Conservatorio Statale di Musica «D. Cimarosa» di Avellino

DIPARTIMENTO DI NUOVE TECNOLOGIE E LINGUAGGI MUSICALI

SCUOLA DI MUSICA ELETTRONICA

DCPL34 - CORSO DI DIPLOMA ACCADEMICO DI PRIMO LIVELLO IN

MUSICA ELETTRONICA

D.M. n. 164 del 3.09.2010

TESI DI DIPLOMA ACCADEMICO DI I LIVELLO

Progettazione di sistemi modulari generativi per la musica elettroacustica

Relatore

Chiar. mo M° Alba F. Battista

Candidato

Giuseppe Bergamino

Matr. 9833

ANNO ACCADEMICO 2017/2018

Indice

Introduzione

Questo lavoro di tesi si occupa della progettazione e realizzazione di un sistema hardware modulare generativo, secondo un approccio improntato sulla liuteria elettronica: saranno parimenti trattati e sviluppati argomenti inerenti agli aspetti musicali, tecnici ed estetici, con lo scopo di gettare basi solide per l'ottenimento di uno strumento musicale che abbia facoltà di produrre autonomamente materiale sonoro. In questo senso si porrà particolare attenzione alla creazione di modelli funzionali all'organizzazione coerente degli eventi nel tempo, dando la facoltà al sistema di limitare le proprie possibilità per direzionare ogni scelta presa sia istantaneamente che su una grande arcata temporale. Formalmente il primo capitolo si occupa della trattazione storica ed estetica dello strumento elettroacustico; il secondo affronta, da prospettive diverse, le molte sfaccettature proprie della musica generativa; negli ultimi due capitoli, con crescente livello di approfondimento, sono esposti gli approcci tecnici e musicali che si sono ritenuti fondamentali da sviluppare per la realizzazione finale dello strumento. Infine, verrà descritta la performance *101010:* un'opera con live electronics in quadrifonia, per esecutore e strumento elettroacustico generativo, realizzata per questo lavoro di tesi.

1. STORIA ED ESTETICA DEGLI STRUMENTI ELETTROACUSTICI

"Sogno strumenti obbedienti alle mie necessità con i quali accedere ad un intero nuovo mondo di suoni inaspettati, che si prestino alle esigenze del mio ritmo interiore"
Edgar Varèse

La musica è tanto l'arte quanto la scienza dell'organizzazione del suono e del silenzio, sia nel tempo che nello spazio[1]. L' uomo ha da sempre avuto la necessità di fare musica, una costante che accompagna la nostra specie dagli albori fino al presente. Analizzare le musiche di un popolo in un determinato tempo restituisce in buona misura una rappresentazione del suo stato culturale, sociale e scientifico-tecnologico. In questo capitolo verrà analizzato il rapporto storico tra musica, scienza e tecnologia, nella misura in cui lo sviluppo tecnologico ha permesso di espandere le possibilità di concepire, produrre e fruire la musica.

1.1 Liuteria musicale: evoluzione e sviluppi

Come ogni organismo complesso, l'uomo ha la necessità di trasmettere e ricevere informazioni; l'evoluzione lo ha dotato di vari sistemi, quali l'apparato fonatorio e uditivo, per soddisfare questo bisogno. Il suono prodotto dalle corde vocali è stato il mezzo più immediato per trasmettere informazioni di vario tipo e non è difficile immaginare che con il progredire della nostra cultura e delle nostre necessità l'apparato fonatorio si sia svincolato dalla sua funzione puramente pragmatica, acquisendo il diritto di poter essere definito uno *strumento musicale.*

Uno strumento musicale è sistema costruito, adattato o modificato allo scopo di produrre suoni *organizzati*, capace di assecondare una necessità estetica e culturale[2]. Non è, quindi, difficile supporre che siano stati proprio il nostro corpo assieme ad altri

[1] Pasceri, C. (2011). *Tecnolgia Musicale: La rivelazione della musica.* Roma: Aracne Editrice.

[2] Sachs C. (1980). *Storia degli strumenti musicali.* Milano: Arnoldo Mondadori.

oggetti di facile reperibilità, usati principalmente come percussioni, gli strumenti a disposizione dell'uomo per produrre suono e, più ampiamente, musica.

Probabilmente il primo oggetto costruito dall'uomo con l'unico scopo di produrre suono risale a circa 40000 anni fa: il flauto di Divje Babe è il femore di un orso lungo 12 centimetri che presenta 4 fori distanti tra loro 3,5 centimetri con diametro di 9 millimetri, le cui dimensioni suggeriscono l'intenzionale ricerca di uno strumento che avesse la possibilità di muoversi per quelle che oggi chiameremo 2 ottave[3].

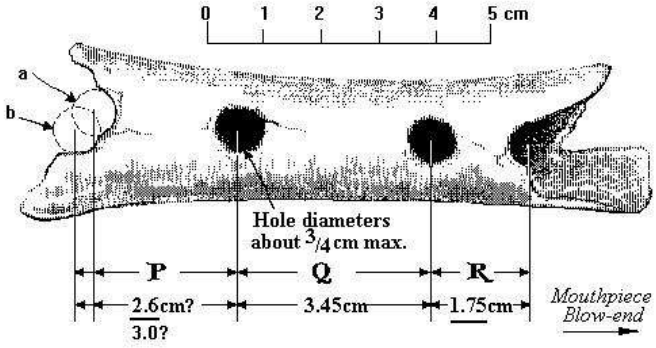


Figura 1.1 Ricostruzione del flauto di Divje Babe. La fattura rispecchia le primitive capacità costruttive dell'uomo di Neanderthal; di notevole interesse le quote che riportano dimensione e distanza tra i fori. (Archivio del Museo nazionale della Slovenia)

Circa 7000 anni fa gli aborigeni australiani scoprirono che le termiti, rendendo cavo un tronco di eucalipto, permettevano di raggiungere una lavorazione non facilmente ottenibile anche con le tecnologie moderne: nasceva il *didgeridoo*, uno strumento a fiato formato da un tubo cavo aperto ad entrambe le estremità lungo dai 120 ai 200 centimetri, antenato degli strumenti ad ancia labiale che permette di ottenere ritmi, suoni staccati e bordoni in rapporto armonico con la risonanza dell'oggetto[4].

Figura 1.2 Tradizionale didgeridoo australiano, spesso le decorazioni richiamano eventi della mitologia aborigena. (Wikimedia Commons)

[3] Kunej D., Turk I.(2000). *New perspectives on the beginnings of music: archaeological and musicological analysis of a Middle Palaeolithic bone "flute"*. In Wallin 2000, N.L., Merker B. & Brown S., *The Origins of Music* (pp. 235–268). Cambridge: MIT Press.

[4] Ricciardi C.(2011). *L'albero che canta – Il didgeridoo.* Roma: Wondermark.

Tuttavia tale strumento, considerata la sua relativa semplicità costruttiva, non garantisce alcuna agilità dal punto di vista melodico: è possibile ottenere facilmente una nota fondamentale data dalla risonanza del sistema e, con un'avanzata padronanza tecnica da parte dello strumentista, anche alcune sue armoniche superiori. Nel corso dei secoli l'uomo è riuscito ad apprendere di più sul fenomeno fisico del suono e ha sfruttato questa conoscenza per fare in modo che un materiale messo in vibrazione assecondasse le sue esigenze anche dal punto di vista musicale. I primi studi sulla fisica acustica, quantomeno per la cultura occidentale, sono da attribuirsi alla scuola pitagorica. Nel I secolo a.C. Pitagora intuì che più *note* successive con frequenze poste in rapporto matematico fra loro avevano coerenza e risultavano piacevoli all'ascolto e in conseguenza di tale osservazione formalizzò il concetto di *scala musicale*[5]. Il sistema diatonico con scale di sette suoni divise secondo intervalli di tono e semitono, ancora oggi in uso per alcuni linguaggi musicali, è l'epigono delle scoperte matematiche in ambito musicale compiute dagli antichi greci.

Intervallo	*Rapporto*
Unisono	1 : 1
Seconda maggiore	9 : 8
Terza maggiore	81 : 64
Quarta giusta	4 : 3
Quinta giusta	3 : 2
Sesta maggiore	27 : 16
Settima maggiore	243 : 128
Ottava	2 : 1

[5] Sachs C. (1943). *The Rise of Music in the Ancient World - East and West.* New York: W.W. Norton&Co.

Figura 1.3 Suonatore di cetra a otto corde, immagine tratta da un'antica raffigurazione vascolare greca del IV secolo. (Wikimedia Commons)

Dimezzando la lunghezza di una corda vibrante, a parità di sezione e materiale, si ottiene la stessa nota con frequenza doppia, quindi un salto di un'ottava: questa nuova consapevolezza ha favorito lo sviluppo di nuovi strumenti musicali e di conseguenza nuove possibilità compositive. Nel corso dei secoli questa scoperta ha permesso di costruire strumenti musicali sempre più elaborati. Le conoscenze acquisite empiricamente e scientificamente in merito alle proprietà dei materiali di trasmettere onde meccaniche e lo sviluppo tecnologico necessario per poterli lavorare, hanno fatto in modo di ottenere strumenti timbricamente più ricchi e strutturalmente più stabili, capaci sia di assecondare al meglio la maggior parte dei parametri musicali esistenti, sia di permettere l'introduzione di nuove variabili nella composizione.

L'evoluzione nell'estetica del repertorio europeo del XVII secolo è stata notevolmente agevolata dal progresso nell'arte *liutaia:* con questo termine si intende l'arte di progettazione, costruzione e, in tempi moderni, di restauro di strumenti a corda ad arco e a pizzico; in senso più lato, ci si riferisce alla produzione di strumenti musicali in genere[6]. Nel 1600 l'Italia era la principale protagonista con le scuole di Cremona e Brescia, le botteghe dei liutai italiani non erano solo luoghi di produzione per ottimi strumenti, ma probabilmente anche i primi centri di ricerca e sviluppo sullo strumento musicale: l'odierna geometria del ponticello mobile degli strumenti ad arco è stata introdotta in quel periodo, garantendo un migliore disadattamento di impedenza acustica tra la cordiera e la cassa armonica e agevolando il moto dell'arco sulla singola corda[7].

---

[6] Vatielli F. (1934). Liuteria. In Enciclopedia Italiana.

[7] Beament J. (1997). *The violin explained: Componets, Mechanism and sound.* Oxford: Oxford University Press.

Figura 1.4 Violino Stradivari Gould (1693). Antonio Stradivari e Antonio Guarnieri del Gesù sono due dei massimi esponenti della scuola liutaia cremonese. (Metropolitan Museum of Art)

La perfezione con cui venivano costruiti questi strumenti ha portato, ad esempio, ad ottenere una stabilità nell'intonazione eccellente, permettendo ai compositori di non doversi porre molti limiti pratici rispetto allo strumento e garantendo una maggiore libertà, sotto vari punti di vista, sia alle composizioni, che allo strumentista stesso. Si è ora in grado di esigere di più dallo strumento, riuscendo a prendere in considerazione una maggiore gamma di parametri musicali. A questo proposito, è emblematica l'evoluzione che ha portato alla nascita del pianoforte. Fino alla fine del 1700 il clavicembalo è stato uno degli strumenti musicali a tastiera più utilizzato nella prassi musicale, sia per la sua natura polifonica, sia per la capacità di muoversi agilmente tra più ottave; la modalità di emissione sonora con tasti che una volta premuti permettevano ad un sistema di leve di pizzicare le corde, era poco pratica del punto di vista del controllo dinamico del suono, risultando in esecuzioni che non riuscivano a soddisfare appieno questo parametro musicale fondamentale.

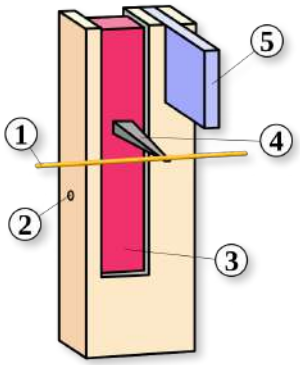


Figura 1.5 Sistema di eccitazione delle corde del clavicembalo. 1) corda 2) asse della linguetta 3) linguetta 4) plettro 5) smorzatore. (Wikimedia Commons)

Il *forte-piano* deve il suo nome al suo primo sviluppatore, Bartolomeo Cristofori, proprio perché permetteva una notevole espressività dinamica, certamente migliore di qualsiasi altro strumento a tastiera esistente. Il progetto iniziale fu ampliato da altri liutai che ne intuirono le potenzialità e ne migliorarono le meccaniche, la struttura e i materiali. Difatti il sistema di eccitazione non era più dato dal pizzico della corda, ma dalla sua percussione mediante un martelletto di legno e furono introdotti pedali che permettevano di estenderne ulteriormente le capacità timbriche: si andava sempre più verso la realizzazione dello strumento oggi noto come *pianoforte*[8].

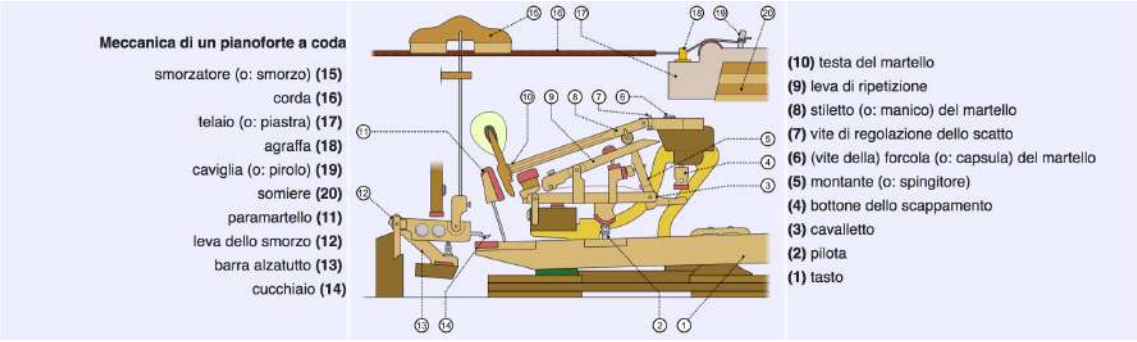


Figura 1.6 Sistema di eccitazione delle corde del pianoforte, notevolmente più elaborato rispetto al meccanismo del clavicembalo. (Wikimedia Commons)

Parallelamente allo sviluppo in ambito musicale, sia dal punto di vista degli strumenti che di estetica compositiva, dal XVI al XIX secolo, l'umanità ha notevolmente arricchito il proprio patrimonio scientifico, comprendendo meglio il mondo che la

[8] Casella A. (1984). *Il pianoforte*. Milano: Ricordi.

circonda e formalizzando leggi che riuscissero a spiegare in buona misura parte dei fenomeni fisici.

1.1.1 La rivoluzione tecnologica dell'elettricità

I primi studi relativi al fenomeno dell'elettricità cominciarono nel 600 a.C. quando il filosofo greco Talete osservò che l'ambra, sfregata con un panno, ha la capacità di attrarre piccoli pezzi di materia; il nome greco di questa resina fossile era *electron,* termine da cui deriva la parola elettricità.

Osservazioni di questo tipo, coadiuvate principalmente da esperienze empiriche, ripresero nel XVI secolo. Nuovi termini come attrazione e repulsione elettrica, conduttore e isolante, condensatore e fluido elettrico, vennero aggiunti al nostro dizionario scientifico.

E' tra la fine del XII e l'inizio del XIX secolo che sono dimostrate sperimentalmente alcune teorie ed enunciate leggi che finalmente diedero la possibilità di imbrigliare questo fenomeno fisico mettendolo a disposizione dello sviluppo tecnologico.

Le principali unità di misura legate all'elettricità portano il nome degli scienziati che ne formalizzarono alcune leggi, come Coulomb, Volta, Ampère, Ohm, Faraday, Henry. Dalla prima metà fino alla fine del 1800 cominciano a comparire le prime applicazioni tecniche, gettando le basi per molta della tecnologia moderna: il telegrafo, il relè, il motore elettrico, la lampadina, il telefono, il trasformatore, le centrali elettriche e il fonografo.

Figura 1.7 Thomas Edison con il suo primo fonografo (1878). (Brady-Handy Photograph Collection, Library of Congress)

Nel 1878 Thomas Edison presentò al mondo questo nuovo strumento che aveva la capacità di *registrare* e *riprodurre* il suono; prima di allora esistevano sistemi meccanici che permettevano di emettere suono a prescindere dalla presenza fisica di un esecutore, macchine che riuscivano a riprodurre musica memorizzata su vari supporti, con il limite che non potevano eseguire suoni arbitrari, né tantomeno registrarne; il fonografo ha, invece, la rivoluzionaria capacità di poter registrare e riprodurre il suono: cambiando il modo di fruire la musica, lo spettatore si svincola dall'esecutore dal vivo, con una ripetibilità idealmente infinita della stessa esecuzione.

Nel 1895 Marconi riuscì con successo a trasmettere il primo segnale radio per oltre 2 chilometri, seguito nei successivi decenni dalle prime trasmissioni radiofoniche che permisero al pubblico di ascoltare brani musicali nelle proprie abitazioni: la radio e il fonografo cambiarono radicalmente la percezione dell'esperienza musicale che ora si astrae da esecutori umani[9].

Figura 1.8 Detector Marconi, utilizzato nei primi esperimenti di trasmissione e ricezione di onde radio nel 1902. (Museo Nazionale della Scienza e della Tecnologia Leonardo da Vinci, Milano)

Finalmente alla fine del 1800 l'evoluzione tecnologica e culturale raggiunse una maturazione tale da permettere di concepire strumenti musicali che non fossero più solo acustici. La scienza acustica applicata alla costruzione di strumenti elaborò nuove tecniche produttive, materiali, colle e leghe metalliche: si era raggiunto un grado di raffinatezza tale da non riuscire più ad introdurre ulteriori sviluppi significativi per tutta la seconda metà del secolo. Aver *catturato* l'elettricità offrì un mondo di nuove possibilità musicali.

---

[9] Menzies R. (1999). *New Electronic Performance Instruments for Electroacoustic Music*. York: University of York.

1.2 Lo strumento elettroacustico prima del 1945

Lavorando in un campo ancora inesplorato, con notevoli difficoltà tecniche, Thaddeus Cahill riuscì a costruire nel 1906 uno dei primi strumenti elettroacustici della storia.

Il *Telharmonium* è un dispositivo elettromeccanico di 200 tonnellate, formato da 145 alternatori (ruote foniche) che riuscivano a produrre correnti alternate a varie frequenze; il metodo di generazione sonora, simile a quella che oggi definiremo *sintesi additiva*, era controllabile tramite due tastiere i cui tasti attivano uno o più dei 145 alternatori; lo strumento era dotato di vari controlli per modificarne il timbro che tendeva ad imitare quello di un'orchestra d'archi. Fu progettato prima dello sviluppo dell'amplificatore elettrico, quindi il suoi alternatori dovevano poter sviluppare una potenza di circa 10000 Watt, il tehlarmonium risultava così uno strumento dalle dimensioni immense, la cui poca praticità non ne favorì lo sviluppo[10].

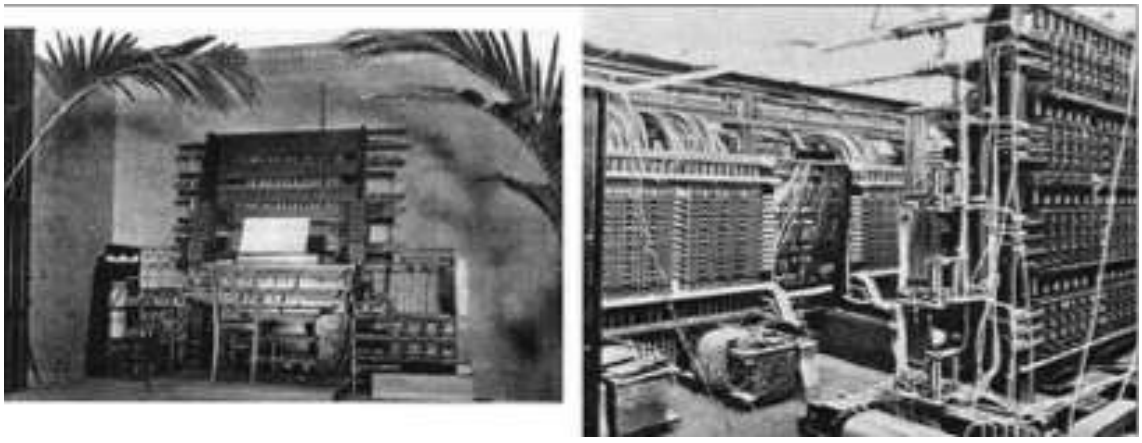


Figura 1.9 Immagini della Telharmonic Hall di New York (1908). La console (sinistra) ricorda la tastiera di un organo a canne, con la quale si controllava il vasto sistema di sintesi sonora (destra) posto nel seminterrato del palazzo. (Electrical World Journal, 1908)

Nello stesso periodo Lee De Forest brevettò la valvola termoionica e il primo *oscillatore*, dando inizio all'era dell'elettronica.

[10] Dunn D. (1992). *A history of electronic music pioneers*. Linz: Ars Electronica.

Nel 1913 Luigi Russolo scrive nel suo manifesto futurista *L'arte dei rumori:*

L'evoluzione della musica è parallela al moltiplicarsi delle macchine[11].

Il suono, inteso come evento utilizzabile in una composizione musicale, non è più esclusiva degli strumenti classici e la linea che divideva esteticamente il suono dal rumore comincia ad essere meno marcata: i due termini coesistono per creare un concetto più ampio. Nel 1914 Russolo e Ugo Piatti costruirono un'orchestra di strumenti elettromeccanici capace di generare suoni inarmonici di vari timbri e fattura: l'*Intonarumori*.

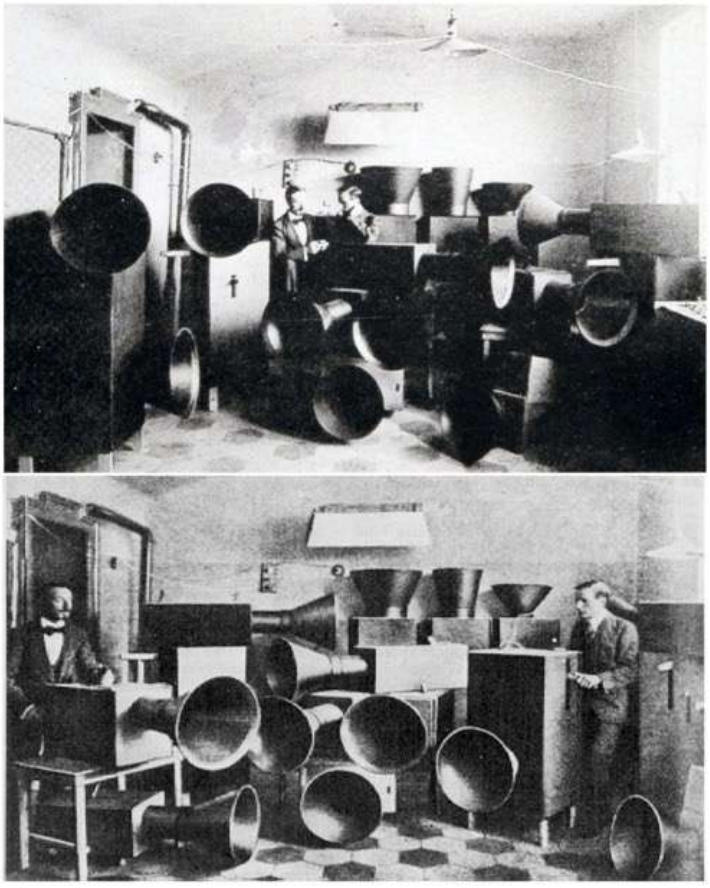


Figura 1.10 Russolo e piatti con ensemble di Intonarumori. In base al suono prodotto, gli strumenti erano classificati per famiglie (crepitatori, gorgogliatori, rombatori, ronzatori, scoppiatori, sibilatori, stropicciato e ululatori), ciascuna delle quali comprendeva a sua volta vari registri (soprano, contralto, tenore, basso). Gli strumenti erano completamente meccanici, l'amplificazione era garantita da un tronco di cono in cartone. (Foto pubblicata nel libro *L'arte dei rumori*)

[11] Russolo L. (1913). *L'arte dei rumori*. Milano: Edizioni futuriste di "poesia".

Uno tra gli strumenti più rivoluzionari di questo periodo fu inventato dal fisico sovietico Lev Seergevic Termen. Nel 1919 egli scoprì che, in determinate condizioni, un particolare circuito elettrico riusciva ad emettere una frequenza all'interno dello spettro udibile se il suo campo veniva perturbato dalla prossimità di un oggetto: questo iniziale *errore* progettuale portò allo sviluppo del *theremin*. Lo strumento si basa su oscillatori che lavorano ad alta frequenza che, influenzati dalla presenza delle mani del musicista nel campo d'onda, sfruttano il fenomeno dei battimenti per creare frequenze nello spettro udibile; lo strumento, monofonico e timbricamente simile ad un soprano, è dotato di due antenne per il controllo dell'ampiezza e della frequenza che rendono facile ottenere effetti di tremolo e vibrato: questa particolare tipologia di interazione senza contatto fisico rende il theremin uno strumento elettronico rivoluzionario nell'approccio sulla *gestualità* e *modalità di esecuzione*[12].

