Sviluppo di un ambiente software per la gestione dei modelli per la sintesi audio additiva.

Graziano Bertini[*], *Lucio Di Giovannantonio*[**], *Massimo Magrini*[**]
Leonello Tarabella[*]

Progetto Finalizzato CNR MADESS II
Sottoprogetto SP3 Architetture e Sistemi VLSI
Linea di ricerca 3.2.2 Sistemi Multimediali

* IEI/CNUCE – CNR
** Collabotori esterni IEI

Indice

1 Sommario

In questa nota vengono riportate le problematiche relative allo sviluppo di un ambiente software su personal computer per la creazione e la gestione di un archivio di modelli di timbro, per il controllo di un sistema VLSI basato sulla sintesi additiva (Progetto Finalizzato CNR MADESS II, Linea di Ricerca 3.2.2 – 1998/2001).

Basandosi su risultati di una procedura di analisi di segnali musicali (basato sulla tecnica di approssimazione lineare, *PLA*) sperimentata in precedenza utilizzando l'ambiente Matlab, nella relazione che segue viene data una descrizione formale del modello di timbro. Inoltre e' stato messo a punto un ambiente software, utilizzando il linguaggio C++, per l'estrazione dei parametri caratterizzanti il modello, per la modifica degli stessi e per la creazione di modelli virtuali. E' stata inoltre implementata una procedura di resintesi che permette l'invio dei dati di codifica, relativi ad un modello di timbro, verso il simulatore software del chip VLSI.

2 Introduzione

Le problematiche relative alla sintesi additiva e al suo crescente interesse negli ultimi anni, sono descritte in una precedente nota tecnica [BMT99], mentre una descrizione completa del sistema di sintesi di riferimento al nostro progetto (Synth chip) e' ampiamente descritto nella relativa bibliografia [BRST97] [BRST97&B] [BRST&B98].

Un problema fondamentale da tenere in considerazione nella messa a punto del sistema è il controllo dell'elevato numero di segnali elementari prodotti dal chip (e a maggior ragione da più chip); infatti ogni singolo segnale richiede almeno tre parametri da aggiornare continuamente a microlivello a cui si aggiunge la gestione dei controlli a livello degli "inviluppi d'ampiezza" e a livello dei "cambi di nota". Questi dati devono essere forniti ai chip di sintesi tramite una apposita unità di interfaccia (verso un host, tipicamente un PC, o verso un dispositivo di input musicale, tipo tastiera MIDI) la quale deve gestire il flusso dei dati in maniera efficiente. In una prima istanza questa unità era stata pensata essere basata su un microprocessore per DSP general purpose; in una successiva rielaborazione del progetto, invece, è stato ipotizzato l'impiego di un altro chip VLSI dedicato realizzato per questo compito di controllo. E' quindi di notevole importanza la ricerca di una tecnica di modellizzazione degli spazi timbrici da sintetizzare che permetta di aggiornare le grandezze caratteristiche delle singole parziali solo in caso di variazioni significative, in modo da ridurre il più possibile lo scambio di dati dall'host verso il controllore citato sopra.

Lo scopo di questa nota è quello di illustrare lo sviluppo di un ambiente software per la gestione dei modelli di timbro nell'ambito della sintesi additiva, che consenta la creazione di un archivio di tali modelli da utilizzare nell'ambito di composizioni sonore. L'ambiente in questione è composto da una serie di tools che consentono la modellazione dei parametri del timbro secondo la codifica adottata, e quindi la creazione di un archivio di modelli di timbro, e da un'altra serie di tools per il controllo e l'invio di tali parametri al sistema di sintesi al fine

di minimizzarne il numero, in modo da consentire una sintesi in "tempo reale" a partire da eventi MIDI. Una volta scelti i modelli da includere nella composizione, mediante uno strumento MIDI è possibile controllare in tempo reale i suoni sintetizzati.

Lo sviluppo di un gestore di modelli di timbro come quello che verrà proposto nel lavoro e che costituisce lo scopo di questa nota, và al di là dell'effettiva disponibilità del chip di sintesi a cui fa riferimento. Infatti data l'elevata potenza di calcolo raggiunta dagli attuali personal computer è possibile implementare la sintesi additiva direttamente su host via software, realizzando un discreto numero di oscillatori digitali e implementando la procedura di resintesi mediante algoritmi opportuni.

3 Architettura del sistema, cenni

Di seguito viene riportato un cenno all'architettura del sistema di sintesi, ampiamente descritto nella bibliografia già citata; ricordiamo che il chip di sintesi può generare in tempo reale 1200 sinusoidi con caratteristiche di buona qualità. Le onde seno sono ricavate matematicamente con filtri IIR risonanti, senza usare la LUT (LookUp Table) semplificando notevolmente il progetto globale. La lunghezza interna di parola adottata (20 bit) consente di ottenere un elevato rapporto segnalerumore (– 80 dB).

Il sistema prevede inoltre la possibilità di impostare i parametri per la sintesi, ampiezza, frequenza e fase a intervalli molto brevi, per sopperire ai rapidi cambiamenti nelle caratteristiche del suono e infine ha la possibilità di ottenere in uscita otto canali indipendenti. La gestione del passaggio dei parametri non può essere eseguita direttamente dall'host in quanto quest'ultimo non è in grado di garantire un flusso continuo di dati, fattore essenziale per l'esecuzione dei brani sonori. La soluzione adottata prevede quindi un controllore CTRL che ha il compito di tradurre i blocchi di comandi che arrivano dall'esterno in parametri accettati dal chip di sintesi mantenendo continuo il flusso di dati significativi verso quest'ultimo. L'architettura del sistema può essere riassunta nella figura seguente

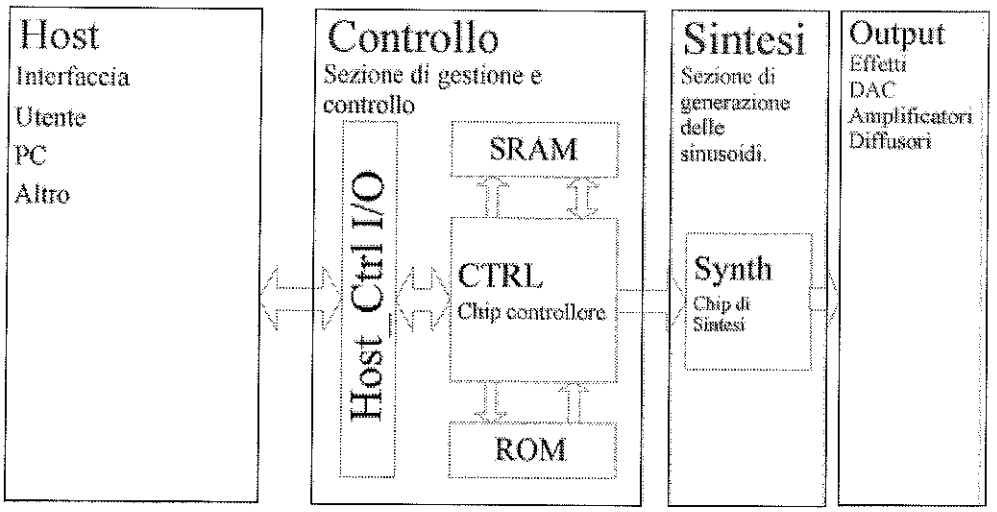

Figura 3-1 Architettura del sistema di sintesi proposto

Il nostro interesse, nella presente nota, è rivolto alla sezione relativa all'host che ha il compito di fornire un interfaccia utente per la gestione del database, e di inviare i comandi al controllore del chip

3.1 Simulatore software

Lo sviluppo hardware di chip VLSI è un'operazione complessa e costosa, quindi prima di procedere alla realizzazione del prototipo occorre procedere ad una accurata fase di validazione del progetto secondo le specifiche funzionali e hardware. Nel caso del presente chip, tale livello di simulazione non è sufficiente, dato che l'uscita del chip consiste in una sequenza di campioni equivalenti a un segnale audio, deve quindi essere in possesso di determinati requisiti in termini di qualità sonora (non devono essere introdotti disturbi, distorsioni e artefatti di vario tipo). Per effettuare questa fase di simulazione è stato approntato un simulatore software che ricalca il più possibile lo schema di principio del progetto stesso, al fine di analizzare le variazioni delle sinusoidi in uscita dai filtri e dai canali al variare dei parametri di ingresso di segnali campioni forniti ai singoli filtri IIR. Tramite una adeguata sperimentazione sui dati ricavati dall'analisi di segnali reali è stato ottimizzato il progetto del chip di sintesi apportando alcune modifiche all'hardware proposto inizialmente (lunghezze di parola per i parametri, intervalli temporali ecc).

Il simulatore è stato sviluppato usando il linguaggio C++ in un precedente lavoro di tesi [Ber2000]; di seguito si riportano alcune specifiche e le modalità di impiego. Il file del simulatore è costituito dall'eseguibile *oscill.exe* il cui funzionamento è descritto dalla seguente sintassi DOS:

oscill siminp.sco

il file di ingresso siminp.sco è appositamente creato da una procedura di resintesi, che organizza i dati del modello di timbro in cluster così come richiesto dal controllore. Una volta prodotto il file *siminp.sco*, possiamo far generare al simulatore i campioni relativi agli 8 canali nei file *channel, channel1, ...channel7.* questi file possono essere esportati successivamente ad esempio come file wave.

Il file in input al simulatore deve contenere i dati così come accettati dal controllore del chip, occorre quindi operare una trasformazione secondo le seguenti regole:

- inserire nel file di input al chip una intestazione che prevede alcuni campi quali il numero d'ordine del cluster, il volume, il numero di oscillatori che devono operare, ecc.;

- trasformare alcune grandezze a seconda del numero di bit con cui vengono rappresentate all'interno del chip, cioè per ogni parametro esiste una funzione che "mappa" il suo valore reale in quello accettato dal controllore.
  - la frequenza (20 bit) F ([0, 22050]) [0; 220 –1]
  - l'ampiezza (16 bit) F ([0, 1]) [0; 216 –1]
  - variazione di ampiezza (16 bit) F ([0, 1]) [0; 216 –1];

- creare un nuovo cluster quando per l'esecuzione del brano musicale è necessario allocare altri oscillatori;

- disallocare alcuni oscillatori dai cluster, quando non vengono più utilizzati (ad esempio alla fine della fase di attacco);
- eliminare un cluster, quando gli oscillatori associati non sono più necessari;

4 La creazione dell'archivio dei modelli di timbro

Il timbro, come accennato nel primo capitolo, è un attributo multidimensionale del suono che permette di distinguere due suoni aventi stessa intensità, ampiezza e durata; in questo capitolo ne daremo una descrizione più dettagliata, arrivando alla formulazione di un modello di timbro che verrà utilizzata per la sua modellazione nella procedura di sintesi; scopo di questo capitolo è quello di illustrare la creazione di un archivio di modelli di timbro da utilizzare nell'ambito della sintesi additiva. Verranno presi in considerazione le caratteristiche salienti che descrivono un modello di timbro e verrà illustrata la procedura di analisi che permette di implementare il modello di timbro in termini matematici oltre che di gestirlo mediante una struttura dati. Verranno fornite altresì alcune procedure per la sua manipolazione da parte dell'utente per permettere la creazione di nuovi tipi di sonorità, sia basate su modelli esistenti, sia create a proprio.

4.1 La procedura di analisi

Per i nostri scopi assumiamo che il timbro sia rappresentato principalmente dalla composizione spettrale del suono, che come è noto, in generale è variabile nel tempo. Occorre innanzitutto precisare che non esiste una singola rappresentazione ottima, non solo per tutti i modelli di suono che potremmo voler analizzare, ma anche per parti di esso; daremo di seguito una panoramica sui modelli più utilizzati, con la precisazione che possono essere usati anche in combinazione fra loro per analizzare qualsiasi tipo di suono:

- *Short-Time Fourier Transform (STFT)*: il modello derivante da questo tipo di analisi è il più generale ma anche il meno flessibile;
- *Sinusoidale*: consente un livello di astrazione maggiore, modellando uno spettro tempo-variante mediante somma di sinusoidi tempo-varianti;
- *Sinusoidale con residui*: in questo modello la rappresentazione sinusoidale viene utilizzata solo per la parziali stabili, i residui vengono modellati mediante componenti stocastiche (rumore filtrato);
- *High Level Attributes*: oltre ai parametri estratti con uno dei modelli precedenti vengono utilizzate altre informazioni che caratterizzano il suono (andamento dei parametri nel loro insieme) che ne permettono una descrizione più qualitativa (in senso anche psicoacustica), ovviamente più parametri aggiungo migliore sarà la rappresentazione ottenuta.

La scelta fra le varie rappresentazioni dipende dagli obiettivi che ci siamo prefissi, che in genere corrispondono a: (1) qualità del suono, (2) flessibilità, (3) memoria richiesta e (4) complessità computazionale; nella maggior parte dei casi l'interesse è rivolto a massimizzare i primi due e a minimizzare gli ultimi due, in quest'ottica

possiamo affermare che la STFT si rivela una buona tecnica per quanto riguarda la qualità e il costo computazionale, ma meno buona per quanto riguarda flessibilità e consumo di memoria; L'impiego delle sole sinusoidi è migliore per quanto riguarda la flessibilità ma con un maggior costo computazionale, mentre la l'aggiunta di rumore modellato può essere considerata una generalizzazione della STFT, offre una buona flessibilità basso consumo di memoria; infine la tecnica High Level Attributes offre la maggior flessibilità, però con un maggior costo computazionale [XS97].

Le procedure di analisi ST-FT e riduzione dei dati sono state ampiamente descritte nella nota [BMT99], in questa sede è stato fatto il porting dall'ambiente Matlab al C++ e sono state implementate alcune funzioni di supporto. In seguito ci occuperemo in modo particolare della ricerca della fondamentale, procedura cha ha subito i maggiori cambiamenti dovuti all'ampliamento della casistica di suoni analizzabili (estensione della procedura a suoni non armonici)

4.1.1 L'estrazione della fondamentale

La frequenza fondamentale è uno degli elementi principali di un modello di timbro, quindi il primo passo è quello di fornire una procedura di analisi per l'estrazione di tale parametro. Sono stati presentati in letteratura diversi algoritmi per tale stima, sia nel dominio del tempo che nel dominio della frequenza, oggi comunque si preferisce operare nel dominio della frequenza, quello utilizzato in questa sede [BMT99] si basa semplicemente sull'analisi di Fourier del segnale audio, o di parte di esso: una volta letti i campioni del segnale audio, si applica l'algoritmo della FFT e se ne calcola lo spettro di ampiezza, ottenuto ciò abbiamo un vettore con i vari campioni della FFT sul quale indagare per la nostra ricerca. L'individuazione automatica della riga dello spettro corrispondente alla frequenza della fondamentale è semi-automatica: viene proposta come principale candidata la *prominent frequency*, cioè quella a cui corrisponde il massimo valore dell'energia; ciò tuttavia non è vero in generale, sarà cura dell'utente la scelta della frequenza effettiva mediante investigazione e selezione sul grafico nel quale viene rappresentato lo spettro di ampiezza da investigare.

