CHALMERS

Digital Communication over Speech Compressed Channel

Sigurdur Sverrisson
Xiaoyun Liang

Division of Communication Systems
Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden

EXE028/2008

Abstract

Transmitting digital data through a voice compressed channel can be desirable in some situations. It is, however, not a straight forward task. In this thesis, two methods are presented and simulated to perform this task. First method uses synthetic speech but it did not give a very good result while the second one with regular communication system modulation has achieved a promising BER.

CONTENTS

1 Introduction

1.1 Background

During the last two decades, telecommunication has become one of the leading technologies in the world and mobile communication, as one of the main branches, is no doubt playing a very important role in everyday's life. Nowadays people are making phone calls at any place and any time. To some extend, people would like to keep their personal or business phone calls confidential. The GSM system, for example, has its own internal encryption on the air link. However, it is easily to be decoded in real time or in some cases it is not used at all.

In order to solve this problem, an external encryption could be implemented by using the existing mobile phones and transmitting the encrypted digital data stream through a voice compressed channel. The GSM data channel is not suitable in this case because it is not designed for real-time application and would cause too much delay compared to the voice channel. Also we want this method to work with all GSM mobiles, so we cannot rely on some method to access the radio channel directly. Therefore we want to use the voice channel. This system could be implemented as an handsfree add-on to the GSM mobiles..

Transmitting digital data through the voice compressed channel is not a straight forward task. The compression is done by extracting parameters from the speech at the encoder and then synthesize it at the decoder. Because of this, the input signal of the channel might be totally different from the output signal. To solve this, two methods will be discussed in this paper. The first one uses synthetic speech. It is basically to map the data to a speech-like signal and then transmit it through the channel. Another method which does not provide speech-like signal is to use the conventional communication system to transmit the data.

In this paper we will have a thorough discussion on how those two methods work and how they will perform in the simulation which have either software or hardware channel. However first the discussion will be focused on speech coding which is specifically relevant to GSM system. Finally the results of both of those methods will be presented and analyzed.

1.2 Problem Definition

The problem of transmitting digital through a voice compressed channel is that these channels are not designed for this purpose, but to transmit speech. For the first method, all the references are about the synthetic speech method and for the other method, there are no reference model that can be used to calculate the BER.

1.3 Notation

Abbreviations

3G	3rd Generation
AbS	Analysis-by-Synthesis
ACELP	Algebraic CELP
AGC	Automatic Gain Control
AMR	Adaptive Multi-Rate
ASK	Amplitude Shift Keying
BER	Bit Error Rate
CELP	Code Excited Linear Predict
DA	Data Assisted
DD	Data Directed
DPSK	Differential Phase Shift Keying
ETSI	European Telecommunication Standardisation Institute
FR	Full Rate
FSK	Frequency Shift Keying
FTP	File Transfer Protocol
GSM	Global System for Mobile communication
LP	Linear Predictor
LPC	Linear Predictor Coefficients
LSF	Line Spectral Frequency
LSP	Line Spectral Pair
LTP	Long Term Predictor
MF	Matched Filter
MSK	Minimum Shift Keying
NDA	Non-Data Assisted
OFDM	Orthogonal Frequency Division Multiplexing
PN	Psudo-random Number
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
RRC	Root Raised Cosine
SIM	Subscriber Identity Module
SMQ	Split Matrix Quantization
VQ	Vector Quantizer
VAD	Voice Activity Detector

Symbols

α	The coefficients of the LP filter
β	The pitch filter gain
D	Pitch filter delay
I	Pitch filter order
L	Size of the processing block

N	Number of samples per symbol
M	Constellation size
R	Autocorrelation
T_s	Symbol time
p	LP filter order
x	Samples after MF
z	Samples after sampling the output from MF

1.4 Speech Coding

1.4.1 Code-Excited Liner Predictor

One of the most widely used speech coding method is CELP. It uses a scheme called AbS and uses codebooks to store the excitation for the time-varying filters. The basic outline of CELP is depicted in Figure (1.1) [1].

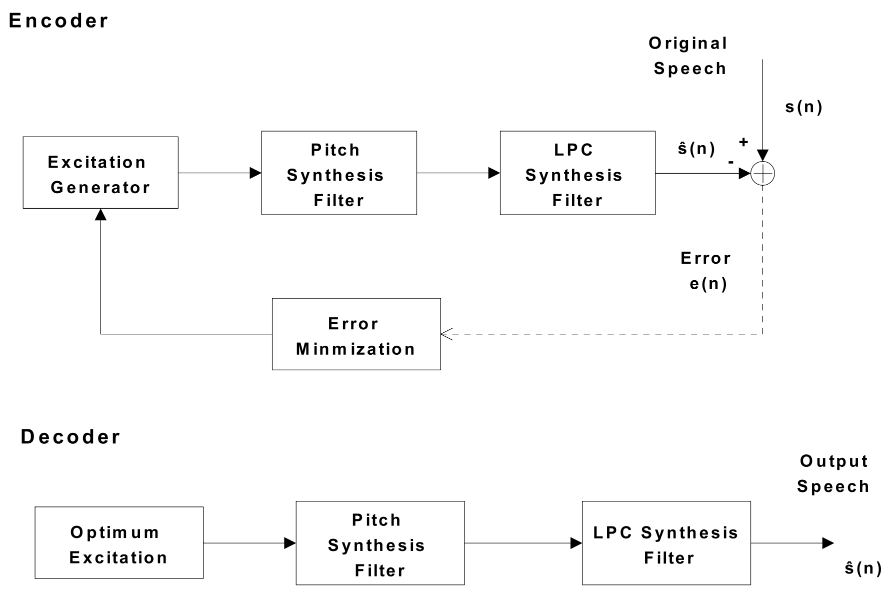

Figure 1.1. *Block diagram showing the basic outline of CELP systems.*

The original speech is first digitized and then partitioned into frames of 20-30 ms and further divided into subframes of 40 or 60 samples. Then the frame is windowed with a Hamming window (often modified to be asymmetric window) which can extend into past or future frames to improve the coding of the boundaries. After that an LP analysis is performed on the frame according to Equation (1.2), most often a 10th order filter is used. This analysis will model the spectral envelope. The synthesizer will then use a filter described in Equation (1.1). The LPC are often converted to LSP or LSF for efficient coding.

For each subframe an LTP analysis is performed which is modeled by Equation (1.3). This is to find the delay D which minimizes the error between the original signal and the signal from the pitch filter. And after that, find the pitch gain, β. The filter order, I, is usually 1 or 3. This LTP process is computationally complicated but still there are several methods to reduce the complexity [1].

$$\frac{1}{A(z)} = \frac{1}{1 - \sum_{i=1}^{p} \alpha_i z^{-i}} \tag{1.1}$$

$$\sum_{j=1}^{p} \alpha_j R_n\left(|i-j|\right) = R_n(i), \qquad 1 \leq i \leq p \tag{1.2}$$

$$P(z) = 1 - \sum_{i=-I}^{I} \beta_i z^{-D-i} \tag{1.3}$$

Once α, D and β have been determined, the excitation will be determined. It is done by using a codebook of excitation vectors and each of these vectors is passed through the LP and pitch filter. Then the one which minimizes the error between the signal and the original signal is chosen.

There are many different variations of CELP, which differ in the way the excitation codebook is stored.[1]. Most notably for our purposes here is ACELP. This method uses small number of pulses which are positioned in interleaved tracks for efficient coding. So each pulse position is severely restricted but together they can make up most of the necessary combinations for good excitation. The pulse amplitudes are usually restricted to be ± 1. In most cases, for efficient coding, there are only 4 to 5 tracks and only 1 or 2 non-zero pulses with either positive or negative amplitude in each track. An example is in Table (1.3) [1].