Figura 1.11 Termen e il suo strumento, in una foto di un concerto del 1924. (Griffiths, Paul (1987). *A concise History of Modern Music*. Thames and Hudson)

Dopo un incontro con Termen negli anni Venti, il francese Maurice Martenot cercò di sviluppare uno strumento elettroacustico che si potesse integrare con le orchestre sinfoniche del tempo. L'*Ondes Martenot* è uno strumento monofonico che sfrutta i principi di generazione del theremin, integrando un'interfaccia a tastiera più familiare per musicisti e compositori: fu quest'intuizione a garantire allo strumento successo e diffusione. L'Ondes Martenot sfrutta la differenza di frequenza emessa da

[12] Holmes T. (1985). *Electronic and experimental music: Technology, music and culture.* New York: Scribner.

due oscillatori, ha un'estensione di sei ottave ed è capace, grazie ad un *ribbon controller,* sia di generare intervalli microtonali, sia di ottenere il vibrato; un'ulteriore particolarità dello strumento è relativa al suo sistema di diffusione: lo strumentista può scegliere tra quattro casse acustiche diverse per amplificarne la voce e ogni altoparlante è costruito con materiali e geometrie tali da ottenere una notevole modifica del timbro[13].

Figura 1.12 Una versione orchestrale dell'Onde Martenot, appartenente alla 7ª serie, prodotta del 1975. Si notino i diffusori incaricati di amplificare lo strumento, ciascuno con differente effetto e sonorità: il diffusore *Palme* (in alto), il diffusore *Principal* (sotto) che, in questo caso, ingloba nello stesso cabinet anche il diffusore *Résonance* e il diffusore *Métallique* (destra). (Wikimedia Commons)

Gli anni Trenta vedono un'esponenziale diffusione in tutta Europa degli strumenti elettronici: le esperienze positive dei primi pionieri e le richieste dei compositori spronano nuovi costruttori ad addentrarsi in questo campo. Buona parte di questi strumenti intendeva esplorare nuove possibilità sonore, altri miravano ad imitare il timbro di strumenti musicali già esistenti. Ne è un esempio l'*organo hammond*.

Laurens Hammond sviluppa il principio di generazione elettromeccanico del telharmonium integrandolo con i nuovi progressi in ambito elettronico ed ottenendo un sistema più stabile, compatto e trasportabile; i suoi strumenti, ancora oggi in produzione, sfruttano le valvole a vuoto principalmente per amplificare e sommare i segnali, realizzando uno dei primi sintetizzatori ad avere un controllo stabile della frequenza e dell'intonazione. Il risultato è un inconfondibile timbro simile a quello di

[13] Holmes T. (1985). *Electronic and experimental music: Technology, music and culture.* New York: Scribner.

organo a canne, in un design significativamente più compatto, caratterizzante di molti generi musicali degli anni successivi. Hammond brevetterà anche un riverbero a molla da associare ai propri strumenti tuttora largamente utilizzato.

Figura 1.13 Organo Hammond, modello C3 del 1954. Spesso questo modello veniva amplificato con un sistema di diffusione Leslie: un altoparlante che integrava un motore per permettere la rotazione del cono, ottenendo particolari effetti di tremolo. (Wikimedia Commons)

L'idea vincente di Hammond è stata associare un *pickup* alle ruote foniche del telharmonium. Il pickup è una bobina di rame che avvolge del materiale ferromagnetico, il campo magnetico di questo elemento, una volta perturbato da un oggetto metallico in movimento, genera una differenza di potenziale proporzionale alla perturbazione subita. Il sistema è facilmente sfruttabile per trasdurre la vibrazione di una corda metallica in un segnale elettrico, principio che permise di amplificare vari strumenti. Comincia la produzione di chitarre, bassi e pianoforti elettrici. Questi sono diretti discendenti di strumenti classici largamente utilizzati nei secoli precedenti, che grazie allo sviluppo tecnologico trovano nuove possibilità espressive.

La ricerca di nuovi timbri da parte dei compositori portò spesso a fruttuose collaborazioni con gli ingegneri dell'epoca: l'incontro tra l'ingegnere Friedrich Trautwein e il compositore Paul Hindemith ne è un famoso esempio. Il *trautonium* è uno strumento monofonico a tastiera sviluppato nella prima metà degli anni Trenta che sfrutta delle valvole al neon per la generazione di onde a dente di sega; questo segnale ha uno spettro particolarmente ricco che contiene teoricamente infinite armoniche pari e dispari della frequenza fondamentale, con ampiezza inversamente proporzionale all'ordine di armonica: il risultato è un timbro molto, se non troppo, duro e chiaro. Trautwein disegnò quindi una serie di filtri, controllabili dal musicista tramite dei

potenziometri, che permettevano di attenuare l'ampiezza delle armoniche modificando il timbro: questo strumento diventa il primo a sfruttare il concetto di *sintesi sottrattiva*.

Figura 1.14 Alfred Hitchcock con i compositori Remi Gassmann e Oskar Scala, mentre lavorano con un trautonium durante la produzione della colonna sonora del film *Gli uccelli* (1963). (Hitchcock Gallery: 8820)

Nel 1939 un ingegnere dei laboratori Bell, Homer Dudley, sviluppa un sistema di analisi vocale per applicazioni non musicali. Il *voder* era un sistema di codifica formato da filtri passa-banda, utili per riprodurre le risonanze del sistema fonatorio umano, e da generatori di segnale per le vocali e rumore per le consonanti. Il *vocoder* era il suo rispettivo decodificatore, cioè sfruttava le informazioni codificate dal voder per sintetizzarle con un segnale portante, creando un segnale risultante che riuscisse a contenerne una parte. Questa tecnologia, prima di diventare di dominio pubblico e di vedere largo uso in ambito musicale, venne sfruttata dagli americani nella Seconda Guerra Mondiale per trasmettere informazioni criptate.

La guerra è tra gli eventi storici che più riesce ad accelerare il progresso tecnologico e a mutare società e cultura; durante la Seconda Guerra Mondiale le necessità di prevalere sul nemico e di conservarsi in vita hanno spronato tutte le parti interessate a sviluppare tecnologie che ci hanno permesso di raggiungere il grado di progresso moderno.

1.3 Anni post-bellici e studi di ricerca musicale

L'influenza della Seconda Guerra Mondiale sull'arte fu ovviamente drastica: la maggior parte delle attività creative e sperimentali cessarono, favorendo la ricerca principalmente in campo militare. Con la fine della guerra la disponibilità di nuove tecnologie, presto riadattate, e il desiderio di discostarsi dalle arti precedenti al conflitto mondiale gettarono basi fertili per un'esponenziale diffusione dei nuovi linguaggi musicali.

Un importante protagonista di questa rivoluzione fu il registratore a nastro che, rispetto ai suoi antenati, rendeva molto più fedele la registrazione del suono e ne semplificava la manipolazione; verosimilmente questo strumento rivoluzionò la musica elettronica più di qualsiasi altro evento accaduto fino ad ora, ridefinendola come genere e svincolandola dalla sola esecuzione live.

Questo crescente fermento culturale e tecnologico si concretizza in nuovi studi di registrazione, spesso veri e propri istituti di ricerca finanziati da aziende pubbliche o private che, con una certa astrazione, possono essere considerati il principale *strumento musicale* degli anni post-bellici[14].

1.3.1 Il GRM e la Musique Concrète

Sullo spirito di ricerca e innovazione del tempo, la *Radiodiffusion Télévision Française* (RTF) fonda il primo studio musicale per la produzione di musica elettronica.

Nel 1951, in seno alla RTF, nasce il *Groupe de Recherches Musicales* (GRM), collettivo formato principalmente da Pierre Schaeffer, Jaques Poullin e Pierre Henry, che ospitò negli anni i più grandi compositori dell'epoca. In questo ambiente Schaeffer perfeziona la sua teoria sulla *musica concreta* creando, sulle orme di Russolo, un sistema per categorizzare il suono tenendo conto di numerosi parametri, sia fisici che percettivi.

L*'oggetto sonoro* è la rivalsa estetica del rumore sul suono; ciò che prima era culturalmente inteso come non utilizzabile in musica, trova ora una solida base filosofica e formale: qualsiasi rumore, registrato, manipolato e organizzato, è una fonte

[14] Chadabe J. (1967). New Approaches to analog-studio design. *Perspectives of New Music*, Vol. 6, 107-113.

sonora legittima, quindi utilizzabile per comporre[15]. E' bello pensare, in questo senso, che il *mondo* stesso possa essere considerato uno strumento musicale[16].

Il GRM, lavorando principalmente con questo tipo di materiale concreto, disponeva di vari strumenti per la manipolazione dei nastri, spesso sviluppati dallo stesso Schaeffer; la liuteria elettronica diventa protagonista della ricerca e produzione musicale, nasce nel compositore la necessità di dover costruire in prima persona i propri strumenti, per agevolare il processo creativo ed esplorare nuove possibilità compositive. In questo modo lo strumento si svincola dalla sola funzione di emissione sonora, diventando parte attiva nell'indirizzare il compositore verso nuovi territori sonori.

Di seguito una lista dei principali strumenti progettati e utilizzati nello studio negli anni Sessanta.

- Il *Morphophone*, un sistema con 10 testine di lettura, che permettevano di ottenere feedback, loop e delay;

- Due *Phonogène*, uno con interfaccia a tastiera e l'altro con potenziometri, che permettevano di leggere il nastro a varie velocità, cambiandone il pitch;

- Il *Potentiomètre d'espace*, un controller che permetteva di gestire una diffusione quadrifonica;

- Il *Magnétophone*, un registratore a nastro a 6 tracce.

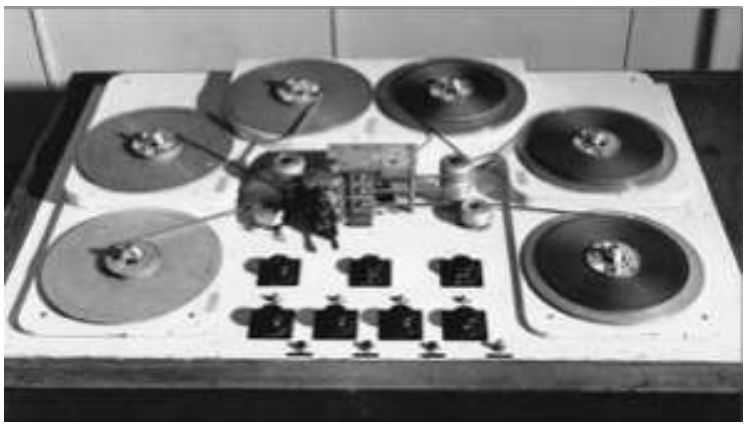


Figura 1.15 Il Magnetophone del 1962. Uno dei primi registratori a nastro a sei tracce. (ORTF, Una GRM Archives)

[15] Schaeffer P. (1966). *Traité des objets musicaux*. Parigi: Seuil.

[16] Schafer R. M. (1977). *The Tuning of the World*. New York: Random House Inc.

Figura 1.16 Foto del Phonogène, con interfaccia a tastiera, del 1967. Lo strumento era progettato in modo tale da poter rispettare gli intervalli classici del sistema temperato: la tastiera di un'ottava (visibile in basso a destra) permetteva di variare la velocità di rotazione del motore, modificando la velocità di lettura del nastro e quindi la frequenza del suono. Tipicamente veniva utilizzato un nastro magnetico incollato in modo tale da formare un anello chiuso, creando un loop ogni volta che uno dei tasti veniva premuto. (Ina GRM Archives)

1.3.2 Lo studio di Colonia e l'Elektronische Musik

Nel 1951 Herbert Eimert fonda a Colonia il primo studio di musica elettronica tedesco finanziato dalla *Westdeutscher Rundfunk* (emittente radio della Germania dell'Ovest); furono proprio due membri di questo gruppo, Werner Meyer-Eppler e Robert Beyer, che l'anno prima coniarono il termine *Elektronische Musik* durante una serie di trasmissioni radio, dove esponevano le nuove possibilità di questo genere musicale.

La musica prodotta nello studio era influenzata in buona parte dalla Seconda scuola di Vienna e guardava con interesse alla dodecafonia di Arnold Schoenberg e Adam Berg, ma sopratutto allo spinto serialismo di Anton Webern.

Karl Heinz Stockhausen, figura emblematica dello studio, estese e sviluppò le idee musicali di Webern[17] fino ad ottenere la totale *serializzazione* delle proprie opere: ogni parametro musicale come frequenza, dinamica, timbro, ritmo, densità, era deciso in funzione della serie numerica scelta, adattata in funzione della variabile che si desiderava controllare. Un approccio del genere era ovviamente incompatibile con l'estetica compositiva della musica concreta, risultando in attrito con lo studio francese del GRM e implicando una necessaria differenza nelle attrezzature: se i francesi

[17] Webern A. (1963). *Verso la nuova musica*. Milano: Bompiani.

preferivano la manipolazione di materiale concreto registrato, a Colonia si prediligeva il suono sintetizzato dagli oscillatori.

Il *Melochord*, ad esempio, disponeva di due sistemi monofonici di sintesi controllati da una tastiera di cinque ottave che permetteva di suonare due note alla volta. Interessante fu l'introduzione di un generatore di inviluppo, con il quale potevano essere cambiati i parametri di attacco, sostegno e decadimento del suono.

Oltre a questo strumento, lo studio disponeva di:

- oscillatori sinusoidali e a dente di sega
- registratore a nastro con velocità di riproduzione regolabile
- il primo registratore del mondo a 4 tracce
- filtri audio, compresi filtri passa-banda
- modulatore ad anello
- generatore di rumore bianco

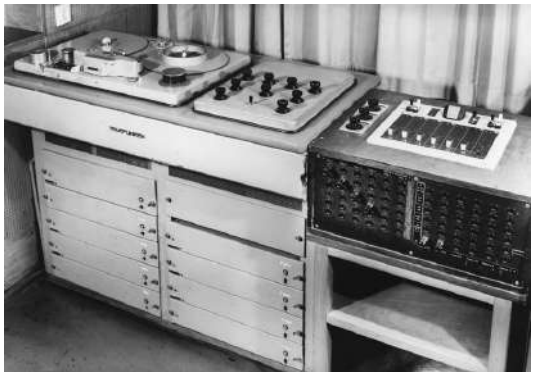


Figura 1.17 Attrezzatura dello studio di Colonia nel 1966. Un banco di filtri e amplificatori (destra) e una console per la registrazione (sinistra). (Stockhausen Verlag)

1.3.3 Lo studio di fonologia della RAI

Il terzo polo europeo per la ricerca musicale venne fondato dalla RAI nel 1955, lo Studio di fonologia musicale di Milano vantò per molti anni la tecnologia più avanzata dell'epoca. La guida di Bruno Maderna e Luciano Berio diede al centro un carattere molto aperto rispetto alla produzione musicale; non ci fu un allineamento estetico né con la corrente francese, né con quella tedesca bensì una ripresa di entrambe dal momento che la musica per i due compositori non poteva avere direzioni sistematiche di

questo tipo. Di conseguenza lo studio disponeva di attrezzatura che riuscisse a soddisfare diverse necessità compositive, comprendendo generatori di segnale, effetti audio e attrezzatura di registrazione e riproduzione.

Lo studio fino al 1960 era composto da questa attrezzatura:

- 9 oscillatori sinusoidali
- generatore di rumore bianco
- generatore di impulsi
- camera riverberante, riverbero a nastro e a piastra
- filtri passa-alto e passa-basso, con 6 frequenze di taglio fra cui scegliere
- filtro passa-banda con frequenza variabile
- banco di filtri a terzi d'ottava
- modulatore ad anello e ad ampiezza
- microfoni
- mixer
- 4 amplificatori ed impianto di diffusione quadrifonica
- 6 registrazioni a nastro monofonici
- 2 registratori a nastro a 2 tracce
- 2 registratori a nastro a 4 tracce

Questo dispiegamento di tecnologia trova però molti compositori dell'epoca piuttosto impreparati; emerge quindi la figura del tecnico del suono, che non si occupa solo della manutenzione e messa in funzione dei macchinari, ma assiste l'artista, in maniera più o meno attiva, durante molte fasi del processo creativo.

Figura 1.18 Il compositore Luigi Nono con Mario Zuccheri, principale tecnico del suono dello studio. (Fondazione archivio LN)

1.4 Il sintetizzatore controllato in tensione

Nel 1948 l'invenzione nei laboratori Bell del *transistor,* con il suo conseguente sviluppo, diede una notevole spinta all'industria elettronica, sebbene sia durante l'inizio del 1960 che avvennero i maggiori sviluppi nel design dei circuiti analogici, con il relativo impatto nella costruzione degli strumenti elettronici.

La nascita del *sintetizzatore analogico* non può essere circoscritta ad un particolare evento o luogo: classico esempio di come una buona idea può nascere simultaneamente in vari contesti per soddisfare un bisogno diffuso. Le attrezzature fino a quel momento a disposizione erano per la maggior parte costruite *ad hoc* per i centri di ricerca; non esisteva un mercato per il pubblico ed il prezzo era conseguentemente elevatissimo; in più non c'era alcuno standard costruttivo: ogni studio aveva i propri sistemi che difficilmente potevano essere integrati con attrezzatura proveniente da altri produttori. I compositori e i musicisti cominciarono a richiedere strumenti che non contenessero necessariamente tutto ciò che si poteva trovare in un centro di ricerca musicale, ma che riuscissero a soddisfare bisogni espressivi individuali, a un costo contenuto e con dimensioni ridotte che permettessero la trasportabilità[18]: la più grande rivoluzione del sintetizzatore consiste probabilmente nell'aver permesso al suono elettronico di svincolarsi dai soli centri di ricerca e studi di registrazione.

Questo è un passaggio molto importante per la storia dello strumento musicale in generale: si passa dalla domanda ai costruttori da parte di un pubblico molto ampio alla necessità da parte dei musicisti di grandi personalizzazioni per appagare il proprio gusto. Lo strumento progettato per molti diventa ora strumento disegnato per il singolo.

Il concetto di *modularità,* spesso associato ai sintetizzatori analogici, è stato introdotto da Harald Bode nel 1960, con il suo *Modular Sound Modification System*. Questo strumento riassumeva in un formato più compatto le possibilità sonore degli studi di registrazione dell'epoca; si componeva di vari moduli per la generazione e modifica sonora, ogni modulo poteva interagire con gli altri tramite dei collegamenti non

[18] Dunn D. (1992). *A history of electronic music pioneers*. Linz: Ars Electronica.

permanenti scelti dal musicista con una conseguente gamma vastissima di possibilità timbriche[19].

Figura 1.19 Bode durante la dimostrazione di uno dei suoi strumenti. (*http://120years.net/the-sound-processor-harald-bode-germany/*)

Ispirato da Bode, nel 1964 Robert Moog comincia la produzione di sintetizzatori modulari, il cui design si rifaceva in buona parte a modelli già esistenti.Semplificandone lo schema, troviamo uno o più oscillatori, il cui segnale viene filtrato e poi processato da un generatore di inviluppo prima di entrare nel circuito di amplificazione. I parametri erano modificabili manualmente o tramite segnali; in alcuni casi si utilizzavano oscillatori a bassa frequenza (LFO), in altri segnali in banda audio dalle varie forme d'onda. Nel 1969, cercando uno strumento dal design compatto, introduce sul mercato il *MiniMoog* che riscontra un enorme successo commerciale. Don Buchla, negli stessi anni, utilizza un approccio molto diverso da Moog, sia dal punto di vista progettuale che estetico: non voleva sintetizzare suoni che si rifacessero a strumenti acustici, né tantomeno sviluppare interfacce familiari per lo strumentista, così da permettere un'esplorazione senza pregiudizi delle nuove possibilità timbriche dello strumento. Buchla disegnò anche sistemi che generavano segnali di controllo randomici che riuscivano a modificare in modo imprevedibile il comportamento di altri moduli, dando

[19] Bode H. (1961). Sound Synthesizer creates new musical effects. *Electronics magazine* (Dicembre 1961).

allo strumento la possibilità di suonare se stesso. Non va sottaciuta la sua particolare sensibilità verso la spazializzazione del suono, progettò dei moduli che davano il controllo su questo parametro musicale fondamentale per la musica elettronica e ancora relativamente inesplorato.

Nel decennio seguente il mercato fu invaso da produttori di sintetizzatori provenienti da tutto il mondo, di seguito sono riportati i principali per area geografica:

- America: Arp, Serge, Oberheim, Moog, Buchla
- Asia: Korg, Roland, Yamaha
- Europa: EMS, Farfisa
- Russia: Polivoks

1.5 Verso la computer music

Analogico è un termine che si riferisce a sistemi dove una data quantità fisica, anche dopo varie trasformazioni (trasduzioni), trova il suo corrispettivo (analogo) in un'altra quantità fisica misurabile. Per spiegare meglio questo concetto si può descrivere la classica catena elettroacustica di registrazione di un suono: ogni stadio del processo è costituito da un diverso sistema fisico, che rimane analogo allo stadio precedente nella misura in cui l'informazione rimane invariata, sebbene rappresentata da una diversa unità di misura. Le molecole dell'aria trasportano l'energia meccanica prodotta da un corpo in vibrazione creando zone di addensamento o rarefazione molecolare, quindi la membrana del microfono capta queste fluttuazioni e le trasduce in un flusso di elettroni. Il segnale elettrico viene amplificato e trasdotto in un campo magnetico dalla testina di scrittura del registratore a nastro. Le variazioni nel campo magnetico, proporzionali a quelle nelle molecole d'aria, incidono sul nastro il segnale captato: il suono è stato così registrato.

Con il termine *digitale* ci si riferisce a un sistema dove una certa informazione è rappresentata attraverso un processo puramente numerico costituito da una successione formata da unità che possono contenere solo due tipi di dati; il bit può avere valore

logico 0 o 1, una quantità è quindi espressa in relazione ai singoli valori, ordinati, dei bit contenuti in una stringa[20].

La musica elettronica, in tutta la sua rapida evoluzione che segue quella tecnologica, inizia a fare i conti con il mondo digitale intorno agli anni Cinquanta, ma i computer erano troppo lenti per eseguire calcoli in tempo reale e spesso i compositori dovevano aspettare ore prima di poter sentire il suono che avevano programmato nella macchina. Per cui il termine *computer music* comprendeva anche il solo sfruttamento di un sistema digitale per generare la partitura o per ottenere dei parametri musicali con calcoli complessi. Il primo uso di un computer per generare suono digitalmente è storicamente associato a Max Mathews, ingegnere dei laboratori Bell che nel 1957 progettò un primitivo sistema di conversione digitale - analogica (DAC), diventando la figura centrale nello lo sviluppo tecnico del computer per scopi musicali e compositivi; tra il 1961 e il 1964 sviluppò un elaborato software per la composizione digitale, il *MUSIC IV*[21]. Nel 1965 la ricerca nei laboratori Bell portò alla prima registrazione digitale di uno strumento acustico: il segnale di una tromba venne registrato, convertito in una rappresentazione numerica interpretabile dal computer, quindi riconvertito nel dominio analogico[22].

La diffusione della computer music viaggia parallelamente con lo sviluppo di interfacce utente più accessibili ed intuitive; all'inizio degli anni Settanta i compositori non devono più essere necessariamente degli esperti di programmazione per utilizzare questo strumento.

Il 1970 vede il primo utilizzo in tempo reale del computer, sfruttandolo per controllare dei parametri di alcuni oscillatori analogici, la frequenza di campionamento di un segnale di controllo di quel tipo era sufficientemente bassa da permettere una gestione live del sistema. Mathews sviluppa così il primo sistema ibrido analogico-digitale, conosciuto come GROOVE, che vantava un'interfaccia utente molto immediata fatta di potenziometri e joysticks.

---

[20] Cremaschi A. e, Giomi F. (2008). *Rumore bianco. Introduzione alla musica digitale.* Bologna: Zanichelli.

[21] Manning P. (1993). *Computer and Electronic Music*. Oxford: Oxford Univ. Press.

[22] Mathews M. (1969). *The Technology of Computer Music*. Cambridge: MIT Press.

Mathews descriverà il sistema in questo modo:

> *La relazione più auspicabile tra il performer ed il computer non è quella di uno strumentista con il suo strumento, ma piuttosto quella di un direttore con la sua orchestra*[23].

Le previsioni di Mathews si concretizzeranno nel trentennio successivo.

Figura 1.20 Mathews con il sistema GROOVE: Generated Realtime Operations On Voltage-controlled Equipment (*120years.net*)

1.5.1 Digital signal processing (DSP)

All'inizio degli anni Ottanta, la disponibilità di computer sempre più performanti rese la sintesi e manipolazione sonora digitale in tempo reale una realtà: Giuseppe Di Giugno, con il suo team di ricerca all'IRCAM di Parigi, sviluppa nel 1981 il *sistema 4X*; questo dispositivo è frutto di una ricerca quinquennale nel campo dell'elaborazione e sintesi sonora digitale in tempo reale e permette la modifica live di qualsiasi suono proveniente dal computer stesso o da una fonte esterna. In questo modo gli strumenti acustici hanno la possibilità di essere integrati nel mondo software, espandendo le proprie possibilità timbriche ed espressive e soddisfacendo la crescente richiesta dei compositori di poter sfruttare la tecnologia digitale nelle loro opere live.

[23] Moore R. (1969). *Elements of computer music*. Upper Saddle River: Prentice-Hall Inc.

Figura 1.21 Giuseppe di Giugno con il suo sistema digitale modulare 4X, 1979 IRCAM. (Archivi IRCAM)

Dal 1983 IBM e Apple invadono il mercato con il *personal computer*: il computer non è più una tecnologia tanto costosa da poter essere utilizzata solo in centri di ricerca, diventa uno strumento alla portata di molti privati. Cominciano a comparire di conseguenza i primi software commerciali per la produzione musicale e lo stesso IRCAM, nel 1985, rilascerà dei programmi specificatamente sviluppati per il personal computer. Nei decenni successivi il computer rappresenta l'unico strumento necessario alla produzione di musica elettronica, riuscendo a racchiudere le funzioni dei più elaborati studi di ricerca musicale del passato e superandone di gran lunga le potenzialità.

1.5.2 Hyperinstruments

Nella prima metà del Novecento, prima che il DSP diventasse una realtà, erano cominciate sperimentazioni per la modifica timbrica e funzionale degli strumenti acustici, resi ibridi da elementi estranei alla loro natura: un esempio è il *pianoforte preparato* di John Cage che, con l'inserimento di chiodi e bulloni all'interno della cordiera dello strumento, riusciva a guadagnare nuove capacità espressive.

