Istituto di Scienza e Tecnologie dell'Informazione

Consiglio Nazionale delle Ricerche

Massimo Magrini

Leonello Tarabella

Graziano Bertini

Un exhibit interattivo dimostratore del Sistema TAU2/TAUMUS

2020

1 Introduzione

Il Terminale Audio TAU2 è stato il primo sintetizzatore musicale polifonico e politimbrico realizzato in Italia funzionante in tempo reale sotto il controllo di un calcolatore general purpose operante in time-sharing.

Il TAU2 consisteva di due parti distinte: una unità digitale e una unità analogica. L'unità digitale era collegata al sistema IBM 370/167 e tramite il software TAUMUS da cui riceveva in data-streaming, una sorta di midi-file ante-litteram, i dati utilizzati per alimentare pilotare l'unità analogica. La generazione del segnale audio era basata sulla sintesi additiva tramite oscillatori sinusoidali analogici: la polifonia era costituita da dodici voci, distribuite su tre uscite audio con quattro note e sette armoniche ciascuna, con possibilità di variare ogni 1/100 di secondo i valori di ampiezza e durata di tutti i parametri.

Il sistema TAUMUS/TAU2, realizzato nella prima metà degli anni '70, è rimasto in funzione fino alla fine degli anni '80. Il TAU2 è ed intimamente legato alla figura di Pietro Grossi, il pioniere dell'Informatica Musicale in Italia che iniziò e condusse le sue ricerche presso il CNUCE di Pisa; Determinanti per la sua esperienza furono l'incontro con il direttore del CNUCE Guido Torrigiani (matematico, umanista e musicologo) ed il contesto tecnologico unico ed irripetibile esistente nella Pisa degli anni '60-70: è grazie infatti alle strutture che negli anni '50 avevano dato vita alla CEP (Calcolatrice Elettronica Pisana) - ancora in essere all'IEI negli anni '70 - e all'intervento del direttore G. Capriz e di F. Denoth, che fu possibile realizzare il Terminale AUdio TAU2 [1][2][3].

Grossi, impadronitosi delle tecniche di programmazione FORTRAN, realizzò in modo del tutto autonomo su sistema IBM370 l'applicazione TAUMUS che dava la possibilità di svolgere tutte le funzioni di codifica di brani musicali classici, di creare di strutture musicali algoritmiche originali e di operare trasformazioni di vario tipo come ad esempio: esecuzioni in modo retrogrado, inversione intervallare rispetto ad una nota, espansione e/o compressione intervallare, cambio di velocità esecutiva, ecc .

A Grossi è stata dedicata la monografia *"L'Informatica Musicale a Pisa", l'esperienza di Pietro Grossi*, edito dalla Pisa University press [4], nella quale sono descritti in dettaglio il progetto, le attività di ricerca, dimostrative e concertistiche effettuate in tutta Italia con il sistema. Nell'occasione del 50esimo dell'Informatica a Pisa che coincide con il 50esimo delle ricerche nel settore del Maestro Grossi, grazie anche all'esperienza nella realizzazione di installazioni analoghe realizzate in passato [5], è stato realizzato un dimostratore in grado di simulare in modo interattivo una tipica sessione al terminale come era condotta dal Maestro.

Il dimostratore è attivo presso il Museo degli Strumenti di Calcolo (Area dei vecchi macelli) con la presenza dell'originale unità digitale del TAU2 ed offre al visitatore la possibilità di caricare brani musicali da un ipotetico archivio, di operare alcune tipologie di trasformazione e di ascoltarne il risultato sonoro [6].

2 Architettura del sistema

L'assemblaggio del sistema HW del simulatore è stato concepito per richiamare all'estetica "retro" delle tipiche sessioni di lavoro al terminale di fine anni 70', inizio anni 80.

