A NEW METHOD FOR MONAURAL SPEECH SEPARATION USING IDEAL BINARY MASK

A PROJECT REPORT

Submitted by

M. NAVEEN NARAYANAN - 312214106066

S. SOMASUNDAR - 312214106104

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

SSN COLLEGE OF ENGINEERING, KALAVAKKAM

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2018

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled "**A NEW METHOD FOR MONAURAL SPEECH SEPARATION USING IDEAL BINARY MASK**" is the bonafide work of "**M. NAVEEN NARAYANAN (312214106066), S. SOMASUNDAR (312214106104)**" who carried out the project work under my supervision.

Dr. S. RADHA	Dr. R. RAJAVEL
PROFESSOR AND	**SUPERVISOR**
HEAD OF THE DEPARTMENT	ASSOCIATE PROFESSOR
Department of ECE	Department of ECE
SSN College of Engineering	SSN College of Engineering
Kalavakkam – 603110.	Kalavakkam – 603110.

Submitted for the Project Viva – Voce examination held on _____________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our sincere thanks to **Dr. Shiv Nadar**, Founder of SSN Institutions for extending a generous hand in providing the best of resources to the college. **Dr. S. Salivahanan**, the esteemed Head of our Institution has been a source of motivation to all the staffs and students of our college. We are very thankful to him.

Our sincere thanks are due to **Dr. S. Radha,** the Head of the Department for her continuous support and motivation throughout our project.

We thank our project co-coordinator **Dr. R. Jayaparvathy,** panel member **Dr. R. Rajavel** for the timely reviews that helped us to improve our project.

We are indebted to our guide **Dr. R. Rajavel,** for his invaluable guidance and for giving us innovative suggestions for the project. We express our gratitude for his patience and unflinching support.

Last, but not the least, I take this opportunity to thank all the staff members of the Department of Electronics and Communication Engineering for their constant support at all times.

Regards are also to our family, classmates and friends who offered unflinching moral support for completion of this project.

ABSTRACT

This project work proposes a new computationally efficient method for monaural speech separation using Ideal Binary Mask (IBM). Speech separation systems using IBM in general consists of time-frequency (T-F) analysis by a bank of Gamma tone analysis filters, ideal binary mask computation using clean speech and noise and finally reconstruction of speech using the computed IBM via synthesis filter bank. This method involves post multiplication of IBM with the output of the synthesis filter bank. Which involves many computations without contributing anything to the final output with increased computational delay.

This project work solves this issue by changing the order of operation in the reconstruction of speech signal from the noisy speech and improves the performance with minimal computational delay. The proposed method multiplies the computed IBM with T-F signals from the output of the analysis filter bank. This in turn makes many noise dominant frames to be zeros and enables the synthesis filter bank to produce the enhanced speech signal with minimal computational delay. The experimental results show that the proposed approach improves the intelligibility and quality of speech in terms of Short Time Objective Intelligibility (STOI) and Signal to Noise Ratio (SNR) respectively. The proposed method also reduces the computation time considerably as compared to the existing approach of monaural speech separation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CASA	Computational Auditory Scene Analysis
IBM	Ideal Binary Mask
T – F	Time Frequency
GUI	Graphical User Interface
ERB	Equivalent Rectangular Bandwidth
ASR	Automate Speech Recognition
BSS	Blind Source Separation
PCA	Principal Component Analysis
SNR	Signal to Noise Ratio
STOI	Short Time Objective Intelligibility

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The human auditory system is an acoustic and cognitive wonder, which has the ability to easily separate a composed speech from the acoustic mixture. This has lot of requirements in the field of multimedia. Hence, until recently, researchers are trying to build computer models of high level functions of the auditory system. Monaural speech separation is a challenging signal processing problem that finds a lot of applications in speech processing such as speech/speaker recognition, voice communication, air-ground communication and hearing aids. Speech separation is the process of separating the target speech signal from acoustic mixture. The acoustic mixture may be another speech or environmental noise or both. Several methods have been proposed for monaural speech separation, like spectral subtraction [20], subspace analysis [23], speech enhancement approaches [17], hidden Markov modeling [8] and sinusoidal modeling. All these methods require some form of prior knowledge about speech or interference and failed to show improvement in speech quality and intelligibility. Hence, we use Computational Auditory Scene Analysis (CASA) [5] [6] [13] [14] [17] [18] [21].

CASA has been introduced recently to separate the target speech from the acoustic mixture based on the principles of human auditory system. Most of the current CASA based speech separation systems uses the analysis and synthesis filter bank pair proposed by Weintraub.Since then this successful model has been used in most of the speech separation systems. CASA based speech separation systems decomposes the input noisy speech into T-F units using analysis filter bank, then the Ideal Binary Mask (IBM) is computed based on the energy of speech signal and noise signal in each T-F unit. The IBM is a binary mask, in which 1 indicates the speech dominant T-F unit and 0 indicates the noise dominant T-F unit. The computed IBM will be applied after the resynthesis of the speech signal using synthesis filter bank. This approach involves many unnecessary computations since most of the frames will have zero values corresponding to the noise dominant T-F units. This will not contribute anything to improve the quality and or intelligibility of the speech signal and also increases the computational complexity. This project work addresses this issue and proposes a new method to reduce the computational complexity by changing the order of operation in the typical monaural speech separation system. In this proposed method, the computed IBM will be multiplied before synthesizing the speech signal. Since many frames have zero values, the synthesis filter bank processes these frames with minimal computational delay and hence the throughput of the overall system can be increased without compromising on the quality and intelligibility of the speech signal.

An experiment has been conducted to evaluate the performance of the proposed approach using IEEE speech corpus [6] and Noisex-92 [15]. The experimental results show that the proposed approach improves the intelligibility and quality of speech in terms of Short Time Objective Intelligibility (STOI) and Signal to Noise Ratio (SNR) respectively. The proposed approach improves the SNR by an average value of 0.29914 dB for babble noise and 0.30748 dB for factory noise. Similarly, the proposed system improves the STOI by an average value of 0.01036 for factory noise and 0.01086 for babble noise. The proposed approach also reduces the computation time considerably as compared to the existing approach of monaural speech separation.

1.2 SPEECH SEPARATION SYSTEMS AND ITS APPLICATIONS

In real environment, if a person is speaking, the speech will get affected by the surrounding noise for example, other people speaking/shouting, passing a car, and many natural sounds. Several applications require a system that separates the intended speaker’s speech from the noisy speech signal. For example, voice communication over cellular phone will get affected by the surrounding noise present at the transmitting end. In an air-ground communication, the pilot speech will be affected by the high level of cockpit noise. In a teleconferencing system, the noise in one location will be broadcasted to all other locations. In the literature, two main approaches are proposed for speech separation. They are, speech enhancement, and blind source separation (BSS). The speech enhancement approach is well suited when the noise is stationary and is not when the noise is non-stationary. However, BSS is the successful speech separation approach for both stationary and non-stationary noisy conditions. BSS which is the most general form of source separation problem aims to extract the unknown speech signals from the acoustic mixture signals without a priori information on signals sources or on mixture signals. These acoustic mixtures may be another speech or environmental noise or both. Monaural speech separation is a challenging problem in speech and signal processing. Monaural speech separation is the process of separating two sound sources from a noisy mixture which is obtained from the recording of a single microphone. This Speech segregation process, also known as the cocktail party problem, refers to the problem of segregating target speech from its background interference. Monaural speech segregation, attempt speech/sound segregation by monaural recordings with one microphone. Recoding by one microphone may be sufficient to design mechanism like human auditory system. This is important for many real-world applications including robust speech and speaker recognition, audio information retrieval and hearing aids design and in artificial intelligence. However, despite decades of effort, monaural speech segregation still remains one of the challenging problems in signal processing. Over the last few decades, researchers have proposed several methods for monaural speech separation such as spectral subtraction [20], subspace analysis [23], hidden Markov modeling [17], sinusoidal modeling [8] and Computational Auditory Scene Analysis (CASA) [12] [18]. In Spectral Subtraction, the spectrum of the noise signal and noisy speech signal are estimated, the denoised speech is obtained by subtracting the spectrum of noise signal from the spectrum of noisy speech signal. In sub space analysis, Eigen decomposition of acoustic mixture has been done and then subspace analysis such as Independent component analysis (ICA), Principal component analysis (PCA) are used to remove the interference from the noisy mixture. In model based approaches, trained models of speech and noise are used for separation. These approaches usually suppose certain properties of interference and then separate composite speech based on these hypotheses. That is why their capacity for speech segregation is much limited than the human capacity. Thus, we are interested by the study of the computational auditory scene analysis (CASA). CASA is the best technique in BSS separation. CASA aims to build sound separation systems that adhere to the known principles of human hearing.

1.3 MOTIVATION

In the past decades, research in the field of speech separation has focused on the suppression of additive background noise. The presence of background noise in speech significantly reduces the quality and intelligibility of speech signal. Degradation of speech severely affects the ability of person, whether impaired or normal hearing, to understand what speaker is saying. The ultimate goal of speech separation is to eliminate the additive noise present in speech signal and restore the speech signal to its original form. Several methods have been developed with some or the other auditory, perceptual or statistical constraints placed on the speech and noise signals. However, in real world situations, it is very difficult to reliably predict the characteristics of the speech waveform. Hence, in effect the speech separation methods are sub – optimal and can reduce the amount of noise in the speech signal to some extent. Due to the sub – optimal nature of these methods, some of the speech signal can be distorted during the process. In effect, there is a tradeoff between distortions in the processed speech and the amount of noise suppressed.

The effectiveness of the speech separation system can therefore be measured based on how well it performs in light of this tradeoff. Exploration of the same is the motivation for this project work. Complexity and throughput of the noise reduction algorithms is also of concern in applications especially those related to portable devices such as mobile communications and digital hearing aids. In view of the above constraints and challenges there is a need to search for methods towards the improved performance of the speech separation algorithms.

1.4 OBJECTIVE

Our Project aims to create a new method for monaural speech separation using ideal binary mask. The main objectives of the project are

- To implement the typical and proposed monaural speech separation system using Matlab.
- To develop a MATLAB – GUI Model for the typical and proposed monaural speech separation system using IBM.

- Compare the computational complexity of these systems and show that the proposed system is less complex.
- Also to show that the proposed system improves speech quality and intelligibility.

1.5 ORGANISATION OF THE REPORT

Following this introduction chapter, the rest of the chapters are organized as follows

Chapter 2 introduces the reader to the speech separation algorithms by giving a theoretical overview; with a stress on speech separation to build a foundation for our project.

Chapter 3 explains about the typical and proposed monaural speech separation systems with relevant diagrams and mathematical equations.

Chapter 4 gives the details and guide user interface modelling using MATLAB – GUI which we have used for the implementation of our project.

Chapter 5 provides the experimental simulation results and observations from the performance of the system is studied.

Finally, **Chapter 6** presents the conclusion of the project and its future scope and development which would be useful in many applications.

1.6 CONCLUSION

Thus an introduction about the typical monaural speech separation system and the proposed monaural speech separation system is discussed. The objective and motivation of the project is listed and a brief overview about CASA and IBM and their role in speech separation is explained in this section.

CHAPTER 2

SPEECH SEPARATION TECHNIQUES

2.1 INTRODUCTION

In speech communication, the speech signal is always accompanied by some noise. In most cases the background noise of the environment where the source of speech lies, is main component of noise that adds to the speech signal. Though the obvious effect of this noise addition is t make the listening task difficult for a direct listener, there are many more far reaching negative effects when we process the degraded speech for some other applications. A related problem is processing noisy speech in preparation for coding by a bandwidth compression system. Hence speech separation not only involves processing speech signals for human listening but also for further processing prior to listening. The main objective of speech separation is to improve the perceptual aspects of speech such as quality, intelligibility, or degree of listener fatigue by removing the noise from the noisy speech signal.

2.2 SPEECH SEPARATION ALGORITHMS

All the speech separation methods aimed at separating a target speech from a noisy speech. Most of the speech separation algorithms are based in one way or the other on the estimation of the background noise. If the noise is more stationary than the speech, it is easy to estimate the noise during the pauses in the speech.

Finding the pauses in the speech is based on checking how close the estimate of background noise is to the signal in the current window. Voiced sections can be located by estimating the fundamental frequency. Both methods easily fail on unstressed voice or short phonemes, taking them as background noise. On the other hand, this is not very dangerous because the effect of these faint phonemes on the background estimate is not that critical.

A number of algorithms have been proposed for speech separation with the primary goal of improving speech quality. These algorithms can be discussed as follows

2.2.1 SPECTRAL SUBTRACTIVE ALGORITHMS

Spectral subtraction is the earliest and simplest method for denoising speech degraded by additive noise. This technique estimates the spectrum of the clean speech signal by the subtraction of the estimated noise magnitude spectrum while keeping the phase spectrum of the noisy signal. Spectral subtraction algorithms were initially proposed by Weiss Et Al in the correlation domain and later by Foll in the Fourier transform domain. The amplitude spectra are employed to give the amplitude spectrum of the enhanced speech as given below

$$S(\omega) = (|X(\omega)| - |N(\omega)|)e^{-jargX(\omega)} \quad (2.1)$$

Another old method is to employ the power spectral densities, giving the spectrum of the enhance speech as follows

$$s(\omega) = \sqrt{|x(\omega)|^2 - |N(w)|^2}e^{-jargX(\omega)} \quad (2.2)$$

Where S(ω) is the spectrum of the enhanced speech

X(ω) is the spectrum of noisy speech

N(ω) is the spectrum of noise

The drawback of this technique is the residual noise. The spectral subtraction method can lead to negative values, resulting from differences among the noise estimator and the actual noise frame. To cope with this problem, negative values must be set to zero, producing Spectral spikes, well known as musical noise. This effect causes an annoying perception of enhanced speech and, therefore, it must be corrected.

Spectral subtraction recovers the temporal structure of speech power spectrum. This can be achieved in two parts namely two dimensional smoothing and controlled noise subtraction to solve the speech probability distribution function discontinuity problem caused by spectral subtraction series algorithms.

Modified spectral subtraction can be used to solve the speech separation problem. In the first stage, noise is extracted from the expanded noisy speech signals using an MSCPE – based algorithm. In the second stage, a modified spectral subtraction and a modified Weiner filter approach are proposed to extract the speech signal.

The conventional spectral subtraction – based space (SS) can technique can also be extended to deal with reverberation. In this proposed approach, the dereverberation parameters of SS are optimized to improve the likelihood of the acoustic model and not just the waveform of the signal. The system is capable of attractively fine – tuning these parameters jointly with acoustic model training for effective use in ASR applications.

2.2.2 STATISTICAL MODEL BASED ALGORITHMS

The speech enhancement problem is posed in a statistical estimation framework. Given a set of measurements, corresponding say to the Fourier transform coefficients of the noisy signal, we wish to find a linear (or nonlinear) estimator of the parameter of interest, namely, the transform coefficients of the clean signal. The Weiner algorithm and minimum mean square error algorithms, among others, fall in this category. Work in this area was initiated by McAulay and Malpass who proposed maximum-likelihood approach for estimating the Fourier transform coefficients (spectrum) of the clean signal, followed by the work of by Ephraim and Malah who proposed an MMSE estimator of the magnitude spectrum. Much of the work with Wiener algorithm was initiated in the speech enhancement field by Lim and Oppenheim.

2.2.3 SUBSPACE ALGORITHMS

Subspace algorithms are rooted primarily on linear theory. More specifically, these algorithms are based on the principle that the clean signal might be confined to a subspace of the noisy Euclidean space. Consequently, given a method of decomposing the vector space of the noisy signal into a subspace that is occupied primarily by the clean signal and a subspace that is occupied primarily by the noise signal, one could estimate the clean signal simply by nulling the component of the noisy vector residing in the "noise subspace". The decomposition of the vector space of the noisy signal into "signal" and "noise" subspaces can be done using well-known orthogonal matrix factorization techniques from linear algebra and in particular, the Singular value Decomposition (SVD) or the Eigen vector – Eigen value factorization.

