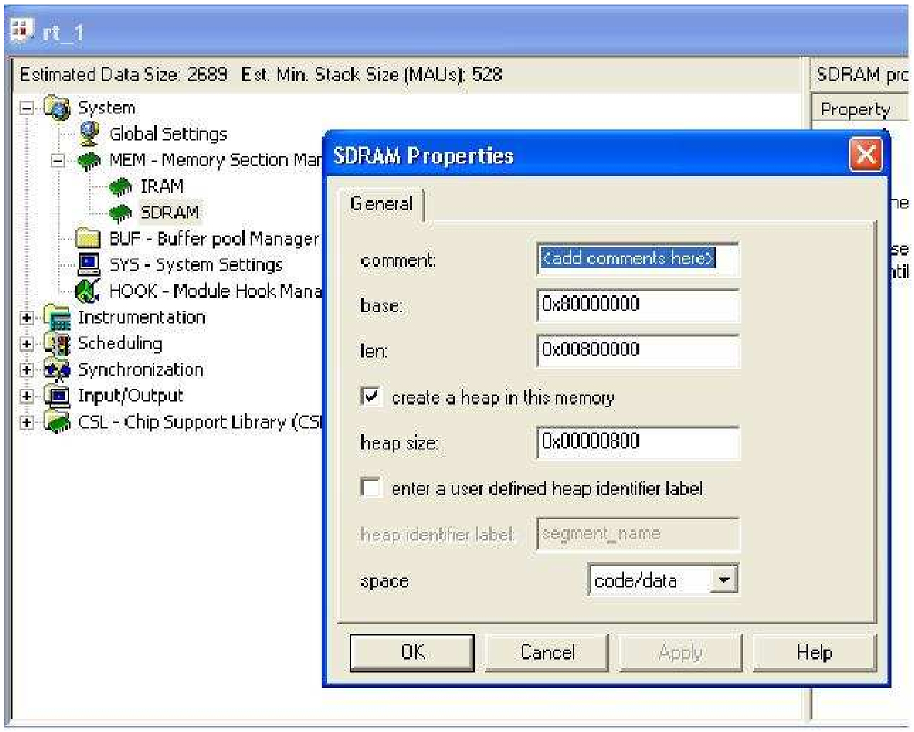
[3]build options, settings from the CCS
[4]Real Time Data Exchange
[5]Run-Time Support


Build Options for C:\Documents and Settings\Group 841\...

```
-g -k -pr -fr"$(Proj_dir)\Debug" -d"_DEBUG" -ml3 -mr1 -mv6700
```

RTS Modifications: Defns No RTS Funcs
Auto Inlining Threshold (-oi):
Endianness: Little Endian
Memory Models: Far Calls & Data (-ml3)
RTS Calls: Are Far (-mr1)
Aliasing: Default
☐ Interrupt Threshold (-mi):
☐ Speculate Threshold (-mh):
☐ Turn Off Software Pipelining (-mu)
☐ Old 6400 Alignment Compatibility (-mb)
☐ Turn Off Reorder of Associative Floating Pt Ops (-mc)
☐ Use Function Subsections (-mo)
☐ Historic C Pointer to Const Alias Disambiguation (-ox)

Figure 4.11: Configuration Scheme using Little Endian

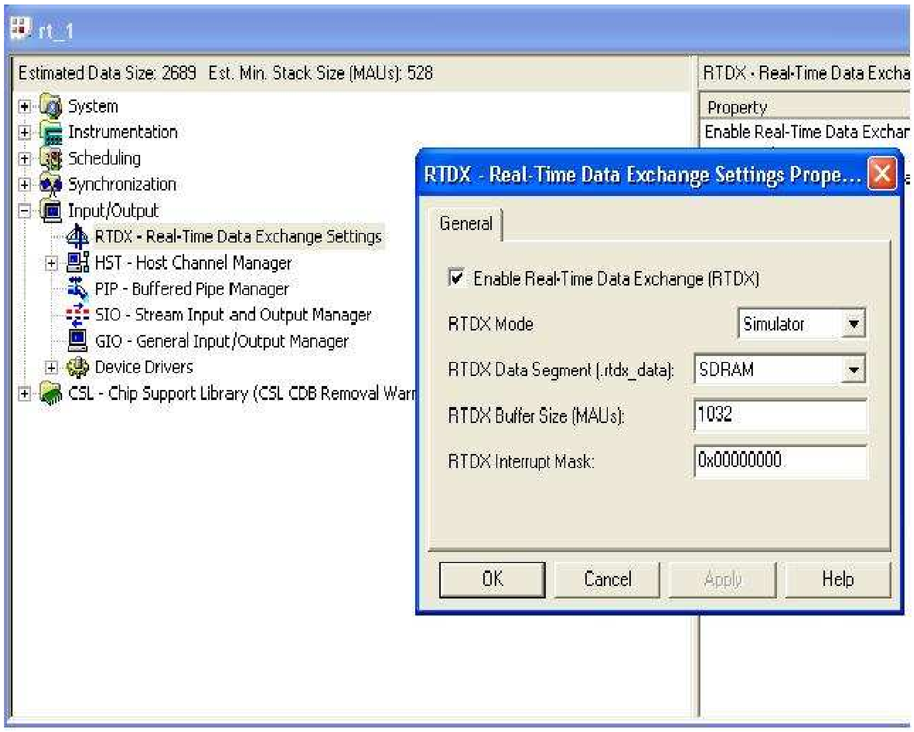


Estimated Data Size: 2689 Est. Min. Stack Size (MAUs): 528

SDRAM Properties

comment: <add comments here>
base: 0x80000000
len: 0x00800000
☑ create a heap in this memory
heap size: 0x00000800
☐ enter a user defined heap identifier label
heap identifier label: segment_name
space: code/data

Figure 4.12: Memory Settings


rt_1
System
Instrumentation
Scheduling
Synchronization
Input/Output
RTDX - Real-Time Data Exchange Settings
HST - Host Channel Manager
PIP - Buffered Pipe Manager
SIO - Stream Input and Output Manager
GIO - General Input/Output Manager
Device Drivers
CSL - Chip Support Library (CSL CDB Removal Warn
RTDX - Real-Time Data Excha
Property
Enable Real-Time Data Exchar
General
OK
Cancel
Apply
Help

Estimated Data Size: 2689 Est. Min. Stack Size (MAUs): 528

RTDX - Real-Time Data Exchange Settings Prope...

☑ Enable Real-Time Data Exchange (RTDX)
RTDX Mode: Simulator
RTDX Data Segment (.rtdx_data): SDRAM
RTDX Buffer Size (MAUs): 1032
RTDX Interrupt Mask: 0x00000000

Figure 4.13: Linker/Complier Settings

- The additional function was written for estimation of the complementary error. C program uses the in-built *erfc() function*. Differently from ANSI-C, CCS does not have this function. For this reason, HOS-VAD code for DSK was supplemented with the function for estimation of complementary error.

CCS HOS-VAD algorithm program was successfully compiled, build and then loaded onto the board for testing. After computation results were displayed in *CCS stdout window*, it was proved that the program is working correctly.

The full code of this program can be found in the CD, which is attached to this report.

4.4.4 Optimizing Program

Due to time constraints, each of the C source code could not be examined and check for potential areas for improvements [31]. However, the following solutions can be considered for improvement:

Iteration path from the memory points (Potential Pointer Aliasing Information):
It involves examining and replacing duplication of loops by checking for dependency or common use of registers. The reason is that CCS IDE often assigns more than one register to the same loop in a code, thereby creating an image of the same loop. These images need to be removed by checking the assembly code and replacing multiple registers with a single register to improve execution speed.

Alternatively, the program can pass more information to the compiler to improve its performance. *This steps will reduce cycles per iteration **5 times.***

Balancing resources with dual-data path:
Generally, CCS runs faster with the even number of binary operations (register operations). One way to balance an odd number of operations is to unroll the loop. For example, if there are 231 number of matrix columns, then instead of 231 memory accesses, the optimized project uses the even number of memory accesses, may be 462.

*This will reduce cycles per iteration by approximately **6.7 times**.*

Packed data optimization of memory bandwidth:
By analyzing a feedback path in the C code, it can be observed that the memory accesses limit the resources the most. It is found that one single 32-bit load instruction effectively performs two 16-bit loads. This is called Packed Data Processing.

