UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN
INGEGNERIA DELL'INFORMAZIONE

Progettazione di un sistema di trasduzione e trasmissione sonora per segnali digitali

Relatore:
LEONARDO BADIA

Laureando:
MADDALENA BOSCARO
1216511

Anno Accademico 2021/2022
Data di Laurea 23 Settembre 2022

Abstract

Questa tesi si propone di analizzare e testare l'efficacia e la robustezza di un sistema di comunicazione il cui funzionamento si basa sulla conversione di stringhe binarie in suoni. La trasmissione sonora viene però inevitabilmente disturbata dal rumore proveniente dall'ambiente circostante che introduce degli errori nel messaggio inviato. Per garantire una comunicazione quanto più affidabile si è deciso di sfruttare il codice di Reed-Solomon: una codifica di canale dall'elevato potere correttivo. Di seguito vengono proposti tre diversi modelli di trasduzione sonora, confrontati attraverso simulazioni Matlab, e valutati sulla base della percentuale di errori commessi e corretti al variare dell'intensità e del tipo di rumore ambientale scelti.

Indice

Capitolo 1

Introduzione

Il progetto, sviluppato in questa tesi, nasce dalla collaborazione con una start-up locale per il progetto TIScode che prevede di veicolare informazione tramite codici sonori. L'idea alla base di questo lavoro infatti, si propone di fornire un metodo differente di comunicazione basato sulla trasmissione, tramite altoparlanti, di un messaggio sonoro costituito di sole note musicali. Al ricevitore un microfono registrerà i suoni percepiti che tenterà di riconvertire nei bit che erano stati inviati in principio. Il sistema è inoltre pensato per essere utilizzato in ambienti urbani: è quindi auspicabile che i suoni emessi rimangano abbastanza contenuti, al fine di evitare eccessivi disturbi alle attività umane. D'altro canto questa restrizione risulta vincolante in situazioni di forte rumore ambientale che rischia di compromettere la trasmissione dati. Ispirandosi ai già esistenti metodi di comunicazione acustica sottomarina [21] [17], si è scelto di adottare codici a correzione d'errore dotati di un'alta capacità correttiva. Si sta parlando dei codici di Reed-Solomon [1] che trovano applicazione in diversi campi, in particolare vengono utilizzati da diversi codici bidimensionali come il Data Matrix [14], uno dei metodi di etichettatura e tracciamento industriale più in uso al giorno d'oggi. Come accade per quest'ultimi, anche il nostro messaggio sonoro, sebbene danneggiato e alterato dall'inevitabile rumore ambientale, può essere "ricostruito" attraverso l'aggiunta di ridondanza applicata dalla codifica di canale.

La modulazione del segnale digitale, come anticipato, avviene attraverso la generazione di onde sonore costituite delle sole frequenze appartenenti alle note musicali. Ciò avviene associando univocamente un gruppo di note, un suono specifico, a una determinata sequenza di bit. L'insieme di suoni risultante costituisce il nostro alfabeto, in particolare all'interno di questa trattazione verranno presentati tre differenti esempi di mapping sonoro.

L'analisi delle prestazioni saranno eseguite all'interno dell'ambiente Matlab, attraverso numerose simulazioni, confrontando il comportamento del sistema a seconda dell'alfabeto utilizzato. Le valutazioni finali permetteranno di comprendere quale sia la migliore soluzione, sulla base della percentuale di bit errati e corretti, in rapporto al tipo e al valore di SNR scelti per il rumore ambientale.

I successivi capitoli di questa tesi sono così strutturati. Nel Capitolo 2 vengono esposti studi e progetti affini all'idea qui sviluppata; nel Capitolo 3 viene descritto il modello di trasmissione completo con particolare attenzione alla fase di modulazione e alla creazione dei tre diversi alfabeti; nel Capitolo 4 sono presentati i risultati ottenuti dalle simulazioni con il confronto tra le diverse strategie e il valore di bitrate risultante. A concludere, il Capitolo 5 presenta le valutazioni finali e discute possibili sviluppi e prospettive future di questa tesi.

Capitolo 2

Stato dell'arte

2.1 Codici di Reed-Solomon

I codici di Reed-Solomon devono il proprio nome a Irving Reed e Gus Solomon che li introdussero per la prima volta nel 1960 in un articolo del *Journal of Society for Industrial and applied Mathematics* [1]. Come vedremo in seguito si tratta di una classe di codici a correzione d'errore che trova largo uso in svariate applicazioni.

Sono codici ciclici non binari del tipo R-S (n,k) composti di simboli di $m \geq 2$ bit ciascuno, tali per cui:

$$0 < k < n < 2m + 2 \tag{2.1}$$

k rappresenta il numero di simboli del messaggio originale, n invece il numero totale di simboli del messaggio codificato. Generalmente per i codici R-S (n,k), i parametri n e k vengono scelti secondo la seguente convenzione:

$$(n, k) = (2m - 1, 2m - 1 - 2t) \tag{2.2}$$

$n - k$ è il numero di bit di parità del codice e, come fatto per i codici binari con la distanza di Hamming, si definisce la distanza minima in questo modo:

$$d_{min} = n - k + 1 \tag{2.3}$$

tale valore rappresenta il numero minimo di simboli per cui differiscono due parole di codice. Dalla distanza è inoltre possibile ricavare il numero di simboli che possono essere corretti dai codici R-S:

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor = \left\lfloor \frac{n - k}{2} \right\rfloor \tag{2.4}$$

Chiamiamo t capacità correttiva del codice. Quest'ultima è direttamente proporzionale alla ridondanza $n - k$ come mostrato in Figura 2.1:

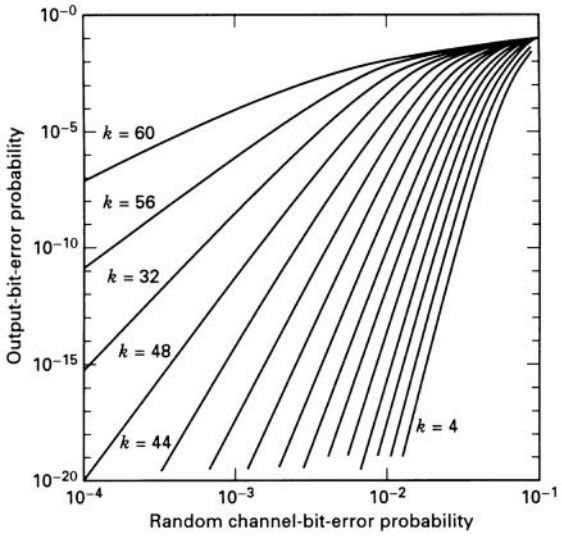


Figura 2.1: Probabilità di errore sul bit di R-S (64, k) in funzione della ridondanza [2]

La lunghezza della parola di codice è fissata a $n = 64$, la lunghezza k del dato pre-codifica, invece, varia da $k = 4$ a $k = 60$ simboli. Così facendo la ridondanza $n - k$ decresce da un massimo di 60 a un minimo di 4. Il vantaggio di aumentare la ridondanza, in Figura 2.1 per k piccoli, si traduce in una probabilità di errore sul bit minore e quindi in una decodifica migliore.

Tuttavia, una ridondanza elevata necessita non solo di una maggiore complessità per implementazioni high-speed ma anche di una banda maggiore per comunicazioni in tempo reale [2]. In generale sono quindi preferibili codici con un rate $R_b = \frac{k}{n} R_c$ più elevato a discapito di una ridondanza e di un potere correttivo minori.

Nel caso più semplice di codice R-S (7,3) è quindi possibile correggere fino a $t = 2$ simboli errati. Ciò significa correggere 2 bit, nel caso in cui i simboli errati presentino un solo bit d'errore, fino ad un massimo di 2·m bit nel caso in cui invece i simboli siano completamente errati. Si può quindi desumere che codici di questo tipo risultino particolarmente prestazionali in situazioni in cui gli errori si presentino come un'unica sequenza sbagliata di bit (*burst error*) [8].

Quando gli errori sono vicini tra loro compromettono infatti solo pochi simboli (Figura 2.2).

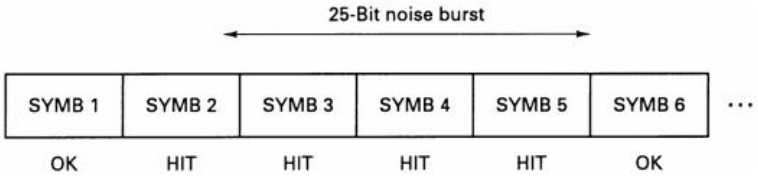


Figura 2.2: Blocco corrotto da un burst error di 25 bit [2]

2.2 Codice Data Matrix

Il codice Data Matrix è un tipo di codice bidimensionale sviluppato dall'azienda americana International Data Matrix. Ciò che differenzia i codici 2D da quelli 1D, come il codice a barre, è innanzitutto la quantità di informazioni memorizzate: se gli unidimensionali raggiungono al massimo qualche decina di cifre, i bidimensionali sono invece in grado di contenere interi messaggi, link, dati e immagini. Un'altra peculiarità dei codici 2D è la tolleranza agli errori, che consente loro di decodificare dati danneggiati anche fino al 30% [11].

Rispetto ai ben più conosciuti codici QR, i Data Matrix si distinguono per le loro dimensioni ridotte, essendo in grado di memorizzare la stessa quantità di informazione in un'area minore. I Data Matrix vengono quindi per lo più utilizzati per marcare piccoli oggetti e trovano largo uso nell'ingegneria industriale: dal settore automobilistico a quello aerospaziale, dal farmaceutico fino all'elettronico [12]. In particolare, tali codici, vengono impiegati nel processo di tracciabilità dei componenti industriali e nell'etichettatura di prodotti farmaceutici e di piccoli componenti elettronici [15]: il codice può essere infatti ridotto fino a occupare un'area di soli 300 μm^2.

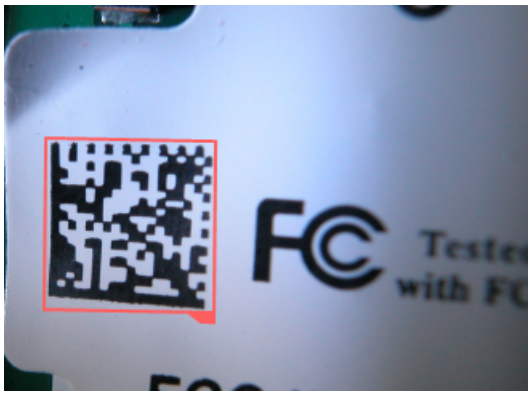


Figura 2.3: Codice Data Matrix su un chip [9]

Entrambi i codici QR e Data Matrix utilizzano il codice di Reed-Solomon per aggiungere ridondanza al messaggio da memorizzare, permettendo in tal modo di decodificare correttamente anche nei casi in cui il codice sia stato parzialmente danneggiato. Di seguito si parlerà della più recente versione del codice chiamata Data Matrix ECC200 e in particolare nella sua versione standardizzata GS1 [14].

