UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA MAGISTRALE IN
INGEGNERIA ELETTRONICA (DM 270/04)

"SVILUPPO E SPERIMENTAZIONE DI UN'APPLICAZIONE MUSICALE INTERATTIVA BASATA SU UNA TRASMISSIONE DATA OVER SOUND"

Relatore: Prof. Antonio Rodà

Laureando: Giulio Pitteri

ANNO ACCADEMICO 2022 – 2023

Data di laurea 07/09/2023

UNIVERSITÀ DEGLI STUDI DI PADOVA

SCUOLA DI INGEGNERIA
CORSO DI LAUREA MAGISTRALE IN INGENGERIA ELETTRONICA

Dipartimento di Ingegneria dell'Informazione

TESI DI LAUREA MAGISTRALE IN
INGEGNERIA ELETTRONICA
(Laurea magistrale DM 270/04)

Sviluppo e sperimentazione di un'applicazione musicale interattiva basata su una trasmissione "Data over Sound"

Relatore: Prof. Antonio Rodà

Laureando: Giulio Pitteri

ANNO ACCADEMICO 2022-2023

Indice

Indice analitico

Sommario

L'applicazione "Listen by Looking" è stata creata per rendere i concerti dal vivo di musica classica più accessibili a quella parte di pubblico che non ha le necessarie conoscenze musicali di base. In effetti, come per ogni forma d'arte, il pieno apprezzamento della musica richiede una certa competenza.
Rivolgendosi ad un pubblico eterogeneo, "Listen by Looking" presenta la musica in una forma grafica che tutti possono immediatamente apprezzare; la forma grafica si muove dinamicamente sincronizzata con la musica dal vivo e viene accompagnata da informazioni utili a comprendere il contesto del brano, come il periodo della partitura, le sezioni musicali e l'evidenziazione dei vari temi presenti nel brano stesso.

L'applicazione può essere utilizzata in una sala da concerto (questa modalità viene chiamata Live) ma anche a casa o a scuola in modalità Archivio. In quest'ultimo caso l'utente ascolta una registrazione musicale sincronizzata con lo spartito o con il Piano Roll.
Nel primo caso la rappresentazione musicale viene automaticamente sincronizzata con la musica dal vivo per mezzo di un segnale distribuito nella stanza tramite una normale scheda Wi-Fi o, in alternativa, utilizzando un algoritmo di score following.
Il segnale Wi-Fi è un beacon Wi-Fi inviato da un server che si comporta come un access point Wi-Fi trasmettendo continuamente una rete Wi-Fi fittizia e, senza entrare troppo nei dettagli, l'informazione di sincronizzazione necessaria all'applicazione è codificata all'interno del nome del beacon. Vale la pena notare che il dispositivo mobile non ha bisogno di negoziare un canale di comunicazione Wi-Fi bidirezionale con il server, evitando così problemi di scalabilità.

Questo utilizzo della trasmissione Wi-Fi è stato implementato e testato con successo su un dispositivo Android (4.4.2), ma a partire dalla versione 5 la piattaforma interpreta questo polling continuo delle reti Wi-Fi disponibili come un'attività dannosa e quindi inibisce il processo di scansione. Inoltre, dispositivi Android diversi presentano diversi tempi di scansione, variando da meno di un secondo a 3-4 secondi, quindi non c'è modo di impostare l'esatta frequenza dei messaggi consegnati per tutti i dispositivi raggiungibili.
Naturalmente questo problema può essere risolto instaurando una connessione Wi-Fi diretta, ma in una tipica sala da concerto (200 posti a sedere) questo richiederebbe la realizzazione di un'infrastruttura Wi-Fi ad-hoc, poichè gli access point commerciali non riescono a connettere più di 10-20 dispositivi contemporaneamente. La soluzione potrebbe essere quella di trasmettere i dati in modalità broadcast, in modo tale che i dispositivi non debbano negoziare un canale di comunicazione bidirezionale con il server. Purtroppo la maggior parte dei dispositivi Android non possiede l'hardware necessario per ricevere la trasmissione broadcast ad eccezione del microfono.

Quest'ultima osservazione ci ha spinto ad utilizzare il microfono per ricevere i dati mediante una tecnica di trasmissione nota come Data over Sound. Questa tipologia sperimentale di trasmissioni permette di inviare dei dati codificati nelle frequenze ultrasoniche a più ricevitori contemporanea-

mente, senza il bisogno di instaurare una connessione con ognuno di essi (broadcast) ne di avere con essi una connessione fisica (wireless).
L'obiettivo principale di questa trattazione è quello di definire le caratteristiche e lo sviluppo di un canale Data over Sound in tutte le sue parti, e viene strutturata nel modo seguente.

La sezione introduttiva riassume brevemente la storia dello sviluppo dell'applicazione "Listen by Looking", elencando le problematiche incontrate e illustrando le soluzioni adottate per risolverle, mentre la sezione successiva riepiloga quello che è lo stato dell'arte per quanto riguarda le trasmissioni DoS, in quali campi vengono applicate con successo, quali sono i progetti e gli standard attualmente attivi e in che modo essi influiscono sul percorso di ricerca corrente.
Successivamente viene illustrata la procedura seguita per definire il canale di trasmissione DoS in tutti i suoi aspetti, quali la scelta della banda di frequenze del segnale, la scelta dello schema usato per i simboli, la profondità di codifica per ogni simbolo e le tecniche disponibili per formare il segnale in trasmissione e per la decodifica da parte del ricevitore.

La sezione successiva riguarda l'implementazione pratica del modulo ricevitore e la sua struttura modulare in classi usata per aggiungere questa nuova funzionalità all'interno del codice dell'applicazione. Verrà anche accennato all'implementazione del ricevitore come applicazione stand-alone per scopi di testing e agli strumenti utilizzati per disporre di un trasmettitore impostabile parametricamente da un lato, e che funzioni senza la necessità di compilarlo in un programma nativo.
La sezione seguente illustra la valutazione e il testing del canale così implementato in vari ambienti di prova, misurandone le prestazioni in termini di bit error rate come funzione di diversi livelli del rapporto segnale-rumore. La sezione conclusiva infine fa il punto sulla trattazione svolta e propone alcuni sviluppi futuri riguardanti questo progetto.

Capitolo 1

Introduzione

L'apprezzamento e l'intrattenimento fornito dalla musica dipendono da una serie di fattori individuali. Anche assistendo allo stesso concerto, le persone tra il pubblico possono sperimentare livelli differenti di coinvolgimento a seconda della loro conoscenza pregressa dei brani, della loro capacità di fare attenzione alle peculiarità del linguaggio musicale (melodia, armonia, ritmo, contrappunto, timbro) e in generale sulle loro basi di conoscenza ottenute con l'educazione musicale.
La stessa esperienza può quindi risultare coinvolgente per alcuni, mentre può essere noiosa per chi non ha gli strumenti culturali per decodificarne il messaggio, e addirittura frustrante per chi non si sente sufficientemente competente in materia.

La tecnologia offre una varietà di strumenti per provare a colmare il divario tra utenti e contenuti. Gli studi sull'interazione uomo-macchina hanno sempre cercato di semplificare e rendere più intuitiva la comunicazione verso contenuti analogici e digitali attraverso la progettazione e la valutazione di nuove interfacce.
L'interazione naturale dell'utente [1], le interfacce tattili [2] e l'interazione basata sulla fisica [3] sono solo alcuni esempi in questo senso.

Ma cosa succederebbe se l'interfaccia fosse solo il tramite per apprendere e arricchire così l'esperienza dell'utente in uno scenario del mondo reale? In questo caso l'obiettivo non sarebbe solo progettare un'interfaccia intuitiva e facile da utilizzare, ma anche quello di creare contenuti e un sistema di interazione adatto per un contesto di apprendimento e di realtà aumentata.

Questa è una delle idee alla base dello sviluppo di "Listen by Looking"[1], un'applicazione musicale mobile interattiva orientata alla fruizione aumentata di musica dal vivo [4]. Lo spunto per sviluppare questa applicazione è partito dal M.° Battel, docente di pianoforte e concertista affermato[2], ed è stata sviluppata come progetto per il corso di "Programmazione di Sistemi Embedded" tenuto dal Prof. Fantozzi. Per questo, oltre alla naturale direzione del progetto verso la realtà aumentata, si è pensato su proposta del maestro di aggiungerci un'impronta per scopi didattici, ovvero prevedere di poter utilizzare l'applicazione anche come ausilio ai docenti per permettere agli studenti principianti di avvicinarsi con maggiore semplicità agli aspetti più tecnici del linguaggio musicale.

[1] `http://sourceforge.net/projects/listen-by-looking/`
[2] `http://www.giovanniumbertobattel.eu`

Proprio per rivolgersi ad un pubblico il più eterogeneo possibile, "Listen by Looking" presenta all'utente la musica in una forma grafica che mira ad essere più facilmente comprensibile; la forma grafica si muove in maniera dinamica in sincronia con la musica dal vivo ed è accompagnata da utili informazioni a corredo per comprendere il contesto del brano, come il periodo artistico nel quale si inserisce il brano, le diverse sezioni musicali in cui è diviso, e i vari temi (motifs) presentati nel corso della sua esecuzione.

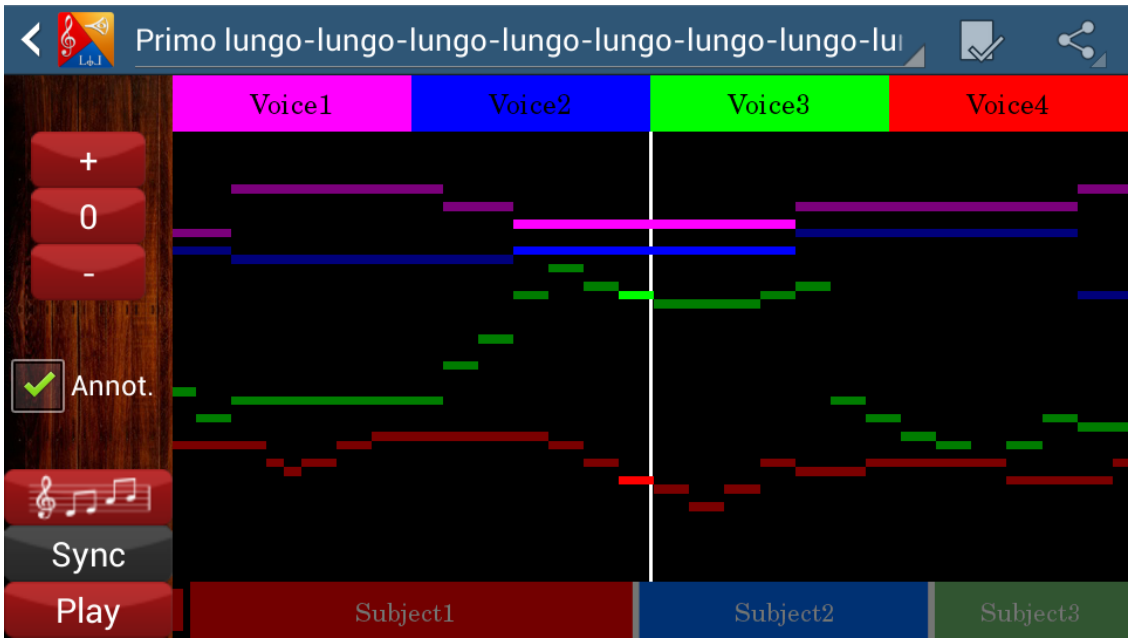

Figura 1.1: Schermata principale dell'applicazione con la visualizzazione a Piano Roll.

L'applicazione supporta due forme grafiche: uno spartito completo con tutte le note e le loro indicazioni ("Score") e una rappresentazione grafica intuitiva della durata e dell'altezza delle varie note con riferimento assoluto ("Piano Roll"). Questa seconda forma è stata denominata Piano Roll poichè ricorda i rotoli di carta perforata che pilotavano la meccanica dei pianoforti meccanici automatizzati di fine Ottocento. Questa rappresentazione è stata introdotta nell'applicazione al fine di aiutare l'ascolto del brano per quegli utenti che non possiedono le conoscenze necessarie per leggere (seguire) una partitura musicale.

L'applicazione può essere utilizzata in modalità "Live" mentre si ascolta la musica in una sala da concerto, oppure in modalità "Archivio" utilizzando le registrazioni dei brani scaricate dall'applicazione stessa e salvate nella memoria del dispositivo mobile.
Il codice software è stato scritto in ambiente Android, scelta dettata principalmente dalla semplicità di installazione e utilizzo dell'ambiente stesso (SDK) e dalla caratteristica open source del sistema Android in sè, con la penalizzazione di non poter possibilmente garantire il corretto funzionamento dell'applicazione su tutti i dispositivi Android esistenti.
Questa app inoltre permette di selezionare e scaricare da un server dedicato i brani interessati da un dato evento, in modo che l'utente volenteroso possa ascoltarli e seguirli a video prima e dopo l'evento live.

Lo scoglio principale per la visualizzazione dello spartito incontrato in fase di sviluppo è stato rappresentato dall'inesistenza in campo musicale di un formato informatico di codifica dei dati considerato lo standard de facto. La soluzione adottata è stata quella di costruire un formato ad-

hoc e di programmare le parti dell'applicazione addette alla sua interpretazione e al suo disegno. Questa scelta, per quanto molto onerosa in termini di tempo e fatica, permette di mantenere tutta la libertà offerta da un formato descrittivo senza perdere alcune caratteristiche predefinite dello spartito in questione, come il suo layout globale.

L'applicazione "Listen by Looking" deve visualizzare ciò che avviene sullo schermo in maniera sincronizzata con i vari eventi musicali. L'allineamento in tempo reale dell'esecuzione musicale dal vivo con le rappresentazioni visive tramite spartito e piano roll viene ottenuto tramite un modulo software chiamato score follower [5, 6]. La funzione principale dell'allineamento musica-partitura è quella di identificare, per ogni intervallo temporale di un'esecuzione, sia essa registrata piuttosto che eseguita dal vivo, la posizione corrispondente su una rappresentazione simbolica della partitura che viene eseguita. Nel caso che l'applicazione stia operando in modalità live viene calcolato un allineamento locale ma il sistema dovrebbe essere in grado di adattarlo, anche significamente, nel momento in cui sono disponibili nuove informazioni.

Da questo punto di vista si incontrano le prime difficoltà implementative, dal momento che generalmente gli algoritmi di score following richiedono una quantità di risorse (memoria e capacità di calcolo) da parte dell'elaboratore sul quale sono caricati che la maggior parte dei dispositivi mobili non riesce semplicemente a fornire.
Consci di questo problema abbiamo pensato ad una possibile alternativa: l'algoritmo di score following, una volta che viene caricato e risulta operativo, ha solo bisogno di ricevere la musica in ingresso per produrre le informazioni di allineamento, e dunque ha poca importanza su quale macchina viene installato, fintanto che riesce in qualche modo a consegnare le informazioni al dispositivo con l'applicazione aperta.

Dunque la soluzione adottata è stata quella di caricare l'algoritmo di score following su di un server (workstation) in modo che potesse generare autonomamente i dati di sincronizzazione sulla base della musica suonata. Tuttavia il problema successivo in questo caso riguardava proprio le modalità di consegna di queste informazioni a una moltitudine di dispositivi in modo più o meno contemporaneo, ovvero senza un delay troppo elevato tra la ricezione del primo e quella dell'ultimo dispositivo.

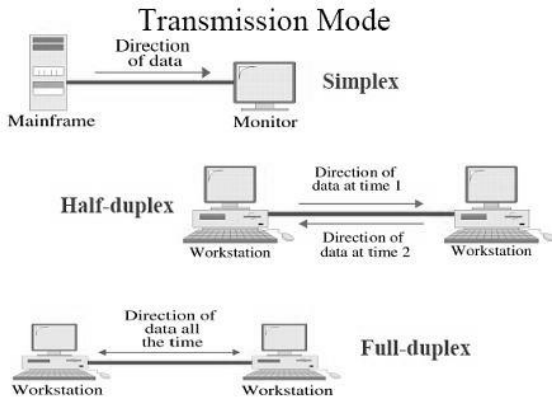

Figura 1.2
Differenza tra le trasmissioni in broadcast (simplex) e quelle complete (duplex).

Nondimeno questa trasmissione dovrebbe idealmente avvenire in modalità broadcast, senza che i dispositivi debbano preventivamente instaurare una connessione col server (handshaking) e senza che il server attenda da tutti una risposta di avvenuta consegna (acknowledgment).

Per questa nuova problematica la soluzione adottata è stata quella di servirsi della tecnologia Wi-Fi, presente in praticamente tutti i dispositivi mobili abbastanza recenti, e di sfruttarla nel modo seguente: un access point programmabile (nella pratica un notebook) viene utilizzato in modo che invii continuamente il beacon relativo ad una rete Wi-Fi fittizia, essendo l'informazione da trasmettere contenuta nel nome stesso della rete, mentre i dispositivi mobili scansionano continuamete l'ambiente al-

la ricerca di nuove reti disponibili, e quando trovano la rete fittizia estraggono l'informazione utile estraendola dal suo nome.
Purtroppo questa soluzione non risulta abbastanza robusta poichè le versioni più recenti di Android interpretano questo tipo di scasione ripetuta (polling) come l'effetto della presenza di un virus e dunque la inibiscono senza nessuna segnalazione ne all'utente ne allo sviluppatore. Esiste tuttavia un altro modulo hardware presente in tutti i dispositivi mobili che per sua natura risulta adatto a ricevere segnale in modalità brodcast, ovvero il microfono.

Da quest'ultima considerazione nasce l'idea per questo progetto, ovvero la necessità di sostituire la trasmissione utilizzando il Wi-Fi viene soddisfatta utilizzando una trasmissione Data over Sound progettata e sviluppata esattamente per questo scopo. Le trasmissioni Data over Sound (DoS) sono un campo di ricerca relativamente recente, in quanto in passato non è mai emerso uno scenario realistico nel quale questo tipo di trasmissioni potessero essere utilizzate con successo per migliorare o risolvere una problematica reale legata allo scenario stesso.
Nella maggiorparte dei casi infatti le trasmissioni elettromagnetiche standard (guidate in cavo o wireless) erano sufficientemente mature per soddisfare tutte le esigenze senza causare ripercussioni significative sull'ambiente o sugli operatori. In tempi recenti tuttavia sono emersi scenari nei quali le trasmissioni convenzionali non possono essere utilizzate, sia per motivi legati alla sicurezza dell'ambiente di lavoro, sia legati all'impossibilità fisica di trasmettere con successo un segnale a radiofrequenza.

Nel caso dell'applicazione "Listen by Looking" un canale di trasmissione DoS risponde ad entrambe le caratteristiche principali rischeste dallo scenario: le trasmissioni DoS sono per loro natura utilizzabili al meglio in modalità broadcast, per motivi legati al mezzo del canale (aria) e alle frequenze disponibili (audio), inoltre un canale DoS può essere istituito senza il bisogno di procurare hardware particolare (trasduttori, convertitori, pickup, etc.) e dunque può essere applicato senza particolari sforzi a una larga varietà di scenari del mondo reale. Nel nostro caso l'unico hardware richiesto dal ricevitore è già presente dei dispositivi mobili (microfono), mentre l'hardware relativo al trasmettitore si riduce ad un altoparlante collegato all'uscita di un normale computer.

Capitolo 2

DoS: lo stato dell'arte

Nella storia delle trasmissioni dati non sono praticamente mai state scelte le onde acustiche come prima soluzione per instaurare un canale di comunicazione tra due dispositivi [7] con una notevole eccezione, ovvero i sistemi di modulazione e demodulazione (modem) appoggiati alle linee telefoniche analogiche per la connessione ad Internet.
Questi dispositivi permettevano ad un elaboratore di accedere alla rete contattando in pratica un server dedicato (messo a disposizione dall'operatore di rete) sostanzialmente effettuando una chiamata acustica. In questo modo il segnale che conteneva l'informazione in ingresso all'elaboratore era un segnale audio[1] e il modem effettuava una decodifica (demodulazione) per ricavare il dato dalla forma d'onda acustica. Naturalmente le comunicazioni tra il server ed il resto dei nodi della rete rimanevano elettriche (a banda larga) ma l'ultimo miglio di comunicazione (soprattutto negli elaboratori ad uso domestico) è rimasto acustico e analogico (doppino di rame) per molto tempo.