*By setting the CCS to use single 32-bit load instruction effectively reduces cycles per iteration by **10 folds.***

Chapter 5

Conclusions

The objective of this project is to exploit the properties of higher-order statistics and Rainer Martin's algorithm for implementing a robust algorithm for voice activity detection and noise reduction mechanism in the presence of noise.

Firstly, the HOS also unveiled the following important properties about cumulants whose relevance goes beyond the goal of VAD application.

- Third order HOS for a Gaussian signal is zero but skewness and kurtosis of voiced speech are nonzero, so may be used as a basis for speech detection or voicing classification. When normalized by the appropriate power of the signal energy, these metrics are independent of signal levels. This makes them convenient as detectors since absolute thresholds may be used.

- Ratio of the appropriate powers of the skewness to that of the kurtosis of voiced speech is independent of signal energy and is confined to a small range for any practical range of the pitch.

- Unvoiced speech in the LPC residual may not be modeled as a harmonic process but rather as a general white process.

Secondly, the Rainer Martin's algorithm revealed the following important properties:

- Varying noise levels have a significant impact on the performance of many speech processing algorithms. It is accurate for medium to high SNR conditions but necessarily biased when no speech is present.

- A priori knowledge of noise variation and noise correlation is helpful to adapt window length and to control the estimation bias.

Unlike other reported work in the area of HOS for speech, a more fundamental approach is taken here whereby analytical derivations were first deduced based on a speech model, thus providing a basis for justifying or refuting the experimental findings.

The rationale for considering the LPC residual is its flat spectral envelope which makes the higher order cumulant derivations for speech more tractable and allows quantifying the bias and variance of the HOS estimators for Gaussian noise.

The Rainer Martin's and Higher Order Statistics algorithm (RM + HOS) were combined together and used for experimental simulations.

5.1 Implementation with Matlab and DSK

Experimental simulations demonstrates the underlying speech model are valid for voiced speech.

The relation between the (RM + HOS) metrics is used as a condition for an improved detection. Consequently, smooth noise power and HOS estimates are derived for the case of Gaussian noise and is used to quantify the likelihood of a given frame being noise. The resulting algorithm combines (RM + HOS) metrics with second-order measures, such as low-band and full-band SNR and the LPC prediction error, to classify frames into one of the two states.

Different noise scenarios were chosen for the performance of VAD algorithm for both the techniques. It can be clearly noted that RM+HOS has better performance even at low SNR values because the algorithm uses the previous samples for the detection of SNR of the present samples which makes it more adaptive for the estimation of the noise in the speech signal.This process helps in better prediction of speech and noise frames.

Compared to HOS-VAD, the proposed algorithm is based on a more analytical framework. It is computationally and conceptually complex and uses a similar parameter set, but gives more improved results. Even though the complexity is high, the results were much better even in the low SNR scenarios.

The proposed algorithm was implemented on C6713 DSK. Two-state machine results from C6713 DSK for 300 samples were similar to those obtained from Matlab simulation. The performance in noise of the two algorithms shows the (RM + HOS) based VAD has superior performance to HOS-VAD in terms of a higher probability of correct speech, noise classification and a lower probability of false classification. This fact suggests that (RM + HOS) based methods have potential in yielding VAD algorithms that would highly promote the current state of the art VAD applications. The work however does not claim these statistics to be superior in and by themselves to second-order statistics. They provide additional information about the signal that is immune to the presence of noise, and that makes them particularly effective in low SNR applications. Clearly, successful algorithms are those that can combine the two approaches and exploit the advantages of both.

5.2 Future work

This area includes:

- Investigating the combination of more metrics and tuning the algorithm with speech recorded in more diverse noise environments.
- Implementation of the frequency version of Rainer Martin's algorithm using spectral subtraction techinque.
- Examining reasons why ANSI-C code (converted with Matlab) did not execute on the DSK even though it compiled and ran without error on a standard ANSI-C compiler.
- Developing the synthesis filter for (RM + HOS) based VAD (if it theoritically possible).
- Optimizing the DSP compatible ANSI-C code of the algorithm to ensure faster execution time.

Appendix A

Higher Order Spectra and Statistics

The estimation of the power spectral density or simply the power spectrum of discrete-time deterministic or stochastic signals has been a useful tool in digital signal processing. Power spectrum estimation techniques have proved essential to the creation of advanced radar, sonar, communication, speech, biomedical, geophysical and other data processing systems [9].

In power spectrum estimation, the signal under consideration is processed in such a way that the distribution of power among its frequency components is estimated. Thus the phase relations between frequency components are suppressed. The information contained in the power spectrum is essentially that which is present in the autocorrelation sequence; this is sufficient for the complete statistical description of the Gaussian signal. However, in the practical situations we look beyond the power spectrum of a signal to extract information regarding deviations from Gaussianity and the phase relations.

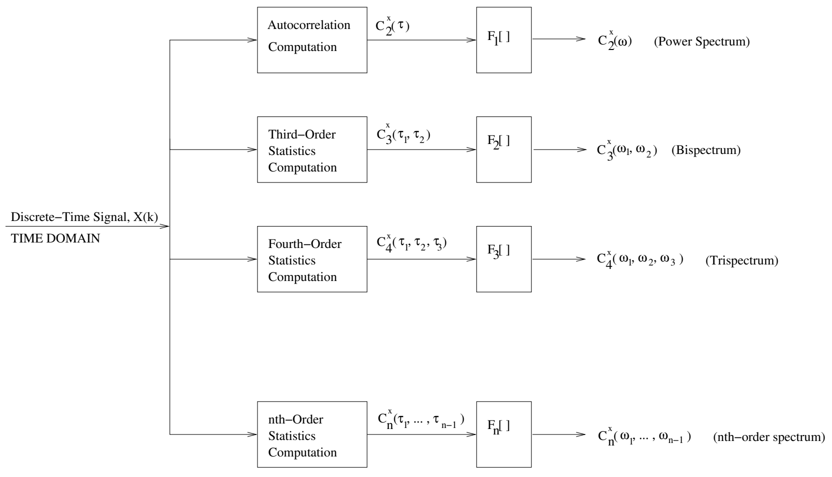

Discrete−Time Signal, X(k)
TIME DOMAIN
Autocorrelation Computation
$C_2^x(\tau)$
$F_1[\ ]$
$C_2^x(\omega)$
(Power Spectrum)
Third−Order Statistics Computation
$C_3^x(\tau_1, \tau_2)$
$F_2[\ ]$
$C_3^x(\omega_1, \omega_2)$
(Bispectrum)
Fourth−Order Statistics Computation
$C_4^x(\tau_1, \tau_2, \tau_3)$
$F_3[\ ]$
$C_4^x(\omega_1, \omega_2, \omega_3)$
(Trispectrum)
nth−Order Statistics Computation
$C_n^x(\tau_1, \dots, \tau_{n-1})$
$F_n[\ ]$
$C_n^x(\omega_1, \dots, \omega_{n-1})$
(nth−order spectrum)

Figure A.1: The Higher-order Spectra Classification Map of the Discrete Signal X(k). F[ ] denotes n-dimensional Fourier Transform

Higher order spectra(also known as polyspectra) defined in terms of higher order statistics (cumulants) of a signal, do contain such information. Particular cases of higher order spectra are the third-order spectrum also called bispectrum which is, by definition, the Fourier transform of the third-order statistics, and the trispectrum (fourth-order spectrum) which is the Fourier trasnform of the fourth-order statistics of a stationary signal. The power spectrum is, a member of a class of higher order spectra classification map of a given discrete-time signal. Higher-order statistics and spectra of a signal can be defined in terms of moments and cumulants, Moments and moment spectra can be very useful in the analysis of deterministic signals whereas cumulants and cumulant spectra are of great importance in the analysis of stochastic signals [22].

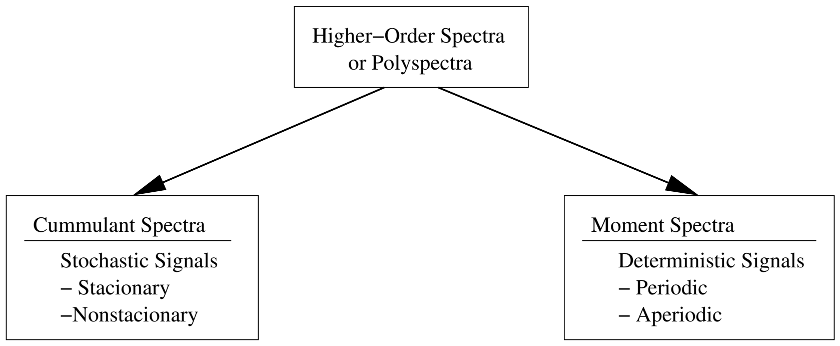


Figure A.2: The Polysepctra Classification Map

The motivations behind the use of higher-order spectra in signal processing are [22].

