UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI

CORSO DI LAUREA TRIENNALE IN SCIENZE E TECNOLOGIE DELLA COMUNICAZIONE MUSICALE

METODI E PROTOTIPI SOFTWARE PER L'ESTRAZIONE DIRETTA DELL'INDICAZIONE DI TEMPO DA UN SEGNALE MP3

Relatore: Prof. Luca A. Ludovico

Correlatore: Dott. Antonello D'Aguanno

Elaborato Finale di:

Maurizio Botti

Matricola 678791

Anno Accademico 2007–08

Indice

Capitolo 1

Introduzione

La fruizione dell'informazione musicale ha subito considerevoli mutazioni dovute principalmente all'avvento di internet e allo sviluppo dei sistemi per la compressione audio. Grazie a queste innovazioni tecnologiche l'informazione musicale sta diventando un bene sempre più dematerializzato emancipandosi dai supporti fisici, utilizzati precedentemente come il veicolo privilegiato per la diffusione dell'informazione stessa. La musica quindi non è più vincolata a supporti fisici quali CD, DVD, nastri magnetici, ma grazie alla rivoluzione digitale è divenuta pura informazione. Questa trasformazione ha portato alla nascita di un nuovo mercato dedicato alla distribuzione dei contenuti musicali tramite la rete. Infatti svariate industrie, negli ultimi anni, si sono interessate a questo settore partecipando alla creazione di numerosi servizi in internet per lo streaming e il download di brani musicali. In questo contesto le collezioni private tendono ad assumere dimensioni equiparabili a quelle dei grandi archivi musicali tant'è che non è insolito per un singolo utente possedere una collezione di diverse migliaia di brani (generalmente in formato MP3). La possibilità di reperire musica on-line e il conseguente incremento delle collezioni private ha fatto sì che problematiche, precedentemente legate esclusivamente ai grandi archivi musicali, si presentassero anche nella gestione delle

collezioni private. L'enorme quantità di informazioni, ora in possesso anche dei singoli utenti, richiede quindi notevoli sforzi per la loro catalogazione; tutto ciò ha portato alla nascita di una nuova branca scientifica, detta MIR[1], che ha l'obbiettivo di sviluppare nuovi sistemi volti a facilitare o a rendere più efficace la catalogazione e la ricerca di informazioni musicali [1]. Grazie a tali sistemi è quindi possibile effettuare catalogazioni o ricerche di brani musicali non rimanendo vincolati ai soli dati catalografici quali autore o titolo, ma utilizzando anche informazioni relative al contenuto del brano in questione, come ad esempio la velocità metronomica, l'incipit musicale del brano considerato ecc.

Il lavoro effettuato in questo elaborato è volto all'individuazione di uno dei parametri musicali utilizzabili in tali attività di ricerca o archiviazione, ovvero la segnatura di tempo. Diversi studi sono già stati effettuati su tale argomento [2][3][4][5][6][7] ma nessuno di questi ha mai preso in considerazione i formati compressi limitandosi a perpetrare la ricerca di tale parametro nella partitura, nel MIDI o nell'audio in formato lineare. Oltre al fatto che i formati compressi sono molto utilizzati nella conservazione di brani musicali su computer, esiste un'altra motivazione che ha indotto questo elaborato ad essere orientato sull'impiego di tali formati, ovvero i vantaggi in termini di memoria e di tempo risparmiati grazie ad un'analisi diretta del file compresso (una trattazione dettagliata è riportata nel Capitolo 3). Pertanto l'algoritmo sviluppato cerca di calcolare la segnatura di tempo di un brano considerando informazioni reperibili direttamente da un MP3 senza che questo venga riportato in formato lineare. Per la precisione le informazioni utilizzate sono relative alla disposizione dei transienti rilevati dall'encoder al momento della compressione di un file audio. Prima dello sviluppo del software si è cercato di comprendere, tramite test di ascolto, se l'informazione ritmica fornita dall'encoder potesse essere sufficiente per l'individuazione della segnatura di tempo da parte di un essere umano e se questo fosse utile per sviluppare un metodo per la rile-

[1] Music Information Retrieval

vazione dell'indicazione di tempo. Successivamente sono state prese in considerazione diverse teorie, non legate ai test effettuati (dal momento che questi si sono rilevati poco utili per lo sviluppo di una teoria efficace per la rilevazione della segnatura di tempo), che hanno portato allo sviluppo di due software differenti entrambi incaricati di rilevare la suddivisione metrica dei brani musicali. In primo luogo si è scelto di utilizzare un metodo basato sull'autocorrelazione, ispirato alla teoria di Brown[4], mentre successivamente si è impiegato un metodo volto alla rilevazione degli accenti metrici presenti nel brano.

Nel Capitolo 2 verrà descritto il formato MP3, prestando particolare attenzione al metodo con cui vengono ricavate le informazioni ritmiche impiegate nell'analisi metrica, mentre nel Capitolo 3 saranno riportati i lavori precedenti relativi all'estrazione della segnatura di tempo. Nel Capitolo 4 verranno invece descritte le modalità di svolgimento dei test di ascolto e verranno riportati e analizzati i risultati di questi. Il Capitolo 5 comprenderà la descrizione degli algoritmi sviluppati e dei relativi test compiuti su questi. Le conclusioni e gli sviluppi futuri saranno contenuti nel Capitolo 6, seguito da due appendici contenenti il codice dei software prodotti; la bibliografia concluderà il lavoro.

Capitolo 2

Il formato MP3

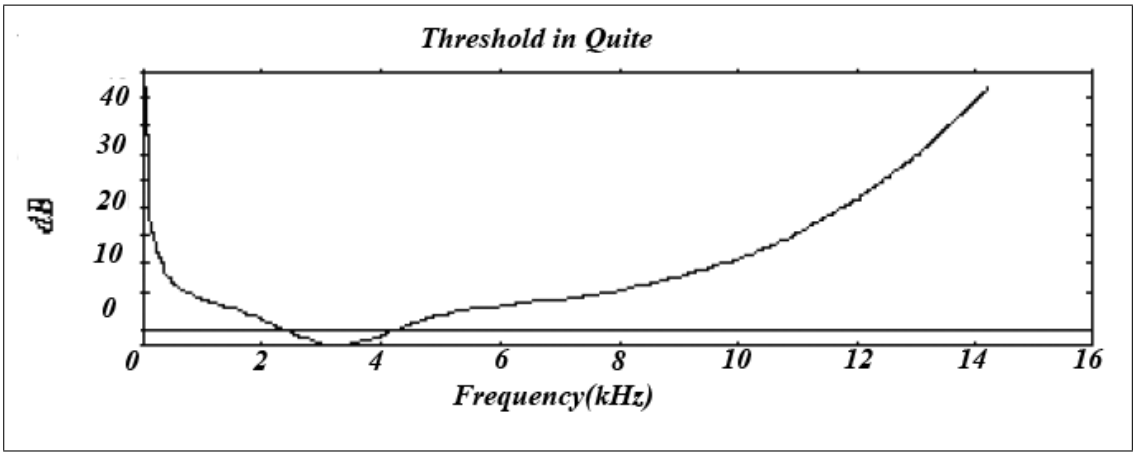
Grazie alla compressione audio è stato possibile ottimizzare la quantità di risorse necessarie per la codifica di un segnale audio. La riduzione del bitrate dei file audio ha quindi portato vantaggi sia nell'ambito dell'immagazzinamento dei file stessi sia in quello della loro trasmissione. Infatti al decrescere del valore del Bitrate, a parità di frequenza sul canale trasmissivo, si avrà un aumento della velocità di trasmissione dell'informazione. Esistono due categorie di algoritmi di compressione: gli algoritmi lossless e gli algoritmi lossy. I primi operano una compressione basata sull'eliminazione delle ridondanze numeriche, tale per cui il segnale audio decompresso risulta essere uguale a quello originale. In questo caso non vi è quindi perdita di informazione. Purtroppo i tassi di compressione sono notevolmente inferiori rispetto a quelli ottenibili con gli algoritmi lossy. Questa seconda categoria di algoritmi, invece, comporta l'eliminazione di informazioni ritenute superflue, ovvero non percepite dall'orecchio umano, attraverso un modello psicoacustico del sistema uditivo che permette la discriminazione di tali informazioni da quelle considerate rilevanti. L'algoritmo adottato nel formato MPEG Layer 3 appartiene a questa seconda categoria per cui per poterne comprendere il funzionamento è necessario focalizzare l'attenzione sul modello psicoacustico, trattando

quindi la fisiologia dell'orecchio.

2.1 Fisiologia dell'Orecchio

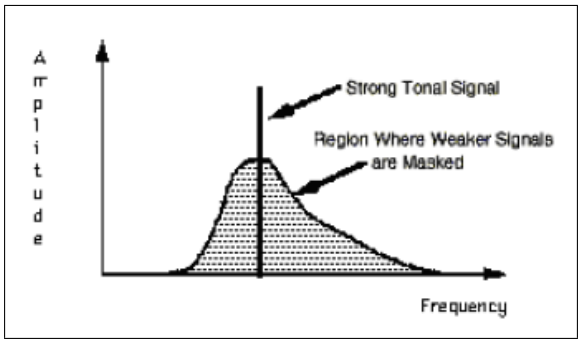
Nell'apparato uditivo umano si possono individuare tre sezioni: orecchio esterno, medio e interno. La prima sezione ha il compito di convogliare le onde sonore verso la membrana timpanica ed è costituita dal padiglione auricolare e dal condotto uditivo; la seconda sezione, composta dal timpano e dalla catena di ossicini, funge come adattatore di impedenza convertendo le onde sonore che si propagano nell'aria in onde idrauliche all'interno della coclea. Quest'ultima costituisce quindi la porzione dell'orecchio interno incaricata di trasdurre le onde idrauliche in impulsi elettrici, che per mezzo del nervo acustico verranno convogliati al cervello. All'interno della coclea è posta la membrana basilare, formata da una moltitudine di sottili fibre elastiche tese tra due creste ossee. Queste fibre si presentano fitte e corte nella regione periferica della coclea e diventano più lunghe man mano che si procede verso la regione interna. Grazie a questa struttura la rigidità della membrana risulta molto più elevata all'apice rispetto alla base, e questo fa sì che oscilli in punti diversi a frequenze differenti. La membrana basilare effettua quindi una conversione del suono in uno spettro e dalle peculiarità di tale conversione dipende ciò che viene effettivamente percepito dal nostro sistema uditivo. Ad esempio la membrana è in grado di vibrare approssimativamente ad una frequenza massima di 20 kHz, mentre la minima frequenza capace di metterla in oscillazione è di circa 20 Hz; questo definisce le soglie assolute relative alla percezione della frequenza del sistema uditivo umano. Oltretutto, a causa della particolare struttura dell'orecchio, la percezione dell'intensità varia in funzione della frequenza. Per esempio un tono di 10 kHz per essere percepito deve avere un intensità maggiore di 10 dB rispetto al minimo necessario per la frequenza di circa 3 kHz, come si evince dalla figura 2.1. Si può notare anche come vengano privilegiate le frequenze tipiche della voce umana, principalmente

Figura 2.1: Curva di percezione del suono del nostro orecchio in stato di quiete [8]



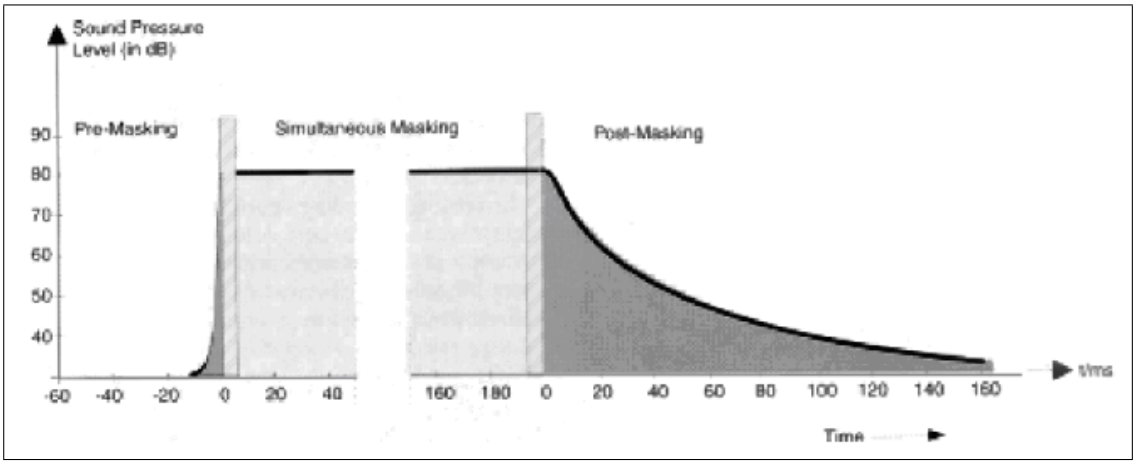
a causa della risonanza del condotto uditivo esterno (circa 3 kHz). Sopra la membrana basilare poggia l'organo del Corti che ha il compito di amplificare le vibrazioni di tale membrana e convertirle in segnali elettrici. L'amplificazione cocleare consente quindi la percezione di suoni di bassa intensità ma è anche responsabile di alcuni fenomeni di mascheramento. Infatti quando due toni di frequenza simile raggiungono l'apparato uditivo, quello a maggiore intensità causerà la soppressione del tono meno intenso. Questo fenomeno è dovuto al fatto che il tono più intenso manda in saturazione gli amplificatori cellulari della coclea nelle zone limitrofe alla vibrazione della membrana basilare indotta dal tono stesso. In questo modo, solo il tono più intenso riesce a stimolare il nervo acustico ed è quindi percepito. Viene quindi definita *banda critica* la regione della membrana basilare in cui un tono provoca il mascheramento dei toni a intensità inferiore. Questa tipologia di mascheramento viene detta *mascheramento frequenziale* e grazie a tale fenomeno il sistema uditivo è in grado di sopprimere una parte dei rumori che fanno da sfondo ai segnali. Nella figura 2.2 la zona ombreggiata rappresenta l'entità del mascheramento frequenziale relativa ad un particolare tono. Oltre a questa tipologia di mascheramento ne esiste un'altra, ovvero il *mascheramento temporale* derivante dal fatto che le reazioni dei nervi uditivi agli stimoli non sono istantanee. Il mascheramento

Figura 2.2: Mascheramento Frequenziale [8]



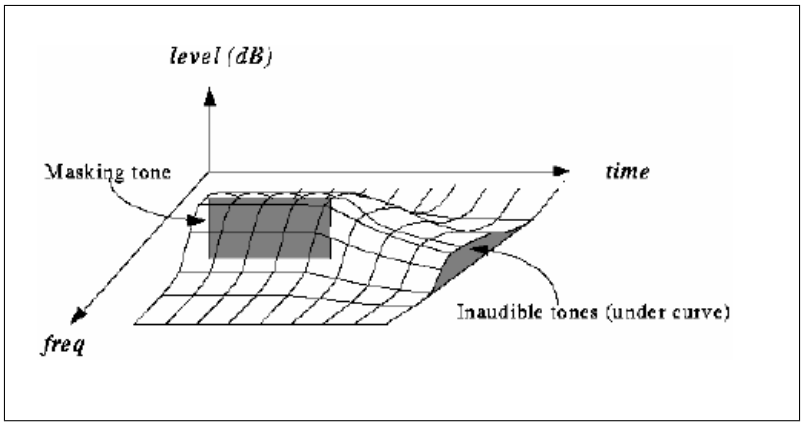
temporale può essere di due tipi: *pre-masking e post-masking*. Nel primo caso un tono interviene mascherando i toni che lo antecedono di pochi millisecondi, nel secondo caso invece la cessazione di un tono produce la perdita della sensibilità uditiva per toni di frequenza simile per un tempo compreso tra i 20 ms e i 500 ms. In figura 2.3 vengono indicati i due tipi di mascheramento temporale relativi ad un tono. Se si considera

Figura 2.3: Mascheramento Temporale [8]



un particolare tono, combinando i diversi tipi di mascheramento, è possibile costruire una superficie tridimensionale dalla quale si possono individuare i toni udibili e quelli mascherati. Ciò è rappresentato nella figura 2.4. L'encoder MP3 quindi rileva e omette

Figura 2.4: Mascheramento Complessivo [8]



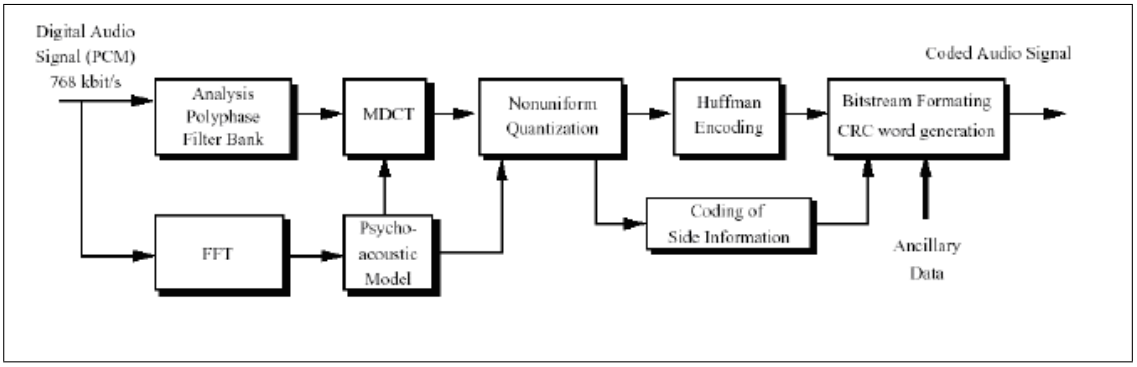
dalla codifica quelle componenti frequenziali che non vengono percepite e per fare ciò tiene conto dei fenomeni appena descritti.