Esiste poi un esempio interessante dal punto di vista scientifico/culturale di una codifica di dati (immagini) attraverso un segnale acustico, e si tratta delle immagini salvate nei dischi dorati lanciati assieme alle due sonde Voyager nel 1977[2]. L'idea alla base di questi dischi era che, nella (remota) possibilità che fossero stati recuperati integri da una qualche civiltà aliena tecnologicamente abbastanza matura (equazione di Drake), fosse stato possibile decodificare le informazioni contenute in essi solo mediante delle altre informazioni (indizi) incisi sulla loro custodia.
Tali indizi sono basati unicamente su costanti fisiche/chimiche ritenute universali e su una mappa delle pulsar nelle vicinanze cosmiche della terra, mentre i dati relativi alle immagini sono contenuti in una traccia audio fisicamente incisa sul disco, esattamente come avviene per la musica incisa su un LP, e gli indizi contengono anche le informazioni per la loro decodifica e visualizzazione (uno degli indizi è il risultato dell'immagine di test).
Naturalmente è possibile che una tale civiltà aliena possa essere tecnologicamente estremamente avanzata e comunque non possedere i concetti per la decodifica (analogica) delle immagini, e sembra che questa eventualità possa non essere così remota , come dimostrerebbe un recente esperimento effettuato chiedendo a degli studenti di informatica di decifrare il contenuto dei golden records senza nessun tipo di aiuto esterno.

[1] Sarebbe più corretto dire un segnale elettrico in banda audio, anche se i primissimi modem erano letteralmente degli accoppiatori acustici con tanto di cornetta.

[2] `https://voyager.jpl.nasa.gov/golden-record/`

In tempi recenti tuttavia sono emersi nuovi scenari nei quali le soluzioni di comunicazione mediante radiofrequenza (o comunque onde elettromagnetiche in generale) non sono utilizzabili, o per le caratteristiche fisiche dell'ambiente (per esempio sott'acqua) oppure per motivi legati a questioni di sicurezza (in ambienti atex, ovvero legati al disinnesco di esplosivi). Inoltre sono presenti alcuni casi in cui una trasmissione di tipo acustico accoppiata ad una tradizionale può migliorare la sicurezza della comunicazione (one-time passwords) [8] oppure come strumento per firmare digitalmente contenuti multimediali (watermarking) [9] e anche come supporto ai sistemi di pagamento mobile [10].

2.1 Le trasmissioni sottomarine

Un classico esempio di ambiente nel quale le trasmissioni tradizionali attraverso onde elettromagnetiche sono problematiche é l'ambiente marino. In particolare l'acqua di mare, che presenta mediamente circa 30 grammi per litro di sali disciolti, risulta un mezzo particolarmente sfavorevole per quanto riguarda l'attenuazione alle frequenze radio comunemente utilizzate in aria, come del resto risulta in tutti i canali di trasmissione in cui é presente un composto ionico (alta conducibilitá). Per dare un'idea della differenza di attenuazione, considerando un normale radiocomando alla frequenza di 40MHz e potenza di emissione nell'ordine dei Watt, un ricevitore immerso in acqua dolce riceve il segnale fino a circa 3 metri, mentre lo stesso ricevitore immerso in acqua di mare riceve il segnale solo fino a circa 30 cm [11].

É quindi evidente che le trasmissioni tradizionali in acqua vengono impiegate solamente a corto raggio, mentre nelle distanze piú lunghe vengono adottate diverse soluzioni alternative dove non é possibile utilizzare le trasmissioni via cavo. Una di queste si basa sull'impiego di trasmettitori e ricevitori ottici: in questo scenario il trasmettitore é composto da uno stadio ottico che immette segnale nel mezzo, di solito utilizzando luce laser o LED, e da un ricevitore ottico che preleva il segnale dal canale e lo decodifica, di solito un fotorivelatore o un fotodiodo. Il vantaggio di questa soluzione risiede principalmente nel fatto che, rispetto all'attenuazione nelle frequenze radio dovuta ai sali, nella banda ottica del visibile tale attenuazione risulta pressoché trascurabile, tuttavia l'utilizzo di trasmissioni luminose presenta alcuni svantaggi non presenti nelle trasmissioni a radiofrequenza, uno su tutti la necessitá dell'allineamento visivo tra trasmettitore e ricevitore (line of sight).

2.1.1 Le bande VLF e ULF

Un'alternativa alle trasmissioni ottiche per coprire lunghe distanze senza il problema del puntamento diretto trova risposta nell'utilizzo delle onde elettromagnetiche a bassa frequenza, in particolare nelle bande chiamate VeryLowFrequency (3-30 kHz) e UltraLowFrequency (300-3000 Hz).
L'idea principale dietro alla scelta di queste frequenze si basa sull'osservazione che il coefficente di assorbimento delle onde elettromagnetiche in un mezzo dipende fortemente dalla frequenza delle onde elettromagnetiche stesse, in prima approssimazione seguendo un andamento proporzionale all'inverso della pulsazione.

Dunque abbassare la frequenza di trasmissione significa ridurre il coefficente di assorbimento e quindi permettere una trasmissione su distanze piú lunghe a paritá di potenza di emissione, e non a caso questa soluzione viene comunemente adottata nelle comunicazioni tra i sommergibili in immersione e le stazioni di terra delle marine militari di alcuni stati (tra i quali Stati Uniti, Russia, India e Cina).

Figura 2.2: Foto aerea della stazione di trasmissione ELF della marina statunitense a Clam Lake, nel Wisconsin. In basso a sinistra si intravede l'inizio dei due bracci del dipolo.

Si tratta tuttavia di una soluzione mirata e circoscritta alla trasmissione di dati semplici principalmente per due motivi: il primo deriva dal fatto che, sebbene ridurre la frequenza di esercizio permetta una minore attenuazione del segnale e quindi una maggiore distanza di trasmissione, si ottiene anche una sensibile riduzione della quantitá di dati che il segnale puó portare.

Questo puó in realtá essere un compromesso accettabile in molte applicazioni dove lo scambio di informazioni viene limitato a brevi token testuali, senonché interviene il secondo motivo limitante e riguarda le dimensioni fisiche di trasmettitore e ricevitore. Infatti sebbene abbassando la frequenza di esercizio viene ridotto il tasso di trasmissione (bitrate) ottenendo in cambio una maggiore distanza di ricezione, anche le dimensioni delle antenne scalano in modo proporzionale alla lunghezza d'onda (ovvero inversamente proporzionale alla frequenza).

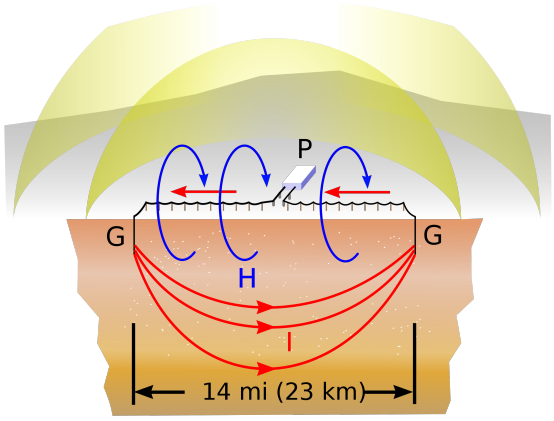

Figura 2.1
Schema di un'antenna dipolo usata per trasmettere onde ELF, dove parte del circuito elettrico é costituito dal percorso in terra.

In figura 2.1 viene mostrata la tipica struttura di un'antenna dipolo usata in una stazione trasmetti-

trice ELF, e risultano subito evidenti le dimensioni orizzontali dei due bracci del dipolo, e da questa considerazione risulta anche evidente perché questo tipo di trasmissione viene principalmente utilizzata in una sola direzione (broadcast), e quindi l'impiego piú frequente di questo sistema risulta quello di segnalare ai sommergibili in immersione di emergere per ricevere altre comunicazioni utilizzando i sistemi tradizionali.
É utile far notare inoltre come, prendendo a riferimento la figura 2.1 e ipotizzando una frequenza di trasmissione di 300 Hz, la lunghezza ideale del dipolo dovrebbe corrispondere a mezza lunghezza d'onda, ovvero nell'ordine delle centinaia di kilometri, e quindi una struttura come quella di figura risulta un dipolo molto poco efficiente con la conseguenza che solo una piccola parte della potenza di alimentazione si riversa nel segnale trasmesso, e quindi nella pratica ogni stazione di trasmissione ELF ha bisogno anche di un'infrastruttura di alimentazione dedicata.
Nella figura 2.2 viene mostrata per riferimento una foto della stazione di trasmissione ELF della marina statunitense di Clam Lake, costruita nel 1982 e dismessa nel 2004.

2.1.2 Deep Siren

Dal momento che le comunicazioni a radiofrequenza sono problematiche in ambiente sottomarino, molti progetti sono nati cercando di risolvere il problema delle comunicazioni con i sottomarini mediante le onde acustiche. Uno di questi prende il nome di Deep Siren ed è stato sviluppato dall'industria Raytheon in collaborazione con la marina degli Stati Uniti.
Si tratta di un sistema di chiamate tattiche (tactical paging) per mezzo di boe acustiche posizionate sulla superficie del mare, ed ogni boa viene contattata in modo tradizionale da un satellite con un messaggio tattico criptato che poi viene trasmesso al sottomarino senza bisogno di estendere l'antenna o di risalire ad altezza periscopio, dal momento che ogni boa estende un trasduttore acustico dal lato di immersione che inietta il segnale audio in acqua [12] [13].

Questo sistema di comunicazione permette di contattare i sottomarini in movimento anche a distanze superiori alle 50 miglia nautiche, mentre le boe comunicano tra di loro e con la stazione base attraverso un collegamento satellitare Iridium, e possono essere schierate sia da una stazione di terra che da una nave o da un mezzo aereo.

Le boe una volta schierate fanno una stima delle condizioni locali del mare (salinità, temperatura, etc.) e valutano qualè la profondità ideale per piazzare il trasduttore acustico, dopodichè si mettono in standby in attesa di comunicazioni dal satellite, con un'autonomia stimata di circa tre giorni in attesa e di due ore di trasmissione a piena potenza.
La codifica di canale è propietaria a bassa frequenza (udibile) progettata per massimizzare la probabilità di ricezione e supporta l'invio di più tipi di messaggi oltre ad una funzione di posizionamento mediante ping. Questo segnale di ping è un tono puro, con frequenza, durata e ripetizione decisi in fase di invio dalla stazione base, e viene utilizzato per rilevare le boe attive, selezionare da quale boa inviare i messaggi e chiederne lo stato e la posizione (mediante GPS). Può anche essere usato eventualmente per far saltare (scuttle) la boa se necessario.

Dal lato del sottomarino viene installato uno specifico ricevitore per decodificare i messaggi acustici, venendo collegato direttamente all'array sonar presente a bordo e funzionando in maniera completamente autonoma, avvisando eventualmente l'equipaggio dell'arrivo di una nuova comunicazione.
La parte software dell'impianto viene fornita da RRK Technologies, mentre le boe e il sistema di lancio sono sviluppati da Ultra Electronics Maritime Systems.

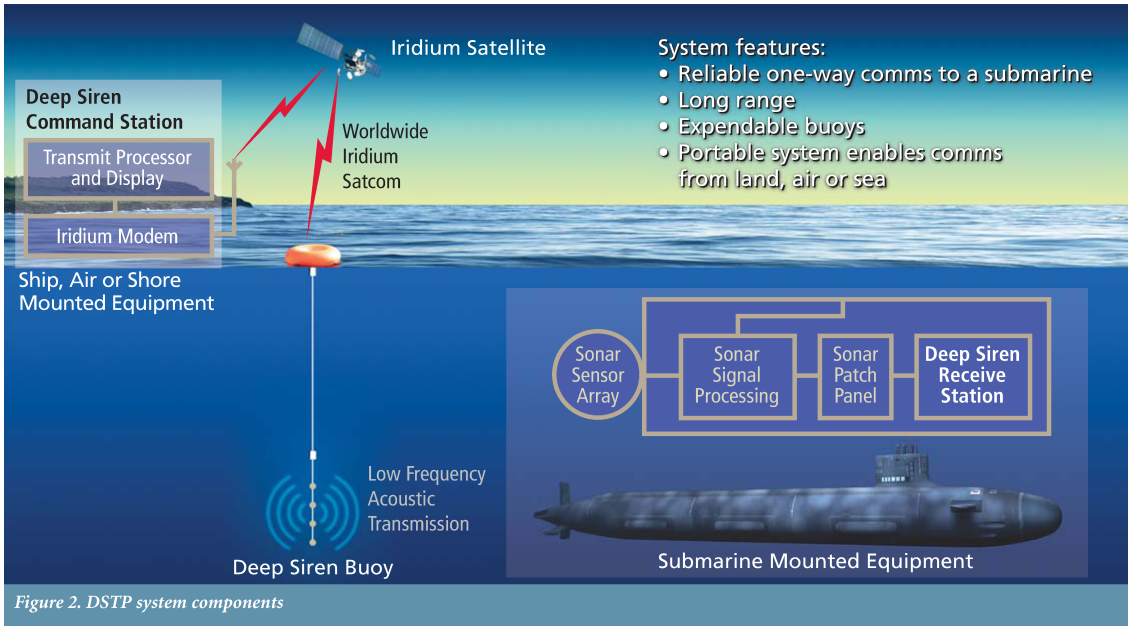

Figura 2.3: Schema del sistema di trasmissione sottomarino Deep Siren.

2.1.3 MIT TARF

Il progetto Deep Siren appena illustrato permette ad un sistema di comunicazione tradizionale di inviare messaggi ad un sottomarino in immersione, ma non consente al sottomarino stesso di inviare una risposta o di iniziare una trasmissione verso la stazione di terra.
Per provare a rispondere a questa problematica il gruppo di ricerca MIT Media Lab ha sviluppato un progetto che sfrutta le comunicazioni di dati attraverso le onde acustiche senza bisogno di un sistema ponte tra l'ambiente marino e l'ambiente in aria.

Tale progetto si chiama TARF, ovvero Translational Acoustic-RF Communication, e presenta una soluzione basata sulla rilevazione delle vibrazioni che la trasmissione DoS sottomarina provoca sulla superficie del mare [14].
Lo scenario è il seguente: un mezzo sottomarino dirige il segnale dati verso la superficie verticalmente, che impattando con l'interfaccia crea piccoli spostamenti verticali del livello dell'acqua. Questi spostamenti poi vengono valutati da un sistema radar che valuta il cambiamento della distanza dalla superficie dell'acqua attraverso lo sfasamento di un segnale radio ad alta frequenza.

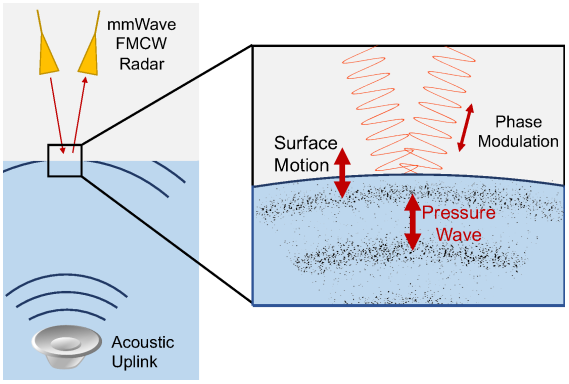

Figura 2.4
Schema rappresentativo del funzionamento del sistema TARF.

Utilizzando questo sistema i sottomarini possono comunicare con i velivoli in aria senza bisogno di emergere e senza bisogno di asset aggiuntivi sull'interfaccia acqua/aria, inoltre in ambito civile può essere applicato alle stazioni di ricerca scientifica dell'ecosistema marino che possono inviare i dati raccolti senza bisogno di essere fisicamente spostate.

Un'ulteriore applicazione può essere quella di applicare il trasmettitore DoS a bordo di mezzi che possono essere dispersi in mare (come gli aerei) in modo che funzionino come segnalatori (beacon), e le operazioni di ricerca possono essere condotte a mezzo di un velivolo che con il radar del TARF ispeziona la superficie del mare per rilevare le vibrazioni del segnale inviato dal beacon.

La parte del trasmettitore sottomarino viene implementata con un altoparlante standard che invia il segnale sonar, che quando raggiunge la superficie ne provoca piccoli spostamenti verticali (dell'ordine di pochi micron). Per raggiungere elevati bitrate il trasmettitore invia segnali a più frequenze contemporaneamente attraverso uno schema di modulazione di tipo OFDM (che verrà ripreso in seguito).
Le vibrazioni vengono poi misurate attraverso una tipologia di radar in grado di processare segnali radio a frequenze elevatissime (tra 30 e 300 GHz) inviando tale segnale direttamente sulla superficie del mare che lo riflette, e la cui fase viene misurata dal radar stesso, di modo che l'oscillazione della fase misurata dal radar contenga l'informazione delle frequenze inviate dal sonar sottomarino.

Naturalmente uno dei problemi incontrati durante lo sviluppo di TARF è che normalmente la superficie del mare non è ferma ma oscilla anch'essa: in condizioni di mare molto calmo l'ampiezza delle onde marine si attesta su pochi centimetri, ovvero molto maggiori rispetto alle vibrazioni indotte dal sonar.
Tuttavia la frequenza di questo ripple si aggira attorno a qualche Hertz e, con un'accorta selezione in frequenza da parte del radar, è possibile distinguere le due tipologie di oscillazioni.

2.1.4 WHOI Nereus

Un altro campo di interesse per le trasmissioni DoS riguarda i veicoli di ispezione sottomarina (ROV, Remotely Operated Vehicle) che in molti casi, date le profondità alle quali devono operare, diventano veicoli pressochè autonomi (AUV, Autonomous Underwater Vehicle).
Questi sono letteralmente dei sommergibili in miniatura robotizzati (unmanned, senza equipaggio) che sono progettati per raggiungere profondità di diverse migliaia di metri, tali da non permettere la sopravvivenza di un pilota umano.

Di solito ad un ROV viene associato un sistema di lancio e recupero chiamato LARS (Launch And Recovery System) e un TMS (Tether Management System) che serve a gestire il tether (ossia l'ombelicale del ROV) in profondità. Tale cavo di comunicazione (ombelicale) serve per trasferire informazioni da e verso il veicolo e può essere sia tradizionale (per flussi di informazione moderati) oppure in fibra ottica (per flussi audio e video ad alta definizione).
Naturalmente la navigazione per questi veicoli risulta alquanto problematica dovendo sempre rimanere attaccati ad un cavo con la nave di ricerca (per esempio ogni sporgenza naturale diventa un rischio di perdita del veicolo) e dunque lo sforzo di sviluppo in questo caso cerca di trovare delle alternative di comunicazione wireless con queste macchine.

Un parziale successo da questo punto di vista arriva dal gruppo di ricerca del Woods Hole Oceanographic Institution (WHOI) che hanno sviluppato un nuovo sistema di comunicazione basato sull'interazione di due tipi di trasmissione in parallelo, uno acustico (DoS) e uno luminoso (laser).
Tale sistema è stato installato a bordo del loro veicolo sottomarino a batteria Nereus, che può operare sia in maniera pilotata da remoto (ROV) che in maniera completamente autonoma (AUV)[3].
Questo veicolo è stato inizialmente progettato per scandagliare il fondale marino e prelevare cam-

[3] `https://web.whoi.edu/hades/nereus/`

pioni di suolo dalle fosse oceaniche profonde anche oltre gli 11000 metri come l'abisso Challenger, e per ottenere questo viene normalmente controllato con un cavo in fibra sottile che deve essere buttato dopo un singolo utilizzo [15].

