Journal of Information Systems and Communication
ISSN: 0976-8742 & E-ISSN: 0976-8750, Volume 3, Issue 1, 2012, pp.-193-198.
Available online at http://www.bioinfo.in/contents.php?id=45

COMPARISON BETWEEN VITERBI ALGORITHM SOFT AND HARD DECISION DECODING

RAHUL VIJAY, SONJUHI MRINALEE AND GARIMA MATHUR

Jaipur Engineering College, Kukas, Jaipur, Rajasthan, India.
*Corresponding Author: Email- garimamathurjec@gmail.com.

Received: January 12, 2012; Accepted: February 15, 2012

Abstract- This paper presents simulation of Viterbi decoder and Fano decoder for decoding the convolutional codes in AWGN channel. Graphs are plotted between Viterbi algorithm and Fano algorithm for decoding the convolutional codes of fixed code rate and fixed constraint length. Results show that performance of Viterbi decoder is better than Fano decoder for the same code rate, constraint length and decoding delay. Convolutional codes are widely used to encode digital data before transmission through noisy or error-prone communication channels to reduce occurrence of errors. This paper presents CC(2,1,3) encoder which belongs to the convolutional encoder's class. It also presents two different decoding techniques based on the Viterbi algorithm: the hard decision and the soft decision encoding techniques. Some simulated results obtained by using these two decoding techniques are presented as well.
Keywords- data processing, signal processing, functional analysis, communication channel, Viterbi, encoder, decoding algorithm, data management.

Citation: Rahul Vijay, Sonjuhi Mrinalee and Garima Mathur (2012) Comparison between Viterbi Algorithm Soft and Hard Decision Decoding. Journal of Information Systems and Communication, ISSN: 0976-8742 & E-ISSN: 0976-8750, Volume 3, Issue 1, pp.-193-198.

Introduction

The task which a designer faces in a digital communication system is of providing a cost-effective facility for transmitting information from one end of the system at a rate and a level of reliability and quality that are acceptable to the user at the other end. Convolutional encoding with Viterbi decoding is a technique that is particularly suited to a channel in which the transmitted signal is corrupted mainly by additive white Gaussian noise (AWGN). Viterbi decoding was developed by Andrew J. Viterbi in 1967[1]. Since then, other researchers have expanded on his work by finding good convolutional codes, exploring the performance limits of the technique, and varying decoder design parameters to optimize the implementation of the technique in hardware and software. The Viterbi decoding algorithm is also used in decoding trellis-coded modulation technique which is used as an effective coding technique for band limited channels as in telephone line modems to squeeze high ratios of bits-per-second in 3kHz- bandwidth analog telephone lines [1,2]. For years, convolutional coding with Viterbi decoding has been the predominant Forward error correction (FEC) technique used in space communications, particularly in geostationary satellite communication networks, such as VSAT (very small aperture terminal)[1]. In the following sections the coder used in this communication will be described as well as the Viterbi decoding technique.

Convolutional codes:

Convolutional codes are well described in the literature [1-9].They are commonly specified by three parameters; (n,k,m), where: n is the number of output bits, k is the number of input bits, and m is the number of shift register stages of the coder. The constraint length L of the code represents the number of bits in the encoder memory that affect the generation of the n output bits and is defined as L = mk. The code rate r of the code is a measure of the code efficiency and is defined by r = k/n.

Code parameters and the structure of the convolutional code:

Figure1 shows the convolutional encoder structure CC(2,1,3)) used in this paper and is built from its parameters. It consists of 3 (m=3) shift register stages and two modulo-2 adders (n= 2) giving the outputs of the encoder. The rate of the code is r = 1/2. The

minimum distance of the code is dmin =3. The outputs of the adders are sampled sequentially yielding the code symbols. The total number p of bit symbols is given by p = n(nb + m) where nb is the total number of bits of information. The outputs V1 and V2 of the adders are governed by the following generator polynomials:
The generator polynomial for the output V_1 is given by
$g_1(x) = 1+ x + x^2$
The generator polynomial for the output V_2 is given by
$g_2(x) = 1+x^2$
$g_1(x)$ and $g_2(x)$ select the shift register stages bits to be added to give the outputs of the encoder which are, for
the case of CC(2,1,3) encoder as follow
$V_1=U_o+U_1+U_2$ (1)
$V_2= U_o+U_2$ (2)
The polynomials give code its unique error protection quality.