2.1 Hardware utilizzato

Per ragioni di comodità di utilizzo all'interno dell'esposizione si è deciso di basare tutto sulla piattaforma Raspberry PI, una micro-calcolatore Linux. Questo piccolo calcolatore, grazie alle sue ridotte dimensioni, è facilmente occultabili in installazioni interattive quali exhibit e simili. Non richiedendo operazioni di avvio e arresto manuali il suo utilizzo è perfetto in una situazione "museale". La potenza di calcolo del Raspberry è ovviamente limitata in confronto ad un calcolatore "standard", ma comunque sufficiente per il funzionamento dell'installazione. Queste le caratteristiche del Raspberry 3b, utilizzato nell'exhibit:

- CPU Quad Core da 1,2GHz Broadcom BCM2837 a 64bit
- GPU Broadcom VideoCore IV
- RAM 1GB
- BCM43438 WiFi e Bluetooth Bassa Energia (BLE) su scheda
- GPIO esteso a 40 pin
- 4 USB a 2 porte
- Uscita audio stereo a 4 poli con porta video composita
- HDMI a dimensioni piene
- Porta fotocamera CSI per connettere una fotocamera Raspberry Pi
- Porta per schermo DSI per connettere lo schermo touchscreen
- Ingresso Micro SD per il caricamento del sistema operativo e l'archiviazione dati
- Alimentazione Micro USB

L'uscita video del mini-computer è stata connessa ad un monitor CRT IBM dall'aspetto "vintage". Anche se di epoca leggermente più recente rispetto ai quelli utilizzati da Pietro Grossi nelle sue sessioni al TAU2, il monitor scelto ha comunque un'estetica molto simile, e comunque coerente con l'epoca. È stata recuperata anche una tastiera IBM dello stesso periodo, connessa al Raspberry tramite un apposito adattatore. Il monitor e la tastiera sono stati quindi posti affianco all'Unità Digitale del TAU2 nell'exhibit, in modo da dare l'impressione che il terminale controlli veramente la macchina (fig.1). Il mouse non è stato utilizzato, privilegiando un'interfaccia a caratteri volutamente "retro". Infine, l'uscita audio del Raspberry è stata connessa ad una cuffia tramite un piccolo amplificatore audio per cuffie