This algorithm has been shown to improve speech quality and reduce listener fatigue, but failed in as far improving speech intelligibility. Recently, a new class of algorithm based on the time – frequency mask has been explored and evaluated for noise reduction applications. In speech enhancement, subspace method is used to reduce noise dominant Eigen values using microphone - array signal processing. In this method less – directional ambient noise is eliminated in the subspace domain by reducing the noise – dominant eigen values. Then the remaining mixture of the multiple directional components is decomposed into single component corresponding to each sound source by the modified MV beam former.

A signal subspace speech enhancement method is also proposed for narrowband noise reduction. Based on the eigen value decomposition of the rank deficient noise covariance matrix, it is shown how to formulate the enhancement algorithm by decomposing the vector space of noisy signal into a signal-plus-noise subspace and a noise-free subspace. The enhancement is performed by estimating the clean speech from the signal plus-noise subspace and adding the components in the noise – free subspace.

Subspace method can also be modified to develop under determined blind speech separation method based on single source detection. Instead of fixing the active source number as a constant or calculating the real source number of every TF point, the proposal algorithm introduces a convex model, taking into account both projection and size of subspace, which avoids increasing the computation cost.

2.2.4 COMPUTATIONAL AUDITORY SCENE ANALYSIS (CASA)

The CASA is the study of auditory scene analysis (ASA) by computational means. In essence, CASA systems are "machine listening" systems that mimic the human auditory system. CASA is the speech separation system. The speech separation process in auditory scene analysis has two main steps: segmentation and grouping. The first step is to decompose the auditory scene in time-frequency zones or segments which are sound elements having coherent structure. The second step is to group segments that may result from the same source in auditory streams. The segmentation and grouping mechanisms exploit acoustic features such as harmonicity, coherent envelope, coherent modulation frequency or amplitude, which are based on the intrinsic characteristics of the sound properties. Typically, CASA extracts one source from a single channel of audio using heuristic grouping rules based upon psychological observations.

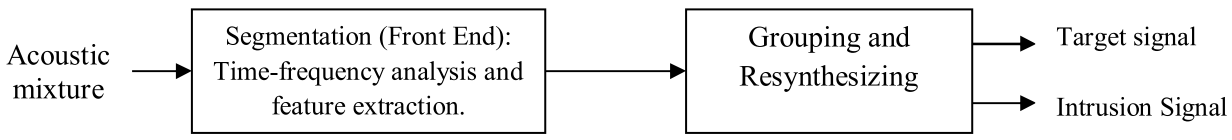

Acoustic mixture
Segmentation (Front End): Time-frequency analysis and feature extraction.
Grouping and Resynthesizing
Target signal
Intrusion Signal

Fig. 2.1 The schematic diagram of the CASA system

2.2.4.1 SEGMENTATION STAGE

The first step of CASA system usually consists of a time-frequency analysis of the signal that mimics the frequency selectivity of the human ear and the characteristics extraction which are useful for the following steps. This is the segmentation of the auditory scene in elementary acoustic features. Typically, the input signal is passed through a bank of bandpass filters; each one simulates the frequency response associated with a particular position on the basilar membrane. The gammatone filter is often used, which is an approximation of the impulse response of the physiologically recorded auditory nerve fibers. Most CASA systems make the device time-frequency representation and the application of a correlogram to extract features and useful information for the following steps as: the autocorrelation of a filter response, the autocorrelation of a filter response envelope, the cross-channel correlation, the dominant fundamental frequency of each frame. The filter bank used is generally composed of 128 gammatone filters (or 64 filters) with center frequencies ranging from 80 to 4000 Hz [9] [21] [16]. The impulse response of this filter has the following form:

$$g(t) = \begin{cases} at^{n-1}e^{-2\pi bt}\cos(2\pi ft + \phi), & t > 0 \\ 0 \qquad , & else \end{cases}$$

where a is the amplitude, $\phi$is the phase, n is the filter order, b is the filter band width (ERB = 24.7 + 0.108 × f), the filter center frequency, t is the time. For each channel, the output is divided into T-F frames with a 50% overlap between two consecutive frames. As a result of this filtering and windowing, the input signal is decomposed into a representation of two dimensional time-frequency (TF) or a collection of TF unit.

2.2.4.2 GROUPING STAGE

After the first stage, we obtain a time–frequency representation in order to extract features that are useful for grouping. The grouping stage presents the problem of determining which components should be grouped together and identified as the same sound. Principal features that are used for grouping are fundamental frequency (F0), harmonicity, onset synchrony, continuity, etc. Then, the signal components are split into groups based on the similarity of their source and location attributes. These groups are the separated signals. In this context, it can be classified into sequential grouping cues (across time) and simultaneous grouping cues (across frequency):

- Sequential grouping is influenced by many of the factors that define the similarity, the frequency proximity, the repetitive character, and the repetition rate of successive sounds.
- Simultaneous grouping is affected by harmonicity, envelope coherence, binaural correlation, amplitude modulation, and frequency modulation.

2.2.4.3 IDEAL BINARY MASK (IBM)

The IBM is defined as the computational goal of CASA [12] [7] [14] [18]. A binary mask is a matrix of binary numbers, defined in the T-F domain. The numbers are assigned by comparing the energy of the target speaker against the concurrent speaker. If the target energy, in a T-F cell, is greater than the one of the concurrent then the value assigned is 1. Otherwise, the value is assigned to 0. This technique has been of great effectiveness in Computational Auditory Scene Analysis (CASA) when used as output representation to label the origins of the mixed speech. When both target and concurrent signals are known, the ideal BM (IBM) could be determined.

The notion of an ideal binary mask (IBM) has been proposed as a primary goal of CASA. In the time frequency representation of the front-end part, the key factor behind the notion of ideal binary mask is to keep the time-frequency regions of the target that are stronger than those of the interference, and delete regions which are weaker than the latter. More precisely, the ideal mask is a binary matrix, where ‘‘1’’ indicates that the energy of the target is higher than the energy of the interference inside the corresponding TF unit and ‘‘0’’ indicates the opposite [4] [5] [12]:

$$M(t,f) = \begin{cases} 1, & if\ s(t,f) - n(t,f) > LC, \\ 0, & else \end{cases}$$

Where s (t, f) is the target energy in a TF unit, n (t, f) is the interference energy. The threshold LC (standing for local signal to noise ratio [SNR] criterion) in dB is typically chosen to be 0, giving a 0dB SNR Criterion, although other SNR Criteria can also be chosen. One thing special about the 0dB criterion is that, under certain conditions, the IBM thus constructed gives the highest SNR gain of all binary masks treating clean target as the signal.

Weintraub was the first who used this approach in a CASA system, which had been adopted by several other researchers. The use of binary masks is motivated by the masking phenomenon of the human ear, in which a weaker signal is masked by a stronger within the same critical band. It is also noted that the reconstruction of a masked signal can be interpreted as a highly non-stationary Wiener filter. The IBM has several properties such as:

- *Flexibility:* Depending on the target and with the same mixture, we can define different masks.
- *Good definition:* The mask is well defined even if there are several intrusions in the speech mixture and we can also estimate several targets from this same mixture.
- The ideal binary mask is more performant than all existing masks. In fact, it gives excellent resynthesis for a variety of sounds.

The IBM has several desirable properties as a computational goal of CASA, including direct correspondence to the auditory masking phenomenon, flexibility in constructing different IBMs out of the same mixture depending on what the target is, and well-defined regardless of the number and types of signals in the mixture. The IBM has also been shown to be important for human speech intelligibility and automatic speech recognition. A number of recent psychoacoustic experiments have demonstrated that target speech reconstructed from the IBM can dramatically improve the intelligibility of speech masked by different types of noise, even in very noisy conditions.

2.3 LITERATURE SURVEY

M. Weintraub, "A theory and computational model of auditory monaural sound separation," Ph.D. dissertation, Dept. Elect. Eng., Stanford Univ., Stanford, CA, 1985

- This research provides a complete analysis of the human auditory system.

- A conceptual approach was introduced to mimic the knowledge and information used by human auditory system to separate sounds.
- A computer model was developed to separate the speech of two simultaneous talkers

V. Hohmann, "Frequency analysis and synthesis using a Gammatone filterbank," ***Acta Acustica united with Acustica,*** **Vol. 88, pp. 433 – 442, Jan 2002.**

- This paper describes an efficient implementation of the 4th-order linear Gamma tone filter based on an impulse-invariant, all-pole design.
- A linear auditory filter bank was constructed from these filters.
- This filter has been used in several applications involving computational auditory peripheral filtering.

Jihen Zeremdini, Mohamed Anouar Ben Messaoud and Aicha Bouzid, "A comparison of several computational auditory scene analysis (CASA) techniques for monaural speech segregation", ***Brain Informatics,*** **Vol. 2, Issue 3, pp. 155 – 166, Sept 2015.**

- This paper mainly focused on the different CASA stages and the IBM for monaural speech segregation.
- It describes the several methods that used CASA to separate composite speech such as Hu and Wang, Zhang and Liu, Zhao and Shao, Li and Guan approaches etc.
- Finally, an evaluation and a comparison was presented for the different monaural speech segregation methods.

Belhedi Wiem, Ben Messaoud Mohamed Anouar, Bouzid Aichl, "Time-Frequency Masks for Monaural Speech Separation: A Comparative Review", ***7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT),*** **2016.**

- This paper focuses the masking effect on Computational Auditory Scene Analysis (CASA) based systems for single channel speech separation (SCSS).
- The CASA system employed in this study is Hu and Wang model.
- The new proposed system uses a soft mask instead of the hard binary mask.
- They have proved that the soft mask provides a better separation quality.

Abrar Hussain, Kalaivani Chellappan, Siti Zamratol M, "Single Channel Speech Enhancement Using Ideal Binary Mask Technique Based On Computational Auditory Scene Analysis", ***Journal of Theoretical and Applied Information Technology,*** **Vol.91. No.1, 15th Sept 2016.**

- In this paper, ideal binary mask which is inspired by the computational auditory analysis is used to analyze and synthesize the input speech signals and masker signals in the time-frequency domain.
- Synthesized signals are evaluated for speech quality measurement in terms of segmental signal-to-noise ratio.
- This study uses Malay language based speech as input speech signals.

D.L. Wang, "Time–Frequency Masking for Speech Separation and Its Potential for Hearing Aid Design," ***Trends in Amplification,*** **Vol. 12, pp.332-353, Dec 2008.**

- This article introduces the T-F masking concept and reviews T-F masking algorithms that separate target speech from either monaural or binaural mixtures, as well as microphone-array recordings.
- This article also surveys recent studies that evaluate the perceptual effects of T-F masking techniques, particularly their effectiveness in improving human speech recognition.

Ke Hu and DeLiang Wang, "An Unsupervised Approach to Cochannel Speech Separation", ***IEEE Transactions on Audio, Speech and Language Processing,*** **Vol.21, No.1, Jan 2013.**

- In this paper, an unsupervised method was proposed for cochannel speech separation.
- The proposed system uses the CASA algorithm for voiced speech segregation.
- The proposed system was compared with several other methods across a range of input SNR conditions.

2.4 CONCLUSION

This chapter discussed in detail about different monaural speech separation techniques and Computational Auditory Scene Analysis (CASA) that are used in improving the quality of speech. The significance of CASA in speech separation and the IBM as A computational goal of CASA is also discussed.

CHAPTER 3

TYPICAL AND PROPOSED MONAURAL SPEECH SEPARATION SYSTEM

3.1 INTRODUCTION

This chapter describes the model of a typical and proposed monaural speech separation system. The typical system uses the approach proposed by Weintraub for speech resynthesis. The typical system consists of analysis and synthesis filter bank pair modelled using gammatone filters. The input to the system is a noisy speech signal and the output of the system is the denoised speech signal. The proposed system is same as the typical monaural speech separation system except for the change in order of operation. The proposed system improves the speech quality and intelligibility in terms of Signal to Noise Ratio (SNR) and Short Time Objective Intelligibility (STOI) respectively. The proposed method also reduces the computation time considerably as compared to the existing approach of monaural speech separation.

3.2 TYPICAL MONAURAL SPEECH SEPARATION SYSTEM

Till now, most of the CASA based monaural speech separation systems use the same framework proposed by Weintraub [12] which consists of analysis and synthesis filter bank pair [20] [5]. The analysis and synthesis filter bank is modeled by using 128 gammatone filters with center frequency ranging from 80 Hz to 4000Hz. The typical CASA based monaural speech separation system is shown below:

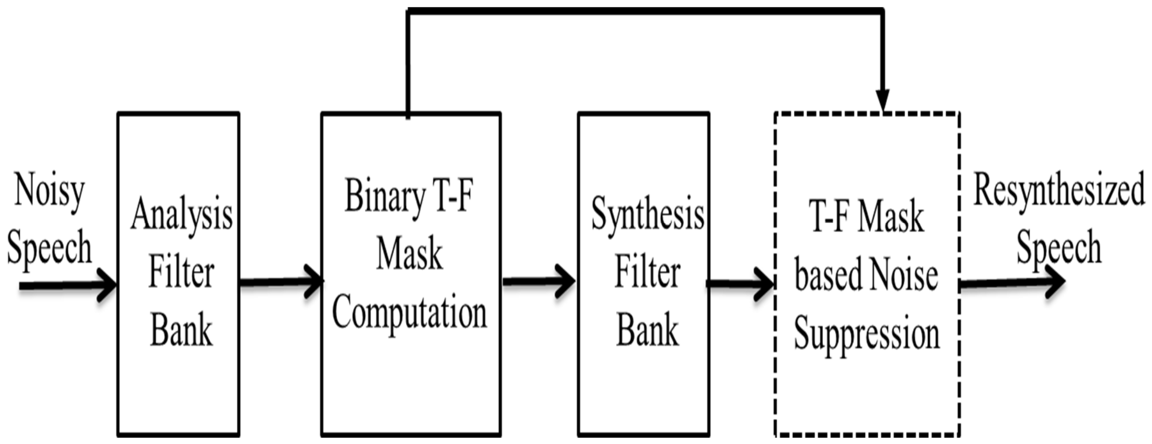


Fig 3.1 Block diagram of Typical Monaural Speech Separation system

In the above system the input noisy speech signal, speech signal and the noise signal is fed to the 128 channels of analysis filter bank. Each channel of the analysis filter bank is modelled using gammatone filter of varying center frequencies. The output of the analysis filter bank from each channel is divided in to T-F units [19] [7] and energy of the speech and noise signal in each T-F unit is compared to determine the IBM. Then the signal is reconstructed using the synthesis filter bank. The synthesis filter bank is also modelled by using the gammatone filter. The output of the synthesis filter is once again framed in to T-F units and multiplied with the computed IBM from the previous stage. Finally, the output from each channel is added to get the denoised speech. The mathematical analysis of the typical monaural speech separation is done in the next section.

3.2.1 SPEECH ANALYSIS

The input noisy speech signal x[n] is fed as input to the 128 channels of the analysis filter bank. The analysis filter bank performs a FFT filtering of the input signal with the impulse response of the gammatone filter. The output of the analysis filter bank $g_i[n]$ is given by

$$g_i[n] = x[n] * h_i[n]; \quad 1 \leq i \leq C. \tag{3.1}$$

where * indicates linear convolution between the input speech/noise/noisyspeech signal x[n] and impulse response of the gammatone filter in ith channel $h_i[n]$ and C is the total number of channels.

