UNIVERSITA' DEGLI STUDI DI MILANO

CORSO DI LAUREA IN SCIENZE E TECNOLOGIE DELLA COMUNICAZIONE MUSICALE

CONTROLLO DI RIFF PER CHITARRA TRAMITE WIIMOTE

Relatore:
Prof. Luca A. Ludovico
Correlatore:
Dott. Adriano Baratè

TESI DI LAUREA DI :
DAVIDE IRTO
Matricola 722023

Anno Accademico 2009/2010

INDICE

INTRODUZIONE…….....3

CAPITOLO 1:

LA CHITARRA E I RIFF ....……..……5

1.1) TIPI DI CHITARRE ..…….……..5

1.2) UN PO' DI STORIA …….6

1.3) LO STRUMENTO.……...8

1.4) I RIFF DI CHITARRA ……...16

1.5) ESEGUIRE UN RIFF…….18

CAPITOLO 2:

TECNOLOGIE E LINGUAGGI COINVOLTI NELLA REALIZZAZIONE DELL'INTERFACCIA……..……20

2.1) NINTENDO WII E WIIMOTE……..……..20

2.1.1) Caratteristiche e specifiche hardware ……..21

2.1.2) Wii Remote……...23

2.1.3) Funzionalità del WiiMote…….24

2.2) DIRECTSOUND …….25

2.3) IL LINGUAGGIO C# ……27

CAPITOLO 3:

PROGETTO CONTROLLO DI RIFF PER CHITARRA TRAMITE WIIMOTE…….……28

3.1) DESCRIZIONE INTERFACCIA UTENTE……..……..28

3.2) CONFIGURAZIONE WIIMOTE……..…….32

3.3) DIRECTSOUND ……33

3.4) LIBRERIA DI RIFF……35

3.5) UTILIZZO DEL GUITAR RIFF CONTROLLER …….36

CONCLUSIONI E SVILUPPI FUTURI ……...48

BIBLIOGRAFIA ……49

Introduzione

Le tecnologie sono in continuo cambiamento e sviluppo e ci hanno coinvolto oramai tanto da non poterne fare a meno.

Il passaggio dall'analogico al digitale è divenuto una concreta realtà anche per le trasmissioni televisive, cui è stato imposto l'utilizzo del digitale terrestre. Quindi, anche chi non era del tutto favorevole a questo tipo di cambiamenti tecnologici, è dovuto sottostare all'esigenza dei più e lasciarsi trasportare ormai nell'era digitale.

Nell'ambito musicale, l'utilizzo di apparecchiature digitali è praticamente normalità, non solo nella fase di produzione, ma anche nella composizione e quindi nella scrittura della musica tramite programmi per l'editing digitale della partitura.

Nel presente elaborato si sfrutterà l'utilizzo di nuove tecnologie interattive, in cui l'utente stesso avrà pieno controllo dell'oggetto sonoro in questione, ovvero i riff di chitarra, e potrà quindi apportarvi modifiche in tempo reale.

L'oggetto tecnologico che ha dato modo di implementare queste funzioni è il controller WiiMote, della famosa console di gioco Nintendo Wii, strumento molto versatile grazie ai dispositivi al suo interno. Questo controller utilizza tecnologia Bluetooth, e permette quindi di poter interagire con l'interfaccia implementata modificandone i parametri.

Quest'elaborato è suddiviso in tre capitoli.

Nel primo capitolo viene ampiamente descritta la chitarra, ovvero lo strumento musicale utilizzato per creare i riff che si andranno poi a manipolare durante l'utilizzo dell'interfaccia. Oltre alla storia di questo strumento, verranno analizzate la sua struttura e le sue componenti cercando di mostrarne il funzionamento meccanico.

Inoltre verrà data la definizione di riff e di come questo venga impiegato nell'esecuzione di un brano suonato con la chitarra, esaminando alcune tecniche che il chitarrista può sfruttare per arricchire l'esecuzione.
Nel secondo capitolo si trovano le informazioni relative alle tecnologie utilizzate per la creazione dell'interfaccia finale quali il linguaggio di programmazione C# adottato per costruire il vero e proprio codice, DirectX DirectSound per gestire i file audio da caricare e infine ciò che rende innovativa l'idea principale dell'elaborato, il Wii Remote.
Il terzo e ultimo capitolo sarà dedicato all'analisi dell'interfaccia creata, al suo funzionamento, a come viene utilizzato il WiiMote, prendendo in esame le porzioni di codice più significative, allo scopo di capire il funzionamento del progetto "Guitar Riff Controller".

1. La Chitarra e i Riff

La chitarra è uno strumento a corde, le quali fatte vibrare producono onde sonore. Le corde sono tese tra il ponticello, posto alla base della cassa armonica, ed il capotasto, posto alla cima del manico, sul quale si trova la tastiera, che permette di accorciare la lunghezza della parte di corda vibrante e a sua volta di suonare la nota o le note desiderate premendo la corda stessa appena dietro il rispettivo tasto. L'accordatura più comune, nota come *accordatura spagnola*, è Mi-Si-Sol-Re-La-Mi dalla corda più acuta alla più grave. Questa accordatura, in cui l'intervallo tra due corde adiacenti è di una quarta giusta (eccetto tra seconda e terza corda, che distano di una terza maggiore), si è imposta per ragioni storiche e per la sua praticità nel formare accordi mediante posizioni della mano sinistra non eccessivamente complesse.

1.1 Tipi di chitarre

Fig. 1.1 – Chitarra Acustica

Le chitarre possono essere suddivise innanzitutto in due categorie, a seconda del modo in cui viene amplificato il suono delle corde in vibrazione:

Fig. 1.2 – Chitarra Elettrica

- acustiche, con un corpo vuoto a formare la cassa armonica, quando è principalmente prevista una amplificazione che sfrutti le naturali proprietà della fisica del suono, secondo un tipo di amplificazione che potremmo definire meccanico;
- elettriche, con un corpo "pieno" e solido nella maggior parte dei casi, per le quali è necessario l'ausilio di una

amplificazione elettrica (tramite collegamento ad una cassa acustica o un amplificatore che funzionano a corrente elettrica). Il suono acustico di una chitarra elettrica a corpo solido è molto debole e poco percepibile, data l'assenza di cassa di risonanza.

1.2 Un po' di storia

Prima dello sviluppo della chitarra elettrica e dei suoi componenti di tipo sintetico ed elettrico, la chitarra veniva definita come uno strumento costituito da "un manico lungo suddiviso in tasti, una piatta tavola armonica di legno, e un fondo piatto, con le due parti laterali spesso incurvate". Il termine chitarra veniva utilizzato per riferirsi ad una serie di strumenti connessi, che sono stati sviluppati ed utilizzati, a partire dal XII secolo, in tutta l'Europa, e successivamente anche nelle Americhe. Questi tipi di strumenti discendevano da quelli già esistenti nell'antica Asia centrale e in India. Per questo motivo si può affermare che la chitarra sia imparentata alla lontana con gli strumenti moderni di queste regioni, come il *Tanbur*, il *Setar*, e il *Sitar*. La più antica rappresentazione iconografica di uno strumento che mostra le caratteristiche principali di una chitarra è un bassorilievo in pietra risalente a 3.300 anni fa, in cui un bardo *Ittito* è intento nel suonare lo strumento.

Fig. 1.3 – Bardo Ittito

Il termine moderno "chitarra", e gli antecedenti, fin dai tempi antichi, sono stati usati per intendere una diversa varietà di cordofoni e come tale è causa di confusione. La parola italiana *chitarra*, l'inglese *guitar*, la tedesca *gitarre*, la francese *guitare*, sono stati tutti adottati dalla *guitarra* spagnola, la quale proviene dall'arabo andaluso قيثارة (*qitara*), derivato anch'essa dal latino *cithara*, a sua volta derivato dal greco antico κιθάρα (*kithara*), che in ultima analisi si pensa possa essere un termine proveniente dal linguaggio persiano antico, dove la parola *Tar* significa "corda".

Lo strumento vero e proprio però, sembra che abbia origini differenti rispetto all'etimologia della stessa. Difatti la chitarra intesa come tale è uno strumento che ha subito diverse influenze dal punto di vista costruttivo. Una comunemente citata fu la comparsa del *liuto* a quattro corde, introdotto dagli invasori arabi nel secolo VIII. Un'altra possibile influenza può essere suggerita dallo scandinavo *lut* a sei corde, che divenne popolare nelle zone europee invase dal popolo vichingo durante il periodo medievale. Come si narra nella leggenda di *Sigfrido*, l'eroe scandinavo *Gunther* (noto anche come *Gunnar*), suonò un liuto mentre moriva in una fossa di serpenti.

Fig. 1.4 – Liuto Arabo

È quindi possibile che la creazione della chitarra, sia dovuta alle diverse influenze degli strumenti a corda e a pizzico provenienti da tutto il Mediterraneo.

Nel 1200 erano in uso due tipi di strumenti dal nome di "chitarra": la *guitarra moresca* (chitarra moresca) e la *guitarra latina* (chitarra latina). La chitarra moresca e la chitarra latina presentavano diverse caratteristiche. La prima aveva una tastiera ampia, il corpo caratterizzato da molti fori di risonanza e la parte posteriore del corpo era di forma arrotondata; mentre la chitarra latina presentava una singola buca di risonanza ed un manico più stretto. Dal XIV secolo, però, i termini *moresca* e *latina* vennero abbandanati, e questi due tipi di cordofoni vennero semplicemente indicati col nome di chitarra.

Lo strumento che viene ampiamente considerato per aver avuto un'influenza fondamentale nello sviluppo della chitarra, è la *vihuela spagnola* (viola da mano), uno strumento simile alla chitarra risalente al XV e al XVI secolo. Questa era dotata di sei corde, di cui cinque doppie, aveva un'accordatura come quella del *liuto*, quindi per intervalli di quarta, e la forma del corpo era simile a quella della chitarra.

Fig. 1.5 – Vihuela Spagnola

Nel tardo 1600 le *vihuela* venivano suonate anche con un arco, portandò così allo sviluppo della *viola da gamba*. Mentre, con il XVI secolo la costruzione della *vihuela* aveva caratteristiche che la accomunano maggiormente alla chitarra moderna.