Track	Pulse number	Possible locations
1	i_0, i_5	0,5,10,15,20,25,30,35
2	i_1, i_6	1,6,11,16,21,26,31,36
3	i_2, i_7	2,7,12,17,22,27,32,37
4	i_3, i_8	3,8,13,18,23,28,33,38
5	i_4, i_9	4,9,14,19,24,29,34,39

Table 1.3. *ACELP excitation table, showing 5 tracks with two pulses in each track and is using a 40 sample long subframe. This is used in the EFR speech coder.*

1.4.2 Enhanced Full Rate

There are a couple of standards for the GSM system, which relate to the speech coding. These are FR, EFR and AMR. Here we will focus on EFR because it is a reasonably new standard and is implemented widely in the GSM system. However AMR, as the newest speech codec, is used in the 3G system and even some new GSM systems. When AMR is using the maximum bit rate, it is compatible to EFR. But since we are focusing on the GSM, we chose to use EFR. The full standard for EFR speech coder can be found in [2].

EFR uses ACELP to compress speech. It processes the speech in blocks of 20 ms and subframes of 40 samples. There are two LP analysis performed on each frame using two different windows. Those windows are asymmetric modified Hamming windows and extend into half of the previous frame.

The codebook has 5 tracks and allows two pulses in each track. That gives 10 non-zeros pulses out of 40 samples. See Table (1.3).

The GSM channel supports 13 kbps but the EFR coder transmits at 12.2 kbps. The additional bits are used for channel coding. This means that for each frame, which is 20 ms, there are 244 bits transmitted.

Voice Activity Detector

Voice activity detector is an algorithm used in the GSM system to determine if the input signal is a speech or not. This is achieved by comparing the energy of a filtered version of the input signal with a threshold. The presence of speech is indicated whenever the threshold is exceeded. If it detects no speech then comfort noise is transmitted instead of the input signal [3].

Speech Frame Format

Parameters	Subframes 1 & 3	Subframes 2 & 4	Total per frame
2 LSP sets			38
Pitch delay	9	6	30
Pitch gain	4	4	16
Algebraic code	35	35	140
Codebook gain	5	5	20
Total			244

Table 1.4. *Bit allocation of the EFR speech coder for a 20 ms frame*

All the parameters are put into a speech frame when they are quantized by the EFR analyzer. As the input of the synthesizer, the speech frame is used to synthesize speech. The bit distribution after the quantization is in Table (1.4). In EFR the LP coefficients are quantized as LSF using SMQ. This results in 5 LSF quantization parameters which index to a corresponding sub-matrix. The pitch delay is an index pointing to the corresponding pitch and the pitch delay is quantized using a non-uniform scalar quantizer. The algebraic code includes the quantized EFR excitation.

The GSM system uses channel coding for these parameters but it uses different level of protection for the different parameters. In EFR there are three classes of protection, class 1a, 1b and 2. Class 1a is protected with cyclic code and convolutional code, class 1b has only convolutional code and class 2 has no protection. There are 182 bits in class 1 and 78 bits in class 2. For more details, see [4]. For our purposes, it would be better if we can transmit our data using the protected bits.

2 Methods

2.1 System Evaluation

In order to evaluate the performance of our systems we need to do some simulations. Analytical methods are not suitable in this case since the system is very complicated and it would take very long time to reach good models. And there are no references about this that we can find. There are many ways to evaluate the system but in this case we will use simulations based not only on software but also include hardware device in some certain part. But before that, we need to simulate the channel by using EFR. The reference C code can be obtained from the ETSI homepage [5]

2.1.1 Software in the loop

Our main development tool is Matlab. This was chosen because it is very efficient to manipulate matrices and also we are very familiar with Matlab. After we have downloaded the C code, we need to make it work in Matlab environment. Matlab is capable to run compiled C code with a built-in functionality. However it requires some modification to the C code and it has to include Matlab methods. After these modifications, the resulting compiled file is called a MEX file. See Appendix (A.1) for more information.

2.1.2 Hardware in the loop

In this case, the channel is not simulated by software but we are using real cell phones instead. However the data transmitted from the cell phones is then processed in Matlab.

The hardware setup is as follows: Two cell phones, Nokia 6111 and Sony Ericsson T290i. Two computers, Intel Macbook and IBM ThinkPad X41. The Nokia phone was connected to the IBM computer by using bluethooth's voice gateway and the Sony Ericsson phone was connected to the Mac with a cable from the company IPdrum and the cable is called Mobile Skype Cable [6]. A diagram of the setup is in Figure (2.1). The Mac is used as the main computer which processes the data and we only record the audio on the PC and transfer it back to the Mac via FTP. The mobiles both have SIM cards from the operator Tele2 in Sweden. For more details about the mobile connection to the computers, see Appendix (A.2)

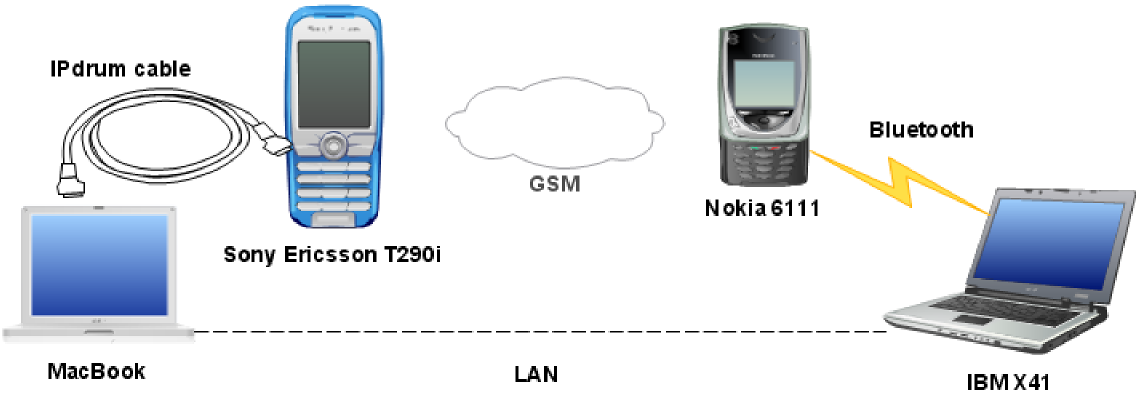

Figure 2.1. *A block diagram showing the connections between the computers and the phones.*

2.2 Synthetic Speech Method

When referring to a voice compressed channel, the first thing that comes to mind is to create a synthetic speech by mapping all the digital data to the speech. Using this method, we should be able to get the best performance because this method uses the properties of the channel. Besides, it should also be able to transmit the data reliably, even under bad radio channel condition.

2.2.1 Previous Work

This method has already been tried out and is described in [7] and [8]. It describes a method of transmitting digital data as synthetic speech using 3 features of speech:

- Speech spectral envelope represented by Line Spectral Frequencies (LSF).
- Fundamental frequency or pitch
- The ACELP excitation shape and energy

These features will be mostly preserved when transmitted through the voice channel. The input data is mapped to those features using 3 codebooks and then input to the synthesizer, see Figure (2.2). In the receiver there is an analyzer for those features and then the receiver will check the codebooks for the closest match to the received parameters, see Figure (2.3). The bit assignment is described as follows: 10 bits for LSF, 5 bits for pitch and 5 bits for energy. The total 20 bits are transmitted in 20 ms. This will give a total bit rate of 1 kbps. Higher bit rate is achieved by bigger codebooks.