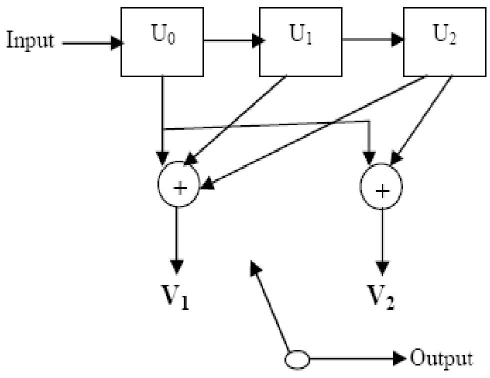


Fig. 1- Convolutional code CC(2, 1, 3)

Coding a message sequence

In this section a 4 bit sequence of 1001is coded to show how the encoder works. The bits are passed through the encoder sequentially as shown in table 1 and the outputs of the encoder are calculated using equations (1) and (2) for each time step.

Table1-Coding process of CC(1,2,3)code

Time instant	Input	U_0	U_1	U_2		V_0	V_1
t= 0		0	0	0		0	0
t=1	**1**	**1**	0	0		1	1
t= 2	**0**	**0**	**1**	0		1	0
t= 3	**0**	**0**	**0**	**1**		1	1
t= 4	**1**	**1**	**0**	**0**		1	1
Reset process	0	0	**1**	**0**		1	0
	0	0	0	**1**		1	1
	0	0	0	0		0	0

The data in bold are the input bits which are shifted in the memory register for every time clock. The code output sequence is: 11 10 11 11 10 11 00 where the last 6 bits (= nm) are the reset bits and they do not contain any information. Table 2 shows the code words of different 4-bit messages calculated by using the procedure described below.

Table2- Code words of different 4-bit messages

Message number	Message	Codeword
1	0000	00 00 00 00 00 00 00
2	0001	00 00 00 11 10 11 00
3	0010	00 00 11 10 11 00 00
4	0011	11 01 01 1100 00 00
5	0100	00 11 10 11 00 00 00
6	0101	11 01 00 01 11 00 00
7	0110	00 11 01 01 10 00 00
8	0111	11 01 10 01 01 00 00
9	1000	00 00 00 11 10 11 00
10	1001	11 10 11 11 10 11 00
11	1010	00 11 10 00 10 11 00
12	1011	11 01 01 00 10 11 00
13	1100	11 01 01 11 00 00 00
14	1101	11 10 00 01 01 11 00
15	1110	11 01 10 01 11 00 00
16	1111	11 01 10 10 01 11 00

State diagram

A state diagram of an encoder can also be used to encode messages. Table 3 gives the output bits for each input bit and the input and output states of the encoder which are the states of U_1 and U_2 shift register stages.

Table3- Input & output states related to input bits

Input bits	Input states		Output bits		Output states	
I	S_1	S_2	V_1	V_2	S_1	S_2
0	0	0	0	0	0	0
1	0	0	1	1	1	0
0	0	1	1	1	0	0
1	0	1	0	0	1	0
0	1	0	1	0	0	1
1	1	0	0	1	1	1
0	1	1	0	1	0	1
1	1	1	1	0	1	1

The encoder outputs V_1 and V_2 are calculated using equations (1) and (2) above. Here the input bit I represents U_0 the states S_1 and S_2 represent U_1 and U_2 respectively. The state diagram of the encoder shown in figure 2 is drawn from table3 and gives a good insight of the encoder operation. Each circle represents a state.