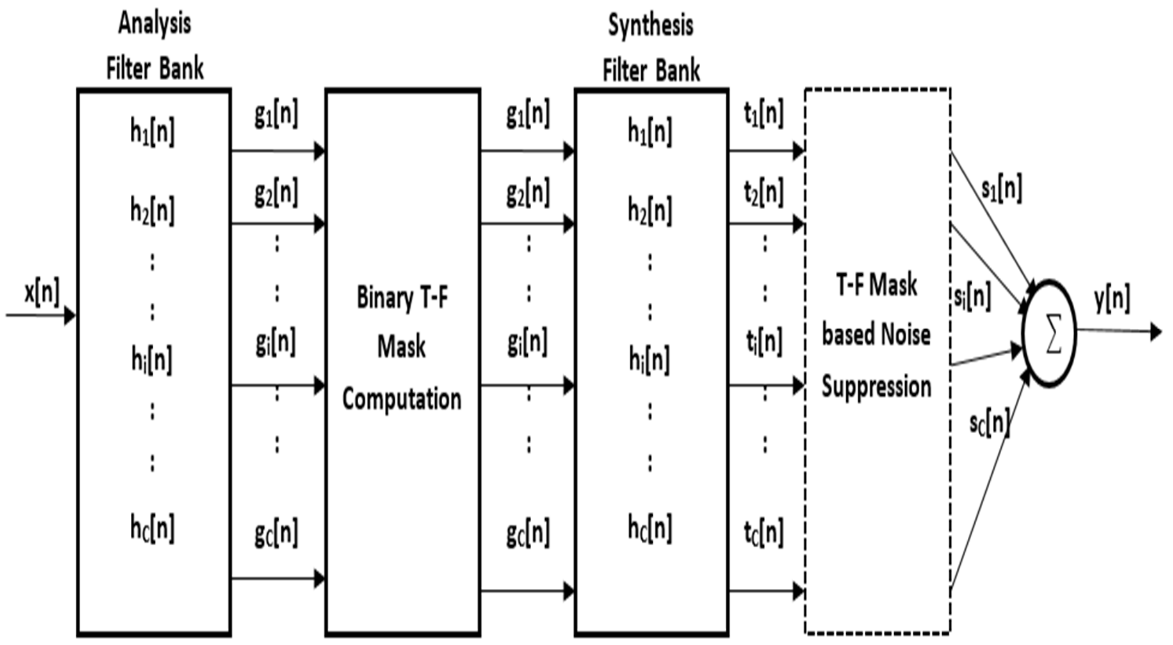


Fig 3.2 Schematic of Typical Monaural Speech Separation system

3.2.2 BINARY T-F MASK COMPUTATION

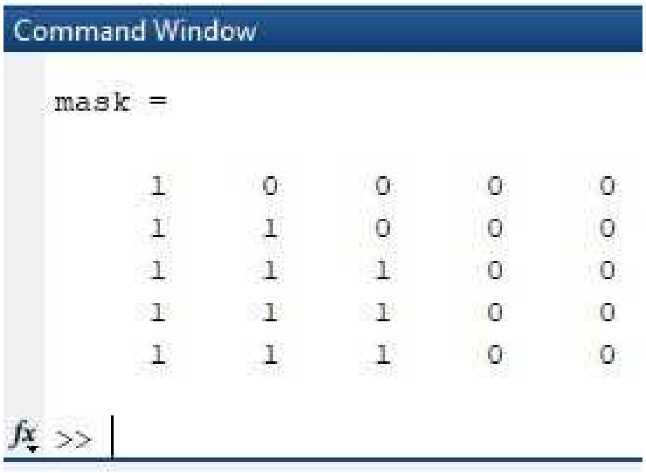
Now the signal $g_i[n]$ in each channel is divided to T-F frames, then the energy of speech signal and the energy noise signal in each frame is computed. The IBM is assigned 1 if it is a speech dominant T-F unit and 0 if it is noise dominant T-F unit.

Speech Energy: $$SE_{ij} = \sum_{m=jR}^{jR+L-1} (gS_i[m])^2 \tag{3.2}$$

Noise Energy: $$NE_{ij} = \sum_{m=jR}^{jR+L-1} (gN_i[m])^2 \tag{3.3}$$

Where $SE_{i,j}$ indicates the energy of speech signal and noise signal in ith channel, jth frame respectively, gS_i and gN_i are the filtered response of speech signal and noise signal in ith channel respectively. L denotes the Frame length and Windowshift R is given by R=L/2. The T-F Binary mask is defined as

$$M(t,f) = \begin{cases} 1, & if\ SE_{ij} - NE_{ij} > 0 \\ 0, & otherwise \end{cases} \tag{3.4}$$



```
mask =

     1     0     0     0     0
     1     1     0     0     0
     1     1     1     0     0
     1     1     1     0     0
     1     1     1     0     0

fx >>
```

Fig 3.3 Example of an Ideal Binary Mask

The above figure shows an example of an Ideal Binary Mask which is obtained by comparing the energies of speech and noise in each T-F unit.

3.2.3 SPEECH SYNTHESIS

The synthesis filter bank performs the inverse operation of the analysis filter bank. The synthesis filter bank is also composed of 128 gammatone filter with the same filter coefficients as that of the analysis filter bank. Usually the coefficients in the synthesis filter bank is the inverse of the analysis filter bank, but in this research work the synthesis filter bank uses the same coefficients as that of the analysis filter bank. Instead, the input of the synthesis filter bank is flipped and the output of the same is also flipped. Thus the original signal can be reconstructed. Hence, after estimating the binary mask, the signal $g_i[n]$ from each channel is flipped and then filtered using the synthesis filter bank. The filtered output is once again flipped and divided in to T-F units. Then the binary mask estimated in the previous stage is now multiplied using a cosine window to obtain the denoised speech. If the binary mask for a T-F frame is assigned 1, then a cosine window is multiplied with corresponding T-F frame and if the binary mask for a T-F frame is assigned 0 then the entire frame is multiplied with zero. The windows are multiplied with 50% overlap. The mathematical expressions for the above steps in typical monaural speech separation system is given below

$k_i[n] = f_i[n] * h_i[n]$

$$= \sum_{m=0}^{\infty} f_i[m]\, h_i[n-m] \tag{3.5}$$

where $f_i[n] = g_i[-n]$.

$$s_{i,j}[m] = \sum_{m=jR}^{jR+L-1} t_i[m]\, p_{i,j}[jR-m] \tag{3.6}$$

where $t_i[n] = k_i[-n]$ and $R=L/2$.

Equation (3.6) shows the process of cosine window multiplication with the T-F unit. The cosine window is multiplied with 50% overlap to the previous frame. Hence, the value of R in equation (3.6) is L/2.

and $p_{i,j} = w[n]$ if $M(i,j) = \begin{cases} 1, & x < 0 \\ 0, & otherwise \end{cases}$

Here w[n] is the sliding cosine window which is defined as,

$$w[n] = \begin{cases} 1 + \cos(2\pi(n-1)/L - \pi)/2\,; & 0 \le n \le L-1 \\ 0; & otherwise \end{cases} \tag{3.7}$$

and finally the output from each channel $s_i[n]$ is added together to get the denoised output speech.

$$y[n] = \sum_{i=1}^{C} s_i[n] \tag{3.8}$$

where C is the total number of channels.

3.3 PROPOSED MONAURAL SPEECH SEPARATION SYSTEM

The proposed system uses the same structure as that of the typical speech separation system. The proposed system differs from the typical system only in the order of operation. The IBM computed after the analysis filter bank filter bank is pre multiplied with the speech signal and then sent to the synthesis filter bank. On multiplying the mask with the speech signal many noise dominant frame will be made zero. The synthesis filter bank takes lesser time to reconstruct the speech signal as lots of zeroes are introduced in the decomposed signal.

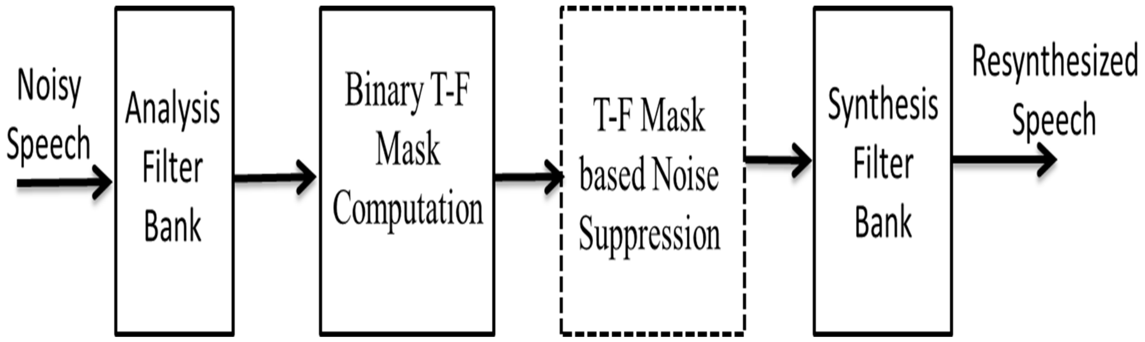


Fig 3.4 Block diagram of Proposed Speech Separation system

The proposed system uses the same set of 128 gammatone filters for analysis and synthesis filter banks. The process of speech separation is same in both the systems till the stage of binary mask computation. In the proposed system the binary mask is multiplied and then resynthesized using the synthesis filter bank. The process of mask multiplication is also the same as in the typical speech separation system. The cosine window is multiplied to the T-F unit if 1 is assigned to the corresponding T-F unit in the IBM. Similarly, the frame is multiplied with zero if zero is assigned to the corresponding T-F unit in IBM.

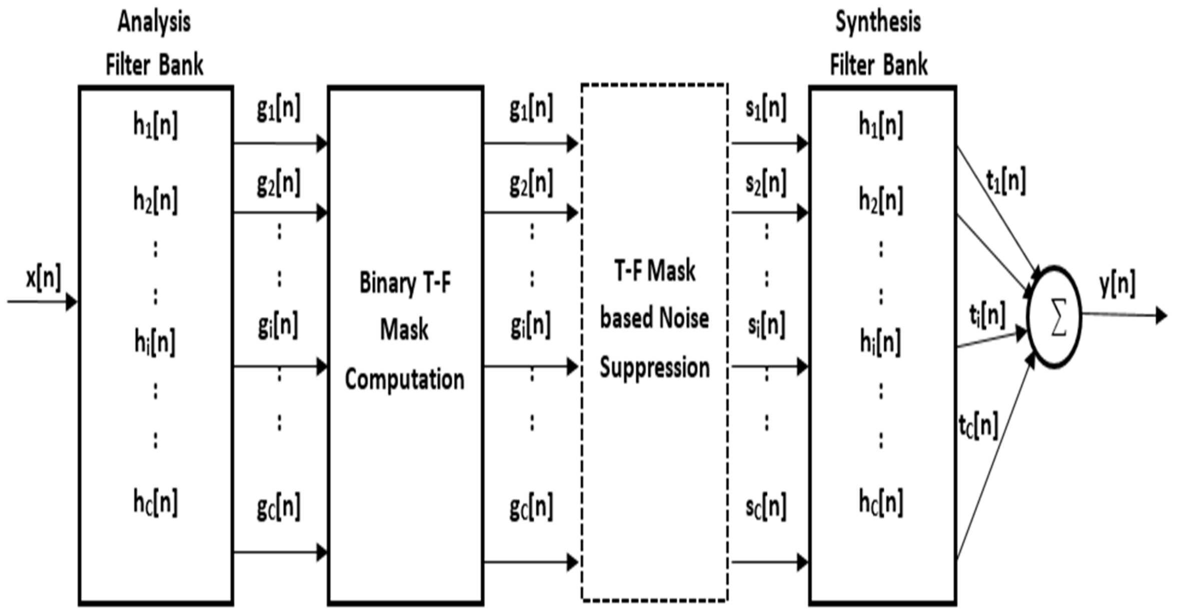


Fig 3.5 Schematic of Proposed Speech Separation system

The mathematical expression for the proposed system differs only after the stage of binary mask computation which is given below

$$s_{i,j}[m] = \sum_{m=jR}^{jR+L-1} g_i[m]\, p_{i,j}[jR-m] \tag{3.9}$$

where $p_{i,j} = \begin{cases} w[n], & if\ M(i,j) = 1 \\ 0, & if\ M(i,j) = 0 \end{cases}$ and R = L/2.

Here w[n] is the sliding cosine window.

In synthesis filter bank, the signal $s_i[n]$ (decomposed speech signal after mask multiplication) is flipped and convolved with the impulse response of the gammatone filter.

$$k_i[n] = f_i[n] * h_i[n]$$

$$= \sum_{m=0}^{\infty} f_i[m]\, h_i[n-m] \tag{3.10}$$

where $f_i[n] = s_i[-n]$.

The output of the synthesis filter bank from each channel $k_i[n]$ is once again flipped and added together to get the denoised output speech y[n].

$t_i[n] = k_i[-n]$ and finally,

$$y[n] = \sum_{i=1}^{C} t_i[n] \tag{3.11}$$

3.4 CONCLUSION

The chapter discussed about the various stages in the typical and proposed monaural speech separation system with mathematical equations and diagrams. This system uses the Weintraub approach for speech resynthesis and the filter banks are modelled using Gammatone filters. The difference between the typical system and proposed system is explained with relevant mathematical equations and diagrams. The proposed improves the speech quality and intelligibility and also reduces the computational delay.

CHAPTER 4

MATLAB GRAPHICAL USER INTERFACE MODEL

4.1 INTRODUCTION

MATLAB is a well-known software package that is widely used for control system design, signal processing, system identification, etc. However, users who are not familiar with MATLAB commands and system identification theory sometimes find it difficult to use, typically because there are many different approaches to system identification. This chapter proposes GUI model for the speech separation system.

4.2 MATLAB SOFTWARE

MATLAB is one of the most famous numerical computation software. It is widely used in control engineering communities and other research communities. MATLAB has three distinctive features.

- **Automatic storage allocation:** Variables in MATLAB need not be declared prior to being assigned. Moreover, MATLAB expands the dimensions of arrays in ord4er for assignments to make sense.
- **Functions with variable arguments lists:** MATLAB contains a large collection of functions. They take zero or more input arguments and return zero or more output arguments. Functions can support a variable number of input and output arguments, so that on a given call not all arguments need be supplied.
- **Complex arrays and arithmetic:** The fundamental data type is a multi-dimensional array of complex numbers. Important special

cases are matrices, vectors and scalars. All computation in MATLAB is performed to floating-point arithmetic and complex arithmetic is automatically used when the data is complex.

4.3 GRAPHICAL USER INTERFACE (GUI)

A graphical user interface is graphical display in one or more windows containing controls, called component, that enable a user to perform interactive tasks. Unlike coding programs to accomplish tasks, the user of a GUI need not understand the details of how the tasks are performed. GUI components can include menus, toolbars, push buttons, list boxes and sliders just to name a few. The number of software and language options that are available to build GUIs.

- FORTRAN (Visual Fortran 6.62)
- C/C ++ (Visual C++ 6/73)
- Java4 (Visual J++6/73)
- MATLAB5 (MATLAB R 13)
- Basic (Visual Basic 6/73)

GUIs created using MATLAB tools can also perform any type of computations, read and write data files, communicate with other GUIs and display data as tables or as plots.