Esempio: *dalla figura [4.1] possiamo vedere che come prominent frequency viene proposta la frequenza di 1050 Hz, mentre possiamo notare che in realtà si tratta della seconda armonica.*

La procedura originaria era stata realizzata in Matlab e in questa sede ne è stato realizzato il porting nell'ambiente di sviluppo Borland C++ Builder utilizzando il linguaggio C++, sia per aver un software totalmente svincolato dal Matlab in modo da poter essere utilizzato come applicazione *stand-alone* senza dover ricorrere ad un programma aggiuntivo, sia per questioni di efficienza legata ai tempi di calcolo di una funzione di Matlab rispetto ad una funzione C++ opportunamente compilata ed ottimizzata dal compilatore. In un primo momento è stata presa in considerazione l'idea di compilare le funzioni Matlab in una DLL da utilizzare nel programma in realizzazione; date le difficoltà incontrate in sede di realizzazione della DLL per problemi di compatibilità software fra Matlab e l'ambiente di sviluppo utilizzato si è pensato di riscrivere tale funzione direttamente in linguaggio C++. A tale scopo sono state implementate delle

procedure per la gestione dei file in formato Wave, e una procedura che realizza l'algoritmo della FFT così come realizzato dalla funzione interna di Matlab.

Alla procedura di analisi è stata aggiunta la possibilità di consentire la scelta, da parte dell'utente, del punto da cui partire con l'analisi per la ricerca della fondamentale, in modo da poter saltare la parte di suono riguardante la fase di attacco, che come esposto nel primo capitolo è caratterizzata da componenti spettrali che si evolvono rapidamente e che quindi potrebbero alterare la determinazione della frequenza fondamentale.

I risultati sono stati molto incoraggianti sia in termini di differenza di tempi di calcolo che di accuratezza dei risultati.

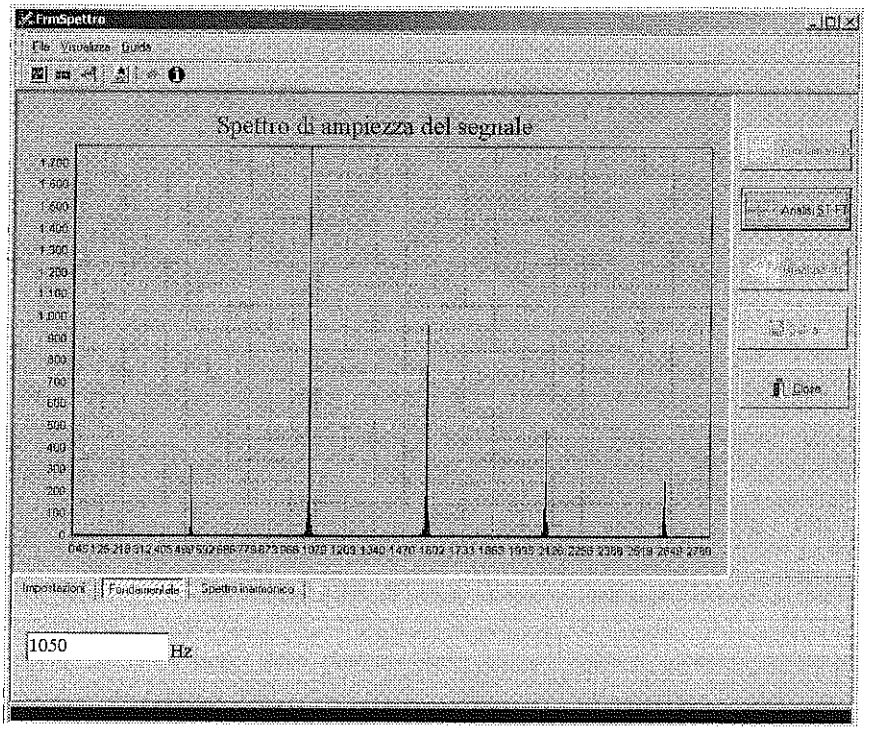

Figura 4-1 Differenza fra prominent frequency e fondamentale

4.1.2 Modifiche sostanziali apportate alla procedura di analisi

E' da notare che il procedimento di analisi descritto porta dei risultati corretti solo nel caso di suoni armonici, in caso contrario l'analisi STFT produce risultati inesatti in quanto le varie frequenze non sono multiple di una fondamentale e le finestre per l'analisi STFT non sono correttamente dimensionate. Nel caso di suono armonico le parziali successive vengono automaticamente fissate a frequenza multipla della fondamentale, la seconda a $2*Freq_{Fond}$, la terza a $3*Freq_{Fond}$, e così via, nel caso di suono inarmonico invece le frequenze sono posizionate diversamente senza nessun rapporto intero con la prima; in tal caso bisogna intervenire manualmente sul grafico dello spettro per fissare le frequenze desiderate e calcolare il rapporto con la prima parziale. Per l'analisi successiva, non possiamo dimensionare le finestre allo stesso modo in quanto non si avrebbe una corretta risoluzione, si procede allora andando ad indagare su ogni singola riga mediamente una DFT, con una finestra di analisi di

dimensione sufficientemente grande. La ricerca dei breakpoints prosegue poi allo stesso modo; l'unica variazione sul risultato finale consiste nel fatto che per ogni parziale è necessario memorizzare anche il rapporto frequenziale con quella a frequenza minore.

4.1.3 Considerazioni relative ai parametri per il rumore di attacco

E' opportuno fare delle considerazioni sulla fase di attacco, tale fase come noto è caratterizzata da componenti che si evolvono rapidamente, è possibile quindi pensare di generare un modello comprendente anche queste parziali, da eliminare però dal processo di generazione appena terminata la fase di attacco. In molti casi però nella fase di attacco compaiono delle componenti di rumore non armonico, composto da un insieme di parziali contigue abbastanza riconoscibili come mostrato in figura [4.2]. La figura seguente mostra lo spettro di una nota di fagotto emessa ad un livello espressivo **f**, comprendente anche la parte di attacco, si noti come vicino alle armoniche nella zona 400, 500 Hz vi sia un' insieme di parziali disposte in maniera irregolare ma comunque centrate attorno alle armoniche stesse.

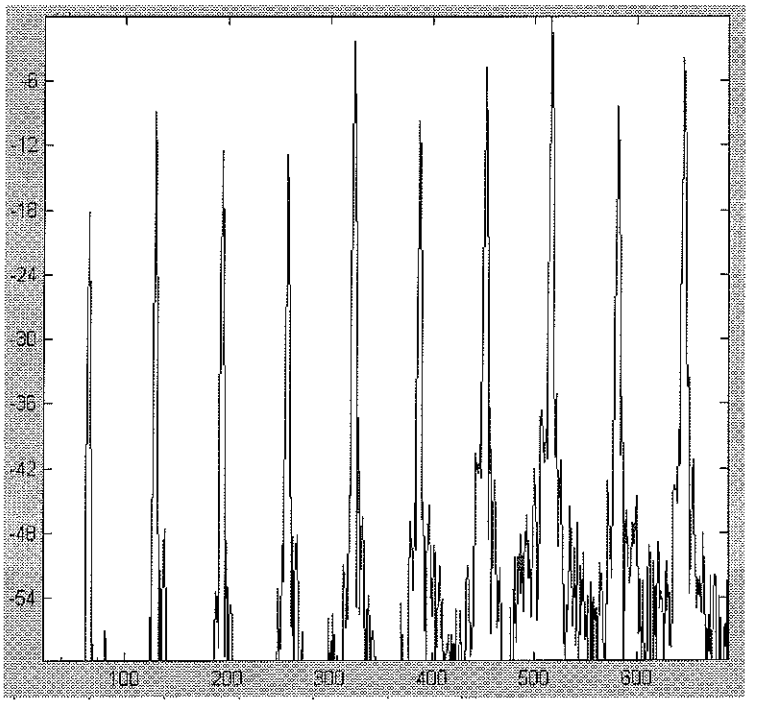

Figura 4-2 Rumore d'attacco

Il tentativo di generazione del suono con la definizione diretta di tali parziali mediante l'aggiunta al modello timbrico non porta a risultati soddisfacenti, a meno che il numero di parziali aggiunte sia molto elevato. Anche l'andamento energetico di tali parziali è difficile da descrivere e modellare. L'origine di tali componenti di rumore, negli strumenti acustici, è in genere da ricondurre alla

parte eccitatrice del sistema (soffio, percussione, pizzico), che nella parte di attacco una rilevanza maggiore in quanto il fenomeno della generazione non è ancora entrato a regime. Tuttavia le componenti del segnale eccitatore, in genere con uno spettro ampio, interagiscono in qualche modo con il sistema risonante o vibrante, con l'effetto che solamente quelle che sono prossime alle frequenze risonanti (o le loro armoniche) del sistema vengono esaltate. Nella figura seguente è riportato lo spettro di un segnale ottenuto modulando in frequenza una sinusoide a 440 *Hz* con un segnale casuale con frequenza di variazione di alcuni *ms*.

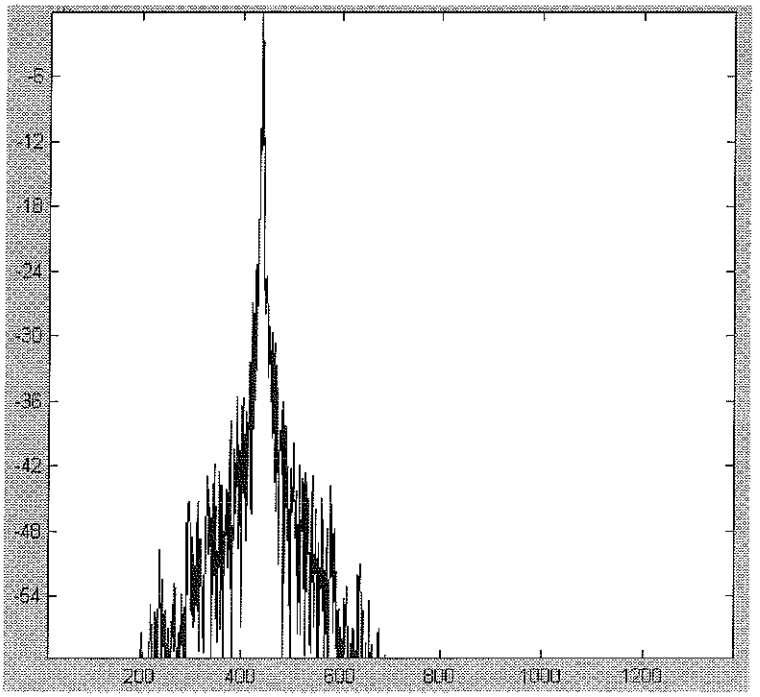

Figura 4-3 Modulazione della frequenza con un segnale casuale

Si può ottenere un comportamento simile a quello delle parziali di rumore nell'intorno delle armoniche della nota di fagotto vista precedentemente. Queste considerazioni hanno suggerito un semplice metodo che produce risultati interessanti almeno su una certa categoria di suoni [BMT99]. Questo metodo consiste nell'effettuare una modesta modulazione in frequenza delle armoniche nella sola parte di attacco, cioè variarne leggermente il valore ad ogni frame, nell'intorno del valore centrale. Questo metodo non introduce ulteriori breakpoint per l'aggiornamento delle frequenze, in quanto nella prima fase di attacco i breakpoint si succedono frame dopo frame, avendo l'inviluppo iniziale una pendenza ripida ed in genere irregolare. Pertanto basta variare le frequenze, in questa fase, nel modo seguente:

$$f = f + V * rand() * \frac{f}{2} - \frac{f}{4} \qquad [4.1]$$

dove f è la frequenza dell'armonica, *rand()* è una funzione che dato un intero ne restituisce uno casuale compreso tra esso e 0, infine V è un valore compreso tra 0 e 1 che varia con un inviluppo nella fase dell'attacco, stabilendo una sorta di indice di modulazione dinamico. Questo inviluppo può ad esempio avere una forma triangolare incentrata sulla metà della fase di attacco. Questo metodo è

stato sperimentato, ottenendo discreti risultati, su segnali di strumenti a fiato come il fagotto. E' stato sperimentato anche su segnali dall'andamento percussivo, come la chitarra. Sebbene questa categoria di suoni non sembri adatta a questo tipo di manipolazione, l'irregolarità introdotta nella primissima fase dell'attacco, congiuntamente ad una leggera espansione della dinamica delle parziali in questo intervallo, rende più credibile l'effetto "pizzicato" caratteristico di questi suoni.

4.2 Il modello di timbro

Una volta terminata l'analisi e la ricerca delle varie caratteristiche del suono da analizzare, è possibile creare il modello timbrico. Tale modello riassume le informazioni riguardanti le parziali del suono, e si può schematizzare in una semplice struttura dati. Un semplice esempio di possibile modello, che tuttavia può rappresentare un grande numero di suoni, è il seguente:

- ***Fond:*** frequenza fondamentale del suono;
- ***N_Arm:*** numero di armoniche utilizzate per la resintesi;
- ***N_Brek:*** numero di breakpoints utilizzati nella resintesi;
- ***N_Frames:*** durata del suono originale in frames;
- ***Brekpoints:*** vettore di lunghezza ***N_Brek***, che riporta per ogni breakpoint il numero di frame in cui esso occorre (si ricorda che ogni frame dura 128 campioni);
- ***Rapporti***: vettore di lunghezza ***N_Arm***, che riporta i valori dei rapporti delle armoniche/parziali con la prima,
- ***Matr_Brek:*** matrice di dimensioni ***N_Arm X N_Brek***, che riporta il valore delle ampiezze delle armoniche/parziali per ogni breakpoint. Tali valori sono normalizzati nell'intervallo 0-1;
- ***startSustain:*** numero di frame in cui si ritiene inizia la fase di sostegno (se esiste) del suono.
- ***startRelease:*** numero di frame in cui si ritiene inizia la fase di sostegno (se esiste) del suono.