2.2 Encoder e Decoder MP3

MPEG (Moving Pictures Expert Group) è un gruppo di lavoro che si occupa dello sviluppo di standard per la codifica di audio e video digitali. Gli standard creati dal gruppo fino ad oggi sono cinque: MPEG-1, MPEG-2, MPEG-4, MPEG-7 ed MPEG-21. MPEG-1 è il primo standard prodotto ed è quello più utilizzato per la codifica dell'informazione musicale, pertanto in questo contesto l'attenzione verrà focalizzata su di esso. Il sistema di codifica audio MPEG-1 è formato da tre entità fondamentali: il formato di codifica, l'encoder e il decoder. Il primo è un insieme di norme che definiscono come deve essere codificata e strutturata l'informazione audio compressa. L'encoder invece è un blocco software che prende in ingresso un file non compresso PCM e lo converte in un formato compresso, mentre il decoder è un blocco software che effettua l'operazione inversa rispetto all'encoder. Generalmente la maggior parte della complessità algoritmica è posta nell'encoder in modo da rendere più rapida possibile la fase di

decoding. Ogni encoder MPEG/audio può comprimere sfruttando diversi algoritmi di compressione. Per quanto riguarda MPEG-1 gli algoritmi esistenti sono tre: Layer 1, 2, 3. Il Layer 1 è il più semplice dei tre pertanto è in grado di operare la compressione più rapidamente rispetto agli altri. A causa della sua semplicità algoritmica i tassi di compressione ottenuti saranno inferiori rispetto a quelli relativi agli altri due layer a parità di qualità audio percepita. Questo algoritmo associa 384 campioni PCM ad ogni frame e il formato di file associato è l'MP1. Il Layer 2 è più complesso del primo, associa ad un frame 1152 campioni PCM e il formato di file associato è MP2. Il Layer 3, tra i tre, è quello che offre le migliori prestazioni arrivando ad ottenere segnali compressi di buona qualità anche con un bitrate compreso tra i 128 kbit/sec e 192 kbit/sec. Questo associa 1152 campioni PCM ad ogni frame e il formato di file associato è MP3. Ogni frame è poi suddiviso in due parti di 576 campioni ciascuna, ognuna delle quali è detta granulo. L'encoder pertanto prende in input un file PCM e lo legge a blocchi di 384, 576, o 1152 campioni a seconda del layer utilizzato. La struttura di un encoder Layer 3 è mostrata nella figura 2.5. Innanzitutto l'encoder prevede un banco di filtri polifasico

Figura 2.5: Struttura di un Encoder MP3 [9]



seguito da una MDCT (Trasformata Coseno Discreta Modificata). Questi due blocchi hanno il compito di convertire il segnale, rappresentato nel dominio del tempo, nella sua analoga rappresentazione nel dominio delle frequenze. Il banco di filtri polifasico

è composto da trentadue filtri passa-banda equispaziati che suddividono ciascun granulo in altrettante sottobande. In ogni sottobanda vengono quindi elaborati 18 campioni PCM. Il segnale elaborato dal banco di filtri polifasico viene poi convogliato al blocco MDCT. Tale blocco è una caratteristica peculiare del formato MP3 in quanto non è presente negli altri Layer [10, 11] ed è stato introdotto per fornire una maggiore risoluzione frequenziale al Quantizzatore Non-uniforme permettendogli di sfruttare al meglio i risultati del modello psicoacustico. Il segnale frazionato nelle 32 sottobande dal banco di filtri polifasico viene infatti ulteriormente ripartito, ogni sottobanda è quindi suddivisa in 18 parti ottenendo così un totale di 576 linee frequenziali. Questi valori sono relativi al primo granulo del frame per cui per ottenere i valori del secondo l'MDCT viene rieseguita con un overlap del 50% [12]. La partizione effettuata dal blocco MDCT non è sempre uguale, infatti in presenza di impulsi di elevata intensità all'interno del segnale ciascuna sottobanda viene ripartita localmente anziché in 18 in 6 parti. La scelta relativa a quale configurazione di MDCT utilizzare è dettata da un parametro calcolato dal modello psicoacustico detto: Entropia Psicoacustica. Tale valore determina quindi il tipo di finestratura da utilizzare ciascuna delle quali corrisponde ad una particolare suddivisione delle sottobande. Le finestrature esistenti sono 4:

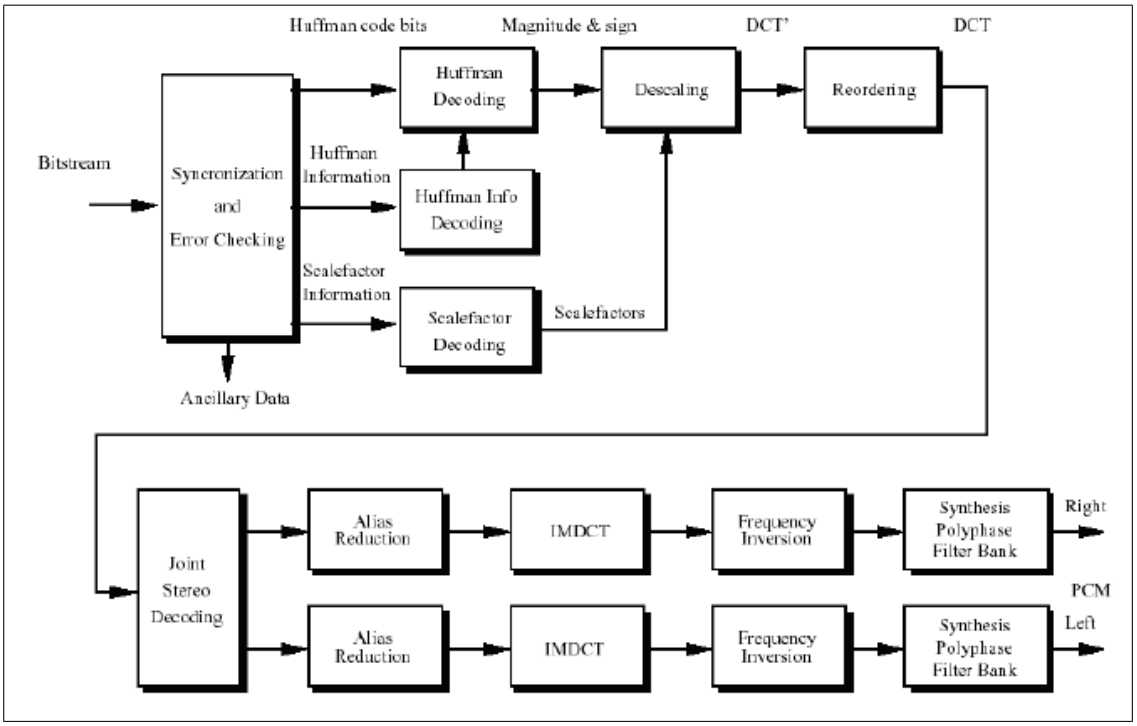
- Long
- Short
- Short-to-Long
- Long-to-Short

La finestra di tipo Long prevede una suddivisione di ciascuna sottobanda in 18 parti ed è utilizzata quando è richiesta una maggiore risoluzione frequenziale. La finestra Short, invece, viene selezionata quando si ha la necessità di avere una maggiore risoluzione temporale, in questo caso le sottobande sono suddivise in 6 parti. Le altre due finestre

sono finestre di transizione che permettono di passare in tempo reale da una finestra di tipo Long a una di tipo Short e viceversa. Un altro blocco che compone l'encoder è l'FFT(Fast Fourier Trasform). Anche questa sezione opera una conversione del segnale rappresentato nel dominio del tempo nella sua corrispondente rappresentazione nel dominio delle frequenze. In questo caso però lo spettro risultante andrà a costituire l'input del Modello Psicoacustico anziché del Quantizzatore-non-Lineare. Tale spettro è caratterizzato da una risoluzione molto elevata che consente al modello psicoacustico di lavorare nella maniera più efficace. Il modello psicoacustico rappresenta l'elemento fondamentale dell'encoder avendo il compito di discriminare le componenti frequenziali del segnale ritenute indispensabili per la sua ricostruzione da quelle considerate superflue. Il suo compito è quello di analizzare lo spettro ricevuto dal blocco FFT e determinare il livello di soglia di udibilità SMR (Signal to Mask Ratio) sfruttando i principi di psicoacustica del sistema uditivo umano precedentemente citati. Vengono quindi riconosciute le componenti frequenziali percepite dallorecchio da quelle non percepibili a seguito del mascheramento e tale informazione viene poi fornita al blocco Quantizzatore-non-Uniforme. Tale blocco ha il compito di codificare lo spettro del segnale, ricevuto dal blocco banco di filtri ibrido, in relazione alle informazioni fornite dal modello psicoacustico. Viene pertanto effettuata una quantizzazione non uniforme nella quale le componenti frequenziali maggiormente percepite vengono codificate con un maggior numero di bit mentre quelle meno percepite vengono codificate con meno bit. L'obbiettivo finale è quello di far si che il rumore di quantizzazione introdotto sia al di sotto della soglia di udibilità individuata dal modello psicoacustico. La codifica numerica dello spettro generata dal Quantizzatore viene poi ulteriormente compressa tramite l'algoritmo di Huffman. Questo effettua una compressione senza perdita di informazione che permette di ridurre ulteriormente la dimensione dell'informazione codificata. Infine l'ultimo blocco dell'encoder ha il compito di impacchettare l'output del blocco precedente secondo la sintassi dello standard MPEG. In pratica i frame e le

informazioni associate vengono inserite in un beatstrem compatibile con lo standard. In questa fase verranno anche estratti dei parametri che consentiranno di effettuare il controllo di correttezza dell'informazione in fase di decoding (CRC word generation). Come detto precedentemente il decoder effettua le operazioni inverse rispetto all'encoder, infatti questo prende in input un formato compresso MPEG e lo riporta nel formato non compresso PCM. La sua struttura è schematizzata dalla figura 2.6 Il primo blocco

Figura 2.6: Struttura di un decoder MP3[9]



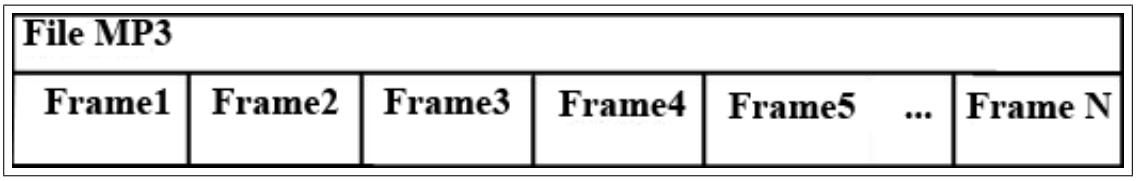
(Syncronization and Error Checking) riceve il bitstream MP3 identifica la posizione dei vari frame ed effettua un controllo di correttezza delle informazioni. Successivamente, grazie alle apposite tabelle, viene effettuata la decodifica di Huffman che genera le 576 linee frequenziali relative al granulo analizzato. Nel caso alcune frequenze siano mancanti queste vengono sostituite da frequenze nulle in modo da ottenere sempre 576 valori. Il blocco Reordering viene attivato solo se in fase di encoding è stata utilizza-

ta una finestra di tipo short dato che tale finestratura provoca un diverso ordinamento delle frequenze. Queste appaiono ordinate prima per finestra e quindi per frequenza, è dunque necessario effettuare un'operazione di ordinamento per ottenere la corretta sequenza dei valori. Altri due blocchi fondamentali del decoder sono l'IMDCT e il Banco di Filtri Polifasico (Synthesis Polyphase Filter Bank) che riconvertono lo spettro del segnale nella sua rappresentazione nel dominio del tempo. Il compito del primo blocco e l'esecuzione della Trasformata Coseno Discreta Modificata Inversa che trasforma lo spettro lineare formato da 1152 (576+576) valori in una matrice composta da 32 bande ognuna caratterizzata da 36 valori. Questo spettro costituisce quindi l'input del banco di filtri polifasico. Ovviamente anche l'IMDCT utilizza la finestratura impiegata in fase di encoding dall'MDCT.

2.3 Formato del File MP3

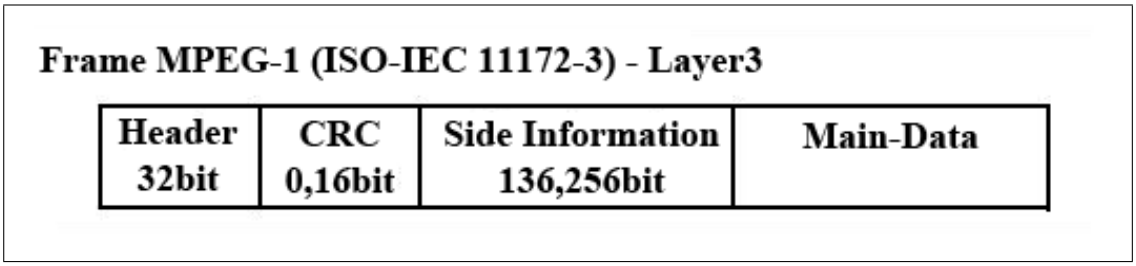
Un file MP3 è organizzato in frame [8, 9, 13] ognuno dei quali comprende le informazioni necessarie per ricostruire i corrispondenti campioni PCM. Ogni frame è indipendente dal resto del file, pertanto è possibile ricalcolare i campioni PCM relativi al frame con le sole informazioni in esso contenute. Questo fa si che la perdita di un frame, ad esempio in ambito di streaming, non comprometta la corretta decodifica del resto del file. In questo caso, infatti, i frame pervenuti vengono decodificati autonomamente mentre al posto del frame mancante verrà generato un silenzio. La figura 2.7 mostra la composizione di un file MP3. Ogni frame è composto da quattro parti fondamentali: header, CRC, side information, Main-Data. L'header contiene informazioni necessarie per descrivere il frame come Synch Word, che consente di individuare l'ordine dei vari frame, il tipo di codifica MPEG utilizzato, il Layer utilizzato(perché il file sia MP3 il Layer utilizzato deve essere necessariamente il 3), il valore del Bitrate in kbit/sec, la frequenza di campionamento, il tipo di codifica di canale, informazioni sui diritti di copyright e

Figura 2.7: Struttura di file MP3



informazioni che indicano se il brano è un originale o una copia. Il campo CRC viene invece utilizzato per il controllo degli errori ed è di fondamentale importanza nell'ambito dello streaming audio su internet dove è molto probabile il verificarsi di errori e la perdita di informazioni. Il campo side information comprende invece quelle informazioni necessarie per la corretta decodifica dei dati audio come: un puntatore all'inizio dei main-data, informazioni sulla posizione delle regioni codificate con Huffman e sulle tabelle impiegate, informazione sulla dimensione dei main-data. Il campo main-data contiene invece lo spettro relativo ai 1152 campioni PCM codificato con l'algoritmo di Huffman.

Figura 2.8: Struttura di un frame MP3



2.4 Effetto Pre-Eco e Window-Switching

Come precedentemente evidenziato il formato MP3 mette a disposizione 4 tipi diversi di finestratura: long, long-to-short, short, short-to-long indicizzate rispettivamente

conn0,1,2,3. La finestratura di tipo long offre la miglior risoluzione frequenziale per i segnali audio con caratteristiche stazionarie. Nelle finestre di tipo long abbiamo 576 linee frequenziali viceversa le finestre di tipo short hanno solamente 192 linee frequenziali con 32 linee frequenziali principali successivamente suddivise in 6 sottobande dallMDCT. 3 finestre short sono quindi raggruppate in un solo granulo in modo da mantenere invariata la dimensione del granulo a prescindere dal tipo di finestra utilizzata. I valori sono ordinati prima rispetto alla finestra e successivamente rispetto alla frequenza, quindi avremo uno schema del tipo:

$$f_{1,1}, f_{1,2}, f_{1,3}, f_{2,1}, f_{2,2}, f_{2,3} \ldots f_{192,1}, f_{192,2}, f_{192,3}$$

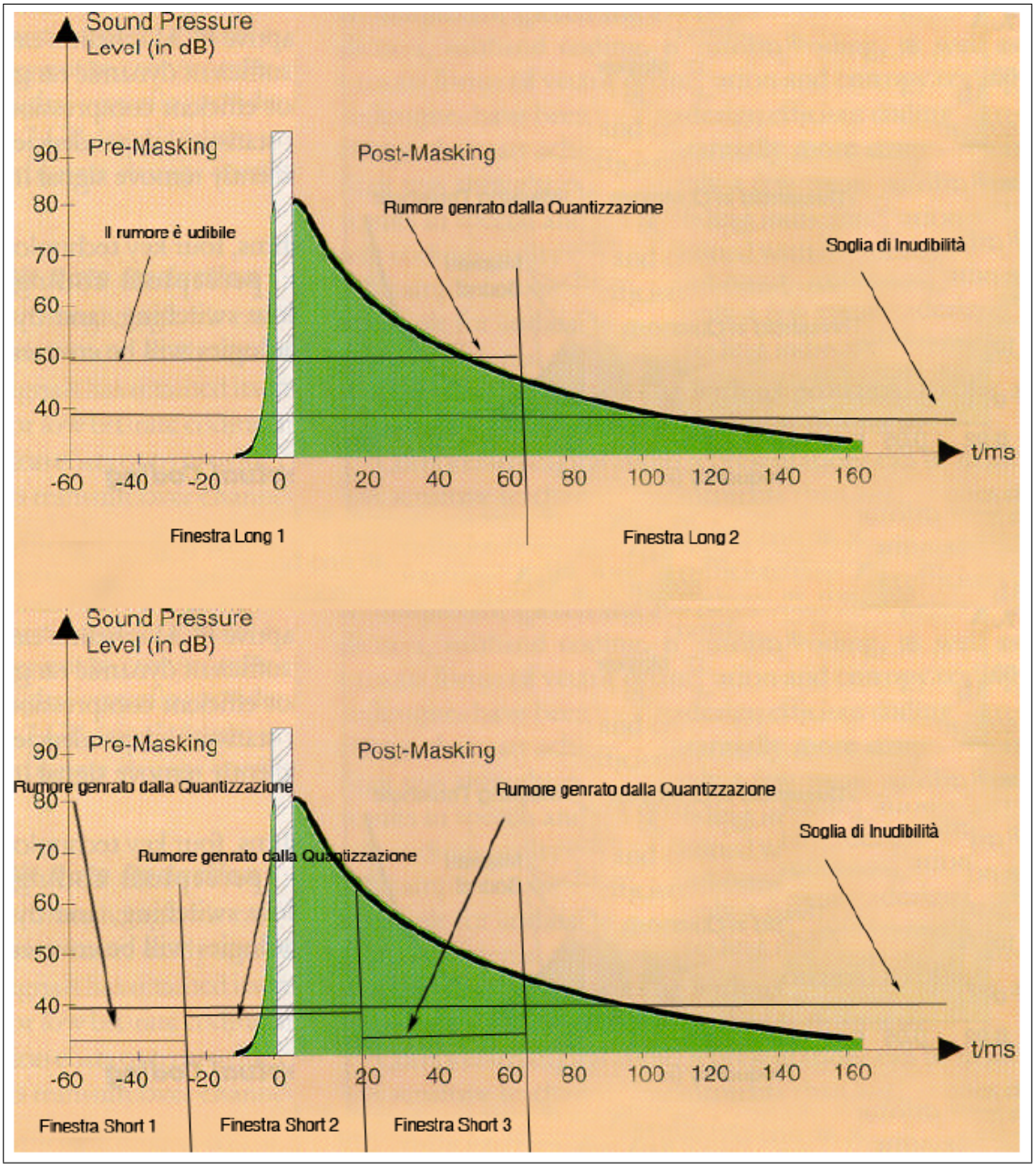
indicando con f la specifica linea frequenziale, con il primo indice la linea frequenziale e con il secondo indice la finestra di appartenenza. Il passaggio da una finestra di tipo long a una tipo short non è istantaneo. Le due finestre Long-to-Short e Short-to-Long hanno proprio questo scopo, cioè permettere la transizione tra le due finestre principali. Poiché l'utilizzo di una trasformata MDCT garantisce una migliore risoluzione frequenziale questa implica anche una peggiore risoluzione temporale. Di conseguenza, la quantizzazione dei valori MDCT genererebbe una distorsione udibile su tutta la finestra Long nel caso di segnali audio con caratteristiche non stazionarie, come ad esempio con forti rumori impulsivi quali colpi di batteria, note con una dinamica particolarmente elevata e così via. Per evitare queste problematiche comunemente note con il termine di Pre-Echo[1] viene quindi impiegata una finestra di tipo short ogni volta che viene rilevato un transiente. In figura 2.9 è schematizzata la comparsa di questo tipo di artefatto. Dalla figura si può subito comprendere l'utilità del cambiamento di finestratura per prevenire l'insorgere di questi artefatti. Come detto questa situazione è tipica nei brani musicali con forte accentuazione ritmica. La transizione da finestra Long a finestra Short è ge-

[1] Con questo termine si intende un tipico artefatto che può essere generato dalla compressione MP3.Questo artefatto è presente subito prima di un forte rumore impulsivo e si presenta come un eco attenuato e distorto del reale segnale presente.

stita trmite il 'Perceptual Entropy criterion': se il valore del PE[2] in un certo granulo è superiore alla soglia predefinita PE-SWITCH l'encoder cambia la finestratura da Long a Short [14][15]. Il metodo per scegliere il tipo di finestra da utilizzare è una parte dello standard MP3 non vincolante, ogni sviluppatore è libero di inserire un proprio criterio di selezione della finestra e può addirittura omettere l'utilizzo di finestre Short, codificando i brani solo tramite finestre di tipo Long. Il valore del PE non è sempre in grado di individuare correttamente la presenza di segnali impulsivi, ad esempio Il valore del PE è influenzato anche dalla distribuzione energetica del segnale oltre che dalle caratteristiche tonali, quindi in letteratura sono stati proposti metodi alternativi per gestire il cambio di finestratura [16].

