UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI INGEGNERIA

Corso di Laurea Triennale
in Ingegneria Informatica e dell'Automazione

Studio e implementazione di un Algoritmo di Pitch Bending tramite filtri digitali All-pass

Study and implementation of a Pitch Bending Algorithm using All-pass digital filters

Relatore:

Prof. Stefano Squartini

Tesi di Laurea di:

Marco Ferramosca

Correlatore:

Prof. Leonardo Gabrielli

A.A. 2020/2021

Ai miei genitori, alla libertà e alle possibilità che mi hanno donato.

A mia sorella per esserci sempre stata.

A mia nonna per avermi insegnato i numeri.

Al mio amico Luigi per aver condiviso con me questo percorso.

A me, alla mia pazienza, al mio carattere.

A 24 Ore per non avermi mai abbandonato.

Indice

Indice delle figure

Indice delle tabelle

1 Introduzione

La seguente trattazione pone come argomento centrale lo studio del pitch-bending tramite differenti metodi. Il pitch-bending consiste in un'alterazione dell'intonazione di una singola nota che viene prodotta da uno strumento come una tastiera o una chitarra elettrica. Non è circoscritto solo a strumenti di questo genere, ma con l'avvento del digitale è possibile applicare quest'effetto a un suono qualsiasi. La tesi si concentrerà sulla simulazione di corde vibranti tramite circuiti digitali e l'applicazione del pitch-bending ad esse.

1.1 Corde Vibranti

Il modello delle corde vibranti rappresenta un argomento centrale nell'ambito musicale. Per descrivere in maniera efficiente tale fenomeno che tocca diversi strumenti, come per esempio il pianoforte, la chitarra e il violino, è importante rifarsi ai modelli proposti dalla fisica.

Innanzitutto, va determinata la differenza fra:

- *onda longitudinale*, le cui oscillazioni sono presenti nel verso di propagazione
- *onde trasversali*, le cui oscillazioni sono perpendicolari alla direzione di propagazione.

Consideriamo, quindi, un'onda trasversale che si sposta con una velocità v in un mezzo elastico unidimensionale (una corda). Stabilito il sistema di riferimento solidale con la corda otterremo che il mezzo risulta muoversi con una velocità $-v$. Andando a studiare in termini di forze, ciò che accade quando la corda attraversa un massimo si ottiene che la velocità risulta dipendere dalla tensione T e dalla densità di massa della corda ρ.

$$v = \sqrt{\frac{T}{\rho}} \qquad (1.1)$$

Introdotta nell'equazione di d'Alembert ci fornisce l'equazione della **corda vibrante**:

$$\frac{\partial^2 y}{\partial t^z} = v^2 \frac{\partial^2 y}{\partial x^2} \qquad (1.2)$$

Tra le soluzioni dell'equazione di d'Alembert ve ne è una in particolare che ci permette di trattare più facilmente il fenomeno, quella delle *onde armoniche* con equazione:

$$y_{\pm}(x,t) = A\cos(kx \mp +\varphi_{\pm}) \qquad A > 0 \qquad \varphi \in [0,2\pi] \qquad (1.3)$$

caratterizzata da un periodo $T = \frac{2\pi}{\omega}$ dalla lunghezza d'onda $\lambda = \frac{2\pi}{k}$ con k numero d'onda, pari al numero di oscillazioni in un tratto spaziale di 2π, A l'ampiezza dell'onda e φ fase iniziale dell'onda. Si parla, quindi, un'onda progressiva e di un'onda regressiva, ossia due onde con velocità opposta, uguale ampiezza e stessa fase inziale.

Dunque, si ha:

$$\begin{aligned} y_+(x,t) &= A\cos(kx - \omega t + \varphi_+) \qquad A > 0 \qquad \varphi \in [0,2\pi] \\ y_-(x,t) &= A\cos(kx + \omega t + \varphi_-) \qquad A > 0 \qquad \varphi \in [0,2\pi] \end{aligned} \qquad (1.4)$$

Introduciamo il principio di sovrapposizione che definisce che due o più onde della stessa natura che si propagano nello stesso mezzo danno luogo a un'interferenza costruttiva se sono in fase oppure a interferenza distruttiva se sono in controfase. Tale principio vale finché il mezzo risulta essere lineare e, considerato che nel caso di un mezzo elastico, è fondamentale che la forza di richiamo $F=-kR$ risulti essere proporzionale allo spostamento R, altrimenti entra in gioco la proprietà di non linearità.

Per il principio di sovrapposizione si ottiene:

$$y(x,t) = A\,[sen\,(\,\omega t + kx + \,\varphi_-) + sen\,(\,\omega t - kx + \,\varphi_+)] \qquad (1.5)$$

Tramite la formula di Prostaferesi si arriva a dire:

$$y(x,t) = 2A\ sen\ (\omega t + \varphi_1)\ cos\ (kx + \varphi_2)] \qquad (1.6)$$

da cui si nota come la dipendenza spaziale e la dipendenza temporale siano separate.

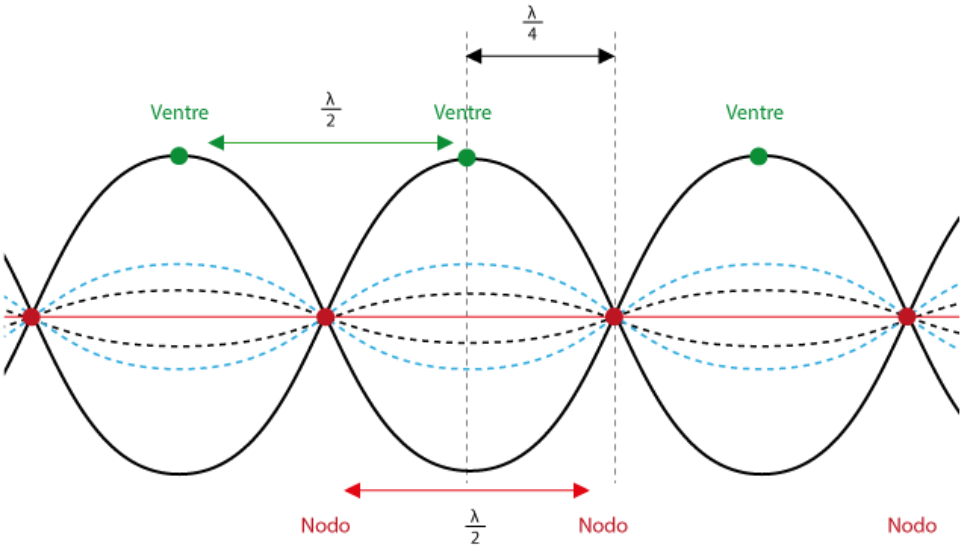


Figura 1 - Onde Stazionarie

Definiamo, quindi,

- nodi: punti in cui l'ampiezza è sempre nulla;
- ventri: punti in cui l'ampiezza è massima.

Due nodi consecutivi distano $\frac{\lambda}{2}$. Due ventri consecutivi distano $\frac{\lambda}{2}$. Un ventre e un nodo consecutivo distano $\frac{\lambda}{4}$.

Dall'equazione generale (1.2) si ottiene la regola generale per le onde stazionarie:

$$D = n(\lambda_n/2) \qquad \text{con n=numero di nodi} \tag{1.7}$$

Considerando la formula (1.3) si può riscrivere come:

$$y(x,t) = y_+\left(t - \frac{x}{v}\right) + y_-\left(t + \frac{x}{v}\right) \tag{1.8}$$

Discretizzando nel tempo e nello spazio si sceglie un intervallo di discretizzazione Δx sufficientemente piccolo per permetterci di descrivere al meglio tutti i punti della corda $x_m = m\,\Delta x$. Allo stesso modo discretizziamo il tempo scegliendo un Δt come istante di campionamento e n è la durata della nostra simulazione ottenendo $t_n = n\Delta t$.

L'equazione si riscrive in tal modo:

$$y(x_m, t_n) = y_+\left(n\Delta t - \frac{m\,\Delta x}{v}\right) + y_-\left(n\Delta t + \frac{m\,\Delta x}{v}\right) \tag{1.9}$$

A partire dalla condizione di stabilità del sistema di Von Neumann che impone la seguente condizione

$$v\frac{\Delta t}{\Delta x} \leq 1 \tag{1.10}$$

Possiamo vincolare l'intervallo di campionamento spaziale rispetto all'intervallo di campionamento temporale con $\Delta x = v\,\Delta t$ risulta:

$$y(x_m, t_n) = y_+(n\Delta t - m\,\Delta t) + y_-(n\Delta t + m\,\Delta t) \tag{1.11}$$

Più in generale otteniamo:

$$y(x_m, t_n) = y_+(n\Delta t - m\,\Delta t) + y_-(n\Delta t + m\,\Delta t) \tag{1.12}$$

1.2 Linea di Ritardo

Una linea di ritardo digitale è un elemento discreto nella teoria dei filtri digitali, che permette di introdurre un ritardo tra il suo ingresso e la sua uscita. Se il ritardo è un multiplo intero di campioni, le linee di ritardo digitali sono spesso implementate come buffer circolari. Ciò significa che i ritardi interi possono essere calcolati in modo molto efficiente. Se un ritardo non è un numero intero di un campione, vengono applicati filtri aggiuntivi per tenere conto della frazione di ritardo diversa da un numero intero. Quindi le linee di ritardo con ritardo non intero sono chiamate linee di ritardo frazionarie.

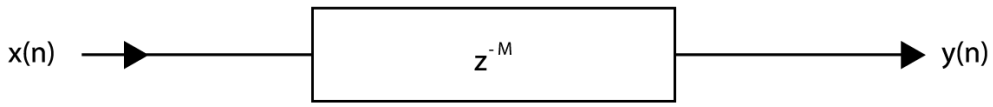


Figura 2 - Linea di ritardo

Consideriamo l'ingresso del segnale x(n) con n=0,1,2,3…. e consideriamo la lunghezza della linea di ritardo di M campioni. Allora l'uscita sarà y(n) = x(n-M).

Prima dell'era digitale, le linee di ritardo erano costose e imprecise in forma "analogica". Ad esempio, i "riverberi a molla ", che sono comuni negli amplificatori per chitarra e utilizzano molle metalliche come linee di ritardo analogiche, pur essendo adeguati a tale scopo, risultano altamente dispersivi e soggetti a captazione del rumore. Grandi ritardi richiedono molle o bobine lunghe nelle implementazioni analogiche. Nel dominio digitale, invece, il ritardo per N campioni è fondamentalmente implementato e ritardi non interi possono essere implementati usando tecniche di interpolazione come interpolazione linear o interpolazione All-pass.

1.3 Guida d'onda digitale

Una guida d'onda digitale è una struttura che guida le onde, come le onde elettromagnetiche o il suono, limitando in una direzione la trasmissione di energia con una minima perdita di quest'ultima. Senza il vincolo fisico di una guida d'onda, le intensità delle onde diminuiscono, secondo la legge del quadrato inverso, man mano che si espandono nello spazio tridimensionale. Questo perché l'intensità di un'onda sferica, un'onda che distribuisce la sua potenza su sfere centrate sulla sorgente, è direttamente proporzionale all'inverso del quadrato dell'ampiezza.