In [9] a real-time hardware system is implemented. The hardware implementation has some additional problems such as AGC, VAD and various filters. The results of

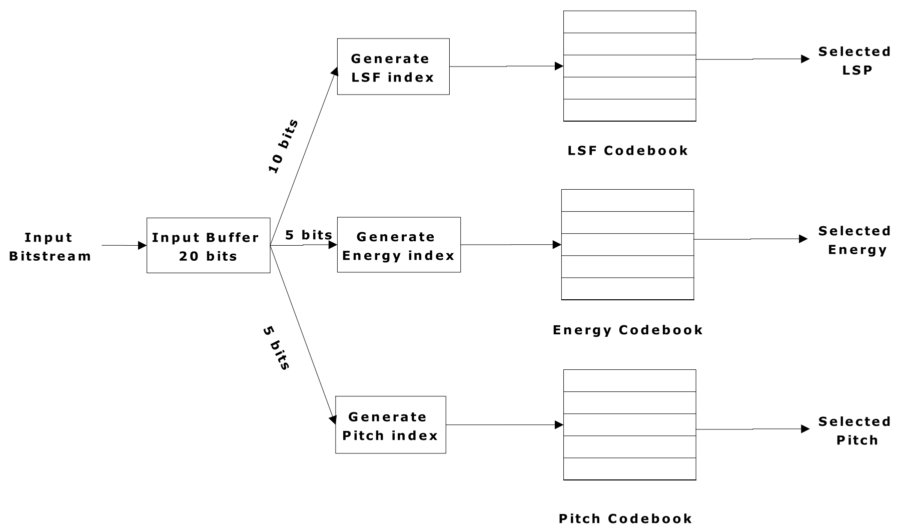

Figure 2.2. *Block diagram showing the modulator for the synthetic speech method.*

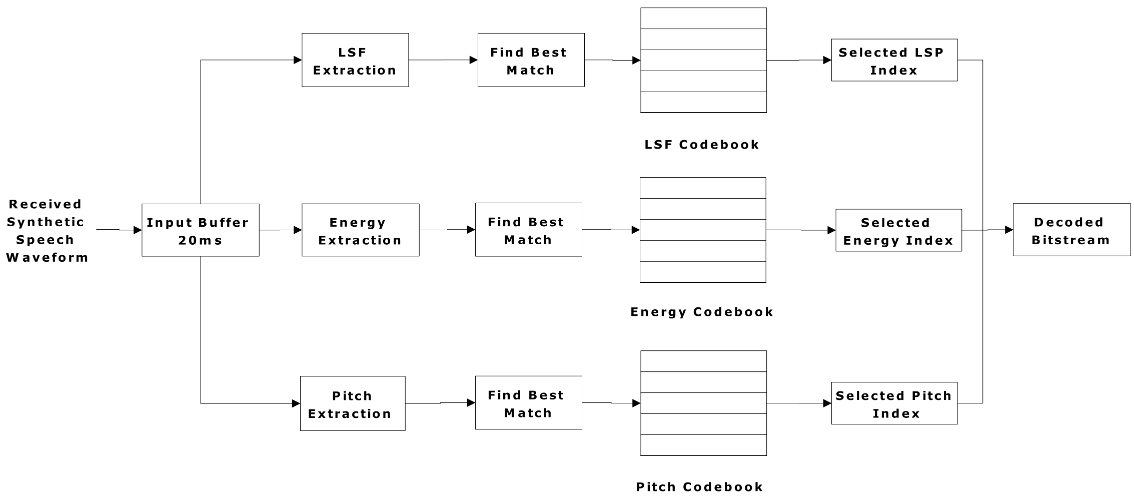

Figure 2.3. *Block diagram showing the demodulator for the synthetic speech method*

these simulations are in Table (2.1).

2.2.2 Our Work

Our goal in this part is to recreate the method described in Section 2.2.1. There are two main tasks to be achieved. One is to decide which kind of speech coder to use and the other is to design the codebooks. For the first task, we consider using EFR speech coder because it is simpler to be implemented. But there are also many other choices e.g. AMR or Speex etc. But for the method described above, we do

Interface	Before channel decoding		After channel decoding	
	Rate kbps	**BER %**	**Rate kbps**	**BER %**
Digital/Analog	3.0	2.9	1.7	0.40
Digital/Analog	3.0	2.9	1.2	0.03

Table 2.1. *The result from the hardware simulations of [9]*

not know what kind of speech codec was used.

The codebook design is the biggest part of this method. It maps the ditigal data to the specific features and then input them to the synthesizer. As mentioned above, it is not specified in the paper how the codebook is designed. The codebook design is dependent on the choice of speech codec. Here are the two methods we tried to use to populate the codebooks.

Histogram method

The idea here is to generate the codebooks with the most frequently used parameters from a sample speech. The assumption here is that it is easier to transmit the most frequently used parameters. After analyzing the sample speech from EFR analyzer, a histogram plot is done for each of the parameters, as can be seen from Figure (2.4).

For example, the LSF parameter has 5 indices and the first one is quantized to 7 bits. If we want to transmit 2 bits on that parameter, then the histogram is split into 4 intervals and the maximum is found in each interval and put into the codebook. This is done to avoid that all the selections are very close. The same is done for the other LSF indices. Thus the codebook is the combination of all the maximum values which is found in the histogram.

The same is done for the pitch and the energy.

Genetic Algorithms

Genetic algorithms is a method for blind optimization. First the problem has to be defined and in GA terms it is called a genome. A genome can be defined in many ways and the effectiveness of this method depends on how good you define the problem. Then we choose a subset of the genomes and that is called the population and then this population will evolve using mutation and crossover of the genomes. Also the fittest genome will survive. Then there is a fitness function to evaluate

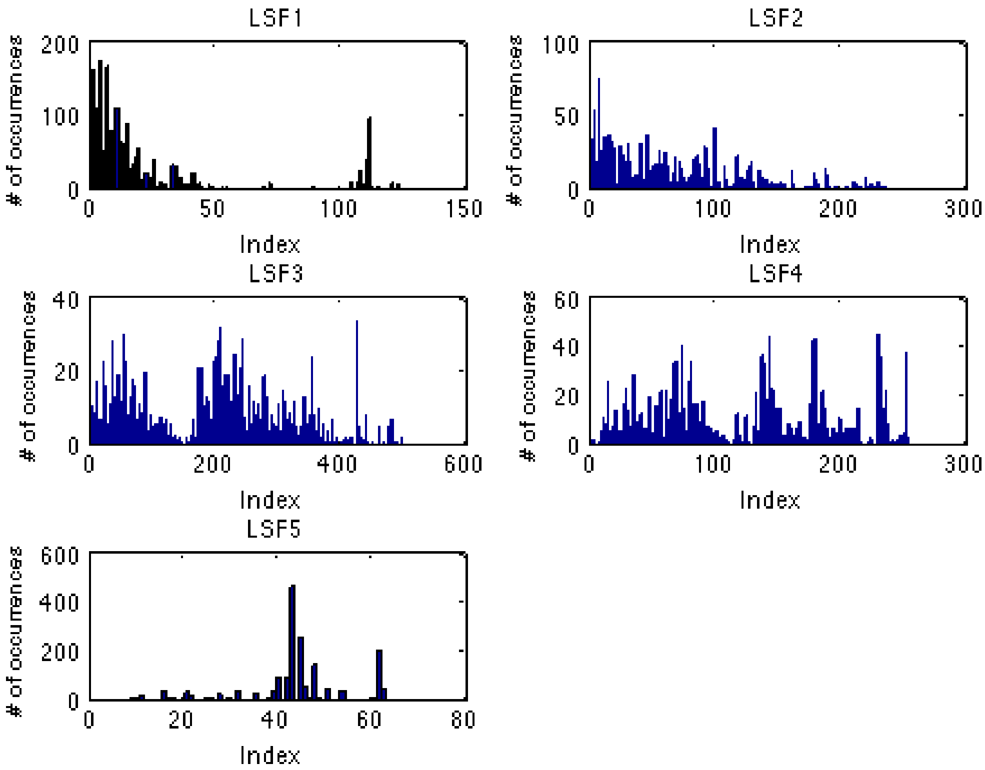

Figure 2.4. *The histogram method. It shows the histograms for the 5 LSF coefficients for sample speech*

which genomes are the best and those are chosen to evolve. This method does not guarantee convergency to the optimal solution. We used the GA toolbox in Matlab to perform the simulations.