Lo sviluppo tecnologico degli ultimi due decenni ha permesso di integrare la sensoristica gestita dai microprocessori con gli strumenti classici.

Il termine *hyperinstrument*[24] è stato coniato da Tod Machover, ricercatore del Massachusetts Institute of Technology (MIT), per identificare la pratica con la quale si

[24] Machover T. (1992). *Hyperinstruments: A progress Report 1987 - 1991*. Cambridge: MIT Media Laboratory.

aumentano le potenzialità di uno strumento acustico grazie all'utilizzo di sensori elettronici: le informazioni rilevate dai sensori sono sfruttate per generare eventi sonori o per manipolare il suono acustico dello strumento stesso.

Esiste una grande varietà di sensori, sfruttati per monitorare le variabili più disparate, dai quali si possono ottenere informazioni provenienti sia dallo strumento che dallo strumentista; questo permette, ad esempio, una generazione o una manipolazione sonora perfettamente coerente con il gesto che sta eseguendo il musicista, proprio perché scaturita dal gesto stesso.

Questo è uno degli esempi più lampanti di come lo sviluppo tecnologico del Ventesimo secolo abbia stravolto il modo di concepire lo strumento musicale, rivoluzionando l'intimo rapporto tra *strumentista*, *compositore* e *strumento*; nel capitolo successivo verrà approfondito ulteriormente questo legame, nella misura in cui i confini che delimitavano palesemente queste tre figure diventano meno marcati e in alcuni casi addirittura inesistenti.

2 STRUMENTI E SISTEMI MUSICALI GENERATIVI

"Con l'avvento della tecnologia elettrica l'uomo estese, creò cioè al di fuori di se stesso, un modello vivente del sistema nervoso centrale."

Marshall McLuhan

Il liutaio rinascimentale sapeva di poter presentare i propri prodotti in un mercato non molto vasto, ma sicuramente definito: un suo violino ben costruito sarebbe stato apprezzato da un gran numero di violinisti, che grazie ad esso avrebbero potuto eseguire un qualsiasi brano della prassi musicale dell'epoca; più violini strutturalmente identici tra loro riuscivano ad appagare le necessità della maggior parte di strumentisti e compositori, lo stesso strumento era concepito *per molti*. Questa proporzione di *un modello per tanti* inizia a venire meno con l'avvento dell'elettronica, soprattutto nel momento in cui il compositore comincia a progettare autonomamente i propri strumenti, ponendo specifiche richieste ai costruttori o diventando egli stesso liutaio. Spesso i brani sono scritti per particolari e unici sistemi, hardware o software, disegnati per soddisfare appieno specifiche necessità compositive: lo strumento acquista valenza *per pochi* o solo *per il singolo*. Il rapporto di *uno per uno* subisce un'ulteriore modifica quando l'idea di strumento si svincola dalla sola funzione di produzione sonora, incaricandosi anche del ruolo di parte attiva nella composizione: lo strumento è costruito *per se stesso*.

E' in quest'ottica che va definita la *musica generativa*, come un'arte creata in parte o in pieno da un sistema autonomo che non sia l'uomo, *capace* di determinare indipendentemente parametri che richiederebbero altrimenti l'intervento dell'artista.

2.1 Generatività ottenuta da variabili di un sistema meccanico

Un sistema è un apparato che, pur essendo costituito da diversi elementi reciprocamente interconnessi, interagenti tra loro o con l'ambiente esterno, reagisce ed evolve come un tutto, con proprie leggi particolari[25].

Il comportamento di un sistema meccanico può essere approssimato più o meno fedelmente da un certo numero di equazioni, capaci di determinarne lo stato e l'evoluzione nel tempo in funzione delle necessarie variabili considerate, la cui variazione prescinde nella maggior parte dei casi dall'intervento dell'uomo. Quindi un apparato meccanico opportunamente progettato può essere sfruttato per generare suono automaticamente, condizione necessaria ma non sufficiente per soddisfare la definizione di generatività: un carillon, anche nella sua versione più elaborata conosciuta come *automa meccanico*[26], è un sistema meccanico musicale che può solo riproporre un limitato numero di brani composti a monte dal suo costruttore.

*Figura 2.1*Automi costruiti nel 1733 da Pierre Jaquet-Droz, oggi ancora funzionanti: uno scrivano (sinistra), un disegnatore (destra) e un musicista (centro) meccanici riuscivano a riprodurre automaticamente piccoli poemi, disegni e motivati musicali. (Musée d'Art et d'Histoire de Neuchatel)

I *tintinnabulum*[27] dell'antica Roma o i *shishi odosh*i giapponesi sono esempi di sistemi meccanici sonori che sfruttano unicamente variabili del mondo fisico per generare eventi sempre diversi nel tempo; sebbene questa generazione prescinda dall'intervento dell'uomo,

[25] Sistema. *In Enciclopedia Treccani*. Roma: Treccani.

[26] Winter-Jensen A. (1987). *Automates et musiques pendules*. Geneve: Musee De L'Horlogerie et De L'émaillerie.

[27] Bonfante L. (1986). *Etruscan Life and after Life: A handbook of etruscan studies*. Detroit: Wayne State University Press.

anche in questo caso è difficile affermare che siano strumenti generativi: i suoni sono prodotti in maniera casuale e non seguono nel tempo alcuna logica musicale definita.

Figura 2.2 Tintinnabulum con la forma di un gladiatore intento a combattere il proprio fallo, rinvenuto a Ercolano; aveva compito apotropaico, cioè allontanare il malocchio e portare prosperità. Il sistema sfrutta il vento per far impattare reciprocamente quattro campanelle di bronzo, fissate alla struttura centrale da cordini o catenine di diversa lunghezza. (Gabinetto Segreto del Museo Archeologico Nazionale di Napoli)

Figura 2.3 Shishi odoshi, letteralmente spaventa-cervi, tradizionalmente usato per allontanare gli animali dalle coltivazioni e in tempi moderni come arredo da giardino. L'acqua riempie un cilindro di bambù, vincolato alla base del sistema da una cerniera a perno: quando la massa d'acqua supera quella posta all'altro capo del tronco, la leva non è più in equilibrio e il legno andrà a impattare sulla roccia sottostante, svuotandosi e producendo suono. Così il cilindro riguadagna la sua posizione iniziale, facendo cominciare nuovamente il ciclo. (Wikimedia Commons)

Quindi per definire uno strumento *generativo* non basta attribuirgli la capacità di produrre automaticamente suoni casuali sempre diversi nel tempo; come ogni altro strumento musicale deve poter produrre musica, differentemente da ogni altro strumento musicale deve poterlo

fare senza l'intervento di un musicista umano: necessita perciò di una gestione temporale degli eventi sonori, di un'evoluzione che abbia valenza estetica.

Il compositore Steve Reich, nel 1965, costruisce uno dei primi sistemi musicali meccanici le cui caratteristiche assecondano il concetto di generatività[28]: *It's gonna rain* è un brano composto con l'ausilio di due registratori a nastro che riproducono lo stesso materiale concreto a velocità leggermente diversa. Il compito di Reich è stato *solo* quello di scegliere il materiale sonoro, l'opportuna velocità di rotazione dei due registratori e la loro condizione di partenza; il resto della composizione si sviluppa autonomamente sfruttando la lieve differenza nella velocità angolare dei motori, facendo evolvere i suoni con sovrapposizioni sempre diverse. La variabile meccanica non è sfruttata per cambiare percettivamente la frequenza del nastro, ma per dare ai due segnali una leggera differenza nella fase, ottenendo un risultato riconducibile al fenomeno fisico dei *battimenti*.
Si immagini di semplificare il segnale riprodotto dal primo registratore con una sinusoide di frequenza 1 Hz; Reich riduce leggermente, circa dello 0.002%, la velocità del secondo registratore, che secondo la semplificazione precedente avrà quindi frequenza di 0.998 Hz. Seguendo questo modello, ponendo entrambi i segnali con ampiezza unitaria avente lo stesso valore di partenza, la frequenza di battimento sarà pari a:

$$F_{batt} = F_1 - F_2 = 1 - 0.998 = 0.002 \text{ Hz}$$

Da cui è possibile ricavare il tempo necessario per cui i due segnali ritornino in fase, riottenendo l'unisono dei due nastri[29]:

$$T_{ciclo} = 1/F_{batt} = 1/0.002 = 500 \text{ s} = 8 \text{ min e } 20 \text{ s}$$

Questo modello molto semplificato restituisce l'idea sulle potenzialità generative del sistema: modificando una sola variabile meccanica si riesce ad ottenere un'interessante evoluzione nel

[28] Wright W. e Eno B. (2006). *Playing with time*. YouTube: Will Wright and Brian Eno - Generative Systems.

[29] Il brano dura 17 minuti e 50 secondi, è diviso in due sezioni di pari lunghezza che utilizzano due diversi nastri magnetici. Il cambio di sezione avviene quindi all'incirca nel momento i cui i due registratori con il primo nastro ritornano in fase.

tempo del materiale sonoro. L'approccio alla composizione basato sul *phasing* viene definito da Reich *per processo*, oggi identificato equivalentemente con minimalismo, dove il termine non fa riferimento a particolati processi compositivi, ma a brani che sono essi stessi processi[30].

Con *Pendulum Music* il compositore fornisce un altro esempio interessante di sistema generativo meccanico: tre o più microfoni pendono da un supporto sospeso, in modo che siano equidistanti dal suolo e liberi di compiere un moto oscillatorio. Ad ogni segnale microfonico è associato un amplificatore e un altoparlante: ogni microfono, nella sua posizione di perpendicolarità col suolo, dista pochi centimetri dal rispettivo altoparlante in modo da innescare il fenomeno di feedback acustico. Gli esecutori, uno per microfono, dovranno semplicemente avviare il moto armonico variando l'angolo del pendolo con la perpendicolare del suolo, rilasciando tutti i microfoni contemporaneamente. Il resto della composizione sarà *eseguita* dall'energia potenziale gravitazionale, dall'energia cinetica dei microfoni e dal loro attrito con l'aria; la durata è variabile in funzione della massa dei microfoni, lunghezza del filo e angolo iniziale, il brano si conclude nel momento in cui il moto oscillatorio termina producendo bordoni di feedback da ogni cassa.

Figura 2.4 Estratto della performance di *Pendulum Music* al CENTRE JOSE GUERRERO (Granata, Spagna). Interpreti: Joan Cerverò, Vìctor Trescolì, Isabel Leòn, Estefania Sànchez. Si notino i quattro microfoni oscillanti e i relativi altoparlanti necessari all'innesco del feedback. (Youtube)

Brian Eno è uno dei principali compositori divulgatori della musica ambient e generativa, ispirato da *It's gonna Rain,* costruisce un sistema analogo per produrre parte dei brani del suo album *Music for Airports*, del 1978. Utilizza sette registratori a nastro con uguale velocità di

[30] Reich S. (1968). *Music as a Gradual Process*.

rotazione, ma con loop di lunghezza diversa; su ogni loop è incisa una nota con una certa percentuale di silenzio, in modo che il sistema riproponga la stessa successione di materiale sonoro dopo un tempo più grande di svariati ordini di grandezza rispetto alla composizione di Reich.

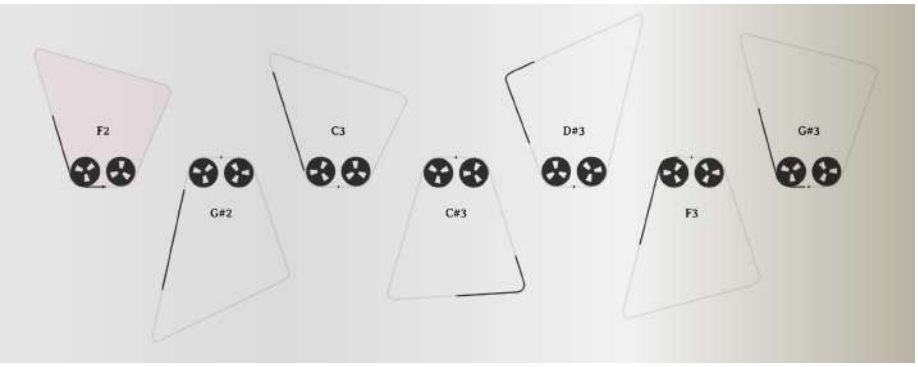


Figura 2.5 Modello del sistema utilizzato da Eno per realizzare la seconda traccia dell'album Music for Airports. Il segmento più scuro di ogni pista rappresenta la porzione di nastro su cui è registrato suono. Si noti come il sistema è progettato per assecondare la tonalità: i nastri con Do e Do#, il cui unisono risulterebbe dissonante, hanno lunghezza simile e sono molto sfasati tra loro, per cui suoneranno contemporaneamente con meno incidenza; Fa e Sol invece si trovano quasi perfettamente in fase con i rispettivi intervalli consonanti di ottava. (How Generative Music Works: *https://teropa.info*)

Questa tipologia di strumenti meccanici generativi fonda il proprio comportamento su variabili fisiche sfruttate secondo le necessità del compositore, che non ha ovviamente alcun modo di intervenire intimamente sulle leggi naturali affinché assecondino ulteriormente le proprie intenzioni; perciò in alcuni casi può essere utile progettare sistemi che seguano regole dettate unicamente dall'artista, un pattern di istruzioni da poter seguire più o meno pedissequamente per raggiungere lo scopo compositivo.

2.2 Musica e algoritmi

Un algoritmo è una sequenza finita e ordinata di istruzioni che definiscono senza ambiguità le operazioni da eseguire per raggiungere un determinato risultato in un tempo non infinito[31].

Gli algoritmi, o più in generale vari insiemi di regole formali, sono utilizzati da secoli per comporre musica; le tecniche utilizzate, ad esempio, nell'organizzazione melodica delle voci di un contrappunto possono essere facilmente ricondotte a processi algoritmici: ogni voce del contrappunto viene sviluppata dal compositore secondo il proprio gusto, attingendo però da

[31] Questa definizione informale prende spunto da modelli matematici come la macchina di Turing.

un limitato insieme di operazioni eseguibili sulla linea melodica; linee melodiche diverse, che seguono quindi *istruzioni* diverse, devono poter coesistere contemporaneamente in modo da avere valenza estetica e soddisfare nella loro totalità le regole dell'armonia.
Ovviamente la bellezza di una Fuga di Johann Sebastian Bach contenuta nel *Clavicembalo Ben Temperato* non può essere sintetizzata con una riduzione così banale, ma la semplificazione risulta interessante se, con una certa astrazione, si considera il contrappunto come un sistema formalmente composto da singoli elementi (le voci) che si sviluppano indipendentemente seguendo determinate istruzioni (le operazioni eseguibili melodicamente) al fine di ottenere uno scopo comune (la costituzione di una Fuga) che soddisfi generalmente delle *meta-istruzioni* (le regole armoniche).

Su queste basi si fonda la moderna concezione di *composizione algoritmica*: piuttosto che comporre un brano, l'artista organizza un sistema in modo che segua autonomamente una serie di regole, più o meno restrittive, arrivando alla generazione della composizione finale.
Quanto più il margine di interpretazione delle regole sarà vago, tanto più saranno vari e imprevedibili i risultati ottenuti dal sistema; sebbene una componente aleatoria possa apparire in contrapposizione con la definizione di algoritmo proposta all'inizio del paragrafo, risulta essere molto interessante dal punto di vista musicale, facendo emergere elementi e rapporti tra le parti non concepiti in partenza dal compositore. Provocatoriamente, per cercare di assecondare il concetto di algoritmo, si potrebbe pensare di introdurre una regola che permetta di non seguire necessariamente tutte le altre regole proposte: variandole in corso d'opera secondo necessità, ignorandole o creandone di nuove[32]. (Si potrebbe introdurre anche una regola che permetta di ignorare qualsiasi regola compresa se stessa, facendo però inciampare l'algoritmo in problemi di ricorsività e paradossi.)

2.2.1 Alea controllata e sistemi generativi sociali

Serenata per un Satellite è una composizione del 1969 con cui Bruno Maderna esplora le possibilità musicali dell'alea controllata: la partitura contiene note scritte in moduli intercambiabili, disposti graficamente nella pagina in modo da consentire agli esecutori di seguire arbitrariamente percorsi diversi ad ogni performance, che ha perciò durata variabile

[32] In effetti è anche su considerazioni di questo tipo che sono sviluppati molti sistemi tendenti all'intelligenza artificiale, le cui caratteristiche verranno analizzate nell'ultimo paragrafo di questo capitolo.

dai 4 ai 12 minuti. Gli strumentisti hanno un ruolo importante nell'interpretazione dell'opera: nonostante le libertà dei *singoli*, devono creare come *insieme* un percorso musicale coerente, che soddisfi le condizioni poste in partenza dal compositore.

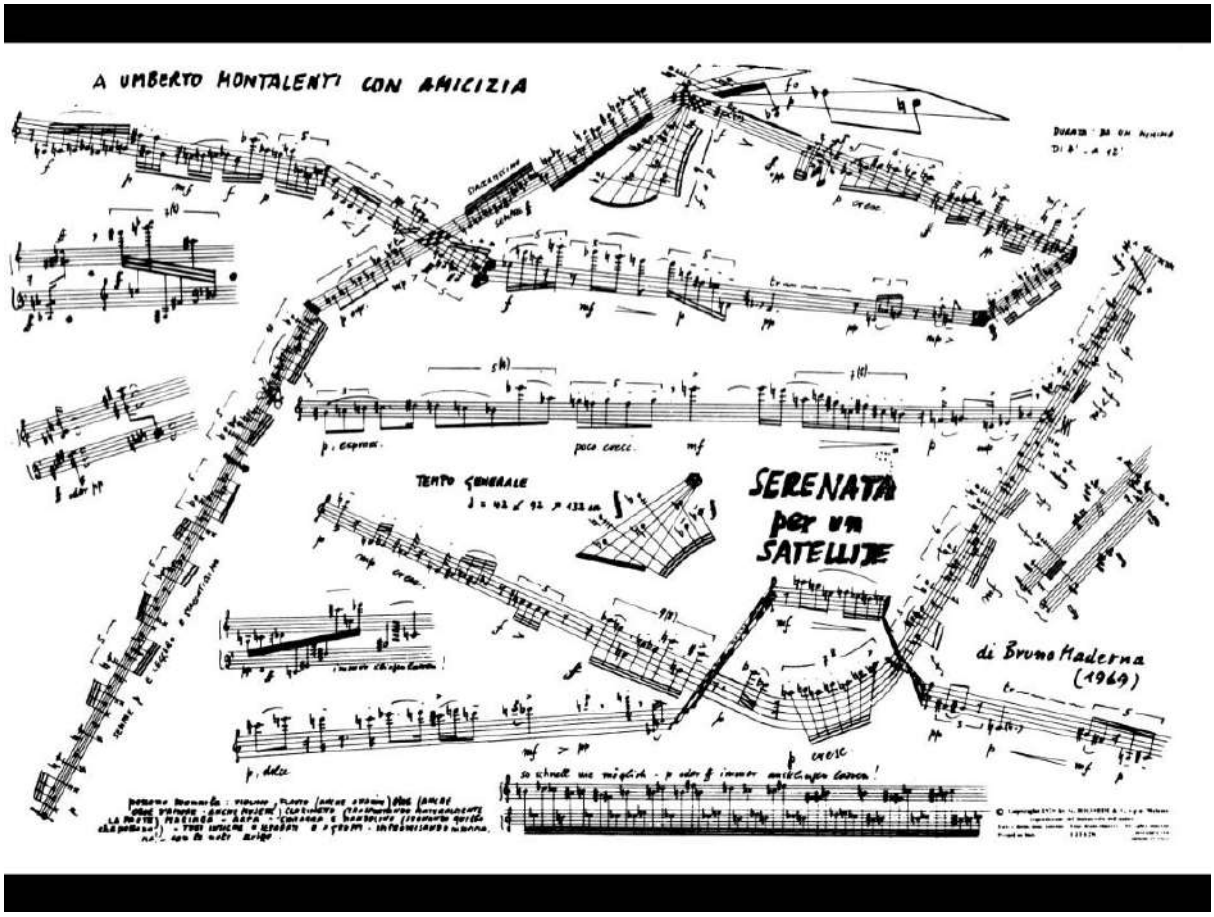


Figura 2.6 Partitura di *Serenata per un Satellite*; con questo unico foglio il compositore fornisce tutte le indicazioni necessarie all'esecuzione dell'opera. Il brano è dedicato al fisico Umberto Montaletti, che aveva progettato e coordinato il lancio del satellite ESTRO I, utilizzato per lo studio delle aurore boreali.

La partitura equivale, in un certo senso, alle leggi fisiche che regolano il comportamento di un sistema meccanico generativo; in questo caso, però, è impossibile prevedere a priori il risultato finale, non avendo modo di ascrivere le innumerevoli variabili a semplici equazioni.
La *scienza generativa* si occupa di questo tipo di fenomeni: indaga il mondo naturale e sociale, analizzandone gli infiniti comportamenti complessi a partire da parametri finiti e deterministici[33]. Da questo punto di vista le istruzioni contenute in *Serenata per un Satellite*,

[33] Crnkovic G., e Giovagnoli R. (2013). *Computing nature: Turing centenary perspective.* Berlino, New York: Springer.

rendono l'ensemble che le segue un semplice esempio di sistema generativo *sociale* con finalità musicali.

In C, di Terry Riley, fornisce un altro spunto di riflessione su questa tipologia di sistemi[34]. La partitura contiene 53 brevi frasi musicali numerate e ordinate, l'organico non è specificato, sebbene il compositore indichi una preferenza per circa 35 strumenti, di cui la maggior parte a suono determinato. L'esecutore può ripetere ogni frase un numero arbitrario di volte, se la stessa frase è suonata da più strumenti non c'è necessità che venga eseguita in unisono; la scelta della frase successiva deve seguire l'ordine dei numeri segnati nella partitura, sebbene si abbia la facoltà di non scegliere necessariamente il numero seguente.
La libertà di esecuzione è limitata da un'altra regola molto interessante: lo strumentista non può trovarsi a più di due o tre frasi di distanza da tutti gli altri elementi dell'ensemble, imponendogli un parametro *oggettivo di feedback* sul rapporto della propria esecuzione con il totale, che non sia solo percettivo come per il brano di Maderna.

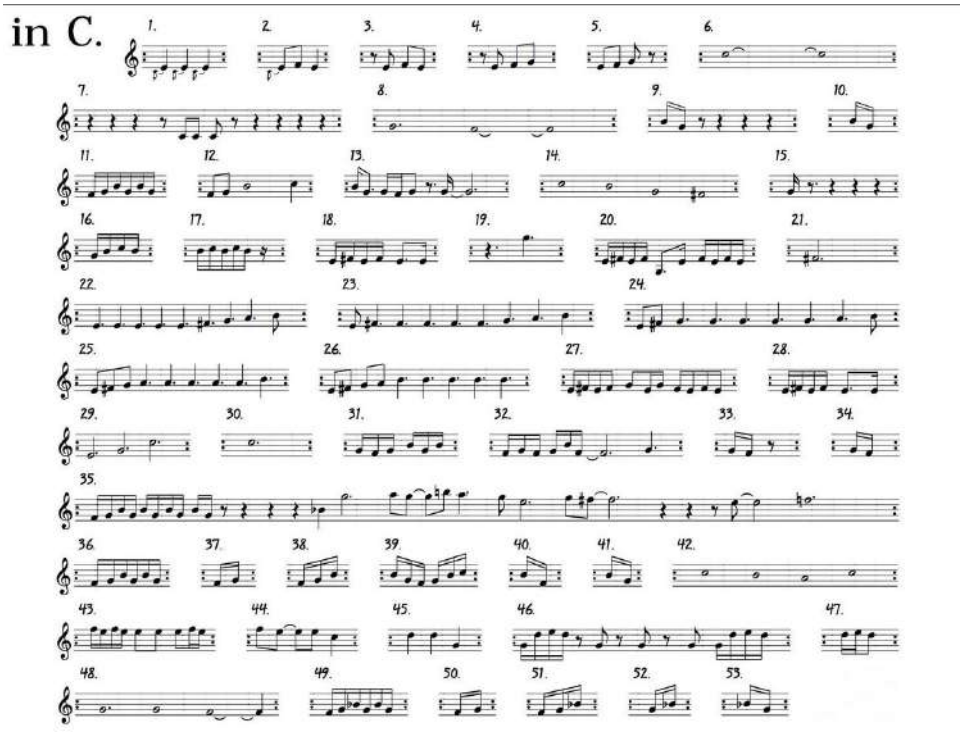


Figura 2.7 Partitura di *in C,* del 1964. Per la sua natura aleatoria la composizione può durare dai 45 minuti a circa il doppio. Si notino le 53 frasi musicali aventi lunghezze diverse. L'insieme di regole per gli strumentisti è contenuto in un'altra sezione della partitura.

[34] Capitoni F. (2016). *In C, opera aperta. Guida ai capolavori di Terry Riley.* Roma: Arcana.

2.2.2 John Cage e il caso come strumento compositivo

I concetti di algoritmo e aleatorietà trovano massima espressione nelle opere del compositore John Cage: la casualità non risiede solo nell'interpretazione delle regole che fornisce per eseguire i propri brani, ma anche nella scelta stessa delle regole e dei parametri necessari alla produzione delle proprie composizioni. Il suo metodo compositivo si basa sul porre delle domande, piuttosto che fare delle scelte, definendo un insieme di risposte ugualmente valide; il ruolo del compositore deve essere solo quello della creazione di uno spazio di possibilità e, per non influenzare ulteriormente l'opera con il suo gusto personale, può attingere a queste informazioni utilizzando algoritmi basati sul caso.