[2]Perceptual Entropy

Figura 2.9: Nel grafico in alto il rumore di quantizzazione è presente su tutta la finestra 1, di tipo Long, con intensità costante ed è udibile subito prima del segnale impulsivo. Nel grafico più in basso, invece, le 3 finestre di tipo Short rendono questo rumore differente per ognuna di loro,permettendo di eliminare l'artefatto del Pre-Echo



Capitolo 3

Stato dell'Arte

3.1 Modello Musicale

Una qualsiasi persona, anche priva di educazione musicale, è in grado di percepire nella musica la ricorrenza di eventi che si ripropongono periodicamente, infatti chiunque è in grado di battere il piede in sincrono con un brano musicale e ad esempio, non è insolito vedere bambini di pochi mesi ondeggiare a tempo con la musica. É quindi possibile riconoscere in un brano una pulsazione che si ripropone costantemente nel tempo e seguirla con il movimento del piede. Ognuna di queste pulsazioni viene generalmente indicata con il nome di beat e la frequenza dei beat, espressa in beat al minuto (BPM), può indicare la velocità di metronomo a cui è stato eseguito un certo brano. Oltre alla frequenza di beat, nella musica, sono presenti altre periodicità individuabili da un ascoltatore come ad esempio il metro. Nella teoria musicale il termine metro indica una struttura caratterizzata dalla ricorrenza periodica di elementi accentuativi, riconoscibile nei vari brani musicali. Tale struttura consente quindi la ripartizione di un brano in misure, ognuna delle quali è caratterizzata da un accento metrico forte collocato all'inizio della misura stessa. Il metro quindi consente di raggruppare le pulsazioni assegnando a

ognuna di queste una diversa tipologia di accento. Tali accenti non sono sempre enfatizzati nell'esecuzione anche se l'apparato uditivo umano tende comunque a riconoscerli e questi possono essere suddivisi in tre categorie: accenti forti, mezzo forti e deboli. I primi, come già detto, indicano l'inizio di una misura mentre gli altri sono disposti sulle rimanenti pulsazioni all'interno della misura stessa. A seconda della periodicità con cui ricade l'accento forte potremo quindi definire il metro come binario, ternario, quaternario, quinario, senario, settenario e così via. Ad esempio nel metro binario l'accento forte si ripropone ogni due movimenti, mentre nel ternario ogni tre e così via. I metri binario e ternario sono detti primari mentre gli altri sono considerati secondari in quanto possono essere pensati come la somma di due o più metri primari. A seconda del tipo di suddivisione del movimento i metri possono essere classificati come semplici o composti. Nei primi ogni movimento è suddiviso in due parti mentre nei secondi si ha una ripartizione ternaria. Nella figura 3.1 viene mostrata la disposizione degli accenti nelle misure caratterizzate dai metri dal binario al quaternario. Generalmente nella scrittura musicale

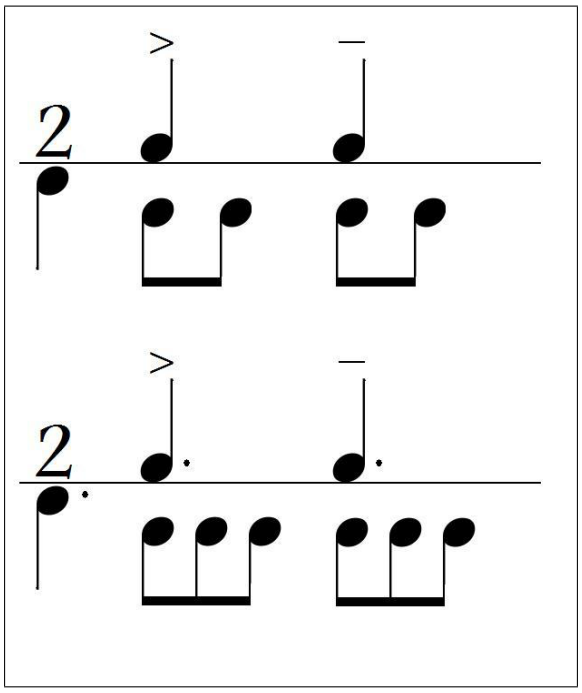
Figura 3.1: Accenti metrici

	SEMPLICI	COMPOSTE
MISURE BINARIE		
accenti principali	> —	> —
accenti secondari	(> –) (> –)	(> – –) (> – –)
MISURE TERNARIE		
accenti principali	> — ≥	> — ≥
accenti secondari	(> –)(> –)(> –)	(> – –) (> – –)(> – –)
MISURE QUATERNARIE		
accenti principali	> — ≥ —	> — ≥ —
accenti secondari	(> –) (> –) (> –) (> –)	(> – –)(> – –)(> – –)(> – –)

per indicare il metro viene impiegata la frazione metrica che può essere rappresentata graficamente in due modi diversi: utilizzando come denominatore un valore numerico

oppure un simbolo corrispondente al valore temporale del movimento nella misura. Nel caso in cui la frazione sia espressa numericamente, numeratore e denominatore hanno significati diversi a seconda del fatto che il metro considerato sia semplice o composto. Nel primo caso il numeratore indica il numero dei movimenti presenti in una misura e il denominatore rappresenta il valore temporale di ogni movimento; nel secondo caso numeratore e denominatore indicano rispettivamente il totale delle suddivisioni contenute nell'intera misura e l'unità di suddivisione. Nella figura 3.2 vengono riportati due esempi relativi a misure binarie. Nel primo esempio la misura è composta da due movimenti

Figura 3.2: Esempio di metro binario con suddivisione binaria e ternaria dei movimenti



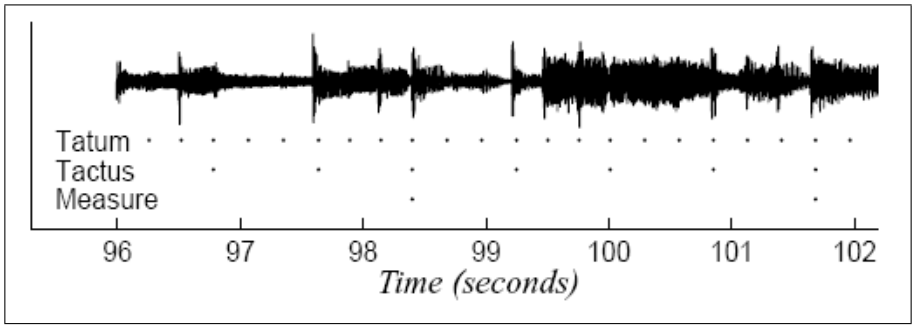
ognuno dei quali è ripartito in due parti; l'indicazione di tempo corrispondente è quindi $\frac{2}{4}$. Nel secondo esempio siamo sempre in presenza di due movimenti ma a suddivisione ternaria. In questo caso la segnatura di tempo utilizzata è $\frac{6}{8}$. La segnatura di tempo di un brano è quindi una di quelle informazioni facilmente rilevabili da un ascoltatore in pos-

sesso di sufficienti competenze musicali. In questo lavoro cercheremo di comprendere se tale rilevazione può essere effettuata anche da un computer.

3.2 Lavori Precedenti Relativi all'Estrazione della segnatura di tempo

Il problema dell'estrazione dell'indicazione di tempo è stato già affrontato in diversi articoli anche se tuttora esistono pochi algoritmi capaci di effettuare tale operazione [17][18], i quali prevedono forti restrizioni per poter funzionare correttamente. La maggior parte di questi algoritmi non lavora direttamente sull'audio ma cerca di estrapolare la segnatura di tempo da una performance, generalmente in formato MIDI, o da una partitura generata dalla performance stessa. Nei lavori analizzati in questo elaborato vengono considerate tre entità legate al metro musicale denominate *Tatum*, *Tactus*, e *Measure*, individuabili nella figura 3.3. Il termine *Tatum* deriva dall'espressione *"tem-*

Figura 3.3: Tatum, Tactus e Measure [19]



poral atom" [20] e indica il valore temporale più piccolo utilizzato in un certo brano musicale. Nel caso specifico dell' esempio riportato in figura 3.3 il *Tatum* corrisponde alla terzina di ottavi. Con *Tactus* viene invece indicata la pulsazione che consente di determinare la velocità di metronomo relativa al brano in esame, ovvero il Beat. Molti

lavori in ambito MIR sono dedicati all'analisi del *Tactus*. In questo lavoro sono stati presi in considerazione solo quelli che indicavano come possibili sviluppi futuri lavori sull'estrazione della segnatura di tempo. Infine il termine *Measure* si riferisce alla durata della battuta musicale nota anche come misura. Nell'esempio della figura 3.3 il *Tactus* corrisponde alla figura del quarto e la *Measure* ad una battuta di $\frac{4}{4}$. Come detto precedentemente gli accenti metrici consentono l'identificazione delle misure all'interno di un brano musicale e pertanto sono molto importanti per l'analisi metrica. Questi enfatizzano i vari momenti della musica e corrispondono generalmente all'inizio dei vari eventi musicali come ad esempio le variazioni timbriche o armoniche. Lerdahl and Jackendoff sostengono infatti che i momenti di *"stress musicale"* presenti nel segnale consentono all'ascoltatore di estrapolare pattern regolari, i quali consentono l'individuazione di periodicità come la ripetizione degli accenti metrici forti:

> *"the moments of musical stress in the raw signal serve as cues from which the listener attempts to extrapolate a regular pattern [21]"*

Molti dei lavori analizzati in questo elaborato prendono quindi in considerazione gli accenti metrici per la rilevazione della segnatura di tempo. La tabella 3.1 mostra un elenco dei più recenti sistemi per l'analisi del metro musicale dalla quale si evince che tali sistemi possono essere suddivisi in due categorie sulla base del tipo di dati che processano. Avremo quindi sistemi che utilizzano come input rappresentazioni simboliche dell'audio (partitura o MIDI) e altri che prendono in input l'audio stesso. La colonna *"Evalutation Material"*, specifica il tipo di materiale musicale preso in considerazione nei diversi lavori, mentre la colonna *"Aim"* indica lo scopo prefissato dai lavori stessi. Il contenuto di quest'ultima colonna evidenzia come solo alcuni di questi lavori siano focalizzati sull'estrazione della segnatura di tempo, anche se il problema viene affrontato da tutti gli articoli citati. Infine le tre colonne *"Approach"*, *"Mid-level Rappresentation"* e *"Computation"* descrivono i tre aspetti principali della tecnica impiegata per

arrivare al risultato dell'analisi. Analizzando la tabella si può riscontrare che i lavori dedicati all'audio sono la minor parte e in questi il formato compresso non è mai stato preso in considerazione. Nei prossimi paragrafi verranno quindi enunciati i principali metodi esistenti per rilevare l'indicazione di tempo dalla partitura, dal MIDI e quindi dall'audio non compresso. Verranno inoltre presi in considerazione alcuni dei lavori esistenti relativi al Beat Tracking e al Tempo Induction su formati compressi.

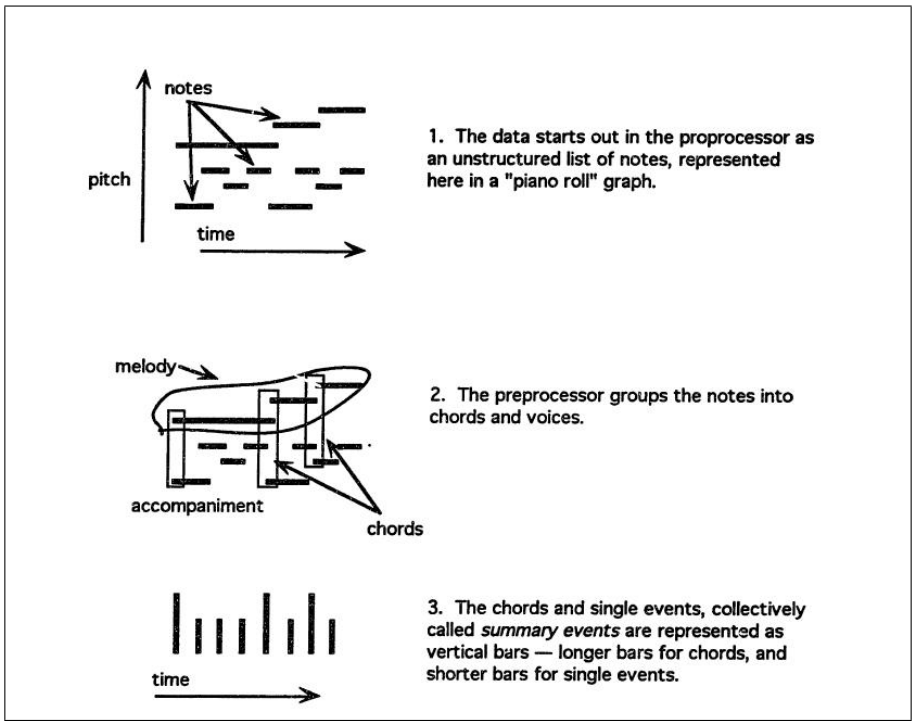
Tabella 3.1: Caratteristiche dei vari sistemi per l'analisi metrica [19]

Reference	Input	Aim	Approach	Mid-level representation	Computation	Evaluation material
Rosenthal, 1992[2]	MIDI	meter,time quantization	Rule-based,model auditory organization	At a preprocessing stage, notes aregrouped into streams and chords	Multiple-hypothesis tracking (beam search)	92 piano performances
Brown, 1993[4]	score	meter	DSP	Initialize a signal with zeros, then assign note-duration values at their onset times	Autocorrelation function (only periods were being estimated)	19 classical scores
Large, Kolen, 1994[22]	MIDI	meter	DSP	Initialize a signal with zeros, then assign unity values at note onsets	Network of oscillators(period and phase locking)	A few example analyses; straightforward to reimplement
Parncutt, 1994[3]	score	meter, accent modeling	Rule-based, based on listening tests	Phenomenal accent model for individual events (event parameters: length, loudness, timbre, pitch)	Match an isochronous pattern to accents	Artificial synthesized patterns
Temperley, Sleator, 1999[23]	MIDI	meter,time quantization	Rule-based	Apply discrete time-base, assign each event to the closest 35ms time-frame	Viterbi; cost functions for event occurrence, event length, meter regularity	Example analyses; all music types; source code available
Dixon, 2001[24]	MIDI, audio	tactus	Rule-based, heuristic	MIDI: parameters of MIDI-events. Audio: compute overall amplitude envelope, then extract onset times	First find periods using IOI histogram,then phases with multiple-agents (beam search)	222 MIDI files (expressive music); 10 audio files (sharp attacks); source code available
Raphael, 2001[25, 26]	MIDI, audio	tactus,time quantization	Probabilistic generative model	Only onset times are used	Viterbi; MAP estimation	Two example analyses; expressive performances
Cemgil, Kappen, 2003[27]	MIDI	tactus,time quantization	Probabilistic generative model	Only onset times are used	Sequential Monte Carlo methods; balance score complexity vs. tempo continuity	216 polyphonic piano performances of 12 Beatles songs; clave pattern
Goto, Muraoka, 1995, 1997[28, 5]	audio	meter	DSP	Fourier spectra,onset components (time, reliability, frequency range)	Multiple tracking agents (beam search); IOI histogram for periodicity analysis; pre-stored drum patterns used in (1995)	85 pieces; pop music;4/4 time signature
Scheirer, 1998[29]	audio	tactus	DSP	Amplitude-envelope signals at six subbands	First find periods using a bank of comb filters, then phases based on filter states	60 pieces with strong beat; all music types; source code available
Laroche, 2001[30]	audio	tactus, swing	Probabilistic	Compute overall loudness curve,then extract onset times and weights	Maximum-likelihood estimation; exhaustive search	Qualitative report; music with constant tempo and sharp attacks
Sethares, Staley, 2001[6]	audio	meter	DSP	RMS-energies at 1/3-octave subbands	Periodicity transform	A few examples; music with constant tempo
Gouyon et al., 2002[31]	audio	tatum	DSP	Compute overall amplitude envelope, then extract onsets times and weights	First find periods (IOI histogram), then phases by matching isochronous pattern	57 drum sequences of 210 s. in duration; constant tempo
Klapuri et al., 2003[7]	audio	meter	DSP, probabilistic back-end	Degree of accentuation as a function of time at four frequency ranges	First find periods (bank of comb filters, Viterbi back-end), then phases using filterstates and rhythmic pattern matching	474 audio signals; all music types