We want to choose the entries for the codebook which give the lowest BER. However, it was difficult to represent this problem in terms of GA. We tried to optimize just the LSF and keep the other parameters fixed. The genome is then a codebook of the LSF parameters because the receiver has to have a codebook to be able to receive. The fitness function calculates the BER for each entry in the genome and then averages them. If we try to set the genome to a single LSF parameter then it is not possible to receive and therefore we cannot calculate the BER which we use for the fitness function. Also we optimize just one parameter here but to get the best results, it should be optimized jointly, which in this method would be too big task.

2.3 Communication System Method

In this section we disregard most of the properties of the speech channel. Instead we transmit and receive the digital data by using conventional communiation system methods.

Modulation

The first step is to choose a modulation scheme for this system. This is done by sending signal through the software channel with different modulations and calculate the resulting BER. The best result is from PSK compared to ASK and FSK. ASK changes the amplitude which is not desirable in this case. The speech coder has AGC and it detects the changes in the amplitude and makes compensation, which will cause errors in the receiver. FSK is also not a good choice here because of a very limited bandwidth (4 kHz). Finally DPSK is chosen since it is simpler to implement and does not need a coherent receiver.

Carrier Frequency

The second problem is about the carrier frequency. The signal is transmitted through a voice channel which operates at a certain bandwidth. The channel is band-limited to 4 kHz because the sampling frequency is 8 kHz. In a regular telephone system, there are filters which filter out low and high frequencies so it is best to choose a middle value in this case. In order to determine this value, a sequence of gaussian white noise is transmitted through the voice channel. The noise has a flat spectrum but the spectrum of the resulting signal does not, as can be seen in Figure (2.5). The most flat part seems to be around 1.5 kHz. However, the maximum bandwidth is achieved if the carrier is 1.8 kHz, so we choose this value as our carrier frequency.

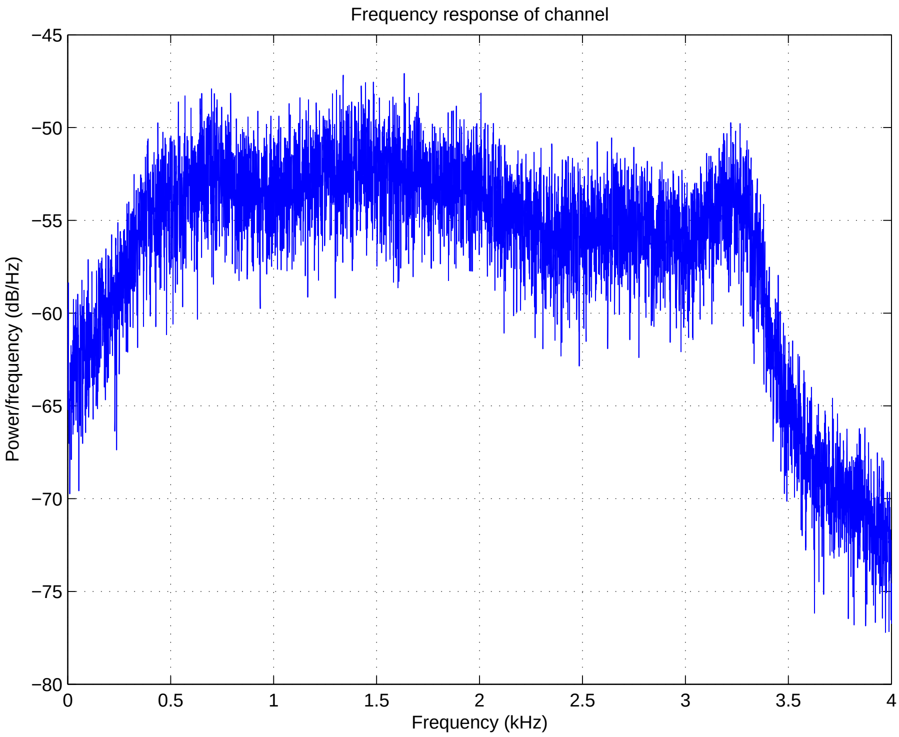

Figure 2.5. *The result of transmitting white gaussian noise through the speech channel.*

Synchronization

The third problem is about good synchronization sequence. There are a number of different synchronization sequences to choose from, e.g. random sequence, Walsh-Hadamard code, Barker code, PN sequence, etc. Barker code and PN sequence have a better autocorrelation property. The System with Waveform Synchronizer is using the Barker 13 code and The System with Bit Synchronizer is using PN sequence. When the data is transmitted through the channel, it is divided into frames. For each frame a synchronization sequence is attached at the beginning. The purpose of doing this resynchronization is because the system is can lose synchronization if there are errors in the synchronization sequence.

2.3.1 System with Waveform Synchronizer

Transmitter

A block diagram of the transmitter is described in Figure (2.6). The first block adds a channel code, here Reed-Solomon code (255,223) is chosen because it is very

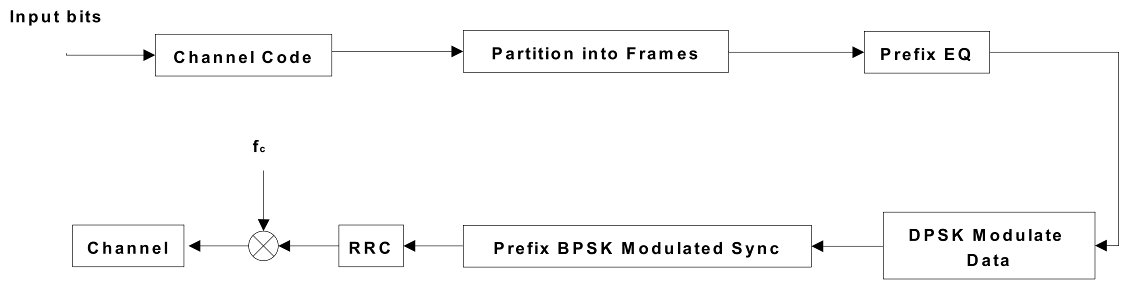

Figure 2.6. *Block diagram for the transmitter of The System with Waveform Synchronizer*

effective to correct the block errors.The next step is to divide the whole sequence into frames of e.g. 1000 bits long. An equalizer training sequence is then prefixed to all the frames. These frames are individually modulated with DPSK. A synchronization sequence with BPSK modulation is added to the sequence later. Here we are using Barker 13 code. After that the whole baseband signal is passed through an RRC filter to create a pulse shape and up-sampled to the desired sampling frequency. A carrier frequency is then multiplied with the the signal to move it from baseband to passband.

As mentioned above, different modulation schemes are used for synchronization sequence and the data. This is because it will be easy for receiver to detect the synchronization sequence.