Questo strumento, però, beneficiò soltanto di un breve periodo di popolarità in Spagna ed in Italia, durante un'era in cui lo strumento predominante era il liuto.

L'ultima pubblicazione musicale rinvenuta per la *vihuela*, risale al 1576.

Nel periodo dal XVI secolo al XVIII, in Spagna, Italia e Francia, divenne popolare l'utilizzo della chitarra barocca a cinque cori (ovvero cinque corde doppie), che dalle documentazioni rinvenute, era un chitarra spagnola nata nella metà del 1500.

In Portogallo, il termine *vihuela* si riferiva alla chitarra, mentre per *guitarra* si intendeva la *chitarra portoghese*, una varietà di *cetra*.

1.3 Lo Strumento

CHITARRA CLASSICA:

Come si può osservare nell'immagine a fianco lo strumento può essere suddiviso in tre sezioni:

- LA PALETTA
- IL MANICO
- IL CORPO

A sua volta, ogni sezione è composta da vari componenti:

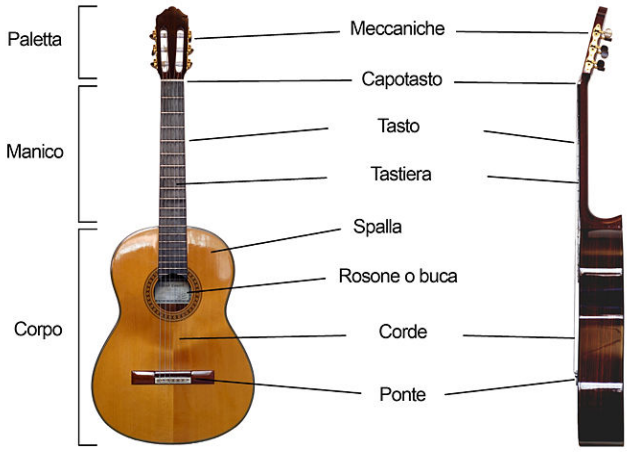

Fig. 1.6 – Chitarra Classica e i suoi componenti

LA PALETTA:

la paletta, chiamata in questo modo in quanto la sua forma ricorda una vera e propria pala. Essa custodisce tutte le *meccaniche* dello strumento. Queste, chiamate anche *Bischeri*, sono delle piccole leve che tendono la corda, dandone l'intesità addatta al suono da esprimere. Maggiore sarà la tensione della corda, più il suono sarà acuto.

Dall'immagine si può notare che alla base della paletta si trova anche un secondo componente: il *capotasto*. Questo come si può evincere dal nome, è il capo dei tasti, ovvero il primo, ed insieme al ponte, posto sul fondo del corpo dello strumento, determina la lunghezza della scala e trasmette le vibrazione delle corde al manico e al corpo.

IL MANICO:

il manico è composto interamente da *tasti*, che a loro volta compongono la *tastiera*. Ogni tasto premuto darà un'intensità diversa alla corda suonata, che a mano a mano che si scende lungo la tastiera, diventerà sempre più acuta.

IL CORPO:

detto anche *body*, si può ulteriormente suddividere in tre sezioni:

- *LA CASSA*: questa è il vero e proprio corpo della chitarra, chiamata in questo modo in quanto è una vera e propria cassa di risonanza, in cui le vibrazioni emesse dalle corde, entrando nel *rosone*, vengono amplificate all'interno di essa, uscendo sempre dal rosone in un'emissione sonora esponenziale.
- *LE CORDE*: le corde sono 6, e vanno considerate dalla più acuta alla più grave. Come già affermato l'accordatura più comune è "MI-SI-SOL-RE-LA-MI", ma ne possiamo trovare di altre come il *drop-D*, in cui l'ultima corda è ulteriormente abbassata di un tono.
- *IL PONTE*: esso costituisce la parte finale della tastiera, che in apparenza può sembrare che termini prima del rosone, ma tecnicamente è possibile esercitare una pressione sulle corde, anche in una zona di "vuoto" trovando i cosidetti *tasti invisibili*.

CHITARRA ELETTRICA

Per la chitarra elettrica si prenderà in considerazione solo la sezione del *corpo*, essendo le altre due identiche sia per struttura che per i componenti. Le uniche differenze saranno dovute dai diversi materiali adoperati durante la fase di costruzione.

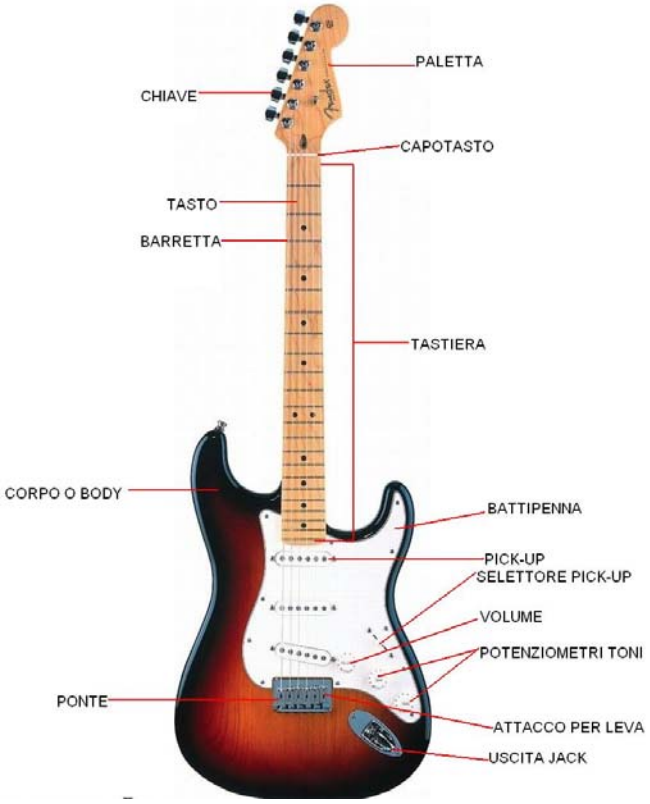

Fig. 1.7 – Chitarra Elettrica e i suoi componenti

Il *body*, come nella chitarra classica, risulta un elemento fondamentale sia da un punto di vista estetico, che da quello timbrico. Esso ospita tutti i componenti che permettono allo strumento di comunicare, per mezzo dell'amplificazione, le informazioni sonore, quali i pick-up, i controlli di tono e del volume, il selettore dei pick-up, la presa per il jack(cavo).

- *I PICK-UP:* questo componente è un elemento molto importante nella catena che genererà il suono finale della chitarra elettrica. Il suo funzionamento è simile ad un trasduttore, ovvero un dispositivo elettronico o elettromagnetico in grado di convertire energia fisica, in questo caso la vibrazione della corda, in energia elettrica. Gli impulsi elettrici verranno poi diretti all'amplificatore, da cui si potrà modificare il segnale, per poi essere trasmesso ai coni dell'alto parlante, attraverso il quale si potrà udire le sue caratteristiche sotto forma di suono fisico.
  Il pick-up della chitarra è solitamente magnetico, nello specifico è composto da una magnete, la cui funzione sarà quella di creare un campo magnetico, all'interno del quale passano le corde. Difatti, il pick-up, può funzionare solo in

prossimità delle corde, ed è per questo motivo che vengono montati proprio sotto di esse.

Possiamo suddividere i pick-up in due categorie: *single-coil*, *humbucker*, con le loro variazioni (pick-up *attivo* e *splittabile*).

Il primo, il cui significato è pick-up a bobina singola, viene introdotto per la prima volta su di una chitarra elettrica da *Leo Fender*.

Fig. 1.8 – Pick-up single coil

La caratteristica principale di questo tipo di pick-up consiste in un sound brillante e con tanti acuti. Nel disegno a fianco, viene mostrata la sezione di un single-coil montato su una *Fender Stratocaster*. Si può notare come le sei espansioni polari siano avvolte dalla bobina di filo di rame, i cui due capi finali dell'avvolgimento vengono uniti ai due cavetti collegati al circuito elettrico della chitarra.

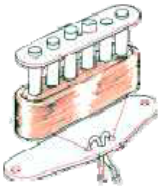

Fig. 1.9 – Avvolgimento bobina nel single coil

Fig. 1.10 – Pick-Up Humbucker

Il pick-up a bobina doppia, l'*humbucker*, ovvero compensatore di ronzio. Il termine ha motivo di esistere, in quanto questo genere di pick-up, è appunto privo di rumori di fondo. Difatti il *single-coil*, pur avendo una bellezza timbrica, presenta un piccolo difetto, ovvero quello di un alto rischio di rumore di fondo. Questo si può evincere dal fatto che tutte le bobine dei pick-up sono molto sensibili alle interferenze derivanti da radiazioni elettromagnetiche, in quanto tendono a raccogliere rumori e ronzii da apparecchi elettrici, amplificatori, onde radio ecc. Nel corso degli anni, è stato messo a punto un sistema in grado di neutralizzare del ronzio, tramite un modello di pick-up ideato da *Seth Lover*, tecnico della *Gibson*, nel 1955.

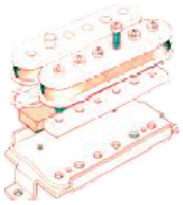

Fig. 1.10 – Avvolgimento bobina nell'humbucker

A differenza del single-coil, nell'humbucker si utilizzano due bobine collegate in serie, in cui nella prima, la corrente passa

con un segnale positivo, mentre nella seconda con segnale negativo. Questi due flussi di corrente, scorrendo in direzioni opposte, si annullano l'una con l'altra, evitando così di trasmettere un eventuale ronzio all'amplificatore. Però a quest'ultimo dovrà giungere anche il suono generato dalla vibrazione delle corde nel campo magnetico, e proprio per questo motivo, le serie di poli di ogni bobina avranno polarità magnetiche opposte.

La qualità sonora del pick-up sarà nettamente diversa da quella del single-coil: l'humbucker, avrà un suono più pieno e una minore definizione delle frequenze acute. Si sono quindi create due tipologie di suono, il suono *strato* (da Fender Stratocaster), caratterizzato dal single-coil, e il suono *Gibson*, caratterizzato dall'humbucker. Le dimensioni del secondo, essendo a doppia bobina sono praticamente doppie rispetto a quelle del single-coil.