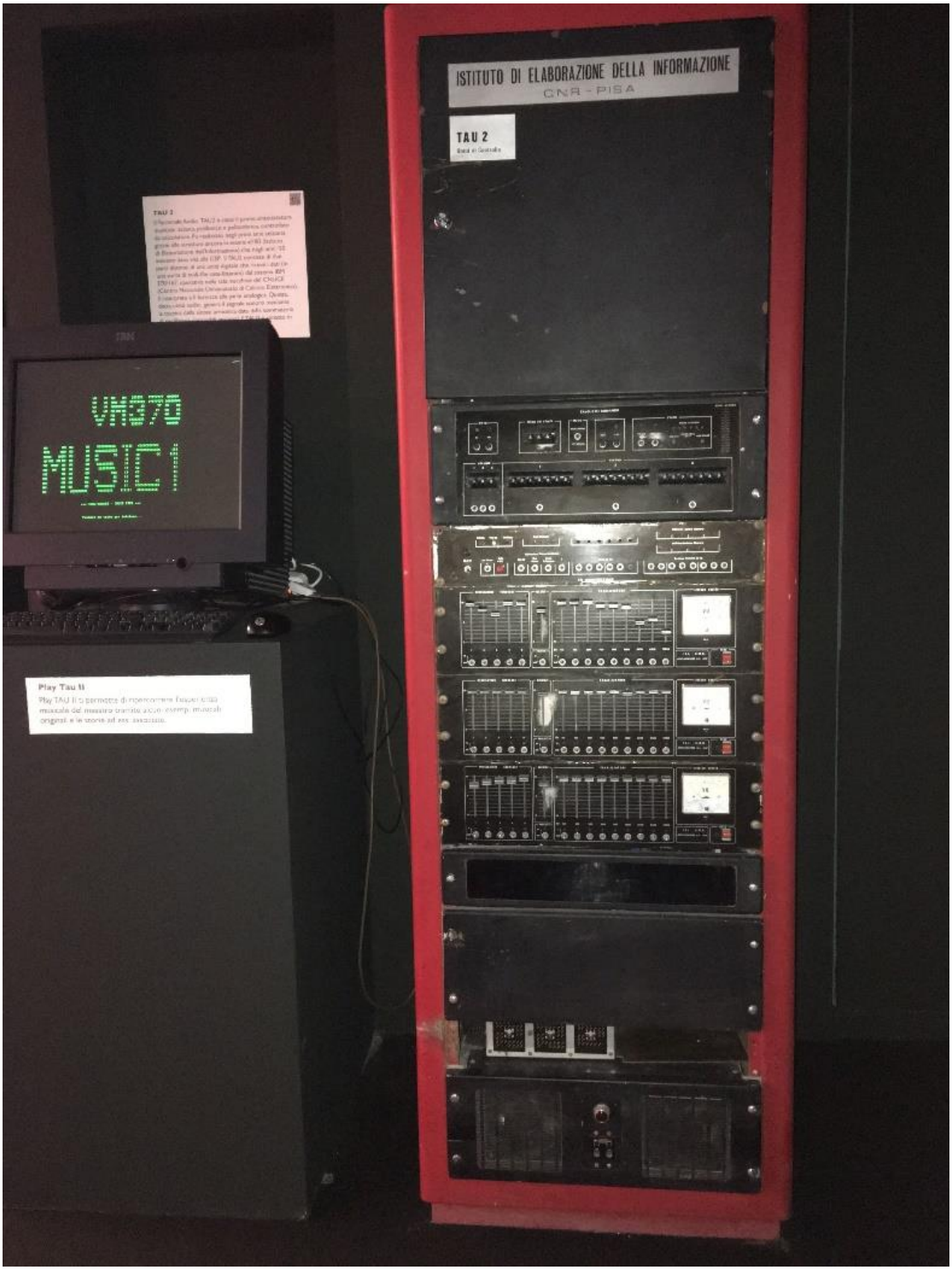

Figura 1: Disposizione del "terminale" accanto al'Unità Digitale del TAU2

2.2 Architettura SW

Al fine di "mimare" una sessione "vintage" al terminale la GUI del software è stata realizzata completamente tramite interfaccia a caratteri. È stato utilizzato un font IBM, ed un colore dei caratteri identico a quello dei monitor a fosfori verdi.

2.2.1 Processing

Il software è stato implementato in Java utilizzando l'ambiente Processing. Processing è un linguaggio di programmazione che consente di sviluppare diverse applicazioni come giochi, animazioni, contenuti interattivi e opere d'arte generativa. Eredita tutta la sintassi, i comandi e il paradigma di programmazione orientata agli oggetti dal linguaggio Java ma in più mette a disposizione numerose funzioni ad alto livello per gestire in modo facile gli aspetti grafici e multimediali. È distribuito nei termini della licenza libera GNU General Public License (GPL) ed è compatibile con i sistemi operativi Linux (inclusa la distribuzione Raspbian di Raspberry), macOS e Microsoft Windows. Processing permette di essere esteso nelle sue funzionalità tramite l'aggiunta di numerose librerie aggiuntive, che coprono i campi più disparati. Le applicazioni sviluppate nell'ambiente possono essere esportate direttamente come eseguibili per le diverse piattaforme.

2.2.2 Struttura dell'applicazione

La struttura dell'applicazione è minimale: il menu a tutto schermo, tramite l'interazione con la tastiera, permette di sceglie una fra 11 tracce musicali (fig.2).
La selezione della traccia mostra, in basso su video, una descrizione testuale della traccia stessa. Premendo invio si attiva l'esecuzione della traccia selezionata.
L'esecuzione, prima di attivare l'output sonoro vero e proprio, mima brevemente a video (fig.3) le operazioni compiute sulla traccia (es. inversioni, trasposizioni etc.). Queste operazioni erano le elaborazioni di base standard che il sistema TAUMUS permetteva di effettuare sugli spartiti originali caricati. Le operazioni principali erano:

- Trasposizione
- Inversione
- Cambio della velocità di riproduzione
- Regolazione dell'ampiezza degli intervalli
- Etc.