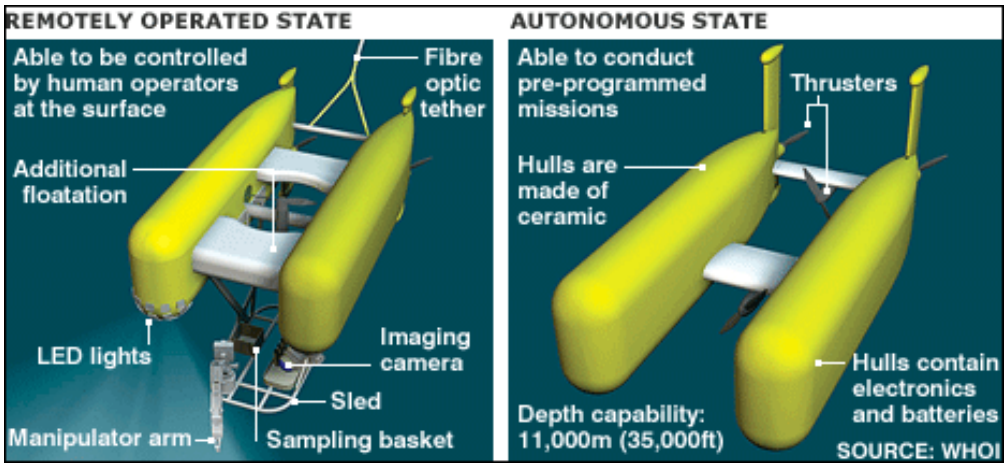

Figura 2.5: Immagine del veicolo sottomarino Nereus dove sono illustrate le due modalità di funzionamento.

La soluzione adottata dal WHOI combina le caratteristiche delle trasmissioni DoS, che in acqua riescono ad inviare segnale anche a grandi distanze ma con bitrale limitati, con le trasmissioni ottiche, che presentano bitrate molto elevati ma su distanze inferiori ai 100 metri.
Il veicolo Nereus è poi capace di passare da un tipo di trasmissione all'altro in maniera autonoma e adattiva, in modo da non dover interrompere le operazioni programmate e contemporaneamente rimanere in contatto con la nave di controllo [16].

Il sistema è stato testato nelle acque di Guam, dove il team di ricerca ha immerso il Nereus a 700 metri di profondità controllandolo acusticamente, e tracciando la sua posizione attraverso un sistema di tracking Sonardyne. Dopodichè il team ha calato in acqua un trasmettitore ottico attaccato alla nave con un cavo in fibra, e quando ha superato la distanza massima dal veicolo, questo ha iniziato a trasmettere video ad alta definizione agli operatori sulla nave di controllo, che hanno potuto comandare un braccio robotico del veicolo sempre per mezzo del controllo acustico.
Il team di sviluppo ritiene che l'utilizzo di questo veicolo con questo tipo di controllo possa essere impiegato con successo in molti scenari, in particolare dall'industria del gas e del petrolio, che utilizzano i veicoli ROV per monitorare lo stato dei pozzi sottomarini e dei dispositivi installati sul fondo degli oceani.

2.2 Progetti e Standard esistenti

Finora sono state illustrate alcune soluzioni che sfruttano il sistema DoS in ambito militare e di ricerca oceanografica, ma queste soluzioni sono state portate avanti in maniera indipendente e ognuna di esse adotta delle soluzioni ad-hoc, di solito in forma chiusa ovvero senza divulgare i dettagli implementativi del sistema di comunicazione.
Tuttavia, con la previsione che le comunicazioni attraverso DoS continueranno ad aumentare di pari passo con l'aumento dei loro scenari applicativi, emerge la necessità da un lato di uniformare le scelte implementative e dall'altro di regolamentare alcuni dettagli riguardanti i segnali utilizzati (banda utile, codifiche di canale, rumore ammesso fuori banda, etc.).

Detto in altri termini, in futuro sarà sempre più pressante la richiesta di avere uno standard per le trasmissioni DoS in ambito civile (commerciale) unito al bisogno da parte degli sviluppatori di avere a disposizione degli strumenti già predisposti (librerie) e delle linee guida (reference) per la creazione di applicazioni in ambito DoS.

2.2.1 JANUS

Un primo passo verso questa direzione è rappresentato dal protocollo JANUS. È stato sviluppato in uno dei centri di ricerca NATO situato a La Spezia, il Centro per la Ricerca e la Sperimentazione Marittima (Centre for Maritime Research and Experimentation, CMRE) ed è ufficialmente diventato nel 2017 uno standard riconosciuto a livello internazionale, il primo nel suo genere [17].

Il protocollo definisce le modalità di codifica delle informazioni in una forma d'onda acustica in modo che la strumentzione esistente possa risultare conforme al protocollo stesso senza particolari difficoltà [18].
È stato chiamato così in onore della divinità romana Giano, il custode delle porte e dei passaggi, in quanto questo protocollo apre la strada a diversi paradigmi di utilizzo delle comunicazioni DoS permettendo l'interoperabilità tra svariati dispositivi, militari e civili, NATO e non NATO.

Nello stesso modo come è avvenuto, non molti anni fa, con il protocollo Wi-Fi, lo standard JANUS ha le potenzialità per fare da traino a d investimenti tecnologici ed aumentare l'interesse di sviluppo verso le comunicazioni tra dispositivi sottomarini, creando potenzialmente una nuova rete di dispositivi interconnessi tra di loro e con il mondo esterno (Internet of Underwater Things).

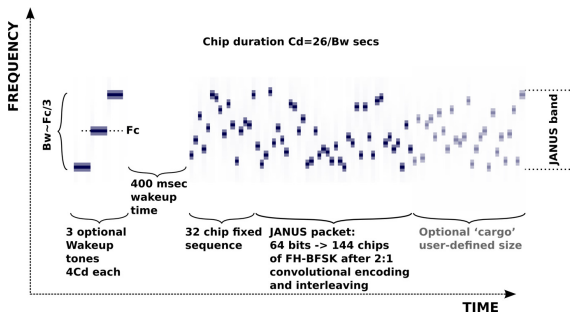

Figura 2.6
Schema delle fasi di trasmissione acustica mediante protocollo JANUS.

Se adottato su larga scala, questo protocollo permette di mettere fine alla attuale situazione di "Torre di Babele" tra i vari modem e sistemi sviluppati da diversi produttori e nazioni, e per questo fin dall'inizio il processo di definizione di JANUS è stato il più inclusivo possibile, coinvolgendo università, industrie e agenzie governative, oltre a vari esperti del settore delle telecomunicazioni digitali.

I test del protocollo JANUS sono stati condotti negli ultimi anni dal CMRE a bordo della nave da ricerca Alliance, utilizzando la rete LOON (Littoral Ocean Observatory Network), che facilita gli esperimenti condotti attraverso robot marini dal team di ricerca creando una rete acustica di monitoraggio ottenuta posizionando dei tripodi contenenti gli strumenti di comunicazione sottomarina sul fondo del mare, ma comunque accessibili dagli operatori anche da remoto.

2.2.2 Sonitalk

Sulla falsariga di quanto fatto con il protocollo JANUS si inserisce un altro progetto orientato alla pubblicazione di uno standard per le trasmissioni DoS. Questo progetto prende il nome di SoniTalk ed è stato sviluppato dal gruppo di Media Computing Research dell'università delle scienze applicate di St. Poelten (Austria).

Codice 1 Esempio di messaggio di stato JSON descritto dal protocollo JANUS

```
{
   "ClassUserID":0,
   "ApplicationType":3,
   "Nationality":"PT",
   "Latitude":"38.386547",
   "Longitude":"-9.055858",
   "Depth":"16",
   "Speed":"1.400000",
   "Heading":"0.000000",
   "O2":"17.799999",
   "CO2":"5.000000",
   "CO":"76.000000",
   "H2":"3.500000",
   "Pressure":"45.000000",
   "Temperature":"21.000000",
   "Survivors":"43",
   "MobilityFlag":"1",
   "ForwardingCapability":"1",
   "TxRxFlag":"0",
   "ScheduleFlag":"0"
}
```

Lo scopo principale di questo progetto è quello di sviluppare un protocollo aperto (open protocol) per le comunicazioni ultrasoniche tra dispositivi che preservi completamente la sicurezza delle informazioni trasmesse. Ad oggi le soluzioni presenti sul mercato sono pressochè propietarie e in parte sollevano alcuni dubbi riguardo la protezione della comunicazione stessa.
Manca tuttora un protocollo aperto e trasparente per questo tipo di comunicazioni, in aggiunta al fatto che spesso le soluzioni propietarie non sono compatibili tra di loro.

Il progetto SoniTalk mira allo sviluppo di un protocollo aperto e trasparente per le comunicazioni DoS tra dispositivi quali smartphones, televisori e dispositivi IoT in generale. Fornisce un canale di comunicazione protetto (criptato) e quindi risulta il punto di partenza per applicazioni e servizi innovativi, quali l'autenticazione sicura (a due fattori), pagamenti attraverso piattaforma mobile, controllo di impianti di domotica, e molti altri [19].
Dunque questo protocollo ha le potenzialità per diventare un open standard nelle comunicazioni ultrasoniche fornendo una comunicazione protetta e aprendo la strada a molti nuovi scenri applicativi.

2.2.3 Applicazioni mobili

Sebbene le trasmissioni DoS siano state quasi sempre impiegate in ambito di ricerca o in campo militare (sottomarini), in tempi recenti sono emersi alcuni progetti di sviluppo di applicazioni per piattaforme mobili, che per effettuare le operazioni di comunicazione usano un canale DoS sfruttando l'hardware in dotazione a praticamente tutti i dispositivi disponibili sul mercato [20].

AquaApp

Un esempio di queste applicazioni si chiama AquaApp ed è stata sviluppata da un gruppo di studenti della scuola di ingegneria dell'università di Washington. Si tratta di un'applicazione orientata a chi pratica immersioni (scuba) per permettere lo scambio di messaggi tra gli smartphones di due o più sub mentre si trovano sott'acqua [21].

A differenza dei progetti illustrati in precedenza, che per il loro funzionamento richiedono dispositivi hardware dedicati e progettati per il loro scopo specifico (trasduttori, microfoni, etc.), questa applicazione si propone di funzionare sui comuni dispositivi mobili in commercio, e da questo punto di vista si può considerare una soluzione completamente software.
Il sistema di comunicazione è stato sviluppato in modo da adattarsi alle differenti caratteristiche in frequenza dei vari dispositivi target, così come ai cambiamenti del segnale ricevuto dovuto alla propagazione multipla (multipath), alla variabilità delle condizioni del canale e alle varie condizioni di rumore.

Il sistema è stato testato in sei diversi ambienti del mondo reale, a profondità tra i 2 e i 15 m e in presenza di imbarcazioni in movimento e pescatori, e i risultati mostrano che riesce ad aggiustare la banda utile del segnale in tempo reale e raggiungere bitrate da 100 bps a 1.8 kbps e distanze di ricezione fino a 30 m, distanza che può essere portata fino a 100m riducendo il bitrate a 10-20 bps. Dal momento che gli smartphones e gli smart watches vengono sempre più spesso utilizzati in ambiente subacqueo, questa soluzione completamente software ha le potenzialità di permettere a qualunque sub con un dispositivo mobile di scambiare messaggi durante un'immersione, per esempio quando la distanza tra due sub risulta tale da non permettere l'utilizzo della cominicazione visiva (hand gestures). Inoltre, essendo un'applicazione open source, il suo sviluppo rimane aperto anche a contributi di sviluppatori esterni[4].

Attendance Book

I tag acustici ad alta frequenza sono spesso impiegati a scopo pubblicitario, ma anche come misuratori di posizionamento e tracciamento. La API Nearby Messages di Google[5] e i beacon di posizionamento di Starbucks sono entrambe applicazioni DoS ultrasoniche ben note tra gli utenti e gli sviluppatori, tuttavia questi sistemi sono poco adatti a trasmettere dati testuali poichè sono inclini a subire i disturbi sonori circostanti, oltre ad avere delle prestazioni fortemente dipendenti da quelle dei convertitori disponibili (ADC o DAC).

L'applicazione Attendance Book si propone di funzionare esattamente come un beacon ultrasonico per inviare messaggi testuali in broadcast utilizzando un controllo di sequenza e dei codici di correzione d'errore, realizzando il comportamento di un registro delle presenze [22].
In questo scenario il dispositivo client riceve in broadcast un messaggio di testo JSON, codificato in UTF-8, ne elabora il contenuto e successivamente informa il server della posizione corrente del partecipante (attendee).

I test hanno mostrato che almeno 5 dispositivi riescono ad elaborare contemporaneamente il messaggio ricevuto e che la trasmissione viene garantita a distanze fino a 3 metri dal beacon, e questo dimostra che le comunicazioni DoS possono essere impiegate con successo in un vasto campo di applicazioni quando sono utilizzate assieme ai controlli di sequenza e ai codici di correzione d'errore.

[4] `https://underwatermessaging.cs.washington.edu/`
[5] `https://developers.google.com/nearby/messages/overview`

2.2.4 SDK

Come abbiamo detto in precedenza, nell'ambito delle trasmissioni DoS esiste finora un solo standard di protocollo di trasmissione (JANUS) che tuttavia risulta ancora scarsamente utilizzato nei progetti esistenti.
Nondimeno sono disponibili per gli sviluppatori alcuni strumenti di sviluppo (SDK), sia propietari che open source, per agevolare la scrittura e la programmazione dei progetti che prevedono uno o più componenti di trasmissione ultrasonica (o acustica in generale).

Alcuni tra gli SDK propietari includono Chirp[6], basato su JavaScript, LISNR[7], basato su librerie C, e SONARAX[8], basato su applicazioni Android.
Per quanto riguarda invece gli SDK open source possiamo citare Audio Network[9], scritto in TypeScript, il progetto Quiet Modem[10], scritto in C, e QuietJS[11], un applet JavaScript da browser per trasmettere dati DoS che si appoggia alla libreria Quiet Modem.

[6] https://developers.chirp.io/
[7] https://lisnr.com/resources/developers/
[8] https://www.crunchbase.com/organization/prontoly
[9] https://github.com/robertrypula/AudioNetwork
[10] https://github.com/quiet/quiet
[11] https://quiet.github.io/quiet-js/

Capitolo 3

Il canale DoS

Per selezionare uno schema di trasmissione adatto bisogna tenere in considerazione lo scenario nel quale andrà utilizzato, ovvero mediante un'applicazione Android durante un concerto con musica dal vivo. La prima considerazione è abbastanza ovvia e riguarda il fatto che la trasmissione non deve intralciare la fruizione della musica da parte del pubblico, dunque dobbiamo escludere tutta la banda di frequenze udibili dall'uomo come utilizzabili dal canale, e quindi un primo requisito per lo schema da utilizzare è quello di funzionare anche avendo una banda limitata di frequenze a disposizione.

Sempre considerando lo scenario tipico, un altro requisito è che la ricezione del dato non può avvenire con un delay troppo elevato dall'istante in cui viene trasmesso, ovvero il tempo trascorso dall'istante di trasmissione a quello di ricezione e decodifica deve rimanere sotto un certo limite.
Inoltre data la variabilità delle caratteristiche principali dell'intero sistema (trasmettitore-canale-ricevitore) sarebbe preferibile uno schema di trasmissione che fosse il meno sensibile possibile alle variazioni dello stato del sistema (attenuazione, fading, drift, etc.)

Date queste considerazioni iniziali lo schema migliore che si può adottare sembrerebbe essere basato sulla modulazione di frequenza (FM), poichè essendo l'informazione trasmessa unicamente contenuta nella frequenza del segnale, questa risulta pressochè invariante rispetto alle variazioni di ampiezza (attenuazione) e alla propagazione multipla (fading), inoltre risulterebbe robusta anche rispetto a variazioni di fase e all'aggiunta di rumore (uniforme).
Tuttavia un tale schema non risulta facilmente applicabile nel nostro caso a causa di un componente utilizzato nella fase di decodifica: tale componente è conosciuto come inseguitore di frequenza (Phase Locked Loop, PLL) e avendo a disposizione all'ingresso del ricevitore solamente una versione già campionata del segnale audio analogico (peraltro a frequenze vicine alla frequenza di Nyquist) risulta impossibile adottare le tecniche note utilizzate nei PLL digitali (zero crossing, campionamento adattivo, etc.) e dunque la scelta della codifica di canale deve ricadere su altre tipologie di trasmissione.

3.1 Prima soluzione: AM

La prima soluzione presa in considerazione si basa su dei concetti analoghi a quelli che vengono utilizzati nell'impiego della supereterodina in alcuni strumenti come gli analizzatori di spettro. L'idea di base è quella di utilizzare la modulazione di ampiezza per traslare in frequenza il segnale contenente le informazioni e farlo così diventare inudibile, dopodichè trasmetterlo e riutilizzare la

modulazione di ampiezza al ricevitore per riportare il segnale in banda base e successivamente poterlo decodificare.

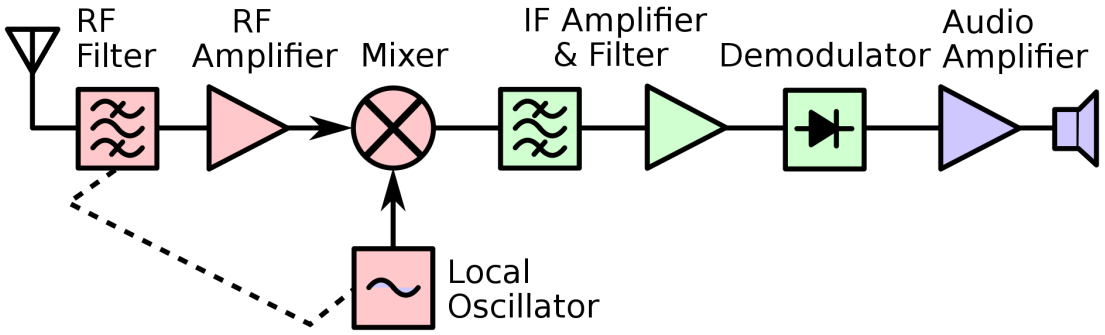

Figura 3.1: Struttura base di una supereterodina.

La struttura schematica del funzionamento della supereterodina si puó osservare in figura 3.1. In un analizzatore di spettro si vuole potenzialmente poter osservare l'andamento di segnali lungo un intervallo di frequenze il più ampio possibile, e questo richiederebbe un banco di filtri talmente numeroso che di fatto la sua implementazione risulta proibitiva.
Per questo motivo la soluzione adotta l'uso di un miscelatore (mixer) che attraverso la modulazione di ampiezza trasporta un segnale dalle alte alle basse frequenze moltiplicando il segnale in esame con il segnale che arriva da un oscillatore locale, e successivamente eliminando i sottoprodotti in alta frequenza attraverso un filtro.
Chiaramente in un analizzatore di spettro lo scopo é quello di portare in bassa frequenza dei segnali in alta frequenza con poca o nulla distorsione per poterne analizzare lo spettro, tuttavia la stessa idea si può applicare anche all'inverso per portare un segnale dalle basse alle alte frequenze per la trasmissione, e successivamente usare la supereterodina per la demodulazione.

Nel nostro caso utilizzare la modulazione di ampiezza per la trasmissione nel canale implica di avere già a disposizione un segnale con spettro in banda base contenente le informazioni da trasmettere, e per poterlo contenere efficacemente nella banda limitata che abbiamo a disposizione per il segnale utile, tale segnale dovrà essere esso stesso a banda limitata, e dunque occorre condizionarlo a valle della sua creazione con i dati in ingresso al sistema.