Nella figura a fianco si può osservare la struttura di un pick-up a bobina doppia sul modello Gibson: la bobina principale è dotata di espansioni regolabili a vite, mentre quella della bobina secondaria rimangono fisse. La serie di poli rimane in contatto con il magnete posto sotto le bobine bloccato sulla piastra base. In alcuni modelli, come l'humbucker PAF della Gibson, il pick-up è racchiuso all'interno di una scatola di metallo. Nelle bobine, il numero dei giri del filo di rame può variare a seconda della scelta timbrica destinata al pick-up.

Il *pick-up splittabile*, dà la possibilità di incrementare la varietà sonora del pick-up. Questo sistema offre quindi più suoni allo strumento, e consiste nell'utilizzo di un selettore *switch*, che si interpone tra il pick-up e il controllo della chitarra. Azionando questo interruttore si potrà quindi scegliere quale delle due bobine del pick-up si utilizzerà. Ovviamente si potrà usare anche nella forma standard con entrambe le bobine.

Il *pick-up passivo* identifica i modelli di pick-up finora analizzati. In pratica, viene riferito al fatto che la generazione del segnale elettrico fatta dal pick-up è pulita, nel senso che non dispone di rinforzi e di controlli (a parte i classici controlli di tono e volume, che sono sempre presenti su tutti i modelli, anche se con qualche rarissima eccezione) che ne alterino in un qualche modo il segnale.

La pasta del suono in un pick-up passivo è quindi naturale, bella o brutta a seconda del modello, ma anche del gusto personale di chi lo utilizza.

Il pick-up **attivo** è un trasduttore a bassa impedenza a cui viene affiancato un circuito elettronico detto di preamplificazione, che, a seconda dei casi, svolge un ruolo differente nel controllo del segnale creato dal campo magnetico del pick-up. Questo circuito può controllare la dinamica delle frequenze attraverso dei potenziometri, frequenze che il più delle volte sono i medi, ma alcune marche offrono anche la possibilità di equalizzare tutto lo spettro delle frequenze, e cioè bassi, medi e acuti, impostati a seconda del tipo di risposta al segnale venga richiesta. Questo circuito viene alimentato da una pila (in genere la classica da 9v) e si attiva nel momento stesso in cui viene inserito il jack nella chitarra.

Fig. 1.11 – Manopola volume nella chitarra elettrica

- *I CONTROLLI DI TONO E VOLUME*: tra i pick-up e la presa jack di uscita dello strumento si possono trovare i controlli di tono e volume, che regolano sia i toni dello strumento, che il volume in uscita. Nel caso di un controllo di un volume, abbiamo un potenziometro che modifica il segnale in uscita per mezzo di un resistore variabile, il quale è collegato alla manopola di controllo posta sul corpo dello strumento attraverso un albero ruotante. Questo contiene una resistenza, solitamente a forma di ferro di cavallo. All'estremità dell'albero, opposto alla manopola, è connesso un contatto eccentrico, il quale, quando la manopola viene ruotata, si sposta intorno alla resistenza. Alle estremità di questa sono collegati due piccoli cavi, che provengono dal pick-up: una delle estremità ha tensione zero, l'altra, al contrario, è al massimo. Durante il movimento della

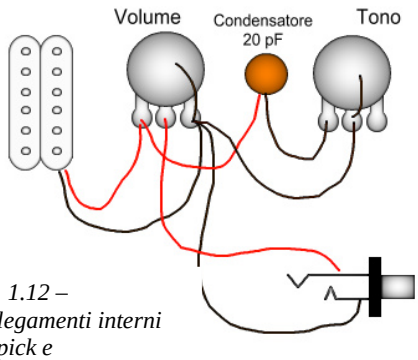

Fig. 1.12 – Collegamenti interni tra pick e potenziometri

manopola, il contatto eccentrico trasmette, in merito alla posizione in cui si trova, il valore della tensione all'uscita del potenziometro.

Fig. 1.13 – Manopola del tono nella chitarra elettrica

Nel caso del potenziometro del tono, anch'esso collegato tramite un albero ad una manopola di controllo sullo strumento. In questo caso, il procedimento è una combinazione tra potenziometro e condensatore, i quali lavorano trasmettendo le frequenze alte e scaricandole a terra. Le frequenze alte passano dal condensatore, che funge da filtro, ma dal quale non passano le frequenze basse. Nello specifico, con la rotazione della manopola, il potenziometro, con il suo contatto eccentrico a scorrimento, determina quante frequenze alte debbano essere scaricate. Con il controllo di tono aperto al massimo, viene inviato all'uscita del potenziometro tutto il segnale generato dai pick-up, mentre chiudendolo verranno filtrate le frequenze alte. Quindi è errato pensare che il potenziometro del tono aumenti le frequenze alte, mentre quando è chiuso siano le frequenze basse ad essere enfatizzate. Difatti, le basse frequenze non vengono assolutamente toccate, mentre quelle alte vengono tagliate in misura maggiore in base a quanto il potenziometro venga chiuso.

- *IL PONTE*: nella chitarra elettrica il ponte assume un'importanza fondamentale. Difatti, oltre al fatto che come nella chitarra acustica ha la funzione di tener ancorate le corde, entrano in gioco diversi fattori che ne fanno aumentare le caratteristiche tecniche. Due sono i tipi di ponte che generalmente si possono trovare montati su una chitarra elettrica: il *ponte fisso* e il *ponte tremolo*, detto anche *ponte vibrato*.


Fig. 1.14 – Ponte tremolo


Fig. 1.15 – Ponte fisso

Nel primo caso, il ponte consiste in una placca di metallo avvitata al corpo sulla quale vengono montate delle sellette mobili, regolabili a vite. Queste permettono

la regolazione dell'intonazione dello strumento, da non confondere con l'accordatura, un'operazione che viene effettuata ritoccando appunto le sellette appunto, in modo da registrare il diapason, cioè la lunghezza attiva della corda attiva dal ponte al capotasto. Le sellette possono essere regolate anche in altezza, in modo da poter impostare l'*action* delle dita sulle corde, ovvero la distanza tra le corde e il manico.

Il *ponte tremolo* ha la funzione principale di far variare l'intonazione delle corde, in senso crescente o calante. Ciò è reso possibile da un ponticello mobile, che viene tenuto in posizione fissa da alcune molle, poste al di sotto del ponte stesso, ed ancorate al corpo dello strumento. Al ponte viene inserita una leva, *leva del vibrato*, su cui viene fatta pressione con la mano destra, muovendo così il ponte e allentando la tensione delle corde. Una volta che la leva viene rilasciata, le molle riportano il ponte alla posizione originaria.

Fig. 1.16 – Ponte Floyd Rose

I problemi relativi alla sua progettazione, hanno sempre riguardato la fase di ritorno delle molle, che non rispondono perfettamente ad un uso forsennato della leva. La ricerca dell'intonazione perfetta ha così portato alla costruzione, a partire dagli anni '70, di modelli di ponte tremolo sempre più sofisticati, quali i sistemi Floyd Rose, Shaller, Steinberger e i più recenti Levinson, Wilkinson e AlecB. Il modello più imitato probabilmente è il *Floyd Rose*. La tecnica grazie alla quale è possibile ottenere dei grandi movimenti nella tensione delle corde, senza per questo comprometterne l'intonazione una volta riportato il ponte alla fase statica, è quella del blocco completo delle corde sia sul ponte, grazie a delle sellette che stringono come in una morsa le corde sul ponte e grazie anche ad un

Fig. 1.17 – Blocca Corde

bloccacorde situato sulla paletta, che non permette alle corde di muoversi dalle meccaniche.

1.4 I riff di chitarra

I **riff** sono frasi musicali, ossia delle successioni di note con una propria identità espressiva, che, spesso, eseguiti nella sezione ritmica da strumenti come, la chitarra o le tastiere, si ripetono frequentemente all'interno di un brano formando così le basi dell'accompagnamento di una composizione.
Un riff può essere semplice, utilizzando singole note o più complesso mediante l'utilizzo di accordi.
Il termine *riff* entrò a far parte dello slang musical negli anni '20, e fu usato principalmente in associazione a generi musicali come il rock o il jazz. Molti musicisti rock usano il termine *riff* come sinonimo di 'idea musicale'. Alcune fonti indicano che *riff* è l'abbraviazione di *rhythmic figure* (figura ritmica), altre lo indicano come abbreviazione e alterazione di *refrain* (refrain->ref->rif->riff). Tuttavia, l'utilizzo in ambito musicale del termine, deriva dalla commedia dove *riffing* sta per "fare osservazioni brevi ed intelligenti su un soggetto".

David Brackett, professore di musica della McGill University, li definisce come delle "brevi frasi melodiche", mentre il compositore Richard Middleton li definisce come delle "brevi figure ritmiche, melodiche o armoniche ripetute, che formano un quadro strutturale" Anche se Rikky Rooksby ammette che non esiste una vera e propria definizione, fa una descrizione dei riff nel rock: "Un riff è una frase musicale, breve, ripetuta, facilmente memorizzabile, spesso con note gravi, che si focalizza sull'energia e l'entusiasmo di una canzone rock."

Riff è diverso dal concetto di *fraseggio*, difatti può includere progressioni di accordi ripetuti; mentre i *licks*, i fraseggi appunto, sono solitamente associati a linee

melodiche composte da singole note, piuttosto che da successioni di accordi. Tuttavia, i licks, come i riff, possono essere impiegati come basi di un intero brano. Nella musica classica questi due termini non vengono utilizzati, bensì, per indicare tipi di strutture di simili al lick o al riff si utilizza il termine *ostinato*. I compositori jazz moderni, nella musica modale e latin jazz utilizzano i riff e i lick, come degli "ostinato". Il riff può essere anche indicato come "hook" (=gancio), definito come: "un'idea musicale, un passaggio od una frase, che fa risaltare la canzone e cattura l'attenzione dell'ascoltatore".
Il compito dell'artista, quindi, è quello di collegare, nell'atmosfera generale, il tema principale, il sound del pezzo e tramandarlo al pubblico, essendo un qualcosa di orecchiabile, in modo da impegnare nel modo più completo l'ascoltatore. Così, il riff, può essere usato come un "gancio" per attirare il pubblico nella piena atmosfera di una canzone.