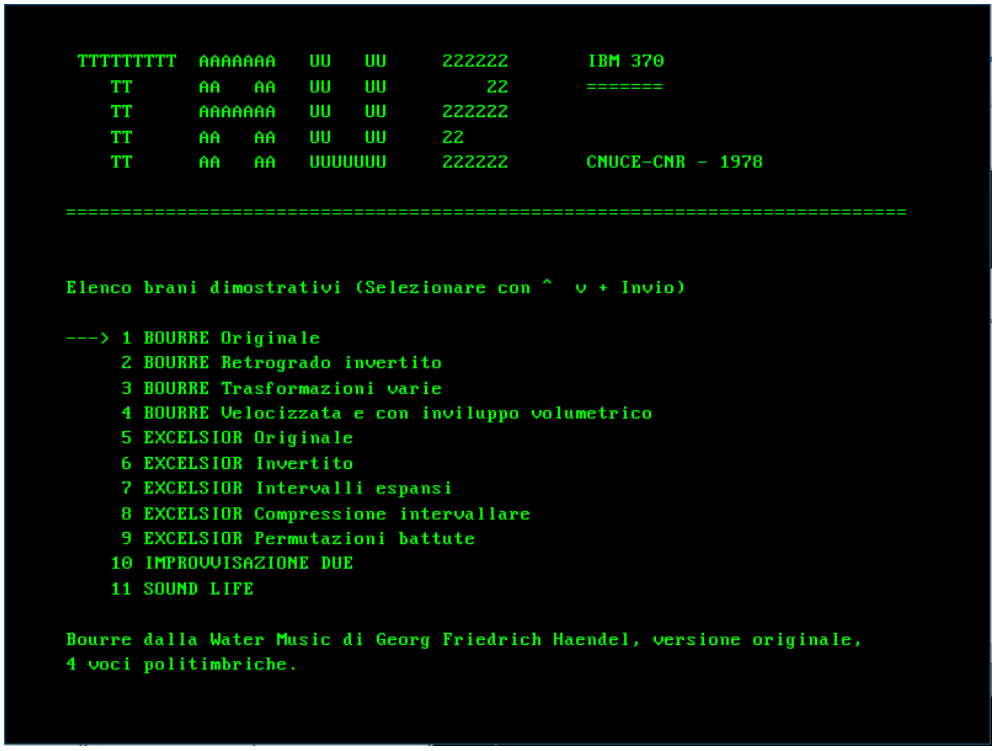

Figura 3: Menu iniziale dell'applicazione

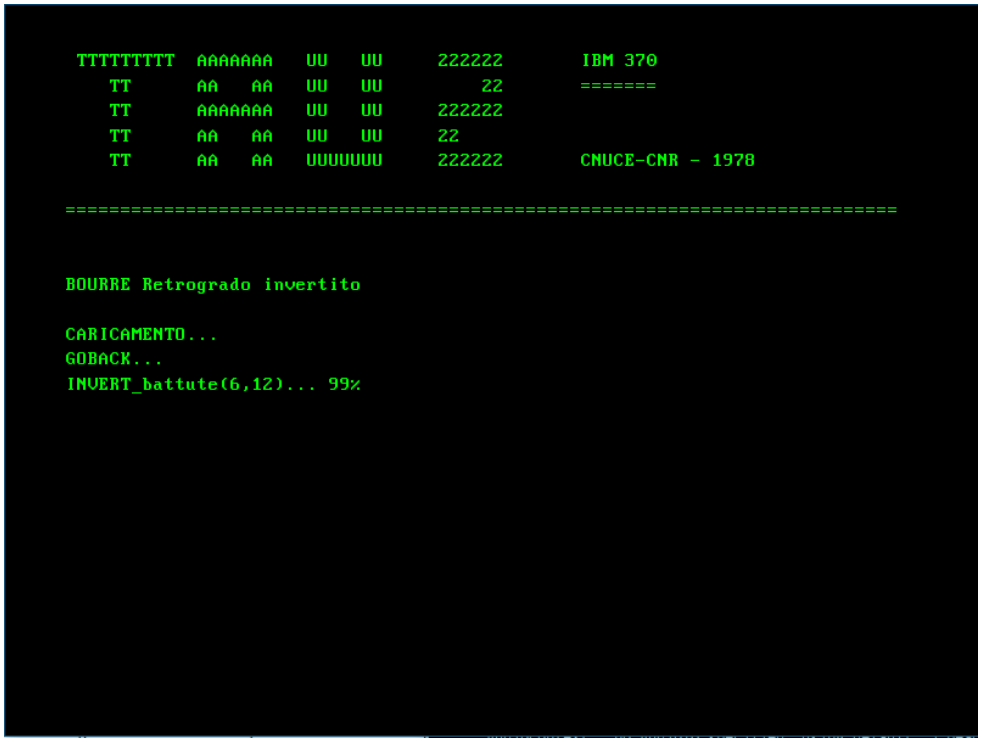

Figura 4: Fase di applicazione delle operazioni sul brano

2.2.3 Libreria Beads

Per quanto riguarda i contenuti audio si è scelto di utilizzare direttamente le tracce originali di Pietro Grossi. È stato necessario testare diverse librerie audio in modo da garantire un funzionamento corretto sulla piattaforma Raspberry (la libreria Sound integrata infatti non funziona su questa piattaforma). La scelta finale è ricaduta su *Beads*, una libreria software scritta in Java per audio in tempo reale. Avviata da Ollie Bown nel 2008, è un progetto open source ed è stato sviluppato con il supporto della Monash University di Melbourne, tramite il progetto ARC Discovery Grant del Center for Electronic Media Art "Creative Ecosystems" e un piccolo finanziamento per Ricercatori in carriera presso la Facoltà di tecnologia dell'informazione. I collaboratori di Beads includono Ollie Bown, Ben Porter e Benito. Oltre a poter caricare ed eseguire file audio in diversi formati, Beads permette anche di costruire algoritmi di sintesi a partire da operatori elementari (*UGen*), e di intervenire in lettura e/o scrittura sul buffer di campioni che viene mandato in esecuzione. Questa funzionalità è stata utilizzata per la visualizzazione grafica durante l'esecuzione.
L'accesso alla libreria audio Beads avviene tramite la creazione di un *Audio Context,* un oggetto sw che permette il controllo delle periferiche audio. La semplice creazione dell'oggetto permette la connessione alla periferica predefinita.