I codici Data Matrix possono essere sia di forma quadrata che rettangolare, sono costituiti da punti bianchi e neri e ogni punto rappresenta un bit: un punto bianco si riferisce a '0' e uno nero a '1'. La dimensione dei punti può variare e ciò dipende dal lettore e marcatore utilizzati, in generale si aggira intorno agli 0.2 - 0.6 mm.

Il contrasto tra le zone bianche e nere diventa di fondamentale importanza nella fase di lettura del dato, esso deve essere almeno del 20% per una decodifica corretta [14].

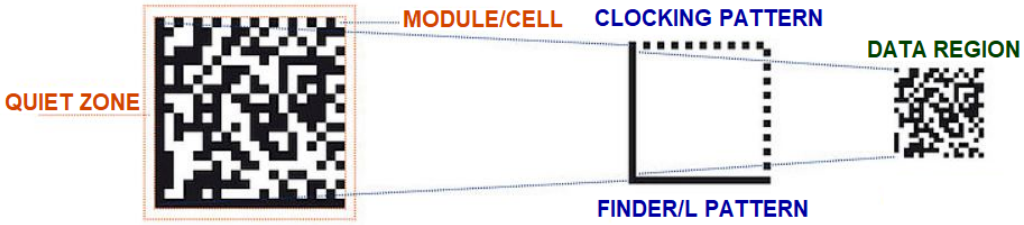


Figura 2.4: Componenti del codice Data Matrix [10]

Il codice bidimensionale si compone poi delle seguenti parti fondamentali:

- Quiet zone: È l'area che circonda la matrice e che deve rimanere libera da ogni interferenza per leggerne correttamente il contenuto. Per questo motivo, deve avere una larghezza almeno pari a una riga o a una colonna, tuttavia per un'affidabilità superiore, le dimensioni possono anche aumentare.

- Modulo o cella: È l'unità fondamentale della matrice, il loro numero (sempre pari) varia da un minimo di 10×10 ad un massimo di 144×144 moduli a matrice. Quando si supera il valore di 32×32 moduli il codice viene diviso in blocchi più piccoli per prevenire distorsioni.

- Clocking pattern: Sul lato destro e in alto, il codice presenta una serie alternata di punti bianchi e neri. Sono utilizzati per determinare il numero di righe e di colonne oltre alla dimensione dei moduli.

- Finder o L pattern: Sul lato sinistro e in basso, è costituito di soli punti neri e viene utilizzato nella fase di lettura per localizzare e orientare il simbolo in modo che possa essere letto in qualunque direzione. Un deterioramento del Finder può così compromette l'intero processo di lettura.

- Data region: È la zona dove risiede l'informazione codificata compresa di dato e ridondanza. In Figura 2.5 viene mostrato il contenuto informativo associato in funzione della dimensione del simbolo.

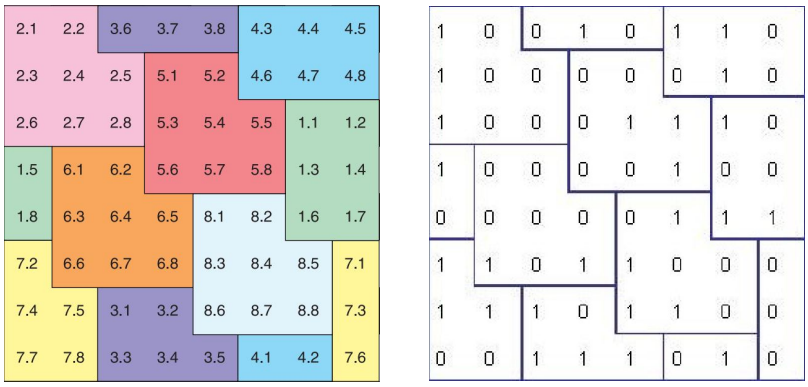
Symbol Size																								
Rows	10	12	14	16	18	20	22	24	26	32	36	40	44	48	52	64	72	80	88	96	104	120	132	144
Columns	10	12	14	16	18	20	22	24	26	32	36	40	44	48	52	64	72	80	88	96	104	120	132	144
Data Capacity																								
Numeric	6	10	16	24	36	44	60	72	88	124	172	228	288	348	408	560	736	912	1152	1392	1632	2100	2608	3116
Alphanumeric	3	6	10	16	25	31	43	52	64	91	127	169	214	259	304	418	550	682	862	1042	1222	1573	1954	2335
Byte	1	3	6	10	16	20	28	34	42	60	84	112	142	172	202	278	366	454	574	694	814	1048	1302	1556

Figura 2.5: Contenuto informativo in funzione delle dimensioni del simbolo [15]

È bene specificare che la dimensione della matrice viene determinata dalla quantità di dati da codificare e non dalla percentuale di correzione d'errore desiderata. È possibile tuttavia, attraverso determinati standard applicativi, scegliere la soluzione migliore per una fissata codifica. In generale, la percentuale di bit corretti si aggira intorno al 20-30% [11]. Come detto, Data Matrix sfrutta il codice di Reed-Solomon per la fase di correzione d'errore, che avviene al momento della scansione del simbolo. Tali errori sono spesso il risultato di problemi di stampa, riflessioni o degradazioni della superficie stampata.

Facciamo ora un esempio di codifica Data Matrix. Supponiamo di voler scrivere la stringa '123456'. La rappresentazione ASCII dei 6 caratteri equivale a 3 byte ('142', '164', '186', in decimale) che corrispondono alle prime tre codewords del codice che stiamo costruendo. Utilizzando l'algoritmo di Reed-Solomon otteniamo altre 5 codewords.

Si hanno così otto codewords che rappresentate in forma binaria sono: 10001110, 10100100, 10111010, 01110010, 00011001, 00000101, 01011000, 01100110. Queste vengono disposte come mostrato in Figura 2.6a.



(a) (b)

Figura 2.6: Codifica Data Matrix [14]

Prendiamo la prima codeword, che si troverà nel riquadro verde: i bit vengono disposti in un ordine preciso partendo dal bit più significativo 1.1 fino a quello meno significativo 1.8. Otteniamo così una matrice di bit (2.6b), a ognuno di essi corrisponde o una celle bianca o una nera (2.7a). Infine vengono aggiunti clocking pattern e L pattern (2.7b) a completamento del codice.

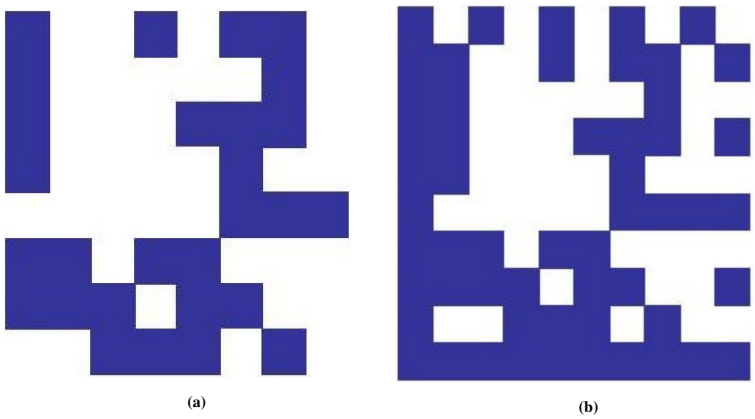


Figura 2.7: Codifica Data Matrix [14]

Il sistema di trasmissione oggetto di questa tesi risulta, per molti aspetti, simile al codice Data Matrix. Entrambi trasmettono informazione attraverso linguaggi multimediali (rispettivamente suoni e immagini) ed entrambi cercano di stabilire una comunicazione quanto più affidabile, sfruttando peraltro il medesimo codice a correzione d'errore.

2.3 Comunicazione acustica sottomarina

Le comunicazioni acustiche sottomarine abbreviate in UAC (*Underwater Acoustic Communications*) sono sistemi di trasmissione che utilizzano le onde sonore come segnali e l'acqua come mezzo trasmissivo. Poiché il suono viaggia più velocemente in acqua rispetto che in aria questa tecnica viene sfruttata anche da molte specie marine per comunicare su lunga distanza. Esistono però svariati fattori che rendono difficoltoso questo tipo di sistemi: propagazione *multi-path* [18], canale tempo variante, banda disponibile piccola e forte attenuazione del segnale. Tipicamente il range di frequenze utilizzate è compreso tra i 10 Hz e 1 MHz. Con frequenze al di sotto di 10 Hz non è possibile trasmettere il segnale senza penetrare anche molto in profondità nel fondale marino, invece per frequenze superiori a 1 MHz accade che le onde sonore vengono assorbite molto velocemente (Figura 2.8). Per tutte queste ragioni le UAC hanno data rate molto bassi se comparate con le comunicazioni terrestri, che fanno uso di onde elettromagnetiche anziché sonore (meccaniche).

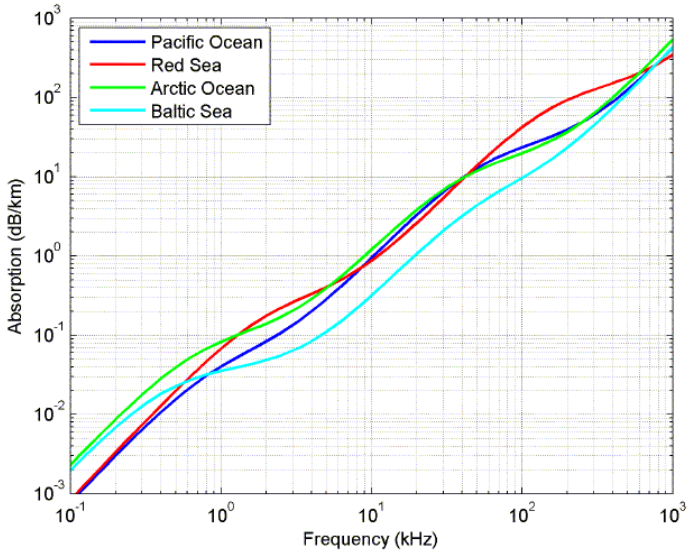


Figura 2.8: Comportamento del coefficiente di attenuazione $\alpha(f)$ [20]

Uno dei primi esempi di comunicazione sottomarina risale al 1945 ed era chiamato AN/WQC-2, soprannominato Gertrude, in dotazione alla marina militare statunitense [16]. Si trattava di un sistema di trasmissione analogica della voce a banda laterale singola (SSB) che faceva uso, sia per la trasmissione che per la ricezione, dei sonar a media frequenza preesistenti.

In Figura 2.9 viene mostrato uno schema della modulazione del segnale: la registrazione vocale passa per un filtro passa basso e la banda viene poi traslata su frequenze più alte. A questo punto, nel caso proposto in figura, viene rimossa la banda laterale (in inglese *sideband* SB) inferiore a frequenze più basse. Il segnale così ottenuto viene trasmesso dal sistema di sonar. Al ricevitore si riporta il segnale in banda base e, supponendo che le bande laterali siano simmetriche, viene rigenerata la parte di banda che era stata soppressa. Infine, dopo un ultimo filtro passa basso, il messaggio viene riprodotto da un amplificatore per essere ascoltato.