Un esempio di riff nella musica rock è dato dall'introduzione di chitarra dei Deep Purple in "Smoke on the water". Un esempio di riff nella musica classica, genere nel quale si usa più propriamente il termine ostinato, è dato dal *Bolero* di Ravel.
Nella musica jazz si caratterizza come una breve frase musicale, semplice, generalmente facile da ricordare, destinata a durare più o meno a lungo, normalmente utilizzata come sottofondo a improvvisazioni solistiche, ma che può anche costituire il nucleo di un brano musicale.
Più in generale la parola riff denota brevità, ripetizione e tensione ritmica ed è spesso associata a uno specifico strumento (nella musica rock principalmente la chitarra): differisce, quindi, dall'assolo, che è una breve esecuzione solistica, senza ripetizione e a volte con virtuosismi.
Per cellule ripetitive più prolungate (o a carattere rilassato) si usa talvolta il termine groove: quest'ultimo è associato a un ritmo marcato e divertente in contrapposizione al ritmo melodico del riff.

1.5 Eseguire un riff

I riff possono essere eseguiti in diversi modi, dipendenti dal contesto sonoro in cui ci si trova. Difatti il chitarrista può scegliere il tipo di effetto più adatto a risaltare la sua esecuzione introducendo elementi come il **palm mute**, ovvero una tecnica in cui le corde vengono smorzate con il palmo della mano destra, ovvero quella che pizzica.
Questa tecnica può avere la semplice funzione di interrompere un suono, "suonare una pausa" o, può essere adoperato in sostituzione delle pause, che risulterebbero altrimenti troppo vuote rispetto ad una linea ritmica continua. Ha quindi lo scopo di abbellire un pattern o lasciare spazio ad altri strumenti.
La tecnica ha sicuramente radici molto lontane, difatti il suo uso è rinvenibile in parecchie celebri composizioni adattate alla chitarra classica, con il nome di *Pizzicato.* Il *Palm Mute* è utilizzato su vasta scala spaziando dal *Rock*, al *Blues*, al *Funky*, al *Punk* e al *Metal.*
Con una chitarra molto distorta ne deriverà un suono molto basso e ricco di armoniche, difatti nel Metal, il *palm mute* è una tecnica utilizzata molto spesso, suonando sulle ultime corde (le più gravi), principalmente su sezioni ritmiche; il suo utilizzo negli assoli ha lo scopo di evidenziare le note basse nell'esecuzione di una scala, data la notevole difficoltà dell'eseguirlo sulle prime corde (le più fini).

Fig. 1.18 – Pedale Cry Baby

Ovviamente però, chitarra elettrica non significa suonare solo con suono distorto, ma anche con il suono dell'amplificatore pulito (*clean*). La sonorità senza distorsioni rimane molto più dinamica. Questo tipo di suono viene utilizzato a piacimento dall'esecutore, in base al genere del pezzo, in base al tipo di parte che si sta suonando, dando così sfumature diverse alla composizione arricchendola di particolari. Un altro effetto particolare si ha con il *Wah* (o anche Wah-Wah), nome onomatopeico, che indica l'effetto del suono prodotto dal taglio e dal reinserimento

graduale delle frequenze alte, creando appunto un effetto simile al suono Wah. Questo particolare tipo di effetto si produce mediante l'utilizzo di pedali, di cui il più famoso è il marchio *Cry Baby* della Jim Dunlop. Il pedale modifica il suono delle chitarra tramite un graduale cambiamento del tono tra acuti e bassi, controllati da un potenziometro azionato dalla spinta del piede sul pedale, rispettivamente verso la punta con aumento delle frequenze alte, e verso il basso con l'aumento delle frequenze medie. Il *Wah* è un effetto molto sfruttato dai chitarristi, a renderlo celebre è stato Jimi Hendrix nel pezzo *Voodoo Child (Slight Return)* e molti altri chitarristi come Eric Clapton, David Gilmour, Santana, Zakk Wylde, Jeff Beck, John Frusciante, Kirk Hammet ecc. Un genere in cui il wah-wah è stato usato spesso è il funk e il reggae.

2. Tecnologie e linguaggi coinvolti nella realizzazione dell'interfaccia

2.1 Nintendo Wii e WiiMote

Fig. 2.1 – Console e controller Wii

Prodotta dalla Nintendo, la Wii è una console per videogames ed è stata la più venduta tra quelle di settima generazione. Presentata per la prima volta nel 2005 con il nome di Nintendo Revolution, questa console avrebbe portato ad una vera e propria rivoluzione del mondo dei videogames. Successivamente venne dato il nome definitivo, noto a tutti come Nintendo Wii. Ciò che la contraddistinse dalle concorrenti Sony Play Station 3 e Microsoft Xbox, fu l'innovativa introduzione di un controller interattivo, il Wii Remote, un controller senza fili, che reagisce alle forze vettrici e all'orientamento rispetto allo spazio tridimensionale. Ciò è reso possibile da un accelerometro a 3 assi presente al suo interno, e tramite una rudimentale videocamera posta ad una delle sue estremità interagisce con la barra sensore rendendo, inoltre, possibile il suo utilizzo come sistema puntatore sullo schermo TV. Proprio questa caratteristica rivoluzionò l'approccio del gioco, portando così la console a divenire la leader nel mercato videoludico. Difatti il punto di forza della Wii non sta nella complessità dei videogiochi o nella prestazione, ma solo nella nuova e innovativa interazione del giocatore con il videogioco stesso.

Il nome "Wii" vuole rappresentare l'idea di un divertimento da condividere con più persone, difatti la pronuncia "Wii" è la stessa di "we" (in inglese, noi) e si può notare come le due "i" rappresentino le sagome stilizzate di due persone.

Fig. 2.2 – Logo Wii

2.1.1 Caratteristiche e specifiche Hardware

L'innovativa console grazie alle sue dimensioni ridotte, al suo peso leggero e al suo design può essere posizionata sia verticalmente che orizzontalmente, in modo da agevolare gli eventuali spostamenti.
Nello slot di inserimento di dischi ottici possono essere caricati tutti i dischi della Wii ed essendo retro compatibile permette l'utilizzo dei cd della console precedente, la Nintendo GameCube. Inoltre è dotata di slot per memorie esterne SD-card utilizzate per salvare i dati di gioco o per caricare file esterni come delle foto. Questo tipo di memorie espandono la già presente memoria interna (512 Mb di memoria flash). Ecco i dati tecnici della Wii:

Processore:
CPU: processore PowerPC-based "Broadway", con processo a 90nm SOI CMOS e frequenza 729 MH
GPU: processore video ATI "Hollywood" GPU con processo a 90nm CMOS e frequenza 243 MHz

Memoria:
88 MB di memoria principale (24MB "interna" 1T-SRAM integrata nel pacchetto grafico,
64MB "esterna" GDDR3 SDRAM)
3MB embedded GPU memoria di texture e framebuffer

Porte e ingressi:

Fino a 16 Wii Remote controller (10 in Standard Mode, 6 in One Time Mode, connessi
wireless via Bluetooth)
4 porte Nintendo GameCube controller
2 slot Nintendo GameCube Memory Card
SD memory card slot (supporta SDHC cards)
2 porte USB 2.0
porta per la “Sensor Bar”
Porta per accessori nel Wii Remote
Porta opzionale USB per tastiera nel message board, nel Wii Shop Channel, e per navigare in Internet
Modulo wireless Mitsumi DWM-W004 WiFi 802.11b/g
Compatibile con USB 2.0 verso adattatre Ethernet LAN
Porta 'AV Multi Out'

Memoria interna:

512MB interni NAND di memoria flash
Capacità di salvataggio espandibile con SD and SDHC card memory (fino a 32GB)
Nintendo GameCube Memory Card (rchiesta per salvataggi della GameCube)
Slot per dischi ottici da 8cm di Nintendo GameCube Game Disc e da standard da 12cm.

Video:

Porta custom 'AV Multi Out', supporta, S-Video (solo NTSC), RGB SCART (solo PAL) e VGA utilizzando un adattatore di terze parti
risoluzione 480p (PAL/NTSC), 480i (NTSC) or 576i (PAL/SECAM), standard 4:3 e 16:9
anamorphic widescreen

Audio:

Principale: Stereo– Dolby Pro Logic II-capable

Controller: speaker interno

Consumi:

18 watts quando è accesa

9.6 watts in standby con connessione WiiConnect24

1.3 watts in standby

2.1.2 Wii Remote

Fig. 2.3 – Wiimote

Il Wii Remote, spesso chiamato WiiMote, è il controller principale della Nintendo Wii. Una delle caratteristiche fondamentali del Wii Remote è il "motion sensing", ovvero sensibilità al movimento, che permette all'utente di interagire e manipolare gli oggetti presenti sullo schermo attraveso il riconoscimento dei gesti grazie all'impiego di accelerometri e sensori ottici. Un'altra importante caratteristica è la porta di espansione, posta nel suo lato inferiore, la quale permette il collegamento di diverse periferiche. Una di queste è il *Nunchuk*, composto da un joystick analogico e due pulsanti dorsali; dotato anch'esso di accelerometri, viene collegato tramite cavo. È possibile anche collegare altre periferiche come: Wii Zapper e Wii Wheel.

Il Wii Remote, si utilizza con una sola mano, ed a differenza delle console precedenti, che impiegavano l'uso di un gamepad tradizionale, è basato su un controllo remoto. Ciò permette di rendere maggiormente intuitiva la sensibilità al movimento, e come un telecomando è dotato di un puntatore. Questa innovazione ha lo scopo di voler attirare un pubblico più vasto, che non necessariamente debba essere composto dai soli amanti delle console.
Il corpo del telecomando misura 148 mm in lunghezza, 36.2 mm di larghezza e 30.8 mm di spessore.
Analizzando la parte anteriore si possono notare i diversi pulsanti: il pad direzionale nella parte superiore, il succesivo pulsante "A", e successivamente i pulsanti "-", "HOME", "-", e a seguire i tasti "1" e "2". Al di sotto di questi si trovano quattro led di colore blu. Quando l'utente non lo utilizza per giocare, alla pressione di qualunque tasto, i led si illumineranno per indicare il livello della batteria del controller. Nella parte anteriore, si trova un solo pulsante "B", di cui la forma assomiglia a quella di un grilletto.