Il caricamento dei file audio avviene tramite l'utilizzo delle classi *SampleManager* e *SamplePlayer*. Una volta creato un player relativo ad un file su disco, al fine di renderne possibile l'esecuzione, è necessario connetterlo allo Audio Context. L'esecuzione viene attivata tramite il metodo statico *start()* della classe Audio Context.
Una volta attivata l'esecuzione, per avere accesso al buffer di campioni in esecuzione è possibile utilizzare i metodi *getBufferSize* e *out.getValue* della classe AudioContext. L'accesso effettuato in questo modo non è indicato per l'elaborazione dei campioni stessi (da effettuarsi invece tramite il meccanismo degli *UGen*) ma piuttosto per il loro utilizzo, come nostro caso, in visualizzatori grafici.

2.2.4 Visualizzatore segnale in GLSL

Per rendere più accattivante l'installazione, pur mantenendo il suo stile minimale, si è deciso di ricorrere ad un tipo di visualizzazione in stile vagamente "retro", cercando un compromesso fra efficacia della stessa e aderenza allo stile grafico dell'epoca. Dopo alcuni test, è stata scartata l'ipotesi di una visualizzazione dei segnali legata al contenuto spettrale (da ottenersi tramite *Short Time Fast Fourier Transform*) in quanto poco funzionale sul tipo di materiale audio da riprodurre. Si è deciso quindi di optare per una visualizzazione diretta del segnale nel tempo, con una modalità stile *Waterfall*. Per non caricare troppo le scarse risorse della CPU del processore con la gestione della Waterfall si è deciso di utilizzare quindi la piccola GPU integrata. Sebbene di scarsa potenza anch'essa, con alcuni accorgimenti è comunque possibile ottenere risultati grafici abbastanza sorprendenti.