1. Suppress additive colored Gaussian noise of unknown power spectrum, the bispectrum also suppresses non-Gaussian noise with symmetric probability density function(pdf).
2. Identify non-minimum phase signals.
3. Extract information due to deviations from Gaussianity.
4. Detect and characterize nonlinear properties in signals as well as identify nonlinear systems.

A.1 Applications

The applications of polyspectra [22] are in the fields of oceanography, geophysics, sonar, comunications, biomedicine, speech processing, radioastonomy, image processing, fluid mechanics, economic time series, plasma physics, sunspot data and so on. Procedures were developed based on polyspectra for deconvolution(or equalization) and signal detection, for the identification of nonlinear; nonminimum phase; and spike-array type processes; for parameter estimation; and detection of quadratic phase coupling, and for detection of aliasing in discrete-time stochastic signals.

A.2 Definitions and Properties

This section gives the introduction of the definitions, properties and computation of higher-order statistics, i.e., moments and cumulants, and their corresponding higher-order spectra [1][9][10].

If $X(k), k = 0, \pm 1, \pm 2, \pm 3, \ldots$ is a real stationary discrete-time signal and its moments up to order n exist, then

$$m_n^x(\tau_1, \tau_2, \ldots, \tau_{n-1}) = E\{X(k)X(k+\tau_1)\ldots X(k+\tau_{n-1})\} \tag{A.1}$$

represents the nth order moment function of the stationary signal, which depends only on the time difference $\tau_1, \tau_2, \ldots, \tau_{n-1}, \tau_i = 0, + - 1, \ldots$ for all i. Clearly, the $2nd$-order moment function, $m_2^x(\tau_1)$, is the autocorrelation of $X(k)$ whereas $m_3^x(\tau_1, \tau_2)$ and $m_4^x(\tau_1, \tau_2, \tau_3)$ are the $3rd$- and $4th$-order moments, respectively.

The nth-order cumulant function of a non-Gaussian stationary random signal $X(k)$ can be writen as (for $n = 3, 4$ only):

$$c_n^x(\tau_1, \tau_2, \ldots, \tau_n - 1) = m_n^x(\tau_1, \tau_2, \ldots, \tau_{n-1}) - m_n^G(\tau_1, \tau_2, \ldots, \tau_{n-1}) \tag{A.2}$$

where $m_n^x(\tau_1, \ldots, \tau_{n-1})$ is the nth-order moment function of $X(k)$ and $m_n^G(\tau_1, \ldots, \tau_{n-1})$ is the nth-order moment function of an equivalent Gaussian signal that has the same mean value and autocorrelation sequence as $X(k)$. For Gaussian signal,

$$m_n^x(\tau_1, \ldots, \tau_{n-1}) = m_n^G(\tau_1, \ldots, \tau_{n-1}) \tag{A.3}$$

and thus $c_n^x(\tau_1, \tau_2, \ldots, \tau_{n-1}) = 0$. Although the equation (A.3) is only true for orders $n = 3$ and 4, $c_n^x(\tau_1, \tau_2, \ldots, \tau_{n-1}) = 0$ for all n if $X(k)$ is Gaussian. Relationships between moment and cumulant sequences of $X(k)$ exist for orders $n = 1, 2, 3, 4$.

1st-order cumulants:

$$c_1^x = m_1^x = E\{X(k)\} \quad \text{(mean value)} \tag{A.4}$$

2nd-order cumulants:

$$\begin{aligned} c_2^x(\tau_1) &= m_2^x(\tau_1) - (m_1^x)^2 \qquad \text{(covariance sequence)} \\ &= m_2^x(-\tau_1) - (m_1^x)^2 = c_2^x(-\tau_1) \end{aligned} \tag{A.5}$$

where $m_2^x(-\tau_1)$ is the autocorrelation sequence. Thus, the $2nd$ order cumulant sequence is the *covariance* while the $2nd$-order moment sequence is the *autocorrelation*.

3rd-order cumulants:

$$c_3^x(\tau_1,\tau_2) = m_3^x(\tau_1,\tau_2) - m_1^x[m_2^x(\tau_1) + m_2^x(\tau_2) + m_2^x(\tau_1-\tau_2)] + 2(m_1^x)^3 \tag{A.6}$$

where $m_3^x(\tau_1,\tau_2)$ is the $3rd$-order moment sequence.

4th-order cumulants:

$$\begin{aligned} c_4^x(\tau_1,\tau_2) = {} & m_4^x(\tau_1,\tau_2,\tau_3) - m_2^x(\tau_1).m_2^x(\tau_3-\tau_2) \\ & m_2^x(\tau_2).m_2^x(\tau_3-\tau_1) \\ & m_2^x(\tau_3).m_2^x(\tau_2-\tau_1) \\ & m_1^x[m_3^x(\tau_2-\tau_1,\tau_3,\tau_1) + m_3^x(\tau_2,\tau_3) \\ & (m_3^x(\tau_2,\tau_4) + m_3^x(\tau_1,\tau_2)] \\ & (m_2^x)^2[m_1^x(\tau_1) + m_2^x(\tau_2) \\ & +m_2^x(\tau_3) + m_2^x(\tau_3-\tau_1) + m_2^x(\tau_3-\tau_2) \\ & +m_2^x(\tau_2-\tau_1)] - 6(m_1^x)^4 \end{aligned} \tag{A.7}$$

If the signal $X(k)$ is zero mean $m_1^x = 0$, and follows from the equations (A.5),(A.6) that the second and third order cumulants are identical to the second and third order moments, respectively. But to generate the fourth order cumulants, we need knowledge of the fourth-order and second-order moments in equation (A.7).

$$\begin{aligned} c_4^x(\tau_1,\tau_2) = {} & m_4^x(\tau_1,\tau_2,\tau_3) - m_2^x(\tau_1).m_2^x(\tau_3-\tau_2) \\ & m_2^x(\tau_2).m_2^x(\tau_3-\tau_1) - m_2^x(\tau_3).m_2^x(\tau_2-\tau_1). \end{aligned} \tag{A.8}$$

By putting $\tau_1 = \tau_2 = \tau_3 = 0$ in equations (A.5), (A.6), (A.7) and assuming $m_1^x = 0$, we get

$$\begin{aligned} \gamma_2^x &= E\{x^2(k)\} = c_2^x(0) \quad \text{(variance)} \\ \gamma_3^x &= E\{x^3(k)\} = c_3^x(0,0) \quad \text{(skewness)} \\ \gamma_4^x &= E\{x^4(k)\} - 3[\gamma_2^x]^2 = c_4^x(0,0,0) \quad \text{(kurtosis)} \end{aligned} \tag{A.9}$$

Normalized kurtosis is defined as $\gamma_4^x/[\gamma_2^x]^2$. Equation (A.9) gives the variance, skewness and kurtosis measures in terms of cumulants at zero lags.

Appendix B

Design Considerations

The overall system analysis and design strategy is shown in the figure (B.1). The design process is divided into five domains. Beginning with the problem analysis, where indepth analysis of the problem is presented. As the time advances, the problem and the requirements specified in problem domain are represented in application, algorithmic and architectural domains. In qualitative domain, the quality control analysis of the finished system is presented, defining the quality control criteria and looks at the external system. Finally, conclusion analyses the design process in all aspects. The overall design process and the interaction among the different domains is iterative.

Problem domain analysis deals with the information required to design the system, the need for such a design and the main purpose of the design. The modeling of problem domain, allows many possibilities which can be used directly or indirectly in order to collect information about the problematic situation.

Application domain mentions about the system usage and analysis of the system. The requirement includes system functions and interfaces with its environment. Application domain and the problem domain have a strong interaction between them. Problem domain analysis gives the requirement to model the system behaviour and thus the system functions and interface requirements are defined. Due to a strong interaction between the two domains, the order of the domain analysis can be interchanged, but it depends on real situation and user requirements.