2.1.3 Funzionalità del WiiMote

Il *WiiMote* presenta diverse funzionalità che lo contraddistingue dai controller delle altre console. Difatti comunica con la Nintendo Wii per via wireless, grazie alla tecnologia Bluetooth a raggio corto, con la quale è possibile gestire fino a quattro controlli ad una distanza massima di circa 10 metri. Ma ciò che lo caratterizza ancora di più è l'accelerometro lineare a tre assi. Questo tipo di accelerometro è in grado di rilevare movimenti rapidi, come ad esempio la battuta del tennista, e integra i dati del sensore ottico. In particolare chi fornisce il riferimento assoluto dei movimenti è proprio il sensore ottico IR (Infrarossi), una camera CMOS da 1.3 mpixel che legge il riferimento alla *Sensor Bar*. Quest'ultima è una barra di sensori infrarossi, in dotazione con la console *Nintendo Wii*, composta due gruppi di led IR accesi fissi ad una distanza di 20 cm, non sono visibili dall'occhio umano.

La distanza tra il *WiiRemote* ed il monitor si ricava in base alla distanza apparente tra i due gruppi di led, mentre l'inclinazione viene ricavata, con una buona precisione, in base all'angolo rispetto al sensore ottico.

L'accelerometro montato sulla *WiiMote* percepisce accelerazioni in un range di +/- 3° con 8 bit per asse e una frequenza di aggiornamento di 100 Hz.

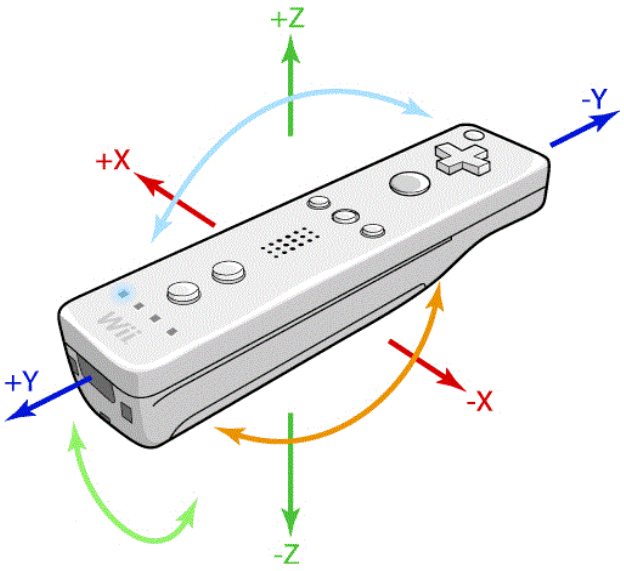

Fig. 2.4 – Rappresentazione degli assi dell'accelerometro del Wiimote

A scopo del seguente elaborato la *Sensor Bar*, in dotazione con la console *Nintendo Wii*, non verrà impiegata nello sviluppo dell'elaborato.

2.2 DirectSound

DirectSound fa parte dei componenti software delle librerie DirectX, in dotazione su i computer che utilizzano, come sistema operativo Windows XP e precedenti.

Questa fornisce un'interfaccia diretta tra le applicazioni e i driver della scheda audio, permettendo così di riprodurre suoni e musica. Inoltre sono presenti delle funzionalità aggiuntive, quali la registrazione dei suoni e il mixaggio; applicazioni di effetti come riverbero, delay, flanger; audio posizionale in 3D utilizzando suoni registrati da un microfono o altro tipo di input e controllando gli effetti nella fase di acquisizione del suono.

Questa sua caratteristica di riprodurre il suono in 3D, ha aggiunto una nuova dimensione ai videogiochi. Difatti prevede la possibilità di modificare uno script musicale in tempo reale, ad esempio, quando l'azione del gioco si riscalda il ritmo della musica potrebbe accelerare.

DirectSound permette anche a diverse applicazioni, di condividere l'accesso alla scheda audio, simultaneamente; e dopo molti anni di sviluppo, DirectSound è divenuto oggi un'*API* (Application Programming Interface) molto evoluto, che fornisce diverse funzionalità molto utili, come la possibilità di riproduzione multicanale ad alta definizione.

I suoni possono essere modificati da effetti tramite diversi *buffer di ingresso* (Secondary Sound Buffers), dopodiché vengono mixati insieme nell'unico *buffer di uscita* (Primary Sound Buffer).

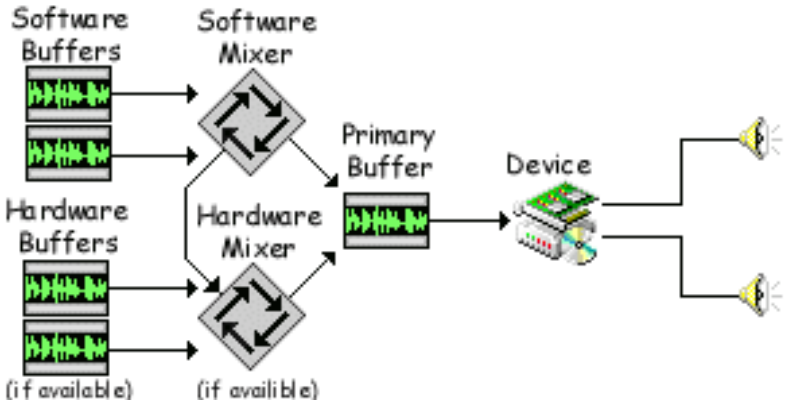

Fig. 2.5 – Architettura DirectSound

2.3 Il linguaggio C#

Il linguaggio di programmazione C# (pronunciato *See Sharp*) è un tipo di linguaggio object-oriented, sviluppato da Microsoft all'interno dell'iniziativa .NET.
La sintassi di questo linguaggio prende spunto da quella del medesimo autore del Delphi, ovvero Anders Hejlsberg, del C++, di Java e di Visual Basic per gli strumenti di programmazione visuale.
Il nome C# fu ispirato dalla notazione musicale, dove il simbolo #, il diesis, indica l'incremento della nota di un semitono, un po' come nel C++, in cui il simbolo ++, indica l'incremento di una variabile di 1.
È stato adottato l'uso di questo linguaggio per lo sviluppo del progetto, per la sua semplicità di utilizzo e di sintassi, e per l'impiego delle specifiche librerie, che si possono trovare sul sito:

http://www.brianpeek.com/blog/pages/wiimotelib.aspx

Queste librerie, create da Brian Peek, sono state molto utili per lo studio del codice relativo al *Wii Remote*.

3. Progetto Controllo di Riff Per Chitarra Tramite WiiMote

3.1 Descrizione Interfaccia Utente

Per la realizzazione di quest'interfaccia è stato utilizzato il linguaggio di programmazione C# con l'ausilio di Visual Studio 2008, librerie DirectX, DirectSound, e il controller WiiMote già visto in precedenza.
L'interfaccia grafica progettata dà la possibilità all'utente di aver controllo su diversi parametri di un riff per chitarra, mediante l'utilizzo del WiiMote.
In dotazione all'interfaccia è presente una modesta libreria di suoni, ovvero diversi tipi di riff che messi in sequenza potranno creare un intero brano.
Il form creato è composto da pulsanti, trackbar e una listbox.

Fig. 3.1 – Interfaccia Guitar Riff Controller

- **Pulsante OPEN RIFF:**

Il pulsante Open Riff, viene utilizzato per caricare un riff nella listbox. Viene aperta una finestra di dialogo che permette di sfogliare nelle varie cartelle presenti nel computer e successivamente si seleziona il file da aprire. (Si consiglia di utilizzare i file presenti nella libreria incorporata). Il codice implementato permette di caricare solo file di formato *.wav.

- **Pulsante CLEAR:**

Questo pulsante permette di cancellare tutti gli elementi presenti nella listbox, e dando l'opportunità di poter cominciare una nuova sessione di controllo.

- **Pulsante PLAY/PAUSE:**

Pulsante per la gestione della funzione *buffer.Play().*

Questo pulsante una volta premuto manderà in play, ovvero riprodurrà, il riff solo se selezionato. Difatti verrà eseguita l'operazione di inizializzazione del *SecondaryBuffer* e quindi successivamente riempito con il file caricato nella listbox. Nel momento in cui esso viene ripremuto, la riproduzione si fermerà, e verrà mantenunta la posizione in cui l'esecuzione è stata interrotta. Alla successiva pressione la riproduzione riprenderà normalmente dal punto in cui era stato fermato.

- **Pulsante STOP:**

Pulsante per la gestione della funzione *buffer.Stop().*

Una volta premuto la riproduzione del file in output cessa.

- **Pulsante NEXT RIFF:**

Questo pulsante permette di scorrere in avanti gli elementi nella listbox, in modo da poter selezionare il riff desiderato e cominciare quindi la riproduzione. Se premuto in

fase di Play permetterà di passare al riff successivo e alla sua riproduzione senza interruzione del suono.

- **Pulsante PREVIOUS RIFF:**

Pulsante simile al pulsante NEXT RIFF. La differenza consiste nella possibilità di scorrere gli elementi della listbox precendenti a quello selezionato.

- **LISTBOX:**

La listbox ha la funzione di elencare i riff caricati dall'utente per la successiva riproduzione. Viene visualizzato il nome principale del riff e l'utente deciderà mediante i radio button quale variazione di riff suonare.

- **VOLUME CONTROL:**

La Trackbar in questione ha la funzione di controllare l'intensità del file audio in riproduzione. Esso può essere modificato dall'utente in tempo reale mediante il trascinamento del cursore della trackbar con il mouse o con il pulsante 2 del Wiimote. Si possono ottenere valori che partono da un valore minimo di -3000, con progressione di intervalli di 200, ad un valore massimo di 0, ovvero la massima intensità consentità da DirectSound.