L'*I Ching* è un testo classico della cultura cinese, risalente al X secolo a.C., utilizzato popolarmente a scopo divinatorio; è stato introdotto in Occidente nel 1697 dal matematico Wilhelm von Leibniz, che vide nei 64 esagrammi contenuti nel libro un ottimo esempio di numerazione binaria[35]: ogni esagramma è composto da 6 linee che posso essere spezzate (valore logico 0) o continue (valore logico 1), la scelta delle quali avviene con un lancio di monete. Cage utilizza spesso gli esagrammi dell'I Ching nelle proprie opere, associando ad ogni simbolo il valore di uno o più parametri, che saranno perciò solo dipendenti dal risultato casuale dei lanci. Riassume così questa tecnica compositiva[36]:

Quindi le risposte, invece di venire dalle mie simpatie e antipatie, provengono da operazioni casuali, e questo ha l'effetto di aprirmi a possibilità che non avevo considerato. […] Le risposte determinate dal caso apriranno la mia mente al mondo intorno.

In *Imaginary Landscape No. 4,* il compositore sfrutta questo metodo per la generazione della partitura, in cui è segnato come 24 strumentisti devono interagire con 12 radio AM, modificando la frequenza di sintonizzazione, l'ampiezza del segnale e il timbro. La variazione di questi 3 parametri è indicata precisamente nello spartito e, differentemente dalle altre composizioni esposte in questo paragrafo, non ha ampio margine di libertà interpretativa: la struttura e la durata del brano rimarranno invariate ad ogni esecuzione. Le componenti generative e aleatorie del brano risiedono nell'algoritmo di scrittura e nel risultato sonoro,

[35] Leibniz W. (1705). *Spiegazione dell'aritmetica binaria*. Saggio.

[36] Cage J. (2008). *Silenzio*. Milano: Shake.

necessariamente variabile in funzione del broadcasting radiofonico della nazione in cui l'opera viene eseguita.

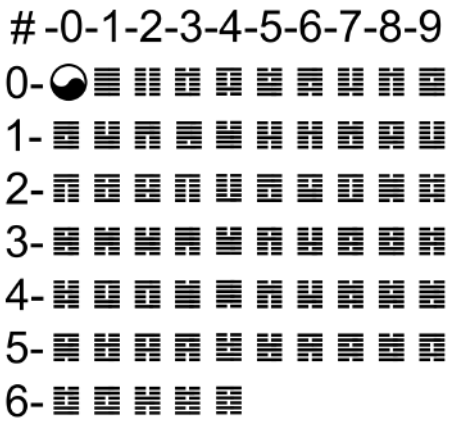


Figura 2.8 Tabella che riporta i 64 esagrammi contenuti nell'I Ching. Il metodo con cui si ottiene il responso divinatorio non è molto lontano da come viene interpretata una tabella di verità dell'algebra booleana. Gli stessi esagrammi hanno una perfetta corrispondenza in una stringa binaria a 6 bit. (Wikimedia Commons)

I brani qui esposti presentano un *prodotto* generativo, nuovo ogni volta a prescindere dal compositore grazie alla natura degli algoritmi e dei sistemi utilizzati, ma che necessita di interpreti umani. Nel paragrafo precedente, invece, Eno e Reich offrono esempi di *metodi* generativi: il brano è indipendente nella sua messa in opera dall'intervento umano, ma rimane pressoché invariato ad ogni esecuzione. Le due tipologie posso essere sfruttate contemporaneamente progettando uno strumento musicale totalmente autonomo che ponga a se stesso delle domande, scegliendo tra un insieme di possibilità le informazioni utili per l'organizzazione e la generazione del suono. In questo senso il computer rappresenta la scelta più performante per la creazione di un sistema generativo che abbia diversi gradi di libertà: il software si presta perfettamente all'implementazione di un gran numero di algoritmi, con operazioni e calcoli che possono essere molto più elaborati e complessi rispetto a quelli gestibili da compositori ed esecutori umani.

2.3 Generatività, software e algoritmi

I primi esperimenti che vedono un computer coinvolto come parte attiva nella composizione prendono spunto dai processi aleatori della musica contemporanea. Come si è visto, Cage decide di non avere il tradizionale controllo diretto sul materiale sonoro delle proprie opere, lasciando molte decisioni al caso e per farlo utilizza metodi spesso meccanici; l'interesse

risiede del risultato casuale delle operazioni piuttosto nel mezzo con cui esse sono ottenute: non è quindi strettamente necessario che sia un uomo a gestire un processo banale come il lancio di una moneta, oltretutto facilmente implementabile in un software.
Il risultato del lancio di un dado, non truccato, a 6 facce è matematicamente rappresentato da una variabile che può assumere, con pari possibilità di incidenza del 16%, uno dei 6 valori. Un computer può riprodurre agilmente un modello probabilistico del genere[37], simulando il lancio contemporaneo anche di migliaia di dadi, associandone poi il risultato a parametri musicali. Sebbene questa possa essere una base di partenza, un software fondato unicamente sul caso sarebbe molto più simile al tintinnabulum romano che a uno strumento generativo, continuerebbe a mancare la produzione di materiale temporalmente organizzato. Motivo per cui, già dalle prime sperimentazioni che coinvolgono il computer, si è cercato di circoscrivere la casualità entro limiti musicalmente utili.

Illiac Suite, del 1957, è una composizione per quartetto d'archi divisa in 4 movimenti, storicamente identificata come la prima opera la cui partitura è interamente generata da un computer[38]. Lejaren Hiller e Leonard Issacson sviluppano il software in modo che scelga casualmente il ritmo, la durata e l'altezza delle note, utilizzando tecniche generative differenti per ogni movimento.
Il quarto movimento, in particolare, sfrutta delle tavole di probabilità per controllare la distribuzione degli intervalli melodici dei singoli strumenti; ogni due battute queste tavole vengono variate modificando la probabilità di ripetere la stessa nota o passare all' intervallo melodico successivo. La prima tabella, quindi le prime due battute, ha il 100% di probabilità di ripetere la stessa altezza e lo 0% di passare a un intervallo diverso, motivo per cui il movimento si apre con la stessa nota sostenuta per tutte le voci. Con il proseguire delle battute, il rapporto tra ripetizione e movimento melodico varia, arricchendo il *vocabolario* prima con intervalli di ottava, poi di quinta, di quarta, di terza maggiore, arrivando fino alla seconda minore e alla settima maggiore. Ogni tavola è programmata in modo che gli intervalli consonanti abbiamo maggiore incidenza: l'unisono è più probabile dell'ottava, che è più probabile della quinta, che è più probabile della quarta e via di seguito. Questo sistema è utile

[37] Verrà poi chiarito il rapporto tra numeri randomici e pseudo-randomici.

[38] Baggi D. L. (1998). *The Role of Computer Technology in Music and Musicology*.

per avere coerenza orizzontale tra le note, trascurando però la verticalità armonica, che sarà solo un *effetto collaterale* dell'organizzazione melodica delle singole voci. La randomicità che caratterizza un processo del genere, rende la composizione poco elaborata anche dal punto di vista strutturale: il programma si concentra su eventi istantanei, senza riscontro effettivo rispetto alle scelte che ha già fatto, che potrà fare e in generale sulla totalità della composizione[39].

Per risolvere questo problema si può ulteriormente limitare il software facendogli seguire modelli analitici, intrinsecamente già solidi, in modo che generi più facilmente materiale strutturalmente coerente. La base di partenza per approcci di questo tipo risiede nella *teoria generativa della musica tonale*[40] di Fred Lerdahl e Ray Jackendoff, a sua volta ispirata dalla *grammatica generativa*[41] di Noam Chomsky.

2.3.1 Grammatica generativa

La grammatica generativa è una teoria linguistica, formalizzata da Chomsky alla fine degli anni Cinquanta, che considera la grammatica come un sistema di regole capace di formare combinazioni di parole, generando frasi sintatticamente coerenti in un dato linguaggio. Può essere rappresentata come un algoritmo utile sia per decidere se una certa frase ha valenza grammaticale, sia per generare teoricamente infinite stringhe *ben formate*[42] a partire da un insieme finito di regole. Uno dei modelli utilizzati più spesso per generare stringhe coerenti, principalmente nell'ambito delle grammatiche libere dal contesto[43], prevede un sistema di riscrittura ad albero, dove un elemento è inserito secondo una precisa regola formale all'interno della struttura, che lo mette in relazione logica con gli altri elementi presenti all'interno di essa.

---

[39] Sandred O., Laurson M., e Kuuskankare M. (2009). *Revisiting the Illiac Suite*. Articolo: www.researchgate.net

[40] Lerdahl F., e Jackendoff R. (1983). *A Generative Theory of Tonal Music*. Cambridge: MIT Press.

[41] Chomsky N. (1956). *Three Models for the Description of Language*. Cambridge: MIT Press.

[42] Nella logica matematica una formula ben formata è una stringa di simboli di un sistema formale che, intuitivamente, rappresenti un'espressione sintatticamente corretta, definita mediante le regole grammaticali del sistema formale stesso. *Enciclopedia della matematica Treccani* (2017).

[43] Hopcroft J., e Ullman J. (1979). *Introduction to Automata Theory, Languages, and Computation* (pp. 77 - 106 e 125 - 137). Boston: Addison-Wesley.

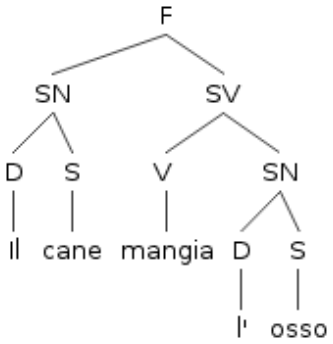


Figura 2.9 Semplice esempio di albero sintattico utile per descrivere le grammatiche libere dal contesto. F è la frase, composta da un sintagma nominale (SN) e un sintagma verbale (SV), a loro volta contenenti articoli (D), sostantivi (S) e verbo (V). (Wikimedia Commons)

La rivoluzione di questa teoria fu quella di fornire un approccio matematico e analitico allo studio della linguistica, di conseguenza permettendo lo sviluppo di software, detti *parser*, capaci di generare e gestire alberi sintattici anche molto più elaborati di quello proposto in figura 2.9. Un programma di questo tipo è applicabile anche al linguaggio musicale e risulta vantaggioso dal punto di vista strutturale, poiché si riesce a scomporre analiticamente un brano partendo dalle macro-sezioni per arrivare ai singoli elementi che le compongono.

Nell'ottica della musica generativa, il software può gestire autonomamente l'inserimento delle figure all'interno dell'albero: il processo può essere lasciato al caso senza preoccuparsi dell'incoerenza nell'organizzazione del materiale, che sarà fornita implicitamente dalle regole che hanno portato alla costituzione del grafo. Esistono altre tipologie di grafi ad albero basate su grammatiche formali, che possono risultare interessanti per la progettazione di uno strumento generativo.

2.3.2 Sistema-L

Un sistema-L è un sistema di riscrittura ad albero che, similmente alla grammatica generativa, consiste in un alfabeto di simboli, capaci di produrre delle stringhe seguendo una serie di regole. La particolarità è che esiste una stringa iniziale, un *assioma*, da cui parte tutta la costruzione delle stringhe successive e, parallelamente, una funzione che le traspone in strutture geometriche. Il nome deriva da Aristid Lindenmayer, botanico e biologo ungherese,

che sviluppa questo sistema per descrivere formalmente il comportamento delle cellule vegetali e per modellare il processo di crescita dei rami delle piante[44].

Più in generale, la sua natura ricorsiva è utile per la descrizione di strutture composte da frattali: incrementando il livello di ricorsività ad ogni iterazione, si passa da elementi semplici a geometrie sempre più complesse.

La grammatica G di un sistema-L è rappresentata dalla seguente ennupla:

$$G = (V, \omega, P) \qquad (\textit{Equazione 2.2})$$

Dove:

- V è l'alfabeto di simboli esistenti, che possono essere variabili o costanti. L'insieme di uno o più simboli costituisce una stringa
- ω è l'assioma, cioè la stringa formata da elementi di V che rappresenta lo stato iniziale del sistema
- P è l'insieme di regole di produzione, che permettono la generazione di tutte stringhe a partire da ω. Una produzione descrive il modo il cui una variabile può essere sostituita da altre variabili o costanti ed è costituita da una stringa di partenza (predecessore) e una stringa di arrivo (successore)

Le regole sono applicate in successione a partire dallo stato iniziale, con la costrizione che per ogni iterazione è necessario utilizzare contemporaneamente, su ogni variabile della stringa, tutte le regole di produzione possibili.

Un esempio molto interessante di questo sistema è proposto dallo stesso Lindenmayer per modellare la crescita di una particolare alga[45], la cui grammatica è così composta:

- variabili: A, B
- costanti: nessuna
- assioma: A
- regole di produzione: (A -> AB), (B -> A)

---

[44] Lindenmayer A. (1968). *Mathematical models for cellular interaction in development*. In: Journal of Theoretical Biology. Volume 18, pp (300—315).

[45] Lindenmayer A., e Prusinkiewicz P. (1990). *The Algorithmic Beauty of Plants*. New York: Springer-Verlag.

Che restituisce per le prime 8 iterazioni:

$n = 0$: A

$n = 1$: AB

$n = 2$: ABA

$n = 3$: ABAAB

$n = 4$: ABAABABA

$n = 5$: ABAABABAABAAB

$n = 6$: ABAABABAABAABABAABABA

$n = 7$: ABAABABAABAABABAABABAABAABABAABAAB

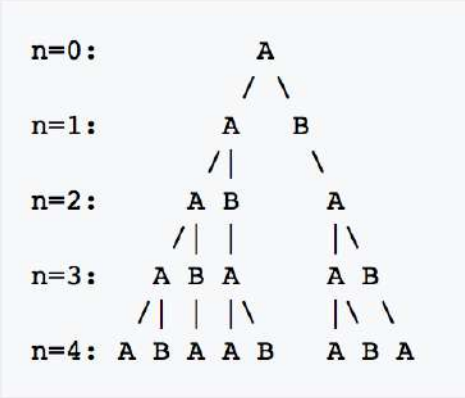


Figura 2.10 Grafo ad albero del modello di crescita dell'albero, rappresentato da 5 iterazioni. Incidentalmente, contando il numero di simboli presenti a ogni passaggio, si ottiene la serie di Fibonacci (primo termine escluso a causa dell'assioma iniziale): 1, 2, 3, 5, 8, 13, 21, 34. (Wikimedia Commons)

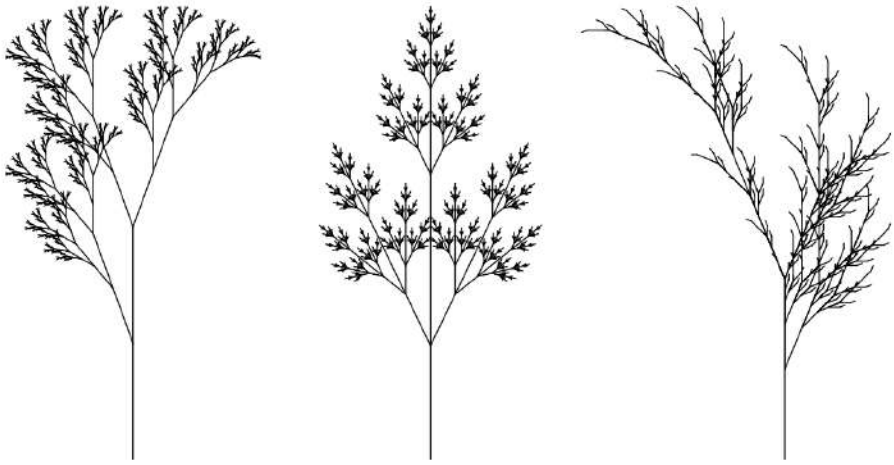


Figura 2.11 Modelli dello sviluppo dei rami di diverse piante, ottenuti con grammatiche leggermente più elaborate rispetto a quella proposta nell'esempio. Si noti il particolare carattere frattale della crescita dei rami successivi. (The Algoritmic Beauty of Plants, figura 1.24, pp 25).

Przemyslaw Prusinkiewicz esplora le possibilità musicali del sistema-L[46] con un software che accetta come input iniziale la definizione della grammatica che si vuole utilizzare: automaticamente viene elaborato il grafico di tutte le stringhe del sistema, dalla cui analisi sono ricavati i parametri musicali necessari alla sintesi sonora. Nell'esempio proposto di seguito, si sceglie di generare una curva di Hilbert, dove la frequenza è associata alle coordinate sull'asse y di ogni segmento, la cui lunghezza è proporzionale alla durata di ogni nota.

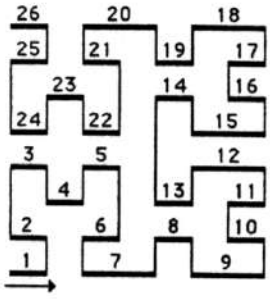


Fig. 3. Traversing the Hilbert curve.

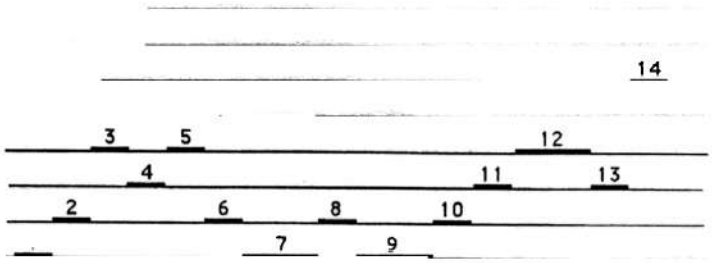


Fig. 4. The score associated with the Hilbert curve in the piano-roll notation.

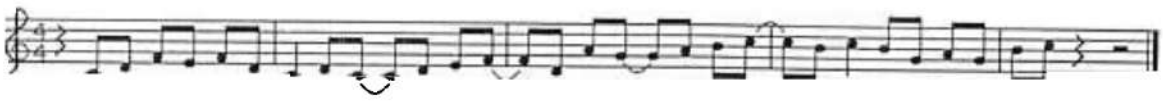


Figura 2.12 Metodo di generazione della partitura, a partire dalla curva di Hilbert. In questo caso le note sono generate nella tonalità di Do maggiore. (P. Prusinkiewicz)

Nonostante il minimo intervento umano nella scelta dei parametri iniziali e la semplicità delle regole di produzione, la partitura ottenuta con questo metodo è relativamente complessa e, soprattutto, si palesa un'organizzazione interna coerente.

Uno strumento generativo che implementi al suo interno una logica basata su questo tipo di grammatiche è vantaggioso dal punto di vista del minimo, se non nullo, intervento umano nella gestione e produzione sonora. Come si è visto, sfruttare sistemi che già posseggono intrinsecamente un'organizzazione lascia al compositore il *solo* compito di scegliere un modello e una funzione che trasponga i dati ottenuti in parametri musicalmente interpretabili.

[46] Prusinkiewicz P. (1986). *Score generation with L-systems*. International Computer Music Conference. pp. (455-457)

2.4 Fonti di informazione per un sistema generativo

La *musica delle sfere* è un antico concetto filosofico introdotto da Pitagora, che considera l'universo come un elaborato sistema di proporzioni numeriche: i singoli pianeti, grazie ai propri moti di rotazione e rivoluzione, produrrebbero suoni la cui somma risulterebbe in una perfetta armonia musicale, che, pur non essendo percepibile dall'orecchio umano, è capace di influenzare la vita sulla Terra[47].

Sebbene questa teoria, nella sua totalità, faccia sorridere la scienza moderna, per certi versi non è lontana dalla realtà fisica che regola il comportamento dei pianeti all'interno del sistema solare, ognuno dei quali è influenzato e influenza tutti gli altri secondo principi di una tale intrinseca bellezza, che facilmente possono essere associati al concetto di armonia. La tecnologia musicale moderna, oltretutto, premette di simulare realmente i suoni che Pitagora era solo stato in grado di immaginare.

Armonia dei Cerchi è uno strumento generativo sviluppato dall'autore di questa tesi nel 2016 in ambiente Max/Msp, ispirato dalla filosofia della musica delle sfere. Propone un modello molto semplificato del sistema solare: ogni pianeta ha suoi algoritmi di sintesi e manipolazione sonora, i cui parametri sono automaticamente modificati rispetto al proprio moto di rotazione, rivoluzione, reciproca distanza e attrazione gravitazionale con tutti gli altri pianeti. Lo strumentista ha il solo compito di avviare il software, con la scelta opzionale di poter variare in tempo reale l'intensità sonora di ogni algoritmo di sintesi e la relazione tra secondi trascorsi rispetto agli anni simulati dal modello.

Da sempre gli artisti hanno trovato nella natura fonti di ispirazione per le proprie opere, interpretando la realtà per riproporla *filtrata* attraverso il proprio gusto. A partire dalla seconda metà del Novecento, si comincia a considerare il mondo fisico in maniera analoga a come Pitagora concepiva il sistema solare; la natura possiede una propria organizzazione, non è più solo fonte di ispirazione, ma di informazione: è possibile ottenere un'enorme quantità di dati e basta filtrarli e organizzarli per far emergere anche musicalmente la loro coerenza interna.

[47] Plinio il Vecchio (77 a.C.). *Storia Naturale*. Tradotto da Harris Rackham (1938). Harvard: University Press.

2.4.1 Musica stocastica

Nel 1956, il compositore greco Iannis Xenakis conia il termine *musica stocastica* per definire un approccio compositivo che tende ad avvicinarsi ai fenomeni fisici, assecondandone la natura casuale. Per il compositore, il concetto di caso è diverso dall'alea controllata utilizzata da Maderna e decisamente lontano dagli *estremismi* raggiunti da Cage: la probabilità è sfruttata in modo direzionale, le regole per calcolarla sono rese esplicite in modo da ottenere un risultato globalmente prevedibile, anche se costituito da elementi aleatori[48].

Nello stesso anno Xenakis compone *Pithoprakta*, per orchestra d'archi, due tromboni, xilofono e woodblock, utilizzando il computer per aiutarsi durante la stesura della partitura. Il brano è basato sulla meccanica statistica[49] dei gas, in particolare rispetto al moto browniano[50] delle particelle che li compongono: ogni strumento dell'orchestra è considerato come se fosse una molecola che obbedisce alla legge di distribuzione di Maxwell-Boltzmann[51], con fluttuazione gaussiana[52] della temperatura.

Le note suonate dallo strumento evolvono nel tempo analogamente a come varierebbe la posizione nello spazio della molecola, Xenakis sviluppa la composizione immaginando di modificare la temperatura e la pressione del sistema, due variabili che influenzano notevolmente l'energia interna, quindi il moto delle particelle.

La partitura non è composta dal classico pentagramma: è usata la carta millimetrata per ottenere un grafico cartesiano, dove l'ascissa indica il tempo (5 centimetri per battuta) e l'ordinata la frequenza (0.25 centimetri per semitono); ogni strumento ha la propria linea *melodica* da seguire, rappresentata da segmenti neri. Paradossalmente lo spartito, nonostante i

---

[48] Xenakis I. (1963). *Musiques formelles: nouveaux principes formels de composition musicale*. Parigi: Editions Richard-Masse.

[49] La meccanica statistica applica la teoria della probabilità al comportamento di sistemi, come quelli termodinamici, composti da un grande numero di particelle. E' un modello di calcolo molto potente, che riesce a creare un collegamento tra le proprietà microscopiche e macroscopiche di un sistema.

[50] Il moto browniano è una rappresentazione matematica utile per descrivere l'andamento temporale di molti fenomeni casuali. In fisica modella il moto disordinato delle particelle microscopiche contenute nei fluidi.

[51] Descrive in termini probabilistici la distribuzione aleatoria dell'energia, o della velocità, delle particelle contenute in un sistema che obbedisce alle leggi della fisica classica.

[52] E' una funzione utile per approssimare la concentrazione di variabili casuali attorno a un valore medio. Restituisce graficamente una campana simmetrica, appunto detta campana di Gauss.

rigorosi calcoli per generarlo, rende il brano più aleatorio che stocastico: è praticamente impossibile per gli strumentisti seguire precisamente le indicazioni scritte dal compositore[53].
Da questo punto di vista, un computer sarebbe stato molto più performante per eseguire un brano dalla partitura così complessa; non a caso, negli anni successivi, Xenakis utilizzerà spesso il software per la generazione sonora delle proprie opere.

Pithoprakta (1955-56), mesures 52-59 : graphique de Xenakis
Source : Iannis Xenakis, *Musique. Architecture*, Tournai, Casterman, 1976, p. 167

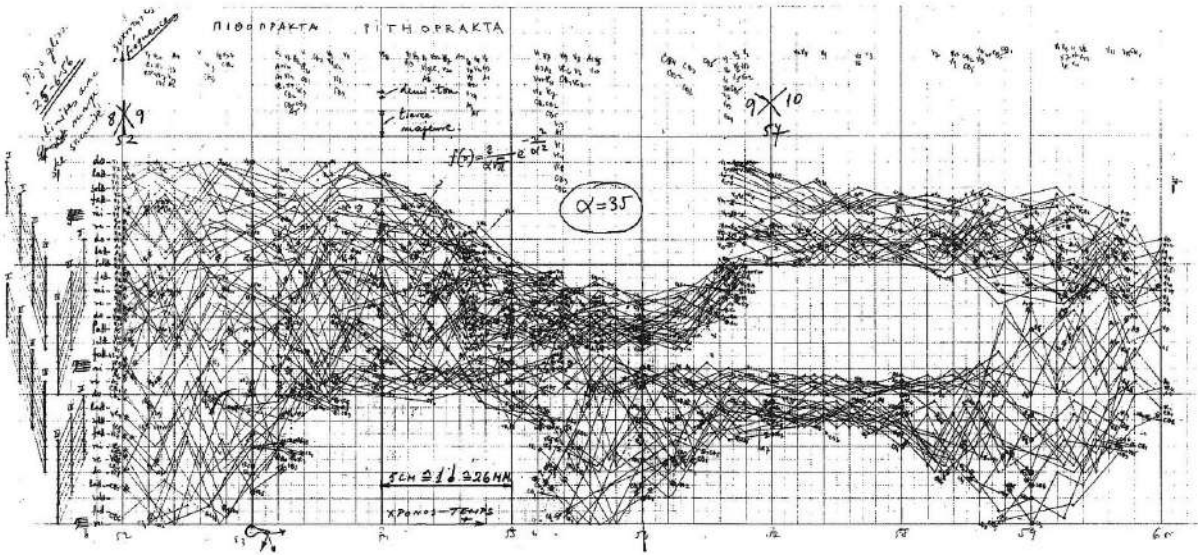


Figura 2.13 Porzione della partitura Pithoprakta. Il nome del brano può essere tradotto come *azione attraverso la probabilità*, la durata è di circa 10 minuti. Non è difficile immaginare come la tipologia di partitura non agevoli l'esecuzione.