3.1.1 Pulse Shaping

Come accennato prima nel nostro scenario tipico sorge l'esigenza di sfruttare nel miglior modo possibile la banda di frequenze nell'inudibile, ovvero di confinare il segnale utile in una banda specifica senza code in frequenza che possono risultare (anche debolmente) udibili.
Bisogna tuttavia tenere in considerazione il fatto che, soprattutto per le frequenze vicine a 18-20 kHz, nella maggior parte dei casi avremo a che fare con un sistema (trasmettitore-canale-ricevitore) che non presenta una risposta in frequenza con caratteristica uniforme (piatta), ma solo localmente uniforme, ovvero con una escursione di ampiezza che si puó ritenere accettabile solamente in un intervallo ristretto di frequenze.
Bisogna anche considerare che, a differenza del caso della supereterodina analogica, abbiamo a che fare con un segnale campionato nel tempo, e di conseguenza esiste una frequenza limite (chiama-

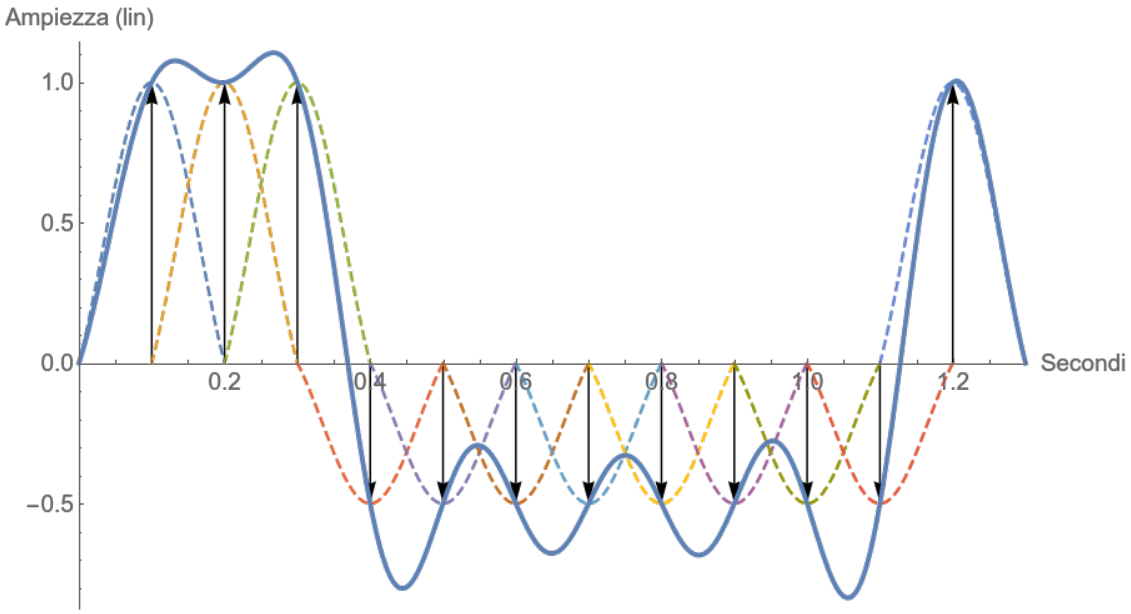

Figura 3.2: Illustrazione del pulse shaping a partire dai singoli Sinc. Gli impulsi originali sono simboleggiati dalle frecce, mentre i singoli Sinc sono tratteggiati. L'asse orizzontale rappresenta il tempo in secondi mentre l'asse verticale l'ampiezza del segnale in scala lineare.

ta frequenza di Nyquist) oltre la quale il segnale in banda base non può essere traslato con la modulazione di ampiezza[1].

Concentriamoci per il momento sul problema principale, ovvero ottenere un segnale contenente le informazioni piazzato in banda base e a banda limitata. Assumiamo anche che le informazioni siano contenute unicamente nell'ampiezza di tale segnale e che i valori obiettivo siano limitati a due (ovvero un singolo bit). Possiamo quindi considerare le informazioni di ampiezza come un treno di impulsi ideali (delta) ognuno moltiplicato per il valore di ampiezza corrispondente, e questo treno viene fatto passare attraverso un filtro a banda limitata che ne riduce lo spettro in frequenza (teoricamente infinito).
Supponiamo inizialmente di voler ottenere una caratteristica in frequenza con una divisione netta tra il segnale utile ed il resto dello spettro, ovvero di usare un filtro ideale a finestra rettangolare in frequenza (filtro brick-wall passa-basso). Questo filtro rimodellerà ogni impulso ideale in ingresso con una funzione simmetrica oscillante decrescente (Sinc) e il segnale complessivo all'uscita del filtro sarà equivalente alla sovrapposizione nel tempo di tutti questi impulsi oscillanti, ognuno dei quali viene centrato sulla posizione del rispettivo impulso ideale in ingresso.

Un segnale così costruito conterrà negli istanti di tempo corrispondenti agli impulsi ideali i valori di ampiezza originali, ed il suo spettro risulterà contenuto in una banda limitata attorno alla banda base. Successivamente il segnale viene moltiplicato per una sinusoide (coseno se si vuole fase nulla) alla frequenza obiettivo ed infine inviato all'altoparlante
In ricezione il procedimento risulta simile a quello di trasmissione: il primo passo è quello di filtrare

[1]Tra l'altro senza il vincolo della banda limitata sul segnale utile, a causa del campionamento emergerebbe anche il fenomeno degli spettri sovrapposti (aliasing)

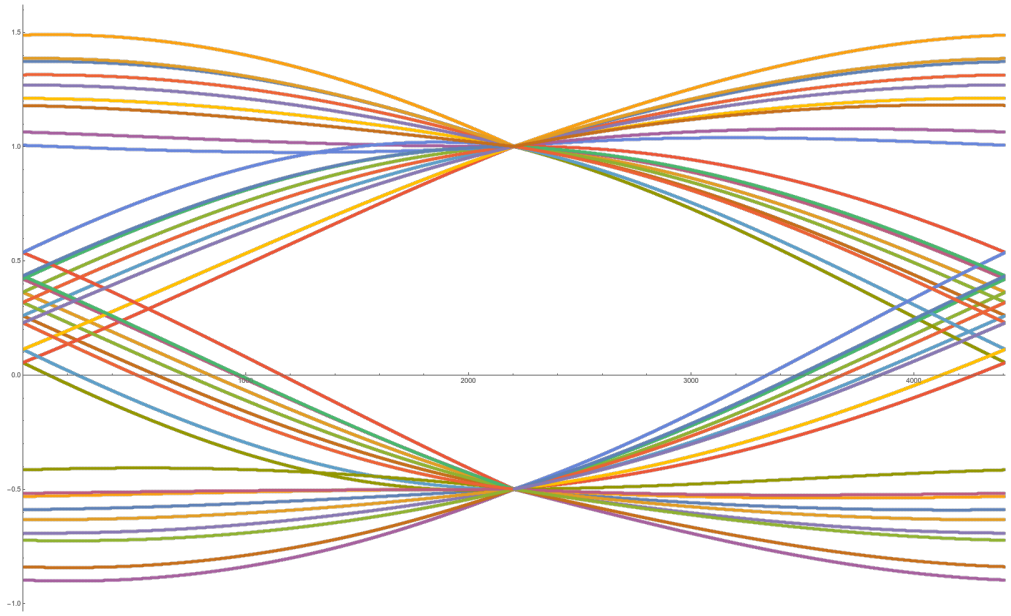

Figura 3.3: Esempio di diagramma ad occhio, utilizzato per valutare l'istante di campionamento dei simboli. L'asse orizzontale contiene l'indice dei campioni audio, mentre l'asse verticale rappresenta le ampiezze in scala lineare.

il segnale ricevuto per estrarre solo la banda del segnale attorno alla frequenza obiettivo e poi moltiplicarlo nuovamente per una sinusoide (sempre alla frequenza obiettivo) per riportare lo spettro del segnale in banda base, e successivamente filtrarlo nuovamente (questa volta in banda base) per eliminare i sottoprodotti della demodulazione.

A questo punto il ricevitore ha ricavato il segnale in banda base ottenuto in trasmissione utilizzando gli impulsi ideali, e deve decidere gli istanti temporali nei quali campionarlo per estrarre le ampiezze relative ai dati trasmessi. Ma per fare questo si può sfruttare il fatto che questo segnale risulta "lento" rispetto alla velocità di calcolo del ricevitore, e quindi si può fare una valutazione complessiva sull'insieme di ampiezze ricevute dal canale fino ad un dato istante.
Detto in altri termini, il ricevitore può tenere traccia di tutto il segnale in banda base ricevuto dal canale, segmentarlo rispetto al periodo di impulso e sovrapporre (raggruppare) nel tempo ogni segmento così ricavato, in modo da avere per ogni istante temporale un insieme (bucket) di campioni audio dei quali può valutarne la dispersione verticale. L'istante di campionamento viene naturalmente scelto come quello che presenta nel suo bucket la dispersione minore.

3.2 Seconda soluzione: OFDM

La soluzione precedente, seppure interessante, soffre principalmente di due problemi: l'attenuazione in ampiezza e l'eccessivo ritardo introdotto in ingresso dai filtri del ricevitore. Il primo si potrebbe mitigare con un decisore a valle del ricevitore che rivaluta periodicamente l'istante di campionamento del simbolo (osservando come cambia il raggruppamento delle ampiezze del simboli al variare dell'istante temporale), tuttavia il ritardo dei filtri in ingresso non si può ridurre più di tanto, pena l'allargamento in frequenza della banda del segnale utile e quindi il suo sconfinamento nelle frequenze udibili, nonchè possibili effetti secondari dati dal campionamento (aliasing).

D'altra parte sia il trasmettitore che il ricevitore nascono da progetto già digitali (nel senso che sfruttano le parti di conversione analogico-digitale esistenti nel sistema utilizzato, il controller audio dell'altoparlante per il trasmettitore e il controller audio del microfono per il ricevitore) e quindi si possono sfruttare anche schemi di trasmissione che richiedono un'elaborazione numerica del segnale non indifferente.
Quello più diffuso è lo schema a multiplazione ortogonale a divisione di frequenza (Orthogonal Frequency Division Multiplexing, OFDM) e si basa su un'importante caratteristica in frequenza di un particolare segnale nel tempo: tale segnale è l'impulso rettangolare ed ha una caratteristica di ampiezza in frequenza proporzionale ad una funzione di tipo Sinc [23].

$$Sinc(f) = \frac{Sin(\pi f)}{\pi f} \qquad Rect(t) = 1 \quad |t| \leq \frac{1}{2} \tag{3.1}$$

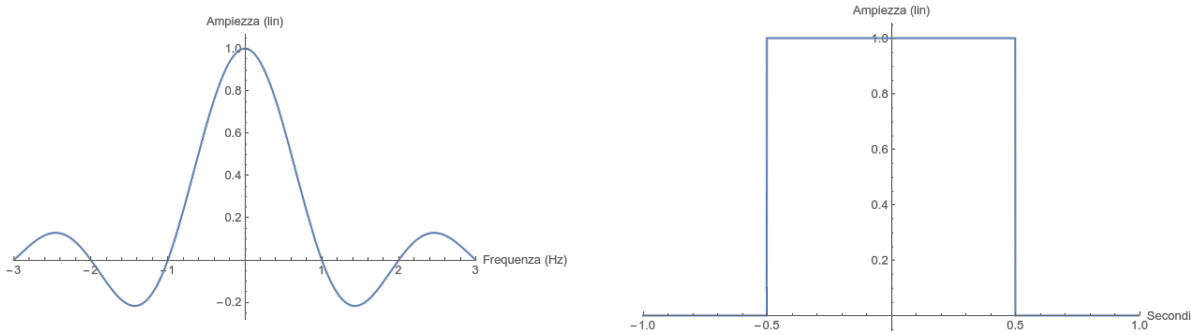

Figura 3.4: Andamento di un Sinc in frequenza e del relativo Rect nel tempo.

Come mostrato nella figura 3.11 la funzione Sinc presenta la caratteristica di annullarsi a distanze multiple di una distanza fissa dalla frequenza centrale, e si può sfruttare questa peculiarità per sovrapporre più funzioni Sinc in frequenza tra di loro. In questo modo, se le loro posizioni reciproche sono scelte accuratamente, la massima ampiezza di ogni funzione sarà allineata verticalmente con gli zeri di tutte le altre, e quindi la sovrapposizione (somma) non ne altererà il valore.

Valutiamo quindi a che distanza gli zeri di questa funzione si trovano dal massimo: data la frequenza di campionamento F_c e la frequenza di simbolo F_s, il primo zero si trova a distanza di $\pm\frac{F_c}{F_s}$ dal massimo e gli altri si trovano a multipli di questa distanza. Dunque una volta stabilita la frequenza di simbolo (normalmente la frequenza di capionamento audio dipende dal sistema scelto) conosciamo anche la spaziatura delle righe associate ad ogni simbolo.
Occorre allora determinare una frequenza di simbolo adatta a questo progetto, e per ottenerla si può tenere in considerazione uno scenario tipico di utilizzo dell'applicazione, ovvero una sala da concerto di medie dimensioni (∼30m in entrambe le direzioni).

Risulta utile premettere che, a differenza della propagazione delle onde elettromagnetiche, la velocità di propagazione delle onde acustiche dipende fortemente dal mezzo nel quale sono inserite, ad esempio in aria si assesta attorno a 430 m/s, e questa (bassa) velocità si traduce in un ritardo (delay) tra l'inizio della trasmissione e l'inizio della ricezione, ma dal momento che il segnale sul quale l'applicazione si dovrebbe sincronizzare è anch'esso un segnale audio (musica) questo ritardo non penalizza significamente la contemporaneità della ricezione.
Ma la bassa velocità di trasmissione genera anche dei segnali secondari, ottenuti da quello principale mediante i suoi rimbalzi sulle pareti della sala, e questi "echi" hanno un estensione nel tempo non

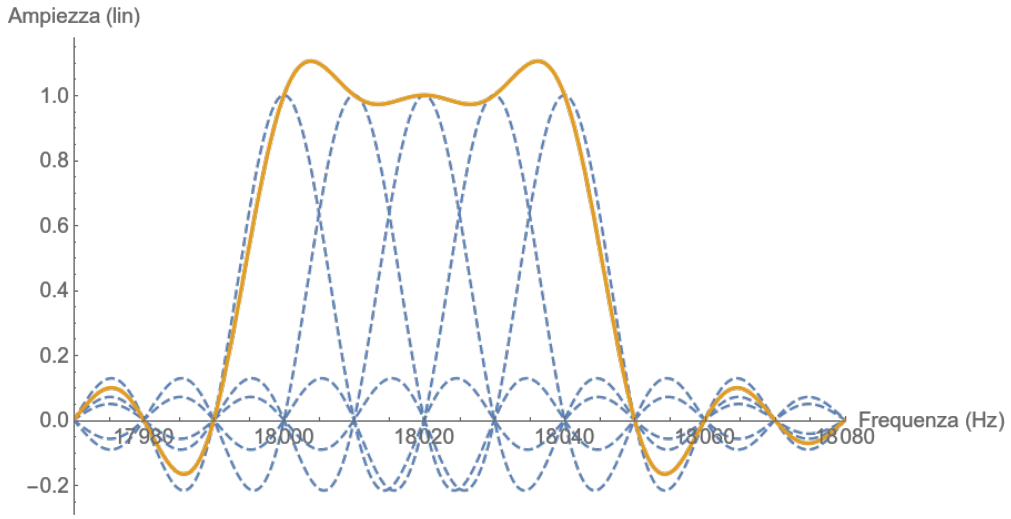

Figura 3.5: Esempio di ortogonalit'a dei Sinc in frequenza. In arancione lo spettro somma degli spettri tratteggiati.

trascurabile, dando origine al fenomeno del fading che concorre alla degradazione complessiva del segnale.

Considerando quindi lo scenario della sala da concerto e, assumendo che il cammino di propagazione secondaria si attesti al massimo in 1.25 volte il cammino diretto[2], allora la percentuale dell'eco del simbolo precedente non si sovrappone a quello corrente per più del 25% ed è ancora possibile la sua corretta decodifica in ricezione.
Inoltre l'impossibilità di variare a piacimento la frequenza di capionamento del segnale audio (in molti casi fissa) impone un limite aggiuntivo alla massima frequenza di simbolo adottabile, e dunque mettendo assieme tutte le considerazioni precedenti al caso peggiore (fondo della sala, posti centrali) risulta ragionevole adottare una frequenza di simbolo pari a 10 $\frac{sym}{s}$ (rapporto tra velocità di propagazione e distamza massima). In ogni caso questa frequenza di simbolo viene presa come un valore iniziale di riferimento e può benissimo essere riconsiderata alla luce di misure specifiche (fading, rumore di fondo, attenuazione) effettuate eventualmente su una o più ambienti reali (auditorium, teatri, etc.).

A questo punto della trattazione abbiamo ricavato la spaziatura in frequenza tra le righe di ogni simbolo e il passo successivo consiste nella scelta della banda in frequenza del segnale utile. In questo caso il primo vincolo risulta essere di natura pratica, ovvero trattandosi di una trasmissione dati in banda audio questa non deve essere avvertita dalle persone presenti in sala, e tantomeno deve interferire con la musica suonata.
Dunque sembrerebbe ragionevole limitare le frequenze utilizzabili alla banda che si estende dai 20 kHz (limite massimo delle frequenze udibili dall'uomo) fino alla frequenza di Nyquist (imposta dal campionamento audio, $\sim$22 kHz nel caso di $F_c = 44.1$ kHz), ma anche questa scelta si scontra con il mondo reale per due motivi: il primo riguarda le funzioni di trasferimento sia dell'altoparlante in trasmissione che del microfono in ricezione, i quali presentano entrambi una attenuazione non trascurabile delle ampiezze alle frequenze maggiori di circa 17 kHz.

[2]L'assunzione riguarda la potenza del segnale di rimbalzo e consiste nel considerare i segnali propagati per più di 1.25 volte la distanza principale non sufficientemente potenti a degradare il simbolo successivo.

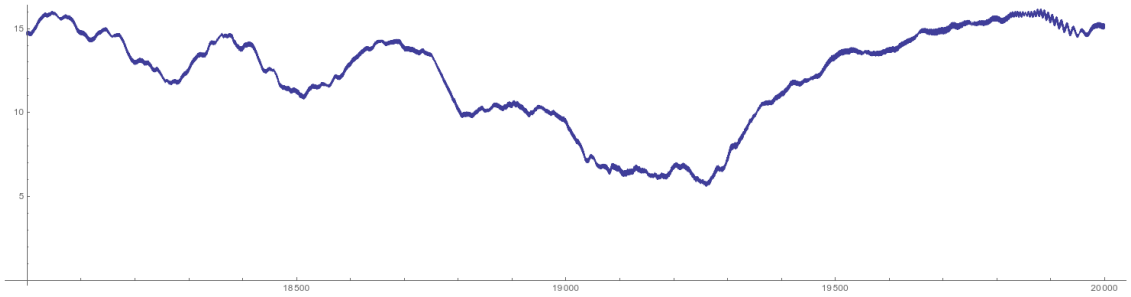

Figura 3.6: Andamento dell'ampiezza della caratteristica di un altoparlante in frequenza. L'asse orizzontale rappresenta la frequenza (in Hz) e l'asse verticale rappresenta l'ampiezza (in dB20).

Il secondo motivo riguarda sempre le funzioni di trasferimento di altoparlante e microfono, ma l'attenzione viene posta sulla loro incostanza ed in particolare sul loro andamento delle ampiezze lontano dall'idealità (caratteristica piatta), e questa considerazione preclude all'utilizzo di tutta la banda nell'inudibile fino alla frequenza di Nyquist.
Risulta quindi una scelta obbligata quella di utilizzare una banda del segnale utile da una parte con un'estensione inferiore all'estensione massima, e dall'altra attorno ad una frequenza inferiore ai 20 kHz, per esempio attorno ai 18 kHz, in modo sia da poter considerare abbastanza costante l'ampiezza in frequenza del sistema trasmettitore-canale-ricevitore (quantomeno nella banda scelta), sia in modo da evitare la pesante attenuazione dello stesso sistema alle alte frequenze. Inoltre il limite dei 20 kHz risulta essere valido per un orecchio perfetto (neonati), dove l'orecchio di una persona adulta raggiunge spesso solo le frequenze fino a 17 kHz.