Algorithmic domain answers the question about the main task of the system. These algorithms developed and analyzed are the heart of the system. After careful analysis of the problem and application domain, algorithmic domain follows. Algorithms are developed keeping in view of the requirements, which are generated in the problem and application domains. Algorithm development or selection of a suitable algorithm from different available algorithms mainly depend upon the problematic situation description in problem domain analysis. As the project is with respect to Voice Activity Detection using Higher order statistics, the concept of higher order statistics is explained. All the

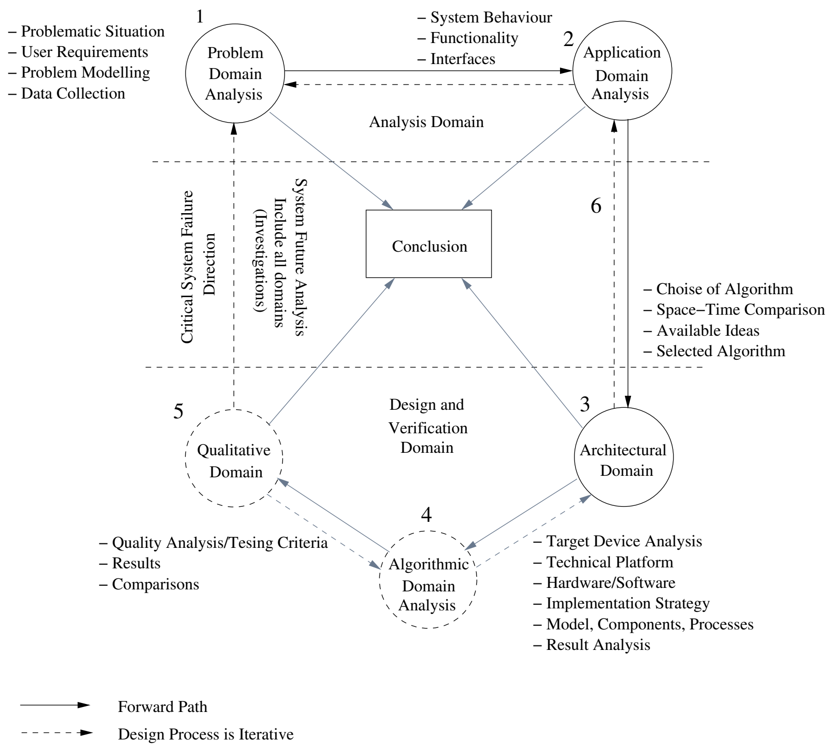


Figure B.1: Design Model

considered algorithms are simulated using Matlab on system models and real signals and the results were analyzed.

Architectural domain gives the information regarding the technical platform. This domain shows the mapping of the algorithm in a specific architecture TMS320C6713. For a selected algorithm there may be many solutions. The selection of the architecture even depends on the specific application and the design. In the project implementation there was no architecture constraint as the DSK kit was provided. Both algorithmic and architectural domains observe the system from inside.

Appendix C

Linear Prediction Coding (LPC)

LPC is considered as one of the most powerful techniques for speech analysis. In fact, this technique is the basis of other more recent and sophisticated algorithms that are used for estimating speech parameters, e.g., pitch, formants, spectra, vocal tract and low bit representations of speech. The basic principle of linear prediction, states that speech can be modeled as the output of a linear, time-varying system excited by either periodic pulses or random noise figure (C.1). These two kinds of acoustic sources are called voiced and unvoiced respectively. In this sense, voiced emissions are those generated by the vibration of the vocal cords in the presence of an airflow and unvoiced sounds are those generated when the vocal cords are relaxed.

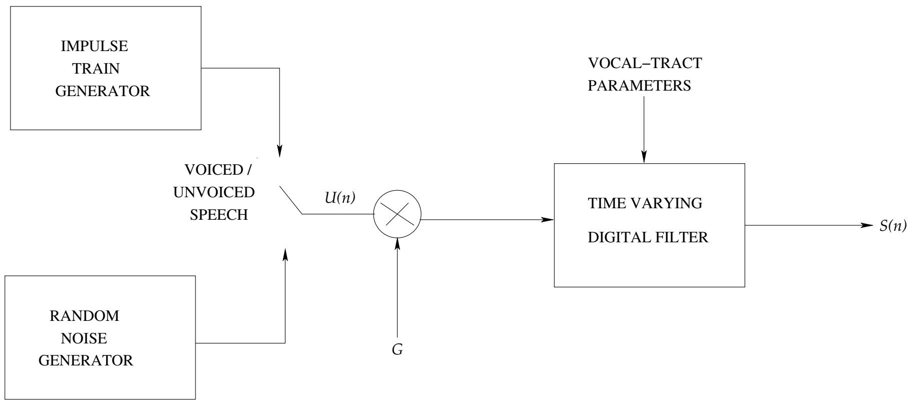


Figure C.1: Speech Synthesis Generated with the LPC Model

Consider the following equation,

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + \ldots + a_p s(n-p) \tag{C.1}$$

where $a_1, a_2, \ldots, a_p$ are constant coefficients. The previous equation can be transformed by including an excitation term $Gu(n)$ to:

$$s(n) = \sum_{i=1}^{p} a_i s(n-i) + Gu(n) \tag{C.2}$$

where G is the gain and $u(n)$ the normalized excitation. Transforming equation (C.2) to the z-domain we obtain

$$S(z) = \sum_{i=1}^{p} a_i z^{-i} S(z) + GU(z) \tag{C.3}$$

and consequently the transfer function will be:

$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-1}} = \frac{1}{A(z)} \tag{C.4}$$

that corresponds to the transfer function of a digital time varying filter. The main parameters that can be obtained with the LPC model are: the classification of voiced/unvoiced, the pitch period, the gain and the coefficients $a_1, \ldots, a_p$. It is important to note that, the higher the order of the model is, the best the all-pole model allows a good representation of the speech sounds. A linear predictor with coefficients a_k is defined with the polynomial $P(z)$:

$$P(z) = \sum_{k=1}^{p} a_k z^{-k} \tag{C.5}$$

whose output is:

$$\widetilde{s}(n) = \sum_{k=1}^{p} a_k s(n-k) \tag{C.6}$$

The prediction error e(n) is defined as:

$$e(n) = s(n) - \widetilde{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k) \tag{C.7}$$

that is the output of a system $A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k}$ and if $a_k = a_k$ we have then $H(z) = \frac{G}{A(z)}$. The main goal is to obtain the set coefficients a_k that minimizes the square of the prediction error in a short segment of speech (typically *10-30ms*) frames. The mean short time prediction error per frame is defined as:

$$E_n = \sum_{m} e_n^2(m) = [s_n(m) - \sum_{k=1}^{p} a_k s_n(m-k)]^2 \tag{C.8}$$

where $s_n(m)$ is a segment of speech selected in the neighbourhood of a sample n : $s_n(m) = s(m+n)$, the value of the coefficients a_k that minimizes the error E_n can be obtained considering

$\frac{dE_n}{da_i} = 0, \qquad i = 1, 2, \ldots, p$ that result in the next equation:

$$\sum_m s_n(m-i)s_n(m) = \sum_{k=1}^{p} a_k^l \sum_m s_n(m-i)s_n(m-k), 1 \leq i \leq p \tag{C.9}$$

where a_k^l are the values of a_k that minimizes E_n.

Defining $\phi_n(i,k) = \sum_m s_n(m-i)s_n(m-k)$, equation (C.9) can be written as

$$\sum_{k=1}^{p} a_k \phi_n(i,k) = \phi_n(i,0), \qquad i = 1, 2, \ldots, , p \tag{C.10}$$

This is a system of p equations with p variables that can be solved to find the a_k coefficients for the segments s_m. It can be demonstrated that:

$$E_n = \sum_m s_n^2(m) - \sum_{k=1}^{p} a_k \sum_{k=1}^{p} s_n(m-k) \tag{C.11}$$

and in the compact form:

$$E_n = \phi_n(0,0) - \sum_{k=1}^{p} a_k \phi_n(0,k) \tag{C.12}$$

Now, the values $E_n(i,k)$ have to be obtained for $1 \leq i \leq p$, $1 \leq k \leq p$, and the a_k coefficients are obtained by solving equation (C.10). Equation (C.12) system can be solved using the following:

- Autocorrelation Method
- Covariance Method

C.1 All Pole Model

Linear prediction and autoregressive modeling are two different problems that can yield the same numerical results. In both the cases, the ultimate goal is to determine the parameters of a linear filter. However, the filter used in each of the problems is different.

In the case of linear prediction, the intention is to determine an FIR filter that can optimally predict future samples of an autoregressive process based on a linear combination of past samples. The difference between the actual autoregressive signal and the predicted signal is called the prediction error. Ideally, this error is white noise.