4.3.1 WORKING OF A MATLAB GUI

A graphical user interface provides the user with a familiar environment to work. The environment contains pushbuttons, toggle buttons, lists, menus, text boxes, and so forth, all of which are already familiar to the user, so that he or she can concentrate on using the application rather than on the mechanics involved in doing things. However, GUIs are harder for the programmer because a GUI- based program must be prepared for mouse clicks (or possibly keyboard input) for any GUI element at any time. Such inputs are known as events, and a program that responds to events is said to be even driven. The three principal elements required to create a MATLAB Graphical User Interface are

- **Components:** Each item on a MATLAB GUI (pushbuttons), labels, edit boxes, etc.) is a graphical component.
  The types of components include graphical controls (pushbuttons, edit boxes, lists, sliders, etc.), static elements (frames and text strings), menus and axes. Graphical controls and static elements are created by the function uicontrol and menus are created by the functions uimenu and uicontext menu.
- **Figures:** The components of a GUI must be arranged within a figure is a window on the computer screen. In the past, figures have been created automatically whenever we have plotted data. However, empty figures can be created with the function figure and can be sued to hold any combination of components.
- **Callbacks:** A mouse click or a key press is an event, and the MATLAB program must respond to each event if the program is to perform its function. For example, if a user clicks on a button, that even must cause the MATLAB code that implements the functions of the button to be executed. The code executed in response to an event is known as a call back. There must be a callback to implement the function of each graphical component of GUI.

The basic GUI elements are summarized in Table 4.1

Table 4.1 GUI components

Element	**Created by**	**Description**
	Graphical Controls	
Pushbutton	Uicontrol	Pushbutton triggers a callback when clicked with a mouse.
Radio button	Uicontrol	A radio button is a type of toggle button that appears as a small circle with a dot in the middle when it is “on”. Each mouse click on a radio button triggers a callback.
Edit box	Uicontrol	An edit box displays a text string and allows the user to modify the information displayed. A callback is triggered when the user presses the Enter key.
Popup menus	Uicontrol	A popup menu is a graphical control that displays a series of text strings in response to a mouse click. When the popup menu is not clicked on, currently selected string is visible.
Slider	Uicontrol	A slider is a graphical control to adjust a value in a smooth, continuous fashion by dragging the control with a mouse. Each slider change triggers a callback.

Static Elements		
Frame	Uicontrol	Creates a frame, which is a rectangular box within a figure. Frames are used to group sets of controls together. Frames never trigger callbacks.
Text field	Uicontrol	Creates a label, which is text string located at a point on the figure. It never trigger callbacks.
Menus and Axes		
Menu items	Uimenu	Creates a menu items that trigger a callback taboos mouse button is released over them.
Axes	Axes	Creates a new set of axes to display data on. Axes never trigger callbacks.

4.3.2 BUILDING A MATLAB GUI

A MATLAB GUI is a figure window to which we add user-operated components. We can select size and position these components as we like. Using callbacks, we can make the components do what we want the user clicks or manipulates the components with keystrokes. We can build MATLAB GUIs in two ways:

- Use **GUIDE (**GUI Development Environment), an interactive GUI construction kit. This approach starts with a figure that we populate with components from within a graphic layout editor. GUIDE creates an associated code file controlling callbacks for the GUI and its components. GUIDE saves both the figure (as a FIG –file) and the code file.

- Create code files that generate GUIs as functions or scripts (Programmatic GUI construction). Using this approach, we create a code file that defines all component properties and behaviors. When a user executes the file, it creates a figure, populates it with components and handles user interactions. Typically, the figure is not saved because the code in the file creates a new one each time it runs.

The code files of the two approaches look different. Programmatic GUI files are generally longer, because they explicitly define every property of the figure and its controls, as well as the callbacks. GUIDE GUIs define most of the properties within the figure itself. They store the definitions in its FIG-file rather than it its code file. The code file controls callbacks and other functions that initialize the GUI when it opens.

By using this procedure, the GUI model for speech enhancement system has been developed. This model uses push buttons, slider, graph, edit text, static text, radio button and button group panel to develop the speech enhancement system. Figure 4.2 shows enhancement system developed in GUI. This system developed for four different types of algorithms such as IBM, IMM, IRM and OSM.

4.4 FEATURES OF THE GENERATED APPLICATION M-FILE

GUIDE simplifies the creation of GUI applications by automatically generating an M-file framework directly from our layout. We can then use this framework to code your application M- file. This approach provides a number of advantages.

- The M-file contains code to implement a number of useful features.
- The M-file adopts an effective approach to managing object handles and executing callback routines. The M-files provides a way to manage global data.
- The automatically inserted sub function prototypes for callback routines ensure compatibility with future releases.

Graphical user interface (GUIs) are being increasingly used in the classroom to provide users of computer simulations with a friendly and visual approach to specifying all input parameters, thus making it easier to describe what is needed to run a simulation. Data entry becomes much easier because of the visual and instead of trying to remember all of the different command line prompts or specifying inputs in nondescript, text input files. The most important benefit of a GUI is that it can post –process the results of the simulation providing the user with instant feedback.

The typical speech separation system and the proposed speech separation system has been tested successfully and both the systems were designed using MATLAB GUI. The Real time audio has been recorder using the microphone and the corresponding denoised speech signal for both the system has been obtained using MATLAB GUI. Simultaneously, the SNR and STOI has been calculated successfully for quality and intelligibility measurements.

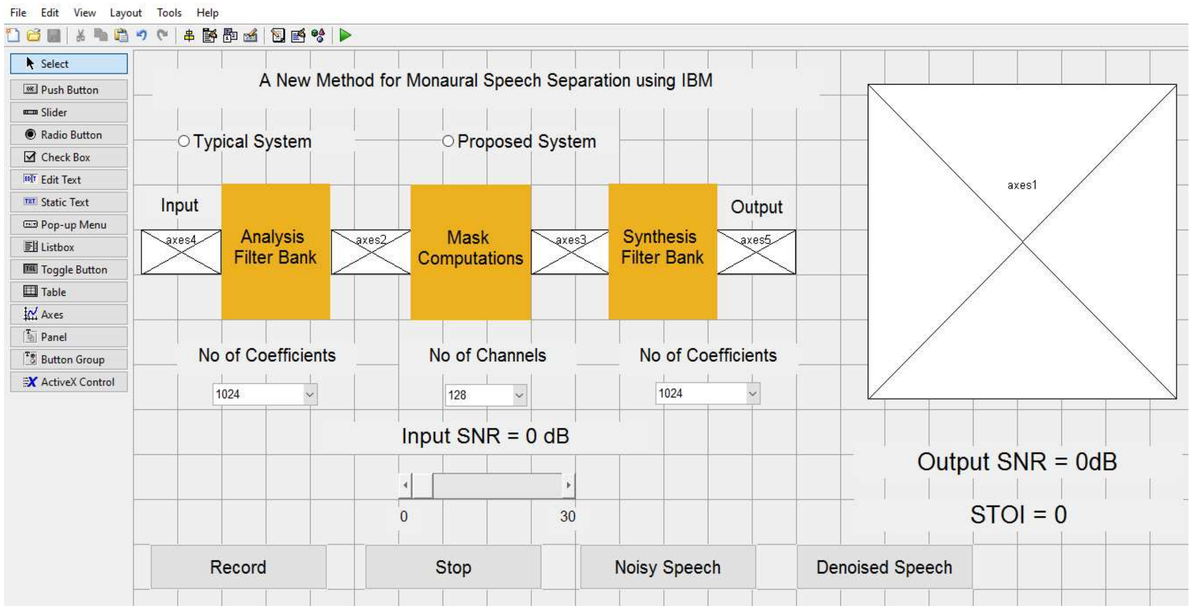


Fig 4.1 Sample.fig file of MATLAB GUI for both the systems

- It consists of two radio buttons to select either the typical speech separation system or proposed speech separation system. The number of co-efficients can be selected for both analysis and synthesis filter bank by using drop down menu box.
- The number of co-efficients which can be selected can be of values 1024, 512, 256. Then the number of channels which is common to all can be of the values 128, 64, 32. The input SNR can be selected using the scroll bar. When the arrow button of the scroll bar is clicked, the SNR increases in steps of 1dB. When the scroll bar is clicked in center, then SNR will be increased in steps of 5dB.
- The *Record button* is used to record the speech using microphone and stop button is used to stop recording the speech. Once, the stop button is clicked, the speech is saved as test.wav file.
- The *Noisy Speech button* is used to display the plot of Noisy speech (i.e.) Amplitude vs. Time plot of the noisy speech on the axes given at the right side of the layout and simultaneously it plays the noisy speech.

- The *Denoised Speech button* is used to display the plot of Denoised speech (i.e.) Amplitude vs. Time plot of the denoised speech on the axes given at the right side of the layout and simultaneously it plays the denoised speech.
- The Output SNR (in dB) will be displayed in the Static text box present below the graph.

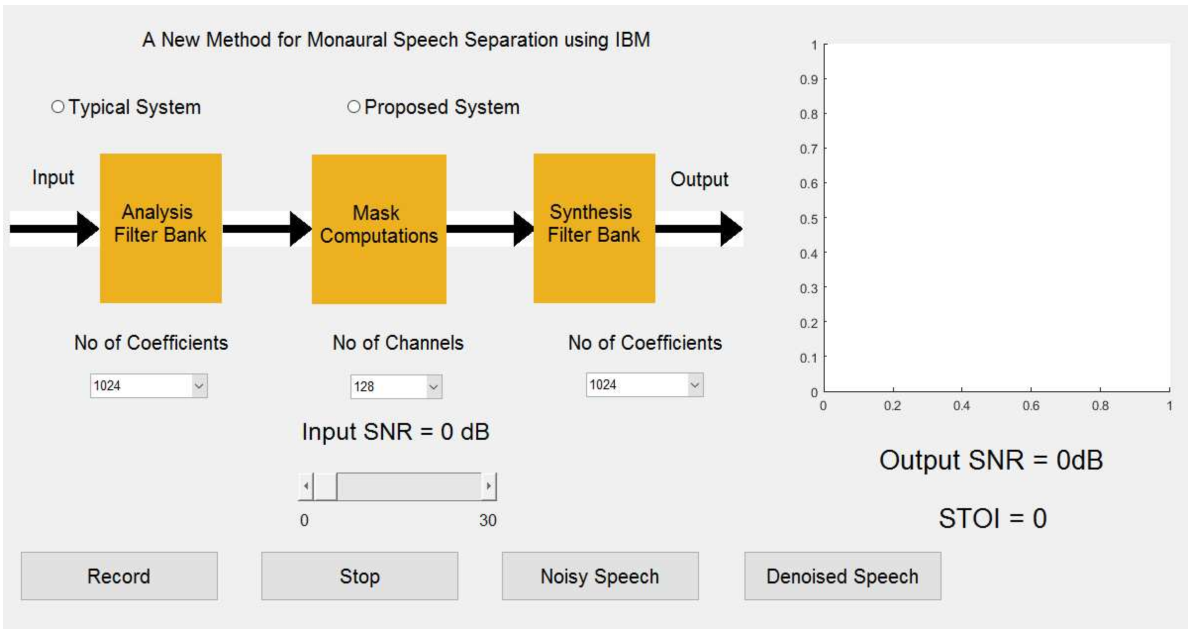
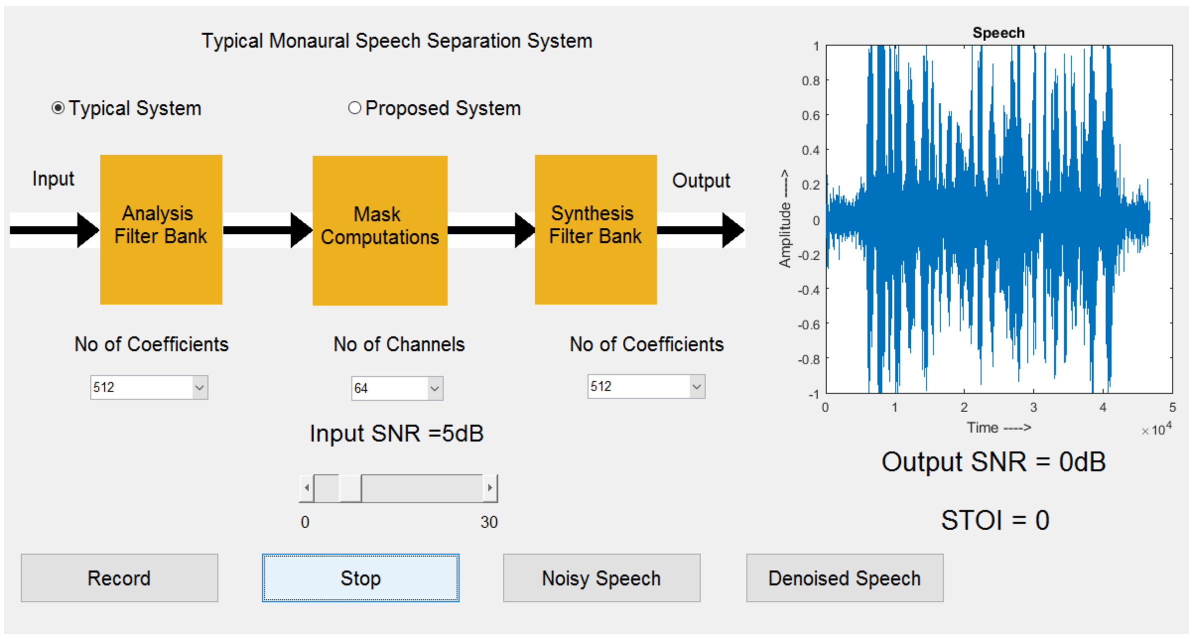


Fig 4.2 Sample output GUI layout without any inputs


Typical Monaural Speech Separation System

(● Typical System) ○ Proposed System

No of Coefficients 512

No of Channels 64

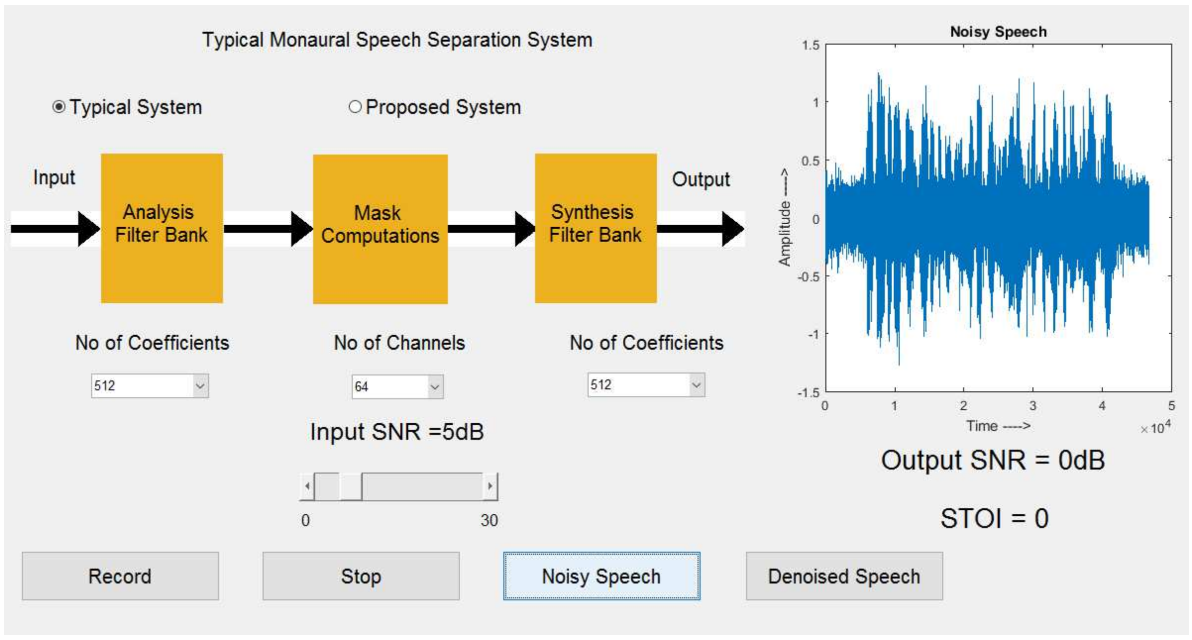
No of Coefficients 512

Input SNR =5dB

Output SNR = 0dB

STOI = 0

Fig 4.3 Plot of Speech signal for the typical monaural speech separation system with number of coefficients at 512 and number of channels at 64 and input SNR at 5dB