Per poter sfruttare le risorse della GPU è necessario utilizzare un *shader GLSL*. GLSL, acronimo di OpenGL Shading Language, è un linguaggio di programmazione ad alto livello per la gestione delle unità shader di una Graphics processing unit basato su linguaggio di programmazione C. Lo scopo di questo linguaggio è quello di permettere ai programmatori un controllo più diretto e immediato delle pipeline grafiche, che non richieda l'uso di codice assembly o di codici specifici. Un'applicazione può inviare quindi specifici piccoli programmi scritti GLSL, gli *shader* appunto, che descrivono come i poligoni e le immagini contigui debbano essere processati fino alla visualizzazione su schermo. I passaggi della pipeline grafica, e in alcuni casi l'intera pipeline, vengono rimpiazzati da questi programmi. Ci sono diversi tipi di shader:

Vertex shader: utilizzati per manipolare i vertici, prendendo la configurazione iniziale del vertice (operando su un solo vertice per volta) e alterandola cambiandone i valori di posizione

Tessellation e Geometry shader: in gradi di creare e manipolare primitive grafiche come linee, triangoli etc.

Fragment (o pixel) shader: questo shader prende le informazioni sviluppate dal vertex processing (vertex shader, tessellation shader, o geometry shader) ed espande le tradizionali operazioni di frammentazione permettendo di operare su ogni frammento individualmente per generare il colore del proprio pixel. Questa è un'operazione altamente parallela che sfrutta appieno le GPU del sistema.

Nel nostro caso è stato implementato un semplice ma molto efficiente Fragment shader per creare una sorta di Waterfall con un effetto di profondità (fig.5). Il core del suo funzionamento si basa in pratica su di una sola linea di codice:

```
gl_FragColor =
0.99*texture2D(texture,vec2(vertTexCoord.x*1.01,vertTexCoord.y*1.01) -
vec2(0.005, 0.01));
```

La costante iniziale <1 introduce la caratteristica di *fade,* il riferimento a coordinate leggermente piu' grandi e shiftate di una costante quella di *scaling*, di *blur*, e di *translate*.

I file di tipo GLSL, quando di piccole dimensioni come in questo caso, possono essere integrati nell'applicazione Processing in un modo *inline,* incorporandoli cioè in una stringa di caratteri da caricare poi con apposite funzioni integrate in Processing.