3.2.1 Estrazione della segnatura di tempo dalla partitura e dal MIDI

Uno dei primi sistemi per la rilevazione dell'indicazione di tempo fu proposto da Rosenthal nel 1992 [2]. Tale sistema, basato sull'emulazione della percezione umana del ritmo e della suddivisione metrica dei brani musicali, lavora su delle performance di piano memorizzate in formato MIDI, le quali sono sottoposte ad un preprocessing in cui le note vengono raggruppate in gruppi melodici e accordi come mostrato in figura 3.4. Per ciascuna di queste performance vengono poi create delle ipotesi relative alle

Figura 3.4: Preprocessing di performance di piano [2]



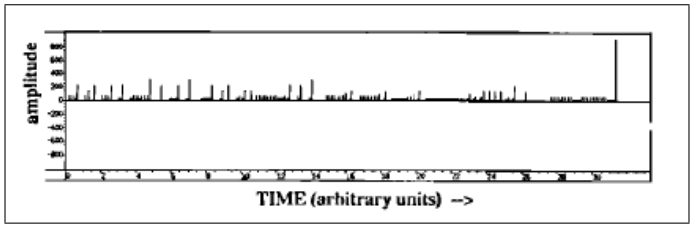
possibili segnature di tempo e vengono applicate una serie di regole adatte a classificare e a mettere a confronto le diverse ipotesi nonché a condurre una ricerca (beam-serch) finalizzata all'individuazione dell'ipotesi migliore. La strategia della beam-serch era stata già proposta da Allen e Dannenberg in [32]. Parncutt, nel suo lavoro [3], ha invece

proposto un dettagliato modello per l'individuazione della segnatura di tempo basato su test di ascolto. I test di ascolto effettuati erano suddivisi in due tipologie, nella prima venivano proposti dei pattern di beat aventi IOI[1] predeterminati. Questi pattern venivano presentati a sei diverse velocità e veniva chiesto agli ascoltatori di battere il piede a tempo con i beat percepiti. Dai risultati di questi esperimenti è emerso che ai tempi più veloci gli IOI più rilevanti per la determinazione dei BPM sono quelli più lunghi (fino ad una ampiezza massima di 1s). Nella seconda tipologia di test venivano riproposti i precedenti pattern con l'aggiunta di un suono, il cui timbro era di volta in volta modificato. Il partecipante al test doveva individuare se tale suono cadesse o meno su di un beat. L'algoritmo di Parncutt calcola la rilevanza di diverse strutture metriche basandosi su di un modello quantitativo della fenomenologia degli accenti e delle pulsazioni ottenuto elaborando i risultati degli esperimenti. Il modello è infatti basato sull'idea che l'accento metrico di un evento dipenda dalla distanza temporale tra esso e il beat seguente, pertanto maggiore sarà la durata dell'IOI tra l'evento considerato e il successivo e maggiore sarà la sua rilevanza. Nel 1993 Brown ideò un metodo per la rilevazione dell'indicazione di tempo dalla partitura basato sulla funzione di autocorrelazione, in cui le informazioni relative all'altezza delle note non venivano prese in considerazione. In questo metodo, descritto in [4], è prevista l'estrazione da una partitura di una singola linea melodica che verrà opportunamente trattata in modo da rendere significativa l'applicazione della funzione di autocorrelazione. Il file che costituirà l'input dell'algoritmo di Brown sarà quindi caratterizzato da una serie di ampiezze opportunamente pesate corrispondenti all'inizio delle note della melodia. Viene infatti assegnato, all'inizio di ogni nota, un valore di ampiezza proporzionale alla durata della nota in questione, mentre ai rimanenti momenti che costituiscono la melodia viene assegnato un valore pari a zero. In questo modo una nota da $\frac{2}{4}$ assumerà un valore di ampiezza doppio rispetto ad una nota da $\frac{1}{4}$ e così via. In questa fase di preprocessing viene anche effettuato un campio-

[1] L'acronimo IOI sta per Inter Onset Interval e indica il tempo che intercorre tra la fase iniziale di un evento musicale e l'inizio dell'evento successivo.

namento con una frequenza pari a 200Hz in modo da mantenere la compatibilità con la maggior parte dei dispositivi MIDI e viene assegnato al quarto il valore temporale di 0,25s. Nella figura 3.5 si può notare il risultato delle operazioni di preprocessing effettuate sul secondo movimento della sonata K. 310 di Mozart. La rappresentazione della

Figura 3.5: Secondo movimento della sonata K. 310 di Mozart [4]

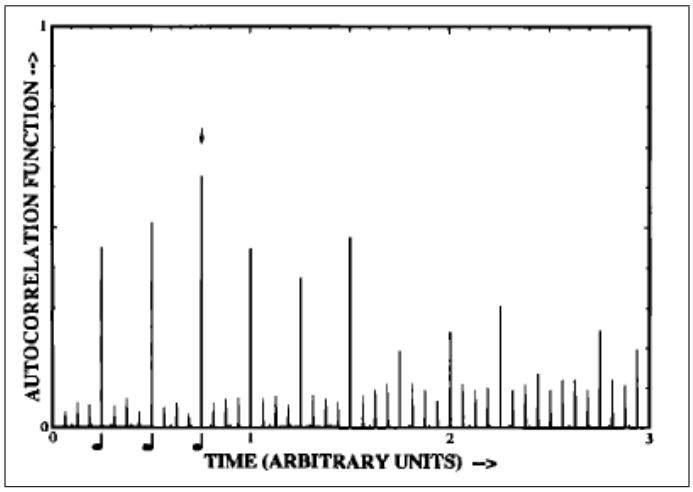


linea melodica così ottenuta costituirà quindi l'input dell'algoritmo e sarà processata con la funzione di autocorrelazione definita come:

$$A[m] = \sum_{n=0}^{N-1} x[n]x[n+m] \tag{3.1}$$

Dove N indica il numero di campioni di melodia considerati e *m* il tempo di autocorrelazione ovvero la traslazione temporale subita dalla melodia per essere confrontata con se stessa. Il valore massimo di *m* viene generalmente scelto in modo da considerare diverse misure nel calcolo della funzione. Il metodo dell'autocorrelazione viene quindi utilizzato perché dà una misura della frequenza con cui si ripropongono eventi che seguono un evento collocato a tempo zero. Quindi se gli eventi ricorrono principalmente all'inizio delle misure come riscontrato da Palmer e Krumhansl in [33] i picchi della funzione di autocorrelazione conterranno le informazioni necessarie per individuare l'inizio delle diverse misure. In questo approccio quindi i picchi che corrispondono alla pulsazione più veloce nella funzione di autocorrelazione consentono di determinare il denominatore della segnatura di tempo mentre il picco con la maggiore ampiezza consente di determinarne il numeratore. Nella figura 3.6 viene mostrato l'andamento della funzione di autocorrelazione relativo ad un brano in $\frac{3}{4}$. Uno dei lavori più recenti nell'ambito del-

Figura 3.6: Autocorrelazione di un brano in $\frac{3}{4}$ [4].



l'analisi metrica a partire dal MIDI è quello di Large e Kolen [22], che hanno associato la percezione metrica alla risonanza e hanno proposto un oscillatore "entrainment"[2] che modifica il suo periodo e la sua fase in funzione del pattern di impulsi rappresentante i tempi di attacco degli eventi musicali presenti nel file MIDI.
Temperley e Sleator in [23]crearono invece un algoritmo basato sulle regole di preferenza descritte da Lerdahl e Jackendoff in [21]. Questo algoritmo produce come output delle gerarchie metriche che permettono l'individuazione dell'indicazione di tempo da un generico file MIDI.

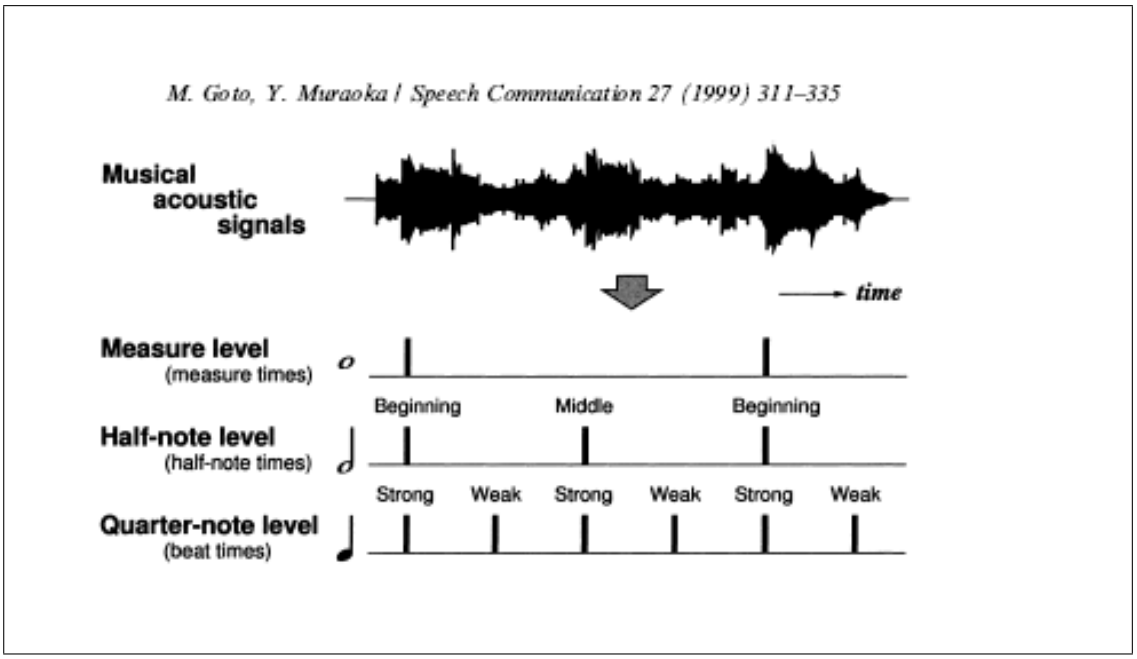
3.2.2 Estrazione della segnatura di tempo da formati non compressi

Come asserito precedentemente gli accenti metrici sono spesso in relazione con le variazioni armoniche all'interno di un brano. Su questo principio si basa il lavoro di Goto e Muraoka [5],il quale può essere visto come un'evoluzione del lavoro precedente [28], dedicato all'individuazione di tre livelli metrici all'interno di brani di musica pop esclu-

[2]Entrainment indica un processo di sincronizzazione tra due oscillatori.

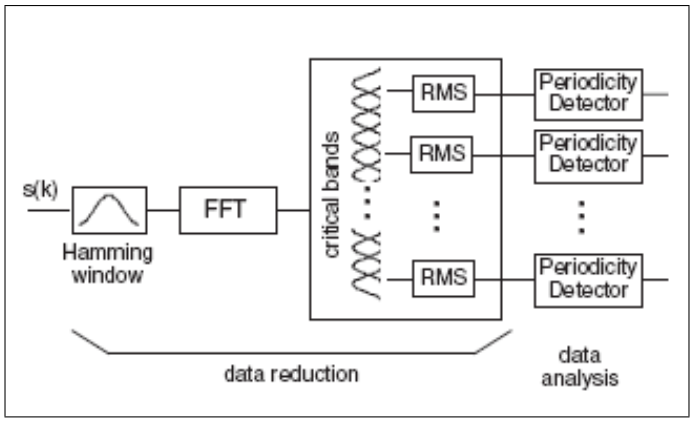
sivamente in $\frac{4}{4}$. Lo scopo di questo lavoro, infatti, non consiste nella rilevazione della segnatura di tempo ma nell'individuazione di tre livelli metrici chiamati: Measure level, Half-note level e Quarter-note level. Il sistema lavora in tempo reale su audio in formato PCM relativo a brani di musica pop caratterizzati da una segnatura di tempo di $\frac{4}{4}$ e da una frequenza di beat per minuto compresa tra 61 e 120. Dei tre livelli precedentemente citati il Quarter-note level corrisponde alla sequenza di beat nel tempo e la sua individuazione consiste in una operazione di beat-tracking effettuata, in questo, contesto tramite un'analisi frequenziale e la funzione di autocorrelazione. L' Half-note level viene invece ricavato stabilendo quale tra i beat precedentemente rilevati possiede un accento metrico forte. Ognuno di questi designa quindi l'inizio di una nota da due quarti. L'individuazione del Measure level corrisponde invece alla rilevazione dell'inizio delle misure, stabilendo quale delle note da due quarti del Half-note level cade all'inizio e quale a metà della battuta. Nella figura 3.7 sono rappresentati i tre diversi livelli metrici individuabili nell'audio. In questo lavoro il procedimento volto a determinare

Figura 3.7: Livelli metrici [5]



i tre livelli metrici è suddiviso in tre passi. Innanzitutto viene effettuato il beat-traking per individuare i quarti, vengono poi esaminate diverse ipotesi relative alla disposizione degli accenti metrici sui beat ed infine viene scelta l'ipotesi migliore secondo criteri di natura musicale. I criteri che consentono la discriminazione dell'ipotesi più rilevante dalle altre sono basati sulla rilevazione delle variazioni armoniche attraverso un analisi frequenziale del segnale audio. È importante specificare che il sistema non è in grado di identificare gli accordi ma solo di rilevare il momento di transizione tra un accordo e l'altro. È infatti quest'ultimo a indicare la presenza o meno di un accento metrico. Anche se il lavoro di Goto non è finalizzato all'estrazione della segnatura di tempo è comunque rilevante perché si occupa di analisi metrica rivolta all'individuazione delle misure e costituisce il punto di partenza di altri lavori come ad esempio quello di Klapuri [19]. Un altro lavoro relativo all'analisi metrica dell'audio in formato non compresso è quello di Sethares e Staley [6] che presenta un metodo, basato su teorie psicoacustiche, per la riduzione dei dati audio finalizzato all'analisi del ritmo nelle performance musicali. Il sistema, come si evince dalla figura 3.8, prevede due fasi: una riduzione dei dati audio in ingresso al sistema e un'analisi delle periodicità. Nella prima fase, il

Figura 3.8: Sistema di analisi di Sethares e Staley [6]



segnale audio rappresentato nel dominio del tempo viene trasformato nell'analoga rappresentazione nel dominio delle frequenze tramite una FFT e suddiviso in 23 bande,

ognuna delle quali ha un ampiezza pari ad $\frac{1}{3}$ di ottava e corrisponde ad una delle bande critiche individuabili sulla membrana basilare dell'orecchio(vedi capitolo 2). Per ridurre la quantità di dati da analizzare ogni banda critica viene poi sottocampionata con una frequenza di campionamento compresa tra 50Hz e 200Hz e viene calcolato il RMS[3]. La seconda fase prevede invece l'analisi dell'output del primo blocco tramite un Periodicity Detector in grado di rilevare le periodicità presenti nel segnale come la ricorrenza di accenti forti. Il metodo proposto da Klapuri [7] nel 2003 può essere visto come una rielaborazione dei metodi di Goto [5] e Sethares [6]. In questo metodo il segnale viene ripartito in 36 bande anziché in 23, come previsto nel lavoro di Sethares e vengono poi ricavate 4 macrobande dalla combinazione delle 36, dette Accent Band. Variazioni armoniche e melodiche vengono quindi rilevate grazie all'analisi delle 36 bande, mentre suoni impulsivi, come colpi di batteria, vengono individuati soltanto grazie all'osservazione delle 4 Accent band. Il sistema rileva contemporaneamente tutti e tre i livelli metrici precedentemente descritti attraverso un modello probabilistico delle loro relazioni ed evoluzioni temporali. Il modello grazie all'individuazione dei tre livelli metrici è quindi in grado di determinare la segnatura di tempo.

3.2.3 Estrazione della segnatura di tempo da formati compressi

Esistono due approcci relativi all'analisi dei segnali audio musicali compressi: un'indagine tradizionale, detta analisi indiretta e un'analisi diretta. Nel primo approccio (analisi indiretta) il segnale MP3 viene riportato in formato PCM e su questo vengono effettuate le varie operazioni di analisi quali Beat Tracking, Pitch Tracking, ecc; nel secondo approccio le analisi vengono invece effettuate direttamente sul file MP3 senza operare

[3] L'aronimo RMS sta per Root Mean Square e indica la radice della media dei quadrati definita come:

$$x_{\mathrm{RMS}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} = \sqrt{\frac{x_1^2 + x_2^2 + \cdots + x_N^2}{N}}$$

alcuna decompressione. In questo caso encoder e decoder sono utilizzati solo per la creazione e fruizione del file audio compresso e non per ottenere un file PCM intermedio sul quale compiere l'analisi. Questo metodo presenta notevoli vantaggi rispetto agli altri due, come ad esempio un minor impiego di memoria e di tempo necessario per operare la compressione e decompressione del file, ma comporta una maggior limitazione relativa alla quantità dei mezzi di indagine utilizzabili. Nelle figure 3.9 e 3.10 sono disponibili le schematizzazioni dei due diversi approcci relativi ai formati compressi.

Figura 3.9: Analisi indiretta di formati compressi.

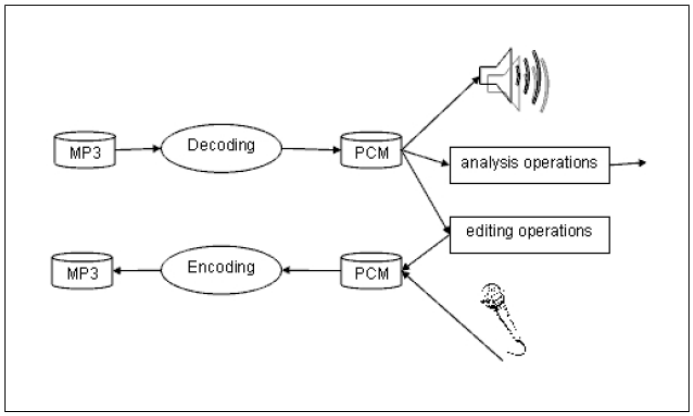
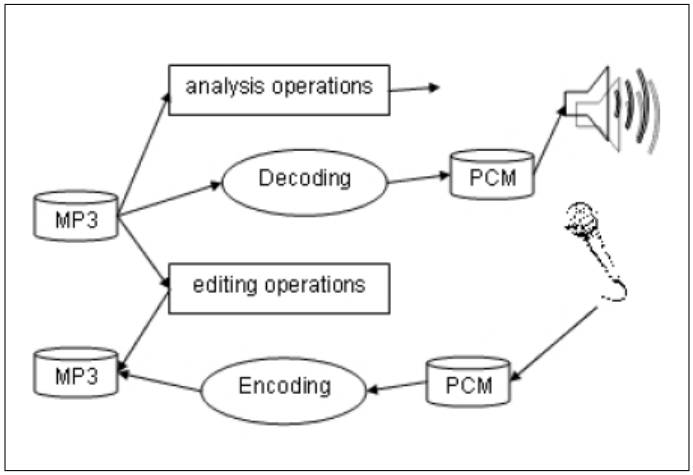


Figura 3.10: Analisi diretta di formati compressi.