Typical Monaural Speech Separation System

(● Typical System) ○ Proposed System

No of Coefficients 512

No of Channels 64

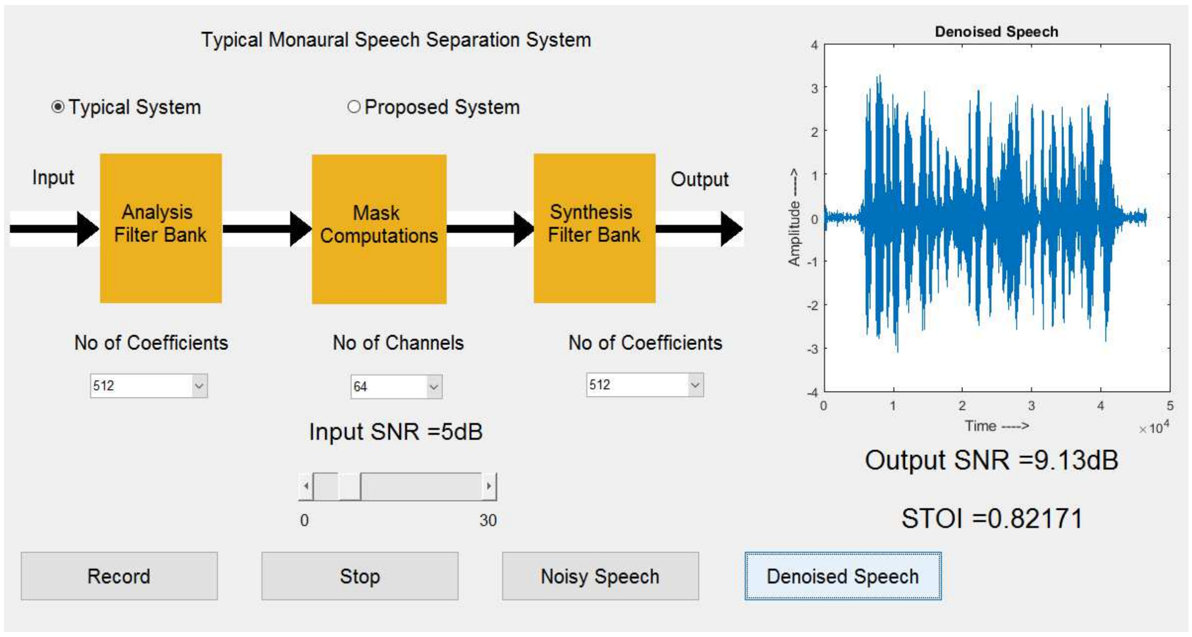
No of Coefficients 512

Input SNR =5dB

Output SNR = 0dB

STOI = 0

Fig 4.4 Plot of Noisy Speech signal for the typical monaural speech separation system with number of coefficients at 512 and number of channels at 64 and input SNR at 5dB


Typical Monaural Speech Separation System

[✓ Typical System] Proposed System

No of Coefficients: 512

No of Channels: 64

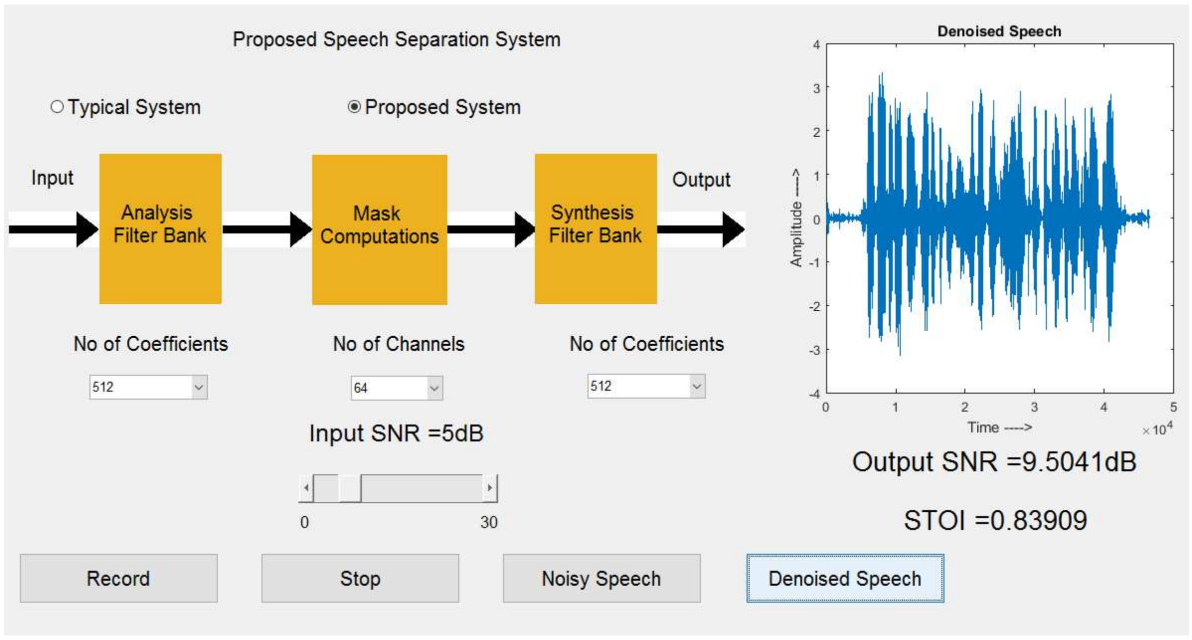
No of Coefficients: 512

Input SNR =5dB

Output SNR =9.13dB

STOI =0.82171

Fig 4.5 Plot of Denoised Speech signal and value of Output SNR and STOI for the typical monaural speech separation system with number of coefficients at 512 and number of channels at 64 and input SNR at 5dB


Proposed Speech Separation System

Typical System [✓ Proposed System]

No of Coefficients: 512

No of Channels: 64

No of Coefficients: 512

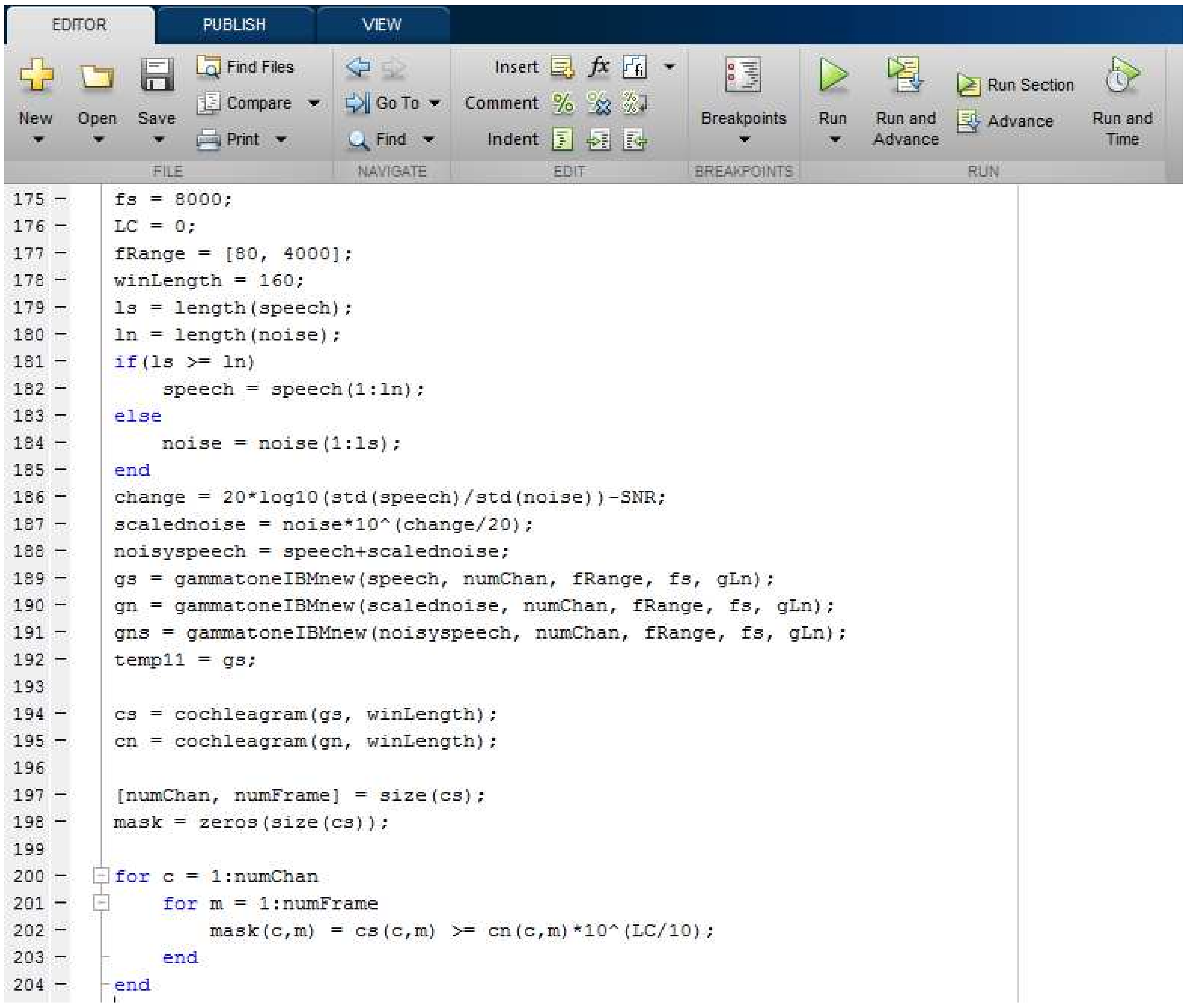
Input SNR =5dB

Output SNR =9.5041dB

STOI =0.83909

Fig 4.6 Plot of Denoised Speech signal and the value of Output SNR and STOI for the proposed speech separation system with number of coefficients at 512 and number of channels at 64 and input SNR at 5dB

Clearly, the figure 4.5 and 4.6 shows that the output SNR for the proposed system is slightly higher than the output SNR of the typical system. Hence, it can be concluded that the proposed system is computationally less complex than the Weintraub system and thereby the throughput of the proposed system is also more than the Weintraub system. The proposed system reduces the complexity of the Weintraub system without compromising on the quality and intelligibility of the speech signal.



```
fs = 8000;
LC = 0;
fRange = [80, 4000];
winLength = 160;
ls = length(speech);
ln = length(noise);
if(ls >= ln)
    speech = speech(1:ln);
else
    noise = noise(1:ls);
end
change = 20*log10(std(speech)/std(noise))-SNR;
scalednoise = noise*10^(change/20);
noisyspeech = speech+scalednoise;
gs = gammatoneIBMnew(speech, numChan, fRange, fs, gLn);
gn = gammatoneIBMnew(scalednoise, numChan, fRange, fs, gLn);
gns = gammatoneIBMnew(noisyspeech, numChan, fRange, fs, gLn);
temp11 = gs;

cs = cochleagram(gs, winLength);
cn = cochleagram(gn, winLength);

[numChan, numFrame] = size(cs);
mask = zeros(size(cs));

for c = 1:numChan
    for m = 1:numFrame
        mask(c,m) = cs(c,m) >= cn(c,m)*10^(LC/10);
    end
end
```

Fig 4.7 Sample code

The full working code of GUI is included in Appendix.

4.5 CONCLUSION

This chapter discussed in details about MATLAB GUI model of both the typical and proposed monaural speech separation system. Both the systems were tested with standard speech and noise database from IEEE speech corpus and Noisex – 92.

CHAPTER 5

PERFORMANCE ANALYSIS AND DISCUSSION

5.1 INTRODUCTION

Simulation is done using MATLAB. MATLAB is used due to its high performance in technical computing. It allows computation and programming in a user friendly environment. The original clean speech signals were taken from IEEE speech corpus and noise signals from Noisex – 92. Both the typical and proposed monaural speech separation systems were implemented using MATLAB. The analysis and synthesis filter banks were modelled using Gammatone filters. The denoised speech is compared with the clean speech and various performance measure parameters such as Signal to Noise Ratio (SNR) and Short Time Objective Intelligibility (STOI) were calculated.

5.2 SPEECH AND NOISE DATABASE

In speech separation systems, CASA based IBM technique is used to separate the speech signal from noisy speech. For computing the IBM, a prior knowledge about speech and noise signal has to be known. The speech signals to conduct the experiment are obtained from IEEE corpus and noise signals are obtained from Noisex – 92 are used to realize the performance of both the typical and proposed speech separation systems.

The speech signals are:

1. Clean.wav – *"The sky that morning was clear and bright blue"*.
2. IEEEFemale.wav – *"The drip of the rain made a clear sound"*.

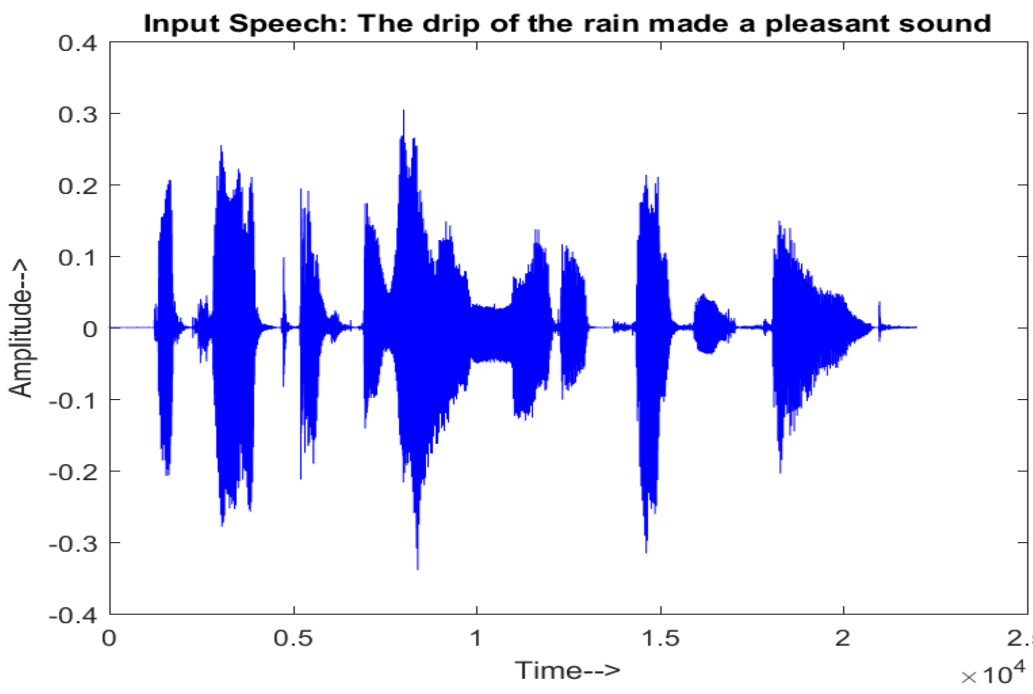


Fig 5.1 (a) Plot of input speech IEEEFemale.wav - "The drip of the rain made a pleasant sound"

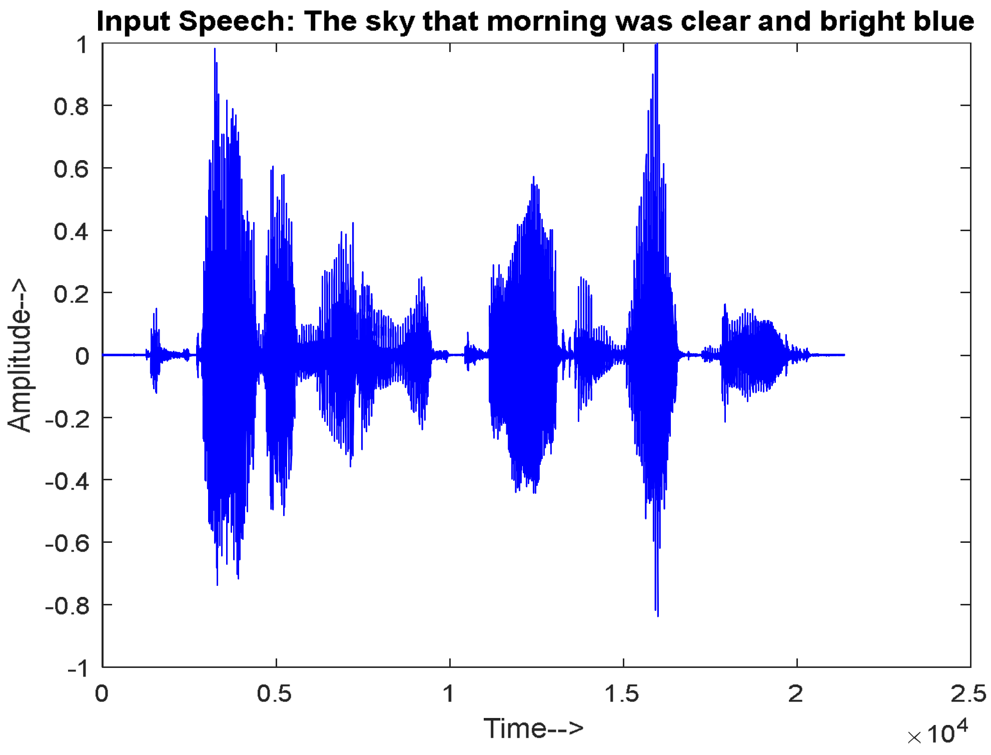


Fig 5.1 (b) Plot of input speech clean.wav - "The sky that morning was clear and bright blue"

The different Noise signals are:

1. *"Babble noise"*.
2. *"Factory Noise"*.
3. *"Speech Shaped Noise"*.