Considerando la formula (1.12) e discretizzandola rispetto al tempo si ottiene la guida d'onda digitale con il seguente diagramma con le linee di ritardo:

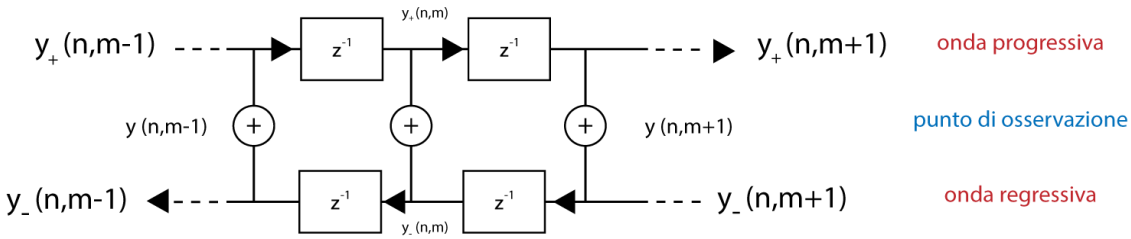


Figura 3 – Guida d'onda digitale

1.4 Algoritmo di Karplus Strong

Il modello di Karplus-Strong prende il nome dai suoi inventori Kevin Karplus e Alex Strong che, all'inizio degli anni '80, presentarono all'International Computer Music Conference di Venezia uno dei primi metodi di sintesi di modellazione fisica in grado di simulare il suono di una corda pizzicata e di alcuni tipi di percussioni.

Tale modello sfrutta la presenza di una linea di ritardo, i cui stati vengono inizializzati con valori random, a cui segue un filtro passa basso la cui uscita corrisponde all'uscita dello strumento che si sta simulando.

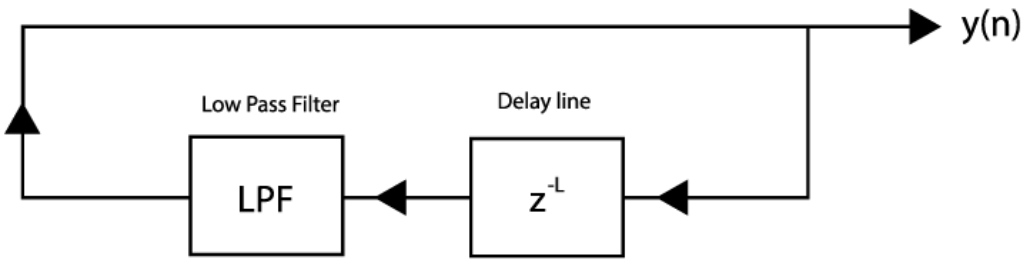


Figura 4 - Schema Karplus – Strong

Nel caso in cui non venga introdotto il filtro passa basso si rischierebbe di avere in uscita un suono periodico che, pari alla lunghezza della linea di ritardo con numerose armoniche, risulterebbe sgradevole all'orecchio. Grazie all'apporto del filtro passa basso si determina il decadimento delle armoniche superiori che si traduce in un suono di una corda pizzicata molto più naturale di quanto non fosse prima.

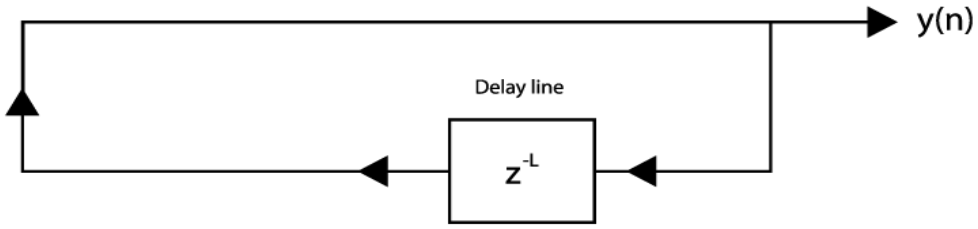


Figura 5 - Schema Karplus - Strong senza filtro passa basso

Dunque, il modello consta di una linea di ritardo e di un filtro passa basso che rappresentano digitalmente la corda e, aggiungendo in input un rumore bianco è possibile ottenere un modello molto affidabile.

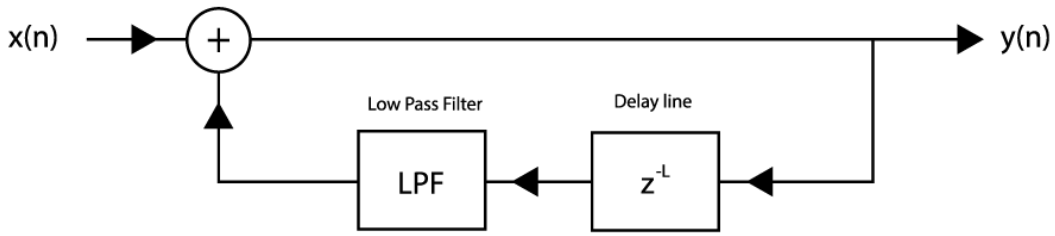


Figura 6 - Schema completo Karplus – Strong

Il filtro digitale è dato dall'equazione differenziale:

$$y(n) = x(n) + \frac{y(n-N) + y(\mathrm{n} - |N+1|)}{2} \tag{1.13}$$

dove x(n) rappresenta l'input al campione n, mentre y(n) l'uscita e N il pitch desiderato del segnale. Il rumore bianco è definito da:

$$x(n) = \begin{cases} A\ u_n & n = 0,1,2,\dots,N-1 \\ 0, & n \geq N \end{cases} \tag{1.14}$$

dove A è l'ampiezza desiderata mentre $u_n \in$ [-1,1] è l'uscita randomica generata.

Consideriamo, a tal punto, una corda, per esempio di una chitarra, che viene pizzicata:

1. Consideriamo una corda a riposo.
2. La corda viene pizzicata e inizia, quindi, a vibrare generando un'onda sonora, la cui frequenza dipende dalla tensione e dalla massa della corda.
3. L'attrito tra l'aria e la corda tende a esaurire l'energia della corda, la cui vibrazione diminuisce lentamente e, dunque, l'onda risulta avere meno armoniche di prima.
4. La corda infine smette di vibrare.

L’introduzione della linea di ritardo e del filtro passa basso permette di ottenere in digitale, rispettando il modello di Karplus- Strong, il medesimo risultato di una corda pizzicata.

1.5 Tipologia di Filtri

Un filtro è un sistema in grado di lavorare su un insieme di dati per estrarre informazioni più importanti di altre, tutto ciò che non occorre. In molti casi i filtri sono usati per la rimozione di anomalie di dati del sensore o per lavorare un segnale per ottenerne una sua versione pulita, senza disturbi di alcun tipo.

Nell’ambito sonoro, ad esempio, esistono i filtri passa basso (che consentono di far passare o trasmettere esclusivamente le basse frequenze, rimuovendo le informazioni irrilevanti.

Tra le varie tipologie di filtri esistono due categorie di nostro interesse, le caratteristiche principali sono desumibili già dai loro nomi, seppur non siano le uniche a identificarli:

- filtri con risposta all’impulso infinita (IIR);
- filtri con risposta all’impulso finita (FIR);

Considerato un segnale, le sue componenti in frequenza vengono ritardate ogni qualvolta entra in un dispositivo come un amplificatore, un filtro o semplicemente propagarsi nell’aria. Questo ritardo può essere diverso per ogni componente in frequenza, generando distorsioni di fase e problemi di scarsa qualità e fedeltà rispetto al segnale originale, soprattutto in caso di ritardi molto ampi. Tutto ciò genera delle grandi complicazioni, ma nel caso in cui si verifichi una condizione particolare per cui si manifesta una coincidenza fra il ritardo di gruppo e il ritardo di fase (ritardo temporale delle varie componenti di frequenza) e, quindi, si parla di fase lineare, allora soltanto in quel caso succede che ogni componente è ritardata dello stesso fattore, eliminando così difficoltà nella gestione del segnale e qualsiasi tipo di distorsione.

1.5.1 Filtri IIR

I filtri IIR richiedono poche operazioni e quindi un basso quantitativo di memoria per cui possono essere ospitati in sistemi con basse prestazioni computazionali e di basso costo; di conseguenza, si ha un costo esiguo in termini di implementazioni e una risposta immediate con una latenza prossima allo zero. Tutti questi pregi però sono accompagnati da due grandi svantaggi: fase non lineare, specialmente vicino alle frequenze di taglio, e stabilità, poiché lavorano con gli input presenti e con l'output passato, che viene reintrodotto per via delle retroazioni presenti nel filtro.

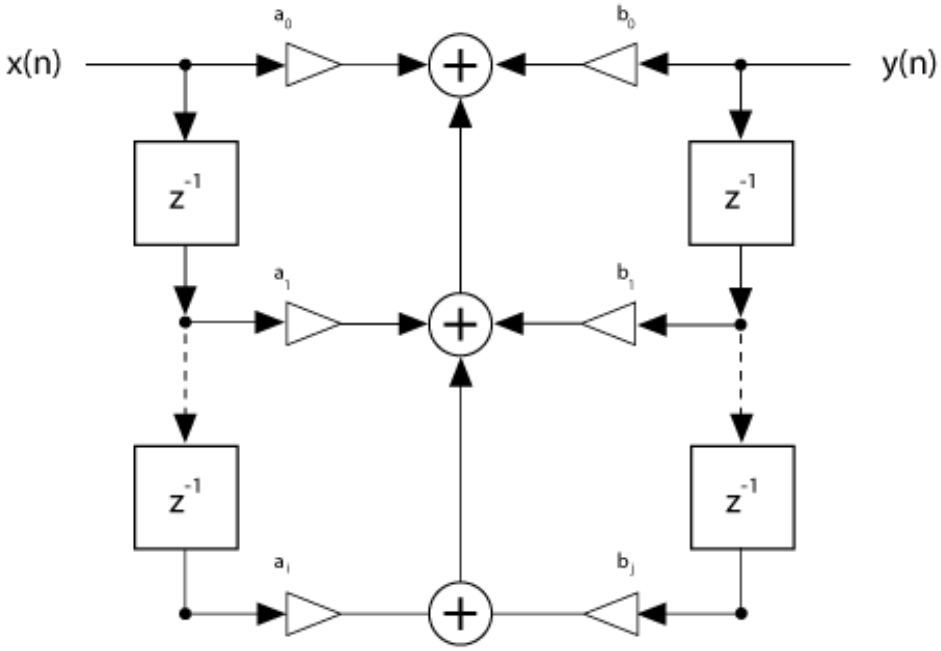


Figura 7 - Filtro IIR

É descritto dalla seguente equazione caratterizzata dai coefficienti del filtro:

$$y(n) = \sum_{k=0}^{n} b_k \, x\,[n-k] \; - \sum_{k=0}^{n} a_k \, y\,[n-k] \tag{1.15}$$

Pertanto, applicando la trasformata Z si ottiene la funzione di trasferimento H(z)

$$H(z) = \frac{\sum_{i=1}^{N} b_j \, z^{-i}}{1 + \sum_{j=1}^{N} a_j \, z^{-J}} \tag{1.16}$$

Dove N è l'ordine del filtro e *ai* e *bj* i coefficienti e si possono individuare gli zeri (radici del numeratore) e i poli (radici del denominatore) del filtro.

1.5.2 Filtri FIR

I filtri FIR sono caratterizzati da fase lineare e impiegati, quindi, in ambiti dove è necessario evitare in qualsiasi modo distorsioni sull'uscita per ottenere dei segnali il più possibile fedeli alla realtà, grazie al fatto che ogni componente in frequenza viene ritardata della medesima quantità; sono stabili poiché non presentano retroazione di nessun tipo al contrario dei filtri IIR. I filtri FIR però richiedono un numero di coefficienti molto più elevato rispetto ai precedenti e questo determina, se sono simmetrici, un'elevata quantità di memoria con un aumento del ritardo di gruppo costante pari a $(N-1)/2$. Sono, pertanto, meno prestanti rispetto ai filtra FIR ma più adatti per gestioni di segnale che non siano in tempo reale.