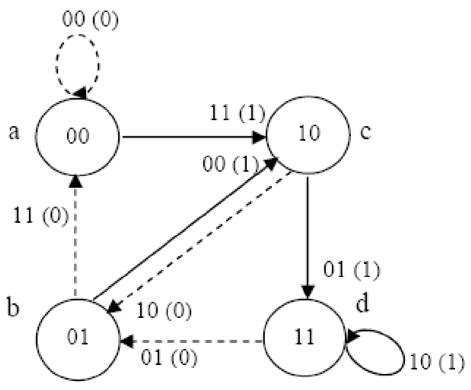
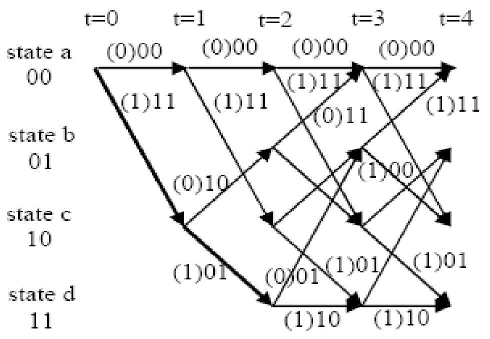


Fig. 2- State diagram of CC(2,1,3) code

The state of the convolutional encoder is represented by the first m-1 shift register stages. At any one time, the encoder resides in one of these states. The lines to and from any state show the possible state transitions as bits arrive. Only two events can happen at each time, arrival of a bit 1or arrival of bit 0. Each of these two events allows the encoder to jump into a different state. The solid lines indicate the arrival of a 1 and the dashed lines indicate the arrival of a 0. The arrows indicate the direction of the transition state. The output of each transition is shown on the arrows and the input bit is in between brackets. This state diagram will be used later in the decoding process.

Trellis diagram

Trellis diagrams represent linear time sequencing events. They can also be used to encode messages. The trellis diagram of CC (2,1,3) encoder shown in figure 3 is drawn from table3. It is built by using the Horizontal axis as a discrete time and all possible $2^{(L-1)}$ states lined up on the vertical axis. One moves through the trellis every time step. New bits arrive every time step. Each state is connected to the next state by the allowable code word for that state. There are only two possible choices at each state. These are determined by the arrival of input bits 0 or 1 which are shown between brackets in the diagram. It can be noticed that the arrows go upwards for a 0 input bit and they go downwards for a 1 input bit. The trellis diagram is unique to each code.



Codeword: 11 01 01 11
Message 1 0 0 1

Fig. 3- Trellis diagram for the code CC(2,13)

For instance to encode 1100 message ,one starts from the state 00 and goes downwards for each input bit 1 and goes upwards for each input bit 0 . After 4 time steps which correspond to the number of input bits, one gets the encoded word 11 01 01 11 (the reset bits are not considered here for simplicity). The trellis diagram of figure3 will be used later to decode code word.

Decoding using likelihood and Viterbi algorithm:

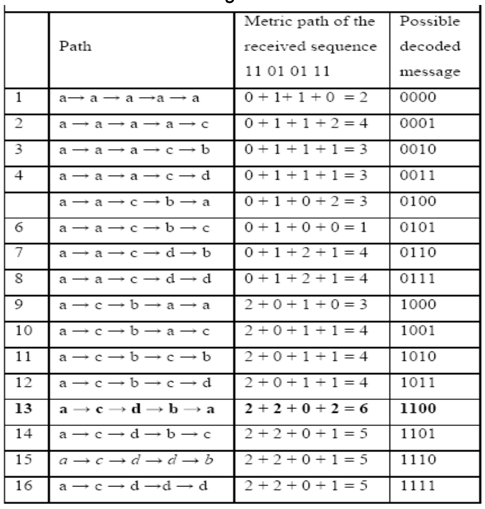
There are several different approaches to decoding of convolutional codes. These are grouped in two basic categories: sequential decoding (Fano algorithm) Maximum likelihood decoding (Viterbi algorithm).Both these methods represent two different approaches to the same basic idea behind decoding [3]. Assume that 4 bits were sent via rate 1/2 code. One receives 8 bits. (and 6 reset bits that are not considered here for simplicity). These 8 bits may or may not have errors one knows from encoding process that these bits map uniquely. So a 4 bit sequence will have a unique 8 bit output. But due to errors one can receive any and all possible combinations of the 8 bits, (28 combinations).The permutation of 4 input bits results in 16 possible input sequences. Each of these has a unique mapping to an 8 bit output sequence by the code as shown in table2.These form the set of permissible sequences and the encoder's task is to determine which one was sent. Let us say one received 11111100. It is not one of the 16 possible sequences shown in table2. How does one decode it? one can do two things 1. compare this received sequence to all permissible sequences and pick the one with the smallest Hamming distance (or bit disagreement) 2. perform a correlation and pick a sequence with the best correlation. The first procedure is basically what is behind hard decision decoding and the second the soft-decision decoding [3].In the following sections these two techniques are described.3.1 Maximum Likelihood and Viterbi decoding Viterbi decoding is the best known implementation of the maximum likelihood decoding [3]. The assumptions made in this technique are as follow:

