UNIVERSITÀ
POLITECNICA
DELLE MARCHE

FACOLTÀ DI INGEGNERIA

Corso di Laurea in Ingegneria Informatica e dell'Automazione

Implementazione Real-time di un Algoritmo di Cancellazione del Crosstalk per Sistemi Audio Immersivi

Real-time Implementation of a Crosstalk Cancellation Algorithm for Immersive Audio Systems

Tesi di laurea di:

TANIA SANTOIANNI

Relatore:

Prof. **STEFANIA CECCHI**

Anno Accademico 2019-2020

INDICE

“numquam deficere.”

Introduzione

Lo scopo di questo progetto consiste nell'implementazione di un algoritmo di cancellazione del crosstalk in real-time utilizzando una piattaforma sviluppata per elaborazioni di Digital Signal Processing (DSP). Il plug-in realizzato su questa piattaforma è in grado di calcolare i filtri di cancellazione del crosstalk considerando l'ascoltatore in una posizione fissa ed inoltre è in grado di filtrare il segnale in ingresso. Sulla base di questo introduciamo lo studio su sistemi audio immersivi, in particolare i transaurali, ovvero quei sistemi in cui nella riproduzione del suono è importante che il canale destro e sinistro arrivino separatamente ai padiglioni auricolari, cosa che è facilmente realizzabile attraverso l'utilizzo delle cuffie ma non con gli altoparlanti, dal momento che in questa configurazione avviene una sovrapposizione tra il segnale proveniente da ogni altoparlante e l'orecchio opposto.

Per risolvere questa problematica si è sperimentata una soluzione che consiste nella cancellazione del crosstalk: viene aggiunto cioè un segnale di cancellazione ad ognuno dei due canali audio così da non far percepire all'ascoltatore il segnale di crosstalk.

La suddetta implementazione è stata sviluppata utilizzando la piattaforma NU-tech, con un'architettura plug-in e un kit di sviluppo software (SDK); essa consiste di PlugIn detti NUTS programmati in C che si possono utilizzare, combinandoli tra di loro nell'ambiente di progettazione utilizzando la sua interfaccia grafica.

La tesi è organizzata nel seguente modo. Nel primo capitolo dell'elaborato vengono presentati i sistemi su cui è basato lo studio, andando a focalizzare l'attenzione su quelli transaurali e sulle problematiche relative al crosstalk. Nel secondo capitolo, viene trattato come principale argomento la cancellazione del fenomeno di crosstalk e gli algoritmi dello stato dell'arte per la sua implementazione, come ad esempio la tecnica RACE (recursive ambiophonic crosstalk elimination), gli approcci basati sulla fast deconvolution, il metodo di time domain deconvolution, il least square, l'algoritmo di least mean square (LMS) e varianti di questo.

Si passa poi, nel terzo capitolo, alla spiegazione dell'implementazione dell'algoritmo in real-time creato, partendo da uno schema a blocchi per poi andare a trattare il codice in ogni suo aspetto.

Nella parte finale, quarto e quinto capitolo, sono mostrati i risultati di questo algoritmo e i vari test sulla piattaforma NU-tech.

Capitolo 1: Introduzione ai sistemi audio immersivi

1.1 Il sistema audio immersivo e la percezione del suono nel mondo reale

Come viene percepito il suono nel mondo reale?

Per rispondere a questa domanda è possibile suddividere l'ambiente in tre elementi: sorgente sonora, ambiente acustico e ascoltatore.

La sorgente è l'elemento che emette onde sonore, l'ascoltatore è colui che le riceve e l'ambiente acustico rappresenta lo spazio dove vengono emesse.

L'onda sonora viaggia nell'ambiente circostante dove è possibile il verificarsi di alcuni eventi: l'onda viene assorbita dall'aria, viaggia verso l'ascoltatore con un percorso diretto, tramite una riflessione di prim'ordine o di secondo ordine rimbalzando rispettivamente una o due volte su un oggetto, oppure può passare attraverso le pareti.

I sistemi audio immersivi sono basati su formati multicanale dove all'ascoltatore in questione vengono aggiunti canali che consentono di far riprodurre suoni orizzontalmente o in alternativa sono in grado di generare un arco verticale all'interno di una stanza. Questo sistema è più comunemente chiamato audio 3D.

La funzione principale di un sistema immersivo è quella di garantire una percezione multimodale che non esiste nell'ambiente fisico reale, immergendo l'utente nella scena audio virtuale.

La riproduzione di un campo sonoro richiede il controllo della variazione della pressione sonora all'interno di un volume per simulare l'effetto di un evento acustico. Il suono è generato da un sistema di altoparlanti o da cuffie ma la percezione dell'ascoltatore è che il suono arrivi da un punto arbitrario nello spazio. Per fare ciò ci sono tre metodi: binaurale (cuffie), transaurale (due altoparlanti) e soundfield (più speaker).

1.2 Percezione del suono attraverso le HRTF (head-related transfer function)

La percezione del suono si basa su caratteristiche direzionali definite appunto dalle Head-Related Transfer Function (HRTF). Le HRTF possiedono tutte quelle informazioni che permettono di individuare da parte del cervello la posizione della sorgente sonora nello spazio 3D. Sfruttando queste funzioni è possibile simulare la provenienza di suoni da specifiche posizioni facendo ascoltare all'individuo il suono originale filtrato tramite le stesse HRTF. In generale vengono definite come il rapporto tra due pressioni sonore misurate in campo libero:

$$HRTF(\phi, \vartheta) = \frac{\text{sound pressure at the ear of the listener}}{\text{sound pressure at head center position}}(\phi, \vartheta),$$

dove (ϕ,θ) indica l'angolo di incidenza dell'onda sonora in elevazione e azimut rispettivamente. La funzione HRTF è definita sia per l'orecchio destro che per l'orecchio sinistro. Aspetto importante da non dimenticare è che le HRTF sono legate alla forma della testa che ha un effetto rilevante sulle onde ad alta frequenza, alla forma del padiglione che possiede delle particolari zone di risonanza che attenuano o amplificano particolari frequenze e riflette le onde con lunghezza d'onda abbastanza limitata a seconda della loro direzione, in ultimo alla posizione delle spalle ed del torso che attenuano le onde sonore provenienti da sotto l'ascoltatore, mentre riflettono quelle provenienti dall'alto.

Queste vengono misurate tramite appositi manichini dotati di padiglioni auricolari o tramite l'uso di microfoni interaurali. Uno di questi è il Kemar, mostrato in Figura 1, dotato di diversi padiglioni auricolari che utilizza l'accoppiatore di Zwislocki che simula il canale uditivo umano. Un altro è il Bruel and Kjaer HATS, mostrato in figura 2,utilizzato nell'acustica in applicazioni quali la valutazione di

cuffie, telefonia, apparecchi acustici e di protezione. La geometria della testa è simmetrica ed è basata sui dati antropometrici di un adulto medio.

Figura 1 Manichino Kemar

Figura 2 Manichino Bruel and Kjaer

L'acquisizione delle HRTF avviene nelle camere semianecoiche. Una camera semianecoica è costituita da una camera schermata, le cui pareti interne ed il soffitto sono ricoperti da materiale anecoico con proprietà fonoassorbenti in un ampio intervallo di frequenze (assorbenti a larga banda). La camera serve ad impedire ai segnali esterni di contaminare l'ambiente interno, mentre il materiale fonoassorbente non permette la riflessione delle emissioni del dispositivo in prova sulle pareti perimetrali e sul soffitto.

1.3 Sistemi binaurali

In un sistema binaurale il suono viene ripreso complessivamente (strumenti ed ambiente nel loro corretto equilibrio, in modo analogo alla nostra percezione uditiva) dal manichino binaurale e registrato direttamente su due soli canali, garantendo una elevata somiglianza tra il suono ascoltato al concerto dal vivo e quello registrato. La registrazione binaurale (a due orecchi) è un metodo di registrazione tridimensionale del suono che ha il fine di ottimizzare la registrazione per il suo ascolto in cuffia, riproducendo il più fedelmente possibile le percezioni acustiche di un ascoltatore situato nell'ambiente originario di ripresa dell'evento sonoro, mantenendone le caratteristiche direzionali a 360°.

Un sistema binaurale è realizzato in due fasi: tramite la registrazione binaurale per la misura di HRTF e quella attraverso l'utilizzo di cuffie.

Funzioni che tengono conto di come l'orecchio umano filtra il suono in arrivo sono le HRTF. Il filtraggio di queste viene effettuato convoluendo il segnale originale con la coppia di HRTF derivate. Per riprodurre N sorgenti sonore posizionate in N differenti posizioni nello spazio 3D, sono necessari N funzioni sia per l'orecchio destro che per l'orecchio sinistro. I principali problemi dovuti all'uso di cuffie sono errori di localizzazione dovuti ai movimenti della testa e un utilizzo prolungato.