Se si vuole inoltre distinguere suoni percussivi da suoni sostenuti, che potrebbero essere trattati in maniera diversa dal sistema di controllo della sintesi (su host), si può aggiungere un flag che segnali la tipologia del suono.

- ***Tipo:*** intero che vale 0 se suono sostenuto, 1 se percussivo ecc..

Si può pensare di includere nel modello anche il vettore che descrive il rapporto delle ampiezze delle armoniche ad un livello di volume differente, in modo da poter sintetizzare facilmente suoni a diversi livelli espressivi. In questo caso si potrebbe modellare il suono a maggior contenuto armonico, e includere il rapporto

delle ampiezze rispetto a quello a minor contenuto (generalmente ad un livello espressivo minore, es. *p*, o *pp*)

- ***AmpRatio:*** vettore di lunghezza N, dove N numero di armoniche. Può essere sfruttato per fornire al controllore quelle informazione che permettono la diversificazione delle variazioni dell'ampiezza delle singole parziali al variare dell'ampiezza complessiva del suono da generale.

Si noti che lo stesso procedimento potrebbe anche essere adottato per eventuali variazioni del contenuto timbrico in relazione alla variazione del registro (fondamentale). Potrebbe cioè essere incluso un vettore che descrive la variazione energetica di ogni parziale del suono nel caso del cambio di registro della fondamentale. Questa tecnica non è stata comunque sperimentata durante lo svolgimento del lavoro.

5 Architettura e sviluppo del software

Lo scopo e' quello di fornire all'utente un ambiente completo mediante il quale gestire i parametri del modello di timbro come descritto nel precedente capitolo, ed utilizzarli in tempo reale per la produzione sonora mediante il controllo di un sistema di sintesi software o hardware.

5.1 Specifiche del sistema

Il sistema software in progetto costituisce l'interfaccia su host verso il mondo esterno del sistema di sintesi di riferimento, con il duplice compito di fornire i dati al controllore del chip, e di consentire all'utente la modellazione di tali dati secondo il risultato sonoro desiderato. Naturalmente le funzionalità saranno sviluppate in modo da operare in un ambiente costituito da un'interfaccia grafica user-friendly stile Windows che ne consente l'utilizzo anche all'utente meno esperto. Possiamo subito dividere il progetto in due grossi moduli: uno che si occupa della gestione dei dati da parte dell'utente permettendo la creazione del database e uno che si occupa dell'interfacciamento col sistema di sintesi.

5.1.1 Gestione database e interfaccia utente

Questo modulo ha il compito di fornire l'interfaccia con l'utente e una serie di procedure per la creazione e gestione del database; esso può essere ulteriormente scomposto in tre sottomoduli, come mostrato in figura 4.1:

- un primo che consente la creazione di modelli di timbro di strumenti esistenti, utilizzando i dati forniti dalla procedura di analisi;
- un secondo che consente la modifica di modelli esistenti per provare ad ottenere nuovi tipi di sonorità;
- un terzo che consente la creazione di nuovi modelli inesistenti.

Nel proseguo del capitolo verranno fornite le specifiche dettagliate dei moduli e i dettagli implementativi relativi al software nella sua interezza.

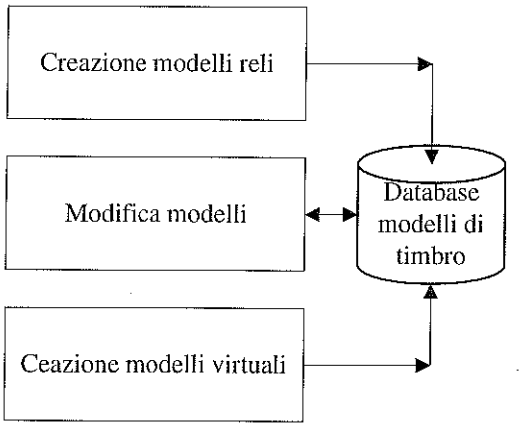

Figura 5-1 Moduli componenti l'applicazione

5.1.1.1 Creazione modelli reali

Il sottomodulo in esame si basa sulla procedura di analisi precedentemente descritta per l'estrazione dei parametri relativi al modello di timbro di uno strumento esistente; l'input è costituito da un file in formato Wave contenente i campioni del suono di cui si vuole estrarre il modello di timbro; la procedura può essere schematizzata con il diagramma a blocchi di figura 4.2.

La prima operazione è la ricerca della fondamentale e del vettore dei rapporti con la prima parziale nel caso di suono non armonico, la fondamentale oltre a far parte del modello di timbro serve per il calcolo della lunghezza della finestra per il passo successivo: l'analisi ST-FT; che permette il calcolo degli inviluppi delle varie armoniche; ultimo passo è il calcolo dei breakpoints

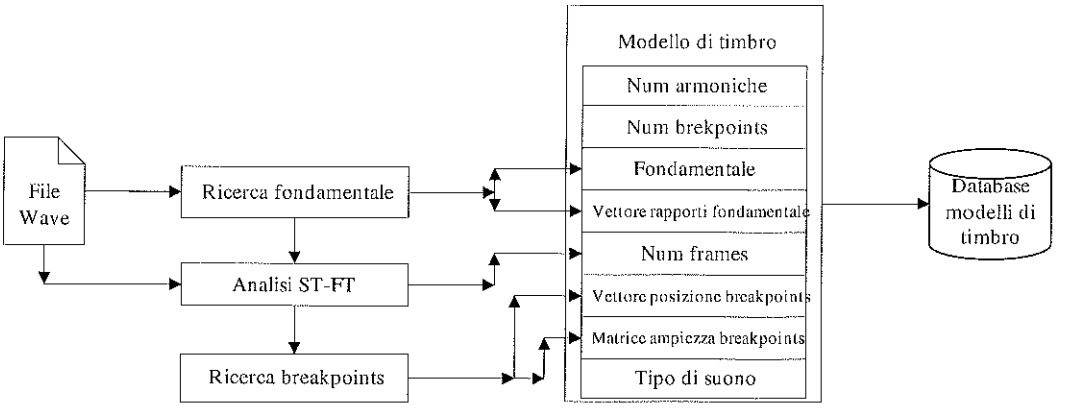

Figura 5-2 Struttura modello di timbro

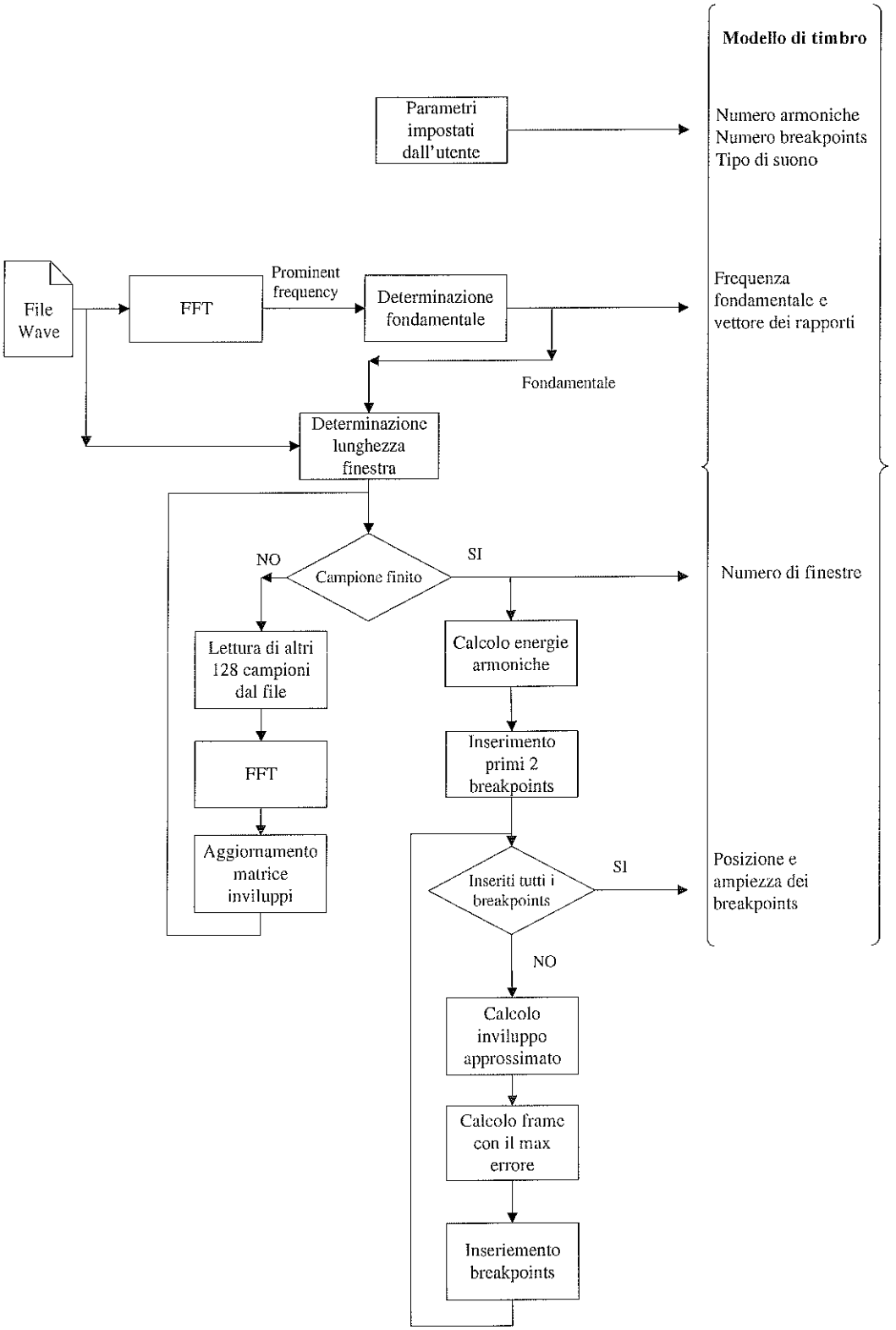

Figura 5-3 Diagramma a blocchi procedura di estrazione modello di timbro

5.1.1.2 Creazione modelli di timbro virtuali e modifica modelli esistenti

Questo sottomodulo permette all'utente di creare propri modelli di timbro da zero, permettendo la sperimentazione di nuovi tipi di sonorità; il modello di timbro viene realizzato secondo le caratteristiche descritte nei capitoli precedenti permettendo una modellazione puntuale degli inviluppi delle singole armoniche.

In una prima fase iniziale vengono impostati alcuni parametri del modello relativamente alla lunghezza del suono, al contenuto armonico e al numero di breakpoints da includere:

- il numero di armoniche;
- il numero di breakpoints;
- la durata in frame del suono;
- il tipo di suono (percussivo, non percussivo, ecc..);
- il contenuto armonico, relativamente alle frequenze delle armoniche superiori, proponendo dei modelli predefiniti;
- la forma dell'inviluppo delle armoniche, anche qui proponendo degli inviluppi predefiniti, impostando alcuni breakpoints secondo il modello di inviluppo scelto.

Sono stati così definiti alcuni parametri iniziali per la creazione del modello di timbro; la fase di modellazione vera e propria permette di modificare a piacimento i parametri impostati andando ad operare nei vari punti dell'inviluppo al fine di modificarne l'andamento intervenendo sui breakpoints (inserimento nuovi breakpoints, spostamento, variazione ampiezza ecc..) e sul contenuto armonico (variazione rapporto con la fondamentale per ottenere suoni di qualsiasi tipo, variazione dell'ampiezza delle singole armoniche ecc..).

Per quanto riguarda la modifica dei modelli esistenti, una volta caricato il modello, dalla finestra di editing è possibile intervenire sui punti dell'inviluppo per variarne i parametri come nella fase di creazione.

5.1.1.3 Il database

Vista la struttura del modello di timbro che comprende diverse matrici e vettori (vettore delle posizioni dei breakpoints, vettore dei rapporti con la fondamentale e matrice delle ampiezze dei breakpoints), di dimensioni consistenti (per alcuni tipi di suoni può essere necessario includere nel modello di sintesi circa 100 armoniche e altrettanti breakpoints), sarebbe risultato molto inefficiente organizzare il database secondo lo schema classico del modello relazionale;

E' stato deciso allora di utilizzare un file binario contenente direttamente tutti i dati del modello di timbro in un determinato ordine; il database in questo caso diventa una semplice raccolta di questi file. Al file che descrive il modello di timbro è stata assegnata un'estensione puramente descrittiva *tmb*, per facilitare l'utente nella ricerca dei file da caricare; al suo interno i dati vengono scritti nella seguente sequenza riga per riga:

- Nome modello;
- Fondamentale (espresso in hertz, intero);
- Numero di armoniche;
- Numero di frames;

- Tipo di suono (attributo ad alto livello);
- Inizio fase di sostegno (numero del frame);
- Inizio fase di rilascio (numero del frame);
- Numero di breakpoints;
- Vettore delle posizioni dei breakpoints (numero del frame in cui cade il breakpoints);
- Vettore dei rapporti con la fondamentale;
- Matrice delle ampiezze delle componenti in corrispondenza dei breakpoints (normalizzati nell'intervallo [0-1].

5.1.2 Allocazione delle parziali verso il sistema di sintesi

Questo modulo ha il compito di gestire l'allocazione dei parametri verso il sistema di sintesi, prelevando i modelli dal database e l'input da un file MIDI o da un controller MIDI. Nelle prove sperimentali effettuate per ogni strumento si è generata una sola nota basata sul modello timbrico costruito, allo scopo di condurre prove comparative con il segnale originale. Questo è stato fatto generando gli opportuni comandi accettati dal simulatore, conformi a quella che dovrebbe essere l'interfaccia verso il *sistema controllore+chip di sintesi;* la procedura che effettua questo test può essere descritta dal diagramma a blocchi di figura 4.5, dove il valore della frequenza *n*-esima è data dalla seguente relazione:

$$f_n = F_0 \cdot Rapporto[n] \cdot \frac{1048575}{44100} \qquad [5.1]$$

mentre la variazione di frequenza viene posta a zero; il valore dell'ampiezza è data da:

$$ValoreBrek \cdot 2^{16} - 1 \qquad [5.2]$$

mentre la variazione di ampiezza dalla pendenza della spezzata che congiunge il breakpoint precedente con quello in questione.