1. The errors occur infrequently, the probability of error is small.
2. The errors are distributed randomly.

The Viterbi algorithm examines an entire received sequence of a given length. The decoder computes a metric for each path and makes a decision based on this metric. All paths are followed until two paths converge on one node. Then the path with the higher metric is kept and the one with the lower metric is discarded. The selected paths are called the survivors. For an N bit sequence, the total number of possible received sequences is 2^N; only $2^{k(L-1)}$ of these are valid. The Viterbi algorithm applies the maximum likelihood principles to limit the comparison to $2^{k(L-1)}$ surviving paths instead of checking all paths. The most common metric used in hamming distance metric. This is just the received codeword& the allowable code word.

The procedure used in veterbi algorithm is as follows in table

Table 4- Path of trellis diagram & their related message



	Path	Metric path of the received sequence 11 01 01 11	Possible decoded message
1	a→ a → a →a → a	0 + 1+ 1 + 0 = 2	0000
2	a → a → a → a → c	0 + 1 + 1 + 2 = 4	0001
3	a → a → a → c → b	0 + 1 + 1 + 1 = 3	0010
4	a → a → a → c → d	0 + 1 + 1 + 1 = 3	0011
	a → a → c → b → a	0 + 1 + 0 + 2 = 3	0100
6	a → a → c → b → c	0 + 1 + 0 + 0 = 1	0101
7	a → a → c → d → b	0 + 1 + 2 + 1 = 4	0110
8	a → a → c → d → d	0 + 1 + 2 + 1 = 4	0111
9	a → c → b → a → a	2 + 0 + 1 + 0 = 3	1000
10	a → c → b → a → c	2 + 0 + 1 + 1 = 4	1001
11	a → c → b → c → b	2 + 0 + 1 + 1 = 4	1010
12	a → c → b → c → d	2 + 0 + 1 + 1 = 4	1011
13	**a → c → d → b → a**	**2 + 2 + 0 + 2 = 6**	**1100**
14	a → c → d → b → c	2 + 2 + 0 + 1 = 5	1101
15	*a → c → d → d → b*	2 + 2 + 0 + 1 = 5	1110
16	a → c → d →d → d	2 + 2 + 0 + 1 = 5	1111

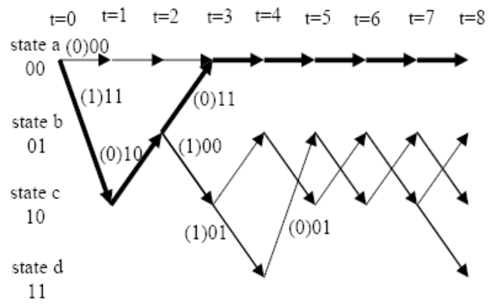
A correct decoding of a corrupted sequence with two errors

The precedent technique demonstrates how the Viterbi algorithm decodes a codeword into its message. In this section, the notion of channel error is considered through an example. Suppose that a code word 00 00 00 00 (00 00 00) of 0000 message is transmitted and at the receiver the sequence 01 00 0100 (00 00 00 00) is received where the digit '1' of the second and sixth symbols are the errors introduced by the transmitting channel. Using the trellis diagram of figure3 and the steps of decoding described above, at the instant t =4, the four survivors shown in table5 are obtained. The path aLa L a La L a having the highest metric (= 8) is the winner. The decoder chooses this path which gives 0000 as a decoded message which is the correct result.

Table 5- Supervivor diagram of a trellis diagram for the corrupted sequence

	path	Metric path	Possible decoded message
1	a→ a → a →a → a	2+2+2+2=8	0000
2	a→ a → a →a → d	2+2+2+0=6	0001
3	a→ a → a →c → b	2+2+0+1=5	0010
4	a→ a → a →c → d	2+2+0+1=5	0011

An incorrect decoding of a corrupted sequence with 3errors*:*
When the number of errors is greater than the code correction capacity, the Viterbi decoder gives incorrect results. Suppose that a transmitted sequence of zeros are corrupted by 3 errors (3 logic '1's); 10010010(000000). Figure4 shows the trellis diagram of the example. The path shown in thick line is the winner and has the highest metric path. It gives a sequence 1000 as a decoded message. The first bit of the message is decoded as '1' instead of '0'. Henceforth the decoding is incorrect.