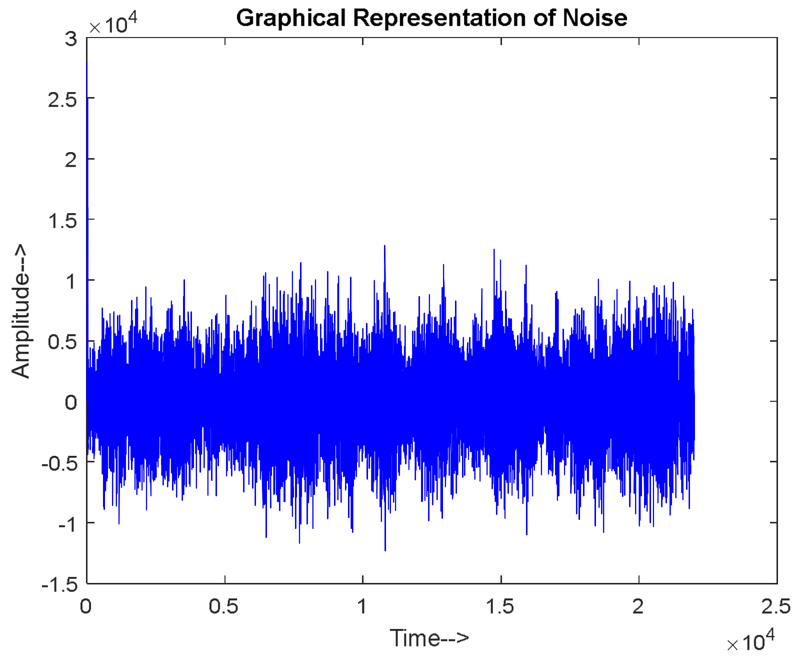


Fig 5.2 (a) Plot of noise signal – Babble noise

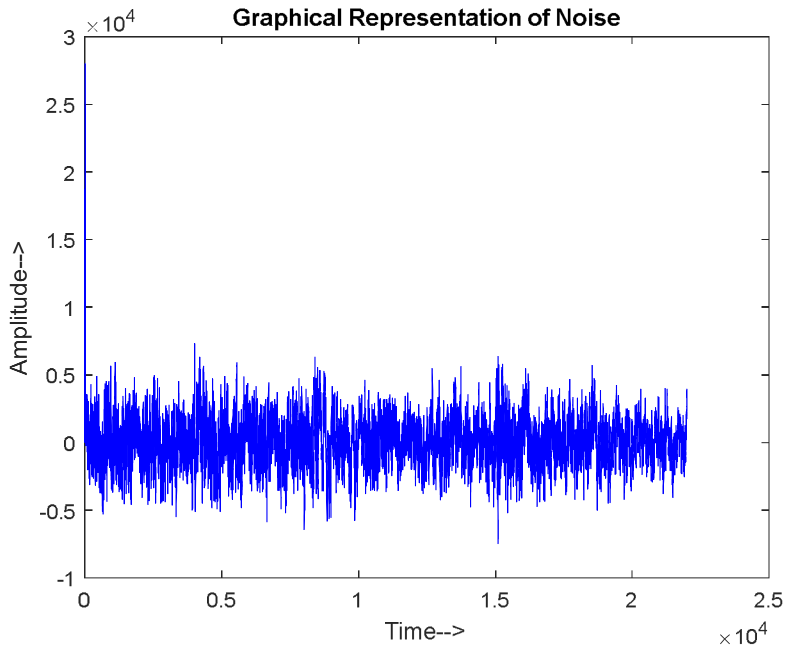


Fig 5.2(b) Plot of noise signal – Factory noise

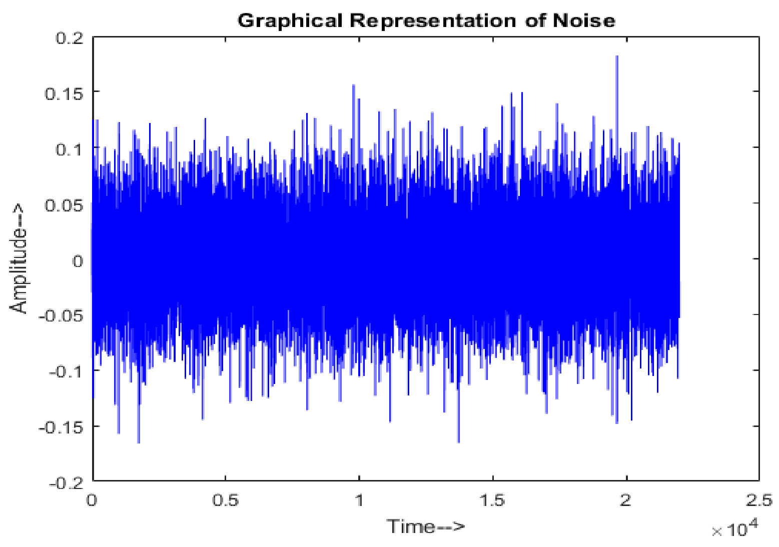


Fig 5.2(c) Plot of noise signal – speech shaped noise

5.3 PERFORMANCE MEASURE

The performance of the typical and proposed monaural speech separation system is discussed in the following sections. The performance is measured using Signal to Noise Ratio (SNR) [18] [18] as a quality measure and Short Time Objective Intelligibility (STOI) [23] as intelligibility measure. The performance of the system is measured by comparing the denoised speech from both the system with that of the clean speech signal.

5.3.1 QUALITY – SIGNAL TO NOISE RATIO (SNR)

Signal-to-noise ratio (abbreviated SNR or S/N) is a measure used in engineering that compares the level of a desired signal to the level of background noise. SNR is defined as the ratio of signal power to the noise power, often expressed in decibels. A ratio higher than 1:1 (greater than 0 dB) indicates more signal than noise and a ratio less than 1:1 (lesser than 0 dB) indicates more noise than the speech signal. The SNR improvement of a signal is calculated using the formula given in equation 5.1.

$$SNR = 10\log\left(\frac{\sum_n S_{oneall}(n)^2}{\sum_n \left(S_{oneall}(n) - S_{out}(n)\right)^2}\right) \tag{5.1}$$

Here $S_{oneall}(n)^2$ stands for the clean input speech signal and $\mathrm{S_{out}(n)}$ stands for the enhanced output speech signal.

5.3.2 INTELLIGIBILITY – SHORT TIME OBJECTIVE INTELLIGIBILITY (STOI)

The STOI is used in this work as the intelligibility measure which is a simple and reliable objective measure based on short time segments. Generally, the value of STOI will be in the range of 0 to 1. The value of STOI is 1 means the enhanced speech is same as the clean speech and 0 means the enhanced speech has no correlation with the clean speech. STOI is an intelligibility measure which compares the temporal envelopes of the clean and degraded speech in short-time regions by means of a correlation coefficient.

5.4 EXPERIMENTAL RESULTS AND DISCUSSION

The above two systems were implemented in Matlab and tested using standard speech database. System Specifications in which the experiments were conducted is given below

Intel® Core™ i5-3210M CPU@2.50Ghz, RAM: 4.00GB, 64- bit operating System, Windows 10 Home edition.

MATLAB Version: R2015a.

In the experiments conducted the number of channels was chosen to be 128 for both the analysis and synthesis filter bank. The length of the impulse response in analysis and synthesis filter bank for both systems was fixed at

1024. The filter banks were implemented using 4th order Gammatone filters. The input speech signals were taken from IEEE corpus and the noise signals were taken from Noisex-92. The performance of both the systems was evaluated in terms of speech quality and intelligibility. The quality of the enhanced speech by the proposed system is measured using Signal to Noise Ratio (SNR) improvement and intelligibility is measures using Short Time Objective Intelligibility (STOI).

Table 5.1: *Filer Bank Specifications*

Parameters	**Analysis Filter Bank**	**Synthesis Filter Bank**
Order	4	4
Number of Channels	128	128
Length of Impulse Response	1024	1024

The noisy speech signals are obtained by mixing the clean speech signal with the noise signals in different SNR values. The SNR values are chosen between -5dB to 15dB in steps of 5dB. The scaling of noise is done using the formula given below,

Scaled noise = noise×10(change/20)

where change = $20\times\log(10\times(\frac{\text{std(speech)}}{\text{std(noise)}} - \text{SNR}))$.

here std(x) is the standard deviation of variable x and the SNR is input SNR for which the signal is to be mixed with the noise. This noisy speech is used to determine the SNR improvement and STOI of the proposed system at various input SNRs.

Figure 5.3 shows the plot of noisy speech by mixing the clean speech (IEEEFemale.wav) with the noise signal (speech shaped noise) at input SNR of 5dB. The output SNR in Typical speech separation system is found to be 9.7100 which is an improvement of 94.2% with respect to the input SNR. The plot of the denoised speech from the typical system is shown in figure 5.4. Now the same signal is processed using the proposed system which gives the output SNR of 9.9143. The output SNR of this system shows an improvement of 98.28% with respect to the input SNR. The output denoised speech signal from the proposed system is shown in figure 5.5.