Facendo una stima ottenuta analizzando alcune risposte in frequenza di altoparlanti e microfoni commerciali (un esempio viene mostrato in figura 3.6) si ottiene una ragionevole larghezza di banda per il canale DoS pari a circa 240 Hz, e scegliendo il limite sinistro a 18 kHz, la banda dutile viene fissata tra 18 e 18.24 kHz.

3.2.1 Costruzione dei simboli

Come visto nella sezione precedente, una finestra rettangolare nel tempo si traduce in una funzione in frequenza proporzionale ad un Sinc, e dunque se la finestra contiene una sinusioide con una data frequenza, lo spettro sarà una funzione Sinc con il massimo centrato sulla frequenza della sinusoide, mentre l'estensione dei suoi lobi laterali dipende dal rapporto tra la frequenza di campionamento e la frequenza di simbolo, $\frac{F_c}{F_s}$.
Sappiamo anche che dalle considerazioni fatte finora abbiamo a disposizione per il canale DoS una banda pari a circa 240 Hz, e ci interessa capire quante funzioni Sinc riusciamo a piazzare dentro questa banda in modo che tutti i massimi siano sovrapposti agli zeri degli altri Sinc.

Per fare questo dobbiamo prima conoscere l'estensione dei lobi laterali dei Sinc (assumiamo tutti i Sinc uguali in frequenza) e ci serve quindi avere a disposizione un valore concreto per la frequenza di campionamento. Ai fini di questo progetto consideriamo un valore di F_c che renda l'applicazione compatibile con il maggior numero di dispositivi Android attualmente in circolazione, ovvero $F_c = 44.1$ kHz.
Con questo valore della frequenza di campionamento, e avendo precedentemente scelto una frequenza di simboli al secondo pari a 10, otteniamo un estensione dei lobi laterali delle funzioni Sinc in frequenza pari a 10 Hz. Dunque una banda di 240 Hz sembra poter contenere un massimo di

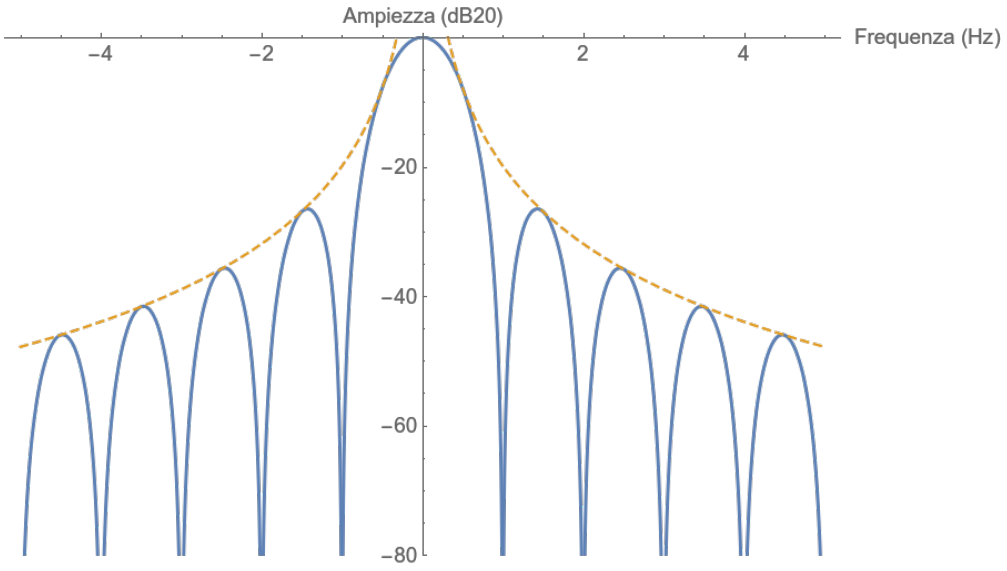

Figura 3.7: Andamento dell'ampiezza del quadrato della funzione Sinc.

25 impulsi sinusoidali sovrapposti (nel tempo) dentro una finestra rettangolare, tuttavia bisogna considerare in modo diverso le righe (Sinc) poste agli estremi della banda in frequenza del segnale utile.

Difatti, sebbene nello schema OFDM i lobi laterali non interferiscono con i valori massimi degli altri Sinc, contengono comunque una parte della potenza complessiva del segnale, e se queste code dovessero scendere troppo al di sotto della frequenza inferiore della banda (18 kHz) potrebbero inserire una parte della potenza del segnale nelle frequenze udibili e dunque disturbare la fruizione della musica dal vivo, come accennato in precedenza.
Abbiamo dunque bisogno di sapere per una funzione Sinc dove si concentra la maggior parte della potenza del segnale, e si può far vedere (con riferimento alla figura 3.7) che i lobi dal terzo in su contengono complessivamente meno del 4% della potenza complessiva (mentre l'ampiezza massima risulta inferiore all'1% dell'ampiezza centrale), e dunque se dentro la banda utile cascano i primi due lobi sinistri della prima riga (Sinc) del simbolo allora questa introduce meno del 2% della potenza del segnale nel resto delle frequenze, tra l'altro prevalentemente concentrata nelle vicinanze della banda utile[3].

Riserviamo allora in frequenza lo spazio per contenere 6 lobi (2 laterali più meta di quello centrale, questo sia a sinistra che a destra della banda) e dunque la nuova estensione di banda utilizzabile diventa pari a 180 Hz, ovvero adatta a contenere 19 righe. Nondimeno non possiamo utilizzarle tutte quante per trasmettere dati a causa di una nuova considerazione, questa volta di natura implementativa.
Difatti lo schema OFDM genera nel tempo ogni simbolo come una serie di onde sinusoidali sovrapposte (superimposed) contenute all'interno di una finestra rettangolare (smussata, come vedremo tra poco) e naturalmente per la corretta decodifica del simbolo è necessario che il ricevitore lo isoli temporalmente dal segnale degli altri simboli, pena una pesante interferenza inter-simbolo in ricezione e la (quasi) garantita ricezione errata dei dati.

[3]L'utilizzo di una finestra smussata (coseno rialzato) riduce ulteriormente questo effetto.

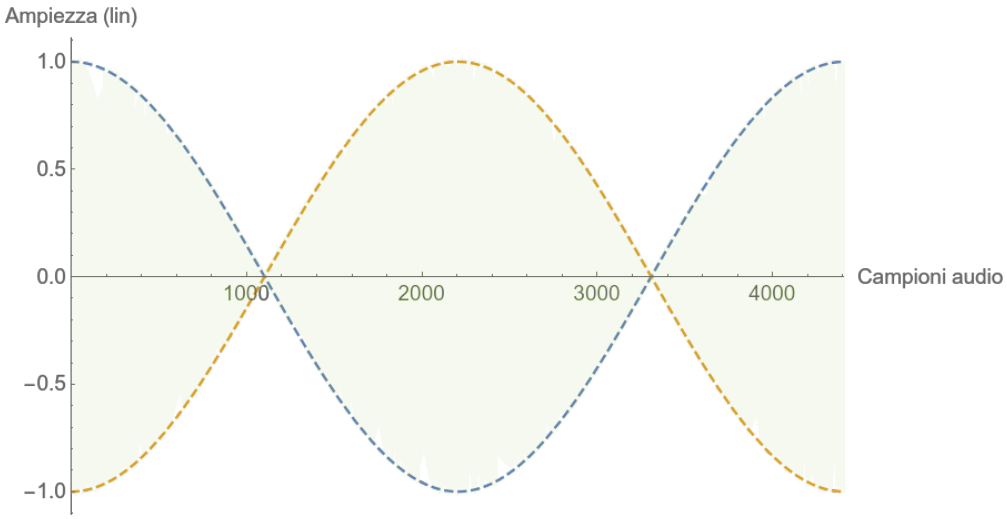

Figura 3.8: Andamento nel tempo del segnale di sync interno al simbolo. La parte in verde rappresenta il segnale completo, mentre quelle tratteggiate rappresentano gli inviluppi di ampiezza.

Normalmente il ricevitore, senza informazioni aggiuntive dall'esterno, purtroppo non riesce a ricavare direttamente dal segnale gli indizi riguardanti l'allineamento temporale dei suoi simboli.
Si potrebbe pensare che inserendo un gap temporale (silenzio) tra un simbolo e il successivo il ricevitore potrebbe utilizzare questi "buchi" per allineare la sua divisione dei simboli nel tempo, ma questa soluzione risulta valida solamente nel caso in cui ogni simbolo presenti all'interno della sua finestra un'ampiezza pressochè costante (o comunque significativamente diversa da zero), e purtroppo questo in generale non si verifica[4]. Inoltre, sebbene il ricevitore possa essere "istruito" sulla trasmissione che sta per avere luogo (mediante un treno di simboli di prova premesso ai simboli dei dati), anche un tale ricevitore istruito può trovarsi nella situazione di perdere l'allineamento a trasmissione già in essere per vari motivi (rumore, fading, etc.).

Possiamo tuttavia sacrificare un po' della banda utile per costruire un segnale da sovrapporre al segnale dei dati e quindi fornire al ricevitore uno strumento utile per allineare temporalmente ogni simbolo, eventualmente anche isolato dal resto della trasmissione. A questo proposito possiamo sfruttare le prime tre righe del nostro simbolo per un duplice scopo: utilizziamo la loro ampiezza come riferimento per le ampiezze delle righe relative ai dati, e sfruttiamo la loro fase in ricezione per valutare il disallineamento complessivo del segnale (group delay).
La scelta quindi è quella di imporre per ogni simbolo (contenente dati) l'ampiezza delle prime tre righe pari rispettivamente a 1,0,1, dove 1 si riferisce ad un'ampiezza di riferimento relativa alle ampiezze delle righe dei dati, e che quindi viene concretizzata una volta costruito il simbolo completo, mentre le fasi delle tre righe vengono imposte a 0.

Possiamo dunque apprezzare l'andamento nel tempo dell'inviluppo di ampiezza di un segnale così definito (come mostrato in figura 3.8) considerando le sue tre righe come fossero piazzate in banda base, e dunque il ricevitore attraverso la stessa operazione può trovare il massimo di tale segnale ed allineare l'inizio del simbolo su di esso.

[4]In ogni caso equalizzare (livellare) l'ampiezza all'interno del simbolo risulta inutile, in quanto distrugge la sua caratteristica in frequenza.

Dopodichè riprendendo la considerazione fatta in precedenza, ovvero che la potenza del Sinc si può considerare esaurita dopo i primi due lobi laterali, lasciamo inutilizzate le prossime quattro righe (per ridurre eventuali interferenze tra il segnale di sync e il segnale dei dati), e alla fine rimangono in tutto 12 righe riservate ai dati del simbolo.

3.2.2 Profondità di codifica

A questo punto della trattazione abbiamo le specifiche riguardo la banda del canale e la divisione in righe dello spettro del simbolo in frequenza, quello che rimane da definire è la profondità di codifica, ovvero quanti bit riuscirà a contenere ogni singolo simbolo.
Per fare questo dobbiamo tenere conto di due fattori,il primo dei quali riguarda la massima distanza concreta che il dispositivo ricevente può avere dal trasmettitore, in quanto l'attenuazione del segnale ricevuto è direttamente proporzionale alla distanza e influisce direttamente sul rapporto segnale-rumore (SNDR).

È altresì vero che molti dispositivi Android recenti (5.1+) hanno in dotazione al microfono un controllo automatico del guadagno audio in ingresso (AGC) e che quindi in presenza di un segnale rilevato in ingresso troppo debole, esso viene automaticamente amplificato (entro un certo limite)[5]. Ma qui interviene il secondo fattore da tenere in considerazione, ovvero che un eventuale AGC si basa sull'ampiezza di tutto il segnale ricevuto e non solo di quello trasmesso nella banda utile, e in uno scenario tipico di utilizzo dell'applicazione il segnale acustico della musica suonata dal vivo copre la maggior parte del tempo il segnale DoS, oppure detto in altri termini l'AGC non si baserà quasi mai sul segnale DoS per aggiustare il guadagno audio in ricezione.

La situazione reale naturalmente può variare molto da concerto a concerto, ma si è scelto comunque di adottare una soluzione prudente, ovvero quella di associare il minor numero possibile di livelli di ampiezza, ovvero due, per ogni riga dati dei simboli. Questa scelta garantisce da un lato una maggior robustezza del canale al rumore generato esternamente, e dall'altro semplifica molto la logica di decodifica del ricevitore, in quanto deve discriminare solo su due livelli di ampiezza e quindi basta un semplice livello di soglia (threshold) della scala verticale. Con questa scelta le ampiezze di ogni simbolo portano in tutto 12 bit.

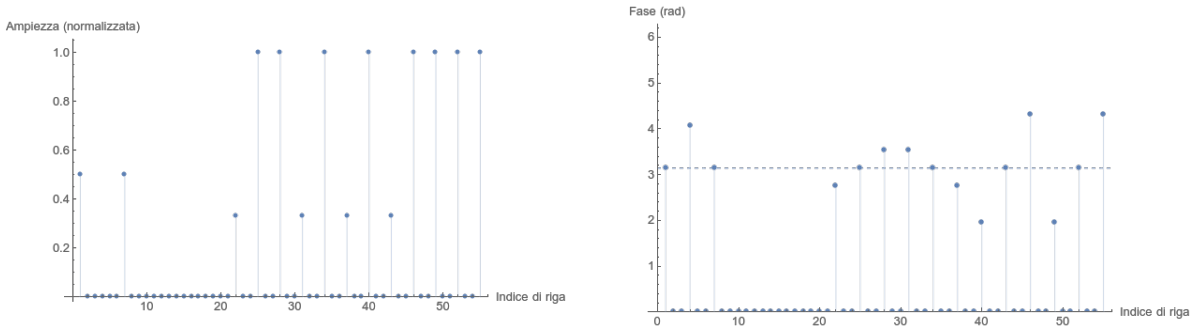

Figura 3.9: Valori di ampiezza e fase delle righe di un simbolo di dati. Le ampiezze sono normalizzate all'ampiezza massima, mentre le fasi sono espresse in eccesso di π (la linea tratteggiata rappresenta fase 0).

[5] È utile osservare che questo tipo di amplificazione agisce anche sul rumore aggiunto dal canale.

Naturalmente ogni riga dati possiede anche un'informazione di fase associata ad essa, e quindi senza aggiungere complessità al simbolo possiamo sfruttare anche le fasi delle righe per trasmettere informazione. La scelta in questo caso è stata quella di riutilizzare la codifica di fase usata per il segnale di sincronizzazione temporale, ma questa volta senza considerare le ampiezze delle righe associate, ovvero considerando di ampiezza uniforme tutte le righe del simbolo (anche se in realtà sono diverse)
Riprendendo quindi quanto fatto per il segnale di sync dividiamo le 12 righe in gruppi di 3, dove ogni gruppo viene interpretato come una modulazione di ampiezza, ovvero l'informazione di fase complessiva è data dallo sfasamento dell'inviluppo di ampiezza equivalente considerando il gruppo di 3 righe come fosse in banda base e con le ampiezze tutte uguali.

Dalle misure di prova fatte in fase di sviluppo è emerso che un numero ragionevole di fasi per ogni gruppo di righe del simbolo si attesta sulla divisione del cerchio (angoli da $-\pi$ a π) in 8 settori simmetrici, e per ognuno di essi la sua fase di riferimento si trova centrata rispetto al relativo intervallo angolare.
Dunque avendo 8 fasi distinte disponibili a gruppi di tre righe per ogni simbolo, ed essendo 12 il totale delle righe dati, ogni gruppo contiene nella sua fase l'equivalente di 3 bit di informazione, per un totale sulle righe di 12 bit, e mettendo tutto assieme ogni simbolo porta 24 bit o 3 byte di informazione, assestando il bitrate complessivo (teorico) del canale DoS a 240 bit/s.

FFT e IFFT

Finora abbiamo illustrato come nello schema OFDM si possano sfruttare intelligentemente le caratteristiche reciproche nel tempo e in frequenza di una data funzione (Rect/Sinc), sorvolando totalmente sul costo in termini di elaborazione di una tale trasformazione.
Difatti la prima operazione che il ricevitore deve eseguire, un volta aver separato temporalmente il singolo simbolo dal resto del segnale, è quella di ricavarne lo spettro in frequenza e valutarne i valori nella banda d'interesse, e il modo più utilizzato per ottenerlo è quello di usare un algoritmo conosciuto come Fast Fourier Transform (FFT) per lo spettro in frequenza, e l'algoritmo inverso (Inverse Fast Fourier Transform, IFFT) per riottenere il segnale nel tempo.

Fortunatamente nel nostro progetto vengono trattati solamente segnali unidimensionali, e sebbene le piattaforme mobili in genere non se la cavino bene nelle operazioni di calcolo intensivo, le prestazioni delle librerie FFT esistenti (sia native che interpretate) adattate alla piattaforma Android sono sufficienti e consentono addirittura un discreto margine di variabilità per quanto riguarda l'utilizzo delle risosre del sistema (sia memoria che processore).
In realtà, dal momento che le frequenze delle righe del simbolo sono note, esiste (quantomeno al ricevitore) un'alternativa alla trasformazione FFT, ovvero correlare il simbolo in ingresso con un segnale sinusoidale ad ognuna delle frequenze attese, e ottenere da questa i valori di ampiezza e fase della rispettiva riga. Tuttavia questa tecnica soffre molto riguardo al fenomeno del mismatch tra le frequenze di campionamento del trasmettitore e del ricevitore (che vengono assunte nominalmente identiche), ma sopprattutto riguardo all'eventuale effetto Doppler [24].
Tale effetto ha come diretta conseguenza lo spostamento in frequenza di tutta la banda utile, e dunque verrebbero alterate significativamente tutte le frequenze delle righe altrimenti note a priori, e anche da questo punto di vista la FFT risulta una scelta migliore, poichè pur non riuscendo a mitigare direttamente il fenomeno può essere utilizzata con successo sia per rilevarlo che per valutarne l'effetto sulla banda del segnale utile (per esempio utilizzando la tecnica zero-padding).

Naturalmente anche il trasmettitore ha un paio di alternative alla trasformazione IFFT per ottenere

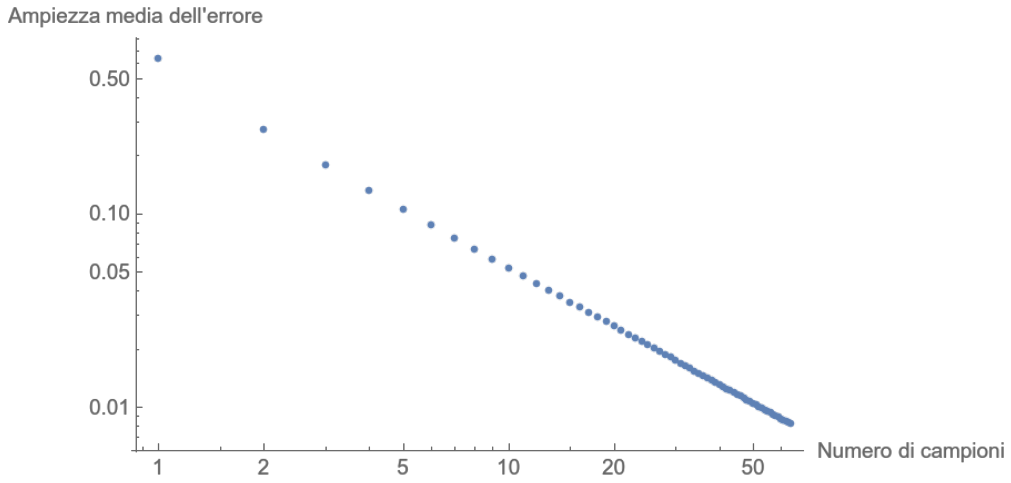

Figura 3.10: Andamento dell'ampiezza media dell'errore (percentuale) di campionamento della funzione coseno al variare del numero di campioni, utilizzando tra due campioni consecutivi un interpolazione lineare.

l'andamento nel tempo del segnale da inviare all'altoparlante, e la prima di queste si basa sulla generazione diretta per ogni riga dei campioni nel tempo di una sinusoide con ampiezza e fase dati dalla riga stessa, in modo analogo a quanto avviene in un oscillatore digitale a frequenza fissa pilotato in ampiezza e fase.
Ma, esattamente come avviene nella FFT, non abbiamo considerato l'eventuale costo in termini di calcolo che comporterebbe l'utilizzo di una funzione coseno nell'ambiente utilizzato per progammare il trasmettitore, giacchè tale funzione verrebbe chiamata per la valutazione di ogni singolo campione audio per l'intera durata della trasmissione.