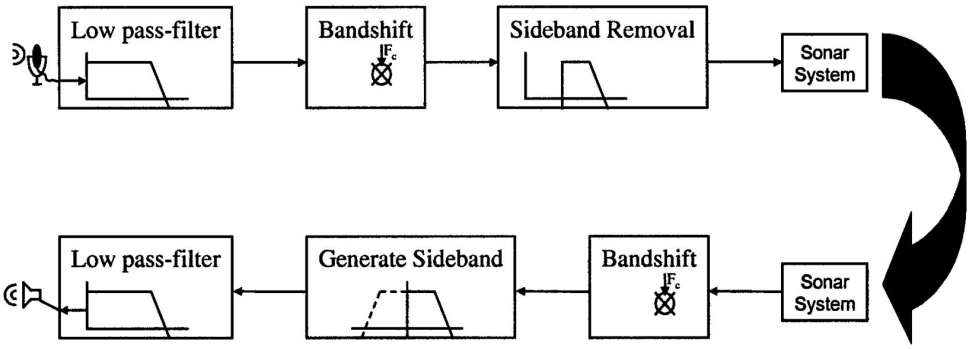


Figura 2.9: Schema a blocchi per la modulazione di ampiezza SSB [16]

Oltre alle limitazioni che in generale affliggono le UAC tuttora, Gertrude soffriva dell'ulteriore barriera che caratterizza i sistemi vocali analogici, i quali tendono a non essere adatti allo scambio di grandi quantità di informazione. La ragione per cui l'AN/WQC-2 risultasse ancora utilizzabile è dovuta al fatto che ci fossero uomini all'ascolto capaci di distinguere (per quanto potessero) il parlato dal forte rumore ambientale. Inoltre, proprio perché analogici, non potevano fare uso dei metodi di crittografia digitali allora conosciuti, e perciò risultavano spesso anche poco sicuri.

Negli anni le nuove tecnologie UAC hanno fatto enormi passi avanti rispetto al 1945 ma mancano ancora al giorno d'oggi reti sottomarine ad-hoc su larga scala che integrino un insieme sempre più eterogeneo di applicazioni e mezzi sottomarini. Nel 2017 viene approvato JANUS [17] un protocollo per la trasmissione sottomarina digitale attraverso onde sonore: si tratta di un progetto NATO, nonché il primo standard UAC rilasciato da un ente internazionale, disponibile sia per applicazioni militari che civili. Di seguito verranno omessi i dettagli del protocollo riguardanti pacchetto dati e accesso al mezzo concentrando la nostra trattazione alla sola parte di codifica e modulazione.

Lo standard JANUS è volutamente robusto e fa uso di un ben conosciuto schema di codifica tale da poter essere adottato da una vasta gamma di strumenti. Il tipo di modulazione utilizzato è chiamato FH-BSFK *Frequency-Hopped, Binary Frequency Shift Keying*, scelto per la sua semplicità nell'implementazione e affidabilità anche in ambienti difficili come quello marino. In aggiunta JANUS è fornito di un sistema per la rilevazione d'errore conosciuto come CRC *Cyclic Redundacy Check*, diversamente da altri progetti UAC che preferiscono invece utilizzare codici a correzione d'errore come il più volte citato Reed Solomon [21].

La tecnica FH viene impiegata solitamente nelle trasmissioni radio e consiste nel far variare la frequenza di trasmissione (F_S) a intervalli regolari in maniera pseudocasuale attraverso un codice prestabilito. La BFSK invece, è una modulazione ortogonale che funziona spostando la frequenza della portante su due diverse frequenze (F_S) ben definite a seconda del bit che si vuole trasmettere [5].

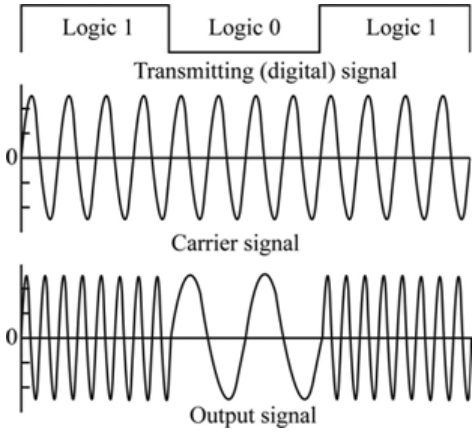


Figura 2.10: Modulazione BFSK

Nel modello JANUS gli indici FH vengono scelti secondo l'aritmetica dei campi finiti di Galois per cui, definito in partenza un numero primo primitivo, questo viene utilizzato per generare 13 coppie equispaziate di frequenze (Figura 2.11). La frequenza finale F_S, selezionata per codificare ogni bit del messaggio, è determinata univocamente dal numero di indice FH e a seconda del valore '1' o '0' assunto dal bit. L'ordine in cui le 13 coppie di toni vengono impiegate è tale da garantire minima interferenza intersimbolo (ISI), causata principalmente dal *multipath fading* [18] e dalla collisione tra i diversi utenti. La sequenza pseudo-ortogonale di frequenze è infatti nota a tutti i potenziali ricevitori ed è qui riportata in Figura 2.11.

FH	bit	freq. (Hz)	FH	bit	freq. (Hz)
0	0	9440	7	0	11680
	1	9600		1	11840
1	0	9760	8	0	12000
	1	9920		1	12160
2	0	10080	9	0	12320
	1	10240		1	12480
3	0	10400	10	0	12640
	1	10560		1	12800
4	0	10720	11	0	12960
	1	10880		1	13120
5	0	11040	12	0	13280
	1	11200		1	13440
6	0	11360			
	1	11520			

Figura 2.11: Mapping sonoro dello standard JANUS [17]

La banda totale B_w si trova così divisa in slot ognuna di larghezza $B_w/26$. Secondo le specifiche JANUS vanno assunti i seguenti valori: la banda viene generalmente scelta pari a $F_c/3$, dove $F_c = 11520$ Hz, e quindi $B_w = 4160$ Hz. Dalla divisione di B_w ne consegue che la larghezza di ogni slot risulta pari a $F_{Sw} = 160$ Hz, invece la durata di ogni frammento sonoro (associato a un bit) assume un valore pari a $C_d = 1/F_{Sw} = 6.25$ ms.

Il sistema così definito è in grado di mantenere una buona comunicazione acustica anche tra trasmettitori e ricevitori che si trovino a diversi kilometri di distanza l'uno dall'altro. Nonostante il data rate, se comparato alle connessioni internet, risulti molto basso (80 bit/s) [22], questo si rivela essere più che sufficiente per la maggior parte delle applicazioni sottomarine esistenti. È importante notare che tale valore è calcolato secondo le specifiche iniziali: JANUS viene disegnata per essere flessibile, e calcola tutti i parametri a partire dalla frequenza di centro banda scelta F_c. Aumentando F_c aumenterà anche la banda, assicurando un tasso di informazione più elevato. Il prezzo da pagare sarà però un accorciamento della distanza massima raggiungibile per la trasmissione, dovuta alla dipendenza dell'attenuazione dalla frequenza, e una minore affidabilità di comunicazione come conseguenza della diminuzione in durata C_d di ogni frammento.

Il progetto JANUS, e in generale tutte le tecnologie UAC, si dimostrano quindi molto vicine all'idea sviluppata in questa tesi: non solo per aver sostituito le onde sonore ai segnali elettrici, ma anche per l'attenzione posta alla ricerca di un mapping sonoro che si dimostrasse adeguato anche in condizioni di forte rumore ambientale.

Capitolo 3

Modulazione per quartine di note musicali

3.1 Block Diagram

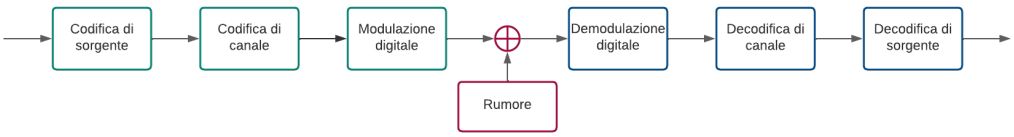


Figura 3.1: Block diagram

Il sistema di trasmissione sonoro, oggetto di questa tesi, può essere schematizzato in blocchi come si vede in Figura 3.1. Per quanto riguarda il primo blocco possiamo dire che l'approccio utilizzato dal sistema è di tipo generale e può essere applicato ai più comuni esempi di codifica di sorgente [5]. Ciò rimane valido anche quando il contenuto da trasmettere risulta essere di natura particolare, come ad esempio nel caso di una codifica incrementale [13], infatti resta possibile sfruttare il sistema, adattandolo alla codifica desiderata.

Il secondo blocco racchiude in sé la fase di codifica di canale. Come anticipato, per questo progetto è stato scelto di ricorrere ai codici di Reed Solomon discussi al paragrafo 2.1. Nelle simulazioni Matlab si è optato per il codice R-S (7,3), con simboli di m=3 bit, perché semplice e veloce e dotato di una ridondanza sufficiente ai nostri scopi. Codici di questo tipo sono però più vulnerabili ai burst error (2.1) in quanto, in caso di rumori forti e improvvisi, gli errori possono concentrarsi e coinvolgere molti simboli di una stessa codeword rendendo la correzione più difficile al momento della decodifica. Per ovviare a questo problema si è deciso di affiancare al codice R-S uno

scrambling dei bit codificati, in modo tale da distribuire i simboli errati non più su un'unica ma su più parole di codice.

I blocchi 3 e 5 (modulazione e demodulazione) verranno discussi più nel dettaglio ai paragrafi 3.2 e 3.3.

Per quanto riguarda invece il rumore, questo viene aggiunto in un secondo momento alla registrazione audio che è stata generata al termine della fase di modulazione. Così facendo, viene simulata la situazione realistica in cui il microfono del ricevitore registra il segnale nelle condizioni di ambiente rumoroso. Le tipologie di rumore scelte per la simulazione sono: rumore AWGN, strada di città, aeroporto e infine ufficio. Tutti ambienti che potrebbero sfruttare questo tipo di trasmissione pensato per poter essere quantomeno tollerato dalle persone che vivono e lavorano in questi luoghi, essendo il messaggio costituito delle sole frequenze delle note musicali.

3.2 Modulazione

Il punto cardine di questo progetto si basa proprio sul tipo di modulazione scelta: una modulazione di tipo sonoro. L'idea generale è quella di dividere il messaggio binario, compreso di dati e ridondanza, in stringhe di 14 bit ciascuno. Ad ognuno di questi frammenti verrà poi associato un suono univoco della durata di 0.35 s composto di 4 frequenze.

Per rappresentare tutte le possibili combinazioni di 14 bit abbiamo bisogno di $2^{14} = 16384$ suoni diversi, che andranno a costituire il nostro alfabeto. Il range di frequenze prese in considerazione è costituito da 87 note (all'interno della banda che va da 27.5 a 3951.07 Hz) e sono essenzialmente tutte quelle che corrispondono ai tasti di un comune pianoforte (escludendo l'ottantottesima nota troppo alta). I modi di mappare 87 note su 4 frequenze sono però moltissimi, infatti il numero di combinazioni di 87 su 4 è pari a:

$$C_{87,4} = \binom{87}{4} = 2\,225\,895$$

Potremmo pensare quindi di costruire più di due milioni di alfabeti differenti. Essendo questa un'impresa più che proibitiva, ci limiteremo a confrontare solamente 3 alfabeti diversi l'uno dall'altro che chiameremo Mappa 1, Mappa 2 e Mappa 3.