Per procedere nella sperimentazione sarà necessario come passo successivo sviluppare un modulo che, a partire da un file che descrive gli eventi musicali ad un livello macroscopico (e.g. uno standard MIDI file), crea un file di *comandi* adatto ad essere eseguito dal simulatore; in questo caso possiamo fare riferimento al diagramma a blocchi di figura 4.6. In questo modo si rende possibile la verifica di impasti sonori formati da più note o modelli timbrici simultanei, e verificare anche possibili colli di bottiglia della generazione polifonica in casi realistici.

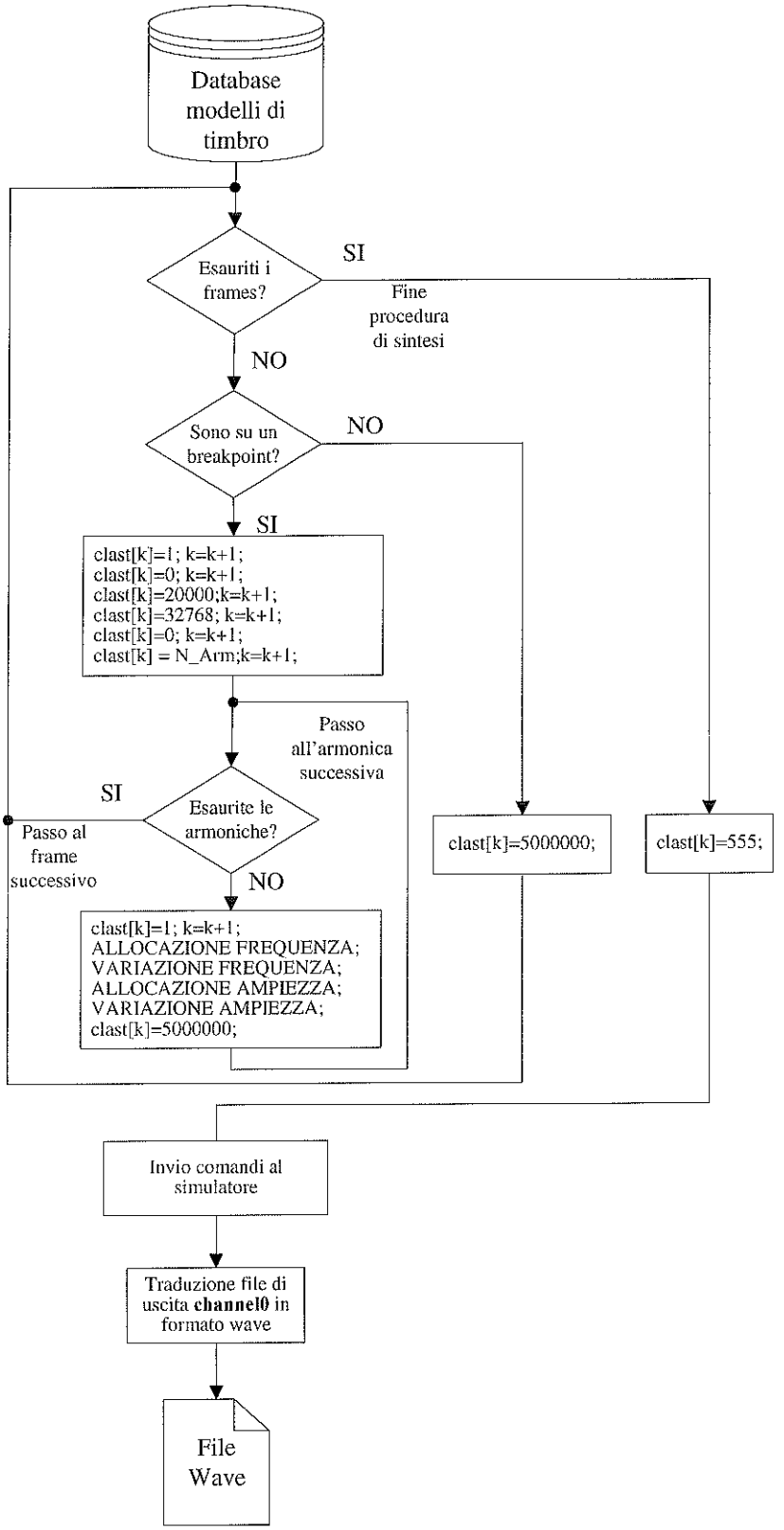

Figura 5-4 Procedura di sintesi per invio comandi al simulatore

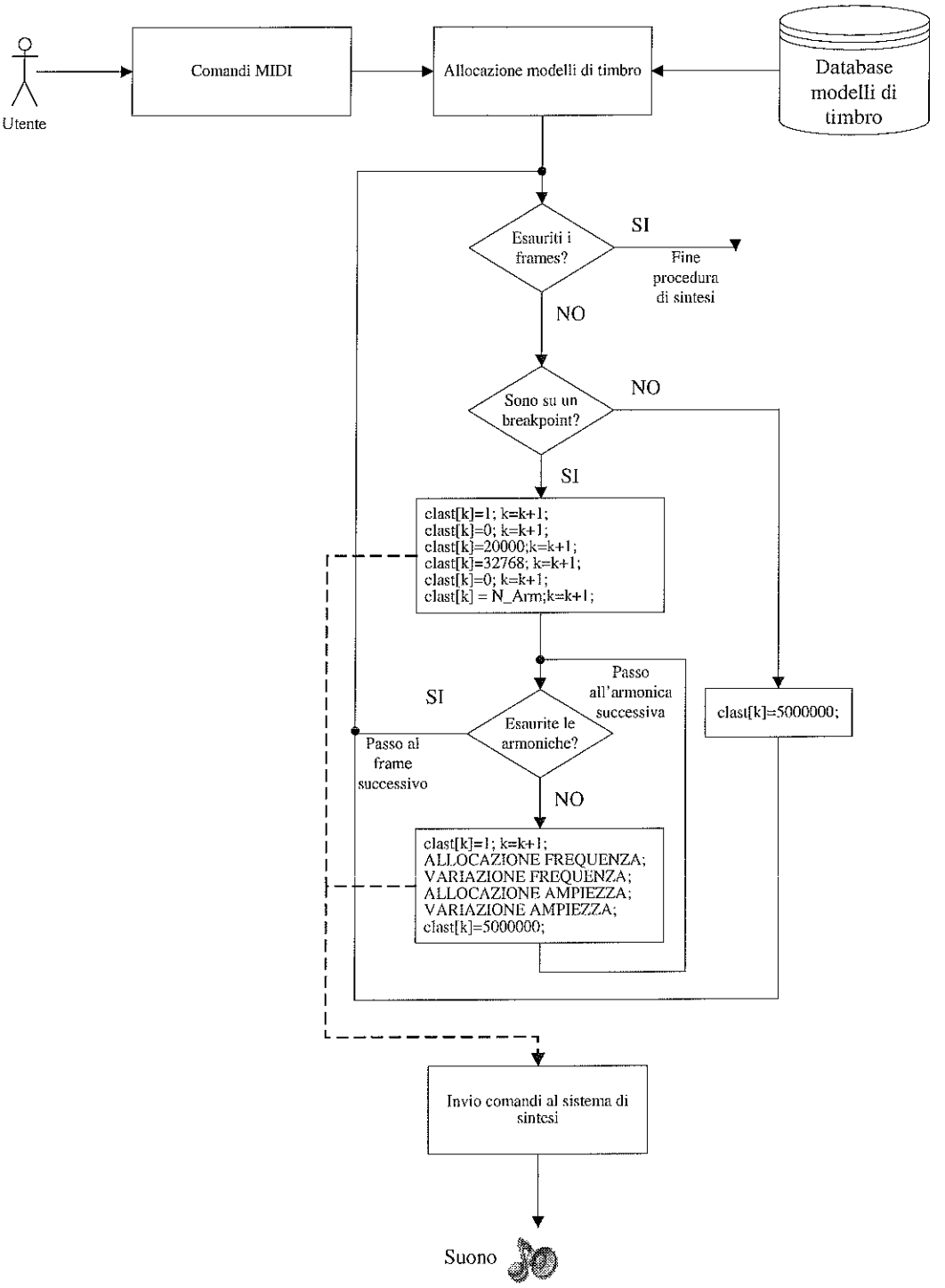

Figura 5-5 Procedura di sintesi per invio comandi al sistema

Ad esempio supponiamo di avere associato sul canale MIDI numero 1 un modello timbrico avente tre parziali con una fase di attacco e di rilascio; la ricezione di un messaggio MIDI di tipo NOTE-ON sul canale relativo, dovrà essere interpretato dal modulo in esame allocando sul chip di sintesi un'insieme di tre parziali con dei parametri correlati al modello di timbro prelevato dal database. Alla ricezione del NOTE-OFF verranno inviati i comandi relativi alla fase di rilascio del modello timbrico. Una volta terminata la fase di rilascio le tre parziali

verranno disallocate. Altri tipi di messaggi MIDI come ad esempio il cambio di volume o il *pitch-bend*, analogamente possono essere tradotti in opportuni comandi verso il sistema di sintesi.

5.2 Progettazione del software

I moduli descritti precdentemente sono stati implementati utilizzando il software di sviluppo Borland C++ Builder versione 5.0, un ambiente di sviluppo molto potente che permette di sviluppare applicativi in C++ mediante una programmazione di tipo RAD, combinando la potenza del linguaggio C++ con la semplicità della programmazione visuale.

L'ambiente di sviluppo si presenta come una semplice barra dei menù dalla quale vengono richiamati i vari moduli che implementano le funzionalità richieste. Ogni modulo è composto da una schermata principale e da una serie di finestre per le impostazioni dei parametri di sintesi. E' prevista una voce relativa alle directory predefinite, mediante la quale impostare la directory del simulatore, quella del database dei modelli di timbro e quella dei suoni sintetizzati; le altre voci consentono di effettuare tutte le operazioni descritte nel paragrafo precedente.

5.2.1 Analisi

E' il sottosistema che permette di estrarre il modello di timbro da un campione sonoro emesso da uno strumento reale, secondo le specifiche del paragrafo 3.1 è stato implementato nel modo che andremo a descrivere. Per la gestione dei file in formato wave è stata utilizzata una libreria freeware che incapsula una classe per il trattamento di tali tipi di file [Cro99]. Per la presentazione dei risultati di analisi è stato usato il *TeeChart*, un componente presente nell'ambiente di sviluppo che permette la rappresentazione di grafici in modo estremamente semplice:

- in fase di progettazione vengono associate al componente i tipi di serie che devono essere presenti sul grafico;
- in fase di esecuzione viene attivata la serie che deve essere rappresentata e caricata con i valori relativi,

permettendo di ottenere grafici di diverso tipo in base alla natura dei dati da presentare, questo offre una migliore comprensione dei risultati di analisi.

Al suo avvio la procedura presenta una schermata iniziale per la scelta del file wave da analizzare e per impostare il numero di armoniche e di breakpoints da includere nel modello in esame;

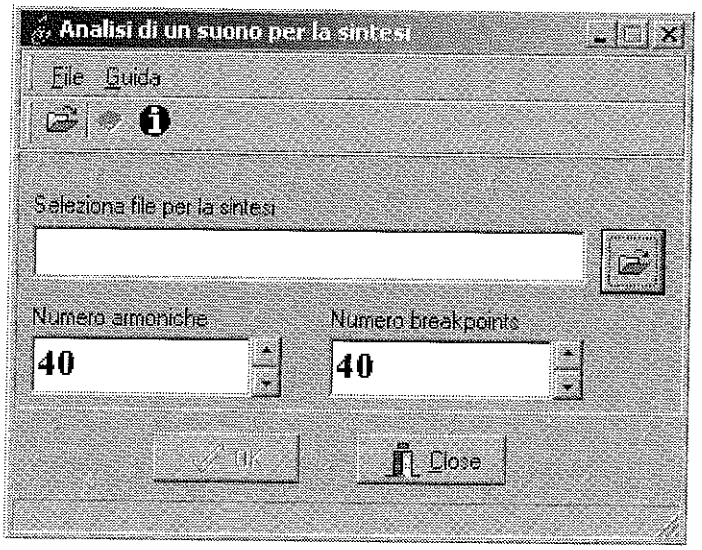

Figura 5-6 Schermata iniziale procedura di analisi

scelto il file vengono letti e caricati in un array i campioni relativi; si presenta così la schermata principale che in questa fase presenta il grafico del file da analizzare, ottenuto plottando i campioni del file da analizzare precedentemente caricati in memoria.

L'analisi procede applicando l'algoritmo della FFT su 32768 campioni del file (per avere una potenza di due e poter applicare l'algoritmo della FFT), partendo da una posizione scelta dall'utente dando così la possibilità, eventualmente, di tralasciare la parte iniziale corrispondente alla fase di attacco che come accennato è ricca di componenti anche non armoniche che si estinguono rapidamente; se la lunghezza del campione è minore di 32768 vengono aggiunti degli zeri per completare la finestra di analisi.

Al termine della procedura si ottiene lo spettro di ampiezza, che viene disegnato sul grafico; si presenta così un grafico che contiene delle righe verticale che rappresentano le componenti spettrali del suono analizzato, in base alla procedura di analisi viene proposta come frequenza fondamentale la *prominte frequency*, se dall'analisi del grafico risulta che la fondamentale è un'altra basta fare click col mouse sulla riga spettrale desiderata. Se le righe spettrali non risultano equispaziate fra loro significa che le armoniche successive non sono multiple di una fondamentale. In tal caso bisogna impostare a mano il vettore dei rapporti con la "fondamentale" facendo click col mouse sulle righe dello spettro; viene calcolato così tale rapporto e inserito nell'array che verrà salvato nel modello di timbro.