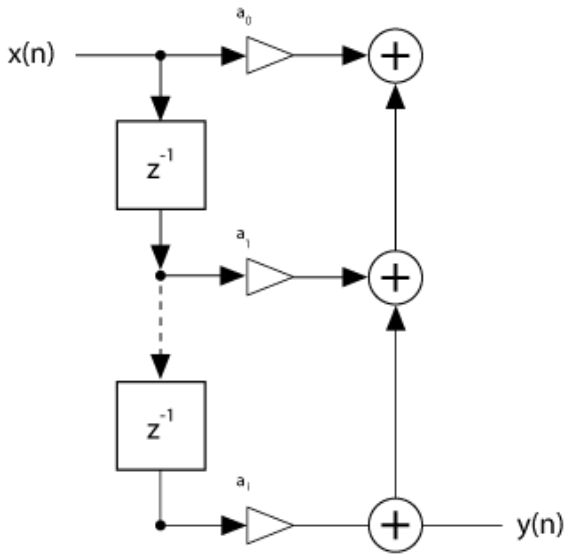


Figura 8 - Filtro FIR

Allora è definibile, a partire dall'uscita,

$$y(n) = \sum_{i=0}^{N} h_k \, x\,[n-k] \tag{1.17}$$

la sua funzione di trasferimento:

$$H(z) = \frac{N(z)}{z^M} \tag{1.18}$$

Con H(z) polinomio in z^{-1} che implica abbia un solo polo centrato in 0 nel cerchio di raggio unitario, dove N(z) è un polinomio di grado M a coefficienti reali. A meno di una costante moltiplicativa, il filtro viene allora univocamente determinato dagli zeri di P(z), ossia dalle soluzioni dell'equazione P(z) = 0.

1.5.3 Stabilità

La stabilità di un sistema in generale è legata alla sua risposta impulsiva. È possibile quindi sostenere che un sistema risulta:

- stabile, se la risposta all'impulso è limitata;
- asintoticamente stabile, se è stabile e la risposta del sistema converge asintoticamente al valore zero;
- instabile, altrimenti.

Il carattere di convergenza della risposta impulsive dipende esclusivamente dalla posizione dei poli della funzione di trasferimento:

$$H(z) = \frac{N(z)}{D(z)} = \frac{b_0 z^n + b_1 z^{n-1} + \cdots b_n}{z^n + a_1 z^{n-1} + \cdots + a_n} \tag{1.19}$$

Il sistema risulta essere:

- asintoticamente stabile, se tutte le radici del polinomio del numeratore N(z), definite come poli del sistema, risultano essere all'interno del cerchio di raggio unitario che ha centro nell'origine nel piano z ovvero se $|p_i| > 1 \; \forall \; i$;
- stabile, se tutti i poli con modulo unitario, ovvero $|p_i| = 1$, hanno molteplicità unitario (si definiscono "poli semplici") e tutti gli altri poli sono ancora contenuti all'interno del cerchio di raggio unitario;
- instabile, se almeno un polo si trova al di fuori del cerchio di raggio unitario centrato nell'origine del piano z.

Come si nota dalle definizioni appena espresse, gli zeri (radici del denominatore) non influiscono mai sulla stabilità del sistema. Dunque, per tornare a quanto affermato nel caso del filtro IIR la **stabilità BIBO,** ovvero dato un ingresso limitato ci aspettiamo in uscita una uscita ancora limitata, e questo dipende solo ed esclusivamente dai poli che dovranno rimanere confinati all'interno del cerchio di raggio unitario.

Quindi è tendenzialmente più facile che il filtro IIR sia instabile, al contrario del filtro FIR, che non avendo praticamente nessun polo (o meglio poli centrati in zero), sarà sempre stabile.

1.6 Ritardo Frazionario

I filtri di ritardo frazionario sono in grado ritardare il segnale di input di una frazione di tempo del periodo di campionamento. Si rivelano, pertanto, utili in numerose applicazioni in cui è fondamentale generare dei ritardi di tempo precisi o modificare le posizioni degli istanti di campionamento, come nel caso delle telecomunicazioni, nella sintesi musicale e nella codifica vocale.

La risposta all'impulso di un filtro di ritardo frazionario ideale è la funzione *sinc*, traslata in avanti di un ritardo D, caratterizzato da una parte intera *floor (D)* e da una frazionaria ottenuta tramite l'operazione *d=D-floor(D),* con n numero del campione:

$$h(n) = sinc(n - D) \tag{1.20}$$

Mettendo a confronto la risposta all'impulso del filtro di ritardo frazionario con *D=3* e *D=3,3* si verifica che, come nel secondo caso, il valore medio della risposta all'impulso ha lunghezza infinita (linea tratteggiata).

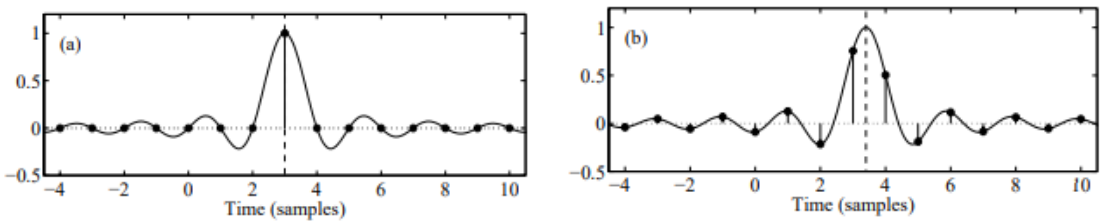


Figura 9 - **(a)** Risposta all'impulso D=3. **(b)** Risposta all'impulso D=3,3

I cerchi neri indicano i valori campionati dalla funzione *sinc*, che nel caso in cui il ritardo D=3 (valore intero), tranne il più centrale, sono zero, mentre nel secondo caso, in cui il ritardo è frazionario, i valori campionati sono diversi da zero. Per questo motivo la risposta all'impulso corrisponde ad un filtro non causale che può essere reso causale da uno spostamento finito nel tempo. Inoltre non ha stabilità BIBO perché la risposta non è sommabile. Dunque è un filtro di ritardo frazionario ideale irrealizzabile nella pratica.

Esistono, come indicato in [1], delle tecniche per crearne di reali tramite filtri FIR:

1. Funzione sinc finestrata (usando una finestra asimmetrica)
2. Approssimazione Ritardo Frazionario con un passabasso con una banda di transizione graduale ottenuto utilizzando una funzione spline di basso ordine;
3. Approssimazione FIR massimamente piatta (interpolazione di Lagrange);
4. Approssimazione dei minimi quadrati pesati (WLS);
5. Metodo di Oetken (approssimazione FD quasi equiripple).

Oppure tramite filtri IIR:

1. Approssimazione di fase ai minimi quadrati (LS);
2. Approssimazione del ritardo di fase LS;
3. Ritardo di gruppo massimamente piatto (Filtro All-pass di Thiran);
4. Progettazione iterativa dell'errore di fase WLS
5. Progettazione iterativa dell'errore di ritardo di fase WLS

Tra tutti il più semplice da realizzare a livello di progettazione che ha dimostrato avere un’efficacia maggiore è il filtro All-pass di Thiran, poiché sono filtri la cui risposta in ampiezza è esattamente piatta e di conseguenza in fase di progettazione è possibile concentrarsi sulle proprietà di fase. Inoltre risulta essere molto accurato sulle basse frequenze.

1.7 All-pass

Un filtro All-pass è un filtro che lascia passare tutte le componenti in frequenze del segnale di ingresso x(t) senza attenuarle o amplificarle in ampiezza mentre associa degli sfasamenti prestabiliti alle diverse frequenze del segnale in ingresso.

Una delle soluzioni più note è quella proposta da Thiran nel 1971 un metodo analitico per i coefficienti di un filtro passa basso a soli poli (categoria rappresentata dalla funzione di trasferimento dei filtri passa basso che presenta tutti gli zeri centrati nell'origine) con risposta di ritardo di gruppo massimamente piatta alla frequenza zero.

Il metodo per definire i coefficienti rispecchia la seguente formula:

$$a_k = (-1)^k \binom{N}{k} \prod_{n=0}^{N} \frac{2d+n}{2d+k+n} \quad con\ k = 0,1,2,\dots,N \tag{1.21}$$

con il coefficiente binomiale definibile come

$$\binom{N}{k} = \frac{N!}{k!\,(N-k)!} \tag{1.22}$$

Fettweis ha mostrato che le formule di progetto possono essere utilizzate per ottenere filtri All-pass, che hanno la stessa proprietà. Quando il ritardo di gruppo desiderato di un filtro All-pass è d è solo necessario effettuare la sostituzione d' = d/2 nella formula di Thiran, poiché il ritardo di gruppo di un filtro All-pass è il doppio del suo denominatore.

La formula di progettazione Thiran per un ritardo frazionario può essere scritta come:

$$a_k = (-1)^k \binom{N}{k} \prod_{n=0}^{N} \frac{D-N+n}{D-N+k+n} \quad con\ k = 0,1,2,\dots,N \tag{1.23}$$

Da cui si ottengono i coefficienti esposti nella seguente tabella a cui seguono i grafici del ritardo di fase:

	a_1	a_2	a_3
$N=1$	$\frac{-D-1}{D+1}$		
$N=2$	$-2\,\frac{D-2}{D+1}$	$\frac{(D-1)(D-2)}{(D+1)(D+2)}$	
$N=3$	$-3\,\frac{-D-1}{D+1}$	$3\frac{(D-2)(D-3)}{(D+1)(D+2)}$	$-\frac{(D-1)(D-2)(D-3)}{(D+1)(D+2)(D+3)}$

Tabella 1 - Coefficienti filtro All-pass

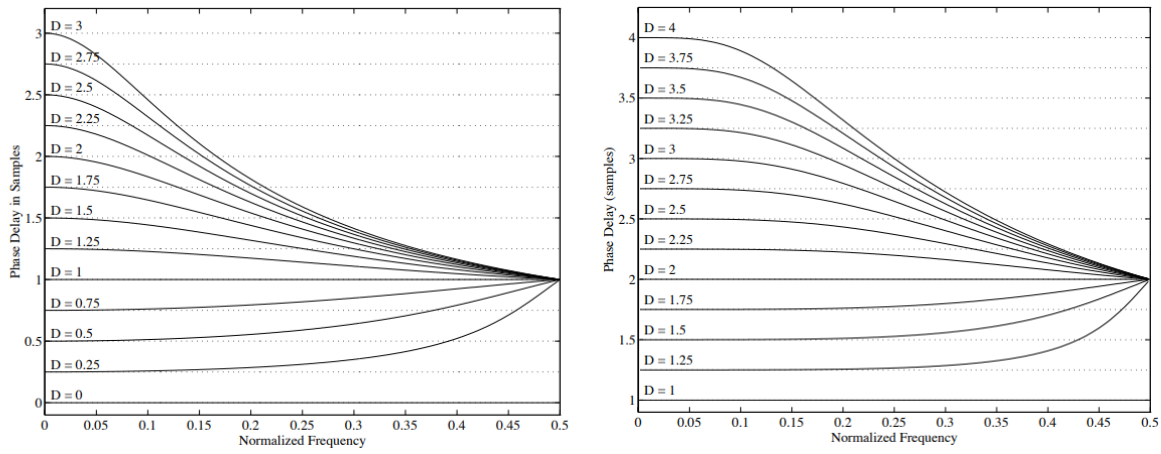


Figura 10 - **(a)** Ritardo di Fase con filtro di ordine N =1. **(b)** Ritardo di fase con filtro di ordine N=2

1.8 Variazione istantanea dei coefficienti in filtri IIR

Partendo da quanto sostenuto nel paragrafo 1.2 è chiaro che risulta essenziale poter cambiare durante il funzionamento di un sistema il valore di ritardo per ottenere il ritardo desiderato, che non dovrà essere costante, altrimenti decade la centralità dell'argomento stesso

Nel metodo descritto da Valimaki ciò che viene fatto è studiare l'implementazione di una linea di ritardo a lunghezza variabile introducendo una variazione dei coefficienti di un IIR senza artefatti.