- **Mappa 1**

  Partiamo dalla scelta di f_1: questa può essere scelta in un range di note che va dalla 1 alla 64, ricordando che abbiamo a disposizione 87 note. Ciò significa che abbiamo 64 modi (64 possibilità) per scegliere il valore da associare a f_1, perciò f_1 è in grado di trasmettere un'informazione pari a 6 bit. Similmente, una volta fissato il valore di f_1, f_2 può essere scelta tra $f_1 + 2$ e $f_1 + 9$ ovvero tra 8 valori differenti portando con sé un'informazione di 3 bit. Il processo si ripete per i valori di f_3 e f_4 come mostrato in Tabella 3.1.

f_1	f_2	f_3	f_4
$1 - 64$	$f_1 + 2 - f_1 + 9$	$f_2 + 2 - f_2 + 9$	$f_3 + 2 - f_3 + 5$
64 possibilità 6 bit	8 possibilità 3 bit	8 possibilità 3 bit	4 possibilità 2 bit

Tabella 3.1: Mappa 1

  In questo modo vengono usate tutte le 87 frequenze anche se le note più alte sono meno frequenti all'interno dell'alfabeto e i suoni risultanti sono tutti centrati su

basse frequenze. Possiamo cercare di fare di meglio in tal senso costruendo una seconda mappa.

- **Mappa 2**

f_1	f_2	f_3	f_4
$10-73$	f_1-9 $-$ f_1-2	f_1+2-f_1+9	f_3+2-f_3+5
64 possibilità 6 bit	8 possibilità 3 bit	8 possibilità 3 bit	4 possibilità 2 bit

Tabella 3.2: Mappa 2

Questa volta la prima frequenza viene scelta in una posizione più centrale per cercare di sfruttare maggiormente le note più alte. Anche questo alfabeto sfrutta tutte le 87 note disponibili e segue la stessa divisione in bit di Mappa 1.

- **Mappa 3**

f_1	f_2	f_3	f_4
$27-58$	f_1-12 $-$ f_1-5	f_2+6-f_1+13	f_1+6-f_3+13
32 possibilità 5 bit	8 possibilità 3 bit	8 possibilità 3 bit	8 possibilità 3 bit

Tabella 3.3: Mappa 3

L'ultimo mapping è quello che più si discosta dai precedenti: la spartizione dei bit per ogni frequenza è più uniforme ma non vengono sfruttate tutte le 87 note, in particolare, vengono volutamente eliminate le prime 15 frequenze più basse che rischiano di fare interferenza con la corrente alternata a 50 Hz.

A questo punto viene generato il segnale audio, della durata di 0.35 s, come la somma di 4 sinusoidi alle frequenze f_1, f_2, f_3, f_4. L'accostamento di tutti suoni che compongono il messaggio danno origine a una musichetta della durata di qualche decina di secondi. Dopo l'aggiunta del rumore la registrazione passa alla fase di demodulazione.

3.3 Demodulazione

Dopo il passaggio del segnale attraverso il canale rumoroso, si ottiene un frammento corrotto dell'originale (Figura 3.2) da cui è ancora possibile ricavare l'informazione di 14 bit che era stata trasmessa. Vedremo in seguito al capitolo 4 fino a che punto questa distorsione viene tollerata senza incorrere in una decodifica errata del dato.

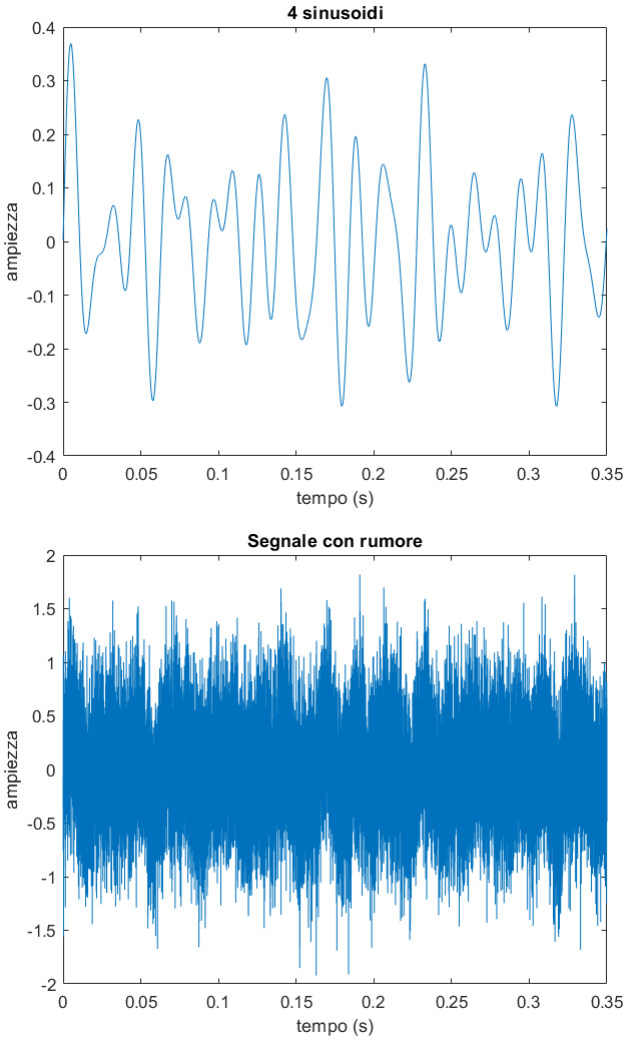


Figura 3.2: Segnale prima e dopo il canale rumoroso

Poiché risulta difficile ricostruire il segnale nel dominio del tempo, conviene sfruttare a nostro vantaggio la densità spettrale di potenza (PSD), che fornisce informazioni sul segnale nel dominio delle frequenze. La PSD del seno è infatti molto semplice e corrisponde a un picco in corrispondenza della frequenza alla quale oscilla. Possiamo quindi sfruttare questa caratteristica per ricavare la quartina di frequenze trasmesse e, di conseguenza, i 14 bit inviati.

La funzione PSD è definita come la trasformata di Fourier della funzione di autocorrelazione R_x:

$$S(f) = \int_{-\infty}^{+\infty} R_x(\tau) \cdot \mathrm{e}^{-j2\pi f\tau} d\tau \tag{3.1}$$

dove:

$$R_x(\tau) = \frac{1}{T_0} \int_{-T_0/2}^{+T_0/2} x(t) \cdot x(t-\tau)^* dt \tag{3.2}$$

Per i segnali periodici possiamo sfruttare la serie di Fourier e scrivere la funzione di autocorrelazione in funzione dei coefficienti di Fourier x_n:

$$R_x(\tau) = \sum_{n=-\infty}^{+\infty} |x_n|^2 \cdot \mathrm{e}^{j2\pi f_n \tau} \tag{3.3}$$

Applicando la trasformata di Fourier a R_x si ottiene infine:

$$S(f) = \sum_{n=-\infty}^{+\infty} |x_n|^2 \cdot \delta(f - f_n) \tag{3.4}$$

Quando il segnale x(t) è la somma di 4 sinusoidi, si hanno i seguenti coefficienti di Fourier x_n:

$$\begin{aligned} x_1 &= \tfrac{A}{2} \cdot \mathrm{e}^{j\phi_1} & x_{-1} &= \tfrac{A}{2} \cdot \mathrm{e}^{-j\phi} \\ x_2 &= \tfrac{A}{2} \cdot \mathrm{e}^{j\phi_2} & x_{-2} &= \tfrac{A}{2} \cdot \mathrm{e}^{-j\phi_2} \\ x_3 &= \tfrac{A}{2} \cdot \mathrm{e}^{j\phi_3} & x_{-3} &= \tfrac{A}{2} \cdot \mathrm{e}^{-j\phi_3} \\ x_4 &= \tfrac{A}{2} \cdot \mathrm{e}^{j\phi_4} & x_{-4} &= \tfrac{A}{2} \cdot \mathrm{e}^{-j\phi_4} \end{aligned}$$

Per cui lo spettro di ciascun frammento, considerando solo la porzione positiva in quanto funzione pari, risulta essere del tipo:

$$S(f) = \frac{A^2}{4}\delta(f - f_1) + \frac{A^2}{4}\delta(f - f_2) + \frac{A^2}{4}\delta(f - f_3) + \frac{A^2}{4}\delta(f - f_4) \tag{3.5}$$

Come mostrato in Figura 3.3 alle PSD di ciascun suono corrispondono quindi 4 picchi alle frequenze delle 4 note musicali di cui è composto.

Come discusso in precedenza questa quartina di note musicali determina in maniera univoca un numero a 14 bit: dalla successione di diversi frammenti si può così ricostruire l'intero messaggio inviato.

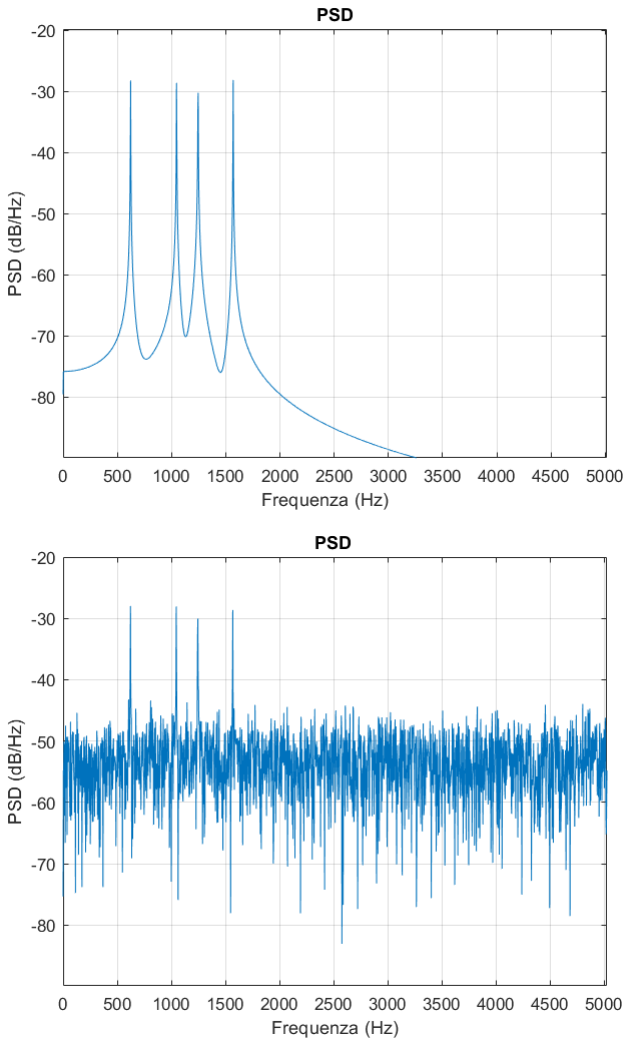


Figura 3.3: Densità spettrale di potenza prima e dopo l'aggiunta di rumore

Un meccanismo simile viene sfruttato anche dal software di successo Shazam [19], utilizzato per il riconoscimento di brani musicali a partire da una registrazione audio della durata di pochi secondi. L'applicazione basa il suo funzionamento sullo spettrogramma: una rappresentazione tempo-frequenza del segnale, costituita da una matrice contente le trasformate di diverse sottosezioni del brano registrato. Dalla matrice si ricavano le cosiddette *feature*, caratteristiche che identificano ogni traccia musicale disponibile nella banca dati: queste vengono scelte selezionando le frequenze a cui corrispondono i picchi delle trasformate, così come implementato per la PSD dal nostro algoritmo. Infine, occorre la fase di *feature matching* che ha lo scopo di trovare un riscontro, quanto più fedele, con un brano già presente nel database.