Codeword: 11 10 00 00 00 00 00 00
Message 1 0 0 0 0 0 0 0

Fig. 4- Trellis diagram of incorrect hard decision decoding

Remark
The results of the last two examples are related to a decision based on the total metric path of the trellis diagram. The path having the highest metric path gives the decoded message. Concerning the hard decision Viterbi decoding, it is observed that the hamming distance between the trellis paths is of importance when it comes to the code error capacity. When the number of errors is less or equal to the minimum distance dmin of the code then the decoder gives a correct answer. However when the number of errors is greater than dmin, the decoder gives an incorrect answer.

Soft Decision Decoding:
In the case of the soft decision technique, the variations of the signal at the output of the demodulator are sampled and quantified. For a Gaussian channel with additive white noise (AWGN), the hard quantification of the received signal, compared with the fine quantification gives losses of about 2dB in S/N ratio [4]. Moreover 8-level quantification reduces this loss to a value of less than 0.25dB, compared to the fine quantification. This means that the 8-level quantification is adequate to this kind of decoding. Figure.5 shows the order of the sampled output of the demodulator with 8 decision thresholds and 8 levels of quantification. The level of transmitted signal is unity. The values {1, 2,3,4} represents the confidence of the received signal where the highest signal represents the number with the highest confidence. For instance, level ± 4 gives a high confidence that the bit is ± 1 logic, whereas ±2 gives less confidence.
Qualification Level -4 -3 -2 -1 1 2 3 4
Decision Threshold

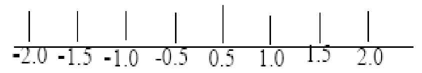


Fig. 5- Level Qualification

Figure.6 illustrates the Viterbi soft decision technique applied to the example used in section (3.1.2) where it is supposed that a sequence of zeros is transmitted and a corrupted codeword is received. In this technique the demodulator passes the quantified level sequences to the decoder. In the absence of noise the received signal is quantified at high negative levels. However if the quantified signal is positive, it means that error has occurred during transmission.

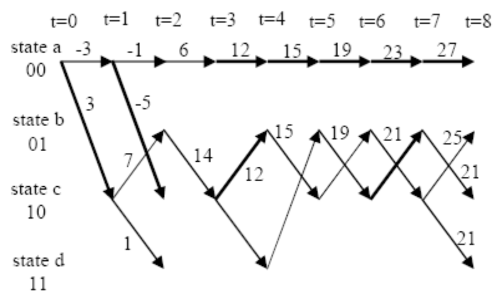


Fig. 6- Soft decision decoding Example

In this case the channel has caused a high voltage with opposite polarity to the transmitted signal. In Figure6, it is noticed that the first three positives quantified levels 2, 1and 1 are the error bits. The first received symbol has confidence levels 2 and 1 and is compared with the symbol with two possible transition outputs 00 and 11 in the trellis. It is noticed that the levels (2, 1) give a measure of confidence of the transition 11. The comparison of the transition outputs starting from the state (00) gives a metric branch = -2+(-1) = -3 for the branch a L a and a metric branch = 2+1 =3 for the branch a L c. At t=2 the accumulated metric paths -1, - 5, 7 and 1 for a L a L a, a L a L c, a L c L b ,and a L c L d respectively. The process of calculating the metric branches is used until t=8, where only 4 survivors are left i.e., a; a ; a ;a ; a; a ; a ;a ; a,a; c ; b ;c ; b; c ; b ;c ; b, a; c ; b ;c ; d; b ; c ;b ; c, a; c ; b ;c; b; c;b;c; d, with 27,21,25, and 21 respectively. Hence the first path with the highest metric is the winner and gives 00000000 as a decoded message, which is correct. This example shows that the soft decision technique gives correct decoding where the hard decision technique gives incorrect decoding.

Simulated results

In this section the results obtained by simulating the hard and soft decision techniques described above are given. Three procedures have been developed in C language to simulate the coding and decoding processes studied in this paper.