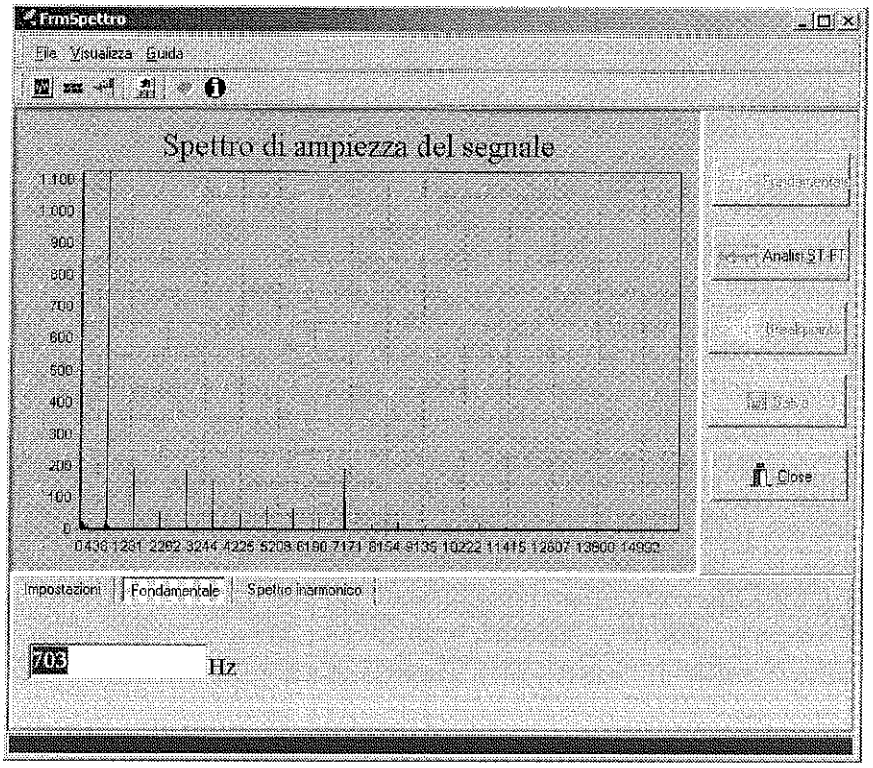

Figura 5-7 procedura di analisi – spettro risultante

Terminata la ricerca della fondamentale si può iniziare l'analisi STFT facendo click sull'apposito pulsante; la prima operazione è il calcolo della dimensione (in frame) della finestra di analisi, che nel caso di suono armonico è data dalla seguente formula:

$$Lungh_{Fin} = \frac{2 \cdot 44100}{Fondamentale} + 1 \qquad [5.3]$$

mentre nel caso di suono inarmonico viene scelta di lunghezza opportuna (da prove effettuate si è deciso di porre in questo caso tale lunghezza a 1024 frames). Il punto nevralgico di questa procedura è l'analisi di Fourier che deve essere condotta sulle varie finestre, raramente tale lunghezza sarà una potenza di due per poter applicare l'algoritmo della FFT, bisogna allora ricorrere alla DFT con conseguente aumento del tempo di calcolo (come nel caso di frequenza fondamentale molto bassa, dell'ordine di qualche decina di hertz). Viene poi calcolato il vettore delle energie delle armoniche e rappresentato mediante un grafico a barre. Successivamente viene applicato l'algoritmo per la ricerca dei breakpoints, che mira ad ottenere una riduzione dei dati di codifica e al termine della quale possiamo salvare il modello di timbro ottenuto; mediamente la riduzione ottenuta è del rapporto 10:1.

5.2.2 Modellazione e modifica

Mediante gli appositi sottomoduli possiamo modificare i modelli di timbro creati col modulo precedente o crearne di nuovi secondo le esigenze dell'utente; le differenze fra tali sottomoduli sono minime, nel senso che tutte le procedure utilizzate in fase di creazione possono essere riutilizzate in fase di modifica di un modello esistente, infatti in questo caso si fa uso della stessa interfaccia utente salvo una diversità in fase di caricamento: infatti nel caso di creazione di un modello di timbro virtuale devono essere impostati alcuni parametri fondamentali del modello, avremo quindi alcune finestre di dialogo che consentono di specificare *fondamentale*, *durata*, tipi di inviluppo, contenuto armonico e *breakpoints*;

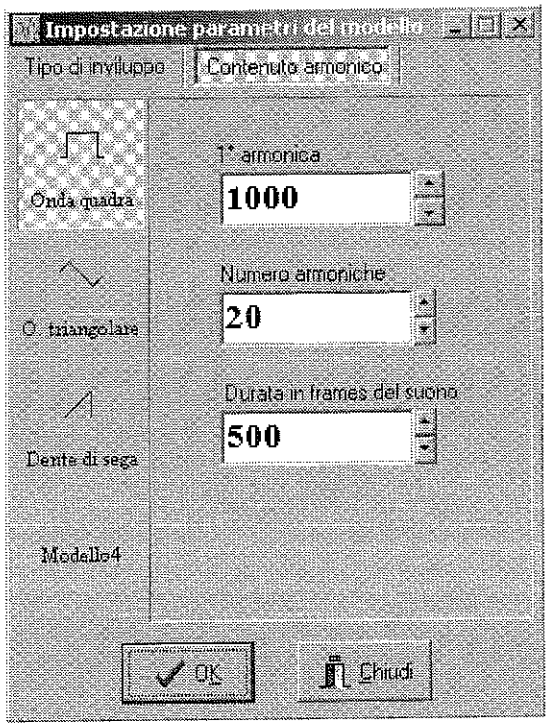

Figura 5-8 Schermata iniziale procedura di creazione modello virtuale

nel caso invece di modifica di un modello creato precedentemente è sufficiente caricare tale modello dal database.
A questo punto le due interfacce convergono verso un modello unico che permette di apportare le modifiche ai parametri predefiniti o al modello caricato. L'interfaccia è strutturata con un grafico in alto a sinistra che mostra gli inviluppi delle varie armoniche e in basso i controlli per la modifica dei parametri degli inviluppi, le tre caselle di testo subito sotto il grafico mostrano l'armonica il breakpoint corrente e la posizione intesa come numero del frame in cui è stato impostato.

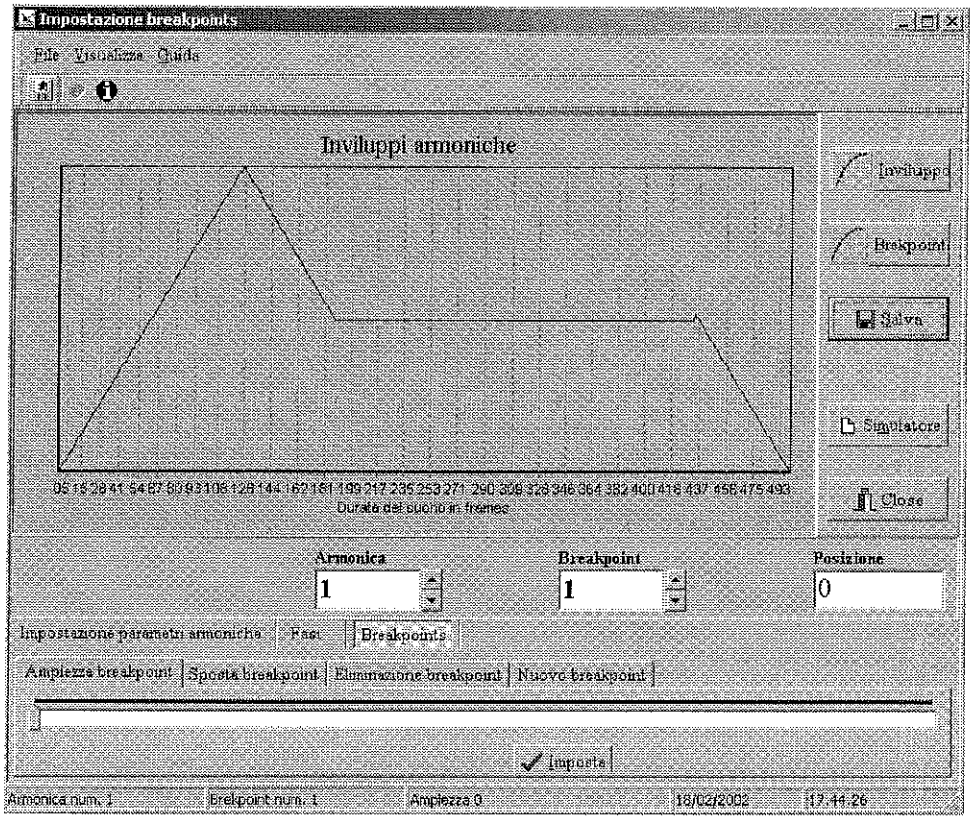

Figura 5-9 Creazione modello virtuale

Al caricamento di tale schermata viene mostrata sul grafico la serie relativa all'inviluppo della prima armonica, l'utente può intervenire sui parametri delle armoniche variando il rapporto delle stesse con la fondamentale, moltiplicando le ampiezze delle parziali per un determinato fattore, impostando eventualmente le fasi di sostegno e di rilascio del suono, intervenendo sui singoli breakpoints variandone l'ampiezza, spostando o eliminandone alcuni o inserendone di nuovi.

Per quanto riguarda la gestione delle armoniche e delle relative fasi è sufficiente inserire i valori nelle caselle di testo opportune; per variare l'ampiezza di un breakpoint bisogna intervenire sulla trackbar delle ampiezze e poi premere il pulsante "*ok*" per rendere effettiva la variazione; per spostare un breakpoint è sufficiente digitare la nuova posizione e premere il pulsante sposta; per eliminare il breakpoint corrente è sufficiente premere il pulsante elimina; per inserirne di nuovi il pulsante *"nuovo"* stabilisce il numero d'ordine del breakpoint inserito, basterà poi digitare la posizione dove inserirlo e premere il pulsante inserisci.

5.2.3 Sintesi

E' il sottomodulo mediante il quale vengono inviati i parametri per la sintesi al controllore del chip, per motivi di tempo non è stato realizzato nella sua interezza, ma limitatamente alla sezione relativa alla sintesi di un singolo suono al fine di verificare la correttezza del sistema implementato. All'avvio la procedura carica un modello di timbro selezionato dall'utente, sarà quindi possibile

impostare alcuni parametri di sintesi e passare il file al simulatore. Terminato il lavoro del simulatore si può esportare il file creato in formato wave e ascoltare il suono così sintetizzato; la figura seguente mostra una schermata tipica di tale procedura.

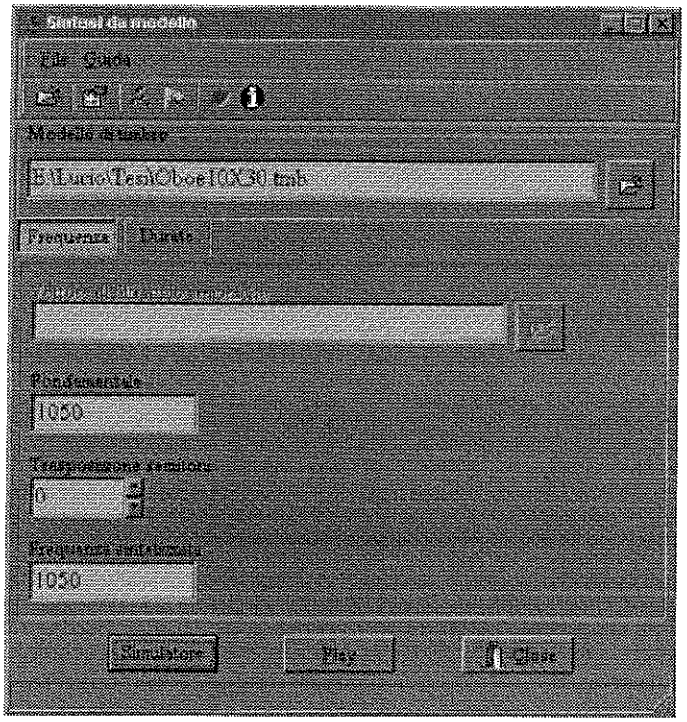

Figura 5-10 Schermata procedura di sintesi

Secondo le specifiche del modulo, una volta caricato il modello e impostati i parametri per la sintesi, viene scandito il modello di timbro lungo tutti i frames, quando ad un frame corrisponde un breakpoint vengono impostati i dati 'per la creazione del file da inviare al simulatore.

Per esportare il file prodotto dal simulatore in formato wave è stata usata la classe per la gestione dei file wave già usata per la parte di analisi [Cro99], leggendo dal file ed effettuando la conversione di formato del dato letto; il file viene letto due volte di seguito per effettuarne la normalizzazione, in modo da poter effettuare un corretto confronto con il file originale; per la normalizzazione di un file wave occorre portare il campione con il valore più alto al valore di 32767 e scalare proporzionalmente gli altri.

In questa fase possono essere impostati alcuni parametri per ottenere degli effetti particolari sul suono di uscita, in particolare sono stati presi in considerazione i due esempi sottostanti:

- la trasposizione dei semitoni per variare così la frequenza della fondamentale (e di conseguenza di tutte le altre), secondo la seguente relazione:

$$Fond \cdot \left(\sqrt[12]{2}\right)^{n} \qquad [5.4]$$

dove n è il livello di trasposizione impostato (n=12 corrisponde ad aumentare di un ottava, n=-12 a diminuire di un ottava);

- la durata delle fasi del suono per ottenere degli effetti di stretching temporale di tutto il suono o parte di esso;

per quanto riguarda il primo punto è sufficiente una semplice moltiplicazione della fondamentale per il rapporto scelto, per il secondo è necessario ricalcolare l'inviluppo del suono in base al rapporto di stretching e alla parte di suono coinvolto.

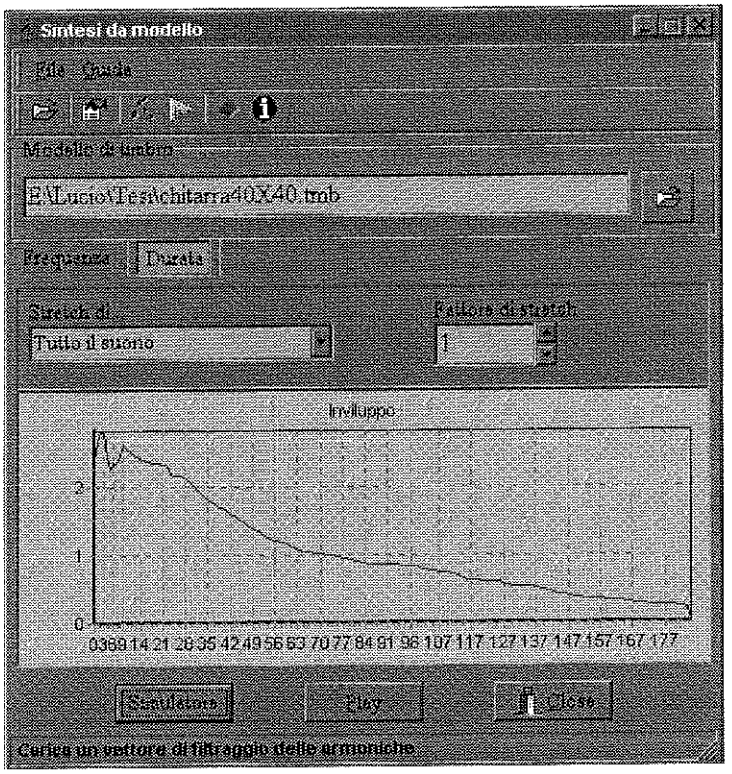

Figura 5-11 Schermata procedura di sintesi visualizzante gli effetti applicabili al suono

Il numero dei breakpoints rimane invariato, ma la loro posizione viene riportata, mediante il fattore di scala, sul nuovo inviluppo; per fornire una visualizzazione grafica della modifica effettuata viene calcolato il nuovo inviluppo e disegnato nel grafico sottostante; ad esempio se si raddoppia la durata del suono la posizione dei breakpoints viene moltiplicata per 2; se invece viene variata solo la fase di attacco i breakpoints che non appartengono a tale fase verranno rimappati così come sono nel vettore originario, a partire dalla nuova posizione della fase di attacco; la seguente tabella mostra un frammento di inviluppo (sulla prima riga è riportato il numero d'ordine del breakpoints, sulla seconda la sua posizione e sulla terza la nuova posizione), è stata effettuato un raddoppio della durata della fase di attacco che inizia col breakpoints 24:

...	20	21	22	23	24	25	26	27	28	...
...	36	40	45	47	50	55	58	63	67	...
...	72	80	90	94	100	105	108	113	117	...