Dalla ricerca bibliografica effettuata in questo lavoro è emerso che non sono stati ancora concepiti dei sistemi capaci di effettuare l'estrazione diretta dell'indicazione di tempo dall'audio in formato compresso. Esistono però alcuni algoritmi, dedicati al beat tracking e al tempo induction, in grado di operare sui formati compressi ottenendo risultati paragonabili a quelli ottenuti dagli algoritmi rivolti ai formati lineari. È necessario precisare oltretutto che la maggior parte dei lavori sul beat/tempo tracking in MP3 sono focalizzati sull'error concealment[4] e non sono finalizzati all'estrazione di dati dall'audio per compiere operazioni di music information retrival [34, 35, 36]. In questi lavori il WSP è sempre utilizzato per raffinare operazioni di analisi compiute sullo spettro del segnale MP3 [34, 35, 36]. In [34], ad esempio, il WSP è utilizzato per ottimizzare l'analisi compiuta sui coefficienti MDCT. Poiché questo algoritmo è studiato per la trasmissione di file MP3 su canali rumorosi l'intero costo computazionale è a carico del decoder, quindi il metodo è applicabile su qualsiasi MP3. Nel suo lavoro successivo, Wang, [36] propone un nuovo schema per l'error concealment basato su MPEG-AAC, che migliora il beat detector proposto in [34]. In [36] sono utilizzati solamente i coefficienti SDFT[5] al posto dei coefficienti MDCT e del WSP. Il metodo proposto in [36] lavora sia sul decoder che sull'encoder di conseguenza non è direttamente applicabile ad un file compresso ma vi è la necessità di comprimere i file con l'encoder appositamente modificato. Viceversa il WSP è utilizzato per la correzione dell'errore in [35]. In questo caso non si pone particolare attenzione all'estrazione dei beat dal brano audio perchè lo schema di correzione si basa sul riconoscimento degli istanti di non stazionarietà del segnale audio, nello specifico: se viene perso un frame contenente finestre di tipo Long l'errore viene recuperato ripetendo il frame precedente, se invece viene perso un frame contenente finestre Short si recupera l'errore ripetendo il precedente frame contenente finestre di tipo Short. Un lavoro che si pone a metà strada tra il beat tracking e l'mp3 è

[4]Recupero dell'errore. Sono quelle tecniche che vengono utilizzate, ad esempio, nel momento in cui si vuole trasmettere un MP3 in streaming, ad esempio su Internet, per recuperare eventuali frame persi.

[5]Shifted Discrete Fourier Transform

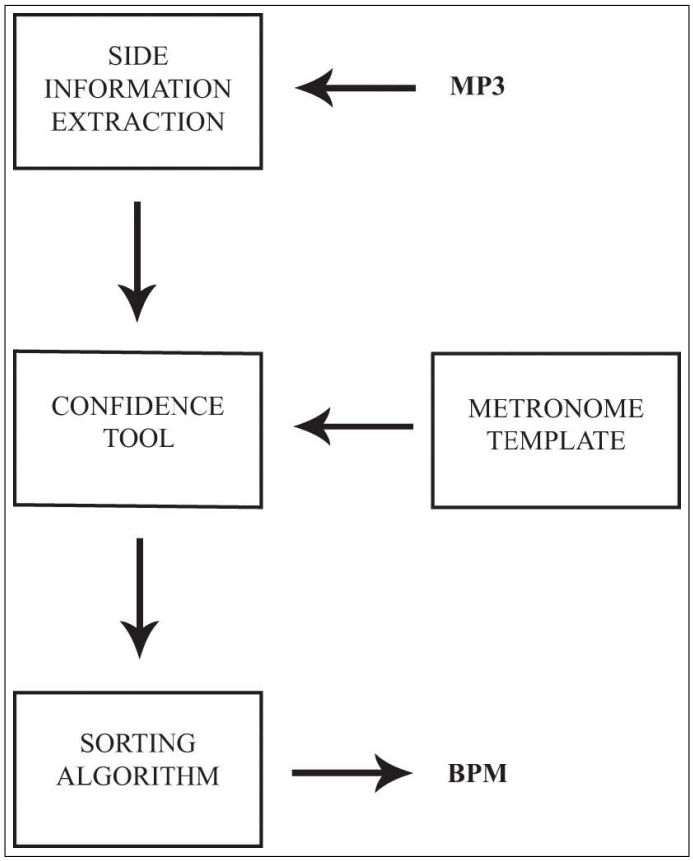
presentato in [37], in questo lavoro si cerca di riconoscere quando un file compresso in MP1[6] contiene musica e quando invece un segnale vocale sfruttando i diversi beat del segnale. In [38, 39] viene invece descritto un algoritmo capace di rilevare i BPM di un file audio in formato compresso considerando esclusivamente informazioni estrapolate dal WSP. In questo paragrafo verrà descritta la struttura generale di tale algoritmo in quanto questo viene utilizzato dal prototipo software sviluppato in questo lavoro di tesi per l'individuazione della velocità di metronomo relativa al brano in esame.

Algoritmo di tempo induction

L'algoritmo descritto in [38, 39] può essere definito come un operatore che rileva le similitudini tra vari treni di impulsi, rappresentanti dei vari tempi metronomici, e il window-switching pattern del brano MP3 preso in esame. Come mostrato dalla figura 3.11 l'algoritmo si compone di 4 blocchi principali a ognuno dei quali è assegnato un compito differente. Il primo blocco, detto Side Information Extraction, prende in input un file MP3 e restituisce al blocco successivo un vettore contenente il relativo WSP modificato in modo da contenere esclusivamente valori pari a 0 e 1. Questo blocco ha infatti il compito di estrarre dal MP3 tutte le side information e di mantenere solo il WSP che sarà poi processato con una funzione in grado di convertire tutti i valori diversi da 0 in 1 e di mantenere invariati i valori pari a 0. Il blocco Metronome Template ha invece il compito di costruire i vari template di metronomo che saranno poi confrontati con il WSP nel blocco Confidence Tool. Vengono quindi generati 560 template rappresentanti i tempi di metronomo compresi tra 40 e 600 BPM ognuno dei quali è un vettore contenente solo valori pari a 0 e 1, la cui lunghezza è equivalente al doppio del numero di frame nel brano MP3. In questo modo il WSP e ciascun template di metronomo possiedono uguali dimensioni e possono essere confrontati effettuando una moltiplicazione elemento per elemento tra i due vettori. Nel blocco Confidence Tool ognuno dei 560

[6]Con MP1 si intende un segnale audio compresso secondo lo schema MPEG layer 1

Figura 3.11: Diagramma a blocchi dell'algoritmo di tempo induction



template viene confrontato con il WSP e in base al numero di beat allineati tra i due vettori viene estrapolato un valore di "confidence". In realtà prima di determinare il valore di "confidence" definitivo per il template corrente, viene effettuata una sincronizzazione tra template e WSP tale per cui il numero dei beat allineati tra i due vettori sia il maggiore possibile. I valori ottenuti in questo blocco vengono immagazzinati in un vettore che costituirà l'input del blocco successivo. Il blocco Sorting Algorithm ordina il vettore in maniera crescente rispetto al numero di beat allineati. La posizione di ogni template del metronomo, ottenuta in questo vettore, viene confrontata con la sua posizione originaria che è equivalente al suo numero di BPM. Il template che si discosta maggiormente in positivo dalla sua posizione originaria rappresenta i BPM del brano.

Capitolo 4

Estrazione della segnatura di tempo con le sole informazioni ritmiche

Oltre ai vantaggi derivanti dall'analisi diretta, trattata nel paragrafo 3.2.3, e quindi dall'utilizzo del WSP, ci sono alcune problematiche relative all'MP3 in ambito MIR che inducono questo lavoro ad essere orientato all'utilizzo delle sole informazioni ritmiche. Queste problematiche verranno enunciate in dettaglio nel prossimo paragrafo.

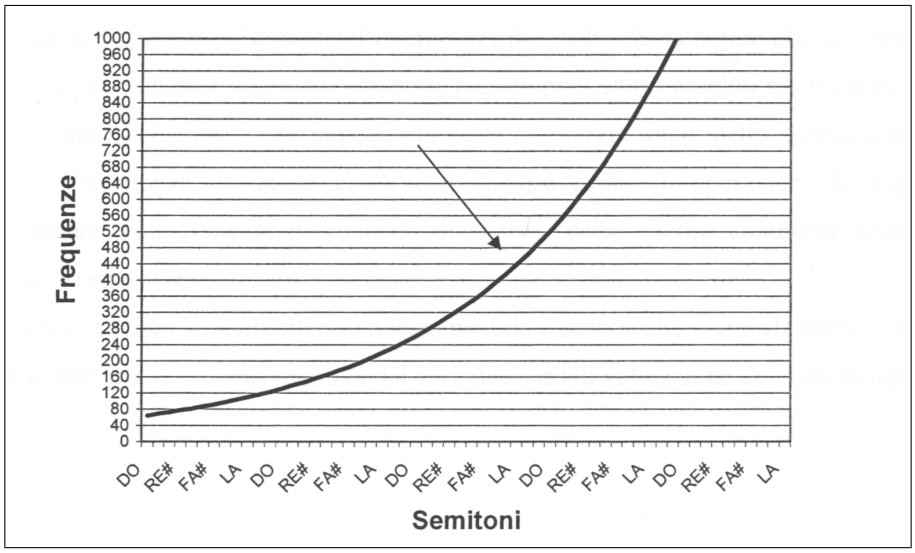
Una grande quantità di software dedicata all'analisi musicale cerca di imitare la percezione umana per individuare i parametri musicali oggetto dell'indagine; in alcuni casi test di ascolto sono stati fondamentali per comprendere i meccanismi percettivi e consentirne l'implementazione all'interno dei software. Un esempio di questa pratica si può trovare nel lavoro di Parncutt [3], dove i principi alla base dell'algoritmo sviluppato erano frutto dell'elaborazione dei risultati dei test di ascolto precedentemente effettuati. Sulla base di questa idea sono stati quindi allestiti dei test d'ascolto per verificare l'effettiva capacità, da parte di un essere umano, di estrapolare l'indicazione di tempo di un brano dalle sole informazioni fornite dal WSP. Grazie ai risultati dei test è quindi stato

possibile valutare se la strada dell'imitazione della percezione umana fosse percorribile o meno.

4.1 Problematiche relative all'MP3 in ambito MIR

Nei precedenti lavori [5][6][7][19], relativi all'individuazione della segnatura di tempo, sono state utilizzate, nella maggior parte dei casi, informazioni relative alle variazioni armoniche presenti nei brani, deducibili grazie ad apposite analisi frequenziali. Queste informazioni venivano estrapolate da file in formato PCM e utilizzate per la rilevazione degli accenti metrici che consentivano la determinazione della segnatura di tempo corrispondente. Purtroppo in formato compresso tale indagine non è attuabile proprio per la natura del file MP3. Infatti la scarsa risoluzione frequenziale dei coefficienti MDCT non consente una corretta individuazione delle note presenti nel segnale e quindi degli accordi da esse formati [40]. Per disambiguare le diverse note è necessario che queste appartengano a linee frequenziali diverse, questa condizione viene violata a partire dalle note subito dopo l'ottava centrale visto che i coefficienti MDCT hanno una risoluzione frequenziale di poco inferiore a 40 HZ. Questo appare evidente osservando la figura 4.1 la quale mostra la disposizione delle note nel dominio della frequenza. A causa dell'impossibilità di effettuare questa tipologia di indagine, in questo lavoro sono state prese in considerazione esclusivamente le informazioni ritmiche, deducibili grazie al WSP, relative ai brani musicali. In letteratura non esistono lavori incentrati sull'individuazione della segnatura di tempo sfruttando esclusivamente tali informazioni. Scopo di questo elaborato è quindi quello di verificare se il WSP contiene informazioni sufficienti per dedurre l'indicazione di tempo.

Figura 4.1: Ogni nota dovrebbe essere in una banda diversa, la freccia indica il La centrale.



4.2 Conversione del WSP in un file WAV

In questa sezione verrà descritto il funzionamento generale dell'algoritmo sviluppato per la creazione dei file WAV oggetto dei test di ascolto. Il codice del suddetto algoritmo è disponibile nell'appendice A.

Innanzitutto è necessario precisare che i file impiegati nella fase di test sono tre per ogni brano considerato e di questi solo due vengono generati dal suddetto algoritmo. Uno dei tre file infatti non è altro che l'MP3 originale, utilizzato in questa fase per verificare le competenze musicali dei partecipanti al test. I due file che costituiscono l'output del software rappresenteranno invece una versione audio del WSP.

In prima istanza il software prevede l'estrazione del WSP dal relativo file MP3 grazie ad una versione modificata del lame, che reperisce tutte le side information mantenendo esclusivamente il WSP; successivamente il WSP viene opportunamente modificato in modo da ottenere un vettore avente unicamente valori pari a 0 e 1. Queste due fasi sono del tutto analoghe a quelle descritte nel paragrafo 3.2.3 relativamente all'algoritmo di tempo induction con la differenza che la trasformazione del vettore viene fatta seguendo

un diverso procedimento. Il risultato di queste due fasi è quindi un vettore, chiamato MWSP[1], che presenta valori pari a 1 nelle posizioni in cui il WSP assume valori diversi da zero e valori pari a 0 nelle rimanenti posizioni. L'MWSP costituirà poi l'input di due diverse funzioni: l'algoritmo di tempo induction modificato e una funzione creata per rendere istantanei i transienti presenti nel file. L'algoritmo di tempo induction in questo caso ha due compiti, ovvero indicare i BPM relativi al file MP3 in questione ed estrapolare un vettore corrispondente al metronomo del brano musicale. Per espletare quest'ultimo compito l'algoritmo è stato modificato in modo tale da conferirgli la capacità di memorizzare la matrice di metronomi generata[2] e di selezionare il metronomo opportuno. L'output di questo blocco software sarà pertanto un vettore corrispondente al metronomo del brano considerato il quale, grazie alle operazioni svolte dall'algoritmo, sarà perfettamente sincronizzato con il brano stesso. L'altra funzione, il cui input è costituito dal MWSP, è chiamata elimina1consecutivi ed ha il compito di modificare ulteriormente il MWSP in modo da rendere tutti i transienti presenti nel file, istantanei. Come detto nel capitolo 2 ogni qual volta si è in presenza di un suono di elevata intensità, come colpi di batteria, l'encoder modifica il tipo di finestratura da Long a Short. Per fare questo vengono utilizzate due finestre di transizione, Long-to-Short e Short-to-Long indicizzate rispettivamente con i numeri 1 e 3. Ne consegue che ogni transiente è caratterizzato da almeno tre frame/granulo, ognuno dei quali, dopo la conversione dal WSP a MWSP, sarà indicizzato con il numero 1. Il compito della funzione è quindi quello di mantenere il primo valore 1 della serie e portare a 0 tutti gli altri. Questo viene effettuato per rendere significativa l'operazione imputata al blocco successivo, chiamato "Or dei due vettori", il quale ha il compito di ricavare un vettore generato dalla sovrapposizione del SMWSP[3] e del metronomo. Il vettore risultante da tale operazione, identificato

[1] Modified Window-switching Pattern

[2] Trattandosi di una funzione lo spazio di memoria utilizzato viene de allocato dopo l'esecuzione causando la perdita di tutte le variabili locali.

[3] l'acronimo SMWSP sta per Still Modified Window-switching Pattern e indica il WSP ulteriormente modificato

dall'acronimo MSMWSP, è quindi formato da un treno di impulsi (il metronomo) a cui sono aggiunti tutti i transienti (non allineati con esso) presenti nel SMWSP, cioè:

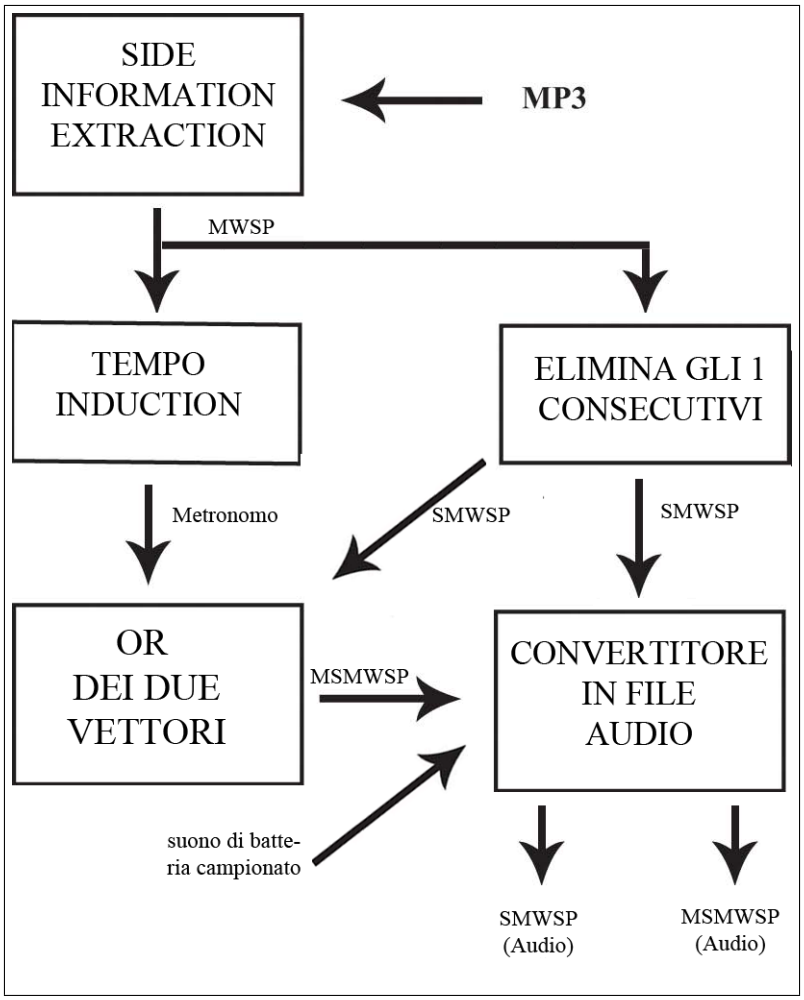
$$MSMWSP = SMWSP \vee Metronomo \tag{4.1}$$

L'ultimo blocco software costituisce il cuore dell'algoritmo in quanto svolge la conversione dei due vettori generati, SMWSP e MSMWSP, in file audio. Per operare tale conversione viene impiegato un file audio in formato PCM, contenente un singolo colpo di batteria, il quale è stato precedentemente registrato e reso disponibile al software. Il blocco quindi opera prima sul SMWSP e poi sul MSMWSP sostituendo ad ogni 0 presente nei vettori un silenzio digitale pari a 576 campioni e ad ogni 1 i primi 576 campioni del file audio considerato. In questo modo i file generati avranno la stessa durata dell'MP3 di partenza e, grazie ad opportuni software audio che consentono di gestire più traccie audio contemporaneamente, potranno essere ascoltati in sincrono con quest'ultimo. Quest'operazione esula dallo scopo dei test ma è utile per capire quali transienti siano stati effettivamente rilevati dall'encoder. La figura 4.2 mostra il diagramma a blocchi relativo all'algoritmo descritto in questo paragrafo.