Per variare la lunghezza della linea di ritardo dobbiamo modificare dopo un certo numero di iterazioni o porre a condizioni i coefficienti del filtro. Questa operazione non è priva di conseguenze: difatti, ciò che si ottiene è che il segnale di uscita al cambio dei coefficienti soffre di un fenomeno transitorio in quanto non solo sono stati modificati i parametri del filtro, ma oltretutto la variabile stato del filtro stesso presenta dei valori che fanno riferimento al filtraggio precedente mentre opera con un nuovo tipo di filtraggio introducendo una discontinuità. Tutto questo porta la variabile di stato a non essere mai "sincronizzata" con il processo di filtraggio che si sta effettuando. Questo effetto non è presente nel caso di filtri non ricorsivi perché in quel caso il filtro ha tutto il tempo di assestarsi con il nuovo cambio dei coefficienti. Inoltre, nell'implementazione dei filtri ricorsivi si rischia che un set di coefficienti e di informazioni della variabile di stato vengano alterati o cancellati, generando così un segnale di uscita totalmente diverso da ciò che ci si aspettava. Va quindi considerato che il fenomeno transitorio e la discontinuità sono eventi che possono presentarsi e la cui unica soluzione è intrinseca al cambiamento dei coefficienti: meno il cambiamento è brusco, più tempo di transizione il filtro ha per assestarsi, modificando i valori di stato gradualmente ed eliminando, in tal modo, le disomogeneità appena descritte.

1.9 Creazione Linea di Ritardo a lunghezza variabile tramite filtri FIR

Per creare una linea di ritardo a lunghezza variabile è utile impiegare dei filtri FIR questo perché, come affermato in precedenza nel paragrafo 1.5.1, sono stabili in tutte le situazioni per via della mancanza della retroazione, che permette di non avere problemi di nessun tipo sui transitori quando i coefficienti del filtro sono modificati. Si definisce, quindi, la lunghezza del vettore dei coefficienti come h che viene indicato con *h(n,D)*. La lunghezza della linea di ritardo può essere cambiata calcolando nuovi coefficienti ad ogni intervallo di campionamento. L'interpolazione risulta essere necessaria quando la lunghezza della linea di ritardo non è caratterizzata da un numero intero ma quando il valore di ritardo *D* è dato da una parte intera $|D|$ e da una frazionaria *d*.

$$D = |D| + d \tag{2.1}$$

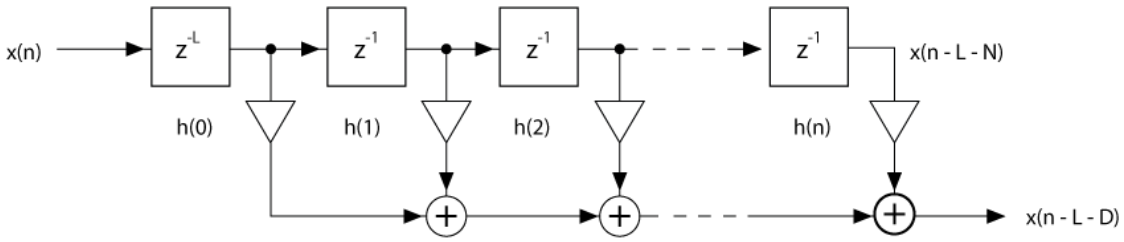


Figura 11 - Lunghezza variabile della Linea di Ritardo vista come un FIR

Una stima del valore ideale del segnale interpolato è data da:

$$x(n - L - D) = \sum_{k=0}^{N} h(k)x(n - L - k) \tag{2.2}$$

dove M è il numero di ritardi unitari prima dell'interpolatore FIR e il vettore dei campioni di segnale da utilizzare è:

$$x_n = [\, x(n - L) \;\; x(n - L - 1) \;\; x(n - L - 2) \ldots x(n - L - N)]^T \tag{2.3}$$

Per evitare di generare discontinuità è fondamentale memorizzare i valori del segnale in ingresso nella memoria del buffer e, a quel punto, cambiare il ritardo, aggiornando i coefficienti. Per evitare latenza nell'elaborazione è importante definire un ritardo massimo, allungando così la delay line che avrà una lunghezza L definita da:

$$L_{tot} = D_{max} + \frac{N}{2} \tag{2.4}$$

Inoltre, i coefficienti devono rimanere in un range accettabile senza allontanarsi troppo da quelli precedenti, motivo per cui il valore di ritardo successivo D_+ deve soddisfare il seguente requisito

$$\frac{N-1}{2} < D_+ < \frac{N+1}{2} \tag{2.5}$$

1.10 Eliminazione del fenomeno transitorio

Per quanto detto fino ad ora il fenomeno dei transitori si presta a essere gestito con efficacia e immediatezza. Questo problema però diventa più complesso nel caso di filtri ricorsivi (IIR) poiché il cambiamento dei coefficienti genera disturbi sui valori futuri delle variabili di stato interne e, di conseguenza, sul segnale di uscita.
Esistono infatti diversi metodi, come indicato in [1], per far sì che questo fenomeno sia meno invasivo:

1. Metodo a dissolvenza incrociata
2. Variazione graduale dei coefficienti usando interpolazione (Mourjopoulos et al., 1990)
3. Matrice di coefficienti intermedi (Rabenstein, 1988)
4. Metodo di commutazione degli ingressi (Verhelst e Nilens, 1986)
5. Aggiornamento del vettore di stato (Zetterberg e Zhang, 1988).

1.10.1 Metodo a dissolvenza incrociata

Il metodo a dissolvenza incrociata consiste nella scelta di due filtri $H_1(z)$ e $H_2(z)$ del filtro variabile nel tempo. Il segnale di ingresso passa per il filtro $H_1(z)$ e viene quindi filtrato. Quando è necessario aggiornare i coefficienti, lo si fa sul filtro $H_2(z)$. Dopo un certo tempo di transizione, affinché il filtro si possa assestare, le uscite dei filtri sono poste in dissolvenza incrociata utilizzando, per esempio, l'interpolazione lineare in maniera tale che solo il filtro $H_2(z)$ filtri il segnale e generi il segnale di uscita. Questo metodo viene poi reiterato nuovamente passando dal filtro $H_2(z)$ al filtro $H_1(z)$.

1.10.2 Variazione graduale dei coefficienti

Mourjopoulos definì un parametro η di variazione del filtro che descrive il numero di coefficienti intermedi per unità di tempo, basandosi su un modello computazionale in grado di replicare le proprietà uditive dell'uomo.

L'effetto di questo metodo è che quando h ha un basso numero di coefficienti la transizione è percepibile all'orecchio umano perché ci sono discontinuità nei parametri del filtro. Aumentando i valori di h da 18 a 35 stati al secondo risulta più fluida la transizione.

All'atto pratico entrambi i filtri $H_1(z)$ e $H_2(z)$ ricevono il segnale e lavorano in parallelo ma, al contrario del metodo precedente, solo uno di questi è collegato all'uscita e dunque il secondo filtro funge da base di appoggio per i nuovi coefficienti che, una volta calcolati, vengono introdotti nel filtro $H_1(z)$. Con questo approccio il filtro ha un tempo più lungo per assestarsi.

1.10.3 Matrice di coefficienti intermedi

In questo metodo si impiega una matrice di coefficienti intermedi. Si prendano in considerazione tre matrici: una iniziale, una intermedia e una di destinazione. Quella intermedia viene usata per compensare e ammorbidire il transitorio tra quella iniziale e quella di destinazione. Lo svantaggio risiede nel fatto che il filtro così implementato è più costoso in termini di operazioni e di memoria.

1.10.4 Metodo di commutazione degli ingressi

Il metodo di commutazione degli ingressi prevede di usare un banco di filtra copia in cascata con coefficienti diversi. L'ingresso viene passato nel filtro $H_1(z)$ e quando è necessario aggiornare i coefficienti l'ingresso è passato nel filtro $H_2(z)$ cancellando il vettore di stato precedente del filtro $H_1(z)$. Le uscite dei filtri vengono sommate a mano a mano affinché il transitorio sia il più piccolo possibile. Verhels e Nilens dimostrano che la risposta dei filtri si attenua dopo che il loro ingresso viene disconnesso.

1.10.5 Aggiornamento del vettore di stato

Il seguente metodo è legato al cambiamento dei parametri di stato in maniera più accurata. Il lato negativo è determinato dal fatto che bisogna conoscere tutti i campioni in entrata a priori, ciononostante fornisce un buon metodo di partenza per algoritmi più efficienti di quelli descritti precedentemente. Si considerino quindi delle copie di $H_1(z)$ che chiameremo $H_2(z), H_3(z), H_4(z)\ldots$ in esecuzioni in parallelo, uno per ogni coefficiente. Tutti i filtri ricevono il segnale in input ma solo $H_1(z)$ è collegato all'uscita. Ogni qualvolta è necessario modificare il coefficiente si fa riferimento al filtro appropriato importando anche la variabile di stato. In questo modo non ci saranno transitori in quanto tutti i filtri sono in stato stazionario per tutto il tempo. Chiaramente introdurre un filtro per ogni coefficiente richiede un impiego di memoria altissimo.

1.11 Metodo di Valimaki

Questo nuovo metodo, introdotto da Valimaki nel 1995 fa fronte in particolar modo alle casistiche in cui è necessario aggiornare i coefficienti del filtro in maniera brusca, introducendo una soppressione della fase transitoria senza l'impiego di grandi risorse applicabile sia a filtri ricorsivi che alla struttura Direct Form I, in cui l'equazione alle differenze viene valutata in maniera diretta, e Direct Form II, che è una forma alternativa della precedente che necessità di N unità di ritardo.

Si tenga in considerazione la struttura Direct Form II la quale risulta essere influenzata dai valori passati di ingresso in proporzione alla lunghezza dell'impulso h(n), che può essere determinata in base alla sua costante di tempo.

Abbiamo dunque un segnale di ingresso x(n) che alimenta due filtri: il filtro IIR $H_1(z)$ denominato filtro del segnale e un filtro $H_2(z)$ in grado di eliminare i transitori determinato solo dal denominatore della funzione di trasferimento $\frac{1}{D(z)}$. Poco prima che il vettore stato del filtro $H_1(z)$ debba essere modificati vengono immediatamente aggiornati le variabili di stato di $H_2(z)$. Il vettore di stato del filtro $H_1(z)$ viene poi aggiornati a partire da quello del filtro $H_2(z)$ dopo un certo numero di campioni e N_a chiamato *tempo di avanzamento:* più è grande più il fenomeno transitorio viene soppresso. Il filtro $H_2(z)$ non è mai collegato all'uscita in quanto serve come base di appoggio per l'aggiornamento dei valori del filtro $H_1(z)$. Dunque, è necessario conoscere N_a campioni prima del momento effettivo di cambiamento i valori del vettore di stato.

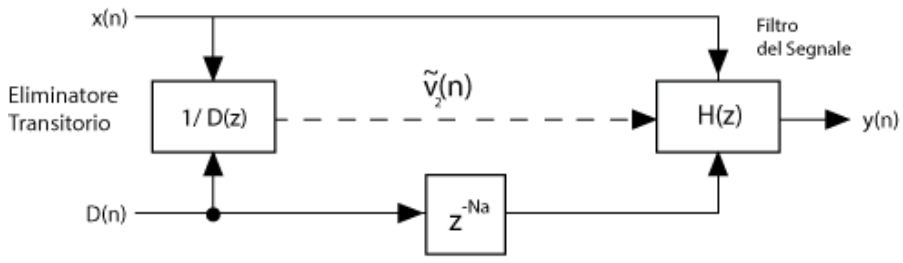


Figura 12 - Implementazione del nuovo Metodo di Valimaki

2 Metodi di Pitch Bending

I metodi discussi nei paragrafi precedenti permettono di effettuare il pitch-bending con modelli della corda come quello di Karplus-Strong. Nel seguente capitolo verrà esposta la fase di sperimentazione in ambiente Matlab.