Nel nostro caso il match ha sempre successo, perché vengono selezionate solo quelle frequenze di picco che corrispondono a una nota musicale presente nell'alfabeto.

Non si è però certi che tale frequenza sia quella corretta a causa delle variazioni dello spettro introdotte dal rumore. Nel caso di rumore AWGN, in cui lo spettro è costante, o quando in generale i rumori sono deboli, rilevare le frequenze trasmesse in principio risulta più facile: come mostrato in Figura 3.3 infatti, i 4 picchi sono ancora ben visibili e rilevabili. La situazione si complica quando, a causa di rumori forti e improvvisi, lo spettro del segnale presenta nuovi picchi dovuti al solo rumore, che superano in intensità quelli originali che si volevano trasmettere. In queste circostanze ci si affida al potere correttivo dei codici di Reed-Solomon.

3.4 Pseudocodice

Di seguito viene riportato lo pseudocodice utilizzato per simulare il sistema di trasmissione proposto, e viene qui diviso nelle sue tre parti fondamenti: codifica, aggiunta del rumore e decodifica.

Algorithm 1 Codifica

Input: Stringa di bit casuali
Output: Registrazione audio del messaggio codificato

// La stringa viene riscritta come una sequenza di numeri decimali a m=3 bit
data ← RESHAPE(input,m)
// 'data' viene riorganizzata in una matrice di k=3 colonne: sulle righe troviamo
// in questo modo i primi 3 simboli di informazione che andranno poi a costruire
// le codewords finali composte di 7 simboli
to_code ← RESHAPE(data,k)
// La matrice 'to_code' è data in ingresso alla funzione rsenc che aggiunge n−k=4
// colonne di ridondanza: ora ogni riga è una codewords del codice R-S (7,3)
coded ← RSENC(to_code)
// Fase di scrambling discussa in 3.1
coded ← SCRAMBLING(coded)
// Dopo aver riportato la matrice 'coded' a un vettore binario questo viene tradotto
// in una sequenza di numeri decimali a 14 bit per la fase di modulazione come
// visto in 3.2
to_send ← RESHAPE(coded)
// Viene scelto uno dei tre alfabeti disponibili
if si vuole usare 'Mappa 1' **then**
    Alphabet ← $Mappa_1$
else if si vuole usare 'Mappa 2' **then**
    Alphabet ← $Mappa_2$
else
    Alphabet ← $Mappa_3$
end if
// Ad ogni numero decimale a 14 bit viene associato un suono composto di
// 4 sinusoidi alle 4 frequenze a esso associate, concatenando frammenti della
// durata di 0.35 s
for i ← 1 to LENGTH(to_send) **do**
    audio ← $0.1\,sin(2\pi f_1 t) + 0.1\,sin(2\pi f_2 t) + 0.1\,sin(2\pi f_3 t) + 0.1\,sin(2\pi f_4 t)$
end for
return audio

Algorithm 2 Rumore

Input: Audio del messaggio codificato
Output: Audio in ingresso a cui è stato aggiunto rumore

// Viene chiesto il valore di SNR in dB e il tipo di rumore desiderato
noise_SNR ← INPUT()
noise_type ← INPUT()
// Si genera il rumore
noise ← CREATENOISE(noiseSNR,noiseType)
// Il rumore viene aggiunto all'audio in ingresso
noise_audio ← audio + noise
return noise_audio

Algorithm 3 Decodifica

Input: Audio con aggiunta di rumore
Output: Messaggio binario decodificato

// Per ogni frammento audio di 0.35 s
for ogni frammento **do**
    // viene calcolata la PSD come spiegato in 3.3
    psd ← PSD(frammento)
    // e da questo si ricavano i 4 picchi corrispondenti alle 4 frequenze
    $[f1, f2, f3, f4]$ ← MAX(psd)
    // per tali frequenze si cerca il corrispondente simbolo in decimale a 14 bit
    // nell'alfabeto
    for i ← 1 to LENGTH(Alphabet) **do**
        // valgono anche permutazioni della quartina
        if $[f1, f2, f3, f4]$=Alphabet(i) **then**
            received ← i
        end if
    end for
end for
// Si procede con lo scrambling inverso
received ← DESCRAMBLING(received)
// Si riorganizza il vettore come una matrice di n=7 righe e k=3 colonne
to_decode ← RESHAPE(received)
// Avviene la fase di decodifica e infine si riporta il messaggio in bit
decoded ← RESHAPE(RSENC(to_decode))

Capitolo 4

Risultati

Lo scopo principale di questa tesi, è quello di ideare un sistema di trasmissione basato su segnali sonori. Poiché le frequenze utilizzate sono percepibili dall'orecchio umano, è importante non solo studiarne l'efficacia in termini di velocità di trasmissione, ma anche l'impatto che potrebbe avere nella vita di tutti i giorni.

Tutti gli alfabeti in esame, seppur costruiti in modo differente, generano melodie pressoché analoghe al momento dell'ascolto. La più piacevole risulta comunque essere quella prodotta da Mappa 3 in quanto, eliminando le 15 frequenze più basse e pure alcune frequenze più alte, i suoni si rivelano essere più armoniosi e meno randomici rispetto alle due precedenti proposte.

Data la piccola entità di tali differenze, per confrontare i tre diversi tipi di mapping sonoro, ne viene studiato il comportamento in relazione al valore di SNR e al tipo di rumore ambientale scelto. Come anticipato, verranno analizzate le prestazioni dei tre alfabeti in risposta a quattro diversi tipi di rumore: AWGN, aeroporto, strada di città e infine ufficio.

Per misurare l'efficienza di ogni sistema faremo riferimento alla probabilità di errore sul bit (Pbit), in inglese *bit error probability*, calcolata nelle simulazioni come numero di bit sbagliati su numero di bit totali trasmessi. Aumentare a sproposito la potenza del segnale, per evitare di commettere errori, non è però una soluzione accettabile perché risulterebbe, non solo dispendioso, ma soprattutto inutilizzabile in ambienti popolati dall'uomo. Per questo motivo, utilizziamo un sistema di tipo FEC (*Forward Error Correction*) ricorrendo ai codici di Reed-Solomon e al successivo scrambling (paragrafo 3.1). Un altro fattore da tenere in considerazione è rappresentato perciò dalla percentuale di bit che sono stati corretti (di colore azzurro nelle figure) grazie all'introduzione della codifica di canale.

Possiamo quindi confrontare il sistema prima e dopo il codice di correzione attraverso le due misurazioni: probabilità di errore sul bit pre-codifica (rappresentata in blu) e probabilità di errore sul bit post-codifica (di colore acquamarina nei successivi grafici). È auspicabile che la Pbit pre-codifica e post-codifica siano basse: questo vuol dire, infatti, che il sistema commette pochi errori e, di conseguenza, la trasmissione avviene correttamente. Al contrario, la percentuale di bit corretti deve essere alta al fine di correggere quanti più errori commessi: tanto più numerose sono le correzioni tanto inferiore sarà la Pbit post-codifica rispetto a quella pre-codifica. Vedremo in seguito che, per valori di SNR contenuti, il codice di Reed-Solomon è in grado di correggere tutti gli errori riscontrati e di azzerare così del tutto la bit error probability post-codifica, assicurando una comunicazione robusta ed efficiente tra i due utenti.

Nel secondo paragrafo di questo capitolo verranno infine discussi i valori risultanti di bitrate e data rate del sistema così realizzato.

4.1 Risposta al rumore

- **Rumore AWGN**

  Il primo tipo di rumore preso in esame è il rumore bianco o AWGN (*Additive White Gaussian Noise*) che, sebbene sia il meno rilevante per simulare il sistema reale, rappresenta il primo punto di riferimento per quest'analisi. Come mostrato in Figura 4.1 la mappa che si comporta meglio appare essere Mappa 3, commettendo meno errori rispetto agli altri due mapping. Mappa 1 invece si dimostra essere la peggiore, sbagliando per tutti i valori di SNR considerati.

  Va inoltre osservato che tutti i modelli vengono messi in difficoltà per valori inferiori a -20 dB quando gli errori diventano tanto numerosi da non poter essere compensati dalle correzioni apportate dalla codifica di canale. Ciò accade in quanto lo spettro del rumore bianco, essendo piatto, è in grado di alterare la PSD risultante solo per valori molto bassi di SNR. Quando ciò accade, lo spettro piatto raggiunge in intensità i picchi in frequenza del messaggio, e di conseguenza gli errori aumentano rapidamente senza possibilità di correzione.