Receiver

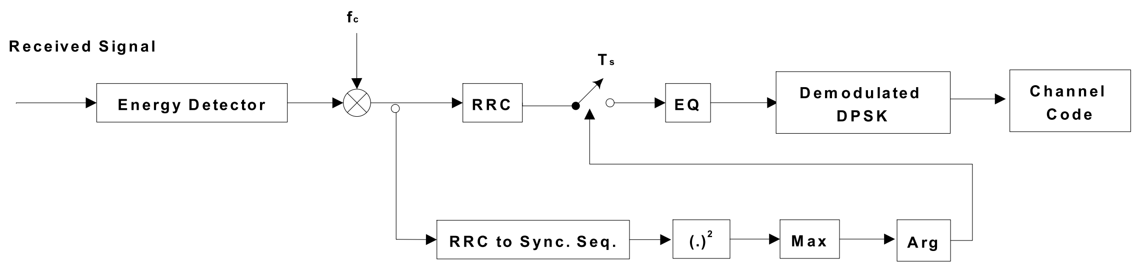

Figure 2.7. *Block diagram for the receiver of The System with Waveform Synchronizer*

A block diagram of the receiver is described in Figure (2.7). The first block is to help us to find an approximate starting point by detecting the energy at the carrier frequency. The next step is to multiply the signal with the same carrier frequence as in the transmitter to move it back to the baseband. Then the sequence is branched into two paths. It is done by using two different matched filters, one is for the pulse shape and the other is for the synchronization sequence to find the beginning point of the data. Then the data is sampled at T_s, the symbol time. After that the whole sequence goes through an equalizer. But first the equalizer should be trained by the equalizer training bits which are put into the beginning of the frame in the transmitter. After it has used all of the training bits, it will switch to a data directed mode [10]. The data is then demodulated with DPSK.

2.3.2 System with Bit Synchronizer

Transmitter

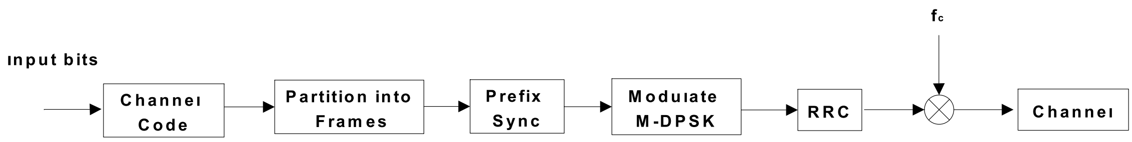

Figure 2.8. *Block diagram for the transmitter of The System with Bit Synchronizer*

A block diagram of the transmitter is described in Figure (2.8). First the bits encoded with channel code and then divided into frames of certain length, e.g. 2550 bits. The same Reed-Solomon channel code is used here as in the previous system. Then a synchronization sequence is prefixed to each frame. Here we are using a PN sequence of length 100. It is necessary to use a longer sequence because the sequence is very susceptible to errors while the Barker code only has a maximum length of 13. After that all the frames are modulated together with M-DPSK. Here the same modulation scheme is applied both in synchronization sequence and data because in the receiver the demodulation is done before detecting the synchronization sequence. Later the whole baseband signal is passes through an RRC filter to create a pulse shape and up-sampled to the desired sampling frequency. A carrier frequency is then multiplied with the the signal to move it from baseband to passband.

Receiver

A block diagram of the receiver is depicted in Figure (2.9). The first block is the energy detector which detects energy at the carrier frequency. It tells approximately where to start processing. Then the received signal is multiplied with the carrier to move it down to a baseband signal. Next an RRC filter, also a MF is used. After that, the signal is down-sampled to the symbol time. This is done by using a timing estimation algorithm to find the optimal sampling points. The algorithm used here is called Square and Filter method which is described in Equation (2.1). This method processes the samples from MF in blocks of length L and assumes that the timing offset is constant in the block we are working in. It also requires that the number of samples per symbol, N, should be 4 or larger.

The optimal sampling point is often located between the samples which we have. So

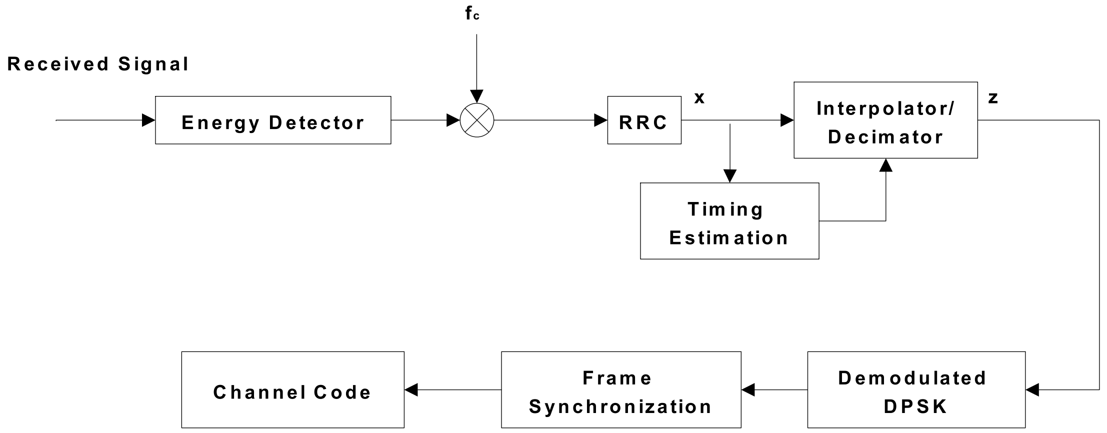

Figure 2.9. *Block diagram for the receiver of The System with Bit Synchronizer*

we need to implement an interpolator to find those samples. Before the parameter Δt is passed to the interpolator, it is processed with Equations (2.2) and (2.3). The first equation is for thresholding, we do not have to interpolate if the time offset is close to 0. The second equation corrects the negative values. When the samples with offset t is found, the output is then sampled at T_s, which is the symbol time. More details can be found in [11].

$$\Delta t = -\frac{1}{2\pi} \arg \left(\sum_{m=0}^{LN-1} |x_{m+1}|^2 \exp(-j2\pi m/N) \right) \tag{2.1}$$

$$\Delta t = \begin{cases} \Delta t & \text{if } |\Delta t| \geq \frac{1}{2N} \\ 0 & \text{if } |\Delta t| < \frac{1}{2N} \end{cases} \tag{2.2}$$

$$\Delta t = \begin{cases} \Delta t & \text{if } \Delta t \geq 0 \\ \Delta t + 1 & \text{if } \Delta t < 0 \end{cases} \tag{2.3}$$

When we have samples at the correct points, we can perform the demodulation. The last step is to synchronize the frames. This is done by using a MF to the synchronization sequence which is used in the transmitter. Resynchronization for each frame is performed in case the synchronization is lost. Finally the channel code is decoded.

3 Results

Our targets for this system is to achieve a bit rate of about 2-3 kbps and BER less than 0.01.

3.1 Synthetic Speech Method

This system was only simulated using the software channel. It did not work very well. First the histogram method gave average BER about 0.3. The GA method did not work at all, because it was not able to optimize the system. It was not easy to represent this problem and GA was not a suitable optimization method in this case.

The problem with this method is the use of the EFR coder. This speech coder has fixed quantization tables which makes it impossible to optimize the codebooks. However, if you have such an speech coder then it could give good results. But such a coder might be necessary to write from scratch and that takes time.

We did not pursue this method any further because we had already got promising result from the other method, so we decided to put more effort into that one.