2.1 Implementazione algoritmo di Karplus-Strong

A partire da quanto esposto nel paragrafo 1.4 segue l'implementazione dell'algoritmo di Karplus-Strong in ambiente Matlab.
Una volta definita frequenza di campionamento e ritardo complessivo si riempie un vettore di coefficienti per la linea di ritardo vista come un filtro FIR e si definisce un coefficiente di damping o di smorzamento utile per simulare l'effetto della corda che prima di arrestarsi oscillerà fino a raggiungere la sua posizione di equilibrio, altrimenti sarebbe in grado di oscillare all'infinito. A seguito della linea di ritardo nello schema precedentemente introdotto, che viene riportato per completezza qui sotto, vi è il filtro passa basso: si sceglie un filtro *Butterworth* di ordine 1 che risulta essere uno dei filtri più semplici in grado di mantenere il modulo della risposta in frequenza il più piatto possibile senza quindi introdurre ulteriori complessità nell'algoritmo. Per terminare si genera un rumore tramite la funzione *rand* per inizializzare il contenuto della digital wave guide.

In questo modo sono stati definiti tutti i parametri necessari per il funzionamento dell'algoritmo: si impiega un ciclo for per avviare il loop nel quale tramite la funzione filter si inizializza la linea di ritardo che, essendo stata generata come un filtro FIR, presenterà solo i coefficienti $b_1 \dots b_n$, mentre al posto dei coefficienti $a_1 \dots a_n$ con valore unitario; segue il filtro passa basso e infine si ha il prodotto tra l'uscita processata dai due filtri con il coefficiente di damping per ottenere l'uscita dell'intero sistema.

2.2 Fase preparatoria

Avendo implementato l'algoritmo di Karplus-Strong si passa alla fase preparatoria che farà da base per i metodi che verranno introdotti nei paragrafi successivi. viene introdotto il primo dei due metodi per il Pitch Bending che è quello dello Step Unitario.

Viene creato un vettore riga *pitchvector,* la cui lunghezza è pari alla frequenza di campionamento, e le cui colonne vengono riempito della lunghezza iniziale della linea di ritardo.

Selezionando il numero di campioni entro cui ci sarà la variazione della linea di ritardo per generare l'effetto del Pitch Bending, grazie alla funzione *linspace*, che rende possibile la generazione di vettori riga linearmente intervallati.

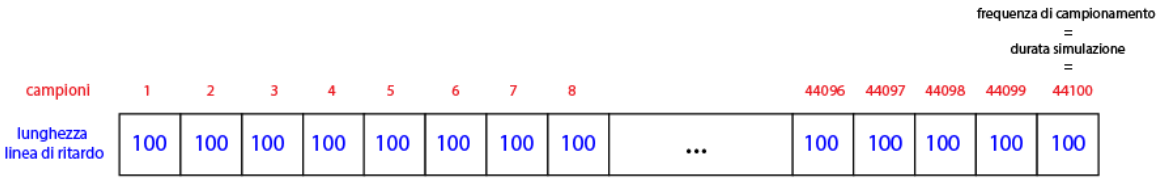


Figura 13 - Vettore riga linearmente intervallato *pitchvector*

Per esemplificare i concetti consideriamo di variare la lunghezza della linea di ritardo come segue:

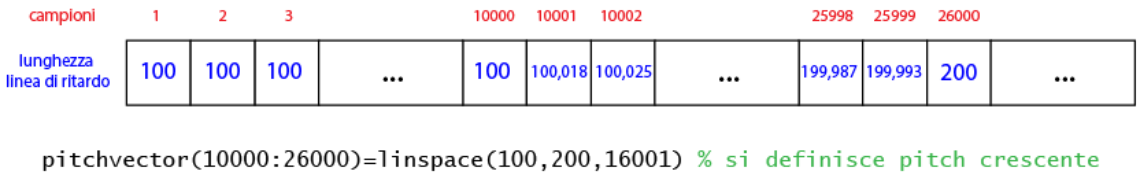
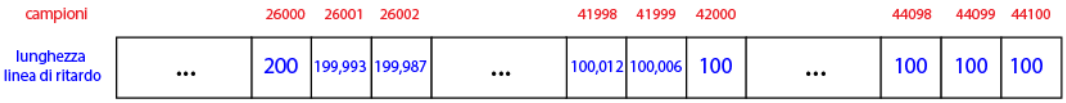


Figura 14 – Vettore riga linearmente intervallato *pitchvector* in cui si definisce pitch crescente

In questo caso si sta facendo variare la lunghezza della linea di ritardo da 100 a 200 dal campione numero 10000 al campione numero 26000 in 16001 campioni. Abbiamo definito il Pitch crescente.

Successivamente a questo possiamo definire il Pitch decrescente facendo tornare la lunghezza della linea di ritardo a 100 dal campione 26000 al campione 42000 in 16001 campioni.




```
pitchvector(26000:42000)=linspace(200,100,16001) % si definisce pitch decrescente
```

Figura 15 - Vettore riga linearmente intervallato *pitchvector* in cui si definisce pitch decrescente

In questo modo abbiamo definito il controllo per variazione della linea di ritardo per tutta la durata della simulazione, che si ricorda essere pari alla frequenza di campionamento.

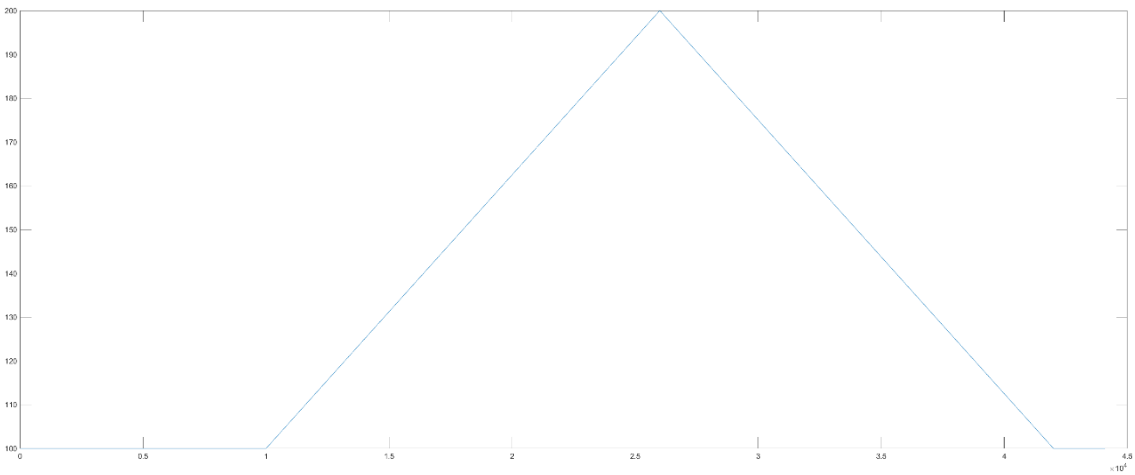


Figura 16 - Grafico del vettore riga linearmente intervallato *pitchvector* per tutta la durata della simulazione

2.3 Variazione della Linea di Ritardo tramite Step Unitari

Viene introdotto il primo dei due metodi per il Pitch Bending che è quello dello Step Unitario.

Consideriamo il vettore riga contenente i coefficienti della linea di ritardo *dwg_b* caratterizzato da valori tutti nulli tranne uno che ha valore unitario che definiremo il "tappo" della linea di ritardo. Spostando il "tappo" ad ogni iterazione possiamo ottenere la variazione della linea di ritardo. Si introduce all'interno del ciclo for un nuovo valore *D(n)* in cui, ad ogni iterazione, viene copiato il valore presente del vettore *pitchvector*. Definiti quindi due nuovi valori *piD* e *iD*, rispettivamente la parte intera del valore precedente al campione n-1, ossia *D(n-1)*, e la parte intera del valore presente al campione n, *D(n)*, si pone nel vettore *dwg_b,* che contiene i coefficienti della linea di ritardo, il valore "1" alla corrispondente colonna *iD* e il valore "0" alla corrispondente colonna *piD*.

$$\begin{cases} dwg\_b(piD) = 0 \\ dwg\_b(iD) = 1 \end{cases} \tag{2.6}$$

Nel caso specifico consideriamo che il valore attuale sul vettore *pitchvector* sia 100, dunque anche *D=100.* Sul vettore dei coefficienti della linea di ritardo *dwg_b* avremo allora che *iD=100* e *piD=99* e quindi il "tappo" sarà nella corrispondente colonna 100.

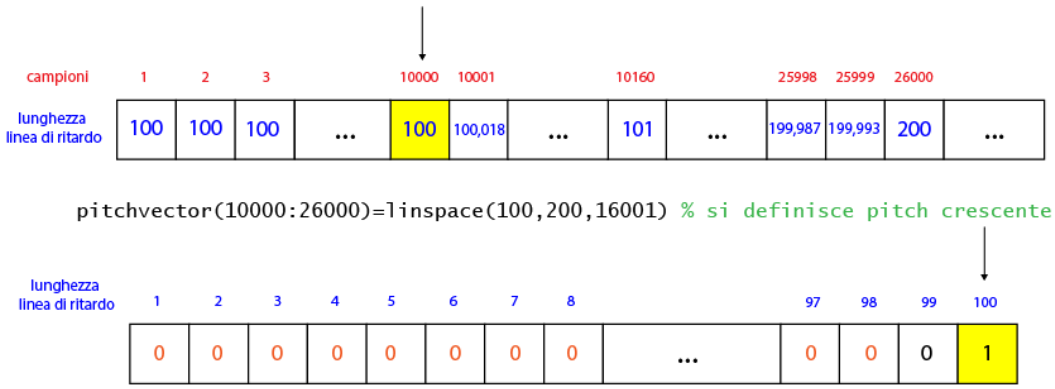


Figura 17 - Inserimento "tappo" sul vettore contenente i coefficienti della linea di ritardo

Dopo alcune iterazioni arrivando al campione 10160, il valore presente di ad è 101 e allora si ha lo spostamento del “tappo” sulla linea di ritardo, che ne genera una variazione in termini di lunghezza.

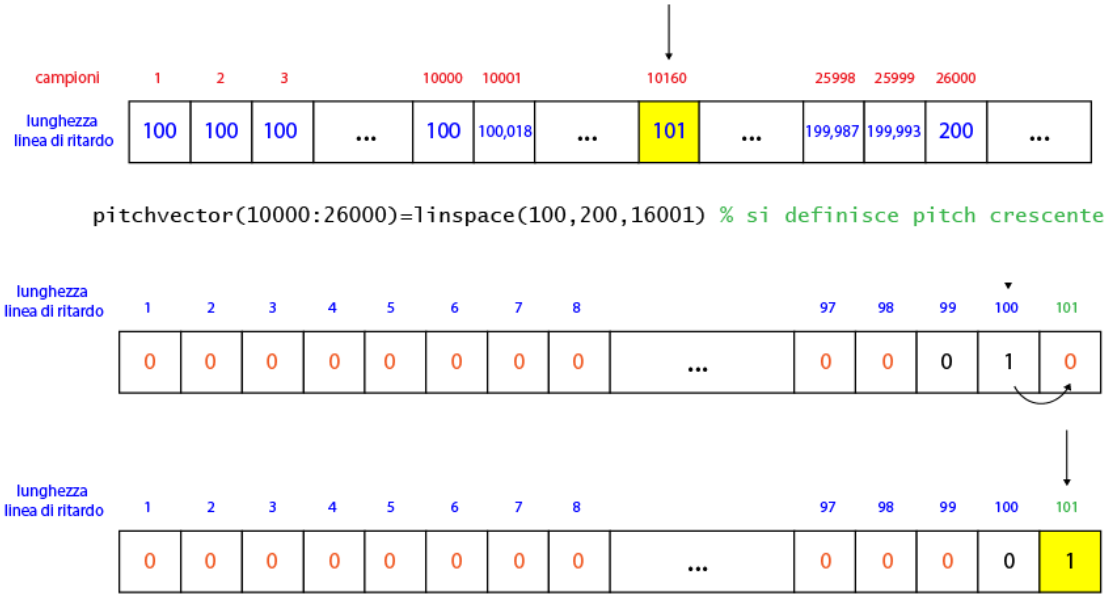


Figura 18 - Spostamento "tappo" sul vettore contenente i coefficienti della linea di ritardo

In questo modo ad ogni iterazione viene spostato il “tappo” permettendo di variare la lunghezza della linea di ritardo, tramite la forma di controlla espressa con il vettore *pitchvector*.