Hard decision decoding

In this method it is assumed that, before it reaches the decoder, the coded sequence is demodulated, sampled and then quantified. The quantification is done in two levels (0: absence of a signal, 1: presence of a signal) which means hard decision. The simulation results for this method are tabulated and shown in tables 6,7, and 8. Table6 shows the decoding results of some uncorrupted received sequence. As can be expected the decoder gives correct message.

Table6- Decoded Uncorrected sequence Hard decision algorithm

Received sequence	Decoded message
11 01 10 10 01 11 00	1111 (000)
11 01 01 11 00 00 00	1100 (000)
11 10 11 11 10 11 00	1001 (000)
00 11 10 11 00 00 00	0100 (000)
00 00 00 11 10 11 00	0001 (000)
11 01 10 01 11 00 00	1110 (000)
00 00 11 10 11 00 00	0010 (000)
00 00 00 00 00 00 00	0000 (000)

In table7 corrupted sequences with one or two errors are considered. Here again the decoder decodes correctly the sequences. If the received sequence is corrupted with one or two errors, the decoder detects these errors and gives a correct message of the transmitted codeword. These errors are called detectable because they are less or equal to correction capacity of the code CC(2,1,3) which is equal to 2.

Table7- Decoded corrected sequence-hard decision Algorithm (with 1 or 2 error)

Transmitted sequence	Received sequence	Nb. of errors	Decoded message
11 10 11 11 10 11 00	11 10 11 11 10 11 01	1	1001 (000)
	11 10 01 11 10 11 00		1001 (000)
	10 10 11 11 10 11 00		1001 (000)
00 00 00 00 00 00 00	01 00 00 00 00 00 00	1	0000 (000)
	00 00 01 00 00 00 00		0000 (000)
	00 00 00 00 10 00		0000 (000)
11 10 11 11 10 11 00	11 10 11 11 10 11 11	2	1001 (000)
	11 10 01 11 10 10 00		1001 (000)
	11 00 11 10 10 11 00		1001 (000)
00 00 00 00 00 00 00	01 01 00 00 00 00 00	2	0000 (000)
	01 00 00 00 00 00 01		0000 (000)
	00 00 00 00 00 00 11		0000 (000)

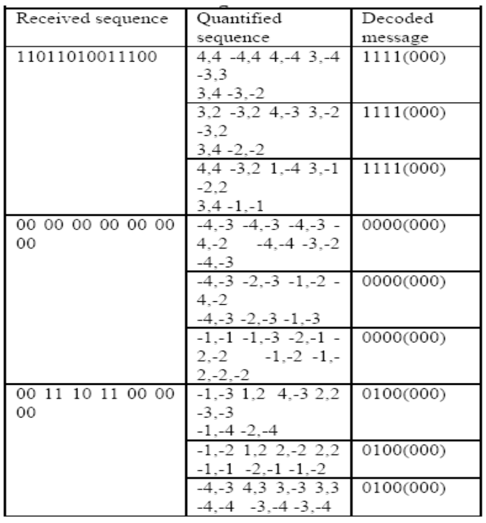
Table8 shows the results obtained for corrupted sequences with three errors. In the case that the number of errors in the sequence is greater than the correction capacity of the code, the decoder gives incorrect results. For instance in the case of the sequence of zeros corrupted with 3 errors 11 10 00 00 00 00 00, the distance between the transmitted and the received sequences, d =3 and dim = 2, the decoder gives the message 1000 instead of 0000. The sequence 1000 was chosen by the decoder because it has the highest metric distance of all the other paths. In the case of the sequence 11 00 00 00 00 00 10, d =3, dmin = 2, the decoder gives a correct message because the third error occurred in the reset bits, and as if the corrupted sequence has dmin = d. The same thing can be said for received sequences having 4 errors or more. Generally the decoder gives incorrect messages.