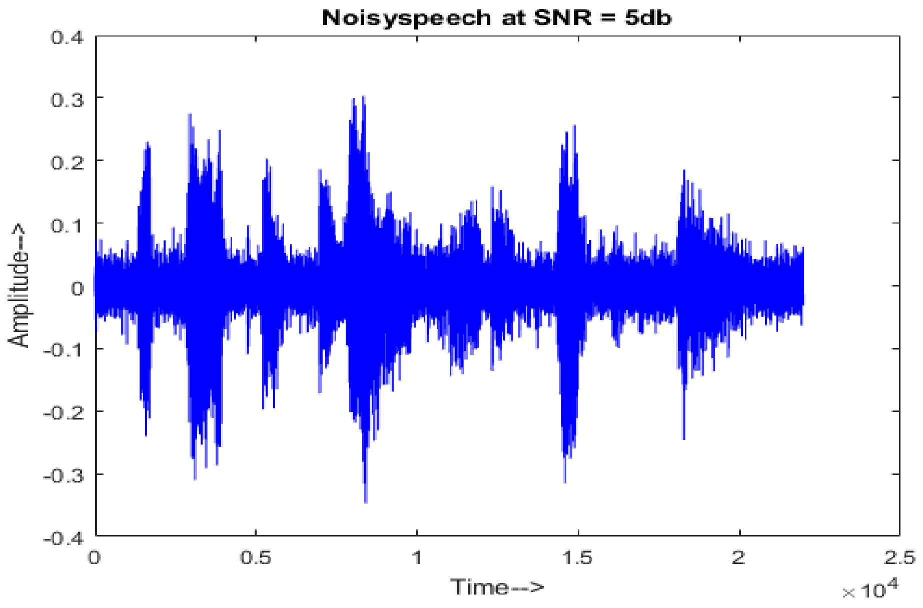


Fig 5.3 Plot of noisy speech signal at SNR = 5dB

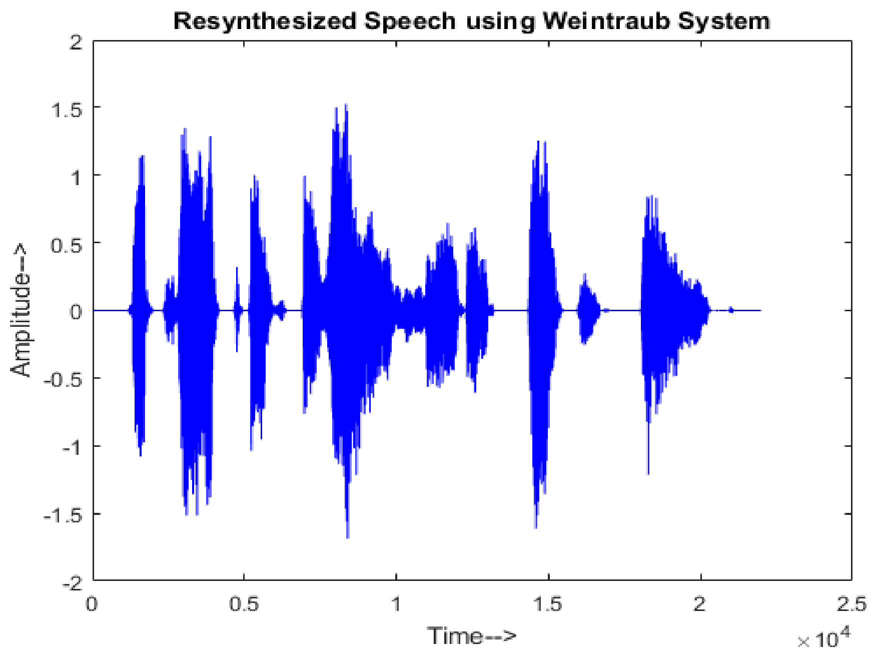


Fig 5.4 Plot of resynthesized speech signal using Weintraub system

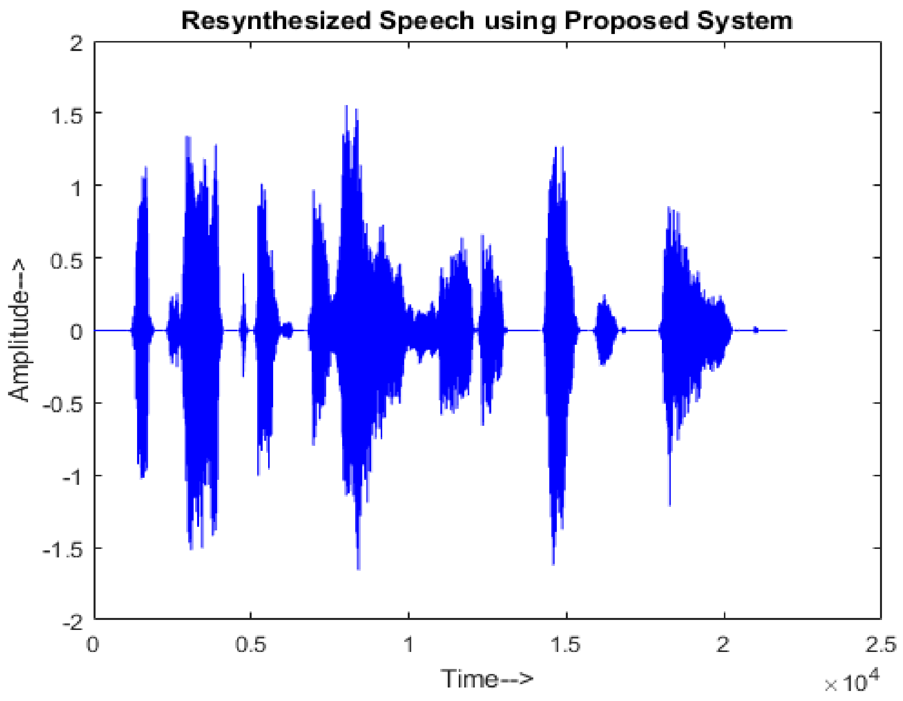


Fig 5.5 Plot of resynthesized speech signal using Proposed system

A clean speech signal from IEEE speech corpus is mixed with the babble noise, factory noise and speech shaped noise from Noisex-92 to obtain the noisy speech signal at SNRs in the range of -5dB to 15dB.The input speech is the IEEEFemale.wav file of length 21982 samples and clean.wav of length 21371 samples, sampled with a sampling frequency of 8000Hz. The noise signal is the factorynoise.wav of length 70001 samples for Table 5.2 and Table 5.6. For Table 5.3 and Table 5.7, the speech signal is clean.wav of length 21371 samples and the noise signal is babble.wav of length 44000 samples. Similarly, Table 5.4 and Table 5.8 uses IEEEFemale.wav as the speech signal and speechshapednoise.wav of length 147989 samples as the noise signal. Tables 5.2 -5.5 gives a quality measure of output SNR and table 5.6-5.8 is gives the STOI values for various input SNRs ranging from -5dB to 15dB. Each tables uses a different combination of speech and noise signal from the speech and noise database.

Table 5.2: *SNR improvement of the proposed system for the factory noise at various input SNRs*

INPUT SNR(dB)	**OUTPUT SNR(dB)**		**IMPROVEMENT(dB)**
	TYPICAL SYSTEM	**PROPOSED SYSTEM**	
-5	7.6064	7.9325	0.3261
0	10.4869	10.8704	0.3835
5	13.9506	14.3267	0.3761
10	17.6725	17.9212	0.2487
15	21.5577	21.7607	0.2030
AVERAGE	14.25482	14.5623	0.30748

Table 5.3: *SNR improvement of the proposed system for the babble noise at various input SNRs*

INPUT SNR(dB)	OUTPUT SNR(dB)		IMPROVEMENT(dB)
	TYPICAL SYSTEM	PROPOSED SYSTEM	
-5	5.9454	6.2761	0.3307
0	8.4391	8.8221	0.3830
5	11.5351	11.9661	0.4310
10	15.0150	15.2413	0.2263
15	18.7670	18.8917	0.1247
AVERAGE	11.94032	12.23946	0.29914

Table 5.4: *SNR improvement of the proposed system for the speech shaped noise at various input SNRs*

INPUT SNR(dB)	OUTPUT SNR(dB)		IMPROVEMENT(dB)
	TYPICAL SYSTEM	PROPOSED SYSTEM	
-5	4.5874	4.9005	0.3131
0	6.6499	6.9077	0.2578
5	9.7100	9.9143	0.2043
10	13.6690	13.8812	0.2122
15	17.6510	17.8724	0.2214
AVERAGE	10.45346	10.69522	0.24176

Clearly, From Table 5.2 - 5.4, it is observed that, the proposed system improves the output SNR with an average value of 0.30748 dB for factory noise, 0.29914 dB for babble noise and 0.24176 dB for speech shaped noise respectively. The last row gives the average of output SNR obtained for the various input SNRs in both the systems This clearly shows that the proposed system improves the speech quality by improving the SNR.

Table 5.5: *Quality Measure*

ANALYSIS FILTER BANK	SYNTHESIS FILTER BANK	OUTPUT SNR (dB)	
		TYPICAL SYSTEM	PROPOSED SYSTEM
1024	1024	9.6742	9.8804
1024	512	9.6742	9.8804
512	512	9.6742	9.8804
512	512/128	9.6659	9.8807

In table 5.5, the length of the impulse response in analysis and synthesis filter bank is changed from 1024 to 512 in both the typical and proposed monaural speech separation systems. The performance of both the systems remains the same even on reducing the length of impulse response of gammatone filters. The last row in table 5.5 uses a length of 512 samples in analysis filter bank but the synthesis filter bank uses a length of 512 for the first 14 channels and the remaining 114 channels uses a length of 128 samples. This is because the human voice frequency lies in the 85Hz to 255Hz. This frequency range corresponds to channels 1 to 14, as a result the length of the impulse response is preserved to be 512 and the remaining high frequency channels are noise dominant hence the length of impulse response in these channels are reduced to 128. This further reduces the computational complexity without compromising on the quality and intelligibility of the denoised speech signal in the proposed monaural speech separation system. On the other hand, the above changes if implemented in the typical monaural speech separation system will reduce the quality of the denoised speech. This is because the mask is multiplied only after the stage of resynthesis in the typical monaural speech separation system. This is one main advantage in the proposed monaural speech separation system.

Table 5.6: *The STOI value of the proposed system for the factory noise*

INPUT SNR(dB)	STOI		IMPROVEMENT
	TYPICAL SYSTEM	PROPOSED SYSTEM	
-5	0.8011	0.8086	0.0075
0	0.8629	0.8709	0.0080
5	0.9016	0.9234	0.0218
10	0.9367	0.9471	0.0104
15	0.9595	0.9636	0.0041
AVERAGE	0.89236	0.90272	0.01036

Table 5.7: *The STOI value of the proposed system for the babble noise*

INPUT SNR(dB)	STOI		IMPROVEMENT
	TYPICAL SYSTEM	PROPOSED SYSTEM	
-5	0.7754	0.7902	0.0148
0	0.8048	0.8262	0.0214
5	0.8560	0.8660	0.0100
10	0.9060	0.9107	0.0047
15	0.9487	0.9521	0.0034
AVERAGE	0.85818	0.86904	0.01086

Table 5.8: *The STOI value of the proposed system for the speech shaped noise*

INPUT SNR(dB)	STOI		IMPROVEMENT
	TYPICAL SYSTEM	PROPOSED SYSTEM	
-5	0.7109	0.7602	0.0493
0	0.8548	0.8595	0.0047
5	0.8833	0.8937	0.0104
10	0.9171	0.9331	0.0160
15	0.9462	0.9504	0.0042
AVERAGE	0.86246	0.87938	0.01692

Similarly, from Table 5.6 - Table 5.8, it observed that, the proposed system improves the STOI with an average value of 0.01036 for factory noise, 0.01086 for babble noise and 0.01692 for speech shaped noise. The value of STOI increases with increase in input SNR for both the systems. This clearly shows that the proposed system improves the speech intelligibility compared to the existing typical monaural speech separation system by increasing the STOI value.

5.5 COMPARISON OF COMPUTATIONAL COMPLEXITY

In addition to the above SNR improvement and STOI, computation time and throughput also considered to show the performance of the proposed system. Table 5.9 – table 5.10 compares the computation time and throughput of the proposed system with the typical monaural speech separation system. In both the tables the computation time and throughput are calculated by varying the length of impulse response of the gammatone filters in analysis and synthesis filter banks.

5.5.1 COMPUTATION TIME

Computation time is defined as the time taken by the speech separation system to perform all the computations. In other words, it is the time required by the system to denoise the input noisy speech signal.

Table 5.9: *Comparison of computational complexity – computation time*

ANALYSIS FILTER BANK	SYNTHESIS FILTER BANK	COMPUTATION TIME (Secs)	
		TYPICAL SYSTEM	**PROPOSED SYSTEM**
1024	1024	1.6775	1.3867 (0.2908)
1024	512	1.6038	1.3429 (0.2609)
512	512	1.4708	1.2992 (0.1716)
512	512/128	1.3813	1.2156 (0.1657)

From table 5.9, it is observed that the proposed system reduces the computation time to denoise the noisy speech signal compared to that of the existing typical monaural speech separation system. It is mainly due to the number of zeroes introduced in the resynthesized signal because of mask multiplication. This enables the synthesis filter bank to process the speech signal with reduced computational delay.

5.5.2 THROUGHPUT

Throughput is defined as the number of samples processed per unit time. Throughput depends on the number of input samples and the computation time. Throughput is the rate at which samples are processed by the system.

Table 5.10: *Comparison of computational complexity – Throughput*

ANALYSIS FILTER BANK	SYNTHESIS FILTER BANK	THROUGHPUT (Samples per Sec)	
		TYPICAL SYSTEM	PROPOSED SYSTEM
1024	1024	453	517 (64)
1024	512	486	524 (38)
512	512	511	548 (37)
512	512/128	527	554 (27)

The reduction in the computation time of the proposed monaural speech separation system results in increased throughput of the system. The last row in the above table 5.10 shows the improvement of throughput in the proposed system compared to the typical speech separation system.

5.5.3 NUMBER OF MULTIPLICATION

The other parameters used for comparing the computational complexity is the number of multiplications and additions involved in the system. The lesser the number of multiplications and additions, lesser will be the computation time. The number of multiplications and additions involved is calculated based on the number of frames, number of samples in each frame, the length of impulse response of Gamma tone filter and length of the input noisy speech signal. The example given below for calculating the number of multiplications and additions uses the IEEEFemale.wav as the speech signal and speech shaped noise as the noise signal. The number of multiplications and additions will not change for change in input SNR. The number of multiplications and additions will change for change in the length of impulse response in analysis and synthesis filter bank.

Table 5.11: *Comparison of computational complexity – number of multiplications*

ANALYSIS FILTER BANK	SYNTHESIS FILTER BANK	NUMBER OF MULTIPLICATIONS	
		TYPICAL SYSTEM	PROPOSED SYSTEM
1024	1024	5768060928	5768060928
1024	512	4327448576	4327448576
512	512	2886836224	2886836224
512	512/128	1924552192	1924552192

The number of multiplications in Analysis Filter Bank = Length of Speech signal × Length of Impulse response × Number of Channels = 21982 × 1024 × 128 = 2881224704.

The number of multiplications in the process of mask multiplication = Number of Channels × Number of Frames × Length of each Frame = 128 × 274 × 160 = 5611520.

Similarly, the number of multiplications in Synthesis Filter Bank = 21982 × 1024 × 128 = 2881224704.

Total number of multiplications = 2881224704 + 5611520 + 2881224704 = 5768060928.

5.5.4 NUMBER OF ADDITIONS

The number of additions in both the systems are calculated for the three stages. The number additions also depend on the length of the input speech signal, number of channels and number of frames. Reducing the number of additions will also reduce the computation time.

Table 5.12: *Comparison of computational complexity – number of additions*

ANALYSIS FILTER BANK	SYNTHESIS FILTER BANK	NUMBER OF ADDITIONS	
		TYPICAL SYSTEM	PROPOSED SYSTEM
1024	1024	5759635712	5759635712
1024	512	4319023360	4319023360
512	512	2878411008	2878411008
512	512/128	1916126976	1916126976

The number of additions in analysis filter bank = Length of Speech signal × (Length of Impulse response -1) × Number of Channels = 21982 × (1024 – 1) × 128 = 2878411008.

Similarly, the number of additions in synthesis filter bank = 2878411008.

The number additions involved in reconstruction of speech from various channels after the synthesis filter bank stage = Length of Speech signal × Number of Channels = 21982 × 128 = 2813696.

Total number of additions = 2878411008 + 2878411008 + 2813696 = 5759635712.

Table 5.13: *Comparison of Typical Speech Separation System and Proposed Speech Separation System.*

PARAMETERS	TYPICAL SYSTEM	PROPOSED SYSTEM
Number of Multiplications	5768060928	5768060928
Number of Additions	5759635712	5759635712
Computational Time(sec)	3.32	2.51
Throughput(samples per sec)	453	517

Table 5.13 uses IEEEFemale.wav as the speech signal and factorynoise.wav as the noise signal. The two signals are mixed at a SNR of 5dB to get the noisy speech. The analysis and synthesis filter bank uses 128 channels and the length of gammatone filter in analysis and synthesis filter bank is taken to be 1024.

The number of multiplications and additions involved in both the systems are same. However, there is reduction of 24.4% in the computation time of the proposed system which is evident from Table 5.13. It is mainly due to the number zeros introduced after multiplying the IBM with the T-F frames of noisy speech signal. From Table 5.13 it is also observed that, the reduction of computation time leads to 14.13 % improvement of throughput.

Hence, it can be concluded that the proposed system is computationally less complex than the Weintraub system and thereby the throughput of the proposed system is also more than the Weintraub system. The proposed system reduces the complexity of the Weintraub system without compromising on the quality and intelligibility of the speech signal.

5.6 CONCLUSION

This chapter discussed the results obtained from the typical and proposed monaural speech separation system using MATLAB. Both the systems were tested with standard speech and noise databases. The two systems were compared in terms of quality - SNR, intelligibility – STOI and computational complexity – computation time and throughput.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This project work proposed a new method for speech resynthesis in monaural speech separation system without compromising on the quality and intelligibility of the enhanced speech. The proposed method improves the speech quality and intelligibility with minimal computational delay and higher throughput by changing the order of operation.

The results of both the systems were obtained by implementing them on Matlab. SNR and STOI were used as quality and intelligibility measures for comparison of the two systems. The results of the experiment conducted for input speech and various noise signals were tabulated in Chapter 5.

FUTURE WORK

The future work of this research work concentrates on implementing the proposed system using hardware. In CASA based the IBM is computed by comparing the energy of clean speech and noise in each T-F units. However, in a real time scenario it is not possible to obtain a clean speech for computing the IBM. So the future work of this research work will concentrate on finding a method to implement the same using hardware.

REFERENCES

1. Abrar Hussain, Kalaivani Chellappan, Siti Zamratol M, (2016) "Single Channel Speech Enhancement Using Ideal Binary Mask Technique Based On Computational Auditory Scene Analysis", *Journal of Theoretical and Applied Information Technology,* Vol.91. No.1.

2. Belhedi Wiem, Ben Messaoud Mohamed Anouar, Bouzid Aichl, (2016) "Time-Frequency Masks for Monaural Speech Separation: A Comparative Review", *7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT).*

3. Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen, (2011) "An algorithm for intelligibility prediction of time frequency weighted noisy speech", *IEEE Transactions on Audio, Speech and Language Processing*, Vol.19, No:7, pp. 2125-2136.

4. D.L. Wang, G.J. Brown, (2006) "Fundamentals of Computational Auditory Scene Analysis", in *Computational Auditory Scene Analysis, D.L Wang and G.J Brown, Wiley-IEEE Press*, pp. 1-38.

5. D.L. Wang, (2008) "Time–Frequency Masking for Speech Separation and Its Potential for Hearing Aid Design," *Trends in Amplification,* Vol. 12, pp.332-353.

6. E.H Rothauser et al (1969), "IEEE recommended practice for speech quality measurements", *IEEE Transactions on Audio and Electroacoustics*, Vol.17, pp.225–246.