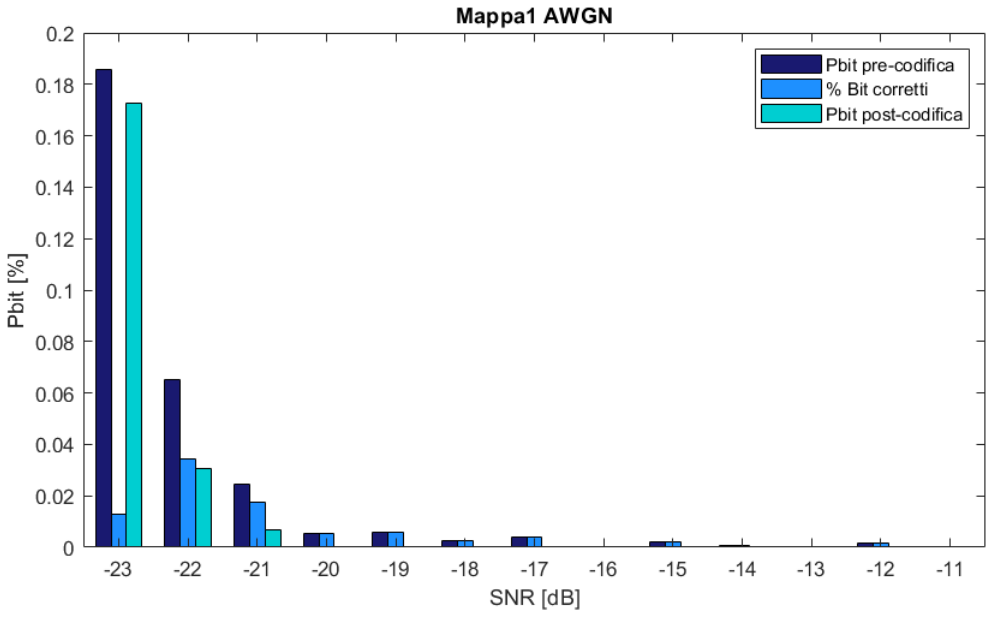
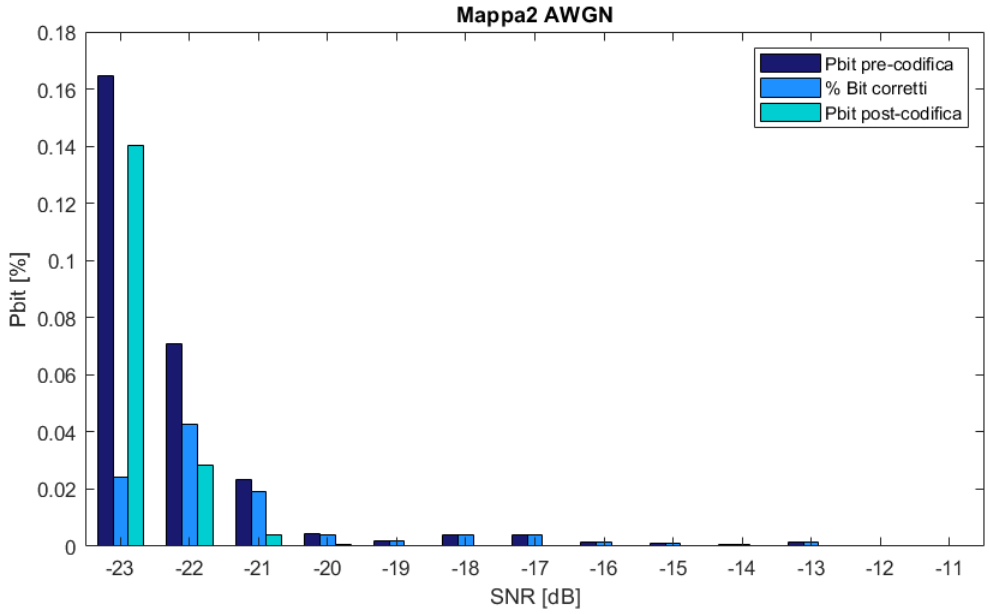
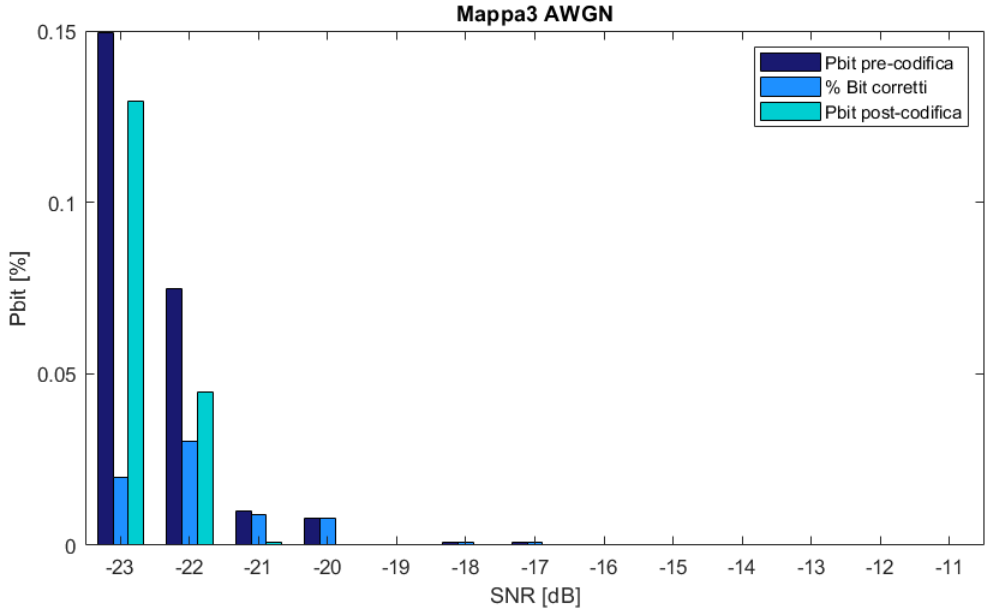


Figura 4.1: Bit error probability pre-codifica e post-codifica rumore AWGN





- **Aeroporto**

  Per le simulazioni Matlab è stata utilizzata una registrazione di un aeroporto nella quale si possono distinguere i passi e il brusìo della folla uniti alla voce degli altoparlanti e al decollo degli aerei. Quest'ultimo rappresenta di fatto l'unico elemento improvviso in un rumore pressoché simile al precedente AWGN.

  Al contrario però di quanto visto nel caso precedente, possiamo notare una notevole differenza di prestazioni tra le prime due mappe e la terza. Mentre Mappa 1 e 2 si comportano in modo simile, il terzo alfabeto compie invece quasi il doppio

degli errori. Ciò ricorrerà anche nei successivi casi analizzati ed è dovuto principalmente al fatto che, nella creazione del terzo alfabeto, è stato utilizzato un range di frequenze più limitato, costituito da 69 note anziché 87. I toni dei frammenti infatti, essendo più ravvicinati fra loro, fanno sì che un rumore improvviso (come il decollo dell'aeroplano) sposti più facilmente il picco in frequenza verso un tono vicino ma diverso da quello che si voleva trasmettere in principio, decodificando così una sequenza di bit differente.

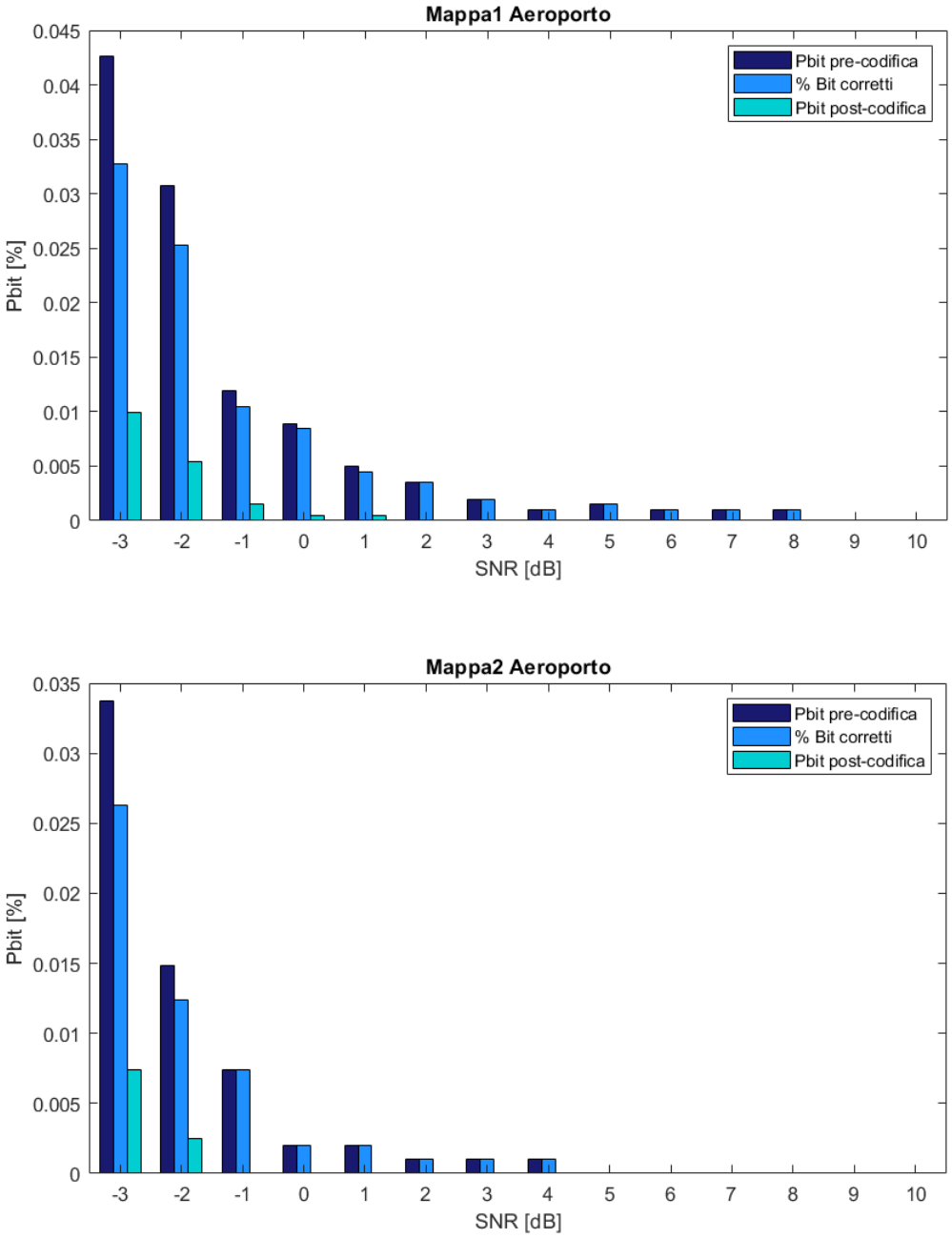
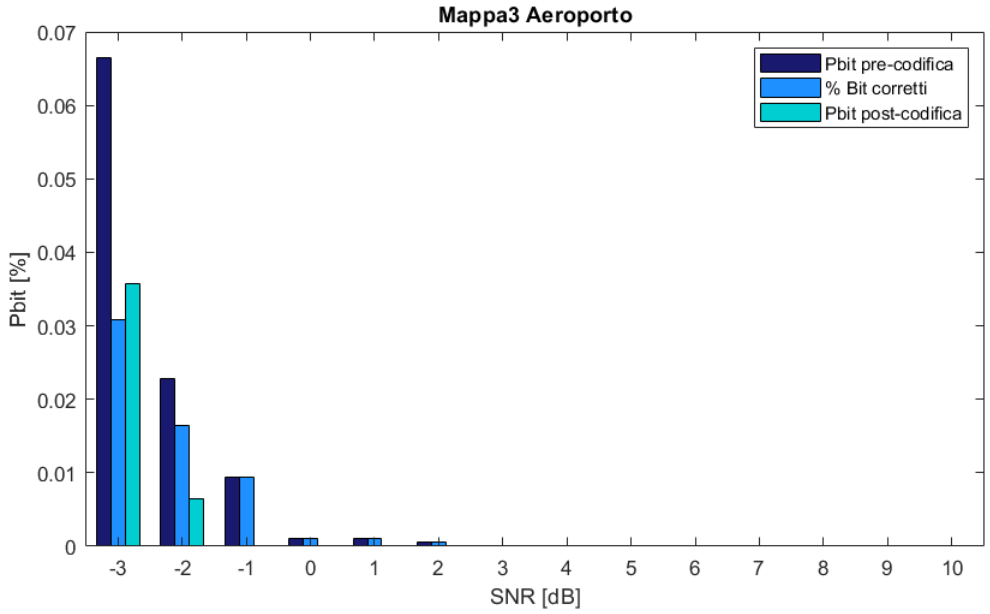


Figura 4.2: Bit error probability pre-codifica e post-codifica rumore Aeroporto



- **Strada di città**

  Nel caso di una strada cittadina molto affollata, caratterizzata dal rumore della folla unito a quello del traffico e dei clacson, è sicuramente il caso che più si discosta dal nostro riferimento di rumore bianco. Salta immediatamente all'occhio la differenza con i punti precedenti: gli errori sono più numerosi e si verificano per tutti i valori di SNR presi in considerazione.