Tuttavia in questo caso la soluzione è piuttosto semplice e riguarda la tecnica di campionamento e interpolazione lineare, ovvero invece di rivalutare continuamente la funzione coseno in modo esatto, si valutano preventivamente in modo esatto solo un numero limitato di punti della funzione, e tra un punto e l'altro i valori vengono calcolati sul segmento di retta che unisce i due punti precalcolati. Questo permette sia di ottenere una funzione equivalente ma molto più leggera in termini di tempo macchina (ovvero più svelta), sia di controllare la quantità di rumore equivalente (errore) introdotta con questa approssimazione, eventualmente aumentando il numero di punti valutati inizialmente (come si può apprezzare nella figura 3.10).
L'altra alternativa riguarda la possibilità di precalcolare l'andamento nel tempo del segnale di tutti i simboli possibili, al posto di precalcolare la singola funzione di un oscillatore, ma dato l'elevato numero di combinazioni (24 bits = 2^{24} = 16777216 simboli) questa possibilità risulta utilizzabile solo su sistemi con le necessarie risorse di memoria, nei quali è sufficiente utilizzare una LUT (Look-Up Table) con ingresso i valori dei singoli bit e uscita l'andamento nel tempo del simbolo corrispondente.

Rimane ancora un particolare implementativo da risolvere, ovvero cosa succede nel tempo durante la transizione tra un simbolo e il successivo, poichè se i segnali dei due simboli vengono semplicemente concatenati si può verificare la situazione in cui tra i due si presenta un salto di fase che genera del disturbo non trascurabile su tutte le frequenze, e dunque anche quelle udibili (nella pratica si sente uno scatto ad ogni transizione di simbolo).
D'altra parte l'impiego sul simbolo di una finestra temporale generica, sebbene permetta di "smussare" i bordi in ampiezza del simbolo vicino ai suoi estremi temporali, ha come effetto indesiderato

di alterare la posizione in frequenza degli zeri della funzione Sinc, annullando completamente lo schema OFDM.

$$Rcos(t) = \begin{cases} 1 & |t| \leq \frac{1-\beta}{2T} \\ \frac{1}{2}[1 + cos(\frac{\pi T}{\beta}[|t| - \frac{1-\beta}{2T}])] & \frac{1-\beta}{2T} \leq |t| \leq \frac{1+\beta}{2T} \\ 0 & |t| \geq \frac{1+\beta}{2T} \end{cases} \tag{3.2}$$

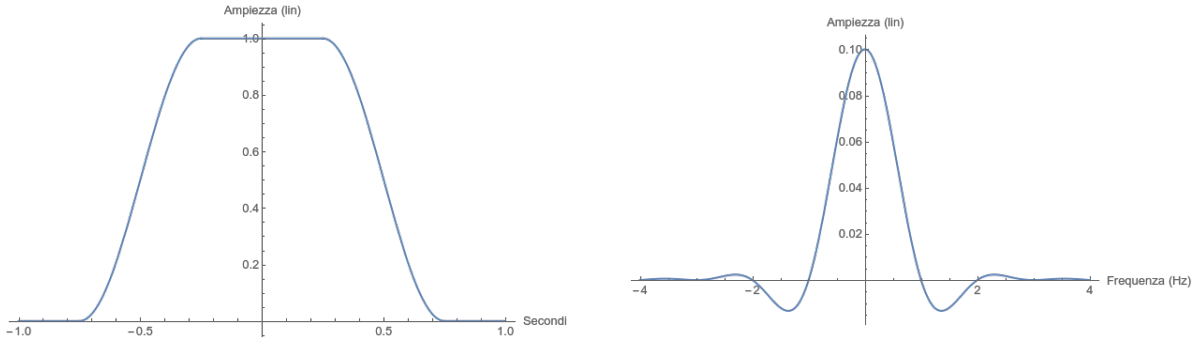

Figura 3.11: Andamento del coseno rialzato in frequenza e nel tempo ($\beta = 0.5$).

Fortunatamente esiste una finestra che fa esattamente quello che vogliamo senza alterare la forma del Sinc in frequenza, ed è la finestra di Hanning (detta anche finestra a coseno rialzato), la cui caratteristica in ampiezza viene mostrata in figura 3.11, e sebbene questa finestra introduca delle code di segnale fuori dall'estensione del simbolo, è sufficiente ricalcolare l'estensione totale del simbolo (e di conseguenza la frequenza nominale di tutte le sue righe) in modo da accomodare questo nuovo fenomeno
Nel nostro caso è stato scelto un valore di roll-off (β) della finestra pari a $\frac{1}{9}$ e reimpostando l'estensione dei simboli ai $\frac{9}{10}$ del valore precedente si possono concatenare tutti i segmenti temporali evitando sia sovrapposizioni che brusche transizioni di fase.

Un ultima precisazione va fatta riguardo alla possibilità del ricevitore di adattarsi alle condizioni che il canale presenta a inizio trasmissione, ovvero di usufruire di una cosidetta fase di addestramento iniziale (training), e per permettere questa fase il ricevitore deve essere in grado di distinguere se è arrivato un simbolo di addestramento oppure un simbolo dati.
Per ottenere questa potenzialità basta utilizzare in maniera oculata le ampiezze delle righe di sync del simbolo, ed in particolare impostando l'ampiezza della seconda di queste righe a zero per i simboli dei dati e al valore delle altre due per i simboli di training, ottenendo così la distinzione cercata senza peraltro interferire con la valutazione del group-delay in nessuno dei due casi.

Capitolo 4

Implementazione Android

Una volta completata la fase di progettazione, di definizione dei parametri della modulazione e delle procedure di codifica e decodifica dei simboli, il passo successivo è stato l'implementazione concreta di un modulo software di ricezione da integrare all'interno dell'applicazione "Listen by Looking"[1]. Questa applicazione è un progetto sviluppato in collaborazione tra l'Università e il Conservatorio di Padova, e consiste nel progetto e sviluppo di una applicazione per dispositivi mobili per la fruizione aumentata di musica dal vivo.

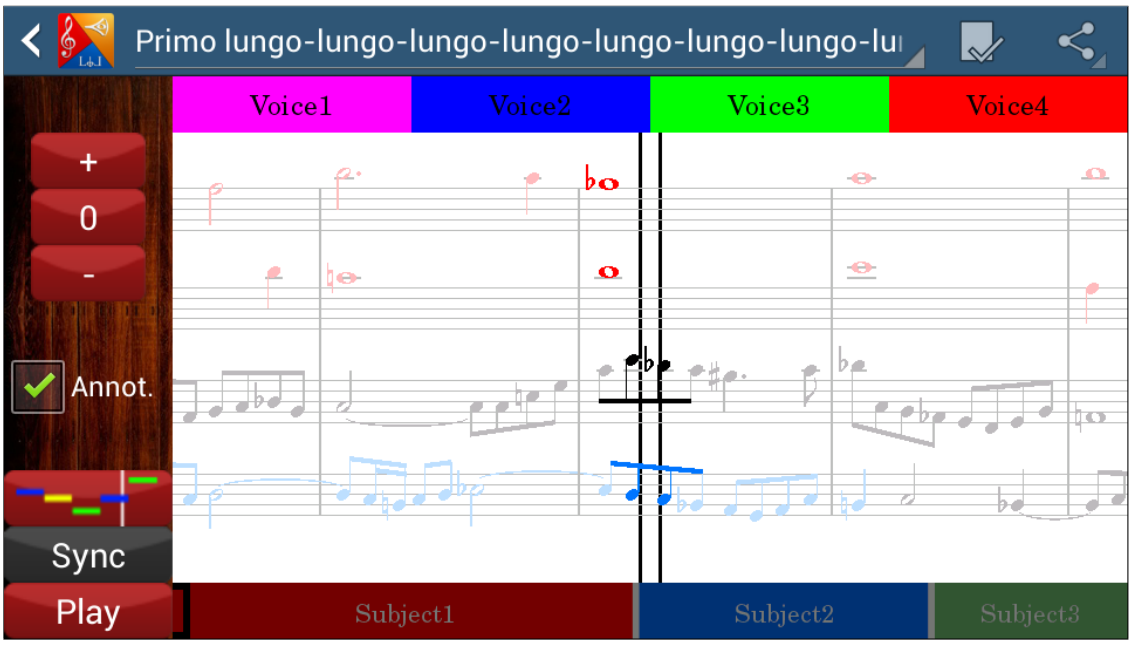

Figura 4.1: Screenshot dell'applicazione "Listen by Looking"

L'applicazione prevede due modalità principali di funzionamento: la prima chiamata Archivio e la seconda chiamata Live. Nella prima l'applicazione carica un pacchetto zip scaricato da un server, contenente i dati relativi ad un singolo brano musicale, e successivamente presenta a schermo una

[1] `https://sourceforge.net/projects/listen-by-looking/`

finestra con uno spartito scorrevole, che l'utente può decidere di scorrere in maniera manuale oppure di far partire uno scorrimento automatico sincronizzato con una registrazione mp3 del brano stesso.

Nella seconda modalità invece lo scorrimento automatico dello spartito avviene sincronizzando la posizione dello spartito in relazione ad un audio esterno, che può essere suonato dal vivo oppure una registrazione riprodotta, e l'applicazione ricava direttamente dall'audio le informazioni utili alla sua sincronizzazione (Score-Following) [25]. Tuttavia questi algoritmi di Score-Following richiedono solitamente una potenza di calcolo che l'hardware delle piattaforme mobili come Android non offrono[2] e quindi la soluzione alternativa prevede un server esterno che ricava le informazioni di sincronizzazione e le trasmette al dispositivo mobile attraverso un canale di comunicazione.
Questo canale nell'applicazione si appoggia ad un uso "improprio" di un router Wi-Fi, ovvero sfrutta la sua modalità di trasmissione in broadcast del nome della rete, e dunque inserendo l'informazione da trasmettere direttamente all'interno del nome della rete (aggiornato con una data frequenza) questa riesce ad arrivare a più dispositivi contemporaneamente senza bisogno di negoziare la trasmissione con ognuno di essi. Purtroppo le ultime versioni di Android (6.0+), dal momento che l'applicazione per ricevere l'informazione deve continuamente ricercare nuove reti Wi-Fi disponibili, vedono questo comportamento come fraudolento (malicious) e lo inibiscono, e naturalmente come accennato nell'introduzione il canale DoS si inserisce nell'applicazione proprio per fornire una soluzione alternativa a questa tipologia di problematica.

4.1 Modularità

La scrittura del ricevitore in ambiente Android è stata portata avanti cercando di mantenere due caratteristiche importanti, la prima riguarda la modularità e consiste nel separare in blocchi distinti le operazioni di ricezione, le operazioni di decodifica e le operazioni di visualizzazione e invio al resto dell'applicazione.
Naturalmente l'applicazione in sè non ha la necessità di visualizzare il risultato della ricezione e decodifica (corretta o errata) ma questo aspetto riguarda la seconda caratteristica del modulo di ricezione, ovvero la sua riutilizzabilità come applicazione stand-alone per fini diagnostici e di testing, come verrà illustrato nel capitolo successivo. Questi blocchi distinti sono stati realizzati sia come classi separate (ricezione/decodifica e visualizzazione/invio) che come metodi distinti all'interno della stessa classe (ricezione e decodifica, visualizzazione e invio), tenendo conto che il flusso di queste quattro operazioni nella maggiorparte dei casi deve procedere in contemporanea e indipendentemente l'una dall'altra, e quindi le loro interazioni devono essere controllate da un gestore di esecuzione parallela (concurrent).

In Android questo tipo di gestione si ottiene attraverso le sezioni Synchronized. Lo scenario tipico in cui si introducono queste sezioni riguarda i programmi nei quali sono presenti più thread di esecuzione e tutti i thread (o almeno una coppia) hanno la necessità di aggiornare delle variabili oppure di alterare la loro esecuzione a seconda del comportamento degli altri thread. Nel nostro caso il blocco di visualizzazione viene lanciato su un thread dedicato e deve controllare che il thread associato alla decodifica abbia prodotto il dato da visualizzare, mentre in caso il dato non sia pronto si deve mettere in attesa che il dato sia prodotto per non sottrarre risorse agli altri thread in esecuzione.
Le sezioni racchiuse dal tag Synchronized soddisfano questi requisiti: per tutti i thread in esecuzione

[2]Negli ultimi anni questa considerazione andrebbe rivista, ma rimane valida l'assunzione che le piattaforme mobili non offrono alte prestazioni riguardo le operazioni di number-crunching.

soltanto uno alla volta può avere in esecuzione una sezione Synchronized, e quindi tutte le altre chiamate ad una sezione Synchronized (la stessa o una diversa) vengono messe in coda, mentre un thread si può mettere in attesa rispetto a un altro (che dovrà segnalargli la fine dell'attesa) attraverso il metodo `wait()`, chiamato sempre all'interno di una sezione Synchronized.

4.1.1 La classe SymbolRecognizer

Il blocco di ricezione e il blocco di decodifica sono entrambi contenuti nella classe SymbolRecognizer, ciascuno lanciato sul proprio thread di esecuzione. La prima operazione che svolge il blocco di ricezione una volta avviato è quella di verificare se è stata avviata una trasmissione sul canale oppure no. Per fare questo il blocco richiede continuamente i campioni audio dal microfono attraverso la classe AudioRecord, salvandoli in un array di ingresso divisi in unità pari alla lunghezza di simbolo (preimpostata), e su ognuno di essi ricava la relativa trasformata di Fourier attraverso la libreria Java JTransforms[3].
Dopodichè, assumendo che normalmente il rumore acustico non si concentri attorno alla banda del canale, su ogni trasformata il blocco confronta la media dell'ampiezza delle righe nella banda di interesse (anch'essa preimpostata) con la media delle ampiezze di tutte le altre righe, e se questo rapporto supera un certo valore (valutato sperimentalmente attorno a 4) allora il blocco di ricezione giudica presente una trasmissione sul canale e si predispone per la fase successiva, mentre nel caso il rapporto risultasse inferiore il blocco rimuove l'unità corrente di segnale dall'array di ingresso e passa ad analizzare l'unità successiva. Detto in altri termini, il primo passo del blocco di ricezione è verificare se è presente oppure no del segnale nella banda del canale.

In caso di trasmissione rilevata, il passo successivo del blocco di ricezione è quello di verificare le caratteristiche di distorsione e sfasamento del canale. Per ottenere questo ogni trasmissione inizia sempre con un treno di 10 simboli di allenamento aventi sia le righe di preambolo che le righe dei dati tutte con la stessa ampiezza e fase nulla, a differenza dei simboli normali dove la seconda riga di preambolo ha ampiezza nulla. Dal momento che il blocco di ricezione sa a priori che tali simboli dovrebbero arrivare con tutte le righe di uguale ampiezza, facendone la media ricava una funzione di correzione da applicare alle ampiezze dei simboli successivi nel caso la risposta in ampiezza del canale non sia piatta.

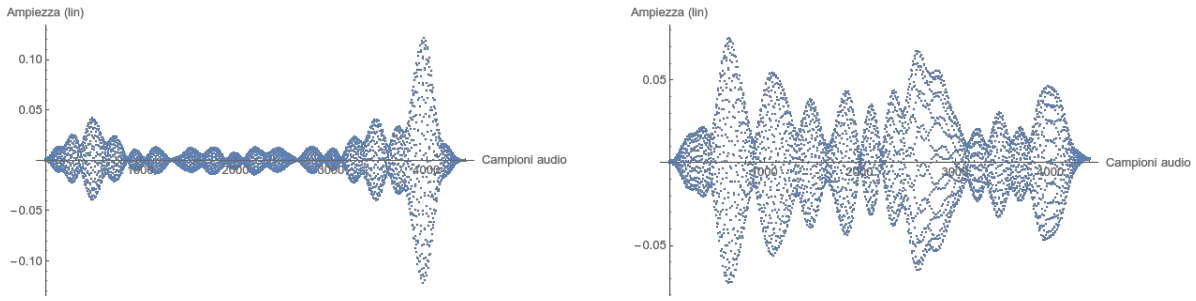

Figura 4.2: Andamento nel tempo del segnale di due simboli, a sinistra un simbolo di allenamento e a destra un simbolo contenente dati.

Sempre da questi simboli di allenamento il blocco di ricezione può valutare lo sfasamento nel tempo

[3] `https://github.com/wendykierp/JTransforms`

(group delay) della trasmissione rispetto alla divisione iniziale del flusso in unità di simbolo e usare l'informazione così ottenuta per ri-allineare i chunk audio dei simboli che verranno serviti al blocco successivo. Questa operazione risulta cruciale per il tipo di trasmissione scelta, essendo l'OFDM molto sensibile rispetto al disallineamento temporale, e nelle fasi preliminari del progetto molti degli errori in ricezione erano dovuti proprio ad un allineamento temporale del segnale in ingresso non ottimale (la verifica è stata fatta col simulatore di ricezione, illustrato nel capitolo successivo).

Finito il treno di simboli di allenamento, il blocco di ricezione esegue tre istruzioni: segna la trasmissione come iniziata e calibrata, segnala (sveglia attraverso il metodo `notifyAll()`) il blocco di decodifica e inserisce le unità audio dei simboli ricevuti in un array di scambio tra i due blocchi, dal quale il blocco di decodifica preleverà i simboli per convertirli nei bit corrispondenti, dopodichè il blocco di ricezione continuerà ad acquisire nuovo segnale fintanto che la verifica della presenza di righe sulla banda del segnale risulta positiva.
Riguardo quest'ultima operazione risulta evidente il motivo per il quale entrambi i blocchi devono operare in ambiente Synchronized: dal momento che entrambi devono accedere ad una risorsa condivisa, ovvero l'array di scambio, solo uno dei due alla volta può modificare tale array aggiungendo o eliminando un elemento, e senza il controllo del tag Synchronized si rischierebbe di produrre degli stati incoerenti dell'array stesso. In ogni caso l'operazione di aggiunta o rimozione di un elemento da un array è abbastanza rapida rispetto al resto del processo e quindi non c'è il rischio che uno dei due processi rimanga in attesa per troppo tempo.

Per segnalare che la trasmissione ha raggiunto la fine vengono sempre aggiunti in coda tre ulteriori simboli di allenamento, processati i quali il blocco di ricezione segna la trasmissione come terminata, informa il blocco successivo che può mettersi in uno stato di attesa (`wait()`) e continua ad analizzare l'audio in ingresso dal microfono per l'inizio di un'eventuale trasmissione successiva, mentre il blocco successivo decodifica gli ultimi simboli svuotando l'array di scambio e si ferma. Il blocco di ricezione è altresì impostato in modo che, nel caso che le condizioni di rumore/distorsione del canale cambino a trasmissione già cominciata, possa ricevere dei nuovi simboli di allenamento intervallati tra un pezzo di trasmissione ed il successivo, in modo da ricalibrare la funzione di correzione delle ampiezze, ma soprattutto di riallineare i chunk audio in caso di ulteriore sfasamento.

Come appena accennato il blocco in cascata alla ricezione è quello di decodifica. Questo blocco si occupa di recuperare l'unità audio corrente rimuovendola dall'array di scambio e di convertirne il simbolo ricevuto nei bit di informazione corrispondenti. Come specificato nel capitolo precedente, nei simboli dei dati le righe di preambolo (3) hanno un'ampiezza pari a metà dell'ampiezza massima, mentre le righe dei dati (12) hanno un'ampiezza massima oppure un terzo dell'ampiezza massima nel caso che rappresentino rispettivamente un uno o uno zero nel bit corrispondente.