3.2 Communication System Method

The simulation is first implemented in the EFR software channel which results in bit error rate 0. The next step is to connect this system to the real hardware channel. Here we have more complicated things to take care of. The parameters used for the simulations are described in Table (3.1). The sampling frequency in all of these simulations is never lower than 8 kHz, since the bandwidth of the channel is 4 kHz. However, it is increased to maintain 8 samples per symbols where necessary. The carrier frequency is found from Figure (2.5) as mentioned before. The frame length was chosen to fit the channel code. This Reed-Solomon code was chosen because it can correct the most amount of errors, or 16 symbols in error.

First we need to think of the bandwidth. From Figure (2.5) we see that the maximum usable bandwidth would be 3 kHz. Here we are using RRC pulseshape with $\alpha = 0.3$ and the bandwidth can be calculated form Equation (3.1). The bandwidth used for different bit rates and constellations size is in Table (3.2). As can be seen from the table it is not possible to transmit at 3 kbps with constellation size 2, since it uses too much bandwidth.

$$B = \frac{1+\alpha}{T_s} = \frac{R_b(1+\alpha)}{\log_2(M)} \tag{3.1}$$

Parameters	Value
Sampling frequency	8 and 16 kHz
Carrier frequency	1.8 kHz
Frame length	2550
RRC:	
α	0.3
Delay	4 samples
Block length of:	
Timing Estimation	200
Channel code:	
Reed-Solomon	(255,223)

Table 3.1. *The parameters used in the simulations.*

Rate	Bandwidth [Hz]	
[kbps]	M = 2	M= 4
1000	1300	650
2000	2600	1300
3000	3900	2600

Table 3.2. *The different bandwidth usage for different bit rates and constellation sizes*

Synchronization problem

The main problem with System with Waveform Synchronizer is that the synchronization is not good enough. This will lead to sub-optimal sampling points after the MF. The result is that the system cannot work at high bit rates and only works for bit rates below 1 kbps. So in The System with Bit Synchronizer, we have made a change. Instead of doing both frame synchronization and symbol synchronization before sampling the output of MF, we do the frame synchronization after demodulation. This is a simpler scheme but it requires some other method to do symbol synchronization, which has been discussed previously: The timing estimation which has Equation (2.1) . This method is difficult to add into The System with Waveform Synchronizer so that leads to our design of The System with Bit Synchronizer.

Different sampling frequence

When performing simulations using real hardware, especially when using computer soundcard, it can lead to slightly different sampling frequency at the transmitter

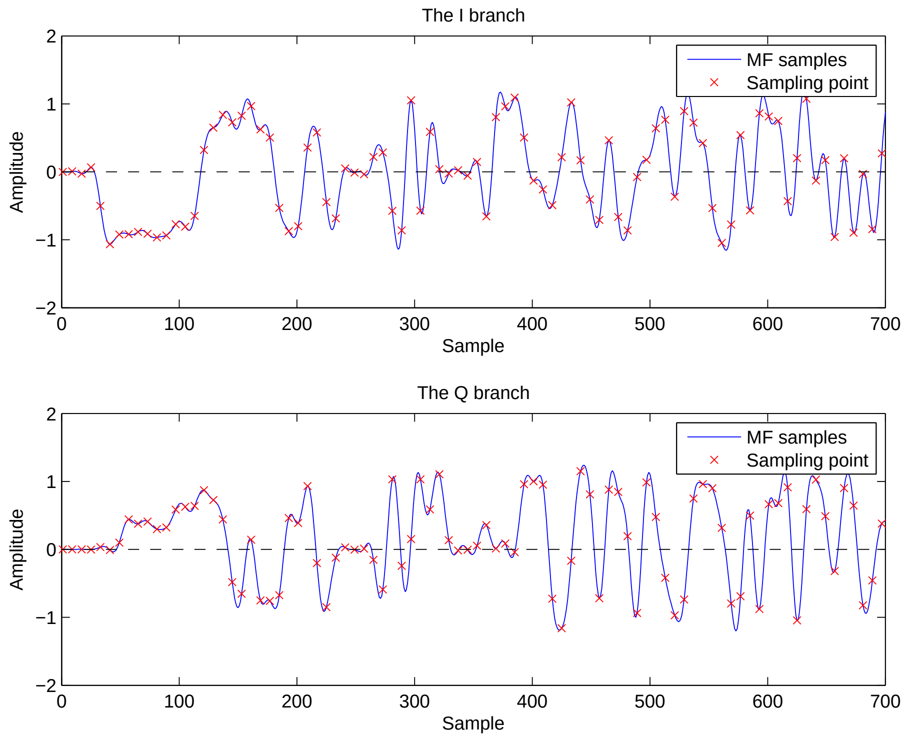

Figure 3.1. *The output of the matched filter and the sampling points from the timing estimation.*

and the receiver. When plotting the samples before modulation, a rotation of the phase can be seen from Figure (3.2). This would make it impossible to use any coherent modulation without phase estimation. However, we have chosen to use DPSK which only relies on the difference of the phase. So if we plot the same plot but now with the phase difference, $z_{n-1}^* \, z_n$, then we don't see this problem, as can be seen in Figure (3.3).

Loss of signal

In many of the simulations a problem of loss of signal is always encountered, as is depicted in Figure (3.4). This could be the result of VAD or a packet loss on the air link. It will lead to block errors and that is the reason why a Reed-Solomon channel code is chosen for this system.

Simulation result

From Figures (3.5) and (3.6) we can see the BER as a function of time. The BER is calculated by cumulate all the previous bits at certain point. As can be seen, the channel code has a good effect on the system.

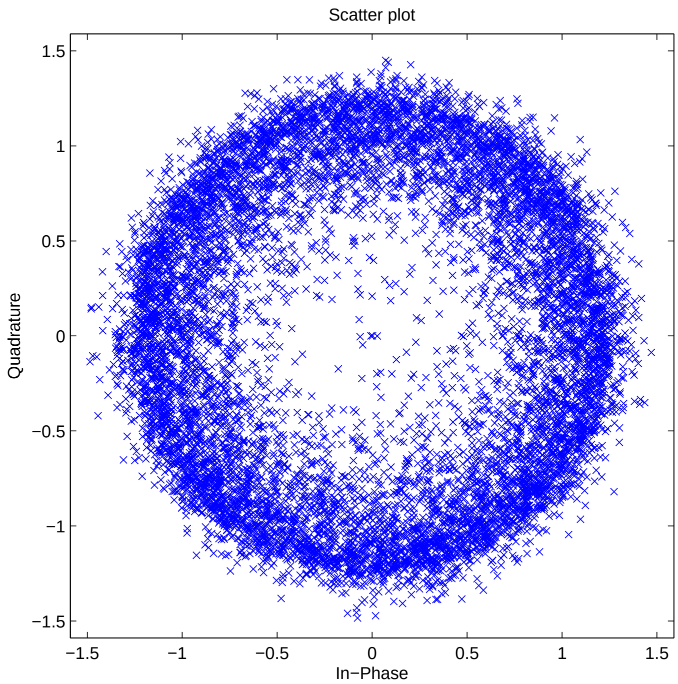

Figure 3.2. *The received constellation. Only two constellation points were used for the transmission.*