2.4 Variazione della Linea di Ritardo tramite Interpolazione Lineare

Il metodo dello Step Unitario permette effettivamente di modificare la lunghezza della linea di ritardo, ma lo fa senza considerare mai la parte frazionaria, variando la lunghezza della linea di ritardo sempre di un valore unitario per volta.

Il metodo dell'Interpolazione Lineare permette invece di modificare la lunghezza della linea di ritardo in maniera progressiva generando quindi un'uscita più fluida rispetto a quella ottenuta con il metodo precedente.

L'interpolazione lineare permette di definire una linea retta tra due campioni vicini, restituendo il punto appropriato lungo tale linea. Consideriamo il ritardo D(n) ciò che accade dipende dalla scelta di un valore η, compreso fra 0 e 1, che rappresenta quanto si vuole interpolare D tra l'istante n e l'istante n+1. Si può definire il valore interpolato come la media ponderata:

$$\widehat{D}(n+\eta) = (1-\eta) \cdot D(n) + \eta \cdot D(n+1) \qquad (2.7)$$

Per η =0 otteniamo $\widehat{D}(n) = D(n)$, mentre per η =1 otteniamo $\widehat{D}(n+1) = D(n+1)$ e definiamo rispettivamente $1-\eta$ e η i pesi dei valori *D(n)* e *D(n+1)*.

Consideriamo il caso in cui *D(n)*=100,1. Si considerano i valori interi 100 e 101 da cui si definiscono le distanze del valore *D(n)* ottenendo i due pesi $(1-\eta) = 0{,}1$ e $(\eta) = 0{,}9$

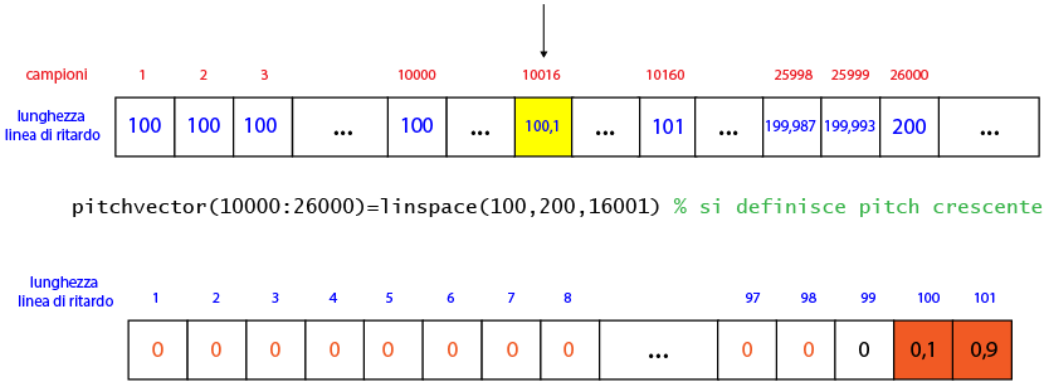


Figura 19 - Definizione dei pesi sul vettore pitchvector contente i coefficienti della linea di ritardo

Facendone la media ponderata otteniamo il valore interpolato:

$$\widehat{D}(n+\eta) = \frac{100 \cdot 0{,}1 + 101 \cdot 0{,}9}{0{,}9 + 0{,}1} = 100{,}9 \tag{2.8}$$

Più in generale allora definiti quindi due i valori *piD* e *iD,* se ne aggiunge uno nuovo che è *fD* che contiene la parte frazionaria derivante dalla differenza tra il valore *D* corrente e la sua parte intera.

$$fD = D - |D| \tag{2.9}$$

E dunque:

$$\begin{cases} dwg\_b(piD-1) = 0 \\ dwg\_b(piD) = 1 - \mathrm{fD} \\ dwg\_b(iD) = \mathrm{fD} \end{cases} \tag{2.10}$$

2.5 Variazione della Linea di Ritardo tramite Step Unitari con filtri All-pass

Il secondo dei due metodi è sicuramente il più efficiente in termini di dettagli sonori: come vedremo poi nei risultati sperimentali è affetto da meno artefatti digitali. Il metodo di variazione della linea di ritardo tramite Step Unitari genera un suono più meccanico per via delle variazioni a scalino che viene generata. In accoppiato con quest'ultimo metodo sono stati introdotti i filtri All-pass per gestire la parte frazionaria e ottenere un risultato migliore comparabile con quello dell'Interpolazione Lineare.

2.5.1 Filtri All-pass

Considerando l'algoritmo esposto fino al paragrafo 3.3 si aggiungono i filtri All-pass che si occupano di gestire la parte frazionaria.

Si scelgono due filtri $H_1(z)$ e $H_2(z)$. Il segnale di ingresso è collegato a entrambi i filtri, ma solo il filtro $H_1(z)$ genera il segnale di uscita. Il filtro $H_2(z)$ viene usato come punto di appoggio per il calcolo dei coefficienti successivi che dopo un certo numero di iterazioni vengono copiati, insieme al vettore degli stati, nel filtro $H_1(z)$.

Consideriamo un caso pratico studiato all'interno di Matlab:

se fino ad ora abbiamo considerato la lunghezza della linea di ritardo pari a 100 ed questa veniva modificato in base ai valori presenti nel vettore *pitchvector*, più nello specifico se i valori interi di *D* al campione n e il valore al campione n-1 fossero diversi, ora si inizia considerando che di base sia lunga 99 campioni; Si impiegano quindi due filtri All-pass caratterizzati inizialmente da un ritardo *Dall-pass* pari a 1: in questo modo stiamo comunque considerando la lunghezza della linea di ritardo pari a 99+1, con un margine per il calcolo del ritardo frazionario che sarà gestito dai filtri All-pass.

Introduciamo il vettore *fD*, con lunghezza pari alla durata della simulazione e quindi alla frequenza di campionamento, che ad ogni iterazione viene aggiornato con il valore frazionario ottenuto dalla formula 3.4 che viene riportato per comodità:

$$fD = D - |D| \tag{2.11}$$

Finché il valore frazionario rimane compreso in un range tra 0 e 0,99 allora la lunghezza della linea di ritardo è data da 100+fD; non appena il valore frazionario assume valori fuori dal range descritto, il valore di D viene incrementato di un valore unitario. In questo modo si ottiene una variazione della lunghezza della linea di ritardo progressiva, senza particolari interruzioni.

2.5.2 Scambio dei filtri

Il filtro $H_2(z)$ è utile per poter aggiornare i coefficienti e il vettore stato del filtro $H_1(z)$. Questo metodo permette di usare solo due filtri, senza doverne usare diverse copie delle quali era necessario già conoscere a priori i coefficienti per ogni filtro, con una netta diminuzione dell'impiego di memoria e un aumento delle prestazioni. Il metodo con cui avviene lo scambio tra i due filtri inoltre è stato studiato in separata sede per non avere, durante il passaggio dai coefficienti di $H_1(z)$ ai coefficienti di $H_2(z)$, discontinuità e per far sì che il passaggio avvenisse nel momento giusto.

Preso un ingresso sinusoidale e due filtri All-pass del secondo ordine con ritardo rispettivamente di *1,5* e *3,5*, si è scelta una soglia (treeshold) dopo la quale potevamo considerare che entrambi i filtri si fossero assestati: nel filtro $H_1(z)$ la fase transitoria termina dopo il campione n. 11, mentre per il filtro $H_2(z)$ dopo il campione n.7 come riportato in figura.

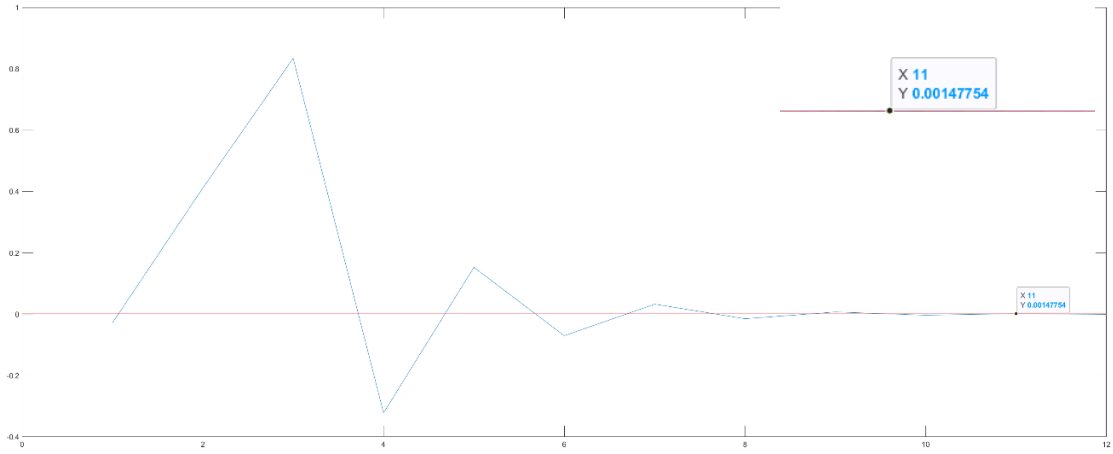


Figura 20 - Fase transitorio del filtro All-pass con D=1,5

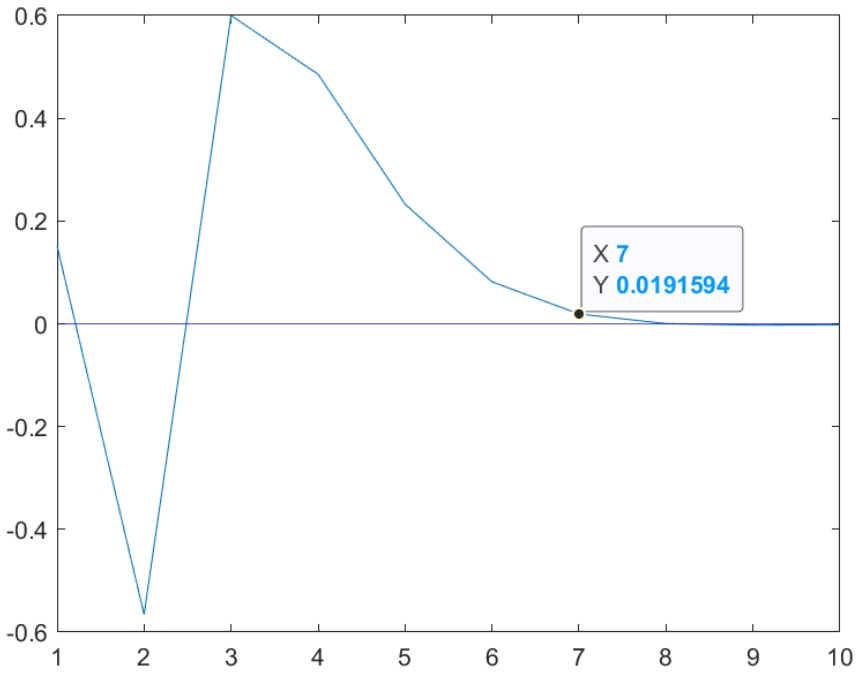


Figura 21 – Zoom della fase transitorio del filtro All-pass con D=3,5

Essendo le fasi transitorio diverse si ha che il passaggio tra i due filtri avviene progressivamente, in modo simile a come si è definito il Pitch: si considera un vettore *up(n),* vettore riga linearmente intervallato, nel quale il viene fatto variare il ritardo D da 1,5 a 3,5 per rendere la transizione dei filtri molto più "morbida".