4.3 Test effettuati

Questa fase consiste nell'effettuare dei test di ascolto nei quali verrà fornita, da parte dei partecipanti al test, l'indicazione di tempo relativa ai file audio ottenuti tramite il procedimento precedentemente descritto. I soggetti dell'esperimento sono stati selezioneati tra individui aventi competenze musicali che consentano loro di svolgere il test proficuamente. È stata quindi prevista una fase di valutazione delle competenze musicali dei soggetti partecipanti al test, nella quale vengono riprodotti degli MP3 opportunamente selezionati e viene chiesto di indicare per ciascuno di questi la relativa segnatura di

Figura 4.2: Diagramma a blocchi dell'algoritmo per la genesi dei file impiegati nei test.



tempo. In questo modo sarà possibile valutare le risposte fornite nella successiva fase del test anche in funzione della preparazione musicale di ogni singolo partecipante. La selezione dei brani è stata effettuata in modo da ottenere una varietà di indicazioni temporali differenti. Infatti sono stati scelti 4 brani in $\frac{3}{4}$, 5 in $\frac{4}{4}$, 3 in $\frac{5}{4}$ ed un solo brano in $\frac{7}{4}$ per un totale di 13 brani, come si evince dalla tabella 4.1. Il criterio di scelta dei brani non è legato esclusivamente alla segnatura di tempo ma anche ad altri due fattori. I brani devono essere tali per cui sia significativa l'estrazione dei BPM, a tal proposito non sono stati presi in considerazioni i brani di musica classica in cui la pulsazione sotto-

Tabella 4.1: Brani utilizzati nella fase di test.

Titolo	**Autore**	**Album**	**Genere**	**BPM**	**Segnatura di tempo**
Mama D.	Mr.Big	Hey Man	Rock	136	$\frac{3}{4}$
Spoiled	Joss Stone	Mind Body & Soul	Pop	110	$\frac{3}{4}$
Manic Depression	Jimi Hendrix	Are You Experienced	Pop	153	$\frac{3}{4}$
Elettro	Anonimo	Sconosciuto	Pop	170	$\frac{3}{4}$
Elbow Grease	Niacin	Time Crunch	Funky	142	$\frac{4}{4}$
Time Crunch	Niacin	Time Crunch	Funky	140	$\frac{4}{4}$
Electrified	Mr.Big	Get Over It	Rock	150	$\frac{4}{4}$
A rose alone	Mr.Big	Get Over It	Rock	130	$\frac{4}{4}$
Dancin' with my devils	Mr.Big	Get Over It	Rock	165	$\frac{4}{4}$
Echo	Joe Satriani	Surfing With The Alien	Rock	137	$\frac{5}{4}$
Trundrumbalind	Joe Satriani	Crystal Planet	Rock	144	$\frac{5}{4}$
Take Five	Dave Brubeck	Time Out	Jazz	129	$\frac{5}{4}$
Discoteca Labirinto	Subsonica	Subsonica	Pop	128	$\frac{7}{4}$

stante rallenta e accelera a seconda dell'interpretazione dei musicisti, e sono stati scelti solo brani in cui la rilevazione dei BPM, con l'algoritmo descritto nel paragrafo 3.2.3, risulti corretta. I brani selezionati appartengono a generi diversi con una predilezione per il rock e il pop, dove l'algoritmo di tempo induction ottiene i risultati migliori. Il motivo per cui vengono impiegati i criteri di selezione sopra enunciati è la necessità di ottenere, come output del blocco Tempo Induction dell'algoritmo descritto al paragrafo precedente, un vettore metronomo rappresentativo dei BPM del brano in esame. Infatti, perché l'operazione compiuta dal blocco "Or dei due vettori" sia significativa è necessario che il brano non presenti variazioni metriche rilevanti e che il vettore metronomo ricavato abbia una frequenza di beat equivalente a quella del brano considerato. Lo scopo dei test consiste quindi nel verificare se le informazioni relative all'indicazione di tempo dei brani vengono conservate anche nel WSP e se queste siano percepibili tramite test di ascolto. Infatti l'ascoltatore dovrà cercare di rilevare, nei file audio generati, quei modelli che consentono di determinare la segnatura di tempo. Durante lo svolgimento dei test è stato anche chiesto ai partecipanti di rilevare la frequenza di beat per ogni file audio esaminato. Sapendo a priori l'attuabilità della determinazione dei BPM tramite software, utilizzando esclusivamente informazioni carpite dal WSP, si è voluto constatare se un essere umano fosse in grado di arrivare alle stesse conclusioni.

4.3.1 Svolgimento dei Test

Ogni partecipante è stato indirizzato verso una postazione opportunamente attrezzata per la svolgimento del test. La postazione era dotata di un PC provvisto di: un paio di cuffie, un player audio per l'ascolto dei file oggetto del test e un metronomo software necessario per la rilevazione dei BPM. Sul PC era anche disponibile un foglio elettronico utilizzato dall'utente per memorizzare man mano i risultati del test. In primo luogo sono stati presentati i 13 brani indicati nella tabella 4.1 ed è stato chiesto di rilevare i BPM e di indicare la rispettiva segnatura di tempo. Questa fase, come detto precedentemente, era volta alla valutazione delle competenze musicali di ogni singolo partecipante. Successivamente sono stati proposti all'ascoltatore i file audio generati dal WSP in ordine casuale ed anche qui è stato chiesto di indicare BPM e segnatura di tempo. Per evitare che il partecipante potesse sfruttare informazioni derivanti dai nomi dei file per mettere in relazione i brani originali e i file audio derivati da questi, sono stati utilizzati degli identificatori numerici al posto dei nomi. Gli identificatori numerici oltre a rappresentare una traccia audio ne indicavano anche l'ordine di ascolto, per cui i numeri dall'uno al tredici identificavano i brani originali mentre i numeri dal 14 al 39 corrispondevano ai file audio generati dal software descritto nel paragrafo 4.2. La corrispondenza tra gli identificatori numerici e gli effettivi nomi dei file è disponibile nella tabella 4.2.

Il foglio elettronico utilizzato in questa fase, oltre a presentare delle sezioni dedicate alla registrazione dei dati anagrafici dei partecipanti, dispone anche di una tabella a tre colonne, ognuna delle quali è predisposta per contenere una tipologia di dati differente. Nella prima colonna sono disposti gli identificatori numerici in ordine crescente mentre nelle altre due sono presenti degli spazi bianchi necessari per l'inserimento dei BPM e della segnatura di tempo relativa ad ogni file audio.

Tabella 4.2: File audio utilizzati nel test disposti secondo l'ordine di ascolto

ID	Nome del file audio	Formato del file	BPM	segnatura di tempo
1	Electrified	MP3	150	$\frac{4}{4}$
2	Spoiled	MP3	110	$\frac{3}{4}$
3	Trundrumbalind	MP3	144	$\frac{5}{4}$
4	DiscotecaLabirinto	MP3	128	$\frac{7}{4}$
5	Take Five	MP3	129	$\frac{5}{4}$
6	Elbow Grease	MP3	142	$\frac{4}{4}$
7	Manic Depression	MP3	153	$\frac{3}{4}$
8	Echo	MP3	137	$\frac{5}{4}$
9	Dancin' With My Devils	MP3	165	$\frac{4}{4}$
10	Mama D.	MP3	136	$\frac{3}{4}$
11	A Rose Alone	MP3	130	$\frac{4}{4}$
12	Elettro	MP3	170	$\frac{3}{4}$
13	Time Crunch	MP3	140	$\frac{4}{4}$
14	SMWSP A Rose Alone	WAV	130	$\frac{4}{4}$
15	SMWSP Dancin' With My Devils	WAV	165	$\frac{4}{4}$
16	SMWSP DiscotecaLabirinto	WAV	128	$\frac{7}{4}$
17	SMWSP Echo	WAV	137	$\frac{5}{4}$
18	SMWSP Elbow Grease	WAV	142	$\frac{4}{4}$
19	SMWSP Electrified	WAV	150	$\frac{4}{4}$
20	SMWSP Elettro	WAV	170	$\frac{3}{4}$
21	SMWSP Mama D	WAV	136	$\frac{3}{4}$
22	SMWSP Manic Depression	WAV	153	$\frac{3}{4}$
23	SMWSP Spoiled	WAV	110	$\frac{3}{4}$
24	SMWSP Take Five	WAV	129	$\frac{5}{4}$
25	SMWSP Time Crunch	WAV	140	$\frac{4}{4}$
26	SMWSP Trundrumbalind	WAV	144	$\frac{5}{4}$
27	MSMWSP A Rose Alone	WAV	130	$\frac{4}{4}$
28	MSMWSP Dancin' With My Devils	WAV	165	$\frac{4}{4}$
29	MSMWSP DiscotecaLabirinto	WAV	128	$\frac{7}{4}$
30	MSMWSP Echo	WAV	137	$\frac{5}{4}$
31	MSMWSP Elbow Grease	WAV	142	$\frac{4}{4}$
32	MSMWSP Electrified	WAV	150	$\frac{4}{4}$
33	MSMWSP Elettro	WAV	170	$\frac{3}{4}$
34	MSMWSP Mama D	WAV	136	$\frac{3}{4}$
35	MSMWSP Manic Depression	WAV	153	$\frac{3}{4}$
36	MSMWSP Spoiled	WAV	110	$\frac{3}{4}$
37	MSMWSP Take Five	WAV	129	$\frac{5}{4}$
38	MSMWSP Time Crunch	WAV	140	$\frac{4}{4}$
39	MSMWSP Trundrumbalind	WAV	144	$\frac{5}{4}$

4.3.2 Risultati dei test

Ogni risposta fornita dal partecipante è stata valutata tramite un punteggio; ad ogni risposta esatta è stato assegnato il valore 1 mentre lo 0 è stato attribuito ad ogni risposta errata. Oltre a questi due valori è stato utilizzato un valore intermedio di 0,5 quando la risposta fornita dall'ascoltatore non era corretta ma comunque coerente con la risposta esatta; ad esempio nella rilevazione dei BPM è stato assegnato il valore 0,5 ogni qual volta la frequenza di beat rilevata dall'utente fosse il doppio, la metà, un terzo o il triplo della frequenza reale. Vista l'eterogeneità dei file audio oggetto del test e la diversa tipologia di rilevazione effettuata dai partecipanti, i risultati ottenuti sono stati messi in relazione tra di loro in base a questi due criteri. Sono stati creati 6 blocchi di risultati ognuno dei quali fà riferimento ad una certa tipologia di file audio (brani originali o SMWSP o MSMWSP) e ad una particolare ricerca di parametri musicali (BPM o segnatura di tempo). L'esito di ogni blocco è ottenuto tramite la somma dei singoli valori in esso contenuti, per cui il massimo risultato ottenibile è 13, situazione che si verifica quando l'ascoltatore non ha commesso alcun errore o imprecisione nella rilevazione del parametro musicale ricercato. Infine sono stati messi in relazione tra di loro i risultati ottenuti dai vari partecipanti al test tramite il calcolo della media, moda, mediana e della deviazione standard. Nella tabella 4.3 vengono riportati i risultati dei test e i valori di media, moda, mediana e deviazione standard relativi a questi.

4.3.3 Valutazione dei risultati

Dall'analisi comparata dei risultati si possono trarre diverse conclusioni; innanzitutto la ricerca della segnatura di tempo è molto più complessa rispetto all'analisi dei BPM e richiede delle competenze musicali più evolute. Infatti, mentre la quasi totalità dei partecipanti è riuscita a percepire correttamente i BPM negli MP3 originali solo una

Tabella 4.3: Risultati dei test

Numero Test	**Risultati brani originali BPM**	**Risultati SMWSP BPM**	**Risultati SMWSP BPM**	**Risultati brani originali ST**	**Risultati SMWSP ST**	**Risultati MSMWSP ST**
1	13	10,5	13	10	4,5	3
2	4,5	4	11	7	5	3
3	13	12	13	13	6	4
4	13	9,5	12	13	4	5
5	13	10	13	8,5	4	4
6	13	13	13	12	5	4
7	13	10	13	13	4,5	5
8	13	11	12	12	5	3
9	13	11	12	12	5	3
10	13	11,5	13	12	4	3,5
Media	12,15 (93,46%)	10,25 (78,85%)	12,6 (96,92%)	11,15 (85,77%)	4,6 (35,38%)	3,85 (29,62%)
Mediana	13 (100%)	10,75 (82,69%)	13 (100%)	12 (92,31%)	4,5 (34,62%)	4 (30,77%)
Moda	13 (100%)	10 (76,92%)	13 (100%)	12 (92,31%)	4 (30,77%)	4 (30,77%)
Deviazione standard	2,69 (22,12%)	2,43 (23,7%)	0,7 (5,55%)	2,06 (18,43%)	0,66 (14,3%)	0,75 (19,41%)

piccola parte è riuscita ad ottenere il punteggio massimo nella rilevazione della segnatura di tempo. Comunque circa l'85% delle risposte relative all'indicazione di tempo nei brani originali era corretta il che ha confermato la presenza di competenze musicali evolute nei vari partecipanti al test. Un'altra considerazione è relativa alla capacità di un essere umano di rilevare i BPM di un brano utilizzando esclusivamente l'informazione contenuta nel WSP. La fase del test volta a dimostrare tale capacità era quella in cui si chiedeva all'ascoltatore di rilevare i BPM ascoltando i file SMWSP. Più del 78% delle risposte fornite in questa fase sono risultate corrette e in alcuni casi l'ascoltatore è riuscito ad individuare la frequenza di beat relativa ad ogni brano senza commettere alcun errore. Si può quindi affermare che il WSP, nella maggior parte dei casi, presenta un contenuto informativo sufficiente, per la rilevazione della frequenza di beat da parte di un essere umano. Lo stesso non si può dire per la rilevazione della segnatura di tempo, infatti i risultati sperimentali relativi alla ricerca di tale parametro suggeriscono l'impossibilità da parte di un essere umano di rilevare l'indicazione di tempo utilizzando esclusivamente le informazioni presenti nel WSP. Come detto precedentemente nell'ultima fase del test sono state presentate due diverse tipologie di file audio ovvero SMWSP e MSMWSP chiedendo all'ascoltatore di indicare la segnatura di tempo corri-

spondente a ciascun file. Le risposte corrette relative alla prima tipologia di file audio sono in media il 35% il che dimostra una scarsa capacità di individuare la segnatura di tempo considerando esclusivamente il WSP. È importante notare come nel secondo test riportato nella tabella 4.3, quello compilato dal soggetto con meno competenze musicali, i risultati relativi alla segnatura di tempo siano molto simili a quelli degli altri partecipanti, denotando come una certa casualità governi la correttezza delle risposte relative a tale parametro. Infatti osservando la tabella si può notare come nelle colonne, *"Risultati brani originali BPM"* e *"Risultati brani originali ST"* sia il secondo test a determinare l'elevato valore della deviazione standard mentre nella colonna *"Risultati SMWSP ST"* questo non influisce particolarmente su tale valore. Per quanto riguarda la seconda tipologia di file, quella generata inglobando il metronomo al SMWSP, le risposte corrette non sono maggiori. Uno dei motivi che ha portato alla creazione di questi file era il pensiero che aggiungendo l'informazione metronomica relativa al WSP considerato fosse più facile individuare la segnatura di tempo. Questo si è dimostrato falso in quanto non solo la percentuale dei risultati corretti è scesa ulteriormente ma il partecipante guidato dall'informazione metronomica è stato indotto a rilevare nella maggior parte dei casi una segnatura di tempo di $\frac{4}{4}$. Come conclusione di questo capitolo si può qundi asserire che le sole informazioni ritmiche non sono sufficienti per l'individuazione dell'indicazione di tempo da parte di un essere umano, di conseguenza non è utile imitare la percezione umana per elaborare un software in grado di rilevare automaticamente la segnatura di tempo da un MP3.

Capitolo 5

Algoritmo sviluppato e relativi test effettuati

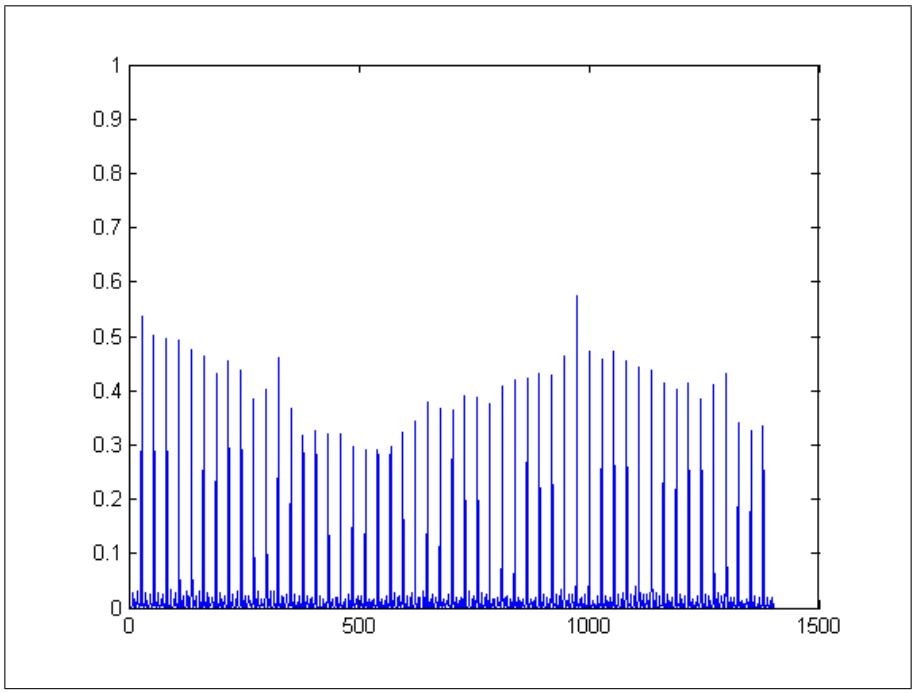
In questo capitolo verranno descritte le diverse strade intraprese per la creazione di un algoritmo in grado di estrarre l'indicazione di tempo da un file MP3. In primo luogo si è voluto applicare la teoria di Brown [4], basata sull'autocorrelazione, per constatare se questo metodo potesse essere utilizzato anche con i formati audio compressi; successivamente, vista l'inapplicabilità del metodo sopra citato è stata intrapresa una differente strada che ha portato allo sviluppo dell'algoritmo descritto in dettaglio nel paragrafo 5.2. L'applicazione della teoria di Brown ha comunque portato a risultati interessanti in particolar modo nell'ambito della rilevazione dei BPM, per cui uno degli sviluppi futuri di questa tesi potrebbe essere legato all'individuazione della frequenza di beat tramite il metodo dell'autocorrelazione.