1.4 Sistemi transaurali e le loro problematiche

L'audio transaurale impiega un sistema di altoparlanti a due canali che crea una sovrapposizione tra ogni altoparlante e l'orecchio opposto. Si dà vita al fenomeno del crosstalk.

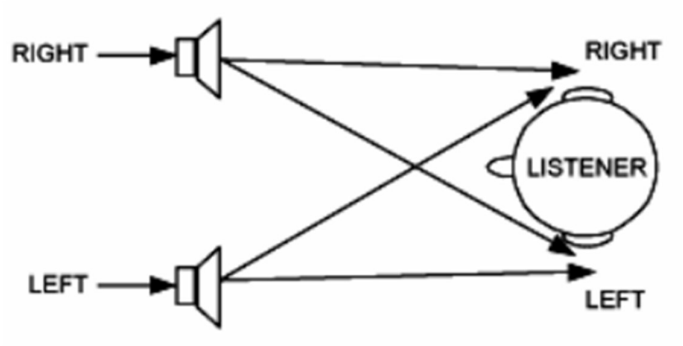

Figura 3 Fenomeno di crosstalk nell'ascolto transaurale

Come mostrato in Figura 3, il segnale dello speaker destro va all'orecchio sinistro e viceversa, motivo per il quale abbiamo bisogno della cancellazione del crosstalk. Le HRTF contengono anche le informazioni relative all' ITD (iteraural time difference) e ILD (interaural level difference). L'ITD viene rilevata quando una sorgente non si trova nel piano mediano. In questo caso l'orecchio contro laterale riceve lo stimolo sonoro in ritardo rispetto l'orecchio ipso laterale. L'uso della ITD è molto preciso e permette di localizzare una sorgente sonora entro un solo grado, ma dal momento che la testa rappresenta un'ombra acustica per le alte e medie frequenze, questo meccanismo è efficace soltanto sulla componente del segnale a basse frequenze. Interaural time difference (ITD) non da altro che la differenza in ordine di tempo (ritardo/anticipo) con cui un suono arriva ad un orecchio rispetto a quando arriva all'altro orecchio. Interaural level difference(ILD) invece, dà la differenza in intensità percepita da un orecchio rispetto a quella percepita dall'altro orecchio in relazione alla stessa stimolazione acustica. La libertà di movimento associata all'ascolto transaurale può però

compromettere la resa del sistema audio 3D: se si considera l'ITD (interaural time difference), risulta evidente come un movimento dell'ascoltatore verso un altoparlante costituisca una distorsione non prevedibile dal progettista e che quindi falserà la corretta collocazione della sorgente. Viene così a individuarsi una regione spaziale limitata, detta sweet spot, all'interno della quale le informazioni spaziali riprodotte saranno correttamente interpretate dal soggetto. In questa regione il cancellatore del crosstalk ha il massimo effetto, al di fuori di essa, invece, la resa del sistema non è prevedibile, né corretta. Il cancellatore va posto a monte del sistema di riproduzione e permette di annullare l'effetto dei contributi incrociati. Si basa sulla risoluzione di questo sistema:

$$\begin{bmatrix} e_l \\ e_R \end{bmatrix} = [H] \cdot [w] \cdot \begin{bmatrix} x_L \\ x_R \end{bmatrix} = \begin{bmatrix} x_L \\ x_R \end{bmatrix}$$

$$w = H^{-1} = \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix}$$

$$H = \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} H_{LL} & H_{LR} \\ H_{RL} & H_{RR} \end{bmatrix}$$

Tramite un sistema binaurale e quindi con l'utilizzo di cuffie, è possibile isolare l'ascoltatore dalle caratteristiche acustiche dell'ambiente fisico nel quale si effettua la riproduzione. Al contrario un ascolto diffuso tramite altoparlanti ne è influenzato e acquisisce le informazioni acustiche dell'ambiente. Tale limite può risulta particolarmente pesante in quelle applicazioni di realtà aumentata, dove lo scenario virtuale si sovrappone a quello reale. L'ascoltatore deve essere posizionato in una posizione fissa tra i due altoparlanti per ottenere il massimo effetto.

Capitolo 2: Lo stato dell'arte

2.1 Il fenomeno del crosstalk

La tecnica di cancellazione del crosstalk è stata introdotta per la prima volta da Bauer 1961 [3] messa in pratica da Schoeder and Atal 1963 [4] e poi usata da Schroeder 1973 [5] per la riproduzione all'interno di sale concerti.

Ci sono principalmente due approcci usati all'interno delle sale: la prima riguarda i sistemi offline per condizioni di ascolto fisse ovvero un algoritmo di fast deconvolution basato su fast Fourier transform [9], un metodo di time domain deconvolution basato su filtri di Wiener deterministici e stocastici [10] e un minimax approximation [11] o least square [12].

La seconda riguarda sistemi online per applicazioni in tempo reale cioè un algoritmo least mean square (LMS) [13] e delle varianti di LMS dove lo step size del filtro adattativo varia sulla base dell'effetto di masking desiderato [14].

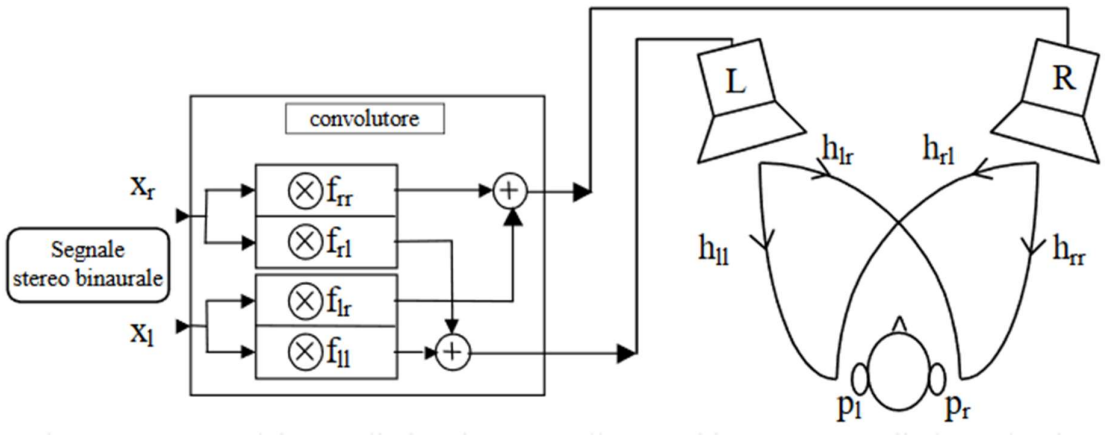

Figura 4 Percorso dei segnali che giungono alle orecchie attraverso gli altoparlanti

In Figura 4 è mostrato un generico steup utilizzato per la cancellazione del crosstalk. Le HRTF simulano il percorso dagli speaker alle orecchie dell'ascoltatore e servono per ricavare i filtri di cancellazione del crosstalk.

Nell'ambiente reale, il filtraggio avviene solo con i filtri f, mentre il filtraggio con le HRTF è rappresentato semplicemente dal percorso effettivo del suono attraverso l'ambiente fino all'ascoltatore. Per queste ragioni, è necessario usare un algoritmo che ritrasformi i segnali che arrivano alle orecchie, in modo che sull'orecchio sinistro arrivi solo xl e sull'orecchio destro arrivi solo xr. Se sono state misurate le quattro risposte all'impulso fra gli altoparlanti e le orecchie (denominate rispettivamente hll, hlr, hrl ed hrr), è possibile calcolare le quattro risposte all'impulso "inverse", denominate fll, flr, frl e frr, capaci di operare l'opportuno pre-filtraggio. Bisogna quindi misurare le quattro risposte all'impulso relative agli altoparlanti e alla testa dell'ascoltatore, il cui effetto è così completamente eliminato dal segnale audio da riprodurre.

2.2 Cancellazione del crosstalk

Restando a quanto detto prima, grazie a questa tecnica viene eliminato il crosstalk e qualsiasi altro filtraggio indesiderato che è dovuto alla risposta in frequenza degli altoparlanti. La funzione di trasferimento relativa alla testa è già compresa nelle risposte all'impulso usate per la convoluzione binaurale, ma il segnale che arriva dagli altoparlanti interferisce nuovamente con la testa dell'ascoltatore, e in questo modo il filtraggio dovuto alla testa viene operato due volte. Viene così eliminata anche questa funzione di trasferimento evitando il doppio filtraggio.