Tuttavia è stato osservato nelle fasi preliminari che basare la decodifica delle ampiezze sull'ampiezza di riferimento (preambolo) spesso non risultava un criterio efficace, per cui il blocco di decodifica implementa tre ulteriori strategie per valutare l'ampiezza di soglia (threshold) sopra la quale la riga rappresenta un uno e sotto uno zero[4].
La prima si basa sul calcolo del valor medio tra l'ampiezza massima e l'ampiezza minima delle righe dei dati del singolo simbolo: questa strategia si comporta bene nei casi in cui le ampiezze, sebbene rumorose, sono raggrupate in due gruppi (cluster) distinti, ma soffre nei casi in cui una singola ampiezza risulta essere molto maggiore della media.

[4]Naturalmente ogni criterio fallisce se tutte le ampiezze rappresentano lo stesso bit (0 o 1), ma questa eventualità si può eliminare in fase di trasmissione (scrambling)

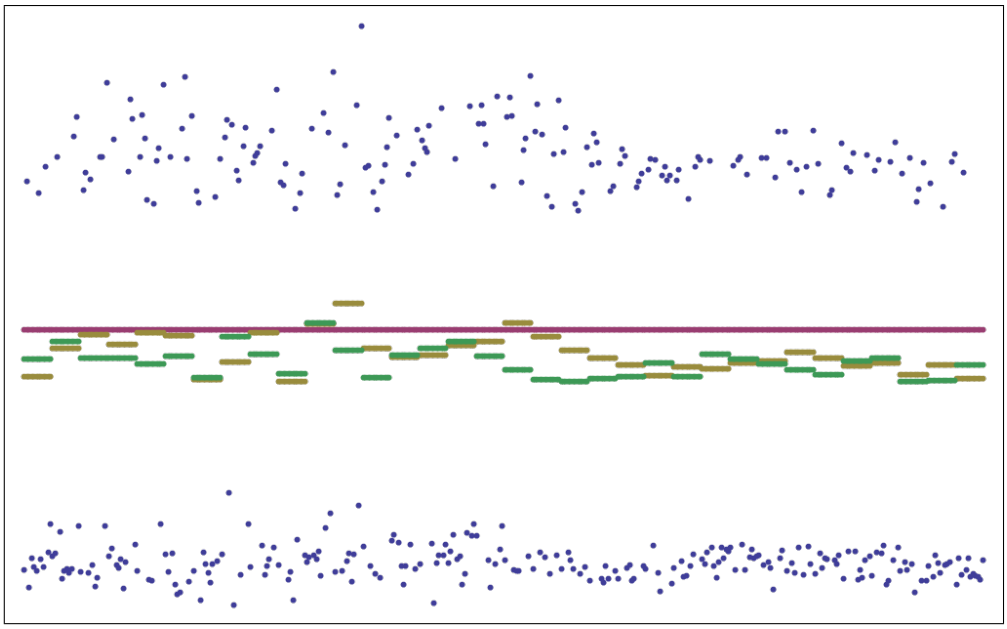

Figura 4.3: Esempio di ampiezze ricevute utilizzando il simulatore di ricezione.

La seconda invece si basa sulla media aritmetica valutata su tutte le ampiezze delle righe dei dati ricevute, sempre all'interno del singolo simbolo. Questo approccio rispetto al precedente si comporta meglio nei casi nei quali una singola ampiezza si discosta molto dalle altre, ma nondimeno soffre nei casi in cui, sebbene le ampiezze siano raggruppate in cluster, uno dei due presenta una distribuzione più allungata rispetto all'altro, ovvero l'ampiezza media rischia di entrare all'interno di uno dei due gruppi.

Infine la terza strategia cerca di trovare il gap più elevato tra le ampiezze, ovvero le ampiezze delle righe dei dati di un singolo simbolo vengono ordinate in ordine crescente, e successivamente si cerca la coppia di ampiezze consecutive con la distanza massima e se ne ricava il valore intermedio. Questa soluzione si comporta meglio nei casi nei quali le altre due soffrono, ma ha prestazioni scadenti in quelli dove la distribuzione delle ampiezze risulta pressocchè uniforme, laddove le altre due ottengono errori solamente su uno o due bit.
Dunque tutte e tre queste strategie sembrano valide in alcuni casi e meno in altri, e quindi nessuna di loro è la migliore a priori, e pertanto il blocco di decodifica le mantiene tutte e tre in previsione di un'ampliamento futuro delle funzionalità del canale DoS, in particolare con l'utilizzo dei codici a correzione d'errore (come verrà accennato nel capitolo successivo).

L'informazione dei bit relativa alla fase invece risiede nei gruppi di tre righe consecutive dei dati, ed è contenuta nello sfasamento equivalente di un'onda sinusoidale rappresentata dalle tre righe come se fossero in banda base. La scelta effettuata è stata quella di far rappresentare ad ogni fase un numero di bit pari a tre, e quindi le fasi delle righe dei dati contengono un totale di 12 bit, che unito ai bit relativi alle ampiezze forma un totale di 24 bit per simbolo, ovvero 3 byte o 3 caratteri testuali.
Da questa scelta risulta evidente che le soglie decisionali riguardanti le fasi sono 8 e corrispondono alla divisione della circonferenza unitaria in ottanti, inoltre essendo lo sfasamento espresso da tre righe invece di una sola risulta più robusta la sua valutazione in decodifica poichè, anche in caso di uno sfasamento dell'intera risposta in frequenza del canale, lo sfasamento dell'inviluppo

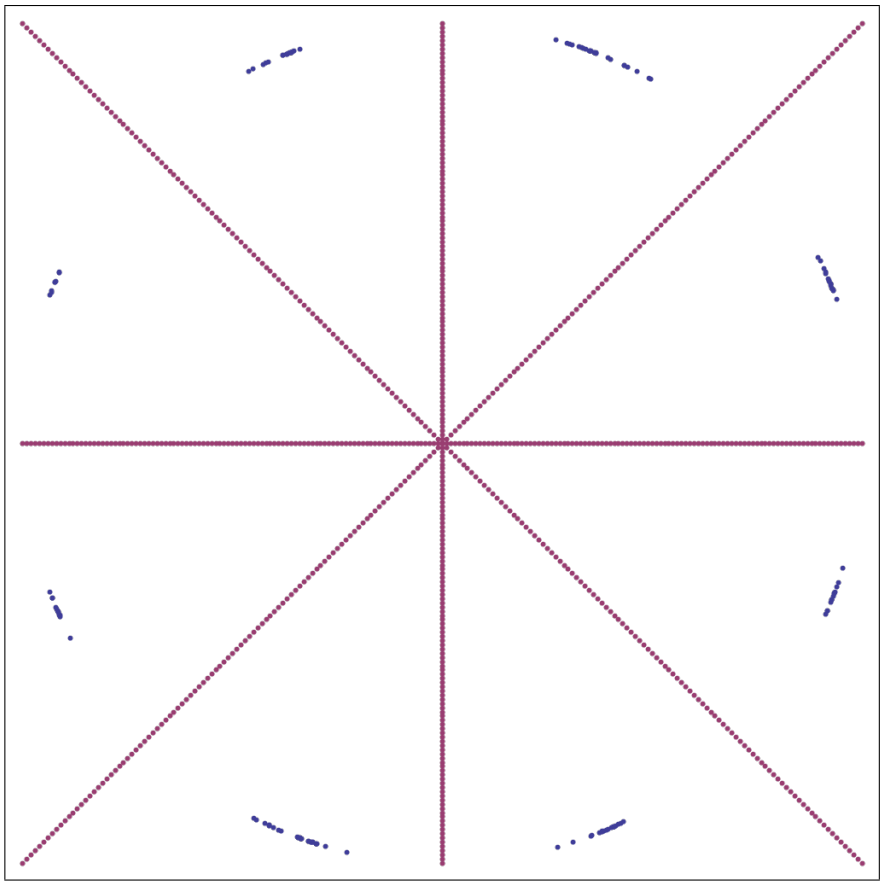

Figura 4.4: Esempio di costellazione delle fasi ricevute utilizzando il simulatore di ricezione.

rappresentato dalle tre righe risulterebbe comunque molto inferiore.

Ottenuti i bit relativi al singolo simbolo, il blocco di decodifica li converte nel loro equivalente carattere testuale (ASCII) e li inserisce in un array di ingresso al successivo blocco di invio, ed esattamente come sopra, essendo questo array una risorsa condivisa da due thread, l'accesso ad esso va regolato dall'ambiente Synchronized.

4.1.2 La classe Analyzer

La classe Analyzer contiene il blocco di visualizzazione che, come detto in precedenza, non serve direttamente alla decodifica ma risulta alquanto utile nelle operazioni di diagnostica. Tale classe estende la classe View di Android e in quanto tale richiede di sovrascrivere tutti i metodi di inizializzazione e di disegno di una finestra in un applicazione separata, soddisfando quindi la richiesta di riutilizzabilità espressa all'inizio.
Il metodo di disegno (`onDraw()`) organizza la schermata dividendola principalmente in tre parti separate da un divisore verticale (linea): la prima contiene una rappresentazione grafica della funzione di correzione delle ampiezze utilizzata dal blocco di ricezione, in modo tale che i valori di riferimento appaiano come dei punti al centro del grafico, mentre l'estensione delle linee rappresenta lo scostamento da tali valori ideali.

La parte centrale contiene il riepilogo di tutte le ampiezze contenute nei simboli di dati ricevuti, unitamente ai 3 livelli decisionali usati all'interno di ogni singolo simbolo, distinti con colori diversi.

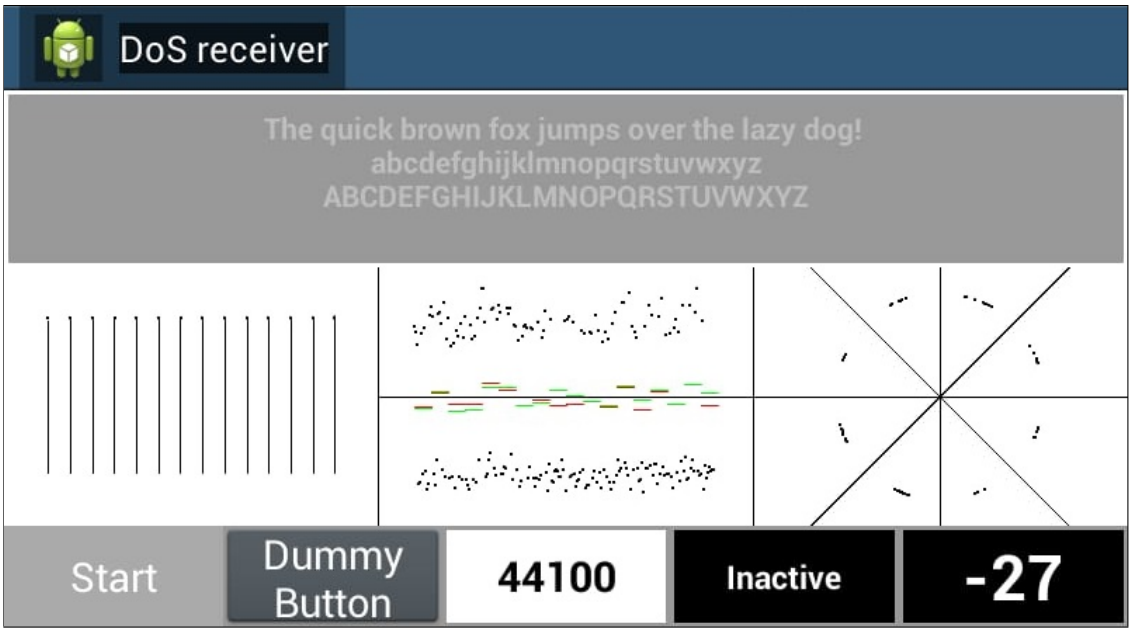

Figura 4.5: Schermata principale del ricevitore stand-alone.

Questa rappresentazione permette di apprezzare visivamente la distribuzione locale delle ampiezze attorno ai livelli di riferimento (rumore) e valutare eventuali attenuazioni avvenute durante il flusso del segnale, questo in quanto essendo il riepilogo basato su tutti i simboli ricevuti non presenta nessuna normalizzazione delle ampiezze rispetto ai valori ottenuti con la calibrazione iniziale.
Da questo grafico risulta poi possibile verificare in quali casi un livello decisionale di ampiezza risulta più efficace di un altro nel corso dell'intera trasmissione.

L'ultima parte contiene una rappresentazione grafica della costellazione di tutte le fasi contenute nei simboli di dati ricevuti, unitamente alla divisione negli otto settori decisionali. In modo analogo al grafico delle ampiezze, questa rappresentazione permette di apprezzare lo scostamento delle fasi dalla loro fase di riferimento dovuto al rumore di trasmissione, e anche di valutare visivamente un'eventuale rotazione globale dei cluster associati alle singole fasi.

Infine un'ulteriore dato prodotto da questa classe riguarda il rapporto segnale/rumore globale relativo alla trasmissione. Questa misura viene ricavata prima dalla trasformata di Fourier relativa ad ogni singolo simbolo ricevuto, e successivamente mediata rispetto al numero di simboli ricevuti.
Risulta opportuno fare una precisazione riguardo alla maniera in cui viene valutato tale rapporto, nel senso che una misura completa dovrebbe rapportare la potenza del segnale nella banda del canale alla potenza del rumore su tutte le altre frequenze, ma nel caso dei segnali audio tale rumore può essere concentrato in una banda di frequenze distanti dalla banda del canale stesso, e quindi una tale misura non sarebbe rappresentativa dei reali effeti del rumore sul segnale utile.

Dunque da questo punto di vista risulta più utile una misura effettuata in questo modo: si considera la potenza media del rumore in un intervallo di frequenze attorno alla banda del canale (nella pratica è stato considerato l'intervallo di frequenze comprese dal limite destro della banda utile fino alla frequenza di Nyquist) e si corregge il valore di tale potenza come se fosse estesa su tutte le frequenze in esame (comprese quelle del canale). In questo modo l'effetto del rumore dovuto alla parte udibile del segnale viene scartato (assumendo trascurabili eventuali effetti dovuti alla saturazione del microfono).

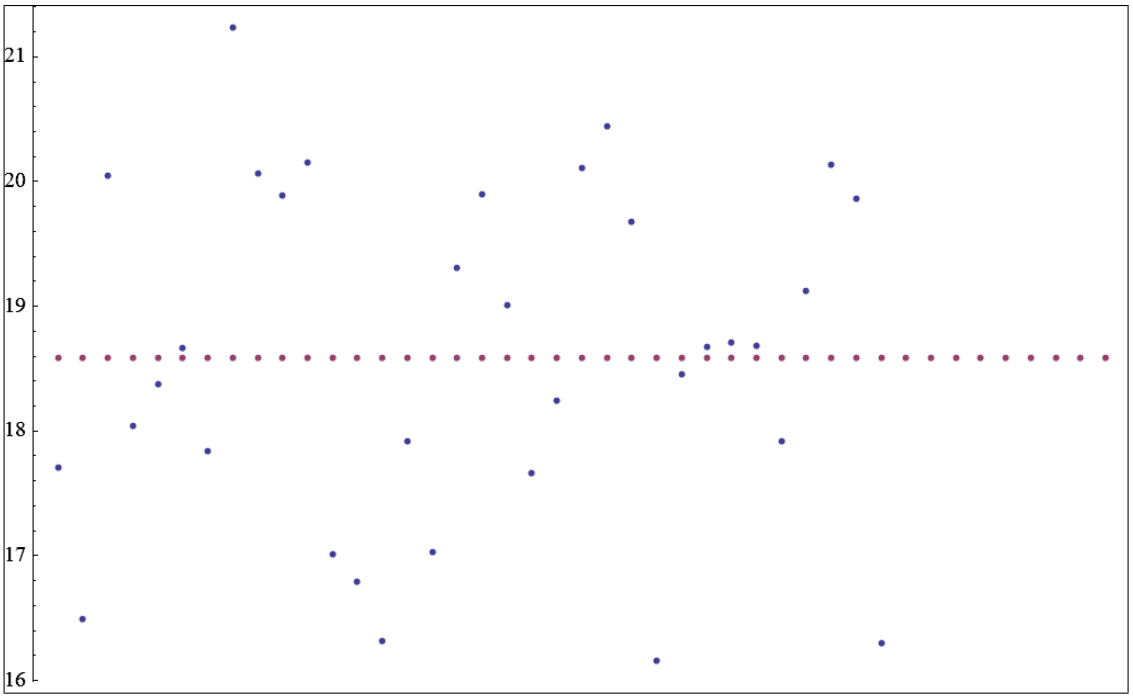

Figura 4.6: Andamento del rapporto SNDR (in decibel) durante una trasmissione. La scala verticale è in decibel (dB20) e la linea tratteggiata rappresenta il valore medio.

Il secondo blocco contenuto nella classe Analyzer è quello relativo alla preparazione e all'invio dei dati all'applicazione "Listen by Looking", e in particolare i dati vengono forniti in ingresso al suo modulo di allineamento interno dello spartito (ScoreView o RollView). L'aspetto della preparazione consiste nell'estrazione delle singole direttive di allineamento (timecode) dal flusso complessivo di caratteri rappresentato dall'array in ingresso, popolato dal blocco di decodifica. La fase di invio poi incapsula le direttive nel formato oggetto dell'applicazione e le trasmette internamente come messaggio (message) al thread di allineamento, naturalmente entrambi in ambiente Synchronized.
È utile precisare che, sebbene il blocco di invio costruisca gli oggetti contenenti le direttive di allineamento, non effettua nessun controllo sulla coerenza di tali direttive relative al flusso dello scorrimento dello spartito, e per le finalità di questo progetto si è assunto che tali incoerenze non si presentino.

Capitolo 5

Assessment

Per testare il canale DoS si è scelto di adottare un ambiente di test diviso in due parti: la prima parte riguarda il trasmettitore che è stato implementato come un singolo modulo scritto in un ambiente di programmazione interattivo (Wolfram Mathematica), mentre la seconda parte riguarda il ricevitore che si è scelto di separare dall'applicazione principale per riversarlo in un applicazione Android stand-alone, questo per permettere la generazione al ricevitore di dati e statistiche utili alla valutazione del segnale ricevuto, cosa questa non ottenibile direttamente dal modulo ricevitore inserito dentro all'applicazione principale.
In aggiunta a questo ricevitore separato, le sue funzionalità principali sono comunque state trasferite nel modulo del trasmettitore in una sezione di simulazione, in modo da poter effettuare delle misure diagnostiche sui segnali ricevuti dal ricevitore reale senza la necessità di preparare un sistema completo trasmettitore-ricevitore del tipo HIL (Hardware In the Loop). Per fare questo il ricevitore stand-alone permette di salvare i segnali ricevuti in dei file di tipo WAV prima che vengano processati dal ricevitore stesso, e successivamente vengono inseriti nella sezione di simulazione del modulo trasmettitore in Mathematica, che automaticamente li processa (offline) e genera risultati e caratteristiche principali dei segnali ricevuti. Questi risultati possono poi essere utilizzati per valutare le prestazioni del canale e per apprezzare le differenze tra diversi altoparlanti (trasmettitori) e microfoni (ricevitori) inseriti in ambienti diversi.