5.1 Applicazione dell'algoritmo di Brown nei formati compressi

Come primo approccio si è quindi cercato di applicare la teoria di Brown anche ai formati compressi utilizzando il WSP, opportunamente trattato, come input dell'algoritmo. Una delle motivazioni che ha indotto all'applicazione di tale teoria è la somiglianza tra i file che costituiscono l'input dell'algoritmo originale e il WSP. Come già asserito nel capitolo 3, i file su cui lavora il software di Brown sono caratterizzati da ampiezze pesate corrispondenti all'inizio delle note segnate in partitura e da una serie di zeri interposti tra un'ampiezza e l'altra; questa morfologia ricorda il WSP con la differenza che in quest'ultimo non vengono prese in considerazione le note del brano in questione ma bensì i transienti rilevati al momento della compressione di un file audio. Oltretutto i transienti nel WSP non sono istantanei, come le ampiezze considerate nella versione originale dell'algoritmo, ma hanno una durata deducibile dal numero di Short-Block utilizzati per la codifica del transiente in questione. Per ottenere una maggiore somiglianza tra le due tipologie di file, il WSP viene quindi modificato tramite un blocco software che ha il compito di pesare i vari transienti in funzione della loro durata e di renderli istantanei. Oltre alla questione della somiglianza tra tipologie di file processati, un'altra ragione ha portato all'impiego del metodo dell'autocorrelazione, ovvero il fatto che le informazioni relative alla frequenza del brano considerato non fossero utilizzate per la rilevazione della segnatura di tempo. Come detto nel paragrafo 4.1, in MP3 si è vincolati alla risoluzione frequenziale della MDCT il che comporta difficoltà nell'impiegare metodi di indagine basati sull'utilizzo delle informazioni frequenziali [40]; per tale ragione il metodo di Brown sembra essere il più adatto, tra quelli visti fin ora, ad essere applicato nell'audio in formato compresso. L'algoritmo sviluppato prevede quindi il calcolo dell'autocorrelazione del WSP. Analogamente al software descritto nel paragrafo 4.2 anche in questo caso viene utilizzato il blocco Side Information Extraction per

reperire il WSP dal file MP3 in esame e convertirlo in un vettore composto esclusivamente da valori pari a 0 e 1. Visto che i transienti presenti nel MWSP risultante non sono istantanei verrà utilizzato anche qui il blocco elimina1consecutivi ma non prima di aver analizzato la durata di ciascun transiente ed averlo pesato in funzione di questa. Tale operazione viene effettuata semplicemente contando il numero di 1 consecutivi che costituiscono un transiente ed assegnando il valore risultante da tale conteggio al primo 1. Dopo di che tutti i valori successivi al primo, di ogni transiente, vengono portati a 0 tramite il blocco elimina1consecutivi. Queste operazioni vengono effettuate proprio per rendere più simile il WSP all'input originale dell'algoritmo di Brown dove ogni nota presente in partitura veniva rappresentata da un ampiezza istantanea il cui valore dipendeva dalla durata della nota in questione (per maggiori dettagli sull'algoritmo di Brown si faccia riferimento al capitolo 3). Una volta ottenuto il WSP così modificato ne viene calcolata l'autocorrelazione e vengono quindi analizzati i picchi della funzione. Anche in questo caso il valore massimo di m (vedi formula 3.1) viene scelto in modo da considerare diverse misure nel calcolo della funzione, pertanto viene impostato a 1400. In questo modo anche con il tempo di metronomo più lento, ovvero 40 BPM, vengono prese in considerazione almeno due misure visto che l'IOI relativo a tale velocità metronomica presenta un ampiezza di 114 frame/granulo. Una rappresentazione grafica della funzione applicata ad un brano in 3/4 è disponibile nella figura 5.1. Secondo la teoria di Brown la distanza tra i vari picchi nella funzione dovrebbe rappresentare l'IOI, necessario per individuare la frequenza di beat, mentre il picco più alto dovrebbe consentire l'individuazione del numero di pulsazioni presenti in una misura. Nella maggior parte dei casi presi in considerazione in questo lavoro la determinazione dell'IOI tramite il calcolo della distanza tra i picchi si è dimostrato corretto; infatti tramite la formula 5.1 è stato possibile calcolare i BPM relativi al brano in esame e verificare se tale valore

Figura 5.1: Autocorrelazione del WSP relativo a un brano in $\frac{3}{4}$.



corrispondesse a quello reale.

$$BPM = \frac{60}{IOI * 0,0131} \tag{5.1}$$

Nell'esempio della figura 5.1 il valore dell'IOI è di 27 frame/granulo il che comporta una frequenza di beat al minuto pari a 169. Il brano considerato nell'esempio è uno di quelli utilizzati durante la fase di test per cui consultando la tabella 4.1 si può facilmente constatare che il valore di BPM rilevato corrisponde a quello reale. Purtroppo la rilevazione della segnatura di tempo non è altrettanto corretta infatti i risultati sperimentali hanno mostrato una tendenza nella rilevazione di misure estremamente ampie. Anche nel caso specifico dell'esempio mostrato in figura 5.1 la misura viene rilevata come composta da 36 pulsazioni. Si può quindi affermare che l'algoritmo creato sfruttando la teoria di Brown non è in grado di estrarre la segnatura di tempo da un MP3 ma può essere utilizzato per comprendere la frequenza di beat del brano in esame.

5.2 Algoritmo per la rilevazione della segnatura di tempo tramite template pesati

Una delle strade intraprese in questo lavoro, per la creazione di un software in grado di determinare l'indicazione di tempo da un MP3, è quella di ricercare gli accenti metrici all'interno del WSP sfruttando l'informazione metronomica fornita dall'algoritmo descritto nel paragrafo 3.2.3. In questo paragrafo verrà quindi presentato l'algoritmo sviluppato seguendo questa strada, il cui codice è disponibile nell'appendice B.
Come detto precedentemente gli accenti metrici forti si collocano all'inizio delle misure definendo il numero di pulsazioni presenti in esse. Nella maggior parte dei casi un musicista tende ad enfatizzare tali accenti durante l'esecuzione per cui con molta probabilità le note con intensità più elevata ricorreranno all'inizio di ogni misura. È quindi plausibile pensare che durante la compressione di un file audio musicale l'encoder tenda a rilevare con maggiore probabilità gli accenti collocati in concomitanza con l'inizio delle misure. Per tale ragione il software sviluppato ricerca con quale periodicità si presentano i transienti del WSP sincronizzati con il metronomo relativo al brano in esame, ricavato grazie all'algoritmo di tempo induction opportunamente modificato, nel tentativo di individuare l'inizio di ogni misura. In prima istanza il software prevede l'utilizzo dei blocchi Side Information Extraction e Time Induction Algorithm, già descritti nei capitoli precedenti, ottenendo così la versione modificata del WSP, ovvero il MWSP, e il vettore metronomo sincronizzato con questo. Dopodiché dal metronomo viene ricavata una matrice composta da undici colonne ognuna delle quali corrisponde al vettore metronomo modificato in modo da conferirgli la capacità di promuovere determinati accenti piuttosto che altri. Per creare tale matrice vengono impiegati undici microtemplate ognuno dei quali è caratterizzato da una dimensione fissa e da una serie di valori prestabiliti. La dimensione del microtemplate, ovvero il numero di elementi in esso contenuti, indica la suddivisione metrica rappresentata dal microtemplate in questione, ad esempio

il tempo ternario viene rappresentato da tre elementi ognuno dei quali corrisponde ad un movimento della misura. I valori assegnati ad ogni elemento, invece, identificano i vari tipi di accenti metrici (forte, mezzo forte e debole). Agli accenti deboli vengono assegnati valori pari a 0 mentre alle altre due tipologie di accento viene assegnato un valore diverso da 0 dipendente dal tipo di suddivisione metrica considerata. La definizione dei microtemplate viene effettuata tramite le seguenti righe di codice:

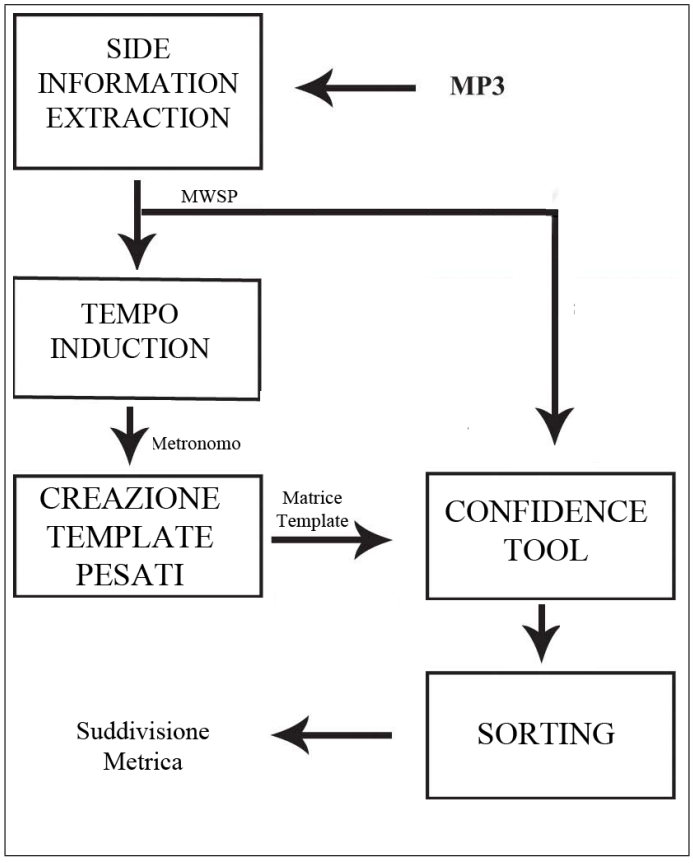
```
TempBin   = [4;0];
TempTern  = [6;0;0];
TempQuat1 = [6;0;2;0];
TempQuat2 = [8;0;0;0];
TempQuin1 = [10;0;0;0;0];
TempQuin2 = [4;0;0;1;0];
TempQuin3 = [1;0;0;4;0];
TempSen   = [12;0;0;0;0;0];
TempSet1  = [13;0;0;0;0;0;0];
TempSet2  = [6;0;0;0;1;0;0];
TempSet3  = [1;0;0;0;6;0;0];
```

Per quanto riguarda l'assegnamento dei valori è necessario specificare che agli accenti forti viene assegnato un valore tanto più elevato quanto minore è la frequenza con cui questo si ripeterà una volta che il microtemplate sarà combinato con il metronomo. Nel caso in cui sia presente anche un accento mezzo forte, come ad esempio nel secondo microtemplate relativo al tempo quinario (`TempQuin2`), l'accento forte presenterà un valore inferiore rispetto al valore assunto nel microtemplate relativo alla stessa scansione metrica ma privo di accenti mezzoforti. Per semplicità, nella costruzione dei microtemplate sono state prese in considerazione solo le suddivisioni metriche più utilizzate in musica, ovvero le suddivisioni dalla binaria alla settenaria, e non è stata considerata

l'eventualità che un brano possa presentare variazioni metriche al suo interno. Grazie ai microtemplate e al metronomo è quindi possibile generare una matrice composta da undici vettori ognuno dei quali presenterà le caratteristiche proprie di ciascun microtemplate. Infatti il primo vettore della serie viene creato sostituendo ad ogni transiente del metronomo i valori contenuti nel primo microtemplate (`TempBin`). Per la precisione al primo transiente viene sostituito il primo valore al secondo il secondo fino a quando i valori del template si esauriscono, alche si sostituiscono i valori ripartendo dal primo. Gli altri vettori vengono generati utilizzando la medesima procedura sfruttando però gli altri microtemplate. La matrice così ottenuta costituirà l'input del blocco "confidence tool" nel quale ogni vettore viene confrontato con il MWSP per verificare con quale periodicità si ripresentano i transienti sincronizzati al metronomo. In questa fase viene quindi effettuata una moltiplicazione elemento per elemento tra il vettore corrente e il MWSP e viene calcolato un valore di confidence relativo a tale vettore. Grazie alla moltiplicazione tutti i transienti presenti nel MWSP non sincronizzati con gli accenti forti o mezzoforti del vettore corrente vengono portati a 0; in questo modo nel calcolo del valore di confidence vengono tenuti in considerazione esclusivamente quei transienti allineati con gli accenti forti o mezzoforti. Il valore di confidence si ottiene semplicemente sommando tra loro gli elementi del vettore risultante dalla moltiplicazione; tale valore viene memorizzato in attesa di essere comparato con quelli relativi agli altri vettori della matrice. In realtà per rendere più efficace il confronto tra i vettori e il MWSP, e il relativo calcolo dei valori di confidence si è tenuta in considerazione l'eventualità che il primo transiente del metronomo potesse non essere quello collocato all'inizio della prima misura. Infatti se si verificasse tale eventualità il template generato dalla combinazione del metronomo e del microtemplate, corrispondente all'effettiva scansione metrica del brano, risulterebbe disallineato con la corretta disposizione degli accenti. Per ovviare a tale problema il template corrente viene fatto ruotare $n-1$ volte dove n rappresenta il numero di elementi contenuti nel rispettivo microtemplate. Ad

ogni rotazione viene calcolato il valore di confidence ottenendo così *n* valori. Tra questi viene quindi selezionato come valore rappresentativo del template corrente quello più elevato. L'output del blocco "confidence tool" è pertanto un vettore contenente i valori di confidence relativi ai vari template fatti ruotare in modo da ottenere il valore più elevato possibile. Tale output viene poi elaborato dall'ultimo blocco del software, detto Sorting, che ha il compito di ordinare i valori di confidence dal più grande al più piccolo, di selezionare il primo di questi e di comprendere a quale template quest'ultimo faccia riferimento. Una volta individuato il template viene riportato il tipo di scansione metrica corrispondente. Il diagramma a blocchi dell'algoritmo è rappresentato nella figura 5.2. Va precisato che con questo metodo non si cerca di disambiguare la figura musica-

Figura 5.2: Diagramma a blocchi dell'algoritmo sviluppato.



le corrispondente al beat ma viene semplicemente ricercata la suddivisione metrica del brano in esame. Per cui l'output del software non è costituito da una frazione metrica ma dall'indicazione relativa al numero di movimenti presenti in una misura. Grazie all'algoritmo di tempo induction è possibile ottenere informazioni relative alla frequenza e al posizionamento dei beat ma non viene fornita alcuna informazione relativa alla figura musicale associata; l'algoritmo sviluppato prende quindi in considerazione il beat senza cercare di scoprire la figura musicale corrispondente.

5.3 Test effettuati

In questo paragrafo verranno descritte le caratteristiche dei test effettuati per valutare le performance dei due software sviluppati e verrà presentata l'analisi dei risultati ottenuti nei due casi.

5.3.1 Test relativi al software modellato sull'idea di Brown

L'unico metodo possibile per testare un algoritmo del genere consiste nella valutazione dei risultati ottenuti dalla sua esecuzione, per cui sono stati scelti diversi brani, differenti per genere e segnatura di tempo, sui quali è stato applicato l'algoritmo. Tra i brani selezionati sono presenti anche quelli utilizzati nella fase di test riportata nel capitolo 4. È importante specificare che la maggior parte dei brani utilizzati per valutare il funzionamento dell'algoritmo sviluppato sono in $\frac{4}{4}$ visto che tale segnatura di tempo è la più utilizzata in musica. Per eseguire il test occorre anzitutto trovare la segnatura di tempo corrispondente al brano in esame per poi metterla a confronto con il risultato presentato dall'algoritmo. È quindi necessario che il tester sia in possesso di una competenza musicale sufficiente a consentirgli di individuare correttamente la segnatura di tempo corrispondente al brano in esame; per cui tale test può venir effettuato solo da persone

con una preparazione musicale adeguata, a differenza del test comunemente impiegato per valutare le performance degli algoritmi utilizzati per la rilevazione dei BPM. Durante lo svolgimento del test sono state quindi ricavate le segnature di tempo relative ad ogni brano e, successivamente, sono state confrontate con i risultati forniti dall'algoritmo assegnando un punteggio pari a 1 ogni qual volta si verificasse una corrispondenza e uno pari a 0 nel caso contrario. Dopo di che è stata calcolata la media relativa ai valori ottenuti. Nella tabella 5.1 sono presentati i risultati generali dell'algoritmo. Analizzando

Tabella 5.1: Risultati generali dell'algoritmo sviluppato seguendo la teoria di Brown.

Numero di brani	**Corretti**	**Corretti considerando i multipli**	**Errati**
20	15%	50%	85%

i risultati del test si possono effettuare diverse osservazioni: innanzitutto il metodo di individuare la suddivisione metrica attraverso la localizzazione del picco più alto nella funzione di autocorrelazione, ottenuta dall'elaborazione del WSP attraverso l'algoritmo creato, si è dimostrato poco accurato, infatti soltanto nel 15% dei casi sono stati ottenuti risultati corretti. Tuttavia da un'analisi più accurata si può notare una tendenza nella rilevazione, da parte dell'algoritmo, di misure estremamente ampie ma comunque multiple della scansione metrica reale. Nel caso in cui tali rilevazioni vengano ritenute corrette la percentuale dei casi in cui l'algoritmo ottiene una valutazione positiva arriverebbe al 50%. Si potrebbe quindi pensare, in futuro, di modificare l'algoritmo in modo da conferirgli la capacità di risalire alla scansione metrica originale utilizzando come punto di partenza queste rilevazioni. Apportando tali modifiche si potrebbe quindi arrivare ad ottenere un software capace di determinare la suddivisione metrica corretta in una buona percentuale di casi concreti.

5.3.2 Test relativi al software dei template pesati

I test relativi al secondo software sviluppato sono simili a quelli visti nel paragrafo precedente. Anche in questo caso, per effettuare il test, il tester deve essere in possesso di competenze musicali evolute che gli consentano di rilevare la segnatura di tempo corrispondente ai brani oggetto del test. Come nel caso precedente il test consiste in una comparazione tra i risultati ottenuti dal tester e quelli forniti dall'algoritmo e l'assegnamento dei punteggi segue lo stesso criterio adottato precedentemente. I risultati vengono riportati nella tabella 5.2. Da un'analisi dei risultati si può osservare che il

Tabella 5.2: Risultati generali dell'algoritmo sviluppato utilizzando i template pesati.