La cancellazione della diafonia è necessaria per regolare il livello sonoro che raggiunge ogni orecchio, quando si visualizza il suono virtuale su due altoparlanti. In passato sono state proposte varie cancellazioni di interferenze. Atal e Schroeder hanno messo in pratica l'algoritmo di cancellazione della diafonia analogica in [2]. Damaske, e Cooper e Bauck l'hanno perfezionato in [6] usando algoritmi di cancellazione di interferenze digitali. Idealmente, la cancellazione della diafonia dovrebbe trovare la matrice inversa della matrice della funzione di trasferimento acustico (ATF) da ciascuno degli altoparlanti a ciascuno dei timpani dell'ascoltatore [7,8]. Non è tuttavia garantito che la matrice ATF sia in minima fase e invertibile, quindi l'inversione diretta potrebbe non essere possibile. Un algoritmo di inversione adattivo può produrre un'approssimazione della matrice inversa della matrice ATF, anche se è richiesta una certa quantità di tempo di allenamento per la convergenza. Quando ascoltiamo la musica, l'ambiente di ascolto cambia spesso a causa del movimento dell'ascoltatore e della variazione nell'ambiente di ascolto. Questo fa cambiare la matrice ATF e l'algoritmo di cancellazione della diafonia deve adattarsi all'ambiente che cambia.

Il crosstalk cancellation viene quindi utilizzato per ottenere la separazione del canale tra le orecchie, meglio spiegato in figura 4.

Da qui è facile vedere che questo sistema può essere descritto nel dominio di frequenza come segue [1] (z = $e^{-}j\omega$):

$$pl(z) = Hll(z)xl(z) + Hrl(z)\,xr(z)$$
$$pr(z) = Hlr(z)xl(z) + Hrr(z)xr(z)$$

dove pl(z) e pr(z) sono rispettivamente i segnali che arrivano alle orecchie sinistra e destra e xl(z) e xr(z) sono i canali destro e sinistro di un segnale binaurale. Il percorso della diafonia può essere annullato (o almeno notevolmente attenuato) con un'adeguata struttura del filtro, come illustrato nella parte superiore della figura.

Questa struttura del filtro deve essere sempre posizionata tra il segnale in ingresso e gli altoparlanti e può essere rappresentata come matrice f, con conseguente percorso di trasmissione completo:

$$p(z) = H(z)f(z)x(z).$$

La corretta riproduzione binaurale è ottenuta se $e(z) = x(z) * e^{j}\omega\Delta$, dove Δ è un ritardo il quale tiene conto del ritardo acustico tra altoparlanti e posizione dell'ascoltatore. Così, la soluzione alla matrice di cancellazione crosstalk è data da : $f(z) = H^{-}1\ (z) * e^{j}\omega\Delta$.

La riproduzione di segnali binaurali non deve necessariamente avvenire attraverso due altoparlanti. Se si utilizzano più di due altoparlanti, la matrice H non sarà più una matrice quadrata e quindi non sarà invertibile. In questo caso, l'inversa può essere utilizzata, dando

$$f(z) = (H^{H}(z)H(z))^{-}1\ H^{H}(z) * e^{j}\omega\Delta$$

dove (*)H rappresenta l'operazione coniugata di trasposizione.

Questa soluzione nel dominio della frequenza fornirà la migliore separazione possibile del canale per una data configurazione dell'altoparlante ascoltatore. Ha, tuttavia, alcuni inconvenienti. La separazione del canale è realizzata per mezzo dell'interferenza costruttiva e distruttiva dell'onda.

Per alcune frequenze, un'ampiezza molto alta è richiesta solo per essere successivamente annullata alle orecchie dell'ascoltatore.
Il guadagno complessivo del filtro CTC deve essere ridotto per evitare il clipping a queste frequenze e con ciò la gamma dinamica del segnale binaurale riprodotto può essere drasticamente ridotta.
Inoltre, se calcolati utilizzando la Discrete Fourier Transform, i filtri CTC ottenuti soffriranno di aliasing ciclico e di non-causalità. Possibili soluzioni sono estendere la risoluzione di frequenza della matrice di trasferimento originale H e applicare la finestra temporale, applicare un vincolo di regolarizzazione o calcolare i filtri inversi nel dominio del tempo.

Mentre parlando nell'ambito dell'Ambiophonic, che prevede un elevato numero di altoparlanti posti intorno alla zona di ascolto, è stata sviluppata una tecnica di cancellazione del crosstalk chiamata RACE (recursive ambiophonic crosstalk elimination) che non prevede l'utilizzo delle HRTF ma un algoritmo ricorsivo. Consiste nell'eliminazione del segnale dall'altoparlante sinistro che raggiunge in maniera indesiderata l'orecchio destro. Questo viene messo in atto tramite l'inversione del segnale originale, applicando un ritardo e una giusta attenuazione. La stessa cosa vale per il segnale percepito dall'orecchio sinistro, ma anche in questo caso il segnale genererà un altro disturbo percepito dall'orecchio destro creando una cancellazione di terzo ordine. Per incombere a ciò, l'algoritmo deve essere manipolato da alcuni parametri quali l'attenuazione d 2-3 dB e il ritardo tra i 60 e 100 μs.

Capitolo 3: Descrizione dell'algoritmo implementato

Nel seguente capitolo viene descritto l'algoritmo per il calcolo dei filtri di cancellazione del crosstalk. Innanzitutto, si va a definire un dominio di frequenza z, dopo di questo vengono generati i segnali che arrivano all'orecchio destro e sinistro dati dalla moltiplicazione del percorso dagli speaker alle orecchie delle HRTF con il segnale stereo binaurale. Da qui il crosstalk può essere annullato con la matrice **F** contenente i filtri di cancellazione. Con riferimento alla Figura 4, il percorso di trasmissione della struttura del filtro è dato dal prodotto della matrice **H** delle HRTF, della matrice **F** dei i filtri e dai segnali in ingresso x, ovvero:

$$[\boldsymbol{p}_l\ \boldsymbol{p}_r] = [\boldsymbol{x}_l\ \boldsymbol{x}_r]\begin{bmatrix} f_{rr} & f_{r_l} \\ f_{lr} & f_{ll} \end{bmatrix}\begin{bmatrix} h_{rr} & h_{rl} \\ h_{lr} & h_{ll} \end{bmatrix} \cong [\boldsymbol{x}_l\ \boldsymbol{x}_r].$$

Dopo aver effettuato queste operazioni si riesce ad ottenere una corretta riproduzione data dai segnali in ingresso per il dominio della frequenza moltiplicati a loro volta per il ritardo acustico tra gli altoparlanti e l'ascoltatore. La matrice F è data dall'inverso della matrice H per la corretta riproduzione binaurale. In questa tesi viene implementato un metodo nel dominio della frequenza per il calcolo della matrice **F.** Per ogni k-esimo bin in frequenza viene quindi calcolata la matrice F(k) nel seguente modo:

$$F(k) = [H^H(k)H(k)]^{-1}H^H(k),$$

dove $H^H(k)$ rappresenta la matrice hermitiana di *H(k).*

In conclusione l'algoritmo fa si che i segnali vengano opportunamente trasformati prima che arrivino alle orecchie, di modo che all'orecchio destro arrivi xr e al sinistro xl.

L'effetto che il filtro ctc, implementato nel codice, deve avere sul sistema è quello di rendere il più piccoli possibili i termini crosslaterali della matrice H.

Per semplificare le operazioni per determinare il filtro si possono considerare uguali tra loro i percorsi ipsolaterali **H**11 = **H**22 = **H**i e controlaterali **H**21 = **H**12 = **H**c e considerare sufficientemente elevata la differenza in modulo tra le risposte sulla diagonale rispetto a quelle fuori diagonale (fenomeno dovuto *all'head shadowing*, ovvero l'attenuazione introdotta dal passaggio delle onde attraverso il cranio), tale che $1 \,/\, \left(1 - \frac{H_c^2}{H_i^2}\right) \approx 1$. Così facendo, la matrice $\boldsymbol{H^{-1}}$ diventa $\begin{bmatrix} 1 & -H_c \\ -Hc & \overline{Hi} \\ \overline{H_\iota} & 1 \end{bmatrix}$.