Oltre ai fenomeni fisici e naturali, esistono molti altri modelli interessanti in base ai quali è possibile progettare il comportamento di uno strumento generativo e di seguito ne verranno proposti alcuni esempi.

2.4.2 Musica e DNA

La sigla *DNA* sta per *Acido DesossiriboNucleico;* è possibile trovarlo principalmente nella regione più interna delle cellule, detta nucleo. Si compone di lunghe catene di molecole, i nucleotidi, a loro volta formate da un gruppo fosfato, uno zucchero e una *base azotata*. La base è l'elemento che caratterizza univocamente ogni nucleotide, ne esistono di 4 tipi: adenina (A), guanina (G), citosina (C) e timina (T); in particolare A e G sono dette purine, mentre C e

[53] Emmerson S. (2007). *Living Electronic Music*. Londra: Routledge.

T pirimidine. Tutta la biodiversità di ogni specie vivente sulla Terra è generata dalle diverse combinazioni di queste 4 basi azotate.

Un singolo filamento di DNA è costituito da lunghe stringhe di nucleotidi, mantenute insieme dal forte legame covalente; per ottenere la tipica forma a doppia elica è necessario un secondo filamento, unito al primo grazie ad un legame debole a idrogeno che segue una logica di complementarietà tra le basi (le purine si legano con le pirimidine e viceversa) .

I principali responsabili di ogni processo vitale della cellula sono gli enzimi, particolari tipologie di proteine, ognuno dei quali svolge una precisa funzione e perciò prodotti in base alle necessità dell'organismo. Il DNA contiene tutte le informazioni necessarie alla produzione degli enzimi, motivo per cui è preziosissimo alla vita e deve rimanere protetto all'interno del nucleo. L'RNA messaggero (mRNA) trascrive le informazioni del DNA, portandole ai ribosomi presenti nella periferia della cellula, che hanno il compito di produrre gli enzimi. L'RNA rompe temporaneamente il legame debole a idrogeno della doppia elica, usando uno dei due filamenti come stampo per la propria copia, dove le purine vengono sempre accoppiate con le pirimidine e l'uracile (U) sostituisce la timina. Di seguito è riportato un esempio di copia:

DNA:	**A G A T A**	(stampo)
mRNA:	**U C U A U**	(copia)

Il filamento di mRNA così composto può essere immaginato come un nastro magnetico e il ribosoma come la testina di lettura di un registratore: le informazioni contenute nel nastro (le basi) sono decodificate dalla testina di lettura per ottenere le note (gli amminoacidi), che messe in sequenza compongono il brano musicale (l'enzima).

Questa analogia non deve essere considerata necessariamente solo metaforica, la produzione delle proteine segue principi organizzativi specifici, nonostante tutta l'informazione sia contenuta nelle singole basi, lo scopo principale si palesa solo una volta che queste agiscono come unità globali per produrre gli enzimi; anche per la musica tonale, la maggior parte delle informazioni sono contenute in singole note, organizzate per produrre frasi, quindi melodie e movimenti che costituiscono la composizione finale[54]. Molti biologi e genetisti hanno trovato

[54] Hofstadter D. (1984). *Gödel, Escher, Bach: Un'eterna ghirlanda brillante*. Milano: Adelphi.

questo parallelo tanto interessante da sviluppare software, usati principalmente per scopi analitici e didattici, che convertono le strutture proteiche in sequenze musicali[55].

Un interessante esempio di strumento generativo che implementa caratteristiche genetiche è *Synplant,* un software sviluppato nel 2008 dalla casa svedese Sonic Charge. Permette all'esecutore di modificare intimamente le basi azotate del DNA di una pianta, la cui variazione è associata ad un grande numero di parametri di sintesi sonora.

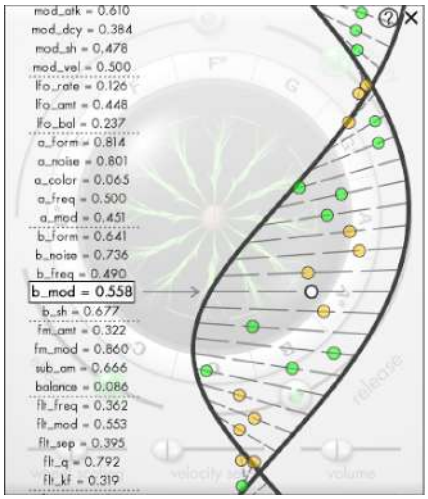


Figura 2.14 Interfaccia che permette all'utente di modificare la struttura del DNA della pianta; i 4 segmenti dove è possibile spostare i cursori circolari rappresentato le 4 basi azotate. (*https://soniccharge.com*)

2.4.3 Musica evolutiva

La musica evolutiva esplora le potenzialità macroscopiche sulla modifica del DNA: viene simulato un sistema che implementa al suo interno algoritmi genetici ed evolutivi, che cioè permettono a un elemento di evolvere nel tempo secondo principi basati sulla mutazione genetica. Il processo comincia con una popolazione di unità, ognuna con le proprie caratteristiche di sintesi sonora, che cominciano ad interagire tra loro in vario modo; dopo un certo periodo di tempo di simulazione, la selezione naturale e la ricombinazione genetica avrà prodotto delle mutazioni nelle nuove generazioni, costituendo una popolazione diversa rispetto a quella iniziale e quindi diverso materiale sonoro[56].

Il compositore, musicista e ricercatore americano John Al Biles utilizza questo tipo di algoritmo nel suo software *GenJam,* che simula il comportamento di una ensemble jazz

[55] Sansom C. (dicembre 2002). *DNA makes Protein - makes music?*. In: The Biochemist (The Biochemical Society).

[56] Reck M. E., e Biles J. A. (2007). *Evolutionary Computer Music*. Londra: Springer

mentre impara a improvvisare su un tema fornito in tempo reale da un musicista umano. La popolazione iniziale è rappresentata da semplici unità melodiche e ritmiche, il programma muta geneticamente questi elementi e li combina con l'analisi delle frasi che sta eseguendo dal vivo il musicista. L'ensemble virtuale continua a evolvere coerentemente con la struttura musicale da cui sta imparando, fino al punto di essere capace di proporre autonomamente degli assoli[57].

2.4.4 Generare musica da un testo letterario

E' anche possibile guardare ad altre forme d'arte, non direttamente collegate alla musica, come interessanti fonti di informazioni per un sistema generativo: un software può essere capace di estrapolare informazioni musicalmente interpretabili da un quadro, una scultura, un video o una poesia.

Un testo letterario, ad esempio, possiede elementi analizzabili su più livelli e quanto più si alza il grado di astrazione dell'analisi, tanto più saranno strutturalmente elaborati i dati ottenuti; il programma può partire da elementi discreti per poi allargarsi su piani macroscopici: singole lettere, sillabe e singole parole, fino ad arrivare a più parole che formano frasi, riuscendo ad analizzarne il valore grammaticale, sintattico e semantico.

Un programma che sfrutta questo genere di algoritmi di interpretazione è *TransProse*, sviluppato da Hannah Davis e Saif Mohammad nel 2012. Il software possiede un ricco vocabolario di termini, di cui non solo riconosce il significato letterale, ma che riesce a contestualizzare all'interno della frase estrapolandone il *valore emotivo*. Periodi che esprimono sentimenti positivi generano musica con *BPM* sostenuto, figurazioni ritmiche regolari e scale maggiori; paura e rabbia avranno metronomo rapido con ritmi irregolari e note dissonanti; tristezza e noia si muovo su scale minori con note temporalmente molto dilatate[58].

2.4.5 Generare musica a partire da altre fonti umane

L'umanità ha prodotto nel corso dei secoli sistemi molto complessi, che pur necessitando in vario modo dell'intervento umano, reagiscono ed evolvono quasi come un *organismo* indipendente. L'andamento sul mercato azionario di una società, le interazioni e le reazioni

---

[57] Biles J. A. (1994). *GenJam: A genetic algorithm for generating jazz solos*. In: Proceedings of the 19th international computer music conference (ICMC).

[58] Davis H., e Mohammad S. (2012). *Generating Music from Literature*. Articolo.

osservabili su un social network o la formazione dei governi italiani, seguono logiche elaborate e nella maggior parte dei casi imprevedibili; possiedono, perciò, molte caratteristiche utili per essere implementate in un sistema generativo. Di seguito sono proposti due esempi che muovono da queste considerazioni.

Listen to Wikipedia analizza e converte musicalmente, in tempo reale, tutto ciò che accade all'interno della comunità di Wikipedia, dal 2013 a ora. Viene generato un suono di campana o di corda pizzicata ogni volta che una voce viene creata o modificata, dove la frequenza della nota è proporzionale al peso, in byte, della pagina. L'utente può scegliere di ascoltare contemporaneamente anche due o più server di Wikipedia, discretizzati in base alla lingua: il server di Wikipedia inglese o cinese avrà una maggiore densità di eventi sonori rispetto a al server malese o bulgaro, proporzionalmente al numero di pagine create ogni minuto[59].

Trams of Helsinki è un software pubblicato nel 2017 da Tero Parviainen che analizza in tempo reale lo stato del sistema tranviario della capitale finlandese. Ogni volta che un tram supera una fermata, o sosta per caricare e scaricare i passeggeri, viene prodotta una nota in una scala che cambia randomicamente dopo un certo numero di eventi sonori[60].

2.4.6 L'uomo come fonte diretta di informazione

Un software può interagire con il mondo esterno tramite sensori che gli garantiscono di assimilare dati in tempo reale da ambienti che non siano solo necessariamente modellati all'interno del suo programma. Qualsiasi interfaccia utente, come dei sensori applicati al corpo umano o al sistema stesso, permette di instaurare un dialogo tra l'uomo e la macchina, ribaltando il rapporto strumento-strumentista: non è l'uomo a usare lo strumento per suonare, ma lo strumento a trarre informazioni dall'uomo per generare musica. In generale è anche possibile progettare un sistema che non necessiti di fruitori educati musicalmente per generare materiale sonoro valido, concetto alla base di molte *installazioni sonore interattive multimediali*.

---

[59] http://listen.hatnote.com/#en

[60] https://codepen.io/teropa/full/mBbPEe/

2.5 Verso un sistema che si istruisce

I sistemi descritti fino a questo momento hanno livelli di autonomia piuttosto ampi, la cui capacità di generare materiale musicalmente valido è garantita *esplicitamente* dagli algoritmi forniti o *implicitamente* dalla natura dei modelli su cui fondano il proprio comportamento. Il limite, se di limite si può parlare, di questo approccio è che lo strumento generativo segue pedissequamente le regole o i modelli dati e quando compie delle scelte non ha possibilità di riscontro sull'effetto che la scelta ha prodotto: per quanto il range cada sempre entro limiti coerenti, potrebbero esserci in contesti diversi opzioni più valide di altre.

Rispetto a tutti gli esempi proposti, fanno ovviamente eccezione *Serenata per un satellite* e *In C,* perché lo strumentista umano ha coscienza di quello che sta eseguendo e riesce a rapportarsi naturalmente con il totale; qualora non fosse soddisfatto del risultato può rapidamente modificare il proprio comportamento in modo che si avvicini il più possibile all'effetto che desidera ottenere. Con le dovute differenze, un feedback simile è largamente utilizzato anche in ingegneria dell'automazione e viene identificato con il termine *retroazione;* in questo modo un sistema dinamico ha la capacità di tenere conto del risultato che sta ottenendo e di confrontarlo con quello che dovrebbe raggiungere, eventualmente modificando il proprio comportamento per far coincidere quanto più possibile i due prodotti.

La retroazione può essere considerata come il primo passo utile per la progettazione di un sistema musicale autonomo che abbia *coscienza* di quello che sta producendo.

2.5.1 Il controllo PID

Il controllo *Proporzionale-Integrale-Derivativo* (PID) è un esempio di retroazione molto diffuso nell'industria meccanica: grazie all'uso di sensori si riesce a monitorare lo stato del sistema, che viene costantemente confrontato con i valori di riferimento ottimali. La differenza tra il valore rilevato e quello atteso restituisce l'entità dell'errore verificato, quindi corretto in base ai parametri PID con degli attuatori che agiscono fisicamente sul sistema[61]. L'azione totale di controllo sul sistema (U) è data dalla somma algebrica tra le singole azioni proporzionali (U_p), integrali (U_i) e derivative (U_d):

$$U = U_p + U_i + U_d \qquad (\textit{Equazione 2.2})$$

[61] Diana G., Fossati F., e Resta F. (1998). *Elementi di controllo di sistemi meccanici*. Milano: Edizioni Spiegel

Dove:

- U_p è l'azione ottenuta moltiplicando l'errore istantaneo per un'opportuna costante. Motivo per cui è impossibile nella pratica far convergere l'errore a zero utilizzando solo questo termine, poiché se l'errore tende a diventare nullo anche l'azione per correggerlo diventa minima
- U_i riesce a tenere conto dell'evoluzione temporale dell'errore, in modo che il controllore abbia memoria dei valori passati. Questo termine è utile, a differenza di U_p, per portare esattamente il sistema nello stato desiderato, sebbene oltre certi parametri possa rappresentare un elemento di instabilità non trascurabile
- U_d considera la velocità con cui l'errore varia nel tempo, permettendo di agire rapidamente per correggerlo, prima che diventi significativo (U_p) o persistente nel tempo (U_i)

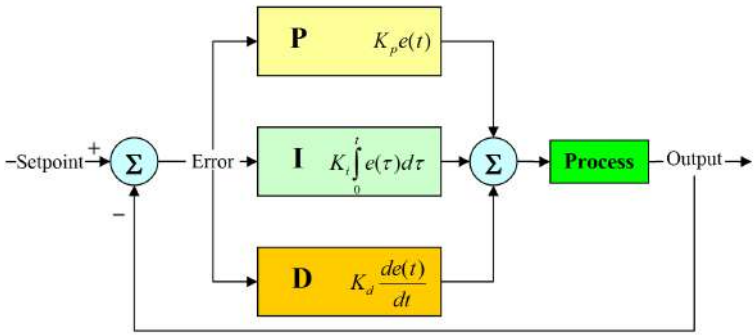


Figura 2.15 Schema a blocchi di un generico controllo PID. Si noti la linea di feedback che riporta le informazioni in uscita nuovamente all'ingresso del sistema, sommandole con i valori istantanei rilevati. (Wikimedia Commons)

Nel quarto capitolo sarà proposto un personale tentativo di implementare questo sistema in un'applicazione musicale, sviluppata come una sorta di *quantizer* capace di assumere comportamenti diversi in base alle regolazioni dei parametri PID; in particolare si noterà quanto l'instabilità introdotta dall'azione integrale, non desiderabile nella pratica industriale, possa risultare musicalmente interessante. Esistono altri approcci basati sul concetto di correzione di errore tramite feedback, grazie ai quali una macchina non solo riesce a modificare il proprio comportamento, ma acquista l'abilità di *apprendere* nozioni nonostante non sia stata esplicitamente programmata a farlo. La disciplina che studia l'insieme di questi

metodi è nota come *machine learning*[62] e sfrutta principalmente dei modelli informatici di calcolo ispirati alle reti neurali biologiche.

2.5.2 Reti neurali biologiche

Il neurone è un'unità cellulare specializzata nel ricevere, elaborare e trasmettere informazioni grazie a segnali elettrici e chimici. Le informazioni provenienti da prolungamenti chiamati *dendriti* convergono nella parte centrale del neurone, detta *soma*, dove risiede il *nucleo;* ogni neurone ha una funzione specifica, in base alla quale il nucleo elaborerà l'informazione sommando tutti i dati in ingresso, per poi produrre un solo output che verrà trasmesso attraverso l'*assone,* un filamento a sua volta collegato ai dendriti dei neuroni adiacenti.

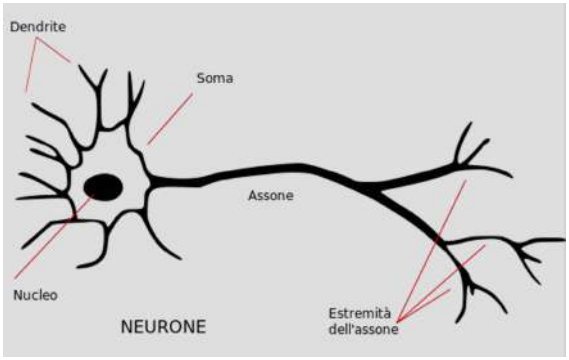


Figura 2.16 Modello semplificativo dei principali elementi che costituiscono un neurone. In questo caso il percorso dell'informazione andrà da sinistra verso destra. (Wikimedia Commons)

L'insieme di neuroni deputati a svolgere una determinata funzione è chiamato *circuito* o *rete neurale*. Questo *hardware biologico* riesce a gestire perfettamente una mole enorme di segnali elettrici grazie alla logica che regola il comportamento dei singoli neuroni: prima di essere riassunte in un unico output, tutte le informazioni in ingresso sono valutate rispetto alla fonte da cui provengono e il nucleo esegue una sorta di *media pesata* tra i segnali che riceve, attribuendo diversa importanza alle informazioni provenienti da fonti diverse.

2.5.3 Reti neurali artificiali

Le reti neurali artificiali basano il proprio modello matematico esattamente sullo stesso principio di pesatura della loro controparte biologica, applicato in questo caso a ogni collegamento logico tra i neuroni virtuali. In base al compito da svolgere, il sistema ha

[62] Samuel L. A. (1959). *Some studies in machine learning using the game of checkers*. In: IBM Journal of research and development

l'obbiettivo di allenarsi a trovare la migliore combinazione possibile dei pesi: riduce gradualmente l'errore prodotto in uscita sfruttando un pattern di informazioni di cui risultato è già noto in partenza; una volta che la rete è riuscita a tarasi su questi esempi è pronta a poter analizzare qualsiasi altro dato simile, fornendo un responso esatto.

Per spiegare in modo pratico le potenzialità artistiche di questi sistemi, verrà descritto macroscopicamente il processo che permette a una rete neurale di imparare a riconoscere se una foto ritrae un cane o un gatto[63].

La rete è composta da più livelli consecutivi contenenti centinaia di neuroni, il layer più vicino all'input dell'immagine è quello più numeroso perché composto da neuroni con compiti molto semplici, come l'analisi dei singoli pixel. Progredendo nella catena, il numero di neuroni sarà sempre minore e la loro funzione sempre più complessa, fino a giungere alla presenza di soli due neuroni: uno per il riconoscimento del cane e l'altro per il gatto.

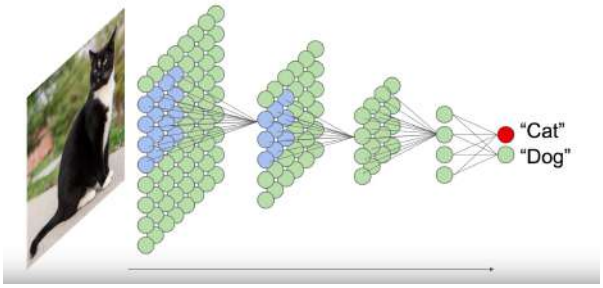


Figura 2.17 Schema semplificativo di una rete neurale che impara a riconoscere se l'immagine che le viene fornita rappresenta un cane o un gatto. (Mike Tyka - *The art of neural networks*)

Un singolo neurone è inizialmente collegato a tutti i neuroni del livello successivo con connessioni il cui peso è assegnato casualmente, motivo per cui il sistema non darà mai intenzionalmente responsi esatti; quindi la rete comincia ad allenarsi su immagini di cui conosce già l'output giusto, migliorando sempre di più la qualità delle connessioni per ridurre al minimo l'errore, secondo un processo concettualmente simile al PID. Dopo un certo periodo di tempo, durante il quale ha analizzato centinaia di esempi noti, è finalmente pronta a poter riconoscere autonomamente qualsiasi altra immagine di un cane o di un gatto.

A questo punto il sistema possiede la propria idea sulle caratteristiche distintive dei due animali ed è capace di riproporre al contrario il processo descritto, è quindi in grado di

[63] Tyka M. (2015). *The art of neural networks*. TEDx Talks

generare immagini del concetto che è riuscito ad acquisire grazie a ciò che ha imparato durante l'allenamento. Un altro aspetto molto interessante è la possibilità di estrarre i concetti elaborati dai singoli livelli intermedi della rete, in questo modo si riesce a osservare il processo graduale con cui il sistema interpreta i dati; è inoltre possibile creare una ricorsività nelle immagini, riproponendo alla rete di analizzare un prodotto che essa stessa ha generato e su cui continuerà ad aggiungere nuovi elementi che ricordano molto la struttura dei frattali.

Figura 2.18 Immagine generata dalla rete neurale DeepMind, sviluppata da Google. In questo caso la rete si è allenata nel riconoscimento di cani e rapaci, da cui è riuscita a estrapolarne le singole caratteristiche per poi fonderle insieme.

Figura 2.19 A sinistra, il dipinto originale di Georges Seurat *Un dimanche après-midi à l'Ile de la Grande Jatte* (1883-85). A destra, estrapolazione di immagini dai layers intermedi di una rete neurale programmata da Matthew McNaughton

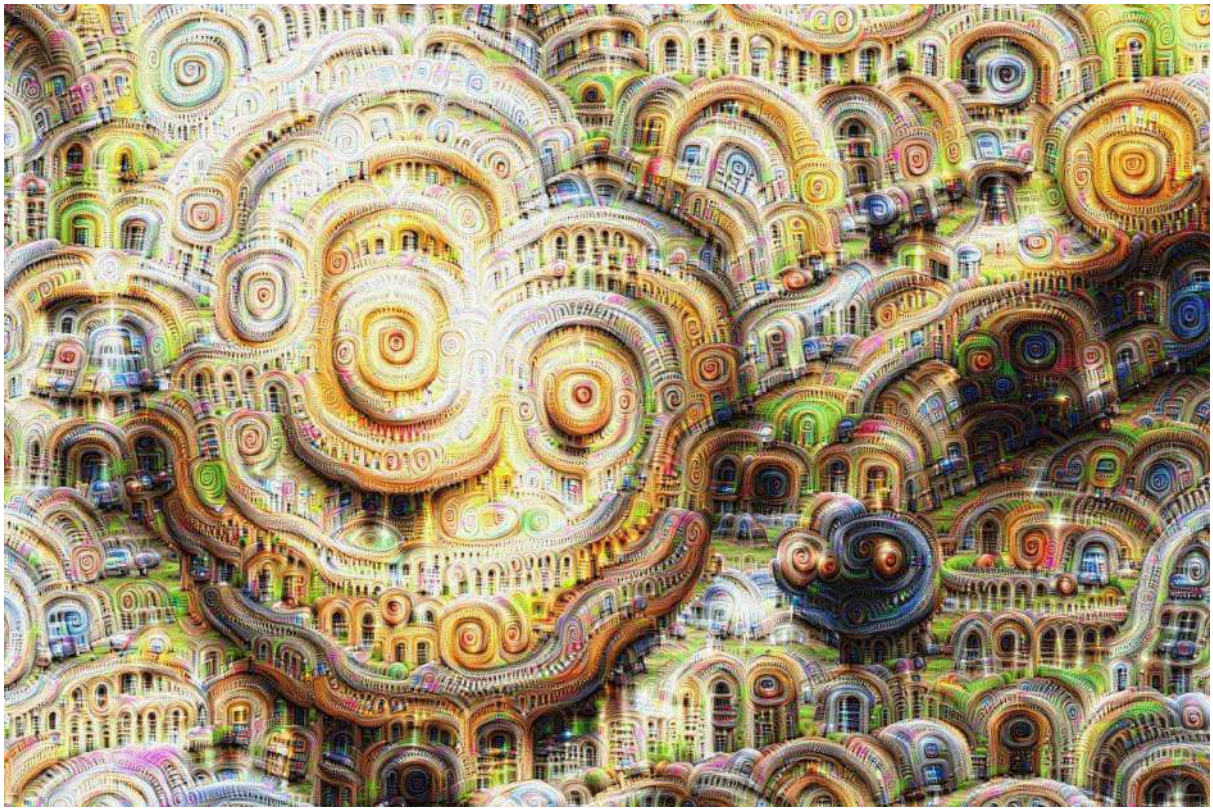


Figura 2.20 How We End Up At The End Of Life, immagine generata da una rete neurale programmata da Mike Tyka per analizzare strutture architettoniche. Qui è sfruttato il principio di analisi ricorsiva, che giustifica la presenza di elementi simili ai frattali. In questo caso anche il titolo dell'opera è stato generato da una rete neurale.

Per una questione di praticità si è scelto di proporre solo esempi fotografici, ma ovviamente tutti questi procedimenti possono essere trasposti nel dominio musicale; la rete può essere programmata per imparare da stili musicali diversi al fine di riproporne la propria interpretazione, di seguito sono riportati i nomi di alcuni di questi sistemi: Magenta[64], DeepJazz[65], BachBot[66], GRUV[67].