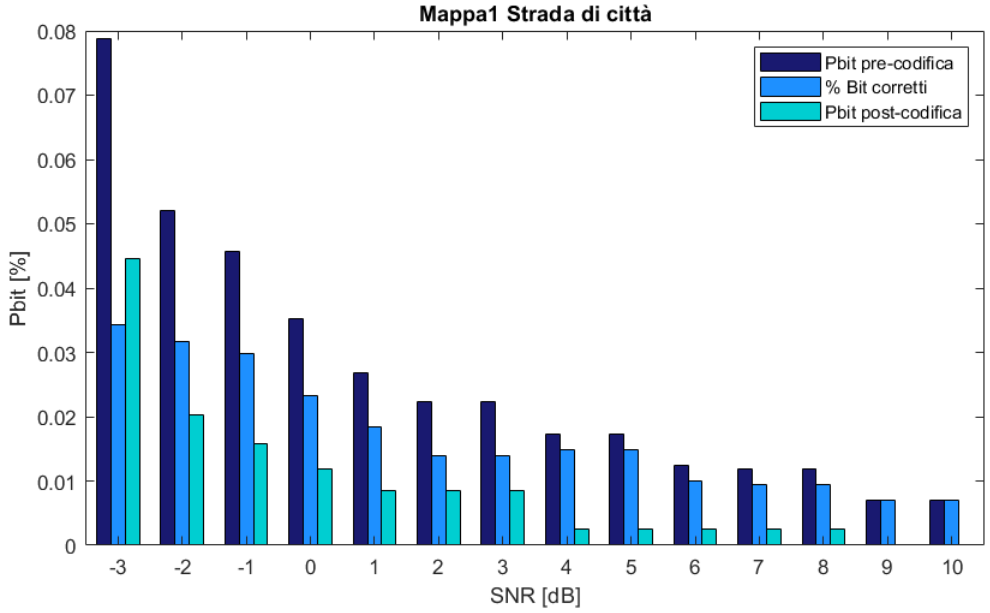
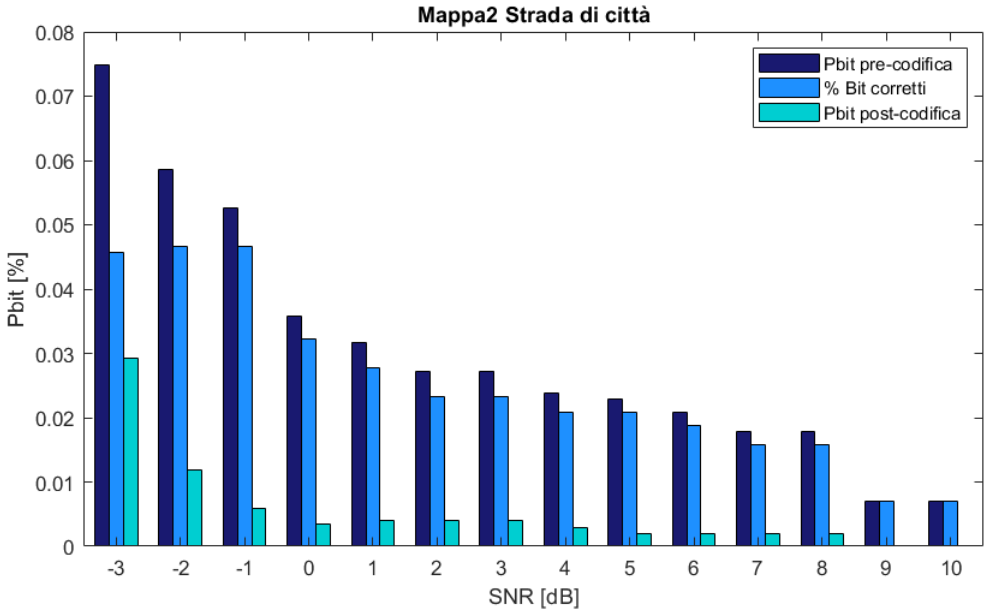
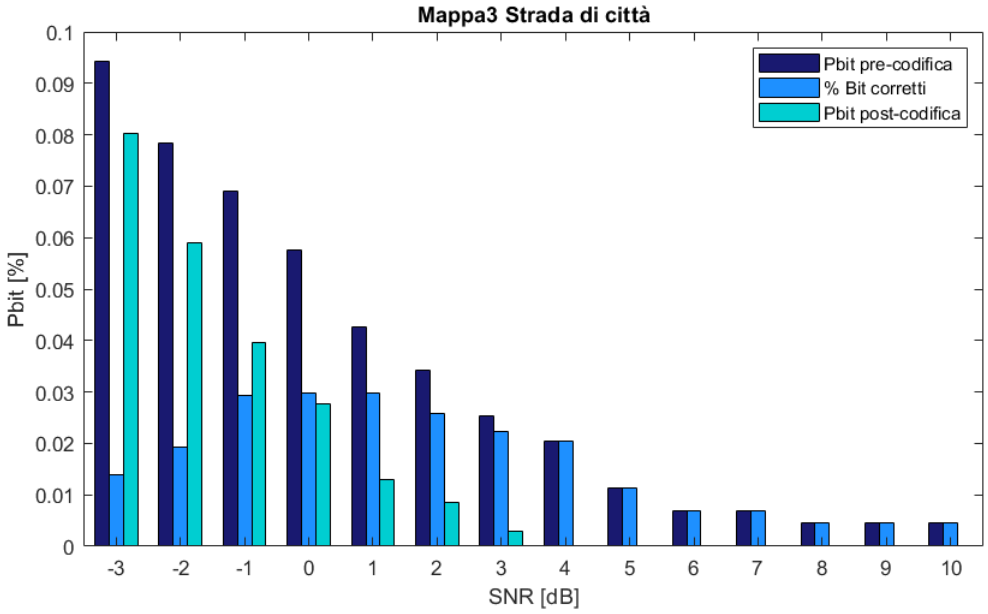


Figura 4.3: Bit error probability pre-codifica e post-codifica rumore Strada cittadina





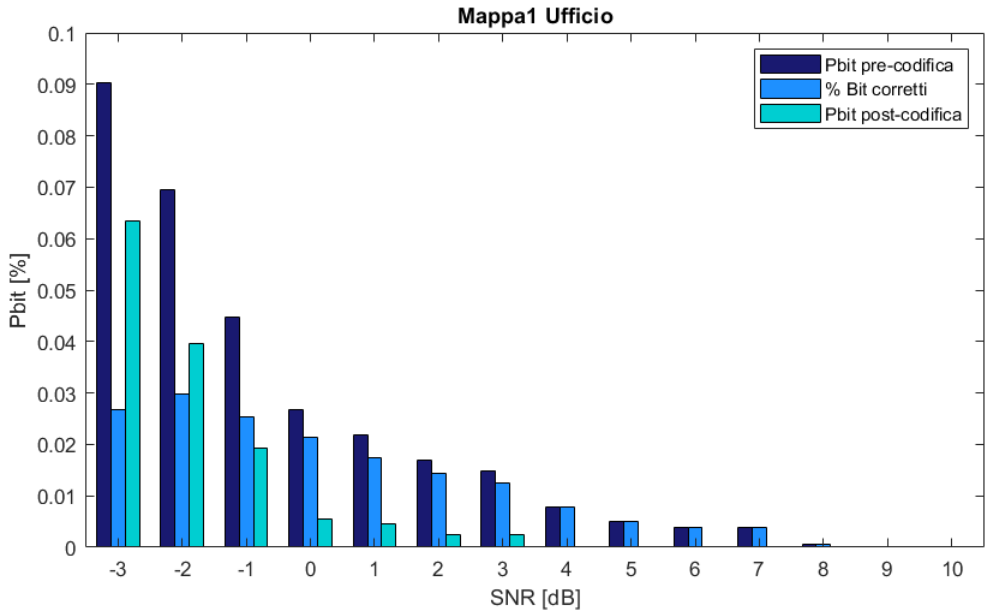
La miglior prestazione tra i tre mapping appare essere Mappa 2. Questo comportamento può essere spiegato ancora una volta dalla scelta eseguita in fase di progettazione del mapping: vengono utilizzate infatti tutte le 87 note a disposizione e i 4 toni vengono presi distanziandoli il più possibile l'uno dall'altro all'interno dell'intervallo di frequenze.

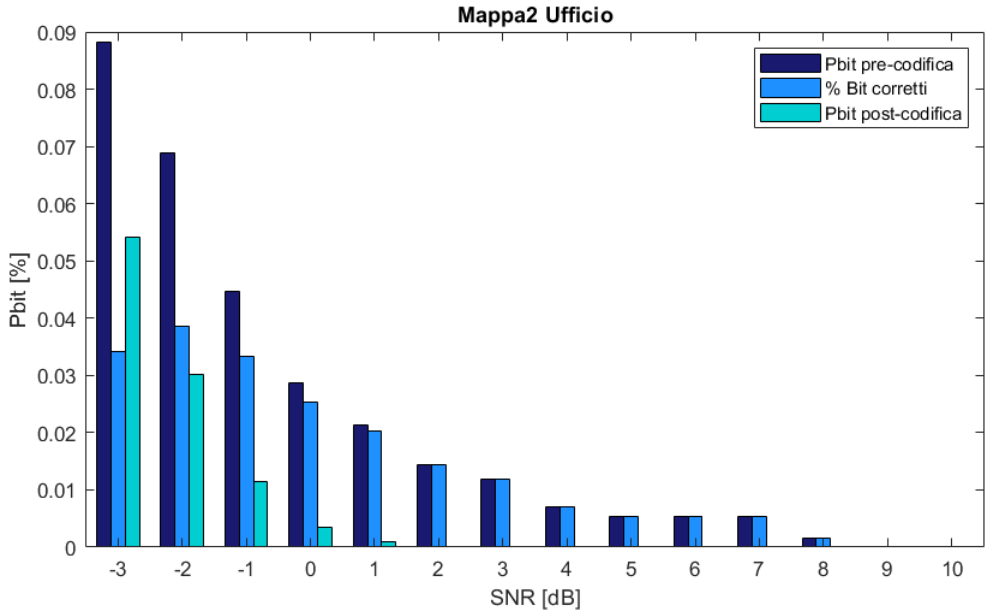
È interessante notare che, a prescindere dal rumore analizzato, la percentuale di bit corretti (rappresentata dalle barre azzurre in Figura 4.3) decresce all'aumentare dei bit errati quando la bit error probability raggiunge il valore di circa 0.07%.

In questi casi gli errori sono numerosi e ravvicinati anche dopo aver effettuato lo scrambling, ciò fa variare un numero di simboli all'interno della parola di codice superiore al limite massimo correggibile dal codice di Reed-Solomon (7,3).

- **Ufficio**

  L'ultimo caso vuole simulare i rumori presenti all'interno di un comune ufficio: sono distinguibili alcune voci, il suono della scrittura sulla tastiera e anche lo squillo di qualche telefono.