For the case of autoregressive modeling, the intention is to determine an all-pole IIR filter, that when excited with white noise produces a signal with the same statistics as the auto-regressive process that is tried to model.

Consider the equation (C.13):

$$s(n) = -\sum_{k=1}^{p} a_k s(n-k) + G\sum_{l=0}^{q} b_l u(n-l) \qquad 1 \leq k \leq p, \qquad 1 \leq l \leq q \tag{C.13}$$

if $b_l = 0$, then the model is referred to as an all pole model or autoregressive model(AR) model. (If $a_k = 0$, it becomes an all zero model). In such a model, the signal $s[n]$ can be assumed as a linear combination of the previous values and some input $u[n]$:

$$s(n) = -\sum_{k=1}^{p} a_k s(n-k) + Gu(n) \tag{C.14}$$

Where G is the gain factor. We can also reduce the transfer function $H(z)$ in (C.12) to an all pole model transfer function:

$$H(z) = \frac{S(z)}{U(z)} = \frac{G}{1 + \sum_{k=1}^{p} a_k z^{-k}} = \frac{G}{A(z)} \tag{C.15}$$

Appendix D

TIA Database

Telecommmunications Industry Association *(TIA) standard TIA/EIA-136-250 describes definitions, methods of measurement, minimum delay and performance requirements for voice activity detectors(VADs). This standard applies to mobile stations operating in the discontinous transmission(DTX) mode.*

This standard consists of ten speech data files, ten truth mark files corresponding to each speech file and four background noise files. This standard defines the minimum performance levels for the VAD; but the manufacturer should attempt to provide the highest possible level of performance [8].

D.1 Test Cases

Test cases have been chosen to exercise the range of VAD processing. Ten speech data files and four noise data files are combined in various ratios to yield 80 cases. Normalization values are in units of dBov[1] and SNR values in units of dB relative to speech.

Source Speech Material
The speech material consists of 10 conversational data files of which five are male and five are female speakers. Each file contains 16 bit PCM data, stored in high-byte/low-byte word format, sampled at a rate of 8kHz, Modified-IRS[2] filtered, and normalized to an average level of -26dBov.

Source Noise Material
The noise material consists of four data files. The files contain noise stored as 16-bit PCM words in high-byte/low-byte format, sampled at a rate of 8kHz, Modified-IRS filtered, and normalized to an average level of -26dBov.

[1]Sound level in decibels with respect to 16-bit overload

[2]Modified Intermediate Reference System which refers to the characteristic spectral shaping of speech signals by the telephone network

D.2 Test Procedure

The section describes the procedure to verify that the VAD implementation meets the minimum performance requirements. The procedure for testing VAD for compliance to the standard consists of the following steps:

1. Generate a test data file of the 80 test cases.
2. Process the test data files with the VAD and produce the mark files.
3. Generate VAD performance metrics for each of the test cases.
4. Evaluate performance metrics for compliance with thresholds.

A software tool was provided to perform the file generation, performance metric calculation, and evaluation (steps 1,3,4 respectively). The C source code file for the objective VAD evaluation tool is ove.c. Step 2 is the execution of the VAD-HOS algorithm.

Generation of Performance metrics
The speech frames are divided into three categories: onset, steady-state, and offsets. Onsets are made up from the first three frames of speech in the beginning of the utterance, offsets are the last three frames, and the steady-state speech frames are those in between. Counts are kept on the number of times the VAD mark agrees with the truth marks for the each category as well as the number of frames in each category. The counts are collected only when the local SNR exceeds -15dB. The local SNRs are calculated with the equation:

$$SNR(n) = 10\log_{10}\Big(e_{speech}(n)/e_{noise}(n)\Big) \tag{D.1}$$

where n is current frame index,
$e_{speech}(n)$ is energy of the current speech at frame n,
$e_{noise}(n)$ is energy of the current noise at frame n.

A delta voice-activity factor (δ VAF) metric is computed as the difference between the VAF and the true VAF divided by the true VAF, where the voice-activity factor is the number of frames called speech divided by the total number of frames [8]. The four performance metrics are:

- Probability of clipping speech onsets.
- Probability of detecting steady-state speech.
- Probability of clipping speech offsets.
- Normalized difference in the VAD's voice-activity factor from truth.

Evaluation of Performance metrics

The performance statistices are accumulated to produce 12 evaluation categories, one for each normalization and SNR level combination. Each categories has four metrics listed in the above section, resulting 48 performance metrics. Each metrics represents the average of over all cases with the same normalization and SNR.

The evaluation mode of the *ove* software tool reads the concatenated output generated by the performance tool and computes evaluation metrics [8]. The evaluation tool then outputs 3 tables in the following order: the thresholds, the evaluation metrics for the VAD, and a table indicating pass or fail for each metric.If the metric meets the threshold, the character p is output; and, if not, the difference between the VAD and threshold is printed. The clipping and voice-activity thresholds are maximums, while the detection thresholds are minimums.

Appendix E

TMS320C6000 Platform

E.1 Overview

The TMS320C6000 platform consists of the TMS320C64x and TMS320C62x fixed-point generations as well as the TMS320C67x floating-point generation. These platforms are for broadband infrastructure, performance audio and imaging applications. The C6000 DSP platform's performance ranges from 1200 to 8000 MIPS for fixed-point and 600 to 1800 MFLOPS for floating point [28].

E.1.1 Platform Highlights

- Optimized for good performance and of use in high-level language programming with three device generations. Fixed-point performance ranges from 1200 to 8000 MIPS and floating-point performance from 600 to 1350 MFLOPS.
- Memory, peripherals and co-processor are combined to meet the needs of targeted broadband infrastructure, performance audio and imaging applications.
- Software compatibility across all C6000 devices.

E.1.2 Code-Compatible Generations

The TMS320C6000 platform consists of three code-compatible device generations:

TMS320C64x: The C64x fixed-point DSPs has clock rates of up to $1GHz$, C64x DSPs can process information at rates up to 8000 MIPS. The built in extensions include new instructions to accelerate performance in key application areas such as digital communications infrastructure, video and image processing [28].

TMS320C62x: These first-generation fixed-point DSPs enables new equipments and energizes existing implementations for multi-channel, multi-function applications, such as wireless base stations, remote access servers (RAS), digital subscriber loop (xDSL) systems, personalized home security systems, advanced imaging/biometrics, industrial scanners, precision instrumentation and multi-channel telephony systems.

TMS320C67x: The C67x floating-point DSPs has the speed, precision, power savings and dynamic range to meet a variety of design needs. These DSPs are used in applications like audio, medical imaging, instrumentation and automotive.

E.1.3 C Compiler

The C6000 DSP platform gives a good performance C language engine with a compiler for the architecture to sustain maximum performance while speeding design development time for high-performance applications. The C compiler/optimization tools balances code size and performance to meet the needs of the application.

E.1.4 C6000 Signal Processing Libraries and Peripherals Drivers

The Signal Processing and the Chip support libraries contain a collection of high-level, optimized DSP function modules and help to achieve good performance than standard ANSI C code.

E.2 DSP Starter Kit

The TMS320C6713 DSP Starter Kit (DSK) developed jointly with Spectrum Digital is designed to speed the development of high precision applications based on TI's TMS320C6000 floating point DSP generation. Can be used in the following areas: speech compression/decompression, speech recognition, text-to-speech, fax/data conversion, modems, protocol conversions, tone generation/detection, and echo cancellation [29].

The C6713 DSK tools includes the simulators from TI and access to the Analysis Toolkit via Update Advisor which features the Cache Analysis tool and Multi-Event Profiler. Using Cache Analysis, developers improve the performance of their application by optimizing cache usage. By providing a graphical view of the on-chip cache activity over time can determine whether the code is using the on-chip cache to get good performance.