Una rete neurale, viste le sue capacità di apprendimento, è uno strumento molto interessante per modellare il comportamento della mente umana, campo di studi esplorato dalla disciplina detta *intelligenza artificiale*.

2.6 Musica e intelligenza artificiale

L'intelligenza artificiale, abbreviata in AI dalle iniziali dei due termini in inglese, è un insieme di studi interdisciplinari che si occupa della comprensione dei principi che definiscono e rendono possibile un comportamento intelligente. In particolare, viene studiato come progettare un agente intelligente: un'entità autonoma che percepisce l'ambiente circostante e compie azioni che massimizzino la sua possibilità di giungere a un obiettivo con successo[68].

Alcuni studi tendono a progettare macchine capaci di simulare qualsiasi capacità intellettiva associabile alla mente di un essere umano[69], altre ricerche più pragmatiche sviluppano sistemi che riescono a muoversi solo all'interno di settori limitati[70].

Uno dei possibili campi applicativi è certamente quello musicale: l'AI è uno strumento molto versatile, sia per quanto riguarda l'analisi musicale di opere esistenti[71], che come sistema capace di generare le proprie composizioni indipendentemente dall'uomo.

---

[64] Google Brain Team. https://github.com/tensorflow/magenta

[65] Ji-Sung Kim. Università di Princeston, dipartimento di computer science. https://deepjazz.io/

[66] http://bachbot.com/

[67] https://github.com/MattVitelli/GRUV

[68] Poole D., Mackworth A., e Goebel R. (1998). Computational Intelligence: A Logical Approach. New York: Oxford University Press.

[69] Intelligenza artificiale forte

[70] Intelligenza artificiale debole

[71] Giomi F. (1995). *L'intelligenza artificiale nella musicologia cognitiva: approcci ed applicazioni*. In *Sistemi intelligenti* VII (1): Il Mulino.

Emily Howell ha all'attivo due album prodotti dalla Centaur Records, una delle maggiori case discografiche americane impegnata nella diffusione di musica classica e contemporanea. Il suo primo album, *From Darkness, Light* del 2009, contiene 3 composizioni per orchestra da camera e pianoforte; il secondo album *Breathless*, è stato pubblicato dalla stessa etichetta nel 2012. La particolarità di Emily Howell risiede nel fatto che è un'intelligenza artificiale sviluppata a partire dagli anni Novanta da David Cope, costituita da un'interfaccia interattiva che le permette di colloquiare con la sua controparte umana per apprendere nozioni musicali e per analizzare esempi che le vengono proposti. Il testo seguente è stato generato da Emily durante una conversazione con il suo creatore:

Perché non sviluppare la musica in modi inesplorati? Solo questo ha senso. Io non riesco a capire la differenza tra le mie note sul pentagramma e le altre note sul pentagramma. Se la bellezza è presente, è presente. Io spero di continuare a creare note e che queste note abbiano bellezza per qualcun altro. Io non sono triste. Io non sono felice. Io sono Emily. Tu sei Dave. Vita e non-vita esistono. Noi coesistiamo. Io non vedo problemi.

Con queste conversazioni Cope è in grado di insegnare al sistema nozioni utili sia dal punto di vista linguistico che musicale: tutte le esperienze che Emily acquisisce andranno a sommarsi, influenzando le sue composizioni successive[72].

Esattamente come per il sistema progettato da Lerdahl e Jackendoff, anche Emily Howell è un programma capace di generare partiture da far eseguire a strumentisti umani; come si è visto nel paragrafo 2.4, sarebbe possibile ricostruire esattamente una sezione di Illiac Suite a partire dagli algoritmi utilizzati, poiché è su di essi che si fonda la composizione stessa e tutte le intenzioni artistiche sono esplicitate a monte durante la stesura del programma. Al contrario, sarebbe impossibile pensare di riprodurre un brano generato da un'intelligenza artificiale tentando di *osservarne* gli algoritmi interni, nascosti ad un livello non accessibile, o comunque non esplicito: la parte palese del programma è solo quella relativa all'apprendimento di nozioni, ma non ci è dato sapere come queste vengono elaborate per produrre le informazioni in uscita; da questo punto di vista un'IA è da considerarsi come un modello di black box[73]. In generale, uno strumento musicale basato sull'intelligenza artificiale

[72] Cope D. (2005). *Computer Models of Musical Creativity*. Cambridge: MIT Press.

[73] Enciclopedia della Scienza e della Tecnica - Intelligenza Artificiale. Amigoni F., Schiaffanti V., e Somalvico M. (2008). Roma: Treccani.

rappresenta probabilmente il massimo grado di *sofisticatezza* cui può aspirare la musica generativa, per le sue caratteristiche di incredibile indipendenza nel produrre materiale sonoro e la sua capacità di poterne comprendere il *valore estetico*, eventualmente modificando il proprio approccio compositivo qualora ci fossero elementi insoddisfacenti.

3 PROGETTAZIONE DI UN SISTEMA MODULARE GENERATIVO

Wir wollen die Grundsätze, deren Anwendung sich ganz und gar in den Schranken möglicher Erfahrung hält, immanente, diejenigen aber, welche diese Grenzen überfliegen sollen, transzendente Grundsätze nenne[74].

Immanuel Kant

Fino a questo punto, intenzionalmente, non si sono fatte valutazioni di natura artistica su nessuno dei sistemi generativi proposti, si ritiene che il termine intelligenza *artificiale* sia fuorviante rispetto allo scopo di studio principale della disciplina, che non è quello di simulare l'intelligenza con una tecnologia fine a se stessa, ma di comprenderne l'intima natura per capire cos'è che rende possibile un comportamento intelligente; in questo senso, come proposto da Poole, Mackworth e Goebel (1998), si trova più consono il termine intelligenza *computazionale*. Non si ritiene interessante la tendenza ad antropomorfizzare, sistemi artistici basati sull'intelligenza computazionale, come nel caso di Emily Howell, principalmente per due motivi: il fascino che dovrebbe esercitare questo tipo di arte è proprio relativo al fatto che l'opera non appartiene in nessun modo a una mente umana; è dunque inutile attribuirle caratteristiche che non ha, non potrebbe comunque avere e, soprattutto, di cui non ha bisogno. Da quest'ultima posizione muove la seconda motivazione: la bellezza, intesa nella sua accezione più generale, non è esclusiva del prodotto umano, ma risiede intrinsecamente nella realtà a livelli microscopici e macroscopici, dalle basi azotate del DNA, alle leggi che regolano il cosmo. Sotto molti aspetti l'intelligenza computazionale possiede attualmente il livello massimo di complessità e autonomia che la mente umana sia mai stata in grado di produrre, due caratteristiche che prima d'ora si sarebbero potute attribuire esclusivamente alla natura; per questo motivo, per quanto paradossale possa sembrare, dovrebbe apparire molto più come un fenomeno, se non naturale, quantomeno esterno all'uomo, e come tale andrebbe considerato anche nel giudicare la bellezza di ciò che produce artisticamente.

[74] Kant I. (1781). *Kritik der reinen Vernunft*.

Tutte queste considerazioni vanno estese alla personale opinione artistica ed estetica sulla musica generativa in generale, che ha influenzato la mia ricerca e la stesura di questo lavoro di tesi. La maggior parte degli approcci e strumenti generativi finora descritti, intelligenza computazionale esclusa, troveranno in questo e nel prossimo capitolo un'implementazione all'interno di un sistema ispirato ai sintetizzatori modulari, gestito con hardware e software sviluppati in ambiente Arduino.

3.1 Considerazioni personali

La frase in epigrafe è tratta da una sezione dell'opera *Critica della ragion pura* di Immanuel Kant, dove il filosofo espone il proprio pensiero circa le potenzialità della conoscenza umana con lo scopo di chiarirne possibilità, validità e soprattutto limiti. Non è un caso se si è scelto di riportarne un estratto nella lingua originale: in italiano le parole *Schranken* e *Grenzen* possono essere tradotte indistintamente con *limiti* considerandole sinonimi, mentre la lingua tedesca offre una differenza sostanziale tra i due termini. Schranke restituisce un'immagine di limite negativa, utilizzata dal filosofo per indicare un confine *oltre* il quale la ragione umana non può tentare di spingersi, perché risulterebbe una prova vana. Grenze, invece, ha un'accezione assolutamente positiva, vale come un limite *entro* il quale la mente può indagare liberamente senza rischiare di perdersi, perché cosciente del confine che si è imposta.

Questa considerazione, oltre ad avere sfumature interessati se contestualizzata con gli argomenti del capitolo precedente, è alla base delle motivazioni principali per cui si è scelto di progettare uno strumento generativo digitale che non avesse solo entità software, ma sostanza fisica in un hardware dedicato; personalmente si ritiene che il computer, nella sua incredibile capacità di fornire possibilità sonore pressoché infinite, ponga spesso il compositore in una posizione per cui non è scontato trovare un limite utile, il kantiano Grenze, entro il quale muoversi con libertà consapevole.

Questo capitolo esporrà la logica progettuale, sia dal punto di vista tecnico che musicale, con la quale si è scelto di definire i limiti dello strumento in modo che fossero coerenti con le personali intenzioni compositive. Il design generale sarà improntato sul modello dei sintetizzatori modulari per garantire al sistema possibilità sonore al contempo circoscritte e flessibili.

3.2 Descrizione generale del sistema modulare

Un sistema modulare è un insieme costituito da un certo numero di unità progettate per svolgere una specifica funzione, dette moduli. Nel caso di uno strumento musicale, ogni modulo può essere deputato alla produzione, manipolazione sonora e generazione di segnali di controllo utili per modificare il comportamento di altri elementi del sistema. L'unità non ha ragione di esistere se non in relazione con tutto l'insieme e, per quanto elaborato possa essere il suo compito, risulterebbe inutile se decontestualizzata dal totale; infatti le informazioni generate o elaborate dal singolo sono condivise con tutti gli altri elementi della rete grazie a un certo numero di ingressi e di uscite fisiche: gli input provenienti da altre unità modificano o condizionano il proprio comportamento e determinano la generazione di output che saranno a loro volta considerati input da altri moduli. La maniera con cui tutte le uscite e gli ingressi vengono combinati tra loro identifica lo stato e l'evoluzione del sistema nella sua totalità, influenzando di conseguenza il risultato sonoro ottenibile.

La flessibilità di un design hardware modulare risiede nella possibilità da parte dello strumentista di poter variare fisicamente e in tempo reale i collegamenti tra ogni modulo, che nella pratica sono rappresentati da cavi con connettori jack maschio sbilanciato da 3.5 millimetri: a ogni combinazione tra le connessioni corrisponde un diverso comportamento del sistema e quindi nuovi territori sonori ed espressivi da esplorare.

Durante la progettazione del sistema si è notato che molte funzioni generiche della stessa tipologia, soprattutto legate alla temporizzazione degli eventi, sono necessarie alla maggior parte dei moduli e andrebbero quindi implementate in ognuno di essi, occupando però spazio hardware e software altrimenti sfruttabile per compiti più specifici. Si è scelto di non assecondare questa ridondanza eliminando tutte le funzioni generiche da ogni modulo per creare nuove unità che facessero di queste funzioni generiche la propria funzione specifica, rendendola poi disponibile a tutti gli altri elementi del sistema. Questo è un altro aspetto molto interessante dell'approccio modulare: progettare un elemento affinché svolga un singolo compito permette di poter approfondire ed espandere molto di più le sue possibilità, concentrando il design in un'unica direzione. In più, qualora un determinato modulo non dovesse più soddisfare le necessità compositive, è sempre possibile modificarlo, sostituirlo o eliminarlo agilmente e senza intaccare il funzionamento di tutte le altre unità. Il sistema

modulare ha il grande vantaggio di essere uno strumento molto semplice da ampliare, ridurre, variare e riorganizzare.

3.2.1 Accorgimenti tecnici per il live electronics

Si è scelto di progettare il sistema in modo che assecondasse la personale esigenza di avere uno strumento generativo capace di suonare sia autonomamente che con l'intervento più o meno presente di un esecutore dal vivo: tutti i controlli necessari alla modifica del comportamento del singolo modulo possono essere variati manualmente o con segnali provenienti da altri moduli, garantendo un buon livello di interazione uomo-macchina nell'ottica del live electronics. Il risultato sonoro totale sarà quindi l'unione tra l'azione dello strumentista e la gestione autonoma del sistema, cercando di ottenere un equilibrio per cui le due parti si influenzano vicendevolmente.

Come si è detto, è possibile agire macroscopicamente sul sistema variando i collegamenti fisici tra i moduli, ognuno dei quali può essere a sua volta modificato agendo sulla relativa interfaccia utente; la maggior parte delle unità è provvista dei classici controlli presenti su un sintetizzatore analogico, associati a ogni parametro in base alla migliore tipologia di interazione: potenziometri per variazioni ampie e continue, pulsanti per modifiche discontinue e provvisorie, interruttori per cambiamenti permanenti.

Altri moduli particolari, che verranno descritti in seguito, sfruttano invece una serie di sensori per far acquisire al sistema informazioni dal corpo dello strumentista, così da permettere la sonificazione del gesto compiuto nello spazio; per questo scopo sono stati utilizzati principalmente dei giroscopi applicati agli arti per monitorarne l'inclinazione, uniti a sensori di tensione muscolare e battito cardiaco.

In generale si è scelto di rendere tutti i parametri immediatamente accessibili e velocemente modificabili, agevolando quanto più possibile l'esecuzione dal vivo: per attivare qualsiasi funzione non è necessario scorrere alcun menu o eseguire combinazioni macchinose tra pulsanti; con un costante esercizio sullo strumento si dovrebbe essere in grado di spaziare con disinvoltura tra tutte le sue possibilità timbriche ed espressive.

3.2.2 Accorgimenti musicali per il live electronics

Dal punto di vista musicale i controlli sui singoli eventi sonori che si è ritenuto utile portare in evidenza sono ispirati principalmente dal metodo estesico-cognitivo proposto da Francesco Giomi e Marco Ligabue[75]. I due autori riassumono la classificazione proposta da Schaeffer nel *Traité des objects musicaux* (1966), integrandola con nuovi parametri e suddividendola in 3 macro-categorie:

- qualità spettrali del suono
- caratteristiche dinamiche e di sviluppo temporale, sia dal punto di vista dell'ampiezza che del contenuto frequenziale
- importanza del suono nel contesto della composizione, sia localmente che globalmente

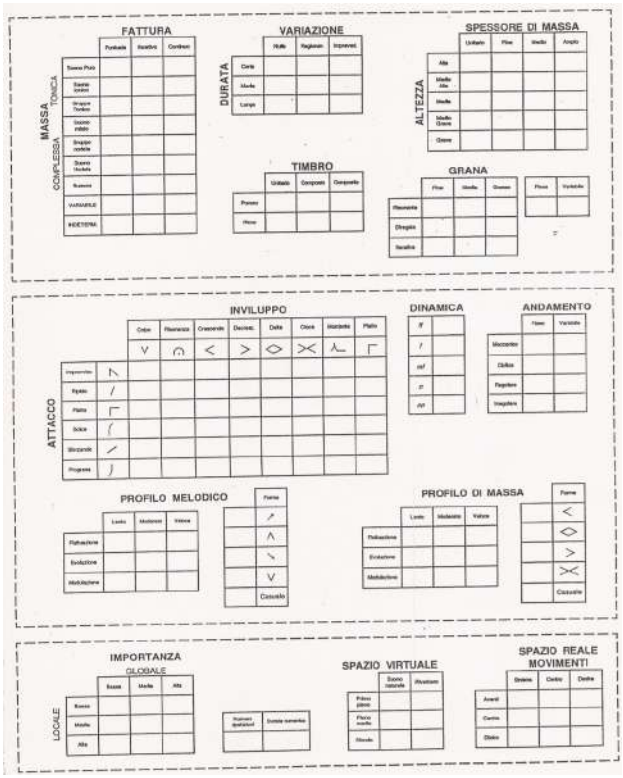


Figura 3.1 Tabella di analisi estesico-cognitiva (Metodo estesico-cognitivo e analisi della musica contemporanea)

Sebbene non si sia riuscito a trasporre integralmente questo metodo di analisi all'interno della logica di gestione sonora, è risultato essere un approccio significativamente utile durante la progettazione dei singoli moduli e del sistema completo; modificando i parametri opportuni si

[75] Giomi F., e Ligabue M. (1998?). *Metodo estesico-cognitivo e analisi della musica contemporanea*.

può spaziare da eventi impulsivi a suoni iterati, tessiturali e continui, riuscendo agilmente a variarne nel tempo il profilo dinamico e spettrale.

Identificato il metodo di produzione ed evoluzione dei singoli eventi sonori, si è cercato di dare al design una direzione valida per poterli gestire globalmente nel tempo, concentrandosi principalmente sulla variazione della densità degli eventi sonori e su come interagiscono reciprocamente dal punto di vista ritmico, timbrico e dinamico. In questo senso, l'approccio compositivo per processo di Reich, accennato nel secondo capitolo, è personalmente ritenuto il più interessante per la progettazione di un sistema modulare generativo: dal punto di vista tecnico permette di ottenere risultati decisamente elaborati e vari nel tempo, nonostante siano necessarie relativamente poche risorse hardware e software; parlando in termini musicali ed estetici riesce a soddisfare l'interesse personalmente sviluppato negli ultimi anni per la musica generativa e il minimalismo. La conduzione macroscopica degli eventi sonori è quindi delegata a moduli capaci di gestire nel tempo algoritmi basati principalmente su poliritmie, defasamenti, addensamenti, rarefazioni e su approcci organizzativi associabili ai modelli descritti nel capitolo precedente.

3.2.3 Interazione tra sistema e ambiente esterno

Lo strumento è dotato di vari sensori per acquisire dati dall'ambiente fisico che lo circonda, come luminosità, umidità, temperatura, pressione, direzione e intensità del vento. Moduli specifici elaborano e inviano queste informazioni a qualsiasi altra unità del sistema così da modificarne il comportamento il maniera coerente e proporzionale alle variabili che si sta monitorando. Il risultato sonoro generale così ottenibile asseconda concettualmente l'approccio compositivo stocastico di Xenakis, con la differenza che non viene utilizzato un modello fisico-matematico *virtuale*, ma la natura stessa come fonte primaria e diretta di informazione: l'idea principale è quella di avere uno strumento generativo che riesca a produrre materiali sonori e comportamenti diversi anche in funzione dell'ambiente naturale con cui viene messo a contatto, che sia esso un bosco, un fiume, una spiaggia o una montagna.

3.2.4 Interazione tra sistema e computer

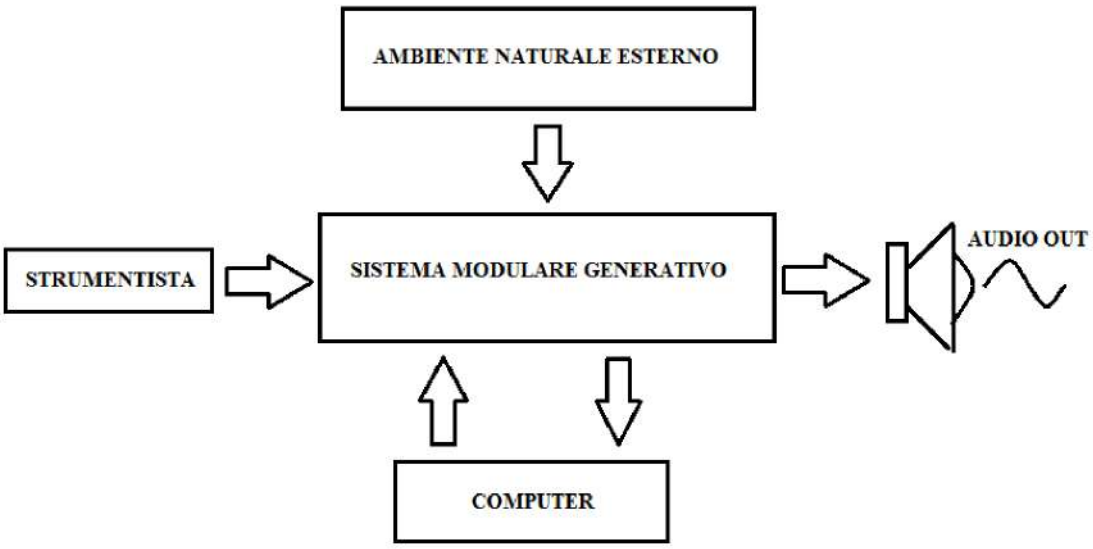
Sebbene quanto detto nella premessa del capitolo rimanga un punto cardine della progettazione, si è ritenuto anacronistico e sfavorevole non considerare in assoluto l'uso del

computer, poiché riesce a semplificare notevolmente determinati processi altrimenti dispendiosi e di non banale implementazione all'interno dello strumento. Come per il sistema GROOVE[76] di Mathews, il computer ha il compito di generare e inviare dei messaggi di controllo a un modulo specifico, che a sua volta li elabora e smista tra tutti gli altri elementi, rappresentando una sorta di assistente al calcolo. Nel 1970 Mathews non aveva altre possibilità di scelta e dovette sviluppare personalmente un metodo di comunicazione; ora si è ritenuta più valida l'opzione rappresentata dal MIDI, vista la sua affidabilità, larga diffusione sul mercato e quantità di informazioni trasmissibili con una sola interfaccia hardware. Con questo protocollo lo strumento non solo riesce a ricevere dati, ma può anche inviarne e nello specifico sono stati progettati due moduli che implementano il MIDI: uno per ricevere informazioni dal computer, convertirle in segnali interpretabili e diffonderle all'interno del sistema (MIDI-IN) e l'altro per svolgere il compito opposto (MIDI-OUT).

3.2.5 Considerazioni analitiche

Riassumendo quanto detto fino a questo punto, lo strumento può variare il proprio comportamento in base a:

- organizzazione interna tra i moduli
- interazione con lo strumentista
- dati provenienti dall'ambiente circostante
- informazioni scambiate con il computer



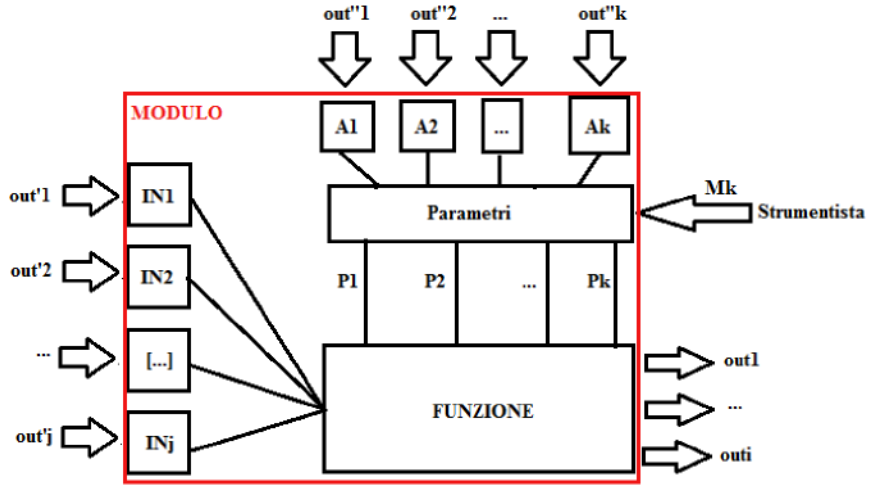
[76] Descritto nel paragrafo 1.5

Questa rete di comunicazione è capace di garantire al sistema un numero molto elevato di comportamenti possibili, a ognuno dei quali corrisponde un diverso risultato sonoro, sia dal punto di vista timbrico che relativamente all'organizzazione ed evoluzione nel tempo.
Di seguito è proposto un tentativo per formalizzare il concetto di combinazioni legato alla modularità, valido per un solo elemento, ma concettualmente estendibile allo strumento totale.
Gli output istantanei di un modulo generico posso essere così espressi:

$$out_i\ (IN_j(out'_j),\ P_k(A_k(out''_k),\ M_k)) \qquad (\textit{Equazione 3.1})$$

Dove:

- il pedice *i* indica il numero di output prodotti dal modulo
- il pedice *j* indica il numero di input elaborati dal modulo
- il pedice *k* indica il numero di parametri che modificano il comportamento del modulo
- out_i sono le informazioni prodotte in uscita da ogni modulo
- IN_j sono le funzioni con cui il modulo analizza le informazioni in ingresso, a sua volta dipendenti da out'_j
- out'_j sono le informazioni valide per essere accettate come input di analisi dal modulo, provenienti dalle uscite di uno o più moduli diversi
- P_k sono le funzioni rappresentate da parametri la cui variazione determina l'elaborazione delle informazioni in uscita
- A_k sono le funzioni che gestiscono la variazione automatica dei parametri, a loro volta dipendenti da out''_k
- out''_k sono le informazioni valide per essere accettate come input di modifica dal modulo, provenienti dalle uscite di uno o più moduli diversi
- M_k sono i termini che rappresentano la variazione manuale dei parametri da parte dello strumentista



E' utile specificare che:

- gli out_i di ogni unità possono essere considerati out'_j o out''_k per gli ingressi IN_j e A_k di qualsiasi altro modulo
- a parità di informazione inviata out_i, ogni modulo potrà interpretarla differentemente in funzione dei propri IN_j e A_k
- molti moduli hanno la capacità modulare sé stessi, cioè considerando i propri out_i come out'_j o out''_k
- gli out_i generati da ogni modulo possono essere elaborati da più unità successive e poi ripresentata come out'_j o out''_k al modulo di partenza
- la stessa informazione può essere duplicata in copie identiche e smistata tra più moduli diversi
- in base alla funzione del modulo, il numero di informazioni prodotte non è necessariamente pari al numero di informazioni ricevute (i >= j oppure i <= j)
- ai valori M_k potrebbero corrispondere un numero maggiore di parametri modificati effettivamente all'interno della funzione

Se si considera la situazione iniziale del sistema per cui nessun elemento è stato ancora collegato con tutti gli altri, un singolo modulo avrà la possibilità di accoppiare i propri out_i con gli IN_j o A_k di tutte le altre unità e il modulo successivo potrà scegliere di inviare i propri dati a qualsiasi altro ingresso valido, esclusi quelli già occupati dal primo modulo; il processo può essere iterato idealmente fino all'ultimo modulo, che avrà un insieme costituito dai pochi

elementi rimasti tra cui scegliere: il numero di combinazioni possibili tra ingressi e uscite seguirà quindi una logica fattoriale.