Capitolo 4: Implementazione in C

L'algoritmo implementato in C è costituito da files header e source. Negli header c'è la dichiarazione di tutte le variabili e dei cicli. I source, invece, sono costituiti dall'allocazione, deallocazione di variabili e dall'esecuzione dei cicli. Distinguiamo i principali file quali il Plugin.cpp e la myLib.cpp. Nel primo sono inizializzate tutte le variabili a zero, vengono riportati i path della cartella, dichiarato il fattore per cui vanno divisi i campioni di ingresso di dimensione 2^15 per una scheda audio a 16 bit. In Figura 5 è mostrato il diagramma di flusso dell'algoritmo. Nella classe process viene implementata la tecnica dell'overlap and save (OSL) per il filtraggio in tempo reale dei segnali di ingresso con la matrice dei filtri di cancellazione **F** e la matrice delle risposte impulsive **H**. Si conclude con la somma delle componenti dei segnali destro e sinistro per poter cambiare il formato da Pack a C. Viene eseguita poi la FFT inversa e copiati i nuovi valori dei vettori nel buffer di uscita, chiudendo il for.

Nella classe Init vengono allocate le variabili e richiamati i file.dat con i valori della matrice H e inizializzata la struttura per la FFT.

Nella classe Delete vengono deallocate tutte le variabili dichiarate nella init.

Nella Mylib.cpp viene riportato il ctc_gen ovvero la funzione che permette di calcolare i filtri di cancellazione a partire delle HRTF.

Per una maggiore efficienza del codice sono state utilizzate le librerie ipp.h e le cmath.

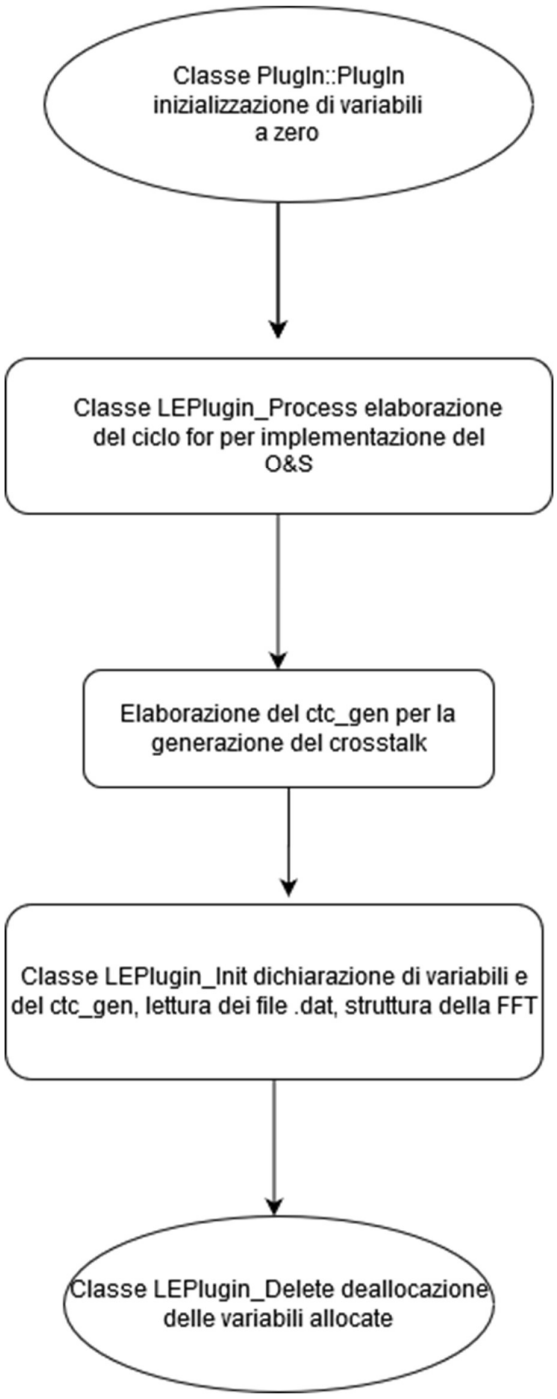

Figura 5 Diagramma di flusso dell'algoritmo

4.1 Introduzione al NU-tech

L'implementazione in tempo reale basata su PC è stata sviluppata utilizzando la piattaforma NU-Tech, una piattaforma software adatta per l'elaborazione audio in tempo reale direttamente su un PC. Un'architettura plug-in semplice e un kit di sviluppo software libero (SDK) consentono allo sviluppatore di scrivere NU-Tech Satellites (Nutss) in C++ e inserirli immediatamente nell'ambiente di progettazione dell'interfaccia grafica. In questo contesto è stato realizzato un plug-in che è in grado di calcolare i parametri di cancellazione della diafonia e di filtrare il segnale in ingresso, concentrandosi sulle operazioni di elaborazione in tempo reale, la libreria Intel Integrates Primitives (IPP) (libreria ad alte prestazioni per il calcolo matematico) è stata utilizzata al fine di massimizzare le prestazioni computazionali.

4.2 Codice C

L'implementazione dell'algoritmo è costituita da due tipi di file: header e source files.

Negli header abbiamo quei files utili a dichiarare le librerie utilizzate e le variabili del linguaggio e delle ipp (intel integrated performance primitives). Nei source files abbiamo il Plugin.cpp rilevante per la spiegazione dell'algoritmo, costituito da classi. La prima classe che incontriamo è la classe PlugIn::PlugIn nella quale sono inizializzate tutte le variabili a zero e dichiarati i puntatori per inizializzare la struttura della FFT.

La classe LEPlugin_Process contiene l'elaborazione del ciclo for per l'implementazione dell'Overlap&Save, inoltre vengono inserite le HRTF nella moltiplicazione in frequenza.

```
//Traslo bl e br di FrameSize volte
ippsCopy_64f(bl + FrameSize, bl, FrameSize);
ippsCopy_64f(br + FrameSize, br, FrameSize);
```

Vengono traslati i vettori bl e br di Framesize volte e gli ingressi vengono copiati in bl e br con la funzione copy delle ipp. Poi viene eseguita una normalizzazione dei valori per un fattore 32768 (2^15) dipendente dal tipo di scheda audio utilizzata(in questo caso 16bit).

```
//Copio i nuovi campioni in bl e br
ippsCopy_64f((double*)(*Input[0]).DataBuffer, bl + FrameSize, FrameSize);
ippsCopy_64f((double*)(*Input[1]).DataBuffer, br + FrameSize, FrameSize);

ippsDivC_64f_I(32768, bl + FrameSize, FrameSize);
ippsDivC_64f_I(32768, br + FrameSize, FrameSize);
```

Viene effettuata la trasformata di Fourier dei due vettori col formato Pack delle ipp ed eseguita la moltiplicazione in frequenza con i filtri C di cancellazione del crosstalk.

```
//FFT di bl e br
ippsFFTFwd_RToPack_64f(bl, Bl, pSpec, pBuffer);
ippsFFTFwd_RToPack_64f(br, Br, pSpec, pBuffer);

//Moltiplicazione in frequenza
ippsMulPack_64f(Bl, C11, BlxC11, fftLen);
ippsMulPack_64f(Br, C21, BrxC21, fftLen);
ippsMulPack_64f(Br, C22, BrxC22, fftLen);
ippsMulPack_64f(Bl, C12, BlxC12, fftLen);
//Somma delle componenti per i segnali sinistro e destro
ippsAdd_64f(BlxC11, BrxC21, Yl, fftLen);
ippsAdd_64f(BrxC22, BlxC12, Yr, fftLen);
```

Dopo aver effettuato la moltiplicazione in frequenza con la chiamata MulPack a 64 bit vengono sommate le componenti dei vettori rispettivamente per il segnale destro e sinistro ottenendo i vettori Yl e Yr.

```
//Moltiplicazione in frequenza
ippsMulPack_64f(Yl, H11, YlxH11, fftLen);
ippsMulPack_64f(Yr, H21, YrxH21, fftLen);
ippsMulPack_64f(Yr, H22, YrxH22, fftLen);
ippsMulPack_64f(Yl, H12, YlxH12, fftLen);
//Somma delle componenti per i segnali sinistro e destro
ippsAdd_64f(YlxH11, YrxH21, Zl, fftLen);
ippsAdd_64f(YrxH22, YlxH12, Zr, fftLen);
```

Si rimoltiplica in frequenza per i vettori H per ottenere poi la somma delle componenti in uscita Zl e Zr, unici due output. Infine, viene eseguita la FFT inversa per ricavare i segnali nel dominio del tempo e prima di aggiornare l'output viene moltiplicato il fattore 2^15.

```
//FFT inversa per tornare nel dominio del tempo
ippsFFTInv_PackToR_64f(Zl, yl, pSpec, pBuffer);
ippsFFTInv_PackToR_64f(Zr, yr, pSpec, pBuffer);


//ippsMulC_64f_I(32768, yl, fftLen);
//ippsMulC_64f_I(32768, yr, fftLen);
ippsMulC_64f_I(32768, yl + FrameSize, FrameSize);
ippsMulC_64f_I(32768, yr + FrameSize, FrameSize);

//Copio i valori nel buffer d'uscita
ippsCopy_64f(yl + FrameSize, (double*)(*Output[0]).DataBuffer, FrameSize);
ippsCopy_64f(yr + FrameSize, (double*)(*Output[1]).DataBuffer, FrameSize);

return COMPLETED;
```