- **BENDING/PITCH CONTROL:**

Questa Trackbar controlla il parametro relativo al pitch del suono, sfruttando le diverse velocità di lettura del buffer dell'audio in output, ovvero trascineremo il cursore verso un parametro basso, la velocità di lettura diminuirà e di conseguenza anche l'altezza del suono, e viceversa. Come per il Volume Control può essere modificato dall'utente con il mouse o il WiiRemote. Il valore predefinito è impostato su 44100, in modo tale da riprodurre il file a velocità naturale 44100Hz. Il valore minimo, spostando il cursore verso sinistra, sarà 28200, e il massimo, verso destra, 60000, con progressione di 4100.

- **TONALITY:**

Questa trackbar, permette di trasporre tutti i riff presenti nella list-box, scegliendo quindi la tonalità in cui suonare. Comprende tutte le note e tutte le alterazioni.

- **Radio Button:**

I quattro radio button hanno la funzione di selezionare una delle modalità di riproduzione del riff selezionato. Difatti nella libreria dei riff troveremo lo stesso riff eseguito in quattro modi diversi in modo che l'utente possa interagire e utilizzare i parametri a proprio piacimento in tempo reale, e senza interruzioni.

DISTORTION: Il primo dei radio button nel form, permette di riprodurre il riff selezionato con effetto di chitarra elettrica distorta.

PALM MUTE: Il secondo radio button presente, riproduce il riff selezionato con effetto *palm mute* di una chitarra elettrica distorta o pulita a seconda della libreria utilizzata.

CLEAN GUITAR: il terzo radio button, riproduce il riff selezionato suonato da una chitarra elettrica con un suono pulito.

WAH: il quarto ed ultimo radio button, riproduce il riff selezionato suonato da una chiatarra elettrica distorta su cui è applicato l'effetto *wah-wah*.

3.2 Configurazione WiiMote

Il procedimento per connettere il controller *Wiimote* al computer mediante la tecnologia Bluetooth è molto semplice. Una volta attivato il Bluetooth del vostro PC tramite apposita chiavetta o, come in questo caso, se già incorporato, è possibile procedere all'abbinamento dei due dispositivi, mantendo premuti i pulsanti 1 e 2 contemporaneamente, e quindi seguire le istruzioni della procedura guidata offerta da Windows.

Il *WiiMote* verrà riconosciuto col nome Nintendo RVL-CNT-01, e facendo doppio click sull'icona relativa (vedi immagine), il computer inizierà a ricercare il device da abbinare. Si dovrà quindi premere i pulsanti 1 e 2 fino al termine della connessione tra i due dispositivi, e quindi essere pronti al loro utilizzo e allo scambio di informazioni.

Fig. 3.2 – Finestra di connessione wiimote

3.3 DirectSound

Per lo sviluppo di questo progetto, viene utilizzato l'API *DirectSound*, grazie alle proprietà offerte per la gestione di file audio.
Implementando DirectSound con linguaggio di programmazione C#, si deve prima di tutto aggiungere il riferimento al progetto, mediante la seguente stringa:

```
using Microsoft.DirectX.DirectSound;
```

in modo così, da poter richiamare i metodi propri e specifici di *DirectSound.*
Il passo successivo sarà quello di inizializzazione dell'API, ovvero originariamente si crea un device, che si occupa di gestire gli eventi di *DirectSound*, che nel nostro caso viene chiamata `dSound`:

```
dSound = new Device();

dSound.SetCooperativeLevel(button1,CooperativeLevel.Priority);
```

La stringa successiva descrive invece un'importante opzione per la creazione del device creato, il `SetCooperativeLevel`. Il parametro utilizzato, `Priority`, nel caso di questo elaborato, sfrutta, quando è possibile, l'accelerazione hardware.
Una volta inizializzato il device, si può passare all'inizializzazione dei flags, indispensabili per attivare i controlli relativi al volume e alla frequenza, che, nell'interfaccia, verranno gestiti mediante le Trackbar:

```
d.Flags = BufferDescriptionFlags.ControlVolume;
          BufferDescriptionFlags.ControlFrequency;
```

Per la riproduzione dei file audio, come già detto verrà inizializzato il Secondary Buffer. Nel caso di questo elaborato vengono utilizzati quattro Secondary Buffer

gestiti dall'array *Sound*. Quest'ultimo permette quindi di bufferizzare i quattro modi di esecuzione del riff selezionato, in modo da agevolare la riproduzione istantanea senza essere soggetta a ritardi dovuti al caricamento, problema incontrato durante lo sviluppo del software. Difatti inizialmente si era progettata l'interfaccia con l'impiego di un solo *SecondaryBuffer*, il quale veniva riempito con il file selezionato all'istante in cui l'utente dava il comando di selezione della modalità di esecuzione. Si è quindi optato per la scelta di quattro buffer indipendenti come si può osservare nel sequente codice:

```
SecondaryBuffer s1 = new
SecondaryBuffer(GetFullyPathedAudioFilename(lstbox.SelectedIndex,
1), d, dSound);
sound.Add(s1);
SecondaryBuffer s2 = new
SecondaryBuffer(GetFullyPathedAudioFilename(lstbox.SelectedIndex,
2), d, dSound);
sound.Add(s2);
SecondaryBuffer s3 = new
SecondaryBuffer(GetFullyPathedAudioFilename(lstbox.SelectedIndex,
3), d, dSound);
sound.Add(s3);
SecondaryBuffer s4 = new
SecondaryBuffer(GetFullyPathedAudioFilename(lstbox.SelectedIndex,
4), d, dSound);
sound.Add(s4);
```

Per la riproduzione si utilizzerà il comando:

```
sound[Suono + 4 * Transpose + Selection * 48].Play(0,
                BufferPlayFlags.Looping);
```

Dove `Suono + 4 * Transpose + Selection * 48` è un parametro che gestisce i buffer in lettura corrispondente al riff selezionato, al modo di esecuzione dello stesso e alla tonalità impostata dalla trackbar. La riproduzione sarà continua `BufferPlayFlags.Looping` in modo che l'utente non debba preimpostare il numero di ripetizioni dell'esecuzione del riff, ma può deciderlo a suo piacimento al momento dell'utilizzo dell'interfaccia.

Per fermare la riproduzione si userà:

```
sound[Suono + 4 * Transpose + Selection * 48].Stop();
```

questo comando interromperà la riproduzione del file contenuto nel buffer in lettura. Un altro particolare comando, `PlayPosition`, permette di riprodurre il file selezionato da una precisa posizione temporale, nello specifico, per ottenere continuità nell'esecuzione durante il passaggio tra un tipo ed un altro, senza che il riff venga eseguito dall'inizio. Da questo punto viene quindi introdotta la variabile `pos`, che indica la posizione del lettore nel buffer, utilizzata per settare la riproduzione del riff selezionato con un tipo preciso.

```
int pos = sound[Suono+4*Transpose+Selection* 48].PlayPosition;
sound[Suono + 4 * Transpose].SetCurrentPosition(pos);
```

3.4 Libreria Di Riff

Un altro importante strumento adottato per lo sviluppo di questo elaborato è rappresentato dalla *Libreria Di Riff*. Questa libreria è composta da diversi file audio, raggruppati in cartelle. Ogni file audio contiene la sequenza di note di un riff, eseguite e registrate appositamente in occasione di questo elaborato. Il modello di chitarra adoperato per le registrazioni è una *Fender Stratocaster Mexico*, combinata con pedale *DigiTech RP350*. Per un corretto uso del software è stato necessario registrare ogni riff nelle diverse modalità di esecuzione, ovvero: con effetto distorto, con tecnica palm mute, con chitarra pulita e con effetto wah-wah. Ogni riff successivamente è stato suonato nelle trasposizioni possibili in modo che l'utente possa avere pieno controllo dei parametri e quindi divertirsi a proprio piacimento.

3.5 Utilizzo del Guitar Riff Controller

Dopo aver analizzato i componenti principali del progetto, quali il *DirectSound* e il controller *WiiMote*, e le loro funzioni, si può finalmente parlare di come queste tecnologie vengano impiegate per dar vita all'interfaccia *Guitar Riff Controller*.
Come già spiegato in precedenza, lo scopo del progetto è quello di creare un'interazione tra utente ed interfaccia mediante l'utilizzo del innovativo controller *Wii Remote*, o semplicemente col mouse.
Con quest'ultimo l'utilizzo dell'interfaccia è decisamente intuitivo, quindi si andrà ad analizzare esclusivamente il funzionamento del *Wiimote.*
Le funzioni assegnate ai diversi pulsanti del controller vengono descritte all'interno del metodo chiamato "`wiimote_WiimoteChanged ()`"dal quale, si estrae lo stato del *Wiimote* tramite la seguente stringa:

```
WiimoteStatews = e.WiimoteState;
```

Lo stato del controller si modifica nel momento in cui l'utente dà qualunque tipo di input, mediante, ad esempio, una semplice pressione di un tasto.
Questi pulsanti vengono gestiti all'interno del codice mediante una serie di *cicli if*, che spesso rimandano a bottoni inizializzati in precedenza all'interno del form, quindi gestibili anche attraverso l'utilizzo del mouse, come nel seguente esempio:

```
if (ws.ButtonState.B && !bPressed)
            {
                bPressed = true;
                btnplay_Click(this, null);
            }
```

In questo caso si prende in considerazione l'evento relativo alla pressione del tasto B nel controller Wiimote. Questo rimanda al pulsante Play/Pause (`btnplay`), il quale gestisce l'evento play o pausa in successione del click su di esso:

```
private void btnplay_Click(object sender, EventArgs e)
{

   if (Selection == -1)
   {
       MessageBox.Show("Apri o Seleziona un Riff!");
   }

   else
   {
       if (isPlay == true)
       {

            sound[Suono+4*Transpose+Selection*48].Stop();
            isPlay = false;
       }

       else
       {
            sound[Suono].Play(0,BufferPlayFlags.Looping);
            isPlay = true;
            sound[Suono].Volume = trkVolume.Value = vol;
            sound[Suono].Frequency=trkPitch.Value=pit=
                                                    44100;
       }
    }
 }
```

Si può osservare che questa funzione gestisce anche il caso in cui non vi è presente alcun riff nella list-box o semplicemente non è stato selezionato avvertendo l'utente con una `MessageBox.Show`:

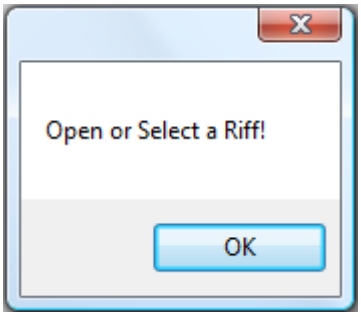

Fig. 3.3 – Messaggio di avviso

La variabile `bPressed` nella porzione di codice relativa alla pressione del pulsante B del controller, insieme alle seguanti variabili:

```
bool plusPressed = false;
bool minusPressed = false;
bool bPressed = false;
bool aPressed = false;
bool homePressed = false;
```

è una variabile di tipo booleano.