3.2.6 Implementare il sistema con Arduino

Arduino è una piattaforma hardware gestita da un microcontrollore programmabile in ambiente di sviluppo integrato, derivato dal linguaggio di Processing e Wiring. Nonostante esistano opzioni molto più performanti, per la realizzazione di ogni modulo si è scelto di utilizzare questa piattaforma vista la sua facilità di gestione, natura opensource, possibilità di scrivere e modificare agilmente il software e costi ridotti.

La stessa interfaccia hardware risulta essere piuttosto comoda per sviluppare dispositivi con design modulare: Arduino UNO, ad esempio, offre 14 pin programmabili per accettare informazioni binarie in input o per generarne in output, sei dei quali capaci di produrre segnali a modulazione di larghezza di impulso (PWM); la scheda è dotata inoltre di 6 pin di ingresso dedicati alla lettura di segnali analogici unipolari, convertiti da un ADC le cui caratteristiche permettono massima frequenza di campionamento pari a 9 kHz e quantizzazione a 10 bit, con conseguente discretizzazione del segnale in 1024 valori.

Esistono molte versioni hardware di Arduino, ognuna con qualità e performance diverse, basate principalmente su microcontrollori a 8 bit prodotti dalla Atmel: per lo strumento generativo si è scelto di gestire ogni modulo con una scheda Pro Mini, che riassume le stesse funzioni dell'Arduino UNO in un formato dalle dimensioni significativamente ridotte e con un costo economico molto minore.

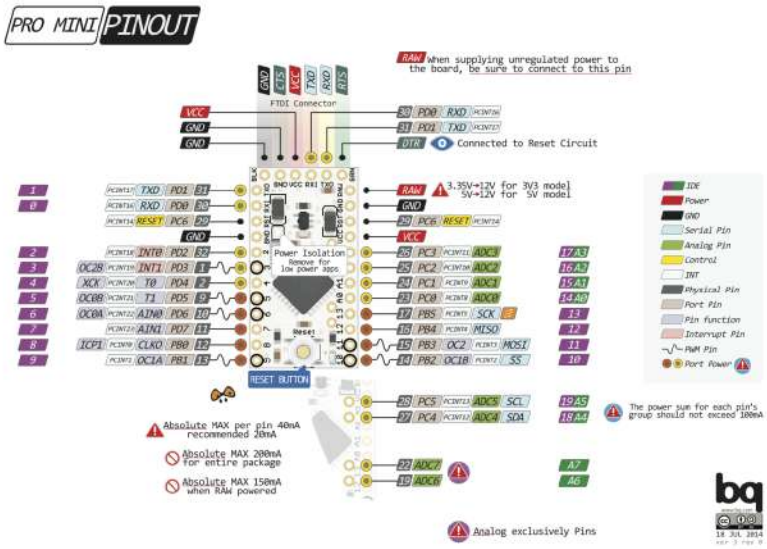


Figura 3.2 Disposizione e funzione dei pin presenti sulla scheda Pro Mini. Le dimensioni fisiche del circuito sono di 33x18 millimetri, contro 68x53 dell'UNO. Sono presenti 14 pin per I/O digitali e due pin analogici in più rispetto all'UNO, per un totale di 8. (*www.yourduino.com*)

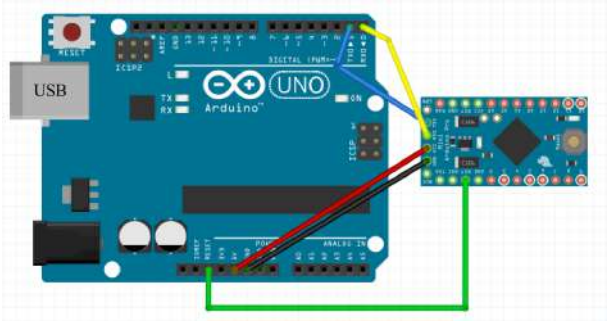
A ogni pin viene fatta corrispondere una funzione necessaria alla gestione del modulo, considerato sia come singola unità che come elemento del sistema completo:

- i pin digitali in modalità di lettura sono utili per analizzare informazioni binarie in ingresso, che possono essere interpretate come IN_j o A_k in base al design specifico del modulo. Vengono sfruttati anche per monitorare i controlli M_k dello strumentista, qualora siano presenti pulsanti e interruttori
- i pin digitali in modalità di scrittura sono utilizzati principalmente per generare e inviare informazioni binarie ad altri moduli. Possono assumere livelli di 0 o 5 Volt, rispettivamente per valori logici pari a 0 o 1
- i pin digitali capaci di generare PWM vengono sfruttati per ricreare un segnale idealmente *analogico.* In prima analisi questo comportamento appare paradossale, ma con degli accorgimenti software e hardware si riesce a ottenere un'informazione continua tra 0 e 5 Volt[77], valida come segnale audio o di controllo
- i pin analogici sono utili per leggere informazioni continue provenienti da altri moduli, utilizzate come ingressi per IN_j o A_k, o per acquisire dati da interfacce utente che sfruttano potenziometri. Determinati moduli usano questi pin per convertire informazioni

[77] Questa tecnica verrà approfondita in un paragrafo successivo

provenienti da sensori sia analogici che digitali; in quest'ultimo caso i pin A4 e A5 sono deputati alla gestione del protocollo I^2C, rappresentando rispettivamente i pin SDA e SCL

- i pin TXD, RXD sono generalmente necessari alla comunicazione seriale, il sistema modulare non sfrutta mai questa capacità in tempo reale, se non in fase di debugging o per il caricamento del software sulla scheda. In particolare viene utilizzato il driver presente su un Arduino UNO come ponte tra il computer e il Pro Mini, sprovvisto di interfaccia USB, secondo questo schema:



Un singolo Pro Mini riesce quindi a gestire agevolmente un buon numero di informazioni diverse e, pur se indispensabili alcuni accorgimenti in applicazioni audio, si ritiene essere un mezzo interessante sia per la progettazione di ogni modulo che per la costituzione dello strumento completo. Creare una rete modulare utilizzando solo schede Arduino garantisce uniformità in termini di componenti hardware utilizzabili (resistenze, condensatori, diodi, potenziometri, pulsanti, led, sensori) e metodologia di scrittura del software; in più definisce automaticamente uno standard per l'alimentazione, realizzazione costruttiva dei moduli e grandezza fisica dei segnali che circolano all'interno dello strumento: è fondamentale che ogni unità del sistema modulare sia perfettamente compatibile con tutte le altre sotto questi aspetti, altrimenti il meccanismo generale potrebbe risultare falsato, errato o, più probabilmente, inservibile.

Avere segnali omogenei è necessario soprattutto dal punto di vista della gestione e produzione sonora: i prossimi paragrafi esporranno la logica con cui queste informazioni vengono generate, acquisite e interpretate in modo coerente da ogni modulo del sistema.

3.3 Gestione di informazioni binarie

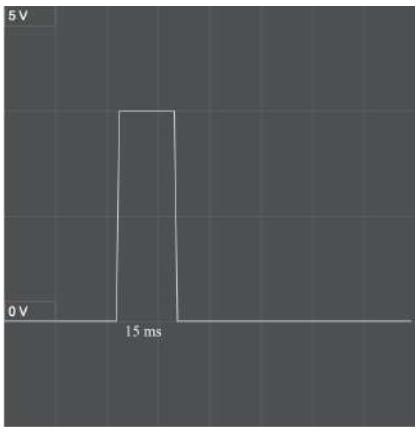
In un computer qualsiasi informazione è rappresentata in forma binaria, vale a dire sfruttando due simboli, indicati usualmente con 0 o 1, per convertirla in una successione di bit: il numero decimale 42, per esempio, è rappresentabile come 101010 dal suo corrispettivo binario.

Nel caso dello strumento modulare in analisi, un'informazione è considerata binaria anche solo quando assume valore pari 0 o 1 (0-5 Volt), non tutti i moduli del sistema sono stati progettati per interpretare più bit in maniera sequenziale come farebbe un computer, cioè tenendo conto dell'evoluzione temporale dell'informazione[78]: la maggior parte di loro analizza istantaneamente un unico bit in ingresso e solo da questo dipenderanno le informazioni prodotte in uscita[79].

In generale le informazioni binarie sono utili al sistema per gestire l'organizzazione degli eventi sonori e la logica interna di funzionamento della maggior parte dei moduli. In base alla propria funzione e dominio di esistenza nel tempo, è possibile dividere questi segnali in 3 tipologie: trigger, clock e gate.

3.3.1 Trigger

Il *trigger* è l'unità binaria più semplice ed elementare presente all'interno della famiglia di informazioni gestibili dal sistema modulare. Il termine inglese può essere tradotto con l'italiano *innesco,* che restituisce una buona immagine della sua funzione: il trigger è usato come impulso per dare il via a un determinato processo, portando alla generazione di suono o di altri segnali di controllo ogni qualvolta che il suo valore logico è pari a 1, cioè con tensione uguale o leggermente minore di 5 Volt.



Un trigger non porta con sé nessun'altra informazione se non quella relativa all'attivazione di eventi, motivo per cui la durata della sua fase diversa da zero è assolutamente ininfluente e può essere scelta arbitrariamente. Personalmente si è ritenuto funzionale un periodo positivo di 15 millisecondi, perché evita con sicurezza la perdita di informazione in fase di lettura e permette il trigger di eventi

[78] Circuito sequenziale

[79] Circuito combinatorio

consecutivi con frequenza anche superiore ai 60 Hz.

Il codice per generare un unico trigger con Arduino è concettualmente molto semplice:

```
void setup() {//funzione eseguita una sola volta
              //all'accensione della scheda

  pinMode(13, OUTPUT); //imposta il pin 13 come uscita

  digitalWrite(13, HIGH); //il pin eroga 5 Volt

  delay(15); //aspetta 15 millisecondi

  digitalWrite(13, LOW); //il pin eroga 0 Volt

}

void loop() { //funzione eseguita ciclicamente

}
```

L'esempio proposto è triviale e risulta inutile in qualsiasi applicazione: in questo modo verrebbe generato un unico trigger all'accensione della scheda. Tuttavia bisogna osservare che il codice non potrebbe essere scritto all'interno della funzione loop, poiché il trigger diventerebbe ciclico: un comportamento del genere può risultare utile solo per determinati moduli deputati a generare treni di impulsi, che verranno descritti nel paragrafo successivo. Un possibile approccio per risolvere questo problema è il seguente:

```
volatile bool evento; //dichiarare variabile binaria globale

void setup() { //funzione eseguita una sola volta all'accensione della scheda

 pinMode(13, OUTPUT); //pin impostato come uscita per il trigger
  pinMode(2, INPUT_PULLUP); //pin impostato come ingresso per trigger esterno

 attachInterrupt(digitalPinToInterrupt(2), evento_ISP, RISING);

}

void loop() { //funzione eseguita ciclicamente

 if (evento == HIGH) { //esegui questa funzione se "evento" è positivo

  evento = LOW; //"evento" è reso negativo per evitare ripetizioni cicliche indesiderate

   digitalWrite(13, HIGH); //il pin eroga 5 Volt

    delay(15); //aspetta per 15 millisecondi

   digitalWrite(13, LOW); // il pin eroga 0 Volt

  } //fine dell'if
} //fine del loop

void evento_ISP () { //definizione della funzione ISP

    evento = HIGH;
}
```

In questo modo verrà generato un solo trigger ogni qualvolta che la variabile *evento* diventa positiva, evitando così problemi di ciclicità.

Sebbene l'esempio non abbia alcuna utilità pratica, è interessante per introdurre un altro concetto molto importante relativo alla lettura e innesco di un trigger: non è sempre detto che un'informazione inviata da un modulo abbia la certezza di essere ricevuta con successo dall'unità destinataria; questo può accadere per vari motivi, il più banale dei quali causato da una o più funzioni *delay()* richiamate nel codice del modulo ricevitore.

3.3.2 Interrupt Service Routines

Oltre che limitando delay ai soli casi di estrema necessità, si può risolvere il problema utilizzando le *Interrupt Service Routines* (ISR), speciali funzioni con la peculiarità di poter interrompere l'esecuzione del codice principale ogni volta che si verifica un tale evento sul pin che stanno osservando. Una ISR offre la certezza di non perdere nessun impulso esterno, pur garantendo il normale funzionamento del programma generale, che riprenderà appena la ISR avrà adempiuto al suo compito; da questo punto di vista è consono mantenere la funzione quanto più snella possibile, come nell'esempio precedente, e utilizzare variabili globali dichiarate *volatile* per far si che siano aggiornate sempre correttamente nel passaggio da una funzione all'altra.

Un aspetto piuttosto interessante è che il codice principale non riprenderà dalla linea in cui è stato interrotto, ma da un altro punto casuale, seguendo una *logica* che non si è in grado di rendere in alcun modo esplicita.

Un Arduino Pro Mini ha la possibilità di utilizzare la ISR sui pin digitali 2 e 3, da impostare in modalità di lettura e su cui è consigliato l'utilizzo di una resistenza di pull-up o pull-down secondo necessità; nel caso in cui siano attivati entrambi i pin contemporaneamente, varrà un principio di priorità tra le due ISR, introducendo un leggero sfasamento, nell'ordine dei microsecondi, tra gli eventi innescati.

La sintassi per la gestione di queste funzioni va richiamata all'interno del setup(), in questo modo:

```
attachInterrupt (digitalPinToInterrupt (numero_pin), nome_funzione, modalità_ISR)
```

Le modalità di attivazione possibili sono le seguenti:

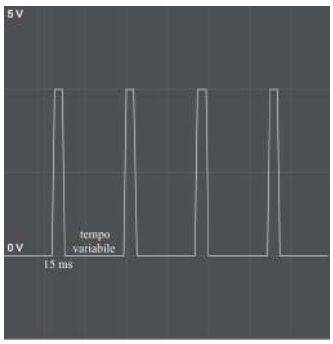
- LOW, attiva la funzione appena il pin legge un valore di tensione nullo

- RISING, attiva la funzione appena si passa da un valore nullo a un valore positivo[80]
- FALLING, attiva la funzione appena si passa da un valore positivo a un valore nullo
- CHANGE, attiva la funzione appena è rilevato un cambiamento di valore

Ciascuna di queste opzioni ha i suoi vantaggi in situazioni diverse; nel caso della lettura di un trigger risulta più utile la modalità RISING, perché garantisce un'ottima sincronizzazione tra gli impulsi ricevuti e gli eventi innescati.

3.3.3 Clock

Il *clock* è il metronomo del sistema modulare, necessario a rendere sincrono il comportamento di più unità. Questa informazione è costituita da segnali di trigger consecutivi, ciclici e



opportunamente distanziati nel tempo, la cui frequenza determina la velocità di esecuzione delle funzioni di ogni modulo. L'unità di misura musicale associabile al clock è il BPM (Battiti Per Minuto), che, tipicamente con riferimento alla semiminima, identifica il numero di eventi presenti in 60 secondi. Può essere spesso utile convertire questa grandezza, ricavando il tempo in millisecondi che intercorre tra due trigger consecutivi (T_{ms} = 60000 / BPM), da cui è un possibile scrivere un programma per generare un segnale di clock con Arduino:

```
int potenziometro_BPM;
 float tempo_BPM;
 float tempo_virgola;
  int tempo_intero;
   float tempo_compensa;

void setup() {
 pinMode(13, OUTPUT); // definisco il pin che genera il clock
}

void loop() {
  //uso un potenziometro per impostare manualmente i BPM
  //e riscalo il valore letto dal potenziometro tra 30 e 240 BPM
  potenziometro_BPM = map(analogRead(A0), 0, 1023, 30, 240);

 tempo_BPM = (float) (60./ potenziometro_BPM); //calcolo secondi per battito
  tempo_virgola = tempo_BPM * 1000.; //riscalo in millisecondi per battito
   tempo_intero = tempo_virgola; // elimino eventuali termini dopo la virgola
 //calcolo eventuale compensazione in microsecondi
    tempo_compensa = (tempo_virgola - tempo_intero) * 1000.;

  digitalWrite(13, HIGH); //il pin eroga 5 Volt
   delay(15); //aspetto 15 millisecondi
  digitalWrite(13, LOW); //il pin eroga 0 Volt
   delay(tempo_intero - 15); //aspetto il tempo necessario per il prossimo clock
                             // considerando i 15ms del trigger già trascorsi
   delayMicroseconds(tempo_compensa);
  //aspetto per un eventuale tempo di scompenso dato da termini dopo la virgola
   }
```

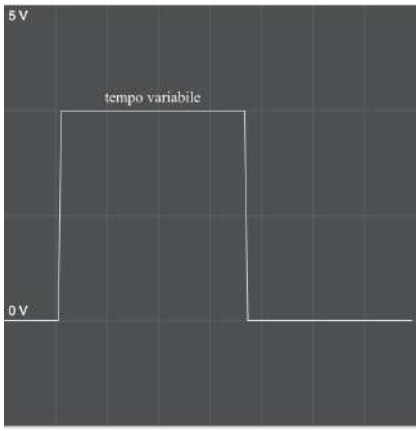
[80] Si è trovato empiricamente che la soglia di attivazione si aggira intorno a 1.5 Volt

Per motivi di maggiore chiarezza si è scelto di costruire l'esempio in questo modo, sebbene il passaggio iniziale di calcolo dei millisecondi a partire dai valori di BPM sia superfluo: sarebbe certamente più pratico per il software dichiarare, senza ulteriori calcoli, il tempo in millisecondi tra due trigger consecutivi, limitando solo gli estremi di delay inferiore e superiore. In questo caso, considerate le funzioni usate e la tipologia di algoritmo, la variabile *tempo_intero* non ha necessità di essere limitata superiormente, ma il suo minimo non può scendere al di sotto di 16 millisecondi, cui equivalgono 3750 impulsi al minuto. Un BPM così sostenuto può essere interessante per ottenere clock capaci di entrare in banda audio, sebbene mantenere un estremo così basso implichi un'escursione troppo ampia nei valori temporali possibilmente selezionabili tramite il potenziometro, con un risultato percettivamente poco valido. Nel capitolo successivo verrà descritto il modulo che è stato progettato per essere il master clock dello strumento generativo, il cui algoritmo risolve questo scompenso; per ora è necessario osservare che dal punto di vista psicoacustico è generalmente inefficace riscalare linearmente (funzione map), i 1024 valori campionati dall'ADC, risultando più conveniente implementare curve con caratteristiche che assecondino maggiormente la percezione umana.
Un altro problema da non sottovalutare è la dose di rumore cui sono tipicamente soggette le informazioni acquisite dai pin analogici: una minima fluttuazione dei valori letti dal potenziometro comporterebbe uno sfasamento indesiderato tra il clock idealmente impostato e quello effettivamente ottenuto. Oltre ad alcune valide soluzioni hardware, può essere utile implementare nel software il seguente accorgimento:

```
int lettura_pot;
 int lettura_pot_pre;
 int scarto;
 const int sens = 10;//sensibilità del controllo
 int valore; //valore da utilizzare
 //esempi di estremi per la funzione di riscalamento
 const int minimo = 15;
 const int massimo = 1000;

void setup() {
//nulla da dichiarare nel setup
}

void loop() {

 lettura_pot = analogRead(A0); //leggo da pin A0

//calcolo scarto tra lettura e lettura precendete del potenziometro
  scarto = abs(lettura_pot - lettura_pot_pre);

  if (scarto > sens) { //se lo scarto supera la soglia di sensibilità

    valore = map(lettura_pot, 0, 1023, minimo, massimo);

     }

  lettura_pot_pre = lettura_pot;

}
```

3.3.4 Gate

Il *gate* è il segnale binario utilizzato dal sistema per trasmette e acquisire informazioni di durata. La differenza sostanziale rispetto a un trigger risiede nella lunghezza variabile della sua fase positiva, per tutto il tempo della quale accade un dato evento o è attiva una certa funzione. In termini musicali, il gate può rappresentare per lo strumento modulare l'equivalente della figurazione ritmica per la notazione classica: una semibreve avrà gate positivi molti più lunghi di una biscroma, così come a diversi tempi di segnale nullo corrispondono valori maggiori o minori di pausa. All'interno del sistema che si è progettato, la lunghezza della fase positiva di un gate può avere tempo minimo di 15 millisecondi, diventando quindi un trigger, e massimo teoricamente infinito, ma tipicamente limitato a 15 secondi.



Due esempi generici per l'acquisizione ed elaborazione di un gate possono essere i seguenti:

```
bool gate_in;

void setup() {

   pinMode(13, INPUT);

}

void loop() {

  gate_in = digitalRead(13);
  while(gate_in == HIGH) {
     funz_es1();
  }
}

void funz_es1() {
  //funzione esempio
  //monitora variabile gate_in
   gate_in = digitalRead(13);
}
```

```
bool gate_in;

void setup() {

 pinMode(13, INPUT);

}

void loop() {

 gate_in = digitalRead(13); //lettura gate

 if (gate_in == HIGH) { //se il gate è positivo
    funz_es1(); //esegui funzione1
 }
  else {  // istruzione opzionale
     funz_es2();
  }

} //fine loop

void funz_es1() {
  //scrivi funzione 1
}

void funz_es2() {
  //scrivi funzione 2
}
```

Un gate può essere sfruttato anche per attivare funzioni o eventi immediatamente dopo la fine della sua fase positiva, interpretando il segnale come se fosse un trigger con comportamento inverso, cioè innescato per variazioni logiche da 1 a 0. Una possibile applicazione di questa tecnica è utile per introdurre lunghi ritardi o swing nella temporizzazione di un evento, il cui

sfasamento dipende dalla durata del gate. A tale scopo risulta pratica una funzione ISR in modalità FALLING, come nell'esempio seguente:

```
volatile bool flag;

void setup() {
 pinMode(13, OUTPUT);
  pinMode(2, INPUT_PULLUP);

  attachInterrupt(digitalPinToInterrupt(2), es_swing, FALLING);

}

void loop() {

  if (flag == HIGH) {
    flag = LOW;
    digitalWrite(13, HIGH);
    delay(15);
    digitalWrite(13, LOW);
  }

}

void es_swing () {
   flag = HIGH;
}
```

Per avere controllo sul tempo di ritardo o swing è necessaria la presenza di un altro modulo che permetta di poter variare la durata del gate nella sua fase positiva:

```
volatile bool flag; //trigger per attivare il gate
int durata_gate; //variabile per controllare la durata del gate

void setup() {
  pinMode(13, OUTPUT);
   pinMode(2, INPUT_PULLUP);
 attachInterrupt(digitalPinToInterrupt(2), trig_in, RISING);
}

void loop() {
//imposto la durata del gate tramite potenziometro
//la durata minima deve essere pari a 15 ms
//la durata masssima non ha limitazioni teoriche
 dutara_gate = map(analogRead(A0), 0, 1023, 15, 8000);

 if (flag == HIGH) { //eseguo funzione alla ricezione del trigger
   flag = LOW;
    digitalWrite(13, HIGH); //attivo il gate
     delay(durata_gate); //aspetto per la durata della fase positiva
    digitalWrite(13, LOW); // disattivo il gate
 }

} //fine del loop

void trig_in (){
  flag = HIGH;
}
```

Secondo i due esempi proposti, l'innesco dell'evento del primo modulo può avvenire al massimo 8000 millisecondi dopo la ricezione del trigger dell'unità deputata a generare il gate. Sebbene l'ultimo algoritmo possa apparire funzionale, ha il notevole svantaggio di sfruttare la funzione delay per un tempo che può essere anche molto lungo, durante il quale il programma non può compiere nessun'altra operazione: rendere la scheda inerte per 8 secondi implica, nel caso più banale, che qualsiasi informazione proveniente dall'interfaccia utente venga ignorata.

Risolvere la problematica di questa *finestra di buio* ha numerosi vantaggi sia dal punto di vista della ricezione di input esterni, che relativamente alla gestione di più eventi interni, a cui è possibile assegnare temporizzazioni indipendenti.

3.3.5 La funzione millis()

La funzione *millis()* si attiva all'accensione della scheda Arduino e tiene conto dei millisecondi passati a partire da quell'istante: è pratica per dare al software una misura del tempo che sta trascorrendo, rendendo in molti casi evitabile l'utilizzo della funzione delay().

Poiché la variabile associata può raggiungere cifre elevate, è generalmente prudente considerare di dichiararla come *unsigned long* per garantire la memorizzazione di 32 bit di informazioni prima dell'overflow, che avverrebbe in questo caso dopo circa 50 giorni dall'accensione.