Tabella 4.1 Esempio di spostamento dei breakpoints

Una volta scelta la particolare modifica da apportare al suono, viene richiamata la procedura apposita che calcola il nuovo vettore dei breakpoints per quel caso specifico e di seguito viene calcolato l'inviluppo:

6 Testing del software e sperimentazione

Per la verifica della correttezza dei risultati ottenuti dal sistema software sviluppato sono state effettuate delle prove su un serie di campioni sonori emessi da diversi strumenti musicali, registrati appositamente in camera semianecoica e campionati a 44100 *Hz*, 16 bit mono, al fine di valutare il grado di somiglianza fra il suono originale e il suono sintetizzato dal sistema. Sono state provate tutte le funzionalità del sistema e per la valutazione qualitativa sono state effettuate prove di ascolto dei risultati, in questa sede vengono riportati grafici riguardanti gli spettri e gli andamenti temporali dei segnali originali e sintetizzati, fermo restando che ciò non costituisce una prova di bontà del sistema.

6.1 Verifica dei risultati

La verifica dei risultati è stata fatta eseguendo prove di sintesi dei suoni precedentemente registrati e ascoltando i suoni sintetizzati; in questa sede per dare un'idea dei risultati ottenuti verranno mostrati gli inviluppi e gli spettri dei suoni esaminati, e riportato il giudizio relativo alla qualità riscontrata Verrà mostrato un esempio per ogni casistica esaminata

6.1.1 Effetti variazione parametri del modello

In questo paragrafo viene messa in luce l'effetto del numero delle armoniche e dei breakpoints sul suono sintetizzato, al fine di mostrare l'incidenza di tali parametri sia dal punto di vista della qualità del suono sintetizzato che dal punto di vista di occupazione di memoria del modello di timbro risultante; facciamo riferimento a un *do* di **oboe** con livello espressivo forte (*f*), il valore della frequenza fondamentale risulta in questo caso di 523 *Hz*; di seguito vengono mostrati l'andamento dello spettro e del segnale nel tempo del suono originale, e successivamente vengono mostrati i grafici relativi ai suoni sintetizzati; dal grafico dello spettro del segnale possiamo notare come il suono in questione sia povero di armoniche, oltre la 15-esima armonica sono tutte 40*db* sotto l'ampiezza più alta, questo fa sì che in fase di resintesi sia possibile ottenere buoni risultati senza usare un numero eccessivo di armoniche. Per suoni più complessi e con frequenze fondamentali più basse, come ad esempio quello di figura 6.4 che rappresenta lo spettro di un *do* di **fagotto**, occorre invece usare un numero maggiore di armoniche (aumento della densità spettrale) per ottenere un suono di buona qualità. Di seguito vengono riportati i grafici relativi alle prove effettuate per mettere in luce gli effetti del numero delle armoniche e dei breakpoints sul risultato di sintesi ottenuto; tali prove sono state effettuate in una prima fase variando il numero delle armoniche e tenendo fisso quelli dei breakpoints, nel

caso successivo variando il numero dei breakpoints e tenendo fisso il numero delle armoniche, i risultati sono mostrati nelle figure [6.5] – [6.28] che mostrano gli spettri e gli inviluppi relativi alle varie prove di sintesi. Nelle figure [6.1] – [6.4] vengono mostrati gli andamenti originali degli spettri e degli inviluppi.

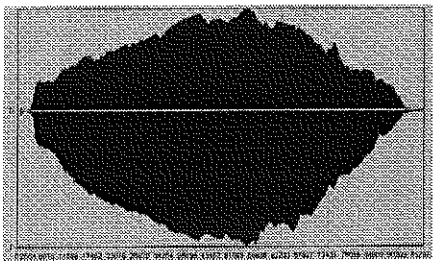

Figura 6-1 do di Oboe – Andamento temporale

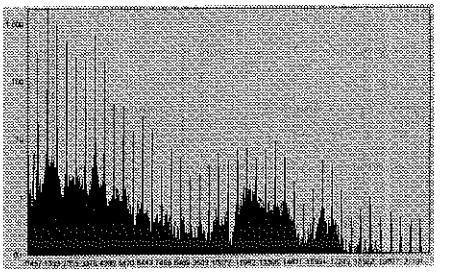

Figura 6-2 do di Oboe – Spettro

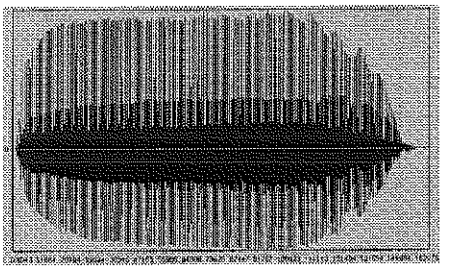

Figura 6-3 do di Fagotto – Andamento temporale

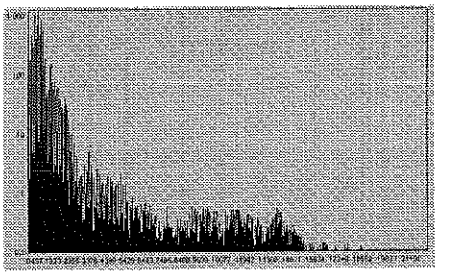

Figura 6-4 do di Fagotto – Spettro

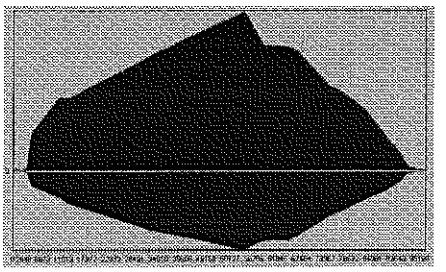

Figura 6-5 do di oboe sintetizzato con 10 armoniche e 40 breakpoints – andamento temporale

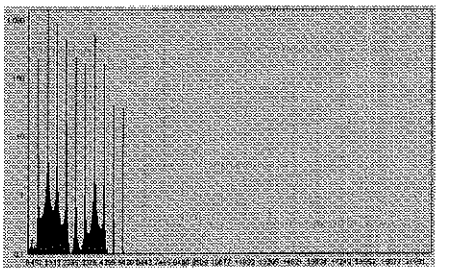

Figura 6-6 do di oboe sintetizzato con 10 armoniche e 40 breakpoints – spettro

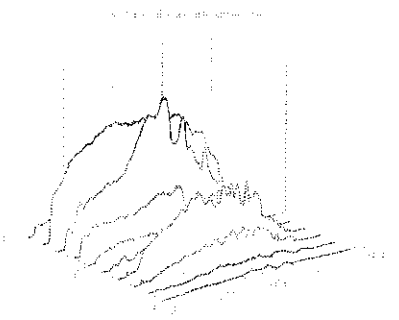

Figura 6-7 di Oboe – inviluppi originali delle armoniche

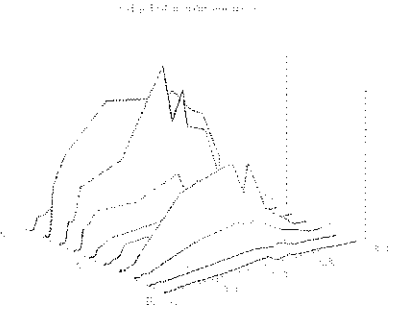

Figura 6-8 do di Oboe sintetizzato con 10 armoniche e 40 breakpoints – inviluppi ridotti delle armoniche

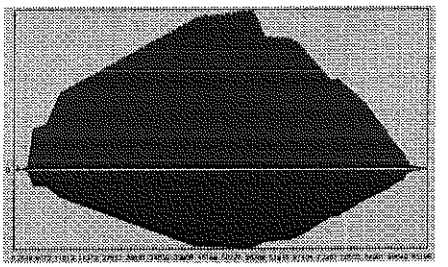

Figura 6-9 do di oboe sintetizzato con 40 armoniche e 40 breakpoints – andamento temporale

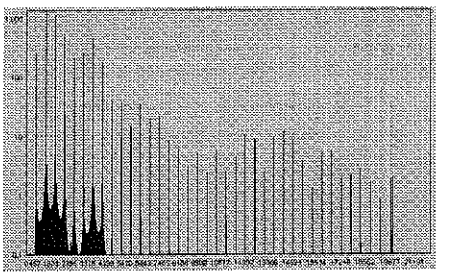

Figura 6-10 do di oboe sintetizzato con 40 armoniche e 40 breakpoints – spettro

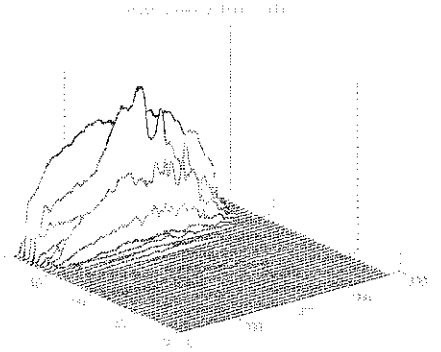

Figura 6-11 do di oboe sintetizzato con 40 armoniche e 40 breakpoints – inviluppi originali delle armoniche

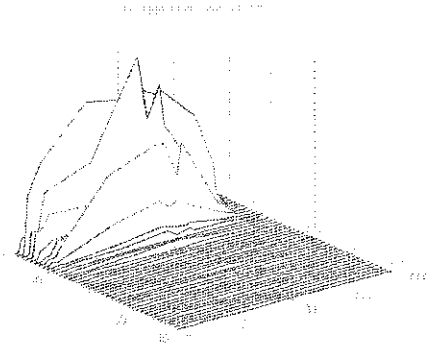

Figura 6-12 do di oboe sintetizzato con 40 armoniche e 40 breakpoints – inviluppi ridotti delle armoniche

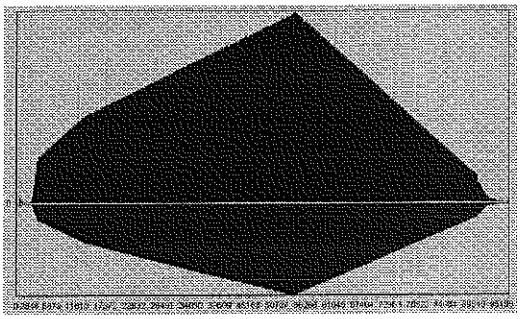

Figura 6-13 do di Oboe sintetizzato con 20 armoniche e 20 breakpoints – andamento temporale

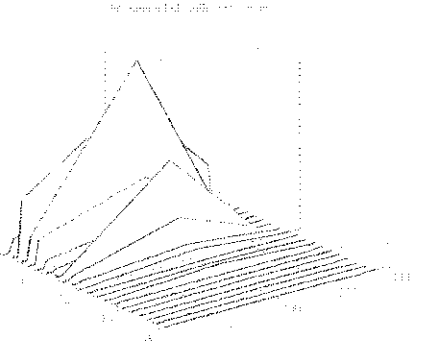

Figura 6-14 do di Oboe sintetizzato con 20 armoniche e 20 breakpoints – inviluppi ridotti delle armoniche

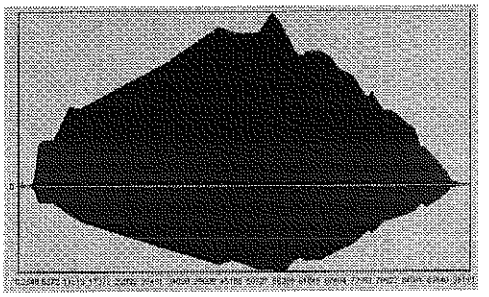

Figura 6-15 do di oboe sintetizzato con 20 armoniche e 80 breakpoints – andamento nel tempo

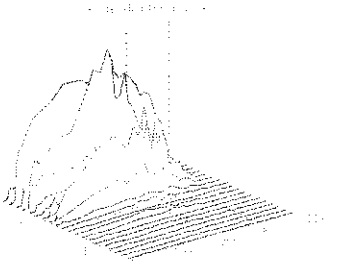

Figura 6-16 do di Oboe sintetizzato con 20 armoniche e 80 breakpoints – inviluppi ridotti delle armoniche

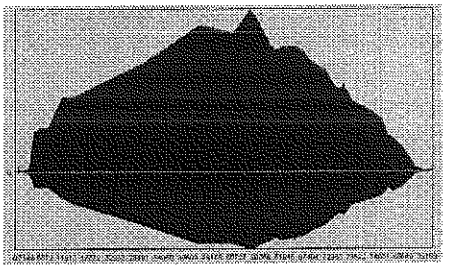

Figura 6-17 do di Oboe sintetizzato con 40 armoniche e 80 breakpoints – andamento temporale

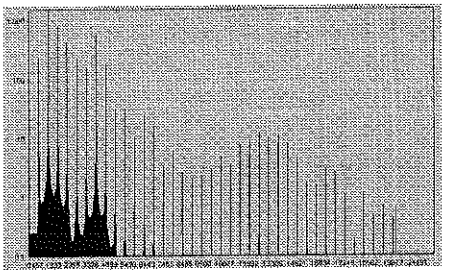

Figura 6-18 do di Oboe sintetizzato con 40 armoniche e 80 breakpoints spettro

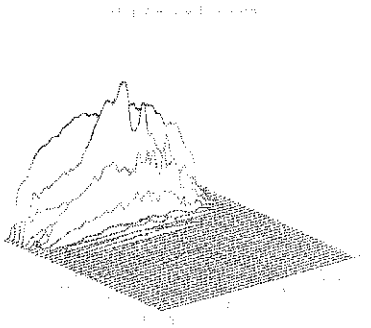

Figura 6-19 do di Oboe sintetizzato con 40 armoniche e 80 breakpoints – inviluppi originali armoniche