Queste variabili, poste inizialmente a `false`, sono state aggiunte in un secondo momento perché si creavano dei problemi durante l'esecuzione del programma in relazione ai tasti premuti. Ovvero, di default, nel momento in cui si preme un tasto questo invia l'input per tutto l'arco di tempo in cui questo è premuto, cessando quindi di mandare informazioni nel momento del rilascio. Essendo molto sensibile, anche una pressione minima veniva considerata come una pressione prolungata. Nell'esempio dell'uso del tasto B, il pulsante del Play/Pausa, in cui con la prima pressione si avvia la riproduzione e con la seconda si ferma, era possibile che al momento della pressione non accadesse nulla o si sentisse un suono di durata pressochè minima per poi bloccarsi instantaneamente, proprio per la sensibilità del controller nel rilevare la pressione del tasto.

Le variabili booleane utilizzate sono in grado invece di aggirare questo ostacolo, in quanto, inizialmente poste a "false", diverranno vere(true) solo successivemente alla

pressione del tasto, dando così l'input una volta soltanto, anche nel caso in cui si mantenga premuto il pulsante. Questo tipo di soluzione è stata presa per i tasti A, B, Home, -, + del controller wii.

GESTIONE RADIOBUTTON:

Si può osservare, invece, che i tipi di esecuzione sono stati assegnati ai tasti del pad direzionale, e ogni pulsante rimanda all'inizializzione del radiobutton relativo.

La pressione del pad con direzione Up, esegue in modalità distorta, Right in Palm Mute, Down in modalità pulita e infine Left per l'effetto wah-wah.

```
if (ws.ButtonState.Up && Suono != 0)
{
    try
    {
        timerOff = true;
        posSound = sound[Suono + 4 * Transpose+Selection*
                                                48].PlayPosition;
        Suono = 0;
        rdBtn_CheckedChanged();
        timerOff = false;
    }

    catch (ArgumentOutOfRangeException)
    {
    }
}

if (ws.ButtonState.Right && Suono != 1)
{
    try
    {
       timerOff = true;
       posSound = sound[Suono + 4 * Transpose+Selection*
                                                48].PlayPosition;
       Suono = 1;
       rdBtn_CheckedChanged();
       timerOff = false;
    }

    catch (ArgumentOutOfRangeException)
    {
    }
```

```
if (ws.ButtonState.Down && Suono != 2)
{
   try
   {
      timerOff = true;
      posSound = sound[Suono + 4 * Transpose+ Selection*
                                          48].PlayPosition;
      Suono = 2;
      rdBtn_CheckedChanged();
      timerOff = false;


   }

   catch (ArgumentOutOfRangeException)
   {
   }
}

if (ws.ButtonState.Left && Suono !=3)
{
   try
   {
      timerOff = true;
      posSound = sound[Suono + 4 * Transpose+ Selection*
                                           48].PlayPosition;
      Suono = 3;
      rdBtn_CheckedChanged();
      timerOff = false;
   }

   catch (ArgumentOutOfRangeException)
   {
   }
 }
```

La variabile Suono, può assumere valori che vanno da 0 a 3, in quanto si riferiscono ai quattro modi di esecuzione del riff, dove 0 è riferito alla modalità distorta, 1 a quella palm mute, 2 al pulito e 3 al wah-wah. La variabile posSound, permette di gestire la posizione della lettura nel buffer durante la riproduzione, in modo tale che, al momento della pressione di uno dei tasti del pad direzionale, l'esecuzione non cessi e in tempo reale venga cambiato il modo di suonare.

La variabile timerOff invece verrà presa in considerazione più avanti, quando si tratterà dell'evento Timer_Tick.

Il ciclo try catch, è stato inserito in un secondo momento, in seguito all'uscita del seguente messaggio:

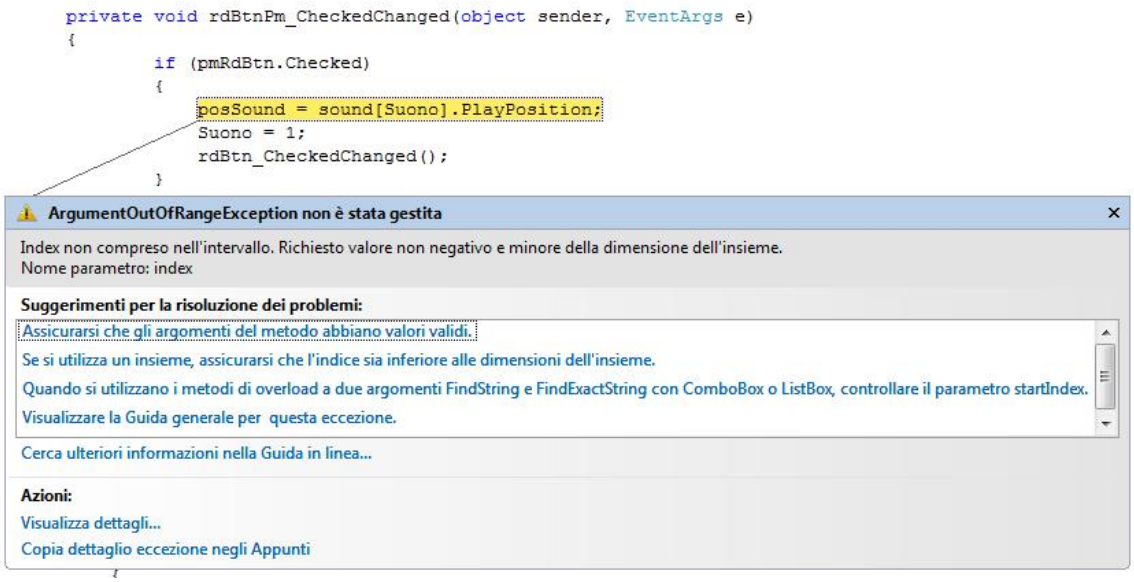

Fig. 3.4 – Messaggio di eccezione non gestita

Questo errore si riferisce al momento in cui si preme uno dei radiobutton senza aver prima caricato nessun file nella listbox. Questa funzione difatti va a leggere la posizione di lettura nel buffer. Essendo in questo caso vuoto, il programma non riesce ad eseguire l'operazione, reagendo quindi con la visualizzazione di un messaggio nel quale viene indicato il nome dell'eccezione non gestita dal programma. Una volta individuato il tipo di eccezione si inserisce nella funzione catch la stringa `ArgumentOutOfRangeException,` risolvendo quindi il problema.

GESTIONE BENDING/VOLUME:

Per la gestione dell'effetto bending e di controllo del volume, viene utilizzata la proprietà che ha reso originale e innovativo il WiiMote rispetto ai controller precedenti, e dalle altre console: l'accelerometro.

Si potrà quindi gestire questi parametri in base al movimento della mano in modo molto intuitivo.

```
float x = e.WiimoteState.AccelState.Values.X;
float y = e.WiimoteState.AccelState.Values.Y;
```

Prima di tutto sarà necessario dichiare due variabili, *x* e *y*, a cui verranno assegnati i valori corrispondenti al movimento lungo le due dimensioni: orizzontale, asse delle ascisse (x), e verticale, asse delle ordinate (y).

```
if (ws.ButtonState.One)
{
    pit = (int)((1 - y) * 15900 + 30000);
}

else
{
    pit = 44100;
}
```

In questo caso al seguito della pressione del pulsante 1 del controller, verrà rilevato lo spostamento verticale del wiimote in modo da alzare, nel caso in cui venga ruotato verso l'alto, e abbassare, nel caso in cui venga ruotato verso il basso, il valore del pitch, gestito dalla variabile *pit*. La funzione `(int)` renderà il valore della variabile pit, da *float*, con la virgola, a *int*, intero, in modo da semplificare la gestione. Come mostra il codice nella parte relativa all'*else*, nel caso in cui il pulsante 1 non venga premuto, il valore del pitch tornerà ad essere quello standard per la riproduzione, ovvero a 44100.
Di comportamento simile è quindi la gestione del volume:

```
if (ws.ButtonState.Two)
{
    vol = (int)((x - 1) * 1500);
}
```

In quest'altro caso, in seguito alla pressione del tasto 2, si gestirà il valore relativo all'intensità sonora, identificato dalla variabile *vol*. A differenza del costrutto per il

pitch, si utilizzeranno i valori dell'asse orizzontale x, in modo da rimandare all'utilizzo di una manopola, quale può essere il potenziometro di una chitarra. Difatti il movimento rotatorio del controller in senso antiorario abbasserà il volume, al contrario il movimento orario aumenterà l'intensità sonora.

TIMERTICK:

Durante la fasi iniziale dell'implementazione del controller Wii nell'applicazione, si osservava che, nel debug, in seguito alla gestione delle funzioni del form con il Wiimote o anche all'utilizzo delle funzioni relative all'accelerometro a tre assi si presentava il seguente errore:

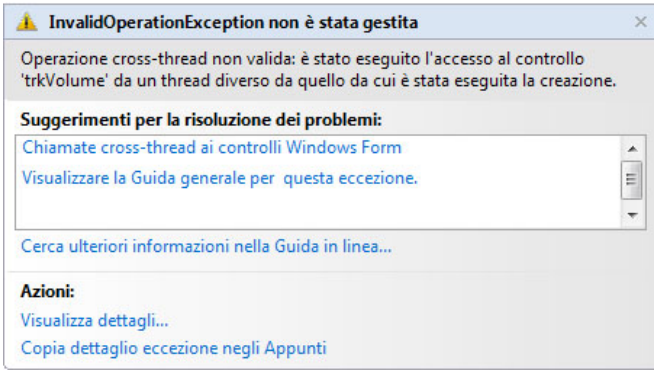

Fig. 3.5 – Messaggio di eccezione non gestita

Questo problema impediva l'interazione tra il Wii Remote e il Windows Form, rendendo così ineseguibile la gestione di qualunque elemento del form quali le trackbar, i pulsanti e la selezione del radiobutton.