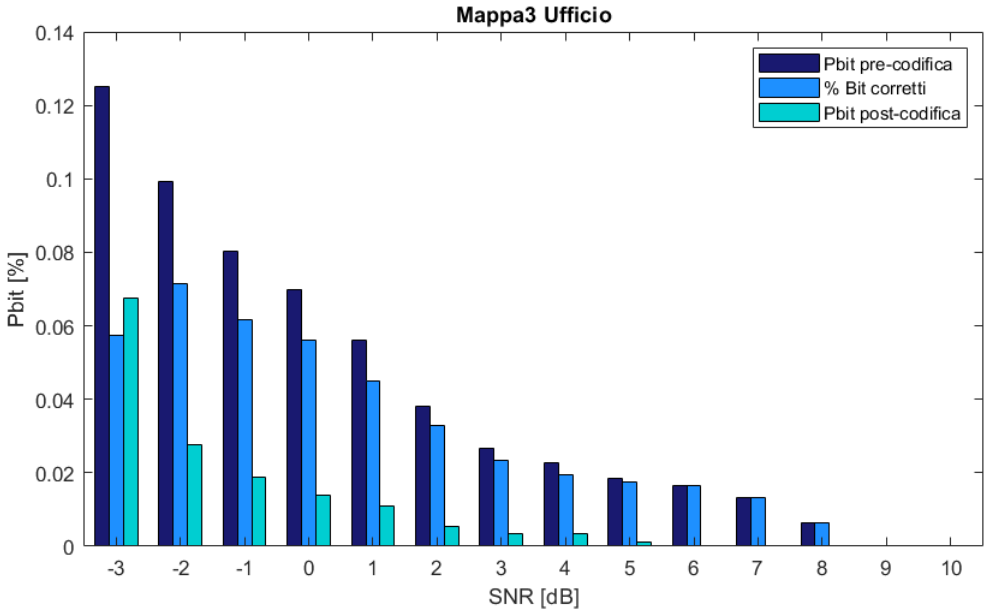


Figura 4.4: Bit error probability pre-codifica e post-codifica rumore Ufficio

Il comportamento anche in questa situazione riflette quello dei precedenti punti. Mappa 1 e Mappa 2 sono le più prestazionali con risultati leggermente migliori per la seconda.

4.2 Bitrate e Data rate

Un altro punto fondamentale da valutare per un sistema di trasmissione, è il valore di bitrate e data rate che viene raggiunto.

Sulla base delle simulazioni è stato scelto per la durata di ogni frammento sonoro il valore di $T = 0.35$ s. Diminuendo ulteriormente tale numero si è riscontrato infatti un notevole aumento degli errori commessi e contemporaneamente un drastico calo del potere correttivo da parte del codice R-S (7,3).

Poiché ogni frammento trasmette 14 bit possiamo calcolare il bitrate come:

$$R_b = \frac{bit\ trasmessi}{T} = \frac{14\ bit}{0.35\ s} = 40\ \frac{bit}{s} \tag{4.1}$$

Di tale rate R_b non tutti i bit trasmessi sono d'informazione ma va considerata anche la ridondanza aggiunta dalla codifica di canale.

Il data rate è quindi dato dalla relazione:

$$R_d = \frac{k}{n}\ R_b = \frac{3}{7} \cdot 40\ \frac{bit}{s} \simeq 17\ \frac{bit}{s} \tag{4.2}$$

dove $k = 3$ sono il numero di simboli d'informazione del codice R-S mentre $n = 7$ è il numero totale di simboli compresi quelli destinati alla ridondanza.

Capitolo 5

Conclusioni e sviluppi futuri

5.1 Valutazioni finali

A fronte delle diverse simulazioni e successive analisi abbiamo potuto verificare l'efficacia di questo sistema di trasmissione. Sebbene tutti i mapping sonori ideati si siano dimostrati adatti al fine di stabilire la comunicazione richiesta, abbiamo riscontrato le migliori prestazioni con il secondo tipo di alfabeto.

Anche la scelta di utilizzare il codice di Reed-Solomon (7,3) abbinato ad uno scrambling dei bit codificati si è confermata vincente. Da un lato ha permesso di correggere, per valori di SNR contenuti, buona parte degli errori commessi anche se dotato di una ridondanza limitata. Inoltre, l'aver preferito codewords composte di pochi simboli ha conferito al sistema maggiore flessibilità: per messaggi di breve durata infatti, una singola parola di codice può essere trasmessa in poco più di mezzo secondo. È bene comunque ricordare che la capacità correttiva del sistema dipende in buona parte dallo scrambling e sono quindi preferibili messaggi di più lunga durata affinché se ne manifesti l'efficacia.

5.2 Prospettive future

Al fine di aumentare il bitrate di 17 bit/s, piuttosto basso, si potrebbe pensare di espandere l'alfabeto anche a frequenze che non rientrino nel gruppo delle note musicali. In questo modo, non solo si potrebbero trasmettere più di 14 bit per frammento sonoro, ma se ne trarrebbe vantaggio anche in termini di robustezza al rumore. Si è infatti notato, anche attraverso le precedenti simulazioni, quanto la "distanza" tra le 4 frequenze scelte influisca sulla percentuale di bit sbagliati.

Un altro punto da tenere in considerazione per successive implementazioni è rappresentato dall'accesso al mezzo e quindi del problema dalla trasmissione simultanea di diversi utenti. Potrebbe risultare ideale un approccio simile allo standard JANUS descritto al paragrafo 2.3, dove viene scelto un meccanismo di *Carrier Sensing Multiple Access* (CSMA) con *Collision Avoidance* (CA) [17]. In alternativa si potrebbe pensare di implementare un sistema *Frequency Division Multiple Access* (FDMA) nel quale i diversi utenti comunicano attraverso mapping differenti costruiti su intervalli diversi di frequenze, in tal caso risulterebbe essenziale l'estensione dell'alfabeto discussa poc'anzi.

Infine, per sincronizzare la trasmissione in ricezione, ed evitare la registrazione sbagliata o del tutto errata del dato, diventa indispensabile l'utilizzo di una stringa di bit, e di conseguenza una sequenza di suoni speciale, adibita a preambolo del messaggio.

Bibliografia

[1] Reed, I. S. and Solomon, G., *Polynomial Codes Over Certain Finite Fields*, SIAM Journal of Applied Math, vol. 8, pp. 300-304. , 1960.

[2] B. Sklar, *Reed-Solomon Codes*, `https://ptgmedia.pearsoncmg.com/images/art_sklar7_reed-solomon/elementlinks/art_sklar7_reed-solomon.pdf`, pp. 1-10.

[3] B. Tomasi, P. Casari, L. Badia, M. Zorzi, *Cross-layer analysis via Markov models of incremental redundancy hybrid ARQ over underwater acoustic channel*, Ad Hoc Networks, 34, pp: 62-74, 2015.

[4] L. Badia, M. Levorato, and M. Zorzi, *Markov analysis of selective repeat type II hybrid ARQ using block codes*, IEEE Transactions on Communications 56, no. 9, pp: 1434-1441, 2008.

[5] Nevio Benvenuto e Michele Zorzi, *Principles of communications Networks and Systems*, John Wiley & Sons, 2011.

[6] P. Targa, *Progetto di una codifica di canale in ambito acustico per segnali digitali*, Tesi di Laurea in Ingegneria dell'Informazione, Università di Padova, 2021.

[7] L. Badia, M. Mastrogiovanni, C. Petrioli, S. Stefanakos, and M. Zorzi, *An optimization framework for joint sensor deployment, link scheduling and routing in underwater sensor networks*, SIGMOBILE Mob. Comput. Commun. Rev. 11, 2007, 44–56.

[8] L. Badia, *On the impact of correlated arrivals and errors on ARQ delay terms* IEEE Trans. Commun., vol. 57, no. 2, pp. 334-338, Feb. 2009.

[9] *Data Matrix su chip* `https://docs.adaptive-vision.com/4.12/studio/filters/Datacodes/ReadMultipleDataMatrices_Deprecated.html`

[10] *Componenti del codice Data Matrix* https://www.pelling.eu/barcode/data-matrix-code-caratteristiche-tipologie

[11] *Data Matrix Bar Code Symbology Specification*, ISO/IEC 16022:2006, ISO/IEC, Sep. 2006.

[12] Reuben Schuitemaker, Xun Xu, *Product traceability in manufacturing: A technical review*, Procedia CIRP, Volume 93, 2020, pp. 700-705, https://www.sciencedirect.com/science/article/pii/S2212827120306922

[13] L. Badia, M. Levorato, M. Zorzi, *Analysis of Selective Retransmission Techniques for Differentially Encoded Data*, Proc. IEEE ICC 2009.

[14] Global Standards One, *GS1 DataMatrix Guideline*, 2018 https://www.gs1.org/docs/barcodes/GS1_DataMatrix_Guideline.pdf

[15] SICK AG, *Competence Guide Direct Part Marking*, 2007 https://cdn.sick.com/media/docs/1/01/101/special_information_competence_guide_direct_part_marking_en_im0058101.pdf

[16] Shinego Michael, Edelson Geoff, Menas Francine, Richman Michael, Nation Robert, *Underwater Acoustic Data Communications for Autonomous Platform Command, Control and Communications*, 2001, https://apps.dtic.mil/sti/pdfs/ADA386718.pdf

[17] J. Potter, J. Alves, D. Green, G. Zappa, I. Nissen, K. McCoy, *The JANUS underwater communications standard*, Underwater Communications and Networking (UComms), 2014, pp. 1-4.

[18] M. Stojanovic and J. Preisig, *Underwater acoustic communication channels: Propagation models and statistical characterization*, in IEEE Communications Magazine, vol. 47, no. 1, pp. 84-89, January 2009.

[19] Avery Wang et al. *An industrial strength audio search algorithm*, Ismir, Vol. 2003, Citeseer 2003.

[20] Ainslie M. A., McColm J. G., *A simplified formula for viscous and chemical absorption in sea water*, Journal of the Acoustical Society of America, 103(3), 1671-1672, 1998.

[21] André Goalic et al. *Underwater acoustic communication using Reed Solomon Block Turbo Codes channel coding to transmit images and speech*, In: OCEANS 2010 MTS/IEEE SEATTLE, IEEE, 2010, pp. 16.

[22] R. Petroccia, J. Alves, G. Zappa, *Fostering the use of JANUS in operationally-relevant underwater applications*,IEEE Third Underwater Communications and Networking Conference (UComms), 2016, pp. 1-5.

[23] Mistry K., Modi H., *Design of High Data Rate and Multipath Efficient Underwater Acoustic Communication System Using OFDM–DQPSK*, In: Satapathy S., Joshi A., Modi N., Pathak N., Proceedings of International Conference on ICT for Sustainable Development. Advances in Intelligent Systems and Computing, vol 40, Springer, 2016, Singapore.

[24] Hai-Peng Ren, Chao Bai, Qingju Kong, Murilo S. Baptista, Celso Grebogi, *A chaotic spread spectrum system for underwater acoustic communication*, Physica A: Statistical Mechanics and its Applications, Volume 478, 2017, Pages 77-92, ISSN 0378-4371.