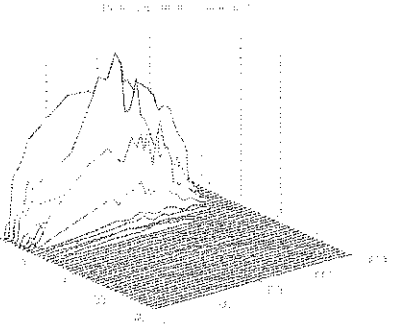

Figura 6-20 do di Oboe sintetizzato con 40 armoniche e 80 breakpoints – inviluppi ridotti armoniche

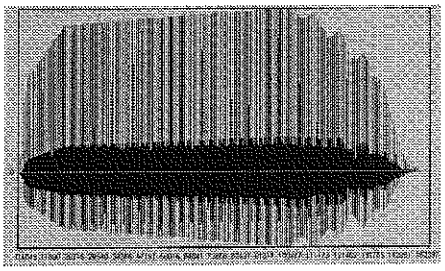

Figura 6-21 do di fagotto sintetizzato con 20 armoniche e 60 breakpoints – andamento temporale

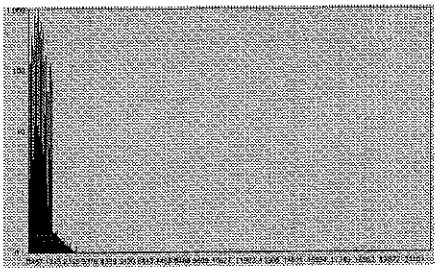

Figura 6-22 do di fagotto sintetizzato con 20 armoniche e 60 breakpoints - spettro

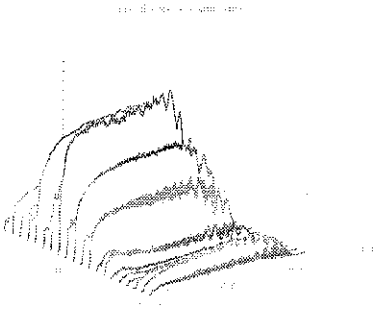

Figura 6-23 do di fagotto sintetizzato con 20 armoniche e 60 breakpoints – inviluppi originali armoniche

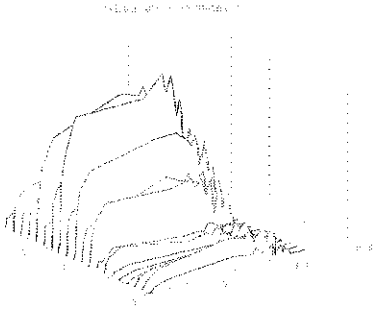

Figura 6-24 do di fagotto sintetizzato con 20 armoniche e 60 breakpoints – inviluppi ridotti armoniche

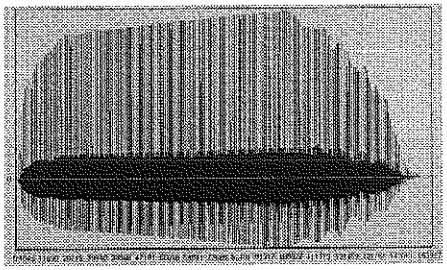

Figura 6-25 do di fagotto sintetizzato con 80 armoniche e 80 breakpoints – andamento temporale

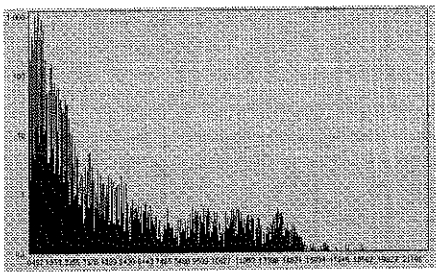

Figura 6-26 do di Fagotto sintetizzato con 80 armoniche e 80 breakpoints - spettro

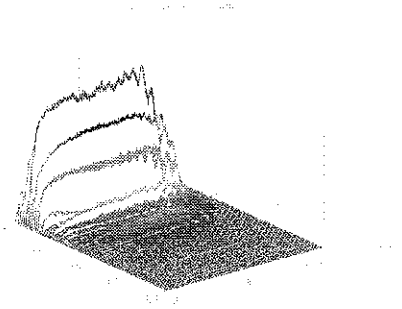

Figura 6-27 do di Fagotto sintetizzato con 80 armoniche e 80 breakpoints – inviluppi originali armoniche

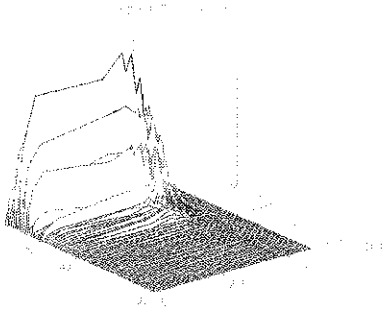

Figura 6-28 do di Fagotto sintetizzato con 80 armoniche e 80 breakpoints – inviluppi ridotti armoniche

6.1.2 Variazioni applicate in fase di sintesi

Di seguito verranno eseguite prove applicando degli effetti al suono sintetizzato per ottenere così sonorità diverse da un unico modello di timbro, in particolare verrà messa evidenziata la possibilità di effettuare una trasposizione della frequenza fondamentale e la possibilità di effettuare uno stretching temporale del suono o parte di esso; come riferimento viene preso un modello di timbro creato a partire da una registrazione della voce umana, le figure seguenti mostrano l'andamento temporale e lo spettro del suono in questione:

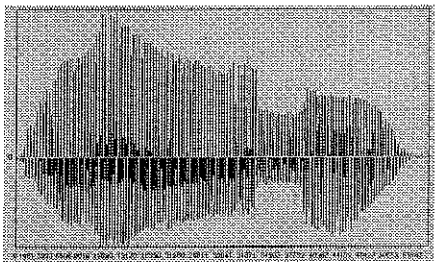

Figura 6-29 Andamento temporale del suono analizzato

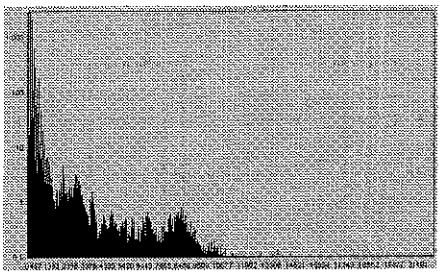

Figura 6-30 Spettro del suono analizzato

Effetti della variazione della frequenza: viene evidenziata la possibilità di effettuare una trasposizione dei semitoni ed effettuare una variazione della frequenza fondamentale del suono di origine, a titolo di esempio viene effettuato la trasposizione della frequenza di un ottava sopra il suono originale; lo spettro del segnale risultante è il seguente:

Effetti di stretching temporale: viene mostrato l'effetto di una variazione della durata del suono: la sua durata viene raddoppiata, l'andamento del suono risultante è mostrato nella figura seguente:

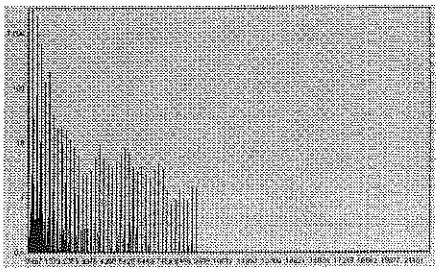

Figura 6-31 Spettro del suono ottenuto applicando la trasposizione di frequenza di un ottava

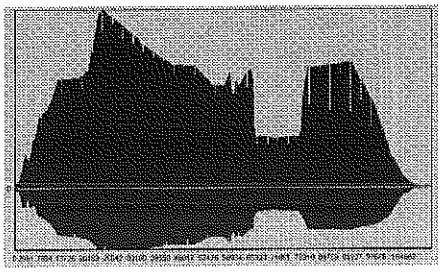

Figura 6-32 Andamento temporale del suono di lunghezza raddoppiata

6.1.3 Prove di sintesi di suoni non armonici

Sono state effettuate prove di sintesi di suoni non armonici, per testare la nuova funzionalità, aggiunta in questo lavoro, alla procedura di analisi; come esempio viene preso un suono di un campanella; lo spettro del segnale è mostrato in figura 6.33, possiamo notare come le componenti non siano tutte equispaziate, bisogna quindi condurre un analisi di tipo manuale selezionando le parziali da includere nel suono sintetizzato.

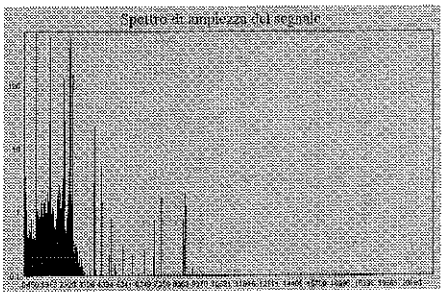

Figura 6-33 Spettro originale del suono analizzato

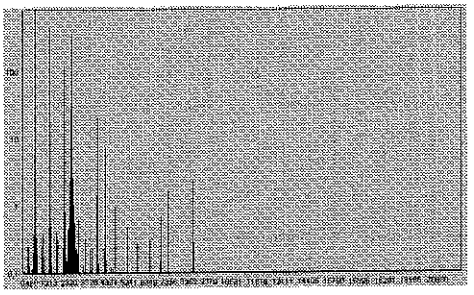

Figura 6-34 Spettro del suono sintetizzato

6.1.4 Variazione caratteristiche del suono

In questo paragrafo viene testata la possibilità di apportare modifiche ad un modello di timbro creato in precedenza, a titolo di esempio facciamo riferimento al suono della voce, vengono variati i rapporti delle prime armoniche, le più importanti, con la fondamentale per renderlo non armonico; lo spettro risultante è mostrato in figura 6.35 e può essere confrontato con lo spettro di figura 6.30

6.1.5 Creazione di un modello di timbro virtuale

In questo paragrafo viene testata la procedura che consente la creazione di modelli di timbro virtuali, dopo aver scelto alcuni parametri caratteristici del modello di timbro che vogliamo creare, possiamo agire sull'inviluppo delle varie armoniche per variarne i breakpoints e le ampiezze al fine di ottenere l'andamento desiderato. E' stato creato un semplice modello basato su uno degli inviluppi proposti, con 10 armoniche e 5 breakpoints e con frequenza fondamentale di $1000Hz$, l'andamento delle armoniche è mostrato nel grafico di figura 6.36

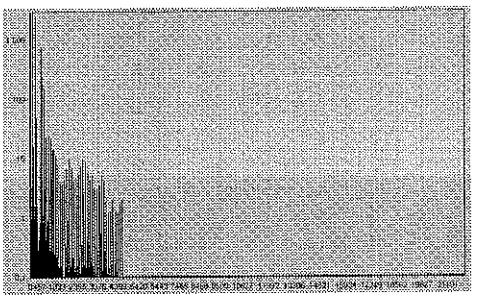

Figura 6-35 Spettro del suono sintetizzato con i rapporti delle prime armoniche variati

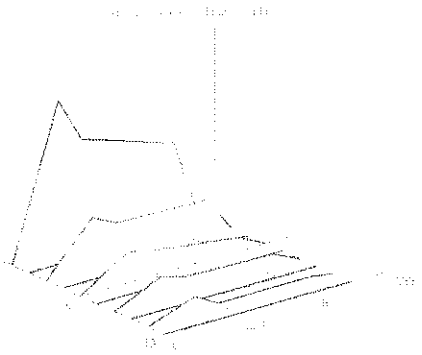

Figura 6-36 Inviluppi delle armoniche relative al modello creato

6.2 Valutazione delle prove effettuate

In questo paragrafo viene riportata una valutazione delle prove d'ascolto effettuate relativamente agli esempi riportati: i test sono stati condotti usando un sistema composto da: una scheda audio *Turtle Beach Pinnacle*, diffusori di riferimento *Yamaha NS10*, amplificatore *NAD3225;* i risultati per le varie categorie di suoni sono riportati di seguito:

Campanoide

- Sintesi con 6 armoniche e 100 breakpoints: possiamo riscontrare che il suono e' troppo povero di frequenze acute
- Sintesi con 13 armoniche e 80 breakpoints: possiamo notare che il suono e' abbastanza brillante, ma il timbro risulta statico rispetto all'originale
- Sintesi con 13 armoniche e 100 breakpoints: sebbene non sia ricco di parziali come l'originale il suono e' abbastanza brillante e dinamico, con una buona simulazione dei battimenti fra le parziali

Fagotto

- Sintesi con 20 armoniche e 20 breakpoints: nonostante l'inviluppo segua l'originale in modo accettabile, il suono sintetizzato manca totalmente delle parziali acute.
- Sintesi con 80 armoniche e 80 breakpoints: il suono sembra abbastanza realistico, sebbene sarebbe migliorabile con più armoniche.

Oboe

- Sintesi con 10 armoniche e 40 breakpoints: l'inviluppo segue l'originale in modo accettabile, ma il suono sintetizzato manca totalmente di acuti.
- Sintesi con 20 armoniche e 20 breakpoints: parziali acute più presenti, ma il suono è troppo statico.
- Sintesi con 20 armoniche e 80 breakpoints: buona qualità complessiva. La fase di attacco (più densa di energia sugli acuti) non è resa in modo ottimale.
- Sintesi con 40 armoniche e 40 breakpoints: il suono è leggermente più statico del precedente, ma è più brillante nella fase di attacco.
- Sintesi con 40 armoniche e 80 breakpoints: il numero dei breakpoints è tale da assicurare una buona qualità complessiva del suono, il numero di armoniche permette una resa più realistica, anche della fase di attacco che risulta abbastanza brillante.

Voce

- Sintesi con 40 armoniche e 80 breakpoints: abbastanza articolato sulle ampiezze, ma povero di componenti a frequenza alta. Considerate le intrinseche difficoltà nella perfetta resintesi della voce con questa tecnica, il risultato è comunque accettabile.
- Sintesi a 1 Ottava Sopra la frequenza fondamentale: il risultato è buono, nonostante non sia ancora stata implementata una tecnica di trasposizione capace di mantenere inalterate le zone formantiche ai vari registri.
- Sintesi a 1 Ottava sopra la frequenza fondamentale e a lunghezza doppia: lo stretching del suono precedente è qui reso in modo perfetto, cosa

impossibile con l'uso di tecniche analoghe che lavorano nel dominio del tempo.