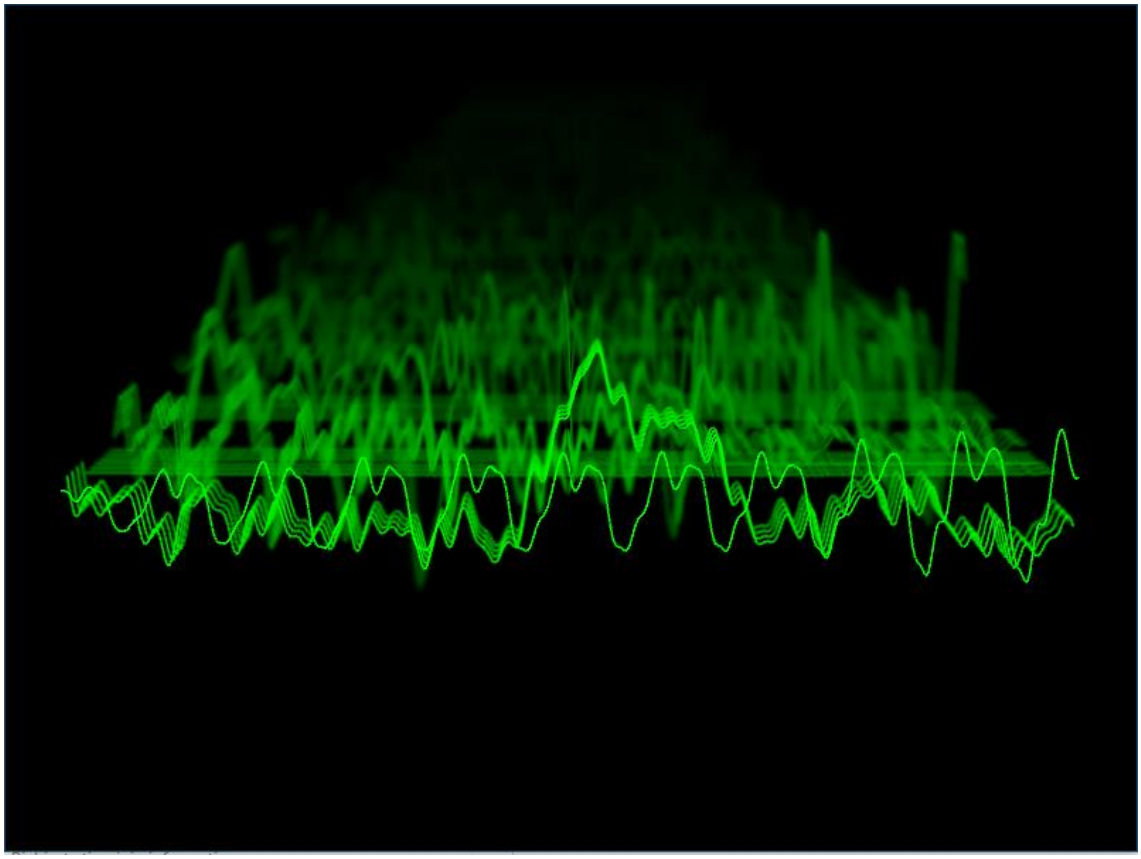

Figura 5: Visualizzazione grafica

2.3 Lista brani

Di seguito elenchiamo nel dettaglio i brani inseriti nell'exhibit selezionabili tramite il menu dell'applicazione, con le loro relative operazioni (simulate a video nell'exhibit) con cui sono stati originariamente costruiti.

1 – BOURRÉE Originale

Bourrée dalla Water Music di Georg Friedrich Haendel, versione originale, 4 voci politimbriche.

```
CARICAMENTO...
PLAY
```

2 – BOURRÉE Retrogrado invertito

Bourrée dalla Water Music di Georg Friedrich Handel, con trasformazione GOBACK che esegue il brano al contrario (dall'ultima nota alla prima)e successivamente anche con trasformazione INVERT, che inverte l'altezza delle note mantenendo i rapporti intervallari. Si noti che questa operazione comporta un passaggio dalla tonalità maggiore a quella minore.

```
CARICAMENTO...
GOBACK...
INVERT_battute(6,12)...
PLAY
```

3 – BOURRÉE Trasformazioni varie

Bourrée dalla Water Music di Georg Friedrich Handel con trasformazioni:

- battute 1-8: compressione intervallare
- battute 9-16: GOBACK INVER
- battute 17-end: espansione intervallare

```
CARICAMENTO...
MODIFY_compress_battute(1,8,0.1)
GOBACK_battute(9,16)
INVERT_battute(9,16)...
MODIFY_expand_battute(17,end,10)
PLAY
```

4 - BOURRÉE Velocizzata e con inviluppo volumetrico

Bourrée dalla Water Music di Georg Friedrich Handel, Con incremento velocità di esecuzione e modulazione volumetrica dalla battuta 5 alla 8

```
CARICAMENTO...
MODIFY_durate(0.5)
MODIFY_volume(5,8,0.5)
PLAY
```

5 - EXCELSIOR Originale

GALOP dal Balletto "EXCELSIOR" di Romualdo Marenco, versione originale.

```
CARICAMENTO...
PLAY...
```

6 - EXCELSIOR Invertito

GALOP dal Balletto "EXCELSIOR" di Romualdo Marenco, con trasformazione INVERT su tutta la lunghezza del brano, e modifica della velocità di esecuzione che inverte l'altezza delle note mantenendo i rapporti intervallari.

```
CARICAMENTO...
INVERT...
MODIFY_durate(0.7)
PLAY...
```

7 - EXCELSIOR Intervalli espansi

GALOP dal Balletto "EXCELSIOR" di Romualdo Marenco, con espansione intervallare

```
CARICAMENTO...
MODIFY_expand(#,10)...
PLAY...
```