Vengono copiati i valori finali nel buffer di uscita e chiuso il ciclo for

Il file Mylib.cpp contiene la funzione ctc_gen per la generazione dei filtri di cancellazione del crosstalk, a partire dalle risposte impulsive. Nella funzione

ctc_gen vengono allocati i vettori e cambiato il formato da Pack a vettori complessi.

```
void ctc_gen(double *H11, double *H12, double *H21, double *H22, double *C11, double *C12, double *C21, double *C22, int N) {
	Ipp64fc *H11c, *H12c, *H21c, *H22c, *C11c, *C12c, *C21c, *C22c;
	H11c = 0;

	H11c = ippsMalloc_64fc(N / 2 + 1);
	PackToC(H11, H11c, N);
	H12c = 0;
	H12c = ippsMalloc_64fc(N / 2 + 1);
	PackToC(H12, H12c, N);
	H21c = 0;
	H21c = ippsMalloc_64fc(N / 2 + 1);
	PackToC(H21, H21c, N);
	H22c = 0;
	H22c = ippsMalloc_64fc(N / 2 + 1);
	PackToC(H22, H22c, N);

	C11c = 0;
	C11c = ippsMalloc_64fc(N / 2 + 1);
	C12c = 0;
	C12c = ippsMalloc_64fc(N / 2 + 1);
	C21c = 0;
	C21c = ippsMalloc_64fc(N / 2 + 1);
	C22c = 0;
	C22c = ippsMalloc_64fc(N / 2 + 1);
```

Viene aperto un ciclo for dove per un migliore utilizzo si fa l'inversa della matrice H e il suo complesso coniugato

```
	for (int i=0; i<N/2+1; i++){
	Ipp64fc H[4], Ht[4], a[2], b[2], c[2], d[2], HtH[4], delta, dst[2], val, HtH_inv[4], C[4], e[2], f[2], g[2], m[2];
		H[0] = H11c[i];
		H[1] = H12c[i];
		H[2] = H21c[i];
		H[3] = H22c[i];
```

Viene trasposta la matrice

```
		Ht[0] = H[0];
		Ht[1] = H[2];
```

```
Ht[2] = H[1];
Ht[3] = H[3];

ippsConj_64fc_I(Ht, 4);
```

Viene fatto il suo complesso coniugato

```
a[0] = Ht[0], a[1] = Ht[1];
b[0] = Ht[2], b[1] = Ht[3];
c[0] = H[0], c[1] = H[2];
d[0] = H[1], d[1] = H[3];
```

Vengono creati dei vettori di appoggio a,b,c,d per eseguire la moltiplicazione matriciale $H^H(k)H(k)$, vista nel capitolo precedente e salvata nel vettore HtH.

```
ippsDotProd_64fc(a, c, 2, &HtH[0]);
ippsDotProd_64fc(a, d, 2, &HtH[1]);
ippsDotProd_64fc(b, c, 2, &HtH[2]);
ippsDotProd_64fc(b, d, 2, &HtH[3]);

ippsMul_64fc(&HtH[0], &HtH[3], &dst[0], 1);
ippsMul_64fc(&HtH[1], &HtH[2], &dst[1], 1);
ippsSub_64fc(&dst[1], &dst[0], &delta, 1);

val.re = -1.0; val.im = 0.0;

ippsMulC_64fc_I(val, &HtH[1], 1);
ippsMulC_64fc_I(val, &HtH[2], 1);
```

Dopo aver effettuato operazioni di moltiplicazione e addizione, viene fatto il determinante per rendere possibile l'inversione e quindi calcolare il termine $[H^H(k)H(k)]^{-1}$, salvato nel vettore Hth_inv.

```
ippsDiv_64fc(&delta, &HtH[3], &HtH_inv[0], 1);
ippsDiv_64fc(&delta, &HtH[1], &HtH_inv[1], 1);
ippsDiv_64fc(&delta, &HtH[2], &HtH_inv[2], 1);
ippsDiv_64fc(&delta, &HtH[0], &HtH_inv[3], 1);
```

A questo punto vengono creati altri vettori di appoggio e,g,f,m per eseguire l'ultima moltiplicazione matriciale che permette di ricavare la matrice C(k), definita come $C(k) = [H^H(k)H(k)]^{-1}H^H(k)$.

```
e[0] = HtH_inv[0], e[1] = HtH_inv[1];
g[0] = HtH_inv[2], g[1] = HtH_inv[3];
f[0] = Ht[0], f[1] = Ht[2];
```

```
m[0] = Ht[1]; m[1] = Ht[3];

ippsDotProd_64fc(e, f, 2, &C[0]);
ippsDotProd_64fc(e, m, 2, &C[1]);
ippsDotProd_64fc(g, f, 2, &C[2]);
ippsDotProd_64fc(g, m, 2, &C[3]);
```

I nuovi valori della matrice vengono copiati in nuove variabili chiamate C che vanno a creare la matrice definitiva

```
C11c[i] = C[0];
C12c[i] = C[1];
C21c[i] = C[2];
C22c[i] = C[3];
```

Si combinano i segnali finali tramite le HRTF, utilizzando i filtri di cancellazione del crosstalk.

La classe LEPlugin_Init contenente la dichiarazione di variabili e del ctc_gen, la lettura dei file .dat e la struttura della FFT.

L'ultima classe LEPlugin_delete dove sono inserite le deallocazione delle variabili allocate in precedenza.

Capitolo 5: Test sperimentali

L'algoritmo implementato è stato testato attraverso diversi test sperimentali riportati nel seguente capitolo. Sono stati eseguiti in totale 5 test considerando diversi segnali di ingresso per la validazione dell'algoritmo:

- Test 1: rumore bianco nel canale destro e silenzio nel sinistro per confrontare i segnali che raggiungono l'ascoltatore.
- Test 2: rumore bianco nel canale sinistro e silenzio nel destro per confrontare i segnali che raggiungono l'ascoltatore.
- Test 3: canzone stereo per fare un confronto delle uscite e una prova di ascolto soggettiva.
- Test 4: Canzone nel canale sinistro e silenzio nel canale destro per confrontare il segnale all'orecchio sinistro con l'ingresso e per l'ascolto.
- Test 5: Due tracce diverse nel canale destro e nel canale sinistro per il confronto tra uscite e ingressi e prova di ascolto.