Alla luce delle precedenti considerazioni possiamo affermare che i risultati ottenuti dal sistema sono stati più che soddisfacenti, ferme restando le difficoltà intrinseche, in parte superate, di suoni inarmonici e caratterizzati da un contenuto frequenziale molto ricco, e quelle dovute a suoni caratterizzati da una forte componente di rumore nella fase di attacco; a questo scopo sono allo studio metodologie per la simulazione del rumore di attacco mediante una veloce modulazione della frequenza delle parziali durante la fase di attacco.

Per quanto riguarda alcune considerazioni sulla scelta del numero delle armoniche e dei breakpoints possiamo affermare che per suoni caratterizzati da un inviluppo di ampiezza molto complesso è necessario scegliere un numero di breakpoints sufficientemente elevato, mentre per suoni più statici posiamo raggiungere buoni risultati con un numero minore di breakpoints. Per quanto riguarda la scelta del numero delle armoniche, e limitandoci al caso di suoni armonici, possiamo affermare che se il suono ha una frequenza fondamentale abbastanza alta (come ad esempio l'oboe) sono sufficienti un basso numero di armoniche, mentre per suoni caratterizzati da un registro grave (ad esempio il fagotto) è necessario includere nel modello un numero maggiore di armoniche.

Un'altra considerazione che possiamo fare in questa sede riguarda la dimensione dei file ottenuti, per valutare a grandi linee lo spazio occupato su disco e quindi le dimensioni di un database abbastanza completo. In questo senso è stato notato che le dimensioni dei file sono direttamente proporzionali al prodotto *numero armoniche x numero breakpoints*, dovuto al fatto che la matrice dei valori dei breakpoints per ogni armonica è di dimensione ***N_Arm X N_Break***). Comunque alla luce dei risultati, le dimensioni dei modelli risultanti non costituiscono un problema, nemmeno per archivi di grosse dimensioni (qualche migliaio di modelli di timbro); infatti per modelli di scarsa qualità (poche armoniche e pochi breakpoints) le dimensioni sono dell'ordine di qualche Kbyte (ad esempio il modello del *do* oboe sintetizzato con 10 armoniche e 40 breakpoints ha una dimensione di 8,35 Kbyte, mentre il *do* di fagotto sintetizzato con 20 armoniche e 20 breakpoints ha una dimensione di 7,86 Kbyte), mentre per modelli più complessi la dimensione è dell'ordine delle decine di Kbyte e per modelli molto complessi si può arrivare a dimensioni che superano i 100 Kbyte (ad esempio il modello del *do* di oboe sintetizzato con 40 armoniche e 80 breakpoints risulta avere una dimensione di 66,9 Kbyte, mentre il modello del *do* di fagotto sintetizzato con 80 armoniche e 80 breakpoints risulta avere una dimensione di 130 Kbyte). Alla luce delle precedenti considerazioni un archivio di un migliaio di modelli di timbro di buona qualità arriverà ad occupare uno spazio su disco di qualche centinaio di Mbyte, che con i costi odierni delle memorie di massa risulta più che accettabile; anche per quanto riguarda l'occupazione in termini di RAM durante una composizione polifonica, il problema non sussiste in quanto il numero dei modelli coinvolti è limitato, nell'ordine delle decine, quindi l'occupazione di RAM è estremamente limitata.

7 Conclusioni e sviluppi futuri

In questo lavoro è stato affrontato lo sviluppo di un ambiente software per la gestione dei modelli di timbro con la sintesi audio additiva, permettendo il controllo dei parametri e l'invio degli stessi ad un eventuale sistema hardware esterno ad un PC. In particolare è stata messa a punto l'interfaccia grafica per fornire un agevole supporto per l'utente, sono state rielaborate le procedure di analisi di suoni, sviluppate per l'ambiente Matlab in precedenti lavori, e ne è stato fatto il porting nel linguaggio C++, apportando anche delle modifiche al fine di ampliare la casistica di suoni analizzabili. In tal senso è stata presa in considerazione la modellazione di suoni non armonici e inoltre sono state sviluppate procedure per la modifica di modelli già archiviati e per la creazione di modelli virtuali. Infine sono stati realizzati tools per la realizzazione di alcuni effetti sonori in fase di resintesi quali: trasposizione dei semitoni e stretching del suono o parte di esso ecc.. I risultati ottenuti sono interessanti lasciando aperta la strada a ulteriori sviluppi per particolari classi di suoni ed effetti, come elencato di seguito:

- *Riduzione del numero delle armoniche*: dalle prove effettuate possiamo notare che per una buona parte di suoni è possibile una resintesi di buona qualità con un numero non eccessivo di parziali (20 - 30). Tuttavia si è anche sperimentato che per ottenere una qualità "professionale", prossima al segnale originale, è necessario impiegare molte più armoniche (80 - 100). Si noti infatti che iniziando a filtrare un segnale audio musicale appena al di sotto di 15 Khz è possibile avvertire subito una mancanza di brillantezza del segnale stesso. Considerando che una buona parte dei segnali musicali ha una "fondamentale" intorno a 200 o anche 150 Hz è facile dedurre che (anche limitandosi agli spettri armonici) se si vogliono riprodurre tutte le armoniche possibili del segnale il loro numero supera facilmente il centinaio. Pur essendo il sistema di sintesi in fase di realizzazione molto potente, nel caso di "partiture" molto complesse e ricche non è escluso si giunga ad un esaurimento delle capacità del sistema stesso. Si rende quindi necessario uno studio più approfondito delle caratteristiche spettrali dei suoni da generare, cercando di operare un ulteriore selezione delle parziali risultanti dalla procedura di analisi. Quest'ultima infatti non tiene conto di eventuali fenomeni percettivi (ad es. il mascheramento), che potrebbero rendere superflua la generazione di un buon numero di parziali.

- *Gestione spettri inarmonici*: sebbene tale aspetto sia stato preso in considerazione con buoni risultati, bisogna tener conto che si tratta di una procedura semiautomatica e che per la scelta delle varie parziali e la determinazione dei rapporti con la fondamentale prevede l'intervento dell'utente; nel caso di suoni caratterizzati da poche componenti spettrali ciò non comporta grosse difficoltà, ma per suoni caratterizzati da un ampio contenuto spettrale la scelta delle frequenze da includere nel modello può risultare pesante. In questo senso potrebbe essere sviluppata una opportuna tecnica per l'individuazione automatica delle varie righe significative dello spettro.

- *Interpolazione e cross-fading spettrale:* si tratta di una tecnica adottata in via sperimentale ma con discreti risultati che consente di gestire con continuità transizioni fra diverse modalità espressive dello strumento modellato.

- *Modellazione del rumore di attacco:* utile per la simulazione del rumore di attacco, presente in molti tipi di suoni; teoricamente sarebbe possibile creare

fasce di rumore sovrapponendo un gran numero di parziali, ma ciò limita fortemente il numero di voci complessive generabili. In questo senso sono allo studio tecniche di modulazione veloce durante la fase di attacco delle parziali, ancora in via sperimentale ma con discreti risultati.

- *Pitch e detuning:* si è notato che una caratteristica del suono resintetizzato (che lo fa riconoscere artificiale) è il pitch della nota generata costante, a differenza dell'originale che si sposta irregolarmente intorno ad una frequenza media. E' stata sperimentata una semplice tecnica di variazione del pitch della fondamentale ad ogni breakpoint, in maniera pseudocasuale e con una deviazione massima dell'1 o 2 %. Il suono generato risulta più gradevole e (nel caso di più note) mai uguale a se stesso. Un'altra interessante tecnica sperimentata è il detuning di frequenza (nell'ordine dello 0.2 %) delle singole parziali. Si genera una sorta di lieve chorusing spettrale che rende il suono generato molto più vivo. Questa tecnica potrà essere implementata in maniera molto semplice se il controller prevede di ricevere anche la costante per le variazioni non lineari della frequenza delle parziali in funzione dell'ampiezza.

- *Trasformazione da eventi a comandi verso il sistema di sintesi:* nelle prove sperimentali descritte in questo documento per ogni strumento si è generata una sola nota basata sul modello timbrico costruito, allo scopo di condurre prove comparative con il segnale originale. Questo è stato fatto generando gli opportuni comandi accettati dal simulatore, conformi a quella che dovrà essere l'interfaccia verso il sistema *controllore-chip di sintesi.* Per procedere nella sperimentazione è necessario quindi sviluppare un modulo che a partire da un file che descrive gli eventi musicali ad un livello macroscopico (e.g. uno standard MIDI file), costruendo un file di *comandi* adatto ad essere eseguito dal simulatore. In questo modo è possibile testare impasti sonori di più note o modelli timbrici simultanei, e verificare anche possibili colli di bottiglia della generazione polifonica in casi realistici. Sarebbe inoltre interessante uno studio circa tecniche per il controllo dello spettro delle note generate durante la loro esecuzione. E' necessario definire quindi un set di parziali le cui caratteristiche sono variabili con continuità durante l'esecuzione, stabilendo anche le modalità del controllo in maniera da trovare un compromesso tra l'espressività dello stesso e l'inevitabile incremento della banda verso il sistema di sintesi. Questi controlli sono paragonabili agli analoghi *pitch-bend* o *control change* dello standard MIDI. Quello che infatti può rendere il sistema interessante, anche in prospettiva di una sua utilizzazione effettiva, è il totale controllo della timbrica dei suoni generati piuttosto che la fedele imitazione di strumenti pre-esistenti, come è in effetti adottata negli strumenti a campionamento, dotati perciò di larghe quantità di memoria.

8 Bibliografia

[BB93] M. Barutti and G. Bertini. An Implementation of the Additive Synthesis Based on FFT^{-1}. Atti del Colloquio di informatica musicale, pages 127-133, 1993

[BCD77] G. Bertini, M. Chimenti, and F. Denoth. TAU2: Un terminale audio per esperimenti di ''Computer Music". Alta Frequenza, vol. 12, Dicembre 1977.

[BF93] G. Bertini and D. Fabbri. MultiC25 un sistema multi DSP con schede LeonardC25. Rapporto Interno del PF/CNR "Sistemi Informatici e calcolo parallelo", 1993.

[BMT99] G. Bertini, M. Magrini, L. Tarabella "Metodologie di analisi di segnali audio orientate al progetto di un sistema per la sintesi additiva basato su chip VLSI dedicati." C.N.R. Nota interna B4-34 1999

[BRST97&B] F. De Bernardinis, R.Roncella, R. Saletti, and P. Terreni & G. Bertini. "A Single Chip 1200 Sinusoids Real Time Generator for Additive Syntesis of Musical Signal". Int. Conf. On Acust. Speech Signal Processing IEEE, 1:427--430, April 1997.

[BRST&B98] F. De Bernardinis, R.Roncella, R. Saletti, P. Terreni. & G. Bertini "A new VLSI Implementation of Additive Synthesis" "Computer Music Journal" 22:3 pp. 49-61, Mass. Inst Technology, Fall 1998.

[Cha81] Gerard R. Charbonneau. Timbre and the Perceptual Effects of Three Types of Data Reduction. Computer Music Journal, 1981.

[FH2001] K. Fitz, L. Haken. "The Continuum Fingerboard and Real-Time Bandwidth-Enanched Additive Syntesis" http://www.cerlsoundgroup.org/Continuum/Cont.html, 2001.

[GG78] J. M. Grey and J. W. Gordon. Perceptual Effects of Spectral Modifications on Musical Timbres. Journal of the Acoustical Society of America, 63:1493--1500, 1978.

[Gre77] J. M. Grey. Multidimensional Perceptual Scaling of Musical Timbre. Journal of the Acoustical Society of America, 61:1270--1277, 1977.

[HB96] A. Horner and J. Beauchamp."Piecewise-Linear Approximation of Additive Synthesis Envelopes: A Comparison of Various Methods. Compuer Music Journal, 1996.

[HFT95] A. D. Houghton, A. J. Fisher, and F. Thierry."An ASIC for Digital Additive Sine Waves Synthesys". Computer Music Journal, 19:3:23:31, 1995.

[Jaf95] David A. Jaffe. Ten Criteria for Evaluating Synthesis Techniques. Computer Music Journal, 19(1):76--87, 1995.

[Jan91] C. Jansen. Sine Circuits, 10000 High Quality Sine Waves Without Detours. Proceedings International Computer Music Conference, 1991.

[Jen99] K. Jensen. "Timbre Models of Musical Sound". Københavns 1999.

[Moo77] J. A. Moorer. "Signal Processing Aspects of Computer Music" A Survey. Computer Music Journal, 1977.

[Pol83] De Poli. A Tutorial on Digital Sound Synthesis Tecniques. Computer Music Journal, 7(4):453--456, 1983.

[RD92] X. Rodet and P. Depalle. A New Additive Synthesis Method Using Inverse Fourier Trasform and Special Envelope. Proc. ICMC, pages 410,411, 1992.

[She62] R.N. Shepard. The Analisis of Proximities: Multidimensional Scaling with an Unknown Distance Function. Psychometrika, 27:125--140, 1962.

[TB92] L. Tarabella and G. Bertini."Informatica e Musica". Milano, 1992.

[XS97] X. Serra. "Integrating complementary spectral models in the design of a musical synthesizer" [published in the Proceedings of the International Computer Music Conference 1997] Barcelona 1997.

Riferimenti Internet

http://www.iua.upf.es/~xserra/articles/spectral-models/

http://www.iua.upf.es/mtg/

http://www.diku.dk/research-groups/musinf/

http://www.cerlsoundgroup.org/Continuum/Cont.html

Materiale file Wave

[Bia] M. Biagi http://ieee.ing.uniroma1.it/ring/ring0/wav.html

[Cro99] D. Cross http://www.intersrv.com/~dcross/wavio.html, 1999

Lavori di tesi

[Ber2000] S. Bertellotti. "Progetto e sintesi logica di una unità do controllo VLSI per un sistema integrato di sintesi musicale additiva" – Tesi di Laurea in Ingegneria Elettronica A.A. 1999/2000.

[Deb94] F. De Bernardinis. "Progetto VLSI di un sistema per la sintesi additiva di un segnale musicale" – Tesi di Laurea in Ingegneria Elettronica A.A. 1993/94.

[PP97] A. Peluso, L. Ponzetta "Metodologie di analisi di segnali audio orientate al progetto di un sistema basato su VLSI per la sintesi di suoni con tecnica additiva" – Tesi di Laurea in Scienza dell'Informazione A.A. 1997/98.