8 - EXCELSIOR Compressione intervallare

GALOP dal Balletto "EXCELSIOR" di Romualdo Marenco, con modifica della velocità di esecuzione e compressione dei rapporti intervallari.

```
CARICAMENTO...
MODIFY_durate(#,#,0.5)
MODIFY_compress(#,#,0.1)
PLAY...
```

9 - EXCELSIOR Permutazioni battute

GALOP dal Balletto "EXCELSIOR" di Romualdo Marenco, con permutazione casuale delle battute.

```
CARICAMENTO...
SHUFFLE...
PLAY...
```

10 - IMPROVVISAZIONE DUE

Composizione algoritmica originale di Pietro Grossi, ottenuta con il comando CREATE con parametri vari. Il comando Create permetteva un controllo molto raffinato sui parametri frequenza, durata, timbro e volume.

```
CREATE()...
SAVE("Improvvisazione DUE")...
PLAY...
```

11 - SOUND LIFE

"Sound Life" è una Composizione algoritmica originale di Pietro Grossi del 1984, ottenuta con il comando CREATE con parametri vari. Il comando Create permetteva un controllo molto raffinato sui parametri frequenza, durata, timbro e volume.

```
CREATE()...
SAVE("Sound Life")...
PLAY...
```

3 Conclusioni

L'exhibit, che ha suscitato un notevole interesse fra i visitatori, è stato inserito nel percorso della mostra *Hello world! L'informatica dall'aritmometro allo smartphone*, collocata all'interno delle celebrazioni di Informatica 50, per il cinquantesimo anniversario dall'istituzione, a Pisa, del primo corso di laurea italiano in Informatica. La mostra è stata curata dal Museo degli Strumenti per il Calcolo con la collaborazione del team de *La Jetée*, specializzato in questo genere di allestimenti. La mostra è stata aperta continuativamente dal 10 ottobre 2019 al 30 aprile 2020, e successivamente su prenotazione a causa dell'emergenza Covid19.

Fra i vari eventi correlati all'esposizione c'è stata la presentazione del progetto Software Heritage, iniziativa internazionale che si propone di raccogliere tutto il software disponibile pubblicamente sotto forma di codice sorgente insieme alla sua storia di sviluppo, e di replicarlo in modo massiccio per garantirne la conservazione e la condivisione. Grazie a questo progetto Il software del TAUMUS/TAU2 è stato reso disponibile liberamente [7].

Bibliografia

[1] G. Bertini, M. Chimenti, F. Denoth - *"TAU2: An Audio Terminal for Computer Music Experiments"* Proceed. Int. Symp. on Technology for Selective Dissemination of Information (San Marino, 1976), ed. IEEE N.Y. pp 143-149.

[2] Bertini G.; Chimenti M.; Denoth F., TAU2: *Un terminale audio per esperimenti di computer music.* Nota tecnica IEI, Pisa, 1976.

[3] Bertini G.; Grossi P.: *Utilizzazione del sistema di computer music TAU2-TAUMUS per l'attivita' didattica e dimostrativa.* Manuale tecnico IEI, Pisa, 1982.

[4] Tarabella L.; Bertini G.; Raffaelli C.; Doni L. (a cura di). *L'informatica musicale a Pisa. L'esperienza di Pietro Grossi al CNUCE e all'IEI istituti del CNR* Contributo in volume Quaderni della Fondazione Galileo Galilei, Pisa University Press, Pisa 2019.

[5] Tarabella L. Bertini G., Magrini M.: *An interactive musical exhibit based on infrared sensor,* Articolo in Rivista Lecture notes in computer science, 2006.

[6] "*Hello world! L'informatica dall'aritmometro allo smartphone, 50 anni di Informatica in mostra a Pisa*", il Museo degli Strumenti per il Calcolo dell'Università di Pisa. https://www.unipi.it/index.php/mostre/event/4717-hello-world-l-informatica-dall-aritmometro-allo-smartphone

[7] https://archive.softwareheritage.org/browse/origin/directory/?origin_url=https://github.com/Unipisa/TAUmus.git