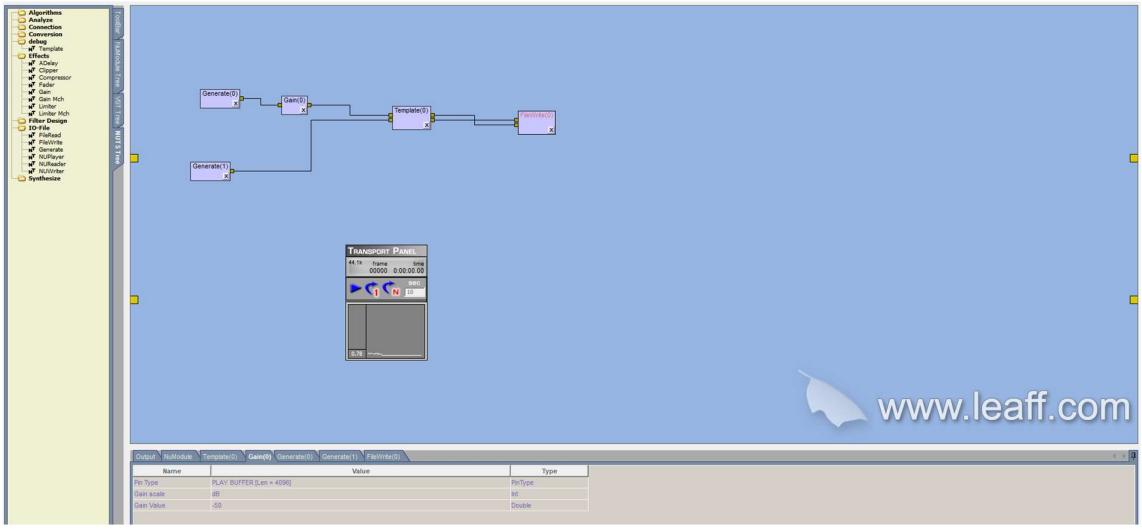

Figura 6 Schema nu-tech del test 1: rumore a destra

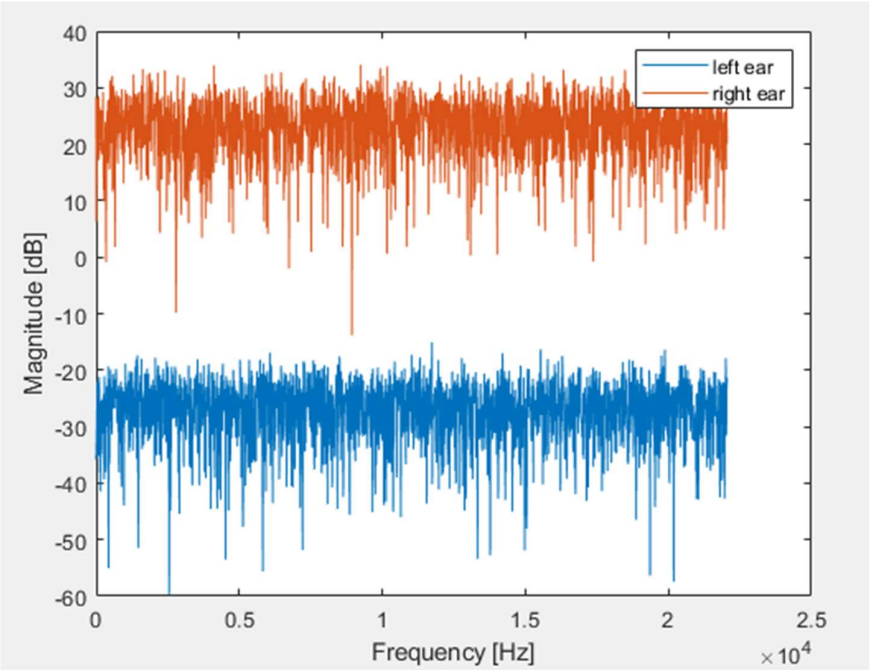

Figura 7 Plottaggio del test 1 (rumore a destra) in matlab

In Figura 6 è riportato lo schema Nu-Tech del test 1. Il Test è stato effettuato caricando sull'interfaccia Nu-Tech un generate (0) attenuato di -50dB collegato ad un gain che permette di monitorare la scala magnitude e un generate (1) impostato a white noise (rumore bianco). Questi vengono collegati al Template dove è stato caricato il path della cartella contenente le HRTF e collegato ad un FileWrite per poter salvare le uscite in un file .wav. Le uscite vengono plottate attraverso Matlab con opportuno codice (plotgraf) per ottenere il risultato mostrato in Figura 7. I due canali, destro e sinistro, sono correttamente separati di circa 50 dB, esattamente come gli ingressi desiderati.

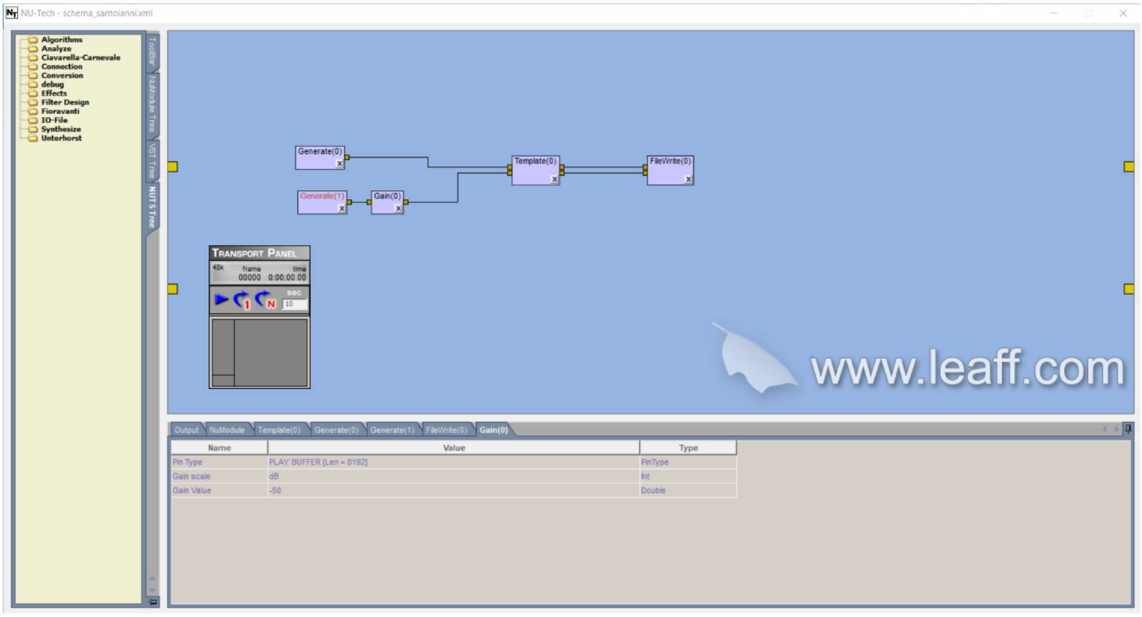

Figura 8 Schema nu-tech del test 2: rumore a sinistra

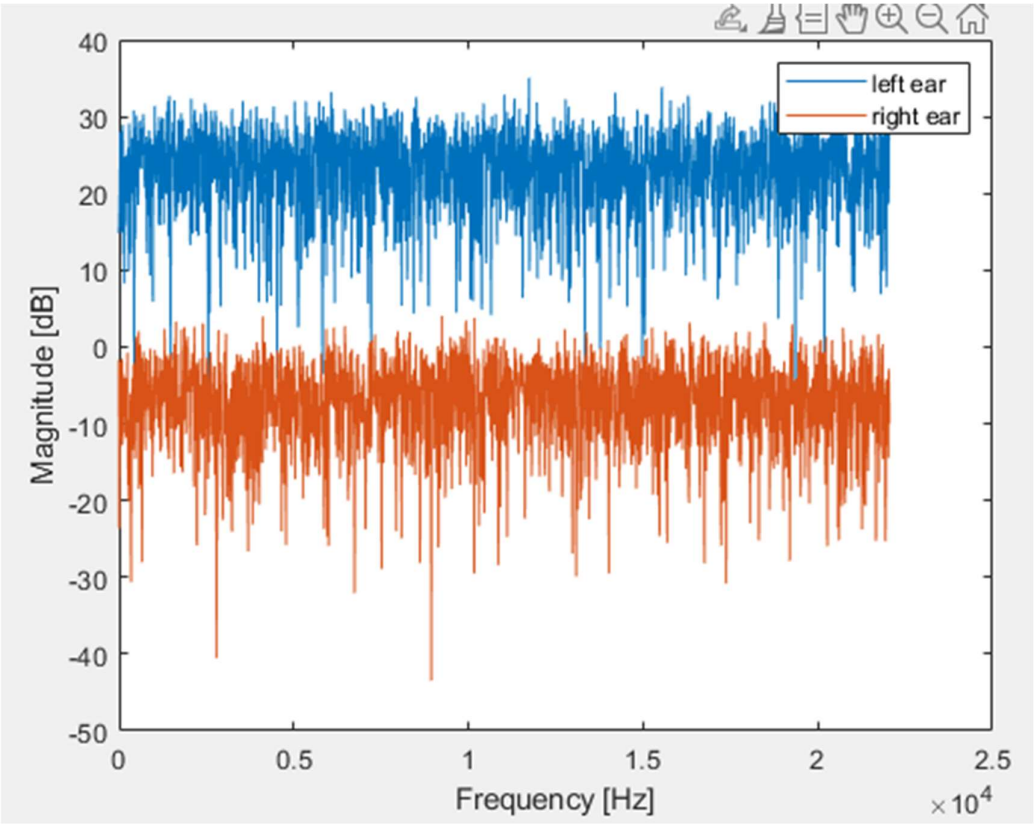

Figura 9 Plottaggio del test 2 (rumore a sinistra) in matlab

In Figura 8 è mostrato lo schema Nu-Tech utilizzato per eseguire il test 2. Il test è stato effettuato in nu-tech caricando sull'interfaccia un generate (1) attenuato di -50dB collegato ad un gain che permette di monitorare la scala magnitude e un generate (0) impostato a white noise. Questi vengono collegati al Template dove è stato caricato il path della cartella contenente le HRTF e collegato ad un FileWrite per poter generare un .wav. Questo .wav viene plottato attraverso Matlab con opportuno codice (plotgraf) per ottenere il risultato visto in Figura 9. I due canali, destro e sinistro, anche in questo caso, sono correttamente separati, anche se in questo caso il segnale che raggiunge l'orecchio destro è meno attenuato rispetto al segnale sinistro ottenuto nel test 1.