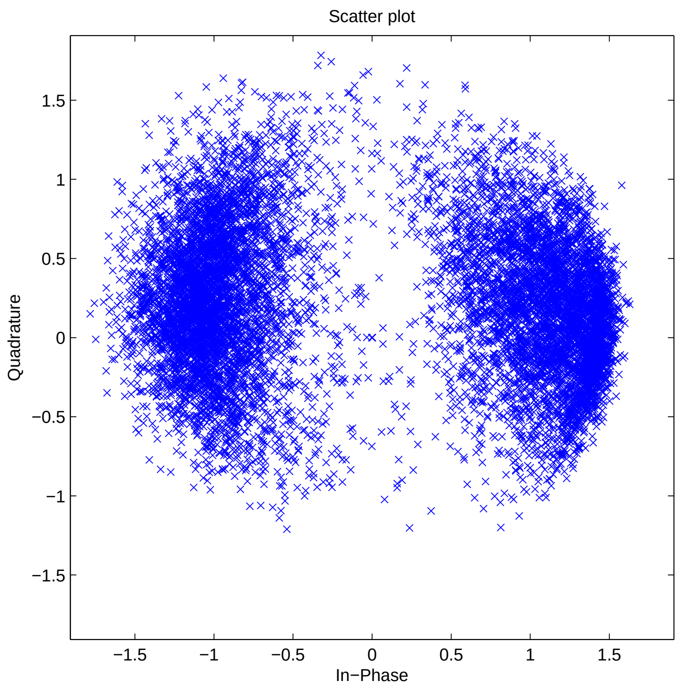

Figure 3.3. *The received constellation after calculating the phase difference between the samples.*

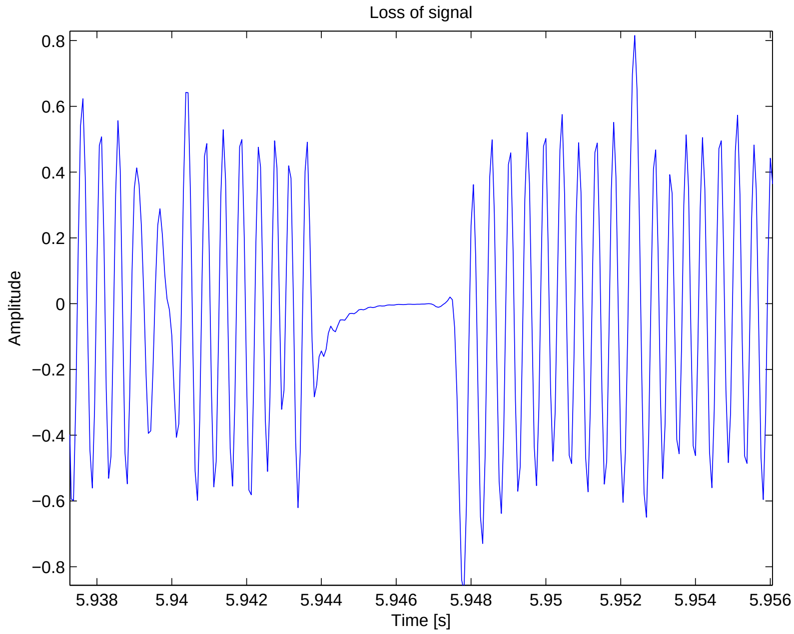

Figure 3.4. *A loss of the received signal.*

In the second figure the BER is calculated in terms of blocks, which has the same length as the transmitted frame. But in each frame the calculation of BER is generally done as the first one. As can be seen, there are peaks in each figure which are caused by the loss of transmission signal. But at date rate 1 kbps, we have otherwise 0 BER.

From the Table (3.3), the best result we have achieved is transmitted at 1 kbps with 4 constellation points. It is not very good compared to [9]. But we have to consider the communication method does not include the special properties of the channel, which to some extend can have big effects on the performance of the system.

Even though the targets that we setup with were not entirely achieved but we are still satisfied with the final results. The bit rate is only slightly lower than expected. And the BER is also close to the target BER.

Rate [kbps]	**Information Rate [kbps]**	**BER**			
		$M = 2$	f_s	$M = 4$	f_s
1	0.84	0.0096	8 kHz	0.0038	8 kHz
2	1.68	0.159	16 kHz	0.076	8 kHz

Table 3.3. *The final bit error rate for System with Bit Synchronizer and different bit rates and constellation size*

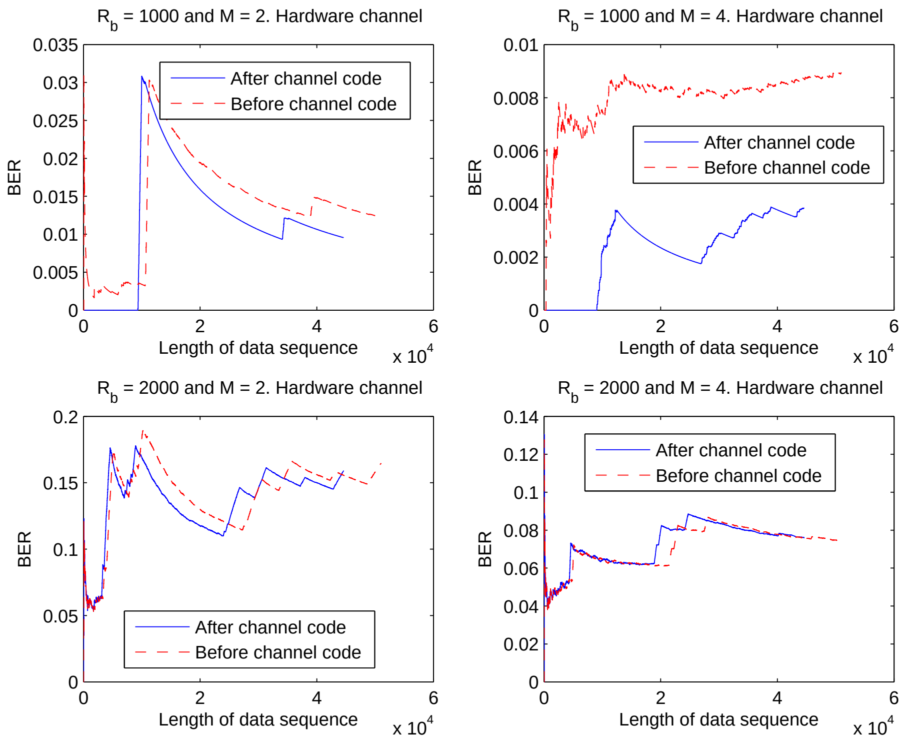

Figure 3.5. *Cumulative BER for different bit rates and constellations sizes for the communication method, System with Bit Synchronizer. This is using the hardware channel.*