Numero di brani	**Corretti**	**Errati**
132	28,03%	71,97%

software riesce a rilevare la scansione metrica dei brani in modo più accurato rispetto all'algoritmo modellato sull'idea di Brown; tuttavia i risultati non sono soddisfacenti in quanto la percentuale di risultati corretti non supera il 28%. Anche se il metodo si è dimostrato inefficace sarebbe interessante portare avanti il lavoro sperimentando nuove tipologie di template che consentano di ottenere risultati differenti. È probabile infatti che pesando in maniera differente i vari accenti all'interno dei template si ottengano risultati migliori o comunque più rappresentativi delle scansioni metriche reali.

Capitolo 6

Conclusione e sviluppi futuri

In questo elaborato si è cercato di sviluppare un software in grado di rilevare la segnatura di tempo dall'analisi di un brano in formato compresso. Come detto in precedenza la scelta di operare direttamente su file MP3 è dovuta al crescente impiego di tale tipologia di file negli archivi musicali, in particolare nelle collezioni private, e ai vantaggi derivanti dall'analisi diretta in dominio compresso. Per la realizzazione dell'algoritmo sono state prese in considerazione due teorie differenti che hanno dato origine a due software distinti. Purtroppo entrambi i software generati si sono dimostrati inefficaci nell'espletare il compito loro assegnato infatti osservando i risultati dei test effettuati si evince come la percentuale di successi, ovvero la percentuale di casi in cui i software sono riusciti ad individuare correttamente il parametro musicale cercato, sia molto bassa. Il secondo software sviluppato, quello che ricerca gli accenti metrici nel brano in esame, si è dimostrato più efficace del primo. Nonostante ciò l'algoritmo creato seguendo la teoria di Brown si è rivelato interessante per diversi aspetti. Innanzitutto il metodo dell'autocorrelazione può portare all'individuazione della velocità metronomica grazie al calcolo dell'IOI, deducibile osservando i picchi della funzione. In secondo luogo si può notare una tendenza nella rilevazione, tramite il metodo citato, di misure costituite da

un numero di movimenti multiplo rispetto a quello reale. Per cui in futuro si potrebbe considerare di modificare il software sviluppato in modo da conferirgli la capacità di risalire alla scansione metrica originale, considerando questi valori multipli come punto di partenza. Per quanto riguarda il secondo software, invece, sarebbe interessante impiegare dei template diversi per la rilevazione degli accenti. Una possibilità da tenere in considerazione è quella di non eliminare gli accenti deboli nel calcolo del valore di confidence ma di assegnare ad essi valori diversi da 0. Oltretutto sarebbe opportuno individuare una strategia più efficace per l'assegnamento dei valori relativi agli accenti forti e mezzo forti.

Appendice A

Codice per la costruzione dei file audio oggetto dei test

In questa appendice viene riportato il codice Matlab utilizzato per generare un file audio a partire dal WSP. Il funzionamento generale di tale algoritmo viene descritto nel capitolo 4 mentre di seguito sarà indicato solo il codice con i relativi commenti.

```
function [] = creawavdawsp()
path = input('inserisci il path del file BlockType.txt:  ','s');
%contiene il path del file blockType.txt
path = [path '\' 'BlockType.txt'];
%monta il path con il nome del file
Txt = dlmread(path,'\n');
%legge il file di testo estraendone una matrice
Aux = Txt;
%si tiene memoria della matrice originale
[r,c] = size(Txt);
```

```
%calcola le dimensioni della matrice
Txt(2:2:r) = [];
%la matrice viene convertita in un vettore
Txt = (Txt(1:r)\&1)+0;
%vengono convertiti tutti valori diversi da 0 in 1
[BPM,SHIFT,Bpm,MetFinale]=BeaTracking(path);
%importa la matrice di metronomi relativa al file blocktype.txt
metr = MetFinale((1:end),(Bpm-40)+1);
%viene estrapolato dalla matrice il vettore metronomo corrispondente
converteinwav(metr);
%crea un file wav corrispondente al metronomo
[Txt] = elimina1consecutivi(Txt);
%i transienti nel WSP vengono resi istantanei
ris=or(Txt,metr);
%vengono combinati il WSP e il metronomo
ris=ris+0;
%il vettore viene riportato di tipo numerico
converteinwav(Txt);
%crea un file wav corrispondente al WSP
converteinwav(ris);
%crea un file wav corrispondente al WSP combinato col metronomo
```

dove le funzioni richiamate nel codice corripondono rispettivamente a:

```
function[]= converteinwav(metr)
%converte un file testuale composto da 0 e 1 in un file wav
y = wavread('C:\ciarli');
```

```
%importa un suono di batteria precedentemente campionato
i = 1;
y(:,2) = [];
%il suono di batteria viene reso mono
s = y(1:576);
[r,c] = size(metr);
m = zeros((576*r),1);
k = find(metr);
[b a] = size(k);
if k(i) == 1
  m(1:576) = s;
  i=i+1;
end
while i <= b
  z = ((k(i)-1)*576;
  m(z:(z+575)) = s;
  i = i+1;
end
%ad ogni 1 presente nel vettore rappresentante il WSP vengono
sostituiti 576 campioni del colpo di batteria
indirizzo = 'C:\STAGE';
%contiene la directory dove sarà creato il file
nome = input ('inserisci il nome del file da creare:   ','s');
%permette di nominare il file creato
newpath = [ indirizzo '\' nome ];
%monta il nome con il path
wavwrite(m,44100,newpath);
```

```
%genera il file audio



function [Txt] = elimina1consecutivi(Txt)
%funzione che rende istantanei i transienti del WSP
k = find(Txt);
%rileva le posizioni nel vettore dei valori diversi da 0
[a b] = size(k);
j = 2;
while j <= a
      if k(j)-k(j-1) == 1
         Txt(k(j)) = 0;
      end
      j=j+1;
end
%sostituisce a tutti gli 1 consecutivi ad un valore 1 iniziale
il valore 0
```

Appendice B

Codice dell'algoritomo per l'individuazione della segnatura di tempo

In questa appendice viene riportato il codice Matlab relativo all'agoritmo descritto nel paregrafo 5.2.

```
convertein0e1
%funzione che importa il file blocktype.txt lo rende
mono e sostituisce ad ogni valore diverso da 0 un 1
[BPM,SHIFT,Bpm,MetFinale] = BeaTracking(path) ;
%importa la matrice di metronomi relativa al file blocktype.txt
metr = MetFinale((1:end),(Bpm-40)+1) ;
%viene estrapolato dalla matrice il vettore metronomo corrispondente
al tempo metronomico del file blocktype.txt
TempBin = [4;0];%definizione dei template
TempTern = [6;0;0];
```

```
TempQuat1 = [6;0;2.5;0];
TempQuat2 = [8;0;0;0];
TempQuin1 = [10;0;0;0;0];
TempQuin2 = [4;0;0;1;0];
TempQuin3 = [1;0;0;4;0];
TempSen = [12;0;0;0;0;0];
TempSet1 = [13;0;0;0;0;0;0];
TempSet2 = [6;0;0;0;1;0;0];
TempSet3 = [1;0;0;0;6;0;0];
BeatPosition = find(metr);
%genera un vettore contenete le posizioni dei beat nel metronomo
NumBeat = size(BeatPosition);
%conta il numero dei beat
Valut = zeros(11,1);
%conterrà i valori di confidence
metrout=zeros(11,size((metr),(1)));
%conterrà i metronomi combinati con i teplate
j=1;
for metro = 2:12
    %vengono ripristinate ad ogni iterazione le variabili metr,
    BeatPosition,NumBeat
    metr = MetFinale((1:end),(Bpm-40)+1) ;
    BeatPosition = find(metr);
    NumBeat = size(BeatPosition);
    switch metro %ogni caso corrisponde ad un template e in ogni
    caso viene convertito il numero di beat del metronomo a un
    multiplo del numero di elementi del template corrente
```

```
case {2}
    Microtemp = TempBin;
    BeatFinali = mod(NumBeat,2);
    metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
    BeatPosition = find(metr);
    NumBeat = size(BeatPosition);
case {3}
    Microtemp = TempTern;
    BeatFinali = mod(NumBeat,3);
    metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
    BeatPosition = find(metr);
    NumBeat = size(BeatPosition);
case {4}
    Microtemp = TempQuat1;
    BeatFinali = mod(NumBeat,4);
    metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
    BeatPosition = find(metr);
    NumBeat = size(BeatPosition);
case {5}
    Microtemp = TempQuat2;
    BeatFinali = mod(NumBeat,4);
    metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
    BeatPosition = find(metr);
    NumBeat = size(BeatPosition);
case {6}
    Microtemp = TempQuin1;
    BeatFinali = mod(NumBeat,5);
```

```
        metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
        BeatPosition = find(metr);
        NumBeat = size(BeatPosition);
    case {7}
        Microtemp = TempQuin2;
        BeatFinali = mod(NumBeat,5);
        metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
        BeatPosition = find(metr);
        NumBeat = size(BeatPosition);
    case {8}
        Microtemp = TempQuin3;
        BeatFinali = mod(NumBeat,5);
        metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
        BeatPosition = find(metr);
        NumBeat = size(BeatPosition);
    case {9}
        Microtemp = TempSen;
        BeatFinali = mod(NumBeat,6);
        metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
        BeatPosition = find(metr);
        NumBeat = size(BeatPosition);
    case {10}
        Microtemp = TempSet1;
        BeatFinali = mod(NumBeat,7);
        metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
        BeatPosition = find(metr);
        NumBeat = size(BeatPosition);
```

```
    case {11}
        Microtemp = TempSet2;
        BeatFinali = mod(NumBeat,7);
        metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
        BeatPosition = find(metr);
        NumBeat = size(BeatPosition);
    case {12}
        Microtemp = TempSet3;
        BeatFinali = mod(NumBeat,7);
        metr(BeatPosition(NumBeat-BeatFinali:end)) = 0;
        BeatPosition = find(metr);
        NumBeat = size(BeatPosition);
end
Dim = size(Microtemp);%è la dimensione del template
rot = Microtemp;%variabile necessaria per la rotazione
for l = 1:Dim(1);%viene montato il template al metronomo
  for i = 1:NumBeat(1)
      metr(BeatPosition(i)) = Microtemp(j);
      j = j+1;
      if j>Dim(1)
          j = 1;
      end
  end
j = 1;  %riporta j a 1
Ultimo = Microtemp(end);%il template viene fatto ruotare
Microtemp(2:end) = rot(1:end-1)  ;
Microtemp(1) = Ultimo ;
```

```
    rot = Microtemp ;
    Comb = metr.*Txt;%viene calcolato il valore di confiden
    CurrentSum =  sum(Comb);
       if Valut(metro-1)<CurrentSum
       %ad ogni rotazione il valore di confidence viene aggiornato solo
       maggiore di   quello calcolato precedenemente
        Valut(metro-1)= CurrentSum
        metrout((metro-1),(1:end))=metr(1:end);
       end
    end
end

 Massimo = max(Valut) ;
 %viene individuato il template che ha ottenuto il miglior risultato
  switch Massimo
  %stampa a video il risultato dell'analisi
        case { Valut(1) }
            disp('il metro è binario');
        case { Valut(2)}
            disp('il metro è ternario');
        case { Valut(3)}
            disp('il metro è quaternario');
        case { Valut(4)}
            disp('il metro è quaternario');
        case { Valut(5)}
            disp('il metro è quinario');
        case { Valut(6)}
```

```
            disp('il metro è quinario');
        case { Valut(7)}
            disp('il metro è quinario');
        case { Valut(8)}
            disp('il metro è senario');
        case { Valut(9)}
            disp('il metro è settenario');
        case { Valut(10)}
            disp('il metro è settenario');
        case { Valut(11) }
            disp('il metro è settenario');
    end
```

Ringraziamenti

Iannanzitutto desidero ringraziare il professor Luca Andrea Ludovico e il dottor Antonello D'aguanno per avermi guidato nella realizzazione di questo elaborato con i loro preziosi consigli e suggerimenti. Un ringraziamento và anche alla mia famiglia, che mi ha permesso di frequntare questo corso di laurea con estrema serenità, e a tutti i docenti di STCM per la loro professionalità e disponibilità. Non posso non ringraziare anche i miei colleghi universitari e amici che si sono sottoposti al noiosissimo test di ascolto consentendomi di portare avanti il lavoro; infine un ringraziamento speciale và alla mia ragazza e a tutti i miei amici che mi sono stati vicini in questo periodo nonchè a mia zia Rosa e a mia mamma (senza di voi probabilmente ora non sarei qui).

Bibliografia

[1] J.S. Downie. Music information retrieval. *Annual Review of Information Science and Technology*, 37(1):295–340, 2003.

[2] D.F. Rosenthal. Machine rhythm: computer emulation of human rhythm perception. 1992.

[3] R. Parncutt. A perceptual model of pulse salience and metrical accent in musical rhythms. *Music Perception*, 11:409–409, 1994.

[4] J.C. Brown. Determination of the meter of musical scores by autocorrelation. *JOURNAL-ACOUSTICAL SOCIETY OF AMERICA*, 94:1953–1953, 1993.

[5] M. Goto and Y. Muraoka. Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions. *Speech Communication*, 27(3):311–336, 1999.

[6] W.A. Sethares and T.W. Staley. Meter and periodicity in musical performance. *Journal of New Music Research*, 30(2):149–158, 2001.

[7] AP Klapuri, AJ Eronen, and JT Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*,

14(1):342–355, 2006.

[8] S. Vidili. Musica digitale: la codifica del segnale audio secondo lo standard mp3. Master's thesis, Poilitecnico di Torino, 1999.

[9] K. Salomonsen, S. Søgaard, and E.P. Larsen. Design and implementation of an mpeg/audio layer iii bitstream processor. *Department of communication technology, AALBORG University*, 1997.

[10] B. Lee, G. Associates, and CA Santa Clara. A new algorithm to compute the discrete cosine transform. *IEEE transactions on acoustics, speech and signal processing*, 32(6):1243–1245, 1984.

[11] T. Sporer, K. Brandenburg, and B. Edler. The use of multirate filter banks for coding of high quality digital audio. In *6th European Signal Processing Conference (EUSIPCO), Amsterdam*, volume 1, pages 211–214, 1992.

[12] G. Vercellesi. Metodi e prototipi software per l'elaborazione diretta di informazione audio in formato compresso mp3. Master's thesis, Università degli Studi di Milano, 2003.

[13] S. Kim, Y. Li, H. Kim, H. Choi, and Y. Jang. Real time mpeg1 audio encoder and decoder implemented on a 16-bit fixed point dsp. 2004.

[14] Iso/iec international standard is 11172-3 - information technology - coding of moving pictures and associated audio for digital storage media at up to about 1.5 mbits/s - part 3: Audio.

[15] Iso/iec international standard is 13818-3 - information technology - generic coding of moving pictures and associated audio, part 3: Audio.

[16] H. Zhaorong, D. Weibei, and D. Zaiwang. New window-switching criterion of audio compression. In *2001 IEEE Fourth Workshop on Multimedia Signal*

Processing, pages 319–323, 2001.

[17] F. Gouyon and S. Dixon. A review of automatic rhythm description systems. *Computer Music Journal*, 29(1):34–54, 2005.

[18] A. Nivuori. Estrazione della struttura metrica e ritmica a segnali audio musicali. Master's thesis, Università degli Studi di Milano, 2003.

[19] A. Klapuri and M. Davy. *Signal processing methods for the automatic transcription of music*. Tampere University of Technology, 2004.

[20] J.A. Bilmes. *Timing is of the essence: Perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive rhythm*. PhD thesis, Massachusetts Institute of Technology, 1993.

[21] F. Lerdahl and R. Jackendoff. *A generative theory of tonal music*. MIT press, 1983.

[22] E.W. Large and J.F. Kolen. Resonance and the perception of musical meter. *Musical networks: Parallel distributed perception and performance*, pages 279–312, 1999.

[23] D. Temperley and D. Sleator. Modeling meter and harmony: A preference-rule approach. *Computer Music Journal*, 23(1):10–27, 1999.

[24] S. Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 2001.

[25] C. Raphael. Automated rhythm transcription. In *In Proc. International Symposium on Music Information Retrieval*, 2001.

[26] C. Raphael. Modeling the interaction between soloist and accompaniment. In *Proc. 14th Meeting of the FWO Research Society on Foundations of Music Research*, 2001.

[27] A.T. Cemgil and B. Kappen. Monte carlo methods for tempo tracking and rhythm quantization. *Journal of Artificial Intelligence Research*, 18(1):45–81, 2003.

[28] M. Goto and Y. Muraoka. Music understanding at the beat level: Real-time beat tracking for audio signals. *Computational Auditory Scene Analysis*, pages 157–176, 1998.

[29] E.D. Scheirer. Tempo and beat analysis of acoustic musical signals. *The Journal of the Acoustical Society of America*, 103:588, 1998.

[30] J. Laroche, C.A.T. Center, and S. Valley. Estimating tempo, swing and beat locations in audio recordings. In *2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, pages 135–138, 2001.

[31] F. Gouyon, P. Herrera, and P. Cano. Pulse-dependent analyses of percussive music. In *IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS SPEECH AND SIGNAL PROCESSING*, volume 4, pages 4174–4174. IEEE; 1999, 2002.

[32] P. Allen and R.B. Dannenberg. Tracking musical beats in real time. In *Proceedings of the 1990 International Computer Music Conference*, pages 140–143, 1990.

[33] C. Palmer and C.L. Krumhansl. Mental representations for musical meter. *Mental*, 16(4):728–741, 1990.

[34] Y. Wang and M. Vilermo. A compressed domain beat detector using mp3 audio bitstreams. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 194–202. ACM New York, NY, USA, 2001.

[35] E. Kurniawati, E. Kurniawan, CT Lau, B. Premkumar, J. Absar, and S. George. Error concealment scheme for mpeg-aac. In *Communications Systems, 2004. ICCS 2004. The Ninth International Conference on*, pages 240–244, 2004.

[36] Y. Wang and S. Streich. A drumbeat-pattern based error concealment method for music streaming applications. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2002. Proceedings.(ICASSP'02)*, volume 3, 2002.

[37] R. Jarina, N. O'Connor, S. Marlow, and N. Murphy. Rhythm detection for speech-music discrimination in mpeg compressed domain. In *Proc. of the IEEE 14th International Conference on Digital Signal Processing DSP 2002*, pages 129–132, 2002.

[38] A. D'Aguanno and G. Vercellesi. Tempo induction algorithm in mp3 compressed domain. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 153–158. ACM New York, NY, USA, 2007.

[39] A. D'Aguanno, G. Haus, and G. Vercellesi. Mp3 window-switching pattern analysis for general purposes beat tracking on music with drums. In *Proceedings of the 120th AES convention*, Paris, France, 2006. AES.

[40] A. D'Aguanno. Metodi e prototipi software per la sincronizzazione automatica di segnali audio musicali compressi mp3 e partiture xml. Master's thesis, Università degli Studi di Milano, 2005.

Elenco delle figure

Elenco delle tabelle