La soluzione scelta per aggirare questo ostacolo è stata trovata con l'aggiunta, al form, di un *timer*.

Questo timer è un elemento del form, che ha la funzione di eseguire una determinata istruzione, in un intervallo di tempo impostato nelle proprietà del timer.

Le evento relativo alla gestione delle azioni che il timer eseguirà sarà `timer_Tick(object sender, EventArgs e)`, ed è stato fissato un intervallo di 10 ms.

A questo punto è possibile gestire gli elementi del form implementati, per mezzo del wiimote, vediamo come:

```
if (timerOff)
    return;
if (Selection < 0)
    Selection = 0;
if (Selection > lstbox.Items.Count - 1)
    Selection = lstbox.Items.Count - 1;
lstbox.SelectedIndex = Selection;

trackBarTonal.Value = Transpose;

if (Suono == 0)
{
    distRdBtn.Checked = true;
}

if (Suono == 1)
{
    pmRdBtn.Checked = true;
}

if (Suono == 2)
{
    cleanRdBtn.Checked = true;
}

if (Suono == 3)
{
    wahRdBtn.Checked = true;
}

if (vol > 0)
{
    vol = 0;
}

if (vol < -3000)
{
    vol = -3000;
}
if (sound.Count > 0)
    sound[Suono + 4 * Transpose+Selection*48]
                                    .Volume=trkVolume.Value=vol;

if (pit > 60000)
{
    pit = 60000;
}

if (pit < 30000)
{
```

```
        pit = 30000;
    }


    if (sound.Count > 0)
        sound[Suono + 4*Transpose+Selection*48]
                                    .Frequency=trkPitch.Value= pit;
```

la variabile *timerOff*, può assumere due valori, *true* o *false*. Inizialmente viene impostata sul false, ma in alcuni casi, di cui alcuni visti in precedenza, si ha la necessità di fermare il timer in modo da eseguire le istruzioni relativi ad eventi specifici, per poi essere successivamente riattivato.
La variabile *Transpose*, assumerà i valori relativi allo spostamento della trackbar della trasposizione, come spiegato nella gestione del traspositore.

GESTIONE TRASPOSITORE:
Un'altra importante funzione dell'interfaccia per il controllo dei riff è il traspositore. Questo parametro viene controllato dalla trackbar relativa, in modo da poter eseguire il riff nella tonalità desiderata e se sfruttata durante l'esecuzione si possono creare modulazioni, difatti la trasposizione avverà in tempo reale.
Per modificare il parametro tramite mouse, basterà semplicemente trascinare il cursore sulla tonalità desiderata; se si volesse utilizzare il *WiiMote*, si premerà il pulsante A, per incrementare, il pulsante Home, per decrementare:

```
if (!ws.ButtonState.A)
    aPressed = false;
if (ws.ButtonState.A && !aPressed)
{
    try
    {
        aPressed = true;
        timerOff = true;
        posSound = sound[Suono + 4 * Transpose+Selection*48]
                                                .PlayPosition;
        if (Transpose < 11)
            Transpose++;
        timerOff = false;
    }
```

```
        catch (ArgumentOutOfRangeException)
        {
        }
    }

    if (!ws.ButtonState.Home)
        homePressed = false;
    if (ws.ButtonState.Home && !homePressed)
    {
        try
        {
            homePressed = true;
            timerOff = true;
            posSound = sound[Suono + 4 * Transpose+Selection*48]
                                                    .PlayPosition;
            if (Transpose > 0)
                Transpose--;
            timerOff = false;
        }

        catch (ArgumentOutOfRangeException)
        {
        }
    }
```

Come spiegato nella parte relativa al timer, la variabile *Transpose*, assumerà i valori impostati dallo spostamento del cursore della trackbar, in modo che venga eseguito il riff con la tonalità desiderata dall'utente.

Anche in questo caso, come nella gestione dei radio button, si utilizza una *try catch* per l'eccezione apparsa nel momento della pressione di uno dei pulsanti A, Home nel momento in cui non siano ancora stati caricati i file audio.

CARICAMENTO DEI RIFF:

I riff che possono essere gestiti dall'interfaccia devono avere delle caratteristiche precise, ovvero un nome che li contraddistingue con una struttura fissa che può essere ad esempio questa:

1BDBD_1_00

Nel momento in cui il riff viene selezionato nell'OpenFileDialog, l'applicazione leggerà il nome del file in cui sono scritte delle informazioni precise.

```
private void AddAudioFilename(String pFilename)
{
    lstbox.Items.Add(1);
    int rowNum = lstbox.Items.Count - 1;
    int pos = Path.GetFileName(pFilename).IndexOf('_');
    lstbox.Items[rowNum]=Path.GetFileName(pFilename).Substring(0,
                                                              pos);
    mSoundFilePaths.Add(pFilename.Remove(pFilename.Length - 8, 8));
}

private String GetFullyPathedAudioFilename(int pIndex, int tipo)
{
    String filename = "";
    if ((pIndex >= 0) && (pIndex < mSoundFilePaths.Count) && (pIndex
                                                  < lstbox.Items.Count))
    {
        filename = (String)(mSoundFilePaths[pIndex]);
    }

    filename += tipo.ToString() + "_";

    filename += Transpose.ToString().PadLeft(2, '0') + ".wav";

    return filename;
 }
```

Ciò che avviene in questa porzione di codice è appunto un'analisi del nome del file selezionato. Presa in considerazione la struttura 1BDBD_1_00, dell'esempio, verrà visualizzato nella listbox dell'interfaccia esclusivamente la parte che precede il primo underscore. Quel nome si riferisce, in questo caso, al primo riff di quella libreria.

Il secondo parametro, ovvero quello compreso tra i due underscore, in questo caso 1, rappresenta in quale modalità il riff è stato registrato. Esso può essere quindi un numero compreso tra 1 e 4, di cui 1 è la modalità distorta, 2 la modalità palm mute, 3 la modalità pulita, e infine 4 la modalità wah-wah.

Le ultime due cifre si riferiscono alla tonalità, di cui 00 è il valore del "C" (DO), i valori che si possono trovare quindi vanno da 00 a 11.

Durante la selezione del riff da aprire, sarà indifferente la scelta sul quale cliccare, dato che verranno caricati tutti i riff che avranno la prima porzione del nome identica. Nella listbox però sarà presente un solo file. Questo per evitare di riempire inutilmente la listbox, e soprattutto per rendere il più efficiente possibile l'utilizzo dell'interfaccia e la selezione dei riff da riprodurre durante un'esecuzione.

Conclusioni e sviluppi futuri

Lo sviluppo dell'interfaccia "Guitar Riff Controller" ha portato al raggiungimento degli scopi prefissati coinvolgendo quindi tecnologie diverse come DirectSound, motore vero e proprio dell'applicazione, e il Wii Remote, controller per i videogiochi Wii. Non solo questo telecomando prodotto da Nintendo può essere impiegato per scopi ludici o musicali, come in questo caso, ma può essere di ispirazione per molti altri utilizzi nell'ambito della gestione di oggetti a distanza, multimediali e non.

Per questo elaborato si è realizzata una struttura minimale dell'interfaccia che potrà venire ulteriormente sviluppata. Per ampliare le sue funzioni ci si potrebbe indirizzare sull'espansione delle librerie dei riff, sia per quanto riguarda i brani che si vorranno eseguire, che per il suono vero e proprio, il timbro della chitarra.

Una possibile idea può essere l'aggiunta del controllo del tono della chitarra come filtro passa-basso, dando modo all'utente di scegliere quale dei pick-up vorrà far suonare.

Aumentando il numero di elementi del form, si potranno sfruttare le espansioni del controller WiiMote offerte dalla Nintendo, come il Nunchuck, in modo da facilitare la gestione dei diversi pulsanti che sul controller principale sono in numero limitato.

Un altro elemento potrà essere la funzione di composizione automatica di riff, in modo da creare brani nuovi e originali.

Si potranno anche aggiungere effetti sonori come delay, chorus, balance e molte altre idee che possono arricchire l'interfaccia.

Le innovazioni continuano a stupirci e la tecnologia è pronta ad offrirci ogni giorno qualcosa di nuovo, di originale, di cui, spesso, inizialmente, si mette in dubbio l'utilità, ma successivamente se ne rimane affascinati e rapiti tanto da non poterne più fare a meno.

Bibliografia

- BRACKETT DAVID, “Interpreting Popular Music” Berkeley, California: University of California Press, 2000

- BRADLEY L. JONES, “Sams Teach Yourself C#™ in 21 Days”, Sams Publishing, 2004

- CHUNG LEE JOHNNY, “Hacking the Nintendo Wii Remote”, IEEE CS. - 2008 IEEE. PERVASIVE computing 39 – http://www.cs.cmu.edu/~15-821/CDROM/PAPERS/lee08.pdf

- JAMES FOXALL, WENDY HARO-CHUN, “Sams Teach Yourself C#™ in 24 Hours”, Sams Publishing, 2002

- MIDDLETON RICHARD, “Studying Popular Music”, Philadelphia: Open University Press, 1990/2002

- PEEK BRIAN and FERNANDEZ DAN, “Coding4fun: 10 .Net Programming Projects for Wiimote, Youtube, World ofWarcraft, and More”, by O'Reilly Media, 2008

- PEEK BRIAN, “WiimoteLib - .NET Managed Library for the Nintendo Wii Remote”, MSDN Alliance, 2008

- RIKKY ROOKSBY, “Riffs: How To Create And Play Guitar Riffs”, Backbeat, 2002

- JOHN SHARP, “Microsoft Visual C# 2008 Passo per Passo”, Mondadori Informatica, 2008

- MSDN Library, Windows Developer Center, DirectX Software Development Kit, 2009