In preparazione al testing del canale sono stati predisposti tre differenti ambienti di prova: il primo è stato approntato all'interno di una camera anecoica (silente) utilizzando al trasmettitore un singolo altoparlante Genelec 8030C, mentre al ricevitore si è utilizzato un microfono omnidirezionale Behringer Ecm8000, avendo sia l'altoparlante che il microfono allineati l'uno con l'altro frontalmente (line of sight). Date le caratteristiche di questo ambiente le misure effettuate sono state prese come riferimento ideale per i test successivi, in quanto la camera silente sopprime tutti gli echi e i rimbalzi del suono sia sulle pareti che sul soffitto e pavimento, eliminando di fatto ogni effetto del fading al ricevitore. Inoltre, essendo sia l'altoparlante che il microfono dispositivi professionali di fascia alta, possiamo considerare la loro caratteristica in frequenza nella nostra banda di interesse pressocchè piatta e quindi con poca distorsione totale da correggere al ricevitore.
Il secondo e il terzo ambiente di prova sono stati scelti come rappresentativi di situazioni del mondo reale, ovvero due configurazioni piazzate in normali stanze (ambiente domestico) senza pareti fonoassorbenti e utilizzando altoparlanti e microfoni di dispositivi commerciali con una caratteristica in frequenza potenzialmente lontana dall'idealità (piatta). In entrambi gli scenari il ricevitore utilizzato è stato un dispositivo Android concreto con il suo microfono interno (embedded): in un caso

è stato scelto uno smartphone Sony Xperia X, mentre nell'altro caso è stato scelto uno smartphone Google Pixel 3 XL; gli altoparlanti scelti invece sono stati in un caso una cassa Tangent Spectrum X5 e nell'altro caso sono stati utilizzati gli altoparlanti standard di un computer Insignia 8A18A.

Figura 5.1: Interno della camera silente presente al CSC.

In tutti e tre gli ambienti le misure sono state effettuate posizionando trasmettitore e ricevitore a diverse distanze l'uno dall'altro, in particolare a 0.5, 1, 1.5 e 2 metri, e ciascuna misura è stata ripetuta in diverse condizioni di rapporto segnale-rumore (SNDR, Signal to Noise and Distortion Ratio), in particolare con valori di SNDR pari a 0dB, 10dB, 20dB e SNDR infinito (ovvero assenza totale di rumore e distorsione). La maniera più diretta per ottenere a priori valori noti del rapporto segnale-rumore è stata quella di sovrapporre al segnale in uscita dal trasmettitore un segnale di rumore aggiuntivo generato separatamente: tale segnale è stato scelto essere un rumore bianco gaussiano e la sua potenza è stata misurata sia in fase di generazione (ovvero in uscita al trasmettitore) che in fase di ricezione (ovvero prima di entrare nel ricevitore) mediante un fonometro digitale.

Ognuno dei segnali di prova utilizzati contiene nella sua parte utile due sequenze distinte di informazioni: la prima consiste in una sequenza di caratteri alfanumerici (ASCII) per testare la capacità del ricevitore di acquisire brevi messaggi testuali, e per provare tutti i caratteri disponibili nell'alfabeto anglosassone si è scelto di utilizzare la frase utilizzata in tipografia per testare le macchine da scrivere: "The quick brown fox jumps over the lazy dog", prima in maiuscolo e poi in minuscolo seguita dalle cifre arabe, per un totale di 100 caratteri.
La seconda sequenza invece è stata generata utilizzando una forma di registro a scorrimento con retroazione, ovvero un array di 32 bit inizializzato con valori casuali, e fatto ruotare di un bit a sinistra per 32 volte in modo da ottenere un totale di 128 gruppi di 8 bit, equivalenti a 128 caratteri. Questa seconda sequenza, seppur ottenuta con un procedimento deterministico, presenta una distribuzione di probabilità abbastanza uniforme di bit e quindi approssima molto bene una sequenza di bit generata in modo completamente casuale.

Per ogni trasmissione ricevuta viene contato il numero di bit ricevuti correttamente e viene valuta-

ta la media del rapporto segnale-rumore (SNDR) visto dal ricevitore, e le informazioni aggiuntive raccolte dal ricevitore Android scritto ad-hoc sono state utilizzate per popolare i grafici delle costellazioni dei simboli ricevuti, con particolare attenzione riguardo le distribuzioni delle ampiezze e delle fasi.
Come detto in precedenza il rumore relativo ad un singolo segnale di prova è stato generato separatamente e poi aggiunto (sovrapposto) al segnale utile, tuttavia la misura del rapporto SNDR è stata effettuata in modo diverso tra le due sequenze di informazione trasmessa: nella sequenza di informazione testuale il rapporto segnale-rumore è stato misurato con un fonometro ad una distanza di un metro dall'altoparlante utilizzato, mentre nella sequenza di bit generata con il registro a scorrimento il rapporto segnale-rumore è stato valutato offline direttamente sul segnale originale prima dell'uscita dall'altoparlante, e quindi la misura di tale rapporto effettuata sul ricevitore può risultare più bassa a causa dell'ulteriore rumore generato dall'ambiente (canale) che si va ad aggiungere all'ingresso del ricevitore.

La tabella con i risultati ottenuti dalle misure nel primo ambiente (camera silente) non viene riportata in quanto i dati ottenuti non sono rilevanti per la valutazione della qualità della trasmissione: in ogni ricezione tutti i simboli sono stati interpretati correttamente (e quindi tutti i bit ricevuti sono corretti) e il canale non ha aggiunto altro rumore in maniera apprezzabile, ovvero il rapporto segnale-rumore misurato al ricevitore era paragonabile a quello trasmesso (il valore minimo misurato risultava essere pari a 1.5 dB rispetto ad un valore di SNDR trasmesso pari a 0 dB), e questo conferma la scelta di considerare il primo ambiente come ambiente di riferimento per le misure successive.
Invece i risultati relativi al secondo e terzo ambiente (ambienti domestici) vengono riportati nella tabella 5.1 per quanto riguarda la sequenza testuale, mentre vengono riportati nella tabella 5.2 i risultati relativi alla sequenza di bit generata utilizzando il registro a scorrimento.

Nei valori delle colonne di sinistra delle due tabelle si può osservare che, per distanze tra trasmettitore e ricevitore inferiori a due metri e quando il rapporto SNDR risulta essere maggiore di 0 dB, il tasso di errore dei bit in ricezione (bit error rate) rimane al di sotto del 15%. Assumendo che gli errori sui bit siano raggruppati in un unico intervallo temporale, ovvero che siano causati da disturbi ambientali con caratteristica impulsiva concentrati in brevi intervalli di tempo (bursts), allora aggiungendo al treno di pacchetti di informazione trasmessi degli ulteriori pacchetti di dati contenenti le informazioni per uno schema a correzione di errore (ECC) risulta possibile recuperare (recover) al ricevitore alcuni dei bit decodificati erroneamente evitando la necessità di un'eventuale ritrasmissione del dato da parte del trasmettitore.

Una possibilità di correzione d'errore può essere rappresentata dallo schema standard di Reed-Solomon, dove aggiungendo 8 simboli di correzione ogni 12 simboli trasmessi risulta possibile ricostruire fino al 15% di errori sui bit, nel nostro caso con un ritardo in ricezione al più di circa due secondi.
Tuttavia nel caso che gli errori sui bit non siano causati da singoli burst di disturbo ma siano causati da un disturbo di fondo costante, e di conseguenza i bit ricevuti erroneamente siano uniformemente distribuiti nel tempo, la codifica di Reed-Solomon risulta alquanto inefficiente e si rende necessario l'utilizzo di una diversa strategia per la correzione d'errore. Una possibilità in questo senso può essere l'utilizzo di codifiche di parità a bassa densità (Low-Density Parity-Check) che in questo nuovo scenario permetterebbero di ricostruire correttamente anche più del 15% dei bit erroneamente decodificati, a costo di trasmettere un numero di pacchetti di correzione d'errore pari a quelli

	Bit error rate (%)		**Measured SNDR (dB)**	
	Setup 2	Setup 3	Setup 2	Setup 3
0.5 mt distance				
No noise	0.0	0.0	37.4	33.5
20 dB SNDR	0.0	0.0	21.3	25.0
10 dB SNDR	0.0	0.0	11.5	15.6
0 dB SNDR	0.0	0.0	2.3	6.3
1 mt distance				
No noise	0.0	5.6	31.5	27.9
20 dB SNDR	0.0	5.4	21.1	22.1
10 dB SNDR	0.0	6.9	11.8	13.4
0 dB SNDR	0.0	6.9	1.5	4.2
1.5 mt distance				
No noise	30.9	23.1	20.6	19.7
20 dB SNDR	27.1	25	15.6	10.1
10 dB SNDR	22.5	22.6	7	7.8
0 dB SNDR	26.4	24.1	-2.6	-1.5
2 mt distance				
No noise	10.9	7.8	24.7	20.3
20 dB SNDR	12	7.5	17.9	16.6
10 dB SNDR	12.2	8.1	9	8.3
0 dB SNDR	7.2	11.8	-0.9	-0.6

Tabella 5.1: Table reporting DoS transmission results measured in the second and third test environments (domestic environments) for the string type test signal.

normalmente trasmessi[1].

Utilizzando successivamente la sezione di simulazione di ricezione contenuta nel modulo scritto con Mathematica, utilizzando le registrazioni dei segnali ricevuti dai dispositivi Android di prova, è stato possibile fare ulteriori considerazioni riguardo al tasso di errori sui bit ricevuti: la prima riguarda la costellazione delle ampiezze dei simboli ricevuti, dove in molti casi l'errore sul singolo bit era dovuto ad un livello decisionale di ampiezza non posizionato in maniera ottimale. In casi come questi risulta vantaggioso l'impiego di più di un algoritmo in contemporanea per la scelta della soglia di ampiezza (threshold), in quanto il controllo della correttezza del dato ricevuto utilizzando diversi livelli di soglia risulta comunque più veloce della ricostruzione del dato stesso attraverso la codifica della correzione d'errore (cioè rilevare è più semplice che correggere).
La seconda considerazione riguarda invece la costellazione delle fasi dei simboli ricevuti, riguardo la quale la gran parte degli errori sui bit era dovuta ad una rotazione complessiva dei punti rispetto al riferimento della costellazione trasmessa. Questo tipo di errori può essere mitigato utilizzando una codifica dei bit relativi ai punti della costellazione basata sullo schema Gray, in modo che se viene erroneamente decodificato il simbolo adiacente nella costellazione l'errore risulta al più di un singolo bit. Una soluzione più robusta consiste invece nel rivalutare periodicamente la rotazione dell'intera costellazione e successivamente di reimpostare gli angoli di soglia decisionale.

[1] È comunque bene ricordare che questo secondo tipo di fenomeno avviene raramente in un ambiente come può essere una sala da concerto

	Bit error rate (%)		**Measured SNDR (dB)**	
	Setup 2	Setup 3	Setup 2	Setup 3
0.5 mt distance				
No noise	0.0	0.4	38.5	33.4
20 dB SNDR	0.0	0.2	10	14
10 dB SNDR	0.0	0.0	0.8	4.1
0 dB SNDR	57.7	0.1	-12.8	-5.6
1 mt distance				
No noise	0.0	11.4	33.6	27.8
20 dB SNDR	0.0	8	10.4	12.1
10 dB SNDR	0.0	6.5	-0.3	2.2
0 dB SNDR	49.1	13	-10.8	-7.8
1.5 mt distance				
No noise	35.6	26.9	21.5	21.3
20 dB SNDR	31.3	24.1	5.2	6.7
10 dB SNDR	30.3	48	-4.1	-3.4
0 dB SNDR	46.9	48	-15	-13
2 mt distance				
No noise	10.7	14	24.9	21.7
20 dB SNDR	11.8	7.2	6.6	6.8
10 dB SNDR	11.8	12.2	-3.3	-2.8
0 dB SNDR	48.3	N/A	-13.3	N/A

Tabella 5.2: Table reporting DoS transmission results measured in the second and third test environments (domestic environments) for the bits type test signal.

Risulta doverosa una precisazione rispetto a quanto scritto in precedenza riguardo il massimo errore sui bit misurato, ovvero che osservando i dati delle tabelle 5.1 e 5.2 relativi alle misure effettuate ad una distanza d 1.5 metri il BER (Bit Error Rate) risulta significativamente superiore al 15%, e utilizzando i dati generati dal ricevitore ad-hoc, e successivamante la simulazione in Mathematica, è stato possibile individuare la causa di tale aumento: un tale elevato tasso di errore viene prodotto da una sfortunata situazione dell'ambiente di prova dove a tale distanza viene ricevuto un forte eco del simbolo precedente che si va a sovrapporre al segnale del simbolo corrrente al ricevitore, causando una distorsione tale da rendere irriconoscibile il simbolo corretto e, dal momento che la banda dei due simboli è la stessa, tale distorsione non altera il rapporto SNDR della misura stessa. Aumentando o diminuendo tale distanza critica si nota dalle tabelle che l'effetto non risulta più presente nelle misure. In casi problematici come questi l'unica soluzione efficace risulta quella di ridurre la frequenza di simbolo, azione che in alcuni ambiti potrebbe non risultare accettabile.

Infine un'ulteriore considerazione che emerge dai dati prodotti dal ricevitore Android risulta essere quella che una (piccola) parte degli errori in ricezione è causata dal mismatch tra le frequenze di campionamento audio del trasmettitore e del ricevitore, e per mitigare tale fenomeno può risultare utile una valutazione periodica di tale scostamento attraverso l'inserimento nella trasmissione di simboli campione noti, attraverso i quali in ricezione si può valutare il rapporto tra le frequenze delle righe ricevute rispetto a quelle teoriche o rispetto a quelle della valutazione precedente.

Capitolo 6

Conclusioni

In questa trattazione abbiamo visto come sia stato possibile sviluppare un canale di trasmissione DoS per risolvere un'esigenza specifica di un progetto esistente, ovvero l'applicazione "Listen by Looking", e di come sia stato possibile adattarlo ai vari vincoli dettati dalle sue condizioni di utilizzo. La fase di assessment ha poi confermato la sua efficacia e, sebbene i risultati non siano perfetti, sono senz'altro incoraggianti e lasciano la strada aperta per possibili miglioramenti futuri.
D'altra parte la stessa fase di assessment può venire estesa in futuro, considerando diversi ambienti del mondo reale ed effettuando misure specifiche su di essi, per migliorare la comprensione sul comportamento di tali ambienti riguardo soprattutto il fading e il rumore generato dall'ambiente stesso.

6.1 Possibili sviluppi futuri

I possibili sviluppi futuri di un canale di trasmissione DoS possono riguardare il suo impiego in scenari diversi da quello dell'applicazione "Listen by Looking", come per esempio la sua integrazione in altre applicazioni esistenti, la sua completa estrapolazione in modo da farlo diventare un sistema di trasmissione general purpose, o anche il suo utilizzo come strumento di valutazione della distanza fisica tra trasmettitore e ricevitore, comportamento ottenibile conoscendo il mezzo nel quale sono immersi e sfruttando la sincronizzazione preventiva degli orologi (wall-clock) di entrambi.

Forse lo sviluppo futuro più consistente riguarda invece la verifica della fattibilità o meno di una codifica DoS che si appoggi su uno schema di tipo FSK (Frequency Shift Key). Difatti tale modulazione permetterebbe una maggior robustezza del canale riguardo sia l'attenuazione di ampiezza, sia riguardo il problema dell'allineamento temporale intrinseco allo schema OFDM.
In questo caso la fattibilità risiede principalmente nella realizzazione di un modulo PLL che sia in grado di agganciarsi stabilmente alla frequenza del segnale trasmesso utilizzando pochi campioni audio (nello schema FSK il tempo di settling non serve alla decodifica), e che sia in grado di agganciare frequenze in ingresso sostanzialmente vicine alla frequenza limite di campionamento.

Elenco delle figure

5 Assessment

Elenco delle tabelle

Bibliografia

[1] Daniel Wigdor and Dennis Wixon. *Brave NUI world: designing natural user interfaces for touch and gesture*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011. 3

[2] Hiroshi Ishii. Tangible bits: beyond pixels. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages xv–xxv, New York, NY, USA, 2008. ACM. 3

[3] Robert JK Jacob, Audrey Girouard, Leanne M Hirshfield, Michael S Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. Reality-based interaction: a framework for post-wimp interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 201–210, New York, NY, USA, 2008. ACM. 3

[4] Giulio Pitteri, Edoardo Micheloni, Antonio Rodà, Carlo Fantozzi, and Nicola Orio. Listen by looking: A mobile application for augmented fruition of live music and interactive learning. In *Proceedings of the 5th EAI International Conference on Smart Objects and Technologies for Social Good*, GoodTechs '19, page 136–141, New York, NY, USA, 2019. Association for Computing Machinery. 3

[5] Arshia Cont. Antescofo: Anticipatory synchronization and control of interactive parameters in computer music. *Proceedings of International Computer Music Conference (ICMC)*, 7 2008. 5

[6] Nicola Orio and François Déchelle. Score following using spectral analysis and hidden markov models. *Proceedings of International Computer Music Conference (ICMC)*, 7 2001. 5

[7] A.A. Huurdeman. *The Worldwide History of Telecommunications*. IEEE Press. Wiley, 2003. 7

[8] Debajit Datta, Lalit Garg, Kathiravan Srinivasan, Atsushi Inoue, G Thippa Reddy, M Reddy, K Ramesh, and Nidal Nasser. An efficient sound and data steganography based secure authentication system. 2021. 8

[9] Nedeljko Cvejic et al. *Digital audio watermarking techniques and technologies: applications and benchmarks: applications and benchmarks*. IGI Global, 2007. 8

[10] A. Vizzarri and F. Vatalaro. m-payment systems: Technologies and business models. In *2014 Euro Med Telco Conference (EMTC)*, pages 1–6, 2014. 8

[11] Hemani Kaushal and Georges Kaddoum. Underwater optical wireless communication. *IEEE access*, 4:1518–1547, 2016. 8

[12] Stew Magnuson. New applications, markets sought for underwater communication system. *National Defense*, 95(690):24–26, 2011. 10

[13] Stew Magnuson. Navy can text stealthy submarines 24/7. *National Defense*, 93(662):39–39, 2009. 10

[14] Francesco Tonolini and Fadel Adib. Networking across boundaries: enabling wireless communication through the water-air interface. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 117–131, 2018. 11

[15] Sandipa Singh, Sarah E Webster, Lee Freitag, Louis L Whitcomb, Keenan Ball, John Bailey, and Chris Taylor. Acoustic communication performance of the whoi micro-modem in sea trials of the nereus vehicle to 11,000 m depth. In *OCEANS 2009*, pages 1–6. IEEE, 2009. 13

[16] N Farr, A Bowen, J Ware, C Pontbriand, and M Tivey. An integrated, underwater optical/acoustic communications system. In *OCEANS'10 IEEE SYDNEY*, pages 1–6. IEEE, 2010. 13

[17] John Potter, Joao Alves, Dale Green, Giovanni Zappa, Ivor Nissen, and Kim McCoy. The janus underwater communications standard. In *2014 underwater communications and networking (UComms)*, pages 1–4. IEEE, 2014. 14

[18] Roberto Petroccia, Gianni Cario, Marco Lupia, Vladimir Djapic, and Chiara Petrioli. First in-field experiments with a "bilingual" underwater acoustic modem supporting the janus standard. In *OCEANS 2015-Genova*, pages 1–7. IEEE, 2015. 14

[19] M Zeppelzauer and A Ringot. Sonitalk: An open protocol for data-over-sound communication. Technical report, Technical report, 2019. 15

[20] W. Jiang and W. M. D. Wright. Progress in airborne ultrasonic data communications for indoor applications. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, pages 322–327, 2016. 15

[21] Tuochao Chen, Justin Chan, and Shyamnath Gollakota. Underwater messaging using mobile devices. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 545–559, 2022. 16

[22] S. Kim, H. Mun, and Y. Lee. A data-over-sound application: Attendance book. In *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 1–4, 2019. 16

[23] S. B. Weinstein. The history of orthogonal frequency-division multiplexing. *IEEE Communications Magazine*, 47(11):26–35, 2009. 23

[24] Jonathan Kaunitz. The doppler effect: A century from red shift to red spot. *Digestive diseases and sciences*, 61, 01 2016. 29

[25] Diemo Schwarz Nicola Orio, Serge Lemouton and Norbert Schnell. Score following: State of the art and new developments. *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, 5 1993. 34

Indice analitico