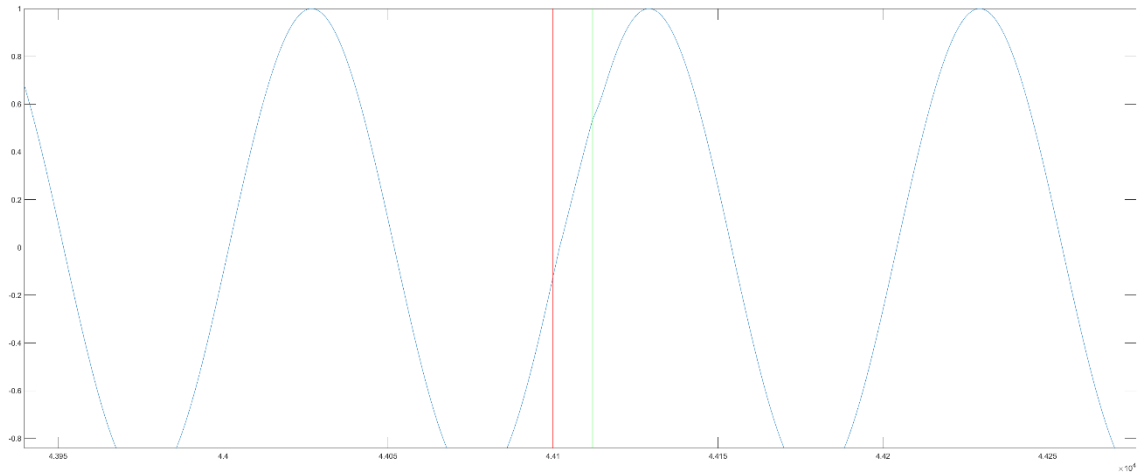


Figura 22 - Scambio dei filtri All-pass. Prima della linea rossa il segnale di uscita è generato solo dal filtraggio di H_1(z), tra la linea rossa e quella verde si ha il passaggio tra i due filtri H1(z) e H2(z), dopo la linea verde si ha solo il filtraggio del filtro H2(z)

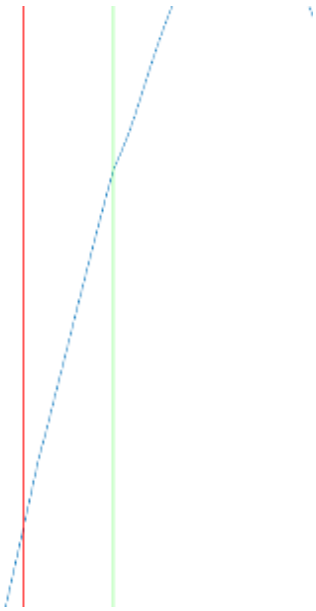


Figura 23 - Zoom della Figura 22

Inoltre, studiando la risposta del filtro si denota come dopo 5 campioni l'effetto del filtro risulta essere trascurabile, dunque lo scambio dei filtri, definito nel metodo dello Step Unitario al quale si associano i filtri All-pass, avviene ogni 5 iterazioni.

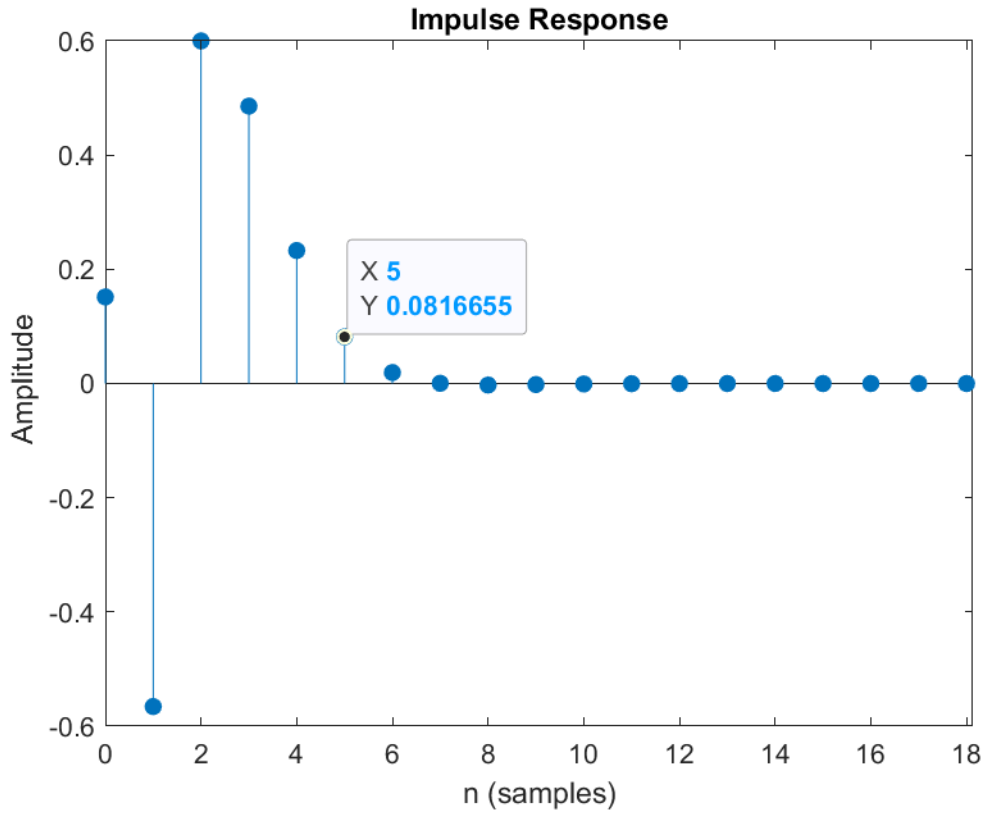


Figura 24 - Risposta all'impulso del filtro All-pass con ordine N=2

3 Risultati Sperimentali

Si riportano i risultati sperimentali ottenuti durante l'implementazione del filtro All-pass e dei metodi precedentemente descritti.

Come si nota dalle Figura 24 e Figura 25 l'implementazione del filtro All-pass è stata effettuata correttamente garantendo tutte le proprietà descritte da Thiran e poi riportate da Valimaki.

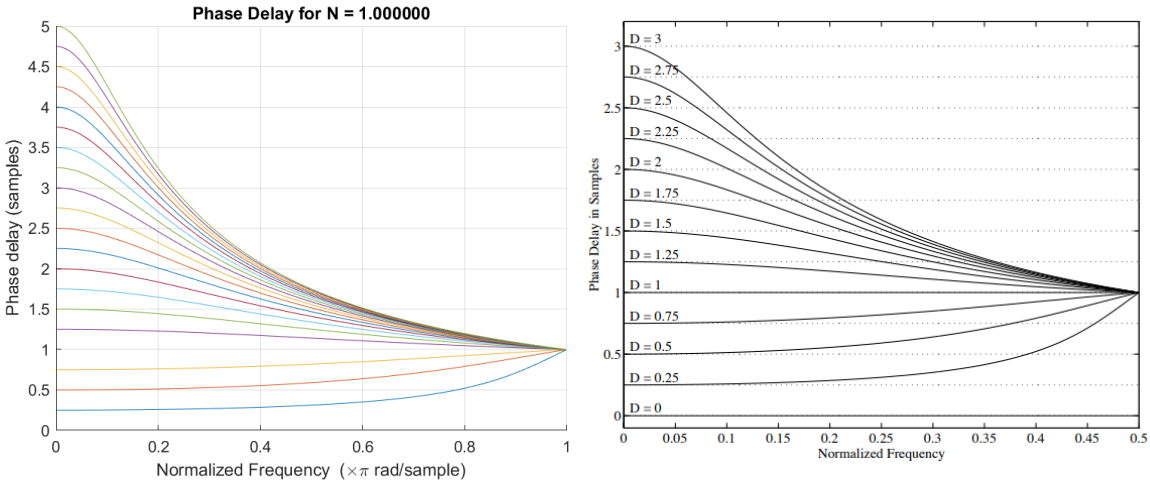


Figura 25 - Filtri All pass con ordine N=1. A sinistra quello implementato in Matlab, a destra quello riportato da Valimaki.

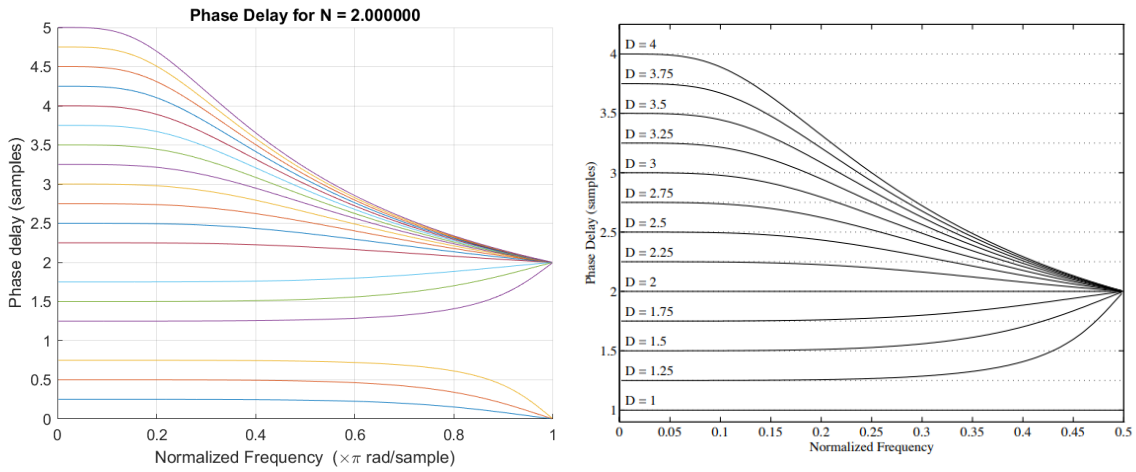


Figura 26 - Filtri All pass con ordine N=2. A sinistra quello implementato in Matlab, a destra quello riportato da Valimaki

Per quanto concerne il metodo dello Step Unitario è possibile denotare dal grafico del segnale di uscita che quest'ultimo è caratterizzato da una mancanza di fluidità e di progressività per via degli artifici digitali precedentemente discussi derivanti dalla variazione della lunghezza della linea di ritardo solo con valori interi, al contrario nel metodo dell'Interpolazione Lineare che risulta essere un metodo più preciso vista la gestione della parte frazionaria, come si nota dagli spettrogrammi.