Table 8-Decoded corrupted sequences-hard decision algorithm (with 3 or 4 error)

Transmitted sequence	Received sequence	Nb. of errors	Decoded message
00 00 00 00 00 00 00	11 10 00 00 00 00 00	3	1000 (000)
	11 00 00 00 00 00 01		0000 (000)
	00 00 01 00 00 00 11		0000 (001)
	00 00 00 00 10 10 10		0000 (000)
11 10 00 10 11 00 00	00 11 01 10 11 00 00	3	0010 (000)
	11 01 00 10 11 00 00		1111(011)
	11 10 00 10 10 01 01		1010(110)
	11 10 01 10 10 01 00		1010(000)
11 01 10 10 01 00	11 11 11 10 01 11 10	3	1111 (000)
	11 01 11 10 00 10 00		1100(000)
	11 10 11 10 01 11 00		1001(101)
	11 01 10 10 00 10 01		1111(110)
00 00 00 00 00 00 00	01 01 01 00 01 00 00	4	1101(101)
	00 01 01 01 00 00 10		0110(111)
	11 10 00 00 00 00 01		1010(110)
	00 00 00 00 10 11 10		0000(010)
11 10 00 10 11 00 00	11 01 00 10 11 00 11	4	1010(001)
	11 11 11 11 11 00 00		1110(000)
	11 10 00 10 10 01 11		1011(100)

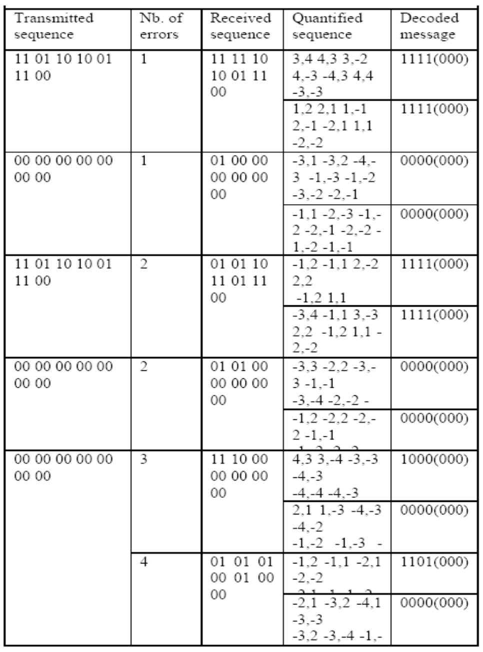
Soft decision technique: In this section an 8-level quantifier is used for the examples studied here. The simulated results are shown in tables 9 and 10. In table9 one can notice that the uncorrupted received sequences are decoded correctly with any value of the quantification level (negative for 0 and positive for 1). From the results of table10, it can be noticed that the soft decision technique can detect any number of errors which is less than the correction capacity of the code independently of the quantification level. If the number of errors is greater than the correction capacity of the decoder (=2), the soft decision technique can decode some sequences correctly (table10). Soft decision decoding technique decodes correctly any sequence having, in one hand, low confidence for the error bits, and in the other hand high confidence for the non-corrupted bits. If the errors bits have high confidence, the message is incorrectly decoded. If the errors occur in the reset bits, the decoder gives a correct message for the received sequence, independently of the confidence of the other bits of that sequence.

Table 9-Uncorrupted received sequence-Soft decision algorithm



Received sequence	Quantified sequence	Decoded message
11011010011100	4,4 -4,4 4,-4 3,-4 -3,3 3,4 -3,-2	1111(000)
	3,2 -3,2 4,-3 3,-2 -3,2 3,4 -2,-2	1111(000)
	4,4 -3,2 1,-4 3,-1 -2,2 3,4 -1,-1	1111(000)
00 00 00 00 00 00 00	-4,-3 -4,-3 -4,-3 -4,-2 -4,-4 -3,-2 -4,-3	0000(000)
	-4,-3 -2,-3 -1,-2 -4,-2 -4,-3 -2,-3 -1,-3	0000(000)
	-1,-1 -1,-3 -2,-1 -2,-2 -1,-2 -1,-2,-2,-2	0000(000)
00 11 10 11 00 00 00	-1,-3 1,2 4,-3 2,2 -3,-3 -1,-4 -2,-4	0100(000)
	-1,-2 1,2 2,-2 2,2 -1,-1 -2,-1 -1,-2	0100(000)
	-4,-3 4,3 3,-3 3,3 -4,-4 -3,-4 -3,-4	0100(000)