The C6713 DSK uses Real Time Data Exchange (RTDX) for Host and Target communications. The DSK includes the Run Time Support libraries and utilities such as Flashburn to program flash, Update Advisor to download tools, utilities and software and a power on self test and diagnostic utility to ensure the DSK is operating correctly. The full contents of the kit include [29]:

- C6713 DSP Development Board with 512K Flash and 8MB SDRAM
- C6713 DSK Code Composer Studio v2.2 IDE including the Fast Simulators and access to Analysis Toolkit on Update Advisor

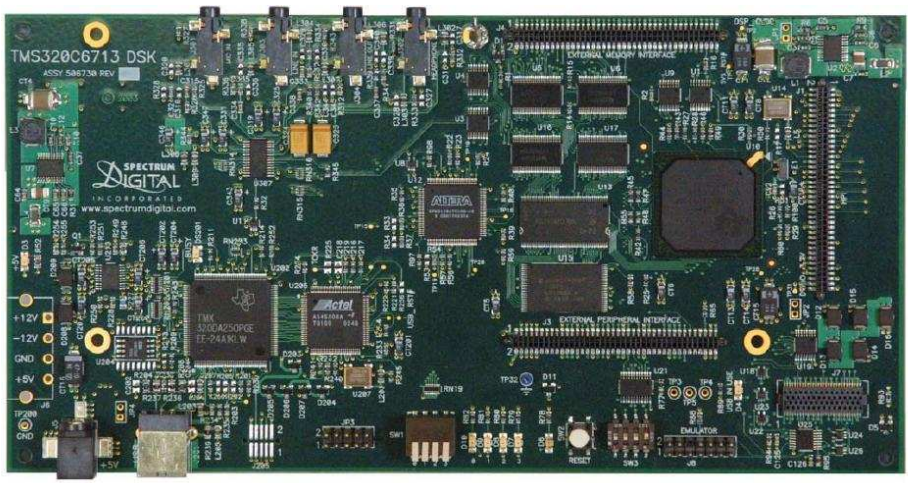


Figure E.1: DSP Starter Kit

- Quick Start Guide
- Technical Reference
- Customer Support Guide
- USB Cable
- Universal Power Supply
- AC Power Cord(s)

E.2.1 Features

The DSK features the TMS320C6713 DSP, a 225 MHz device delivering up to 1800 million instructions per second (MIPs) and 1350 MFLOPS. Other hardware features of the TMS320C6713 DSK board include [29]:

- Embedded JTAG support via USB
- High-quality 24-bit stereo codec, TI TLV320AIC23 codec
- Four 3.5mm audio jacks for microphone, line in, speaker and line out
- 512K words of Flash and 8 MB SDRAM
- Expansion port connector for plug-in modules
- On-board standard IEEE 1149.1 JTAG interface for optional emulator debug
- 4 user definable LEDs
- 4 position dip switch, user definable
- +5V universal power supply

E.2.2 Physical Specifications

The TMS320C6713 DSK is designed on a multi-layer printed circuit board using surface mount technology. The printed circuit board measures 8.75 x 4.5 inches(222 x 115 mm.). The C6713 DSK operates off $+5$ volts at $400mA$. Its operating temperature range is 0-70deg C.

E.2.3 Software

The TMS32C6713 DSP can be used through TI's Code Composer Studio DSK development platform the tool which run on Windows environment. Code Composer Studio features for the TMS320C6713 DSK include [29]:

- A Integrated Development Environment (IDE), optimizing C/C++ compiler assembler, linker, debugger, and DSP BIOS, an editor for code creation, data visualization, a profiler and a flexible project manager.
- DSP/BIOS real-time kernel
- Target error recovery software

Appendix F

Speech Signal-Important Features

F.1 Speech Generation

Figure (F.1) portrays a medium saggital section of the speech system in which we view the anatomy midway through the upper torso as we look on from the right side. The gross components of the system are the lungs, trachea (windpipe), larynx (organ of speech production), pharyngeal cavity (throat), oral or buccal cavity (mouth), and nasal cavity (nose). The pharyngeal and oral cavities are usually grouped into one unit referred to as the vocal tract, and the nasal cavity is often called the nasal tract. Accordingly, the vocal tract begins at the output of the larynx (vocal cords, or glottis) and terminates at the input to the lips. The nasal tract begins at the velum and ends at the nostrils. When the velum is lowered, the nasal tract is acoustically coupled to the vocal tract to produce the nasal sounds of speech. Air enters the lungs via the normal breathing mechanism. As air is expelled from the lungs through the trachea, the tensed vocal cords within the larynx are caused to vibrate by the air flow. The air flow is chopped into quasi-periodic pulses which are then modulated in frequency in passing through the throat, the oral cavity, and possibly nasal cavity. Depending on the positions of the various articulators (i.e., jaw, tongue, velum, lips, mouth), different sounds are produced.

The lungs and the associated muscles act as the source of air for exciting the vocal mechanism. The muscle force pushes air out of the lungs and through the trachea. When the vocal cords are tensed, the air flow causes them to vibrate, producing so-called voiced speech sounds. When the vocal cords are relaxed, in order to produce a sound, the air flow either must pass through a constriction in the vocal tract and thereby become turbulent, producing so-called unvoiced sounds, or it can build up pressure behind a point of the total closure within the vocal tract, and when the closure is opened, the pressure is suddenly and abruptly release, causing a brief transient sound.

The three blocks seen in F.2, Generator, Vocal tract, and Radiation are indicated. A switch is shown between the Generator and the Vocal Tract, which separates the generation of voiced and unvoiced speech.

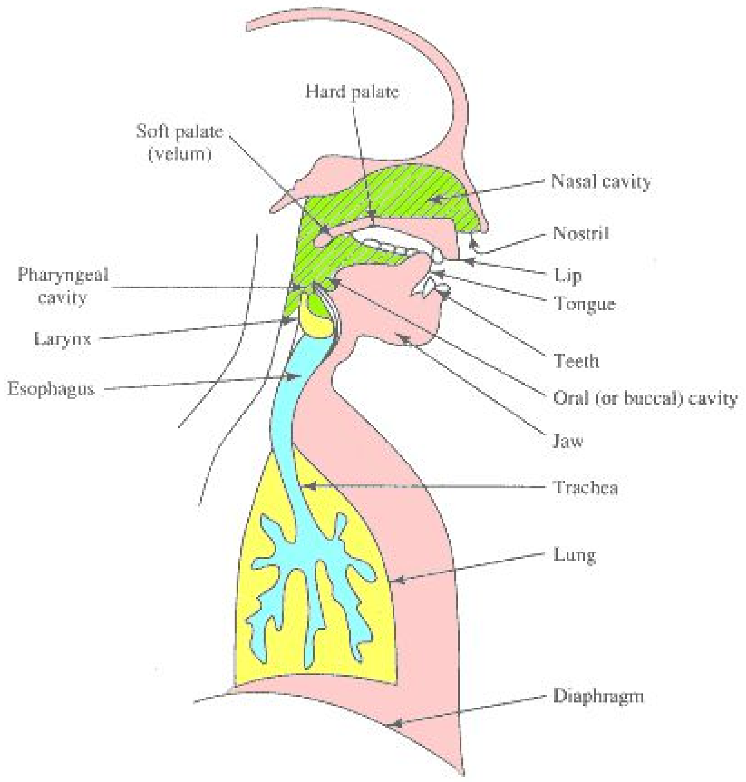


Figure F.1: Schematic View of Human Speech Production Mechanism

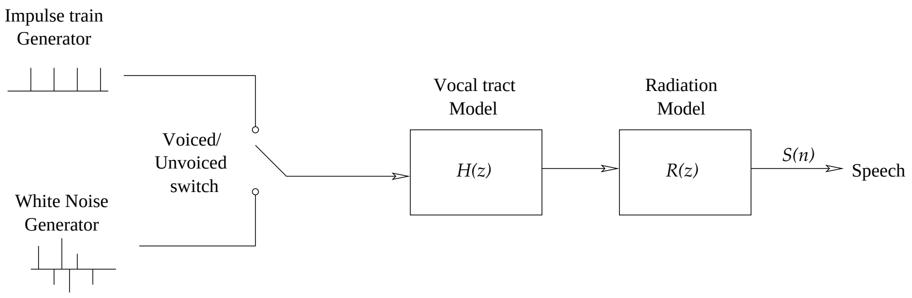


Figure F.2: Block Diagram of Speech Production.

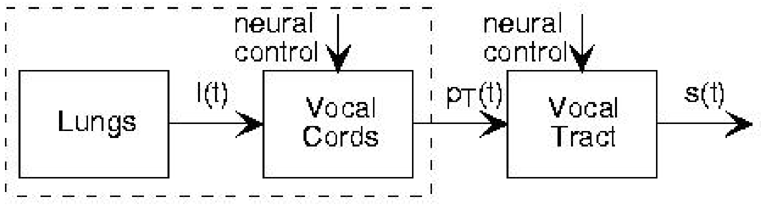


Figure F.3: The System Model for the Vocal Tract.