7. G Hu, D Wang, (2004) "Monaural speech segregation based on pitch tracking and amplitude modulation," *IEEE Transactions on Neural Networks*, Vol. 15, Issue. 5, pp. 1135–1150, Sept 2004.

8. J. Jensen, J.H.L. Hansen, (2001) "Speech enhancement using a constrained iterative sinusoidal model," *IEEE Transactions on Speech and Audio Processing,* Vol.9, Issue.7, pp. 731-740.

9. Jihen Zeremdini, Mohamed Anouar Ben Messaoud and Aicha Bouzid, (2015) "A comparison of several computational auditory scene analysis (CASA) techniques for monaural speech segregation", *Brain Informatics,* Vol. 2, Issue 3, pp. 155 – 166.

10. Ke Hu and DeLiang Wang, (2013) "An Unsupervised Approach to Cochannel Speech Separation", *IEEE Transactions on Audio, Speech and Language Processing,* Vol.21, No.1.

11. Lawrence R. Rabiner, Ronald W. Schafer, (2011) "Time Domain Methods for Speech Processing", in *Theory and Applications of Digital Speech Processing,Lawrence R. Rabiner, Ronald W. Schafer, Pearson Education, Prentice Hall,* pp 287-379.

12. M. Weintraub (1985), "A theory and computational model of auditory monaural sound separation," Ph.D. dissertation, Dept. Elect. Eng., Stanford Univ., Stanford, CA.

13. Masoud Geravanchizadeh and Reza Ahmadnia, (2014) "Monaural Speech Enhancement Based On Multi-Threshold Masking," *In Blind Source Separation*, G.R. Naik, W. Wang, Springer Berlin Heidelberg, pp.369-393.

14. N. Harish Kumar, R. Rajavel, (2014) "Monaural speech separation system based on optimum soft mask," *IEEE Int. Conf. on Computational Intelligence and Computing Research,* 18-20.

15. Noisex-92(2014).
"http://www.speech.cs.cmu.edu/comp.speech/Section1/Data/noisex.html".

16. R.D. Patterson, I. Nimmo-Smith, J. Holdsworth, P. Rice, (1988) "An efficient auditory filterbank based on the gammatone function," *MRC Applied Psych*. Unit. 2341.

17. Sameti H, Sheikhzadeh H, Deng L, Brennan RL, (1998) "HMM based strategies for enhancement of speech signals embedded in nonstationary noise", *IEEE Transactions on speech and audio processing*, Vol.6, Issue.5, pp. 445–455.

18. S. Shoba, R. Rajavel, (2017) "Adaptive energy threshold for monaural speech separation" *International Conference on Communication and Signal Processing.*

19. Shoba. S, Rajavel. R, (2017) "Image processing techniques for segments grouping in monaural speech separation" *Circuits, Systems, and Signal Processing (Springer)*, "https://doi.org/10.1007/s00034-017-0728-x".

20. Stevan F. Boll (1979) "Suppression of acoustic noise in speech using spectral subtraction" *IEEE Transactions on Acoustics, Speech and Signal Processing,* Vol. 2, pp. 113-120.

21. V. Hofmann (2002), "Frequency analysis and synthesis using a Gammatone filter bank," *Acta Acustica united with Acustica,* Vol. 88, pp. 433 – 442.

22. V.A. Mane, Prof. Dr. S. B. Patil, (2016) "Survey of Methods and challenges in Computational Auditory sense analysis", *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering*, Vol. 4, Issue 9.

23. Y. Ephraim and H. L. Trees, "A signal subspace approach for speech enhancement", (1995), *IEEE Trans. Speech Audio Processing,* Vol. 3, pp.251 – 266.

APPENDIX

The full code of Matlab GUI has been included below:

```
function varargout = Fourth_reviewrecord(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                'gui_Singleton',  gui_Singleton, ...
                'gui_OpeningFcn', @Fourth_reviewrecord_OpeningFcn, ...
                'gui_OutputFcn',  @Fourth_reviewrecord_OutputFcn, ...
                'gui_LayoutFcn',  [], ...
                'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
end

function Fourth_reviewrecord_OpeningFcn(hObject, eventdata, handles,
varargin)
handles.output = hObject;

handles.Fs = 8000;
handles.recorder = audiorecorder(handles.Fs,8,1);
axes(handles.axes2);
imshow('images.png');
axes(handles.axes3);
imshow('images.png');
axes(handles.axes4);
imshow('images.png');
axes(handles.axes5);
imshow('images.png');

[a,map] = imread('audio.jpg');
[r,c,d] = size(a);
x = ceil(r/30);
```

```
y = ceil(c/30);
g = a(1:x:end, 1:y:end, :);
g(g==255)=5.5*255;
set(handles.play_noise, 'CData', g);

guidata(hObject, handles);
end
function varargout = Fourth_reviewrecord_OutputFcn(hObject, eventdata,
handles)
varargout{1} = handles.output;
end

function popupmenu_nocmc_Callback(hObject, eventdata, handles)
end

function popupmenu_nocmc_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
end

function pushbutton_ns_Callback(hObject, eventdata, handles)
 [speech]=audioread('test.wav');
noise_pop = get(handles.noise, 'Value');
if noise_pop==1
   [noise]=audioread('babble.wav');
elseif noise_pop==2
   [noise]=audioread('factorynoise.wav');
elseif noise_pop==3
   [noise]=audioread('speechshapednoise.wav');
end

SNR = get(handles.slider1,'Value');
ls = length(speech);
ln = length(noise);
if(ls >= ln)
   speech = speech(1:ln);
else
   noise = noise(1:ls);
end
change = 20*log10(std(speech)/std(noise))-SNR;
scalednoise = noise*10^(change/20);
```

```
noisyspeech = speech+scalednoise;
soundsc(noisyspeech);
axes(handles.axes1);
plot(noisyspeech);
xlabel('Time ---->');
ylabel('Amplitude ---->');
title('Noisy Speech');
end


function pushbutton_dns_Callback(hObject, eventdata, handles)

option1 = get(handles.radiobutton1, 'Value');
option2 = get(handles.radiobutton2, 'Value');
SNR = get(handles.slider1,'Value');
a = get(handles.popupmenu_nocmc, 'Value');
if a==1
   numChan = 128;
elseif a==2
   numChan = 64;
elseif a==3
   numChan = 32;
end
b = get(handles.popupmenu_nocafb, 'Value');
if b==1
   gLn = 1024;
elseif b==2
   gLn = 512;
elseif b==3
   gLn = 256;
end
c = get(handles.popupmenu_nocsfb, 'Value');
if c==1
   gL = 1024;
elseif c==2
   gL = 512;
elseif c==3
   gL = 256;
end

[speech]=audioread('test.wav');
noise_pop = get(handles.noise, 'Value');
if noise_pop==1
```

```
    [noise]=audioread('babble.wav');
elseif noise_pop==2
    [noise]=audioread('factorynoise.wav');
elseif noise_pop==3
    [noise]=audioread('speechshapednoise.wav');
end
fs = 8000;
LC = 0;
fRange = [80, 4000];
winLength = 160;
ls = length(speech);
ln = length(noise);
if(ls >= ln)
    speech = speech(1:ln);
else
    noise = noise(1:ls);
end
change = 20*log10(std(speech)/std(noise))-SNR;
scalednoise = noise*10^(change/20);
noisyspeech = speech+scalednoise;
gs = gammatoneIBMnew(speech, numChan, fRange, fs, gLn);
gn = gammatoneIBMnew(scalednoise, numChan, fRange, fs, gLn);
gns = gammatoneIBMnew(noisyspeech, numChan, fRange, fs, gLn);
temp11 = gs;

cs = cochleagram(gs, winLength);
cn = cochleagram(gn, winLength);

[numChan, numFrame] = size(cs);
mask = zeros(size(cs));

for c = 1:numChan
    for m = 1:numFrame
        mask(c,m) = cs(c,m) >= cn(c,m)*10^(LC/10);
    end
end

filterOrder = 4;
winShift = winLength/2;
increment = winLength/winShift;
[numChan,numFrame] = size(mask);
sigLength = length(noisyspeech);
r = zeros(1,sigLength);
```

```
r1 = zeros(1,sigLength);
for i = 1:winLength
   coswin(i) = (1 + cos(2*pi*(i-1)/winLength - pi))/2;
end

phase(1:numChan) = zeros(numChan,1);
erb_b = hz2erb(fRange);
erb = [erb_b(1):diff(erb_b)/(numChan-1):erb_b(2)];
cf = erb2hz(erb);
b = 1.019*24.7*(4.37*cf/1000+1);

midEarCoeff = zeros(1,numChan);
for c = 1:numChan
   midEarCoeff(c) = 10^((loudness(cf(c))-60)/20);
end

gt = zeros(numChan,gL);
tmp_t = [1:gL]/fs;
for c = 1:numChan
   gain = 10^((loudness(cf(c))-60)/20)/3*(2*pi*b(c)/fs).^4;
   gt(c,:) = gain*fs^3*tmp_t.^(filterOrder-1).*exp(-
2*pi*b(c)*tmp_t).*cos(2*pi*cf(c)*tmp_t+phase(c));
end

AllOneMask=zeros(numChan, numFrame);
AllOneMask(:,:)=1;




if option1 == 1
   for c = 1:numChan
      temp1(c,:) = gns(c,:);
      temp1(c,:) = fliplr(temp1(c,:))/midEarCoeff(c);
      temp2 = fftfilt(gt(c,:),temp1(c,:));
      temp1(c,:) = fliplr(temp2(1:sigLength))/midEarCoeff(c);
      weight = zeros(1, sigLength);
      for m = 1:numFrame-increment/2+1
         startpoint = (m-1)*winShift;
         if m <= increment/2
```

```
            weight(1:startpoint+winLength/2) =
weight(1:startpoint+winLength/2) + mask(c,m)*coswin(winLength/2-
startpoint+1:end);
         else
            weight(startpoint-winLength/2+1:startpoint+winLength/2) =
weight(startpoint-winLength/2+1:startpoint+winLength/2) +
mask(c,m)*coswin;
         end
      end
      r = r + temp1(c,:).*weight;
   end
   for c = 1:numChan
      temp11(c,:) = fliplr(temp11(c,:))/midEarCoeff(c);
      temp21 = fftfilt(gt(c,:),temp11(c,:));
      temp11(c,:) = fliplr(temp21(1:sigLength))/midEarCoeff(c);
      weight = zeros(1, sigLength);
      for m = 1:numFrame-increment/2+1
         startpoint = (m-1)*winShift;
         if m <= increment/2
            weight(1:startpoint+winLength/2) =
weight(1:startpoint+winLength/2) + AllOneMask(c,m)*coswin(winLength/2-
startpoint+1:end);
         else
            weight(startpoint-winLength/2+1:startpoint+winLength/2) =
weight(startpoint-winLength/2+1:startpoint+winLength/2) +
AllOneMask(c,m)*coswin;
         end
      end
      r1 = r1 + temp11(c,:).*weight;
   end

   else if option2 == 1
      for c = 1:numChan
         weight = zeros(1, sigLength);
         for m = 1:numFrame-increment/2+1
            startpoint = (m-1)*winShift;
            if m <= increment/2
               weight(1:startpoint+winLength/2) =
weight(1:startpoint+winLength/2) + mask(c,m)*coswin(winLength/2-
startpoint+1:end);
            else
```

```
                weight(startpoint-winLength/2+1:startpoint+winLength/2) =
weight(startpoint-winLength/2+1:startpoint+winLength/2) +
mask(c,m)*coswin;
            end
        end
        temp1(c,:) = gns(c,:).*weight;
        temp1(c,:) = fliplr(temp1(c,:))/midEarCoeff(c);
        temp2 = fftfilt(gt(c,:),temp1(c,:));
        temp1(c,:) = fliplr(temp2(1:sigLength))/midEarCoeff(c);
        r = r + temp1(c,:);
    end

    for c = 1:numChan
        weight = zeros(1, sigLength);
        for m = 1:numFrame-increment/2+1
            startpoint = (m-1)*winShift;
            if m <= increment/2
                weight(1:startpoint+winLength/2) =
weight(1:startpoint+winLength/2) + AllOneMask(c,m)*coswin(winLength/2-
startpoint+1:end);
            else
                weight(startpoint-winLength/2+1:startpoint+winLength/2) =
weight(startpoint-winLength/2+1:startpoint+winLength/2) +
AllOneMask(c,m)*coswin;
            end
        end
        temp11(c,:) = gs(c,:).*weight;
        temp11(c,:) = fliplr(temp11(c,:))/midEarCoeff(c);
        temp21 = fftfilt(gt(c,:),temp11(c,:));
        temp11(c,:) = fliplr(temp21(1:sigLength))/midEarCoeff(c);
        r1 = r1 + temp11(c,:);
    end
  end
end
  SNR = 10*log10(sum(r1.^2)/sum((r1-r).^2));
  SNRout = strcat('Output SNR = ', num2str(SNR) , 'dB');
  set(handles.output_snr, 'String', SNRout);

  stoi_val = stoi(speech,r,8000)
  STOIout = strcat('STOI = ',num2str(stoi_val));
  set(handles.stoi, 'String', STOIout);

  axes(handles.axes1);
```

```
    plot(r);
    xlabel('Time ---->');
    ylabel('Amplitude ---->');
    title('Denoised Speech');
    soundsc(r);
end


function popupmenu_nocafb_Callback(hObject, eventdata, handles)
end


function popupmenu_nocafb_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function popupmenu_nocsfb_Callback(hObject, eventdata, handles)
end
function popupmenu_nocsfb_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function slider1_Callback(hObject, eventdata, handles)

SNR = get(handles.slider1,'Value');
SNR1 = strcat('Input SNR = ', num2str(SNR) , 'dB');
set(handles.input_snr,'String', SNR1);

end

function slider1_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
```

```
function radiobutton1_Callback(hObject, eventdata, handles)
if (get(hObject,'Value') == get(hObject,'Max'))
   set(handles.radiobutton2, 'Value' ,0);
   set(handles.title, 'String', 'Typical Monaural Speech Separation System');
end
end
function radiobutton2_Callback(hObject, eventdata, handles)
if (get(hObject,'Value') == get(hObject,'Max'))
   set(handles.radiobutton1, 'Value' ,0);
   set(handles.title, 'String', 'Proposed Speech Separation System');
end
end


function record_Callback(hObject, eventdata, handles)
record(handles.recorder);
end

function stop_Callback(hObject, eventdata, handles)
stop(handles.recorder);
global myRecording
myRecording = getaudiodata(handles.recorder);
axes(handles.axes1);
plot(myRecording);
xlabel('Time ---->');
ylabel('Amplitude ---->');
title('Speech');
audiowrite('test.wav',myRecording,8000);

end


function noise_Callback(hObject, eventdata, handles)

end


function noise_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
```

```
end


function play_noise_Callback(hObject, eventdata, handles)

noise_pop = get(handles.noise, 'Value');
if noise_pop==1
   [noise]=audioread('babble.wav');
   soundsc(noise);
   axes(handles.axes1);
   plot(noise);
   xlabel('Time ---->');
   ylabel('Amplitude ---->');
   title('Babble Noise');
elseif noise_pop==2
   [noise]=audioread('factorynoise.wav');
   soundsc(noise);
   axes(handles.axes1);
   plot(noise);
   xlabel('Time ---->');
   ylabel('Amplitude ---->');
   title('Factory Noise');
elseif noise_pop==3
   [noise]=audioread('speechshapednoise.wav');
   soundsc(noise);
   axes(handles.axes1);
   plot(noise);
   xlabel('Time ---->');
   ylabel('Amplitude ---->');
   title('Speechshapednoise Noise');
end

end
```