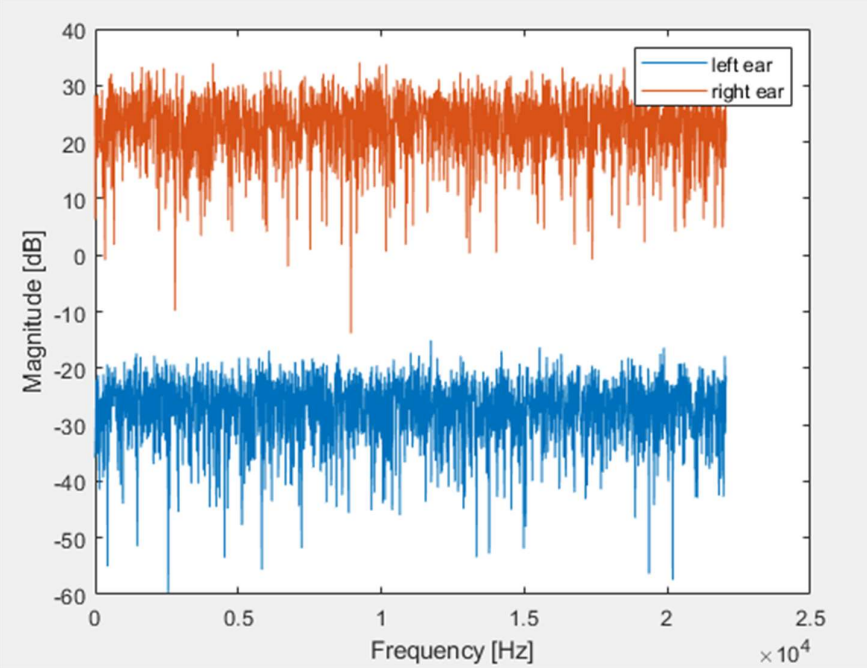

Figura 7 Plottaggio del test 1 (rumore a destra) in matlab

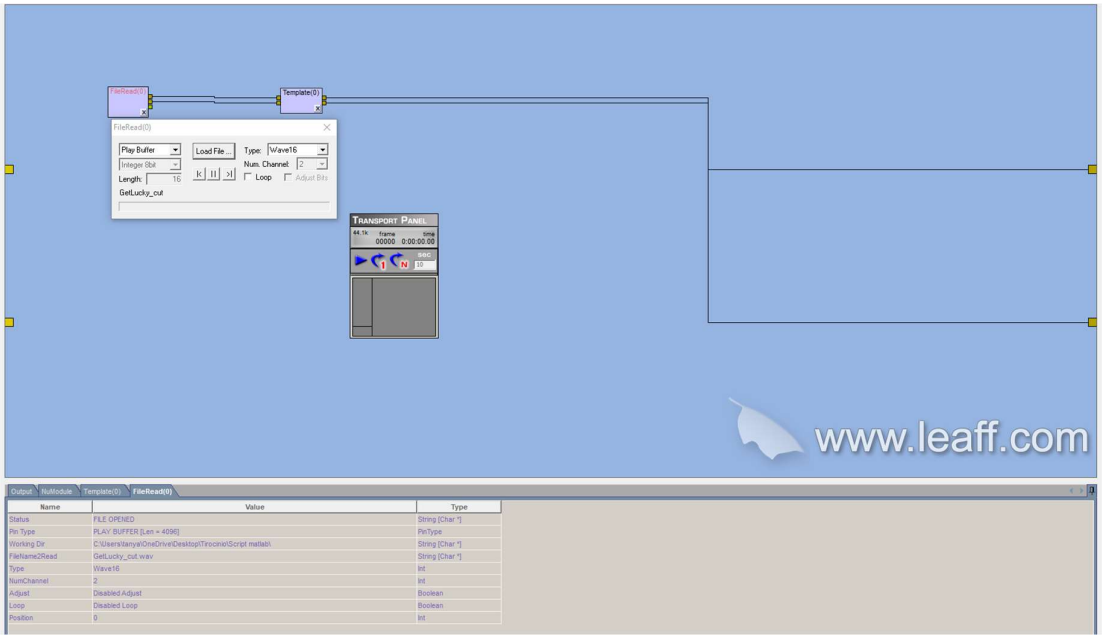

Figura 10 Schema nu-tech del test 3: canzone stereo

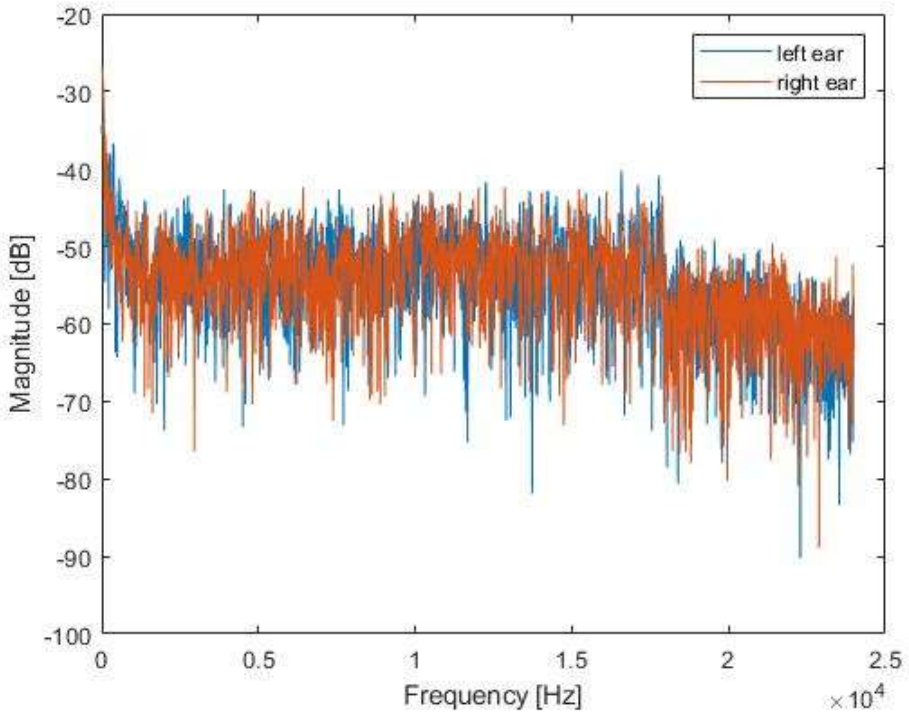

Figura 11 Plottaggio in matlab delle uscite ricavate dal test 3(canzone stereo)

In Figura 10 è mostrato lo schema Nu-tech utilizzato per effettuare il test 3, considerando come ingresso una canzone stereo. Tramite la piattaforma nutech utilizzando il nuts di FileRead , è stato caricato un brano, collegato correttamente con il template e poi in output sulla piattaforma.

In Figura 11 sono riportate le curve di uscita ottenute e graficate su matlab. È stata eseguita anche una prova di ascolto che ha dimostrato una corretta riproduzione del brano.

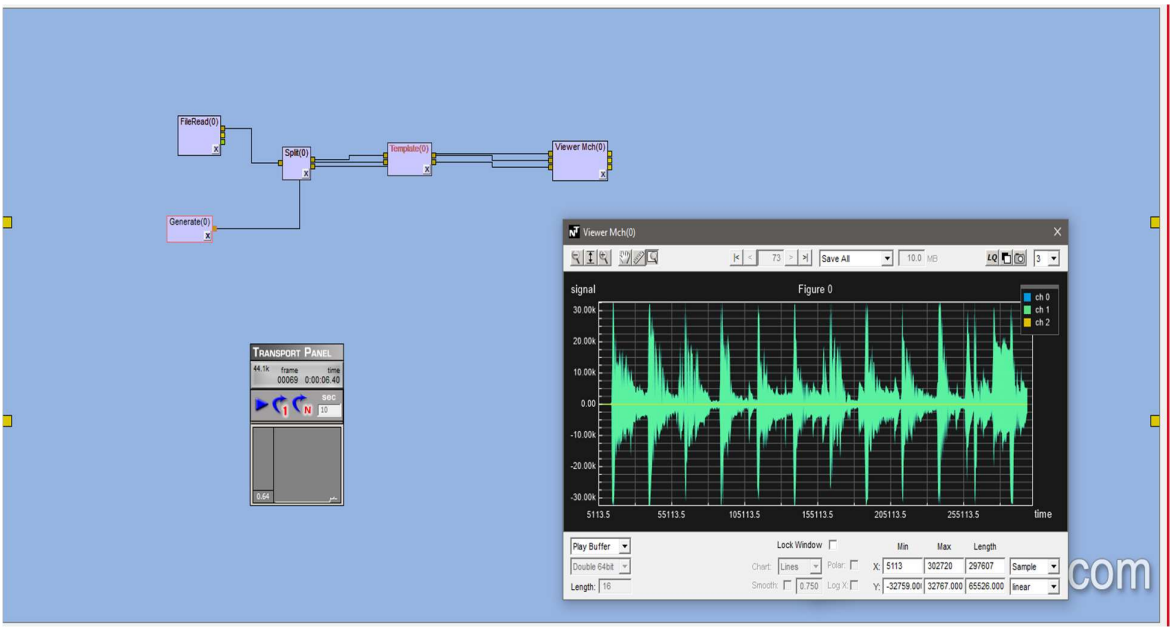

Figura 12 Schema nu-tech del test 4(canzone a sinistra)