F.2 Pitch and Formants

The period of the vocal cord's output for vowels is known as the pitch. Vocal cord tension is governed by a control input to the musculature; in system's models we represent control inputs as signals coming into the top or bottom of the system. Certainly in the case of speech and in many other cases as well, it is the control input that carries information, impressing it on the system's output. The change of signal structure resulting from varying the control input enables information to be conveyed by the signal, a process generically known as modulation.

The vocal cords' periodic output can be well described by the periodic pulse train $p_T(t)$, with T denoting the pitch period. The spectrum of this signal contains harmonics of the frequency $1/T$, what is known as the pitch frequency or the fundamental frequency F_0. Before puberty, pitch frequency for normal speech ranges between $150 - 400$ Hz for both males and females. After puberty, the vocal cords of males undergo a physical change, which has the effect of lowering their pitch frequency to the range $80 - 160Hz$ [27]. If we could examine the vocal cord output, we could probably discern whether the speaker was male or female. This difference is also readily apparent in the speech signal itself. model of vocal tract.

In the figure (F.3) The signals $l(t), p_T(t)$, and $s(t)$, are the air pressure provided by the lungs, the periodic pulse output provided by the vocal cords, and the speech output respectively. Control signals from the brain are shown as entering the systems from the top. Clearly, these come from the same source, but for modeling purposes we describe them separately since they control different aspects of the speech signal.

Simplifying the speech modeling effort and assuming that the pitch period is constant, we collapse the vocal-cord-lung system as a simple source that produces the periodic pulse signal (F.3). The sound pressure signal thus produced enters the mouth behind the tongue, creates acoustic disturbances, and exits primarily through the lips and to some extent through the nose. Speech specialists tend to name the mouth, tongue, teeth, lips, and nasal cavity the vocal tract. The physics governing the sound disturbances produced in the vocal tract and those of an organ pipe are quite similar. Whereas the organ pipe has the simple physical structure of a straight tube, the cross-section of the vocal tract varies along its length because of the positions of the tongue, teeth, and lips. These

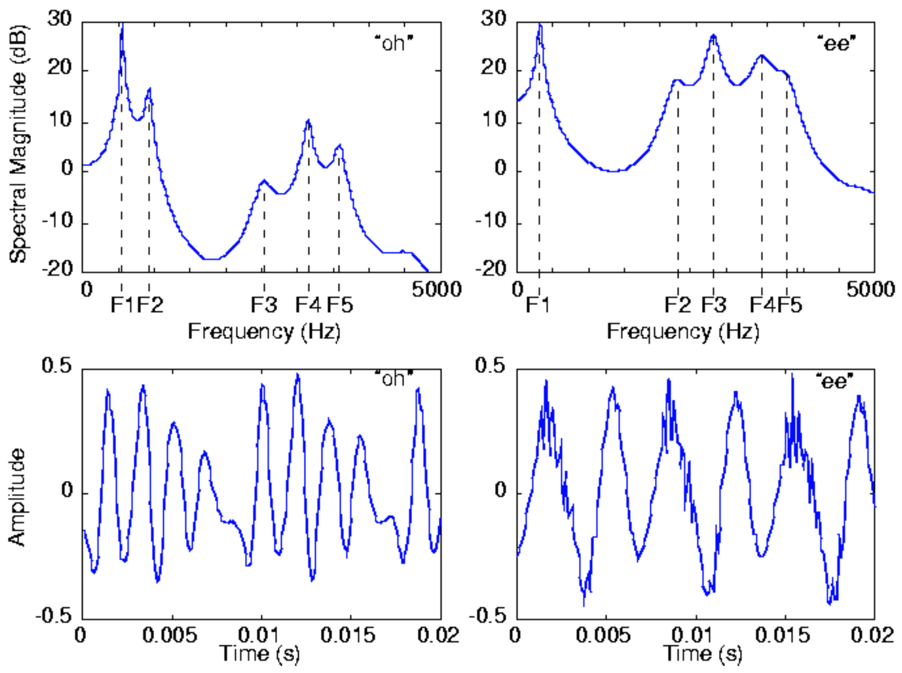


Figure F.4: The Ideal Frequency Response of the Vocal Tract for Sounds "oh" and "ee".

positions that are controlled by the brain to produce the vowel sounds. Spreading the lips, bringing the teeth together, and bringing the tongue toward the front portion of the roof of the mouth produces the sound "ee". Rounding the lips, spreading the teeth, and positioning the tongue toward the back of the oral cavity produces the sound "oh". These variations result in a linear, time-invariant system that has a frequency response typified by several peaks, as shown in figure (F.4).

The figure (F.4) represents the sounds "oh" and "ee" shown on the top left and top right, respectively. The spectral peaks are known as formants, and are numbered consecutively from low to high frequency. The bottom plots show speech waveforms corresponding to these sounds.

These peaks are known as formants. Thus, speech signal processors would say that the sound "oh" has a higher first formant frequency than the sound "ee", with F2 being much higher during "ee". F2 and F3 (the second and first formants) have more energy in "ee" than in "oh." Rather than serving as a filter, rejecting high or low frequencies, the vocal tract serves to shape the spectrum of the vocal cords. In the time domain, we have a periodic signal, the pitch, serving as the input to a linear system. We know that the output-the speech signal we utter and that is heard by others and ourselves-will also be periodic. Example time-domain speech signals are shown in (F.4), where the periodicity is quite apparent.

F.3 LPC Order

Linear Predictive Coding (LPC) is often used by linguists as a formant extraction tool. There are a few important details about LPC that may help avoid common analysis errors. LPC analysis assumes that a signal is the output of a causal linear system. It also assumes that the vocal-tract system is an all-pole filter and that the input to the system is an impulse train. Because of these assumptions, LPC analysis is appropriate for modeling vowels which are periodic and for which the vocal-tract resonator does not usually include zeroes (e.g., in nasalized vowels). The order of an LPC model is the number of poles in the filter. Usually, two poles are included for each formant $+2-4$ additional poles to represent the source characteristics. For adult speakers, average formant spacing is in the $1000Hz$ range for males and in the $1150Hz$ range for females. The LPC order is related to the sample rate of the audio file: $10000Hz$ - LPC order $= 12-14$ (males) and $8-10$ (females); $22050Hz$ - LPC order $= 24-26$ (males) and $22-24$ (females). LPC usually requires a very good speech sample to work with [30]. Many recordings done with omnidirectional microphones contain too little speech detail and too much noise to ascertain reliable LPC readings.

Appendix G

Working Process

G.1 Project Management

1. We decided to use A3 Paradigm to guide us implement our project
   - Application
   - Algorithms
   - Architecture
2. The six point approach was used throughout the entire project i.e. questions (6W model) such as who, what, how, when, whom, why were asked during the entire discussions and project implementation

G.2 Expectations for the Project

G.2.1 Define the Problem

We expect to clearly define the problem by applying the A3 paradigm and the 6W's model.

G.2.2 Good Report

We hope to present a project that is acceptable to the requirements of the study board, a dependable report which can be referred to anytime, a report that is precise and concise.

G.2.3 Meet the Deadline

The MATLAB program, implementation on DSP Tool Kit and Project Report shall be ready just before the deadline so the necessary checks/reexamination can be carried out with before presentation

G.2.4 Share individual Information

Group members shared new information they found in the course of the project.

G.3 Others

The group improved communication

Individual Responsibility was taken seriously

G.4 Implementation Plan

The general ideas on how we solved our problems:

To share different tasks among group members.

To divide whole group into smaller divisions to be able to deal with several tasks at a time.

The table G.1 represents the general schedule and milestones for the proposed project.