Table 10-Decoded corrupted sequences-Soft decision algorithm



Transmitted sequence	Nb. of errors	Received sequence	Quantified sequence	Decoded message
11 01 10 10 01 11 00	1	11 11 10 10 01 11 00	3,4 4,3 3,-2 4,-3 -4,3 4,4 -3,-3	1111(000)
			1,2 2,1 1,-1 2,-1 -2,1 1,1 -2,-2	1111(000)
00 00 00 00 00 00 00	1	01 00 00 00 00 00 00	-3,1 -3,2 -4,-3 -1,-3 -1,-2 -3,-2 -2,-1	0000(000)
			-1,1 -2,-3 -1,-2 -2,-1 -2,-2 -1,-2 -1,-1	0000(000)
11 01 10 10 01 11 00	2	01 01 10 11 01 11 00	-1,2 -1,1 2,-2 2,2 -1,2 1,1	1111(000)
			-3,4 -1,1 3,-3 2,2 -1,2 1,1 -2,-2	1111(000)
00 00 00 00 00 00 00	2	01 01 00 00 00 00 00	-3,3 -2,2 -3,-3 -1,-1 -3,-4 -2,-2 -	0000(000)
			-1,2 -2,2 -2,-2 -1,-1	0000(000)
00 00 00 00 00 00 00	3	11 10 00 00 00 00 00	4,3 3,-4 -3,-3 -4,-3 -4,-4 -4,-3	1000(000)
			2,1 1,-3 -4,-3 -4,-2 -1,-2 -1,-3 -	0000(000)
	4	01 01 01 00 01 00 00	-1,2 -1,1 -2,1 -2,-2	1101(000)
			-2,1 -3,2 -4,1 -3,-3 -3,2 -3,-4 -1,-	0000(000)

Conclusion

The numerical results show that the performance of the system increases as the decoding delay increases in case of Viterbi decoding. For Viterbi decoding when delay D=6K required SNR is 3.9 dB, for D=5K required SNR is 4.1dB. For D=5K, Rate=1/2 K=3 required SNR is 9 dB in case of Fano decoding and 4.1dB in case of Viterbi decoding. The gain in SNR for the BER of 10^{-3} is 4.1 dB for D=6K, Rate=1/2 and K=3.The overall performance of the Viterbi decoding algorithm is better than that Fano algorithm for same decoding delay.The paper has presented hard and soft decision decoding techniques. A hard decision technique can detect any number of errors which are less than or equal to the correction capacity of the code. However, for three or more errors (in the case of CC(2,1,3) encoder) , the decoded sequence is generally incorrect. The soft decision technique decodes correctly any corrupted sequence with one or 2 errors independently of the quantification levels attributed to the symbols of a given received sequence. For three or more errors, this technique detects these errors if they have a low quantification levels and the non-corrupted bits have a high confidence. The noise corrupting the transmitted signal is generally low compared with the signal and cannot reach the signal level. Therefore the confidences of the error bits are generally low and the soft decision technique can detect them. As a conclusion, it can be said that the soft decision technique is powerful compared with the hard decision technique and is suitable for AWGN channels.

References

[1] Anderson J.B., Aulin T. and Sundberg C.E. (1986) *Digital Phase Modulation*.

[2] Rimoldi B. (1988) *A decomposed approach to CPM, IEEE Trans. Inform. Theory*, 34, 260-270.

[3] Aulin T. and Sundberg C.E.W. *Continuous Phase Modulation -Part I: Full response signaling, IEEE Trans. Commun*., 29, 196-209.

[4] Aulin T., Rydbeck N. and Sundberg C.E.W. (1981) *Continuous Phase Modulation-Part II: Partial response signaling, IEEE Trans. Commun*., 29, 210- 225.

[5] Massey J.L. (1984) *Int. Zurich Seminar*, F11(67)- F17(73).

[6] Rimoldi B. (1988) *Continuous phase modulation and coding from bandwidth and energy efficiency, PhD dissertation, Swiss Fed. Inst. Technol*.

[7] Rimoldi B. (1989) *IEEE Trans.Commun*., 37, 897-905.

[8] Quinn Li (1996) *On bandwidth and energy efficient digital modulation schemes, PhD dissertation, Sever Institute, University of Washington*.

[9] Pizzi S.V. and Wilson S.G. *Convolutional coding combined with Continuous Phase Modulation, IEEE Trans. Commun*., 33, 20-29.