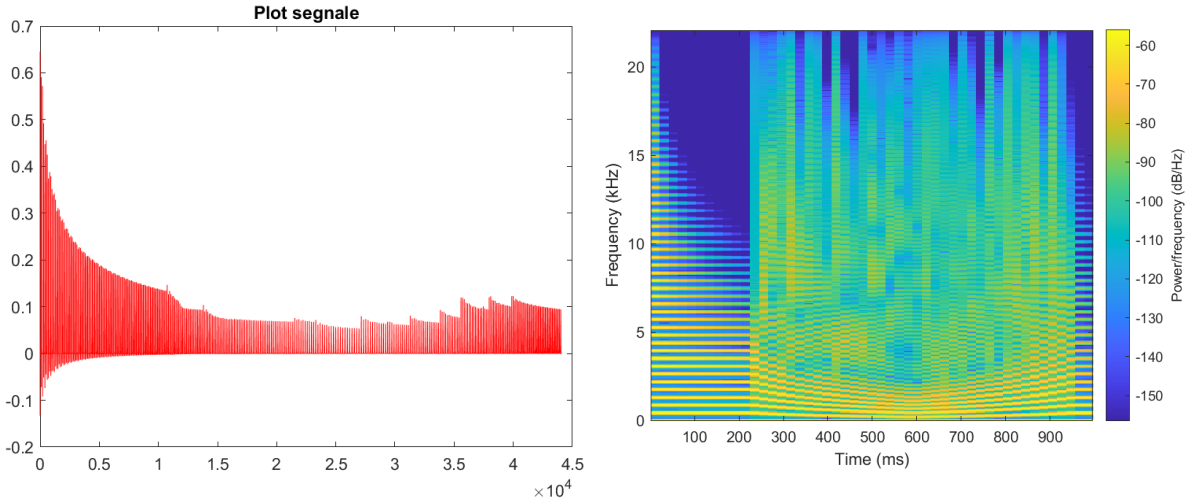


Figura 27 – A sinistra grafico del segnale di uscita usando il metodo dello Step Unitario, a destra il suo spettrogramma

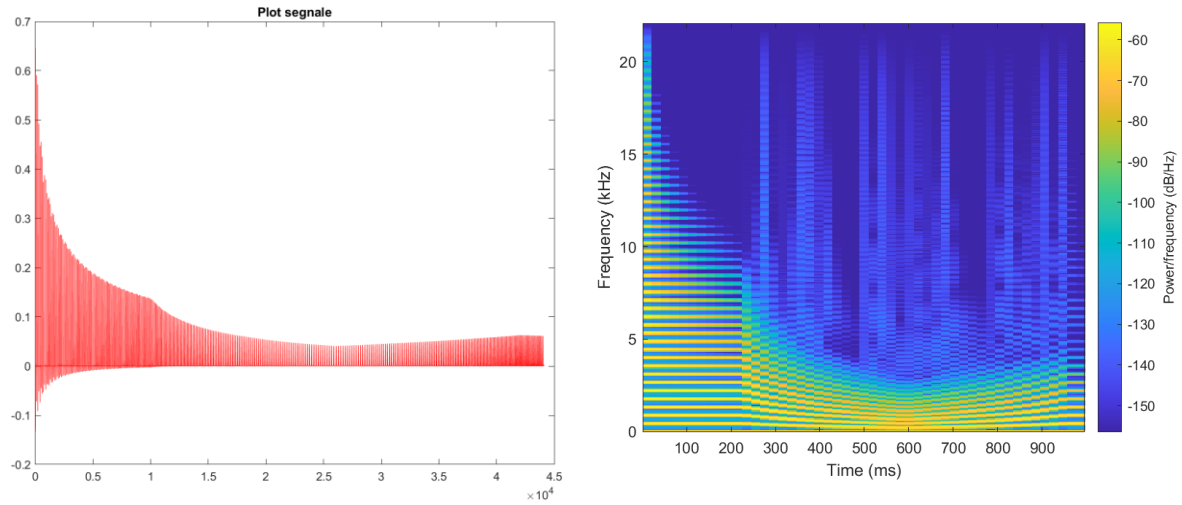


Figura 28 – A sinistra grafico del segnale di uscita usando il metodo dell'Interpolazione Lineare, a destra il suo spettrogramma

Introducendo la gestione del ritardo frazionario sul metodo dello Step Unitario tramite filtri All-pass, di cui vengono riportati i grafici del ritardo di fase che sono identici fra loro, il risultato del segnale di uscita presenta pochissimi artefici digitali aggiungendo dei dettagli che erano assenti nell'Interpolazione Lineare.

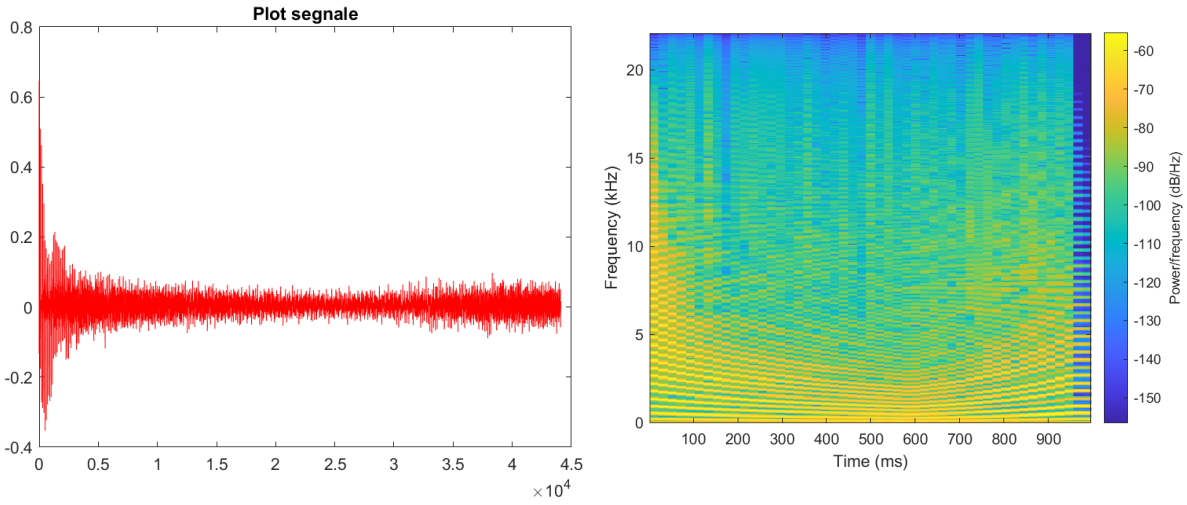


Figura 29 - Grafico del segnale di uscita usando il metodo dello Step Unitario con filtri All-pass

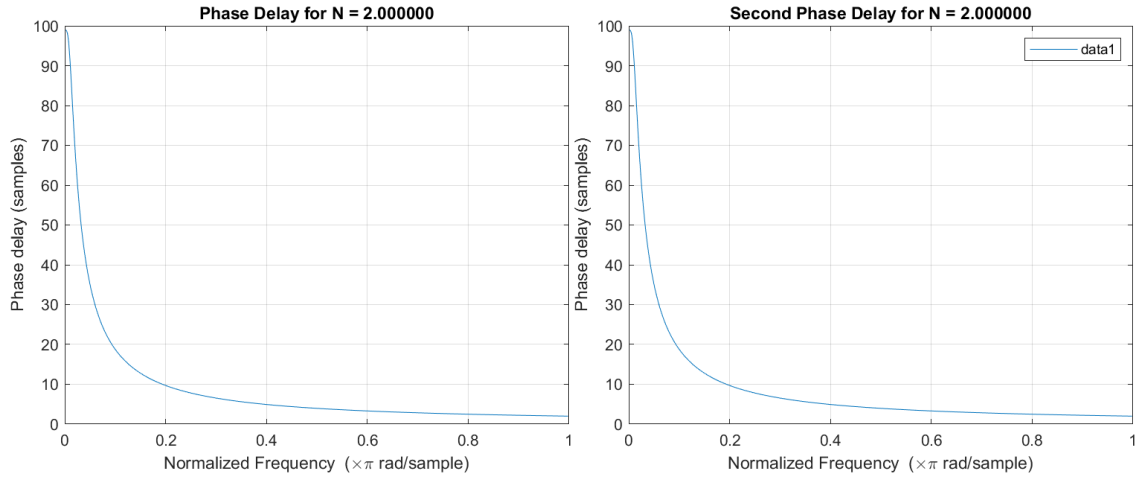


Figura 30 - Ritardo di fase rispettivamente del filtro H1(z) e del filtro H2(z)

A livello di costo computazionale vengono resi disponibili i dati di utilizzo della memoria e dei tempi di esecuzione tramite la seguente tabella:

Metodi	Tempo di esecuzione	Memoria usata da Matlab
Step Unitario	0.915030 secondi	2768 MB
Interpolazione Lineare	0.930137 secondi	2726 MB
Step Unitario con All-pass	2.772020 secondi	2797 MB

Tabella 2 - Confronto tra i metodi con parametri di tempo e memoria

Come si nota dai dati riportati, considerati i pregi introdotti dall'ultimo metodo il tempo di esecuzione è circa il triplo, ma si parla sempre di un tempo bassissimo mentre la memoria usata è paragonabile con quella impiegata dagli altri metodi con un piccolo aumento.

4 Conclusioni Finali

Si sono quindi studiati diversi metodi per la variazione della linea di ritardo per poter poi sfruttare questo principio per generare l'effetto del Pitch Bending, obiettivo principale. Tra i metodi proposti e discussi sicuramente quello degno di nota è quello che implementa lo Step Unitario con l'uso di filtri All-pass che permette di dimostrare e sottolineare l'importanza di questi filtri e i vantaggi che essi introducono a livello sonoro, per la presenza di maggiori dettagli e di un suono estremamente "fluido", eliminando almeno all'udito umano gli artefatti digitali e ottenendo un ottimo rendimento anche a livello computazionale.

5 Bibliografia e Sitografia

G. Albertini, “Appunti di elettromagnetismo, ottica e onde”, 2017, cap. Appendice 11 “Onde Stazionarie”, pp 201-205

Vesa Välimäki, “Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters”, 1995, cap. 3.4:” Design Methods for Fractional Delay Allpass Filters”, pp 105 – 113

Vesa Välimäki, “Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters”, 1995, cap. 3.5:” Time-Varying Fractional Delay Filters”, pp 114 – 122

Charles R. Sullivan, “Extending the Karplus-Strong Algorithm to Synthesize Electric Guitar Timbres with Distortion and Feedback”,1990, Computer Music Journal, Vol. 14, No. 3, pp 26 – 37 [Online]. Available: https://www.jstor.org/stable/3679957?read-now=1&seq=1#page_scan_tab_contents [Consultato il giorno 03-09-2021]

David A. Jaffe and Julius O. Smith, “Extensions of the Karplus-Strong Plucked-String Algorithm”,1983, Computer Music Journal, Vol. 7, No. 2, pp 56– 69 [Online]. Available: https://www.jstor.org/stable/3680063 [Consultato il giorno 03-09-2021]

Cristian Secchi, Dispense “Controlli digitali”, cap. “Sistemi a tempo discreto”, pp 22 – 33 [Online]. Available: http://www.automazione.ingre.unimore.it/pages/corsi/materialedidattico/CD0910/CD02-Sistemi%20a%20tempo%20discreto.pdf [Consultato il giorno 05-09-2021]

G.Grossi Dispense “Metodi per l’Elaborazione dei segnali”,2016 , cap. 8:” Filtri Digitali FIR e IIR”, pp 1 – 8 [Online]. Available: http://ens.di.unimi.it/dispensa/cap8.pdf [Consultato il giorno 08-09-2021]

Julius O. Smith III, “Physical Audio Signal Processing”, W3K Publishing, 2010, ISBN 978-0-9745607-2-4. Copyright © 2021-05-21, Center for Computer Research in Music and Acoustics (CCRMA), Stanford University [Online]. https://ccrma.stanford.edu/~jos/pasp/Delay_Line_Interpolation.html#sec:delaylineinterp [Consultato il giorno 10-09-2021]

Julius O. Smith III, “Physical Audio Signal Processing”, W3K Publishing, 2010, ISBN 978-0-9745607-2-4. Copyright © 2021-05-21, Center for Computer Research in Music and Acoustics (CCRMA), Stanford University [Online]. https://ccrma.stanford.edu/~jos/pasp/Linear_Interpolation.html [Consultato il giorno 10-09-2021]

“Ritardo di gruppo e ritardo di fase”, [Online]. Available: https://gaz.wiki/wiki/it/Phase_delay [Consultato il giorno 15-09-2021]