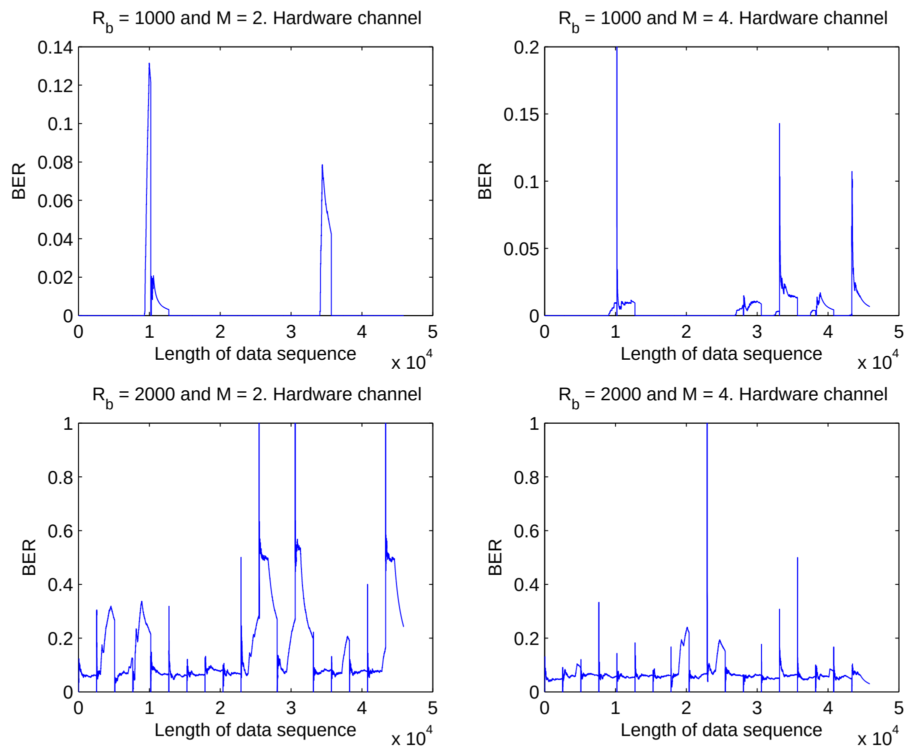

Figure 3.6. *Cumulative BER which is calculated in frame length blocks*

4 Conclusion

The main task of this thesis is to transmit the digital data through the voice compressed channel. We have tried two methods to accomplish this task, the synthetic speech method and the communication method.

The first one does not give very promising results. The choice of EFR codec has limited the possibility of optimization. However, the results in the literature are better with this method.

The second method gives reasonable good results. When the system is evaluated with the software channel, no errors occur. On the other hand, when it is evaluated with hardware channel, the results are very close to the targets that we setup with. But is more challenging and more problems have to be taken into account, like synchronization and difference in sampling frequencies.

5 Future Work

This is a very interesting subject for someone who enjoy the work within speech coding. And of course there are still a lot of work could be done regarding this thesis. For the first method, more investigation could be put to find a better analyzer/synthesizer pair, which will enable easy optimization of codebooks. For the second method, it can be investigated if other modulation methods, such as OFDM, CPM or QAM, can give better result. It could also be evaluated how this method performs under bad radio channel condition.

However, before that is done, a further study of the channel should be performed. This channel is not an easy one and we do not have a model for it. It should be expected that the channel is highly non-linear, so many assumptions that are made for linear channels cannot be made here. One can view the channel as changing with the input, so for example SNR measurements should be done for different input signals.

An important measurement would be to measure the noise added by the channel because it could be different depending on the frequency. If this is known, then this information could be used with OFDM and choose different modulation for different frequency depending on the conditions there.

It is also very important to choose a suitable channel code. A channel code could be chosen from the examination of the errors. For example, if you have a block errors of length 4, then you could choose a Reed-Solomon code which can correct 4 errors. One must note that it is possible that the channel code can make more errors if not chosen carefully.

Bibliography

[1] A.M. Kondoz. *Digital Speech*. Wiley, 2004.

[2] ETSI. Digital cellular telecommunications system (phase 2+); enhanced full rate (efr) speech transcoding; (gsm 06.60 version 8.0.1 release 1999). Technical report, ETSI, 1999.

[3] ETSI. Digital cellular telecommunications system (phase 2+); voice activity detector (vad) for enhance full rate (efr) speech traffic channels; (gsm 06.82 version 8.0.1 release 1999). Technical report, ETSI, 2000.

[4] ETSI. Digital cellular telecommunications system (phase 2+); channel coding (3gpp ts 05.03 version 8.9.0 release 1999). Technical report, ETSI, 2005.

[5] ETSI. Etsi publications download area. `http://pda.etsi.org/pda/queryform.asp`, 2008.

[6] IPdrum. Ipdrum mobile skype cable. `http:\\www.ipdrum.com`, 2008.

[7] N. Katugampala, S. Villette, and A.M. Kondoz. Secure voice over gsm and other low bit rate systems. Technical report, Center for Communication Systems Research, 2003.

[8] Stephane Pierre Villette, Simon N. Fabri, and Ahmet M. Kondoz. Data transmission over a compressed speech channel. Patent Application: WO 03/071521 A1, Feb 2003.

[9] N.N. Katugampala, K.T. Al-Naimi, S. Villette, and A.M. Kondoz. Real time data transmission over gsm voice channel for secure voice and data applications. Technical report, University of Surrey, United Kingdom, August 2005.

[10] John G. Proakis and Masoud Salehi. *Communication System Engineering*, volume Second Edition. Prentice Hall, 2002.

[11] Heinrich Meyr, Marc Moeneclaey, and Stefna A. Fechtel. *Digital Communication Receviers*. John Wiley & Sons, INC, 1998.

A Appendix

A.1 EFR code

First you have to be able to compile the code and that is platform specific. The supported platforms for the code are Windows using Cygwin/Mingw and various UNIX systems, like Linux and Solaris. All simulations are performed on an Intel MacBook but it is not supported in the code. So this requires slight modifications to the file *typedef.h* by copying the settings from Linux to OSX.

```
#elif defined(__unix__) || defined(__unix)
        || defined(__osx__)
```

Then the next step is to create the MEX files which will be compiled with matlab. The files that need to be modified are *coder.c*, *decoder.c* and *ed_iface.c*. New files are then created prefixed with *efr_*. The most important changes are the following

```
#include <mex.h>
#include <string.h>
```

And then the main function has to be replaced by

```
void mexFunction(int nlhs, mxArray *plhs[],
        int nrhs, const mxArray *prhs[])
```

After that all the lines related to reading the data from a file can be deleted. We need to add new code to get the data from Matlab.

```
Word16 *speech_ptr = NULL;
speech_ptr = (Word16 *)mxGetData(prhs[0]);
```

Later when we need to access the data, it can be copied into a defined pointer:

```
memcpy(new_speech, speech_ptr,
                L_FRAME*sizeof(Word16));
speech_ptr += L_FRAME
```

The data can be copied back into Matlab:

```
Word16 *output_ptr = NULL;
memcpy(output_ptr, serial,
                (SERIAL_SIZE - 1)*sizeof(Word16));
output_ptr += SERIAL_SIZE - 1;
```

What we have done is to run *make* to compile all the dependencies. Now those three files, *efr_coder.c*, *efr_decoder.c* and *efr_ed_iface.c*, which previously modified, are compiled within Matlab

```
mex efr_file.c dependencies.o
```

where the files *dependencies.o* can be found in the makefile. The result is a MEX file, which can be run within Matlab. More information about MEX files can be found in the Matlab help.

A.2 Mobile connection

The cable has 3 connectors, one for microphone, one for speaker and one USB connector. It is designed to work with Skype but we are only interested in sending audio to the mobile. Here we have a slight problem: the microphone in the mobile does not turn off unless it is initialized with the USB cable. This software is for Windows only and it is not possible to run it on Mac. Also when we tried to use the cable just with the PC then we got some humming when we recorded, which might have been caused by a ground loop. So instead we connect the microphone to the Mac and the USB cable to the IBM computer, which has the IPdrum software installed.

To make sure that the phone was initialized correctly, we had to take these step when connecting. First we have to make sure the IPdrum software is running on the computer. Then the Sony Ericsson phone is on and the usb connector is connected to the PC. Then we had to wait until the program said that the phone could not be authenticated, then the initialization process was finished. The error message is of no concern here because we are not going to use the cable with Skype.

We also have a problem when connecting the mobile via bluetooth. We have a bluetooth dongle, but it does not allow us to use the voice gateway which we try to use in a couple of computers and none of them allowed us to use the voice gateway. It did not work with the built-in bluetooth in the Mac. The only computer where it works is the IBM ThinkPad with the built-in bluetooth.