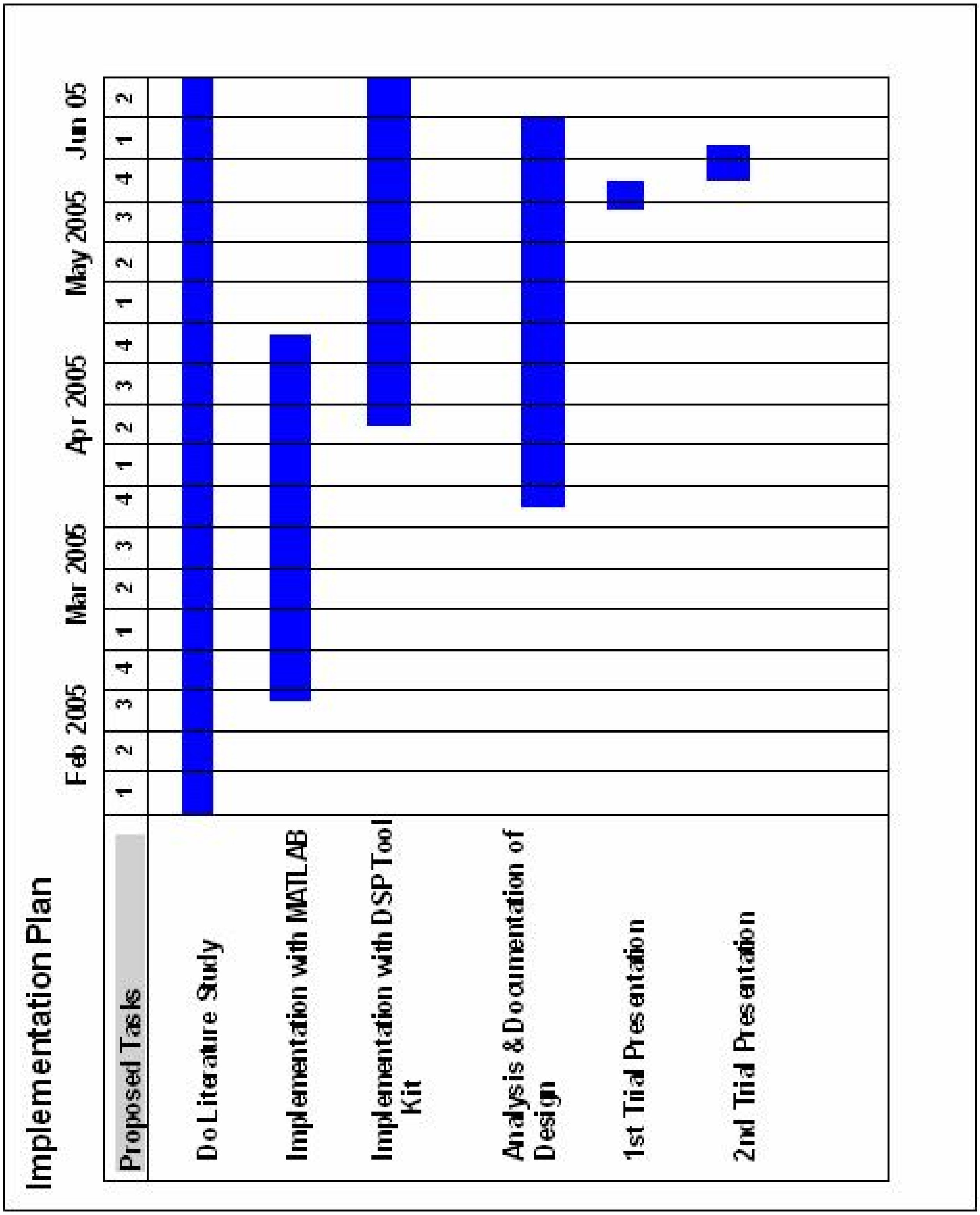


Figure G.1: Implementation Plan

Bibliography

[1] Elias Nemer, Rafik Goubran and Samy Mohmoud, "Robust Voice Activity Detection Using Higher-Order Statistics in the LPC Residual Domain", IEEE Transactions on Speech and Audio Processing, vol.9, no.3, March 2001 pp.217-231.

[2] Beritelli, F.; Casale, S.; Cavallaero, A, "A robust voice activity detector for wireless communications using soft computing", Selected Areas in Communications, IEEE Journal on ,Vol. 16 , Issue 9 , Dec. 1998, pp. 1818 - 1829.

[3] R. Fulchiero and A. Spanias, "Speech enhancement using the bispectrum", in Proc. Int. Conf. Acoustics, Speech, Signal Processing, Vol. 4, 1993, pp. 488-491.

[4] G. Gabor and Z. Gyorfi, "On the higher order distributions of speech signals", IEEE Trans. Acoust., Speech, Signal Processing, Vol. 36, pp. 602-603, April 1988.

[5] Beritelli.F; Casale.S; Ruggeri.G; Serrano.S, "Performance evaluation and comparison of G.729/AMR/fuzzy voice activity detectors", Signal Processing Letters, IEEE , Vol. 9 , Issue 3 , March 2002, pp. 85 - 88.

[6] Rainer Martin, "An Efficient algorithm to estimate the instantaneous SNR of speech signals", Proc. EURO-SPEECH '93, pp. 1093-1096, Berlin, Sept. 21-22, 1993.

[7] Martin, R.: "Spectral Subtraction Based on minimum Statistics", EUSIPCO-94, Edinburgh, Scotland, 13.-16 September 1994, pp. 1182-1185.

[8] "TDMA minimum performance standards for discontinuous transmission operation of mobile stations", TIA doc. and database IS-727, June 1998.

[9] C.Nikias and J.Mendel, "Signal Processing with Higher-Order spectra", IEEE signal processing magazine, July 1993.

[10] Jerry M.Mendel, "Tutorial on Higher-Order Statistics (Spectra) in Signal Processing and System Theory: Theoretical results and Some Applications", IEEE Transactions on Signal Processing, vol.79, no.3, March 1991.

[11] Stephen W. Laverty, Donald R. Brown, "Improved voice activity detection in the presence of passing vehicle noise", Worcester Polytechnic Institute; March 17, 2005.

[12] Joachim Stegmann, Gerhard Schroder, "Robust voice-activity detection based on the wavelet transform", Deutsche Telekom Berkom, Deutsche Telekom AG; 7-10 Sept, 1997.

[13] Virginie Gilg, Christophe Beaugeant, Martin Schonle, Bernt Andrassy, "Methodology for the design of a robust voice activity detector for speech enhancement", International workshop on acoustic echo and noise control (IWAENC2003)

[14] Jia-Lin Shen, Jein-Weih Hung, Lin-Shan Lee, "Robust Entropy- based endpoint detection for speech recognition in noisy environments", Institute of Information Science, Academia Sinica, Taipei, Taiwan, Republic of China, April 14, 2005

[15] R.W. Wall,"Simple Methods for Detecting Zero Crossing", Proceedings of The 29th Annual Conference of the IEEE Industrial Electronics Society Paper 000291, April 14, 2005.

[16] W. Bastiaan Kleijn, Tom Bäckström, and Paavo Alku, "On Line Spectral Frequencies", IEEE Signal Processing Letters, Vol. 10, no. 3, March 2003.

[17] http://www.jdrosen.net/e6880/ April 14, 2005.

[18] S. Gokhun Tanyer and Hamza Ozer, "Voice activity detection in nonstatinary noise", IEEE transactions on speech and audio processing, vol. 8, no.4, July 2000.

[19] Chen Dong, Kuang Jingming, "A robust voice activity detector applied for AMR", Department of Electronic Engineering, Beijing Institute of technology; 21-25 August 2000.

[20] D.K. Freeman, G. Cosier, C.B. Southcott and I. Boyd, "The voice activity detection for the pan-european digital cellular mobile telephone service" in Proc. Int. Conf. acoustics, speech, signal processing, May 1989, pp. 369-372

[21] Philippe Renevey and Andrzej Drygajlo, "Entropy based voice activity detection in very noisy conditions", Swiss center for electronics and microtechnology, Swiss federal institute of technology, March 17, 2005.

[22] Chrysostomos L.Nikias and Anthina P. Petropulu, "Higher-Order Spectra Analysis A nonlinear signal processing framework", PTR Prentice Hall,Englewood Cliffs, NJ: Prentice-Hall, 1993.

[23] J.Chen, K.K. Paliwal and S.Nakamura, "Subtraction of Additive noise from corrupted speech for robust speech recognition", school of Microelectronic Engineering, Griffith University, Brisbane, QLD 4111, Australia, April 15, 2005.

[24] Qifeng Zhu and Abeer Alwan, "The effect of additive noise on speech amplitude spectra: a quantitative analysis", IEEE Signal Processing Letters, Vol. 9, No. 9, September 2002.

[25] http://en.wikipedia.org/wiki/Noise_(physics), April 5, 2005.

[26] http://en.wikipedia.org/wiki/White_noise, April 5, 2005.

[27] http://cnx.rice.edu/content/m0087/latest, April 12, 2005.

[28] http://www.kanecomputing.co.uk/pdfs/dsk_6713a.pdf, April 10, 2005.

[29] http://focus.ti.com/lit/ds/symlink/tms320c6713.pdf, April 10, 2005.

[30] http://www.historicalvoices.org/spokenword/resources/audiotech/lpc.php, April 12, 2005.

[31] Code composer studio guide 2004, Texas Instruments, May, 2004