In Figura 12 è mostrato lo schema nu-tech del test 4, realizzato con uno split e un viewer multichannel. In questo caso si considera una canzone al canale sinistro e silenzio nel canale destro. Il viewer del Nu-tech permette di mostrare: l'ingresso sinistro in blu (ch0), l'uscita sinistra in verde (ch1) e l'uscita destra in giallo (ch2). L'ingresso sinistro e l'uscita sinistra sono totalmente sovrapposte, mentre l'uscita destra è praticamente nulla, come desiderato. Anche in questo caso è stata fatta

una prova di ascolto che ha confermato i risultati ottenuti dai grafici, quindi il suono è stato percepito quasi completamente all'orecchio sinistro.

Il test 5 viene effettuato con due brani rispettivamente posti in due canali diversi e in Figura 13 è mostrato lo schema Nu-Tech. L'ingresso destro è stato confrontato con l'uscita destra (ch0 in blu e ch1 in verde del grafico in basso rispettivamente) e l'ingresso sinistro con l'uscita sinistra (ch0 in blu e ch1 in verde del grafico in alto rispettivamente). Anche qui le curve si sovrappongono, quindi la separazione dei canali avviene perfettamente. Infatti, all'ascolto si riesce a percepire sull'orecchio sinistro il segnale che proviene dal segnale acustico sinistro, come riporta l'esempio, trattasi di un brano pop, e sull'orecchio destro il segnale proveniente dal canale destro nel quale è caricato un brano classico.

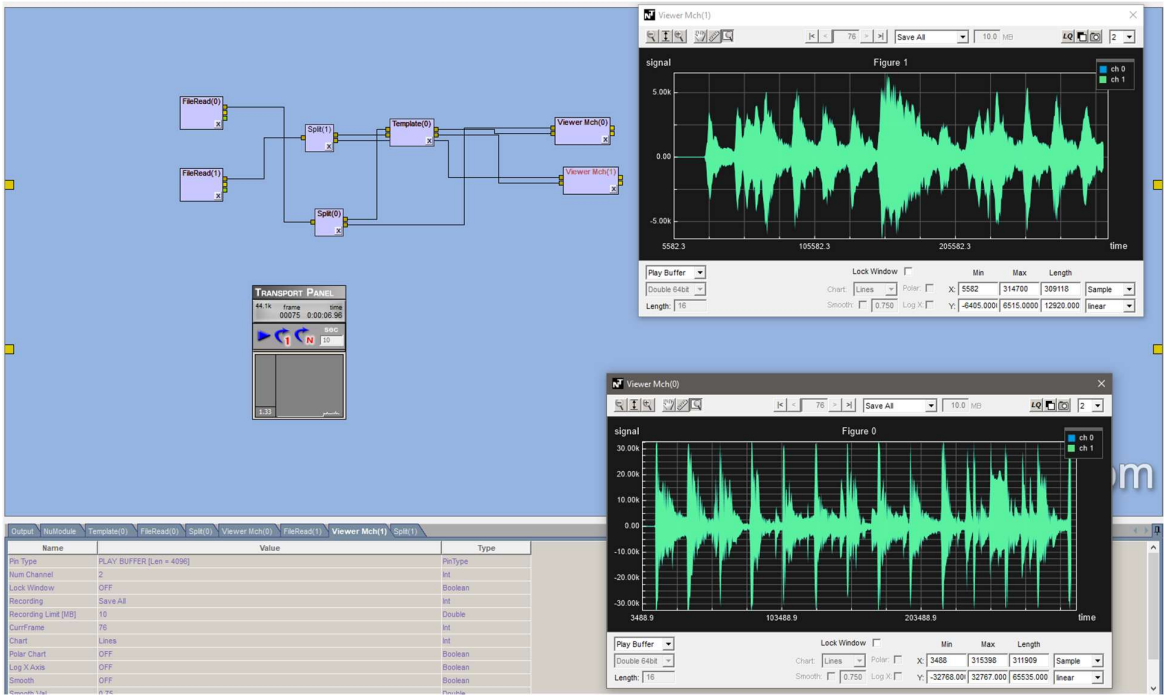

Figura 13 Schema nu-tech del test 5 con due brani su canali separati

Conclusioni

La presente tesi ha riguardato l'implementazione real-time di un algoritmo di cancellazione del crosstalk. Per l'implementazione è stata sfruttata la piattaforma Nu-Tech, che permette di programmare dei plug-in in linguaggio C++. Il codice è risultato correttamente funzionante. I test sperimentali hanno riguardato il confronto fra i due segnali (destro e sinistro) che raggiungono le orecchie dell'ascoltatore e il confronto fra le uscite e gli ingressi. I risultati sperimentali hanno dimostrato una buona cancellazione del crosstalk sia attraverso dei test oggettivi che attraverso dei test soggettivi informali.

Sviluppi futuri saranno portati avanti mediante l'acquisizione di diverse HRTF, considerando diverse posizioni dell'ascoltatore e degli altoparlanti in modo da valutare l'algoritmo presentato in funzione del set-up di ascolto.

RIFERIMENTI

[1] Review Of The Crosstalk Cancellation _Lter Technique, Bruno Masiero, Janina Fels, Michael Vorlander

[2] A Novel Adaptive Crosstalk Cancellation Using Psychoacoustic Model For 3d Audio, JunSeong Kim, SangGyun Kim, and Chang D.Yoo

[3] BE. B. Bauer, "Stereophonic Earphones and Binaural Loudspeakers," J. Audio Eng. Soc., vol. 9, no. 2, pp. 148-151, (1961 April.)

[4] Schroeder, M. R., and B. S. Atal. "Computer simulation of sound transmission in rooms." Proceedings of the IEEE 51.3 (1963): 536-537

[5] Schroeder, Manfred R. "Computer models for concert hall acoustics." American Journal of Physics 41.4 (1973): 461-471

[6] Schroeder, M. R., and B. S. Atal. "Computer simulation of sound transmission in rooms." Proceedings of the IEEE 51.3 (1963): 536-537

[7] P. Damaske, “Head-related two-channel stereophony with loudspeaker reproduction”, J. Acoust. Soc. Am. 50, 1109-1115 (1971)

[8] D.H. Cooper and J.L. Bauck, “Prospects for transaural recording”, J. Audio Eng. Soc. 37 (1/2), 3-19 (1989)

[9] O. Kirkeby, P. A. Nelson, H. Hamada, and F. Orduna-Bustamante, “Fast deconvolution of multichannel system using regularization” IEEE Trans. Speech Audio Processing, vol. 6, pp. 189-194, 1998

[10] Sang-Myeong Kim and Semyung Wang, “A Wiener fiter approach to the binaural reproduction of stereo sound” J. Acoust. Soc. Am., vol. 114(6), pp. 3179-3188, Dec. 2003

[11] I. Rao. Harsha, V. John Mathews, and Young- Cheol Park, “Inverse fiter design using minimax approximation techniques for 3-D audio” Proc. IEEE ICASSP, vol. 5, 14-19 May 2006.

[12] Yiteng (Arden) Huang, Jacob Benesty, and Jingdong Chen, “On crosstalk cancellation and equalization with multiple loudspeakers for 3-D sound reproduction” IEEE Signal Processing Letters, vol. 14, pp. 649-652, Oct. 2007

[13] L. Lim and C. Kyriakakis, “Multirate adaptive fitering for immersive audio” Proc. IEEE ICASSP, vol. 5, pp. 3357-3360, May 2001

[14] JunSeong Kim, SangGyun Kim, and Chang D.Yoo, “A novel adaptive crosstalk cancellation using psychoacoustic model for 3D audio” Proc. IEEE ICASSP, vol. 1, pp. 185-188, 15-20 Apr. 2007