Il seguente programma sfrutta la funzione millis per generare due segnali di gate con lunghezza indipendente:

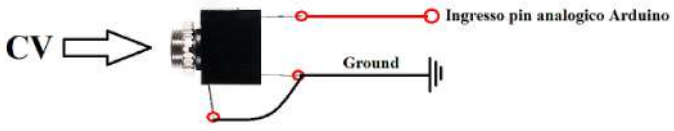
```
int durata_gate[2]; //variabili per lunghezza gate
volatile bool trig_in[2]; //variabili per trigger esterni
 bool out[2] = {LOW, LOW}; //variabili per monitorare out
  unsigned long millis_pre[2] = {0, 0}; //variabili per informazioni sul tempo
 const int array_pin[2] = {13, 12};
 void setup() {
 pinMode(13, OUTPUT); //pin generazione primo gate
  pinMode(12, OUTPUT); //pin generazione secondo gate
   pinMode(3, INPUT_PULLUP); //trigger primo gate
    pinMode(2, INPUT_PULLUP); //trigger secondo gate
 attachInterrupt(digitalPinToInterrupt(3), input_1, RISING); //ingresso trigger1
  attachInterrupt(digitalPinToInterrupt(2), input_2, RISING); //ingresso trigger2
}

void loop() {
for (int i = 0; i < 2; i++) {
 durata_gate[i] = map(analogRead(i), 0, 1023, 15, 8000); //imposta durata del gate

  if (trig_in[i] == HIGH) {
   trig_in[i] = LOW;
    out[i] = HIGH;
     digitalWrite(array_pin[i], HIGH); //attivo il gate i
      millis_pre[i] = millis(); //memorizzo il tempo attuale
 }
//se l'out i è attivo e il tempo trascorso dalla sua attivazione è pari alla lunghezza del gate i, allora lo disattivo
if ((out[i] == HIGH) && ((millis() - millis_pre[i]) >= durata_gate[i])) {
    out[i] = LOW;
    digitalWrite(array_pin[i], LOW);
  }
 } //fine del for
} //fine del loop
void input_1 () {
 trig_in[0] = HIGH;
  }
void input_2 () {
 trig_in[1] = HIGH;
 }
```

3.4 Gestione di informazioni continue

Oltre alle informazioni binarie, il sistema modulare progettato fonda il proprio principio di gestione su segnali che posso assumere valori di tensione continui all'interno dell'intervallo tra 0 e 5 Volt, storicamente definiti dai costruttori di sintetizzatori analogici con il termine C*ontrol Voltages* (CV). Le informazioni scambiate tramite CV appartengono a una famiglia ampia, i cui compiti possono essere molto diversi tra loro: nell'ottica dello strumento generativo sono sfruttati per modificare automaticamente qualsiasi parametro che richiederebbe altrimenti l'intervento del musicista, grazie a moduli progettati per la loro generazione, elaborazione e ricezione. Di seguito è proposto un esempio che permette di associare a un solo parametro l'azione congiunta di CV e controllo manuale:

```
int val_pot;
 int val_CV;
 int parametro;
 const int massimo;
 const int offset = 20;

void setup() {

}

void loop() {
// acquisisco informazioni controllo manuale
val_pot = map(analogRead(A0), 0, 1023, 0, massimo);

// acquisisco informazioni CV
val_CV = map(analogRead(A1), 0, 1023, 0, massimo);

// sommo le azioni nel parametro finale
//il valore di offset può essere anche nullo
//è utile nel caso sia necessario avere un estremo inferiore
parametro = val_pot + val_cv + offset;

}
```

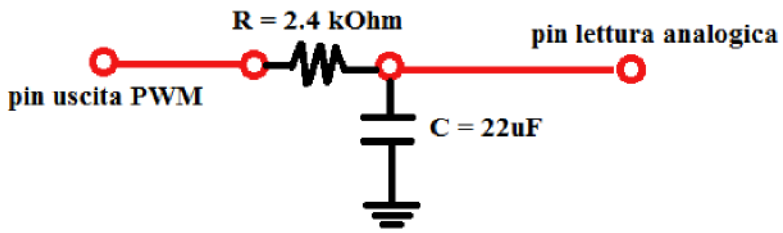
La variabile *parametro* può essere quindi usata per modificare qualsiasi funzione tenendo conto dell'azione congiunta di strumentista e CV (termini M_k e A_k); qualora sia presente un segnale al pin analogico A1, va considerato il potenziometro come un suo generatore di offset positivo. Dal punto di vista hardware è tipicamente utilizzato un potenziometro lineare da 10000 Ohm per la modifica manuale e un jack femmina per la ricezione della tensione di controllo, evitando problemi di rumore indesiderato se nessun maschio è collegato al suo ingresso in questo modo:



3.4.1 Generare un segnale continuo con Arduino

Un Arduino Pro Mini non possiede uscite DAC dedicate, sebbene si possa simulare un comportamento sufficientemente simile con la funzione *analogWrite()*, che rappresenta il corrispettivo di analogRead in fase di scrittura. Determinati pin digitali[81] possono generare onde quadre unipolari di cui è dato controllare il duty cycle, cioè la porzione di periodo per cui il segnale ha valore diverso da zero: questo rapporto tra durata del segnale alto e basso è tipicamente espresso in percentuale, discretizzata da Arduino tramite 256 valori per ottenere un'informazione in uscita con 8 bit di profondità. La funzione analogWrite interpreta questo byte e modifica di conseguenza il duty cycle dell'onda compiendo una modulazione della larghezza d'impulso (PWM dall'acronimo inglese di Pulse Width Modulation), per cui a valore 0 corrisponde un segnale costantemente nullo, così come a 255 corrispondono 5 V; in teoria il numero 128 equivarrebbe a circa 2.5 V, in realtà per fare in modo che il pin analogico di un'altra scheda legga questo valore è necessario un accorgimento hardware.

Pilotare l'intensità luminosa di un LED in PWM è molto semplice con Arduino, sebbene si tratti in effetti di un *inganno* percettivo: la tensione che passa attraverso il componente è sempre logicamente pari a 0 o 1, di fatti il LED non fa altro che spegnersi e accendersi per periodi più o meno lunghi, ma con una frequenza troppo alta rispetto alle possibilità di *campionamento* dell'occhio umano, che *filtra* questa informazione facendo percepire un'intensità luminosa variabile. E' appunto con un filtro passivo passa basso RC che è possibile ottenere un valore di tensione continua (DC) a partire da una PWM, rendendo il segnale generato da Arduino perfettamente interpretabile da un analogRead o da qualsiasi altro circuito analogico. Non si esporranno i principi teorici di dimensionamento del filtro, sicuramente necessari in una prima fase di progettazione e in ogni caso poi affiancati da varie prove empiriche, viene semplicemente proposto di seguito il circuito che si è ritenuto essere la soluzione più funzionale da implementare all'interno dello strumento.



[81] I pin digitali 11, 10, 9, 6, 5 e 3.

Tutti i segnali di controllo descritti di seguito prevedono l'utilizzo di questo filtro all'uscita dei rispettivi pin PWM di generazione.

3.4.2 LFO

LFO è l'acronimo inglese per Low Frequency Oscillator, un oscillatore la cui frequenza ricade al di sotto della banda udibile, tipicamente non eccedendo i 20 Hz. Questo tipo di segnale è utile per controllare automaticamente un parametro in modo periodico, caratterizzando la tipologia di variazione in funzione della forma d'onda generata dall'oscillatore.

Nel capitolo precedente è stato descritto il principio fisico che regola l'evoluzione temporale del brano Pendulum Music: come si è osservato, un pendolo semplice oscilla secondo un moto armonico, influenzato dalla forza di gravità che tende a fargli guadagnare sempre la posizione centrale di perpendicolarità con il suolo. Immaginando di eliminare tutti gli smorzamenti causati dagli attriti, il pendolo avrà un profilo di velocità ciclico, con valore massimo quando la massa si trova nella posizione centrale e nullo agli estremi, accelerando o decelerando nel tendere alle fasi di inversione del moto.

Un LFO sinusoidale è utile per modellare il comportamento appena descritto: i parametri controllati automaticamente da questo segnale restituiscono, per la maggior parte dei casi, la percezione di una variazione non lineare nel tempo, poiché soggetti ai cambiamenti periodici di accelerazione e decelerazione. Da questo punto di vista, è possibile ottenere un risultato più costante nel tempo con un LFO triangolare, che può essere considerato come un pendolo con la capacità di poter invertire istantaneamente la direzione del proprio moto, senza variare il modulo della velocità. Altri segnali classici producibili da un oscillatore a bassa frequenza sono l'onda quadra e l'onda a dente di sega: la prima ha la caratteristica di variare ciclicamente un controllo tra soli due valori e quindi associabile a una manipolazione binaria; nel secondo caso il segnale parte dal valore minimo per arrivare linearmente all'estremo superiore, raggiunto il quale riguadagna istantaneamente l'ampiezza nulla di partenza.

A parità di parametro controllato, con ognuna di queste forme d'onda si otterranno risultati sonori o, più generalmente, comportamenti differenti da ogni modulo: nell'ottica dello strumento generativo, è spesso utile controllare unità diverse con vari LFO, aventi opportuna ampiezza, frequenza, fase. In questi casi è interessante notare come la forma finale ottenuta sia plasmata da segnali singoli che pur se elementari nella loro ciclicità, una volta combinati

siano capaci di produrre risultati notevolmente complessi e vari nel tempo, con una periodicità tanto dilatata da non risultare più percepibile.

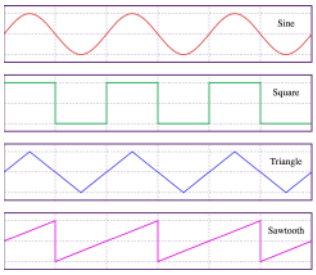


Figura 3.3 Rappresentazione grafica delle forme d'onda descritte. L'onda a dente di sega nell'immagine è crescente, ma può avere andamento opposto. (Wikimedia Commons)

Queste forme d'onda possono essere generate dal codice proposto di seguito:

```
byte val_saw1 = 0; //byte per sfruttare il troncamento dell'overflow
 byte val_saw2 = 255;
  byte val_sqr;
   byte indice_sqrt = 255;
    byte val_tri = 0;
     byte indice_tri = 0;
      float val_sin_f = 0;
       byte val_sin;
       byte indice_sin;
  int tempo_step; //tempo di attesa tra il calcolo di ogni valore
 //secondi_tot_periodo = (tempo_step * 255) / 1000 * 1000 -> da microsecondi a secondi
 //freq_LFO = 1/secondi_tot_periodo
  const double greco = 3.141592;
  const byte soglia_sqr = 127; //è possibile renderla variabile per impostare il duty cycle
  const byte soglia_tri = 127; //è possibile renderla variabile

void setup() {
//definisco pind PWM di generazione
 pinMode(11, OUTPUT); //onda a dente di sega crescente
  pinMode(10, OUTPUT); //onda a dente di sega decrescente
   pinMode(9, OUTPUT); //onda triangolare
    pinMode(6, OUTPUT); //onda sinusoidale
     pinMode(5, OUTPUT); //onda quadra
}
void loop() {
//imposto delay tra 100 microsecondi (~40 Hz) e 16383 (~0.25 Hz)
 tempo_step = map(analogRead(A0), 0, 1023, 100, 16383);
  delayMicroseconds(tempo_step); //aspetto per calcolo step successivo

if (indice_sqrt >= soglia_sqr) {
      val_sqr = 255;
  }
    else {
      val_sqr = 0;
    }

    if (indice_tri <= soglia_tri) {
      val_tri = indice_tri * 2;
  }
    else {
      val_tri = 255 - indice_tri * 2;
    }

indice_sin = indice_tri - 127;
 val_saw1++;
  val_saw2 = 255 - val_saw1;
   indice_sqrt--;
    indice_tri++;
     indice_sin++;

    val_sin_f = (((cos(greco * (indice_sin/127.00))) * 0.5) + 0.5) * 255.00;
     val_sin = val_sin_f;

   analogWrite(11, val_saw1);
    analogWrite(10, val_saw2);
     analogWrite(9, val_tri);
      analogWrite(6, val_sqr);
       analogWrite(5, val_sin);
}
```

3.4.3 Segnali randomici

Oltre che con le forme d'onda precedenti, è possibile controllare un parametro con qualsiasi altro segnale, non limitando la sua frequenza a oscillazioni infrasoniche o, soprattutto, periodiche: può essere interessante assegnare il valore dell'ampiezza istantanea di un onda in maniera casuale, per ottenere comportamenti non prevedibili nel tempo.

In generale per ottenere una sequenza numerica realmente randomica è necessario un apparato hardware che acquisisca informazioni da fenomeni fisici microscopici, come il rumore termico, così da non introdurre ciclicità prevedibili anche dopo periodi di osservazione molto lunghi o idealmente infiniti; un computer, al contrario, può sfruttare solo algoritmi deterministici capaci di approssimare questi comportamenti, introducendo necessariamente periodicità e prevedibilità, motivo per cui sarebbe più rigoroso definire le sequenze prodotte digitalmente con il termine pseudo-randomiche. Il seme di un generatore digitale casuale identifica le caratteristiche cicliche delle sequenze producibili: in Arduino questo valore è definito dalla funzione *randomSeed(),* che accetta come argomento una variabile a 32 bit, quindi garantendo più di 4 miliardi di opzioni possibili. Immaginando di produrre da ogni seme una sequenza di un minuto, contenente un numero casuale al secondo, sarebbero necessari circa 8000 anni affinché le successioni si ripropongano uguali a sé stesse, un ciclo abbondantemente sufficiente per applicazioni musicali; per questo motivo tutti i segnali casuali scambiati tra i moduli dello strumento verranno considerati randomici, sebbene siano prodotti digitalmente. Di seguito un codice di esempio per generare in PWM 3 segnali randomici:

```
int pot1, pot2, minimo, massimo;
byte rand1, rand2, rand3;
int rand3_pre = -1;
volatile bool trigger;

void setup() {
pinMode(11, OUTPUT); //pin generazione PWM valore randomico
 pinMode(10, OUTPUT);
  pinMode(19, OUTPUT);
pinMode(3, INPUT_PULLUP); //pin trigger esterno
randomSeed(analogRead(A0)); //sfrutto rumore letto da un pin analogico scollegato
 attachInterrupt(digitalPinToInterrupt(3), genera_numero, RISING);
}

void loop() {

 if (trigger == HIGH) {
   trigger = LOW;
pot1 = analogRead(A1) * 0.25; //imposto massimo valore casuale uscita 1
 pot2 = analogRead(A2) * 0.25; //imposto massimo valore casuale uscita 2
  rand1 = random(pot1); //genero primo valore casuale
   rand2 = random(pot2); //genero secondo valore casuale
  minimo = min(pot1, pot2); //calcolo estremi terzo valore casuale
   massimo = max(pot1, pot2) + 1;
    rand3 = random(minimo, massimo); //genero terzo valore casuale

while (rand3 == rand3_pre && pot1 != pot2) { //evito ripetizioni successive dello stesso numero
       rand3 = random(minimo, massimo);
     }
 analogWrite(11, rand1); //scrivo sulle uscite digitali
  analogWrite(10, rand2);
   analogWrite(9, rand3);
    }
rand3_pre = rand3;
}

void genera_numero () { trigger = HIGH; }
```

Non è quasi mai detto che una sequenza totalmente randomica possa risultare musicalmente interessante o funzionale alla gestione di altri moduli, per regolarizzare la generazione dei numeri casuali può essere utile fornire principi organizzativi di vario tipo, come nel caso della passeggiata aleatoria:

```
int pot_falcata, posizione, pot_meta, pot_meta_pre, meta = 50, falcata, direzione, minimo, massimo;
 bool var_direzione, flag = LOW;
  float pot_deviazione, dev_min, dev_max;

void setup() {
randomSeed((analogRead(A0) + analogRead(A0) + analogRead(A0)) * 0.3); //imposto il seme in base alla posizione iniziale dei pot
  posizione = meta; //max 250 min 15
  pinMode(11, OUTPUT); //pin generazione segnale
  pinMode(3, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(3), in_1, RISING);
}

void loop() {

pot_meta = analogRead(A0); //meta della passeggiata
if (abs(pot_meta - pot_meta_pre) > 10) { //se vario la meta, la posizione istantanea parte da quel punto
  meta = map(pot_meta, 0, 1023, 15, 255);
 posizione = meta;
}

pot_falcata = map(analogRead(A1), 0, 1023, 2, pot_meta * 0.5); //imposto il numero di posizioni possibili ad ogni passo
pot_deviazione = (float)map(analogRead(A2), 0, 1023, 10, 99) * 0.01; //imposto la deviazione possibile rispeto alla meta
dev_min = 1 - pot_deviazione;
dev_max = 1 + pot_deviazione;
minimo = constrain((dev_min * meta), 0, 255);
massimo = constrain((dev_max * meta), 0, 255);

if (flag == HIGH) { //alla ricezione di un trigger esterno
 flag = LOW;
 var_direzione = random(2); //direzione randomica
 if (var_direzione == 0) { direzione = 1; }
  else { direzione = -1;}

 falcata = (random(pot_falcata) + 1) * direzione; //la falcata può essere positiva o negativa

  posizione = posizione + falcata; //vario la posizione precedente in base alla falcata attuale

  if (posizione <= minimo || posizione >= massimo) { //limito le possibilità

     posizione = posizione + falcata * -1;
  }

  analogWrite(11, posizione); //scrivo sul pin PWM
}
  pot_meta_pre = pot_meta;
}

void in_1() { //funzione attivata alla ricezione del trigger
  flag = HIGH;
}
```

3.4.4 Rampe e inviluppi

Un inviluppo, o rampa, è un segnale di controllo che permette la variazione temporale di un parametro a partire da una condizione di innesco, differentemente da un LFO non ha necessariamente comportamento ciclico e la propria evoluzione nel tempo mira a interpolare determinati valori in più fasi, con funzioni generalmente lineari, esponenziali o logaritmiche. Un classico esempio di inviluppo è l'ADSR (Attack, Decay, Sustain, Release), di cui è possibile impostare il dominio temporale e gli estremi di ogni fase per ottenere una tensione di controllo utile soprattutto alla modifica del timbro di un suono.

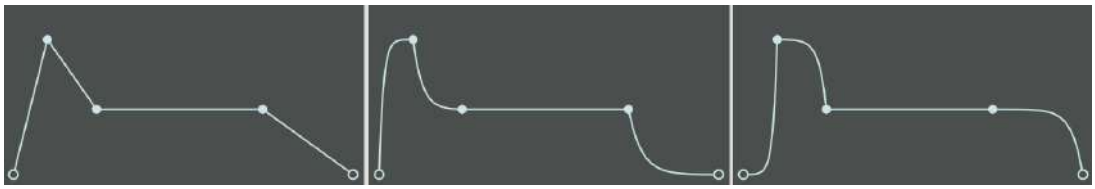


Figura 3.4 Alcuni esempi di inviluppi ADSR con andamenti lineari, esponenziali e logaritmici

Un inviluppo può avere un numero variabile di stadi possibili, per il sistema progettato si è ritenuto vantaggioso considerare rampe con una sola fase ascendente e discendente (AD), attivabili tramite trigger, in questo modo:

```
bool flag = LOW;
 float fatt_inviluppo, exp_A = 1., exp_D = 0.5 ;
  int val_inviluppo, tempo_A, tempo_B;

void setup() {

 pinMode(11, OUTPUT); //pin generazione inviluppo
 pinMode(3, INPUT_PULLUP); //pin trigger esterno
 attachInterrupt(digitalPinToInterrupt(3), in_1, RISING);
}

void loop() {
 tempo_A = map(analogRead(A0), 0, 1023, 100, 15000);
  tempo_D = tempo_A;// map(analogRead(A1), 0, 1023, 100, 15000);

if (flag == HIGH) { //alla ricezione del trigger esterno
    flag = LOW;

for(int i = 0; i <= 255; i++) { //fase Attack
 exp_A = map(analogRead(A2), 0, 1023, 10, 1000) * 0.01;
     fatt_inviluppo = pow(i/255.00, exp_A);
      val_inviluppo = fatt_inviluppo * 255;
    delayMicroseconds(tempo_A);
     analogWrite(11,val_inviluppo);
      }

for(int i = 255; i >= 0; i--) { //fase Decay
   exp_D = map(analogRead(A3), 0, 1023, 10, 1000) * 0.01;
     fatt_inviluppo = pow(i/255.00, exp_D);
      val_inviluppo = fatt_inviluppo * 255;
    delayMicroseconds(tempo_D);
     analogWrite(11, val_inviluppo);
      }
    } //fine trigger
} //fine loop

void in_1() { flag = HIGH; }
```

Un'ulteriore modalità di innesco può essere ottenuta con un gate, alla ricezione del quale viene attivata la fase di attacco e il cui valore massimo viene mantenuto fintantoché il gate ha valore logico unitario; si passa quindi allo stadio di decadimento nel momento in cui il segnale di controllo diventa nullo, generando in questo modo un inviluppo AHD (Attack, Hold, Decay).

3.4.5 Il protocollo Volt/ottava

Il segnali descritti fino a questo punto hanno lo scopo di poter variare automaticamente qualsiasi tipo di parametro musicale in funzione della tipologia di ingresso scelto per la loro acquisizione. Modificare la frequenza di un oscillatore con un LFO risulta interessante nella

maggior parte dei casi, ma può precludere al sistema la possibilità di interfacciarsi con determinati linguaggi musicali che fanno del controllo preciso del pitch un parametro compositivo fondamentale, sia dal punto di vista istantaneo, che rispetto al rapporto nel tempo tra più eventi. Il protocollo Volt/ottava ha storicamente permesso ai sintetizzatori modulari di codificare al proprio interno una logica capace di suddividere lo spettro sonoro secondo le unità di tono e semitono: ogni ottava è discretizzata nel range di 1 Volt, a sua volta ripartito in 12 intervalli di tensione che assecondando il rapporto frequenziale del temperamento equabile. Sebbene questo aspetto sia considerato personalmente marginale, il protocollo Volt/ ottava risulta comunque vantaggioso da un punto di vista percettivo, perché permette di associare al parametro fisico della frequenza un giusto corrispettivo psicoacustico; riscalare linearmente i valori letti da un potenziometro per controllare la frequenza di un oscillatore, ad esempio tra 50 e 5000 Hz, implica una risoluzione troppo grossolana guardando all'estremo inferiore e inutilmente precisa, perché impercettibile, quando si tende al limite superiore. Piuttosto che implementare funzioni di compensazione in fase di lettura o scrittura, si è scelto di sviluppare un personale protocollo di comunicazione con ingressi e uscite hardware dedicate, avendo l'accorgimento ulteriore di codificare determinate regole tonali per permettere al sistema di interagire, anche sotto questo aspetto, con qualsiasi altro strumento e linguaggio tradizionale.

Il seguente array rappresenta la codifica di 5 ottave suddivise per intervalli cromatici, cui corrispondono i valori di duty cycle necessari a far generare in PWM queste informazioni.

```
const int scala_cromatica [64] = {0, //nota off e incremento di semitono = +4
//+ 48 per incrementare di ottava, do   do#  re   re#   mi  fa  fa#  sol sol#   la  la#   si
                                    4,   8,  12,  16,  20,  24,  28,  32,  36,  40,  44,  48,
                                   52,  56,  60,  64,  68,  72,  76,  80,  84,  88,  92,  96,
                                  100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 144,
                                  148, 152, 156, 160, 164, 168, 172, 176, 180, 184, 188, 192,
                                  196, 200, 204, 208, 212, 216, 220, 224, 228, 232, 236, 240,
                                  244, 248, 252};
```

Scelto il valore della tonica, è possibile quantizzare l'uscita prodotta in funzione di un certo numero di scale, codificate come incremento rispetto alla fondamentale impostata, grazie a questi array:

```
//valori da addizzionare alla tonica per ottenere scale diverse
  const int scala_maggiore[37] = { 0,   8,  16,  20,  28,  36,  44,
                                   48,  56,  64,  68,  76,  84,  92,
                                   96, 104, 112, 116, 124, 132, 140,
                                  144, 152, 160, 164, 172, 180, 188,
                                  192, 200, 208, 212, 220, 228, 236,
                                  240, 248};

    const int scala_minore[37] = { 0,   8,  12,  20,  28,  32,  40,
                                   48,  56,  60,  68,  76,  80,  88,
                                   96, 104, 108, 116, 124, 128, 136,
                                  144, 152, 156, 164, 172, 176, 184,
                                  192, 200, 204, 212, 220, 224, 232,
                                  240, 248};

const int scala_aumentata[31] = { 0,  12,  16,  28,  32,  44,
                                  48,  60,  64,  76,  80,  92,
                                  96, 108, 112, 124, 128, 140,
                                 144, 156, 160, 172, 176, 188,
                                 192, 204, 208, 220, 224, 236,
                                 240};

    const int scala_cinese[27] = { 0,   8,  16,  28,  36,
                                   48,  56,  64,  76,  84,
                                   96, 104, 112, 124, 132,
                                  144, 152, 160, 172, 180,
                                  192, 200, 208, 220, 228,
                                  240, 248};

     const int scala_araba[37] = { 0,   4,  16,  20,  28,  32,  44,
                                   48,  52,  64,  68,  76,  80,  92,
                                   96, 100, 112, 116, 124, 128, 140,
                                  144, 148, 160, 164, 172, 176, 188,
                                  192, 196, 208, 212, 220, 224, 236,
                                  240, 244};
```

```
const int scala_orientale[37] = { 0,  4,  16,  20,  24,  36,  40,
                                  48,  52,  64,  68,  72,  84,  88,
                                  96, 100, 112, 116, 120, 132, 136,
                                 144, 148, 160, 164, 168, 180, 184,
                                 192, 196, 208, 212, 216, 228, 232,
                                 240, 244};

 const int scala_esatonale[32] = { 0,   8,  16,  24,  32,  40,
                                   48,  56,  64,  72,  80,  88,
                                   96, 104, 112, 120, 128, 136,
                                  144, 152, 160, 168, 176, 184,
                                  192, 200, 208, 216, 224, 232,
                                  240, 248};

const int scala_ottotonica[37] = { 0,   8,  12,  20,  24,  32,  36,
                                   48,  56,  60,  68,  72,  80,  84,
                                   96, 104, 108, 116, 120, 128, 132,
                                  144, 152, 156, 164, 168, 176, 180,
                                  192, 200, 204, 212, 216, 224, 228,
                                  240, 248};

const int scala_enigmatica[37] = { 0,   4,  16,  24,  32,  40,  44,
                                   48,  52,  64,  72,  80,  88,  92,
                                   96, 100, 112, 120, 128, 136, 140,
                                  144, 148, 160, 168, 176, 184, 188,
                                  192, 196, 208, 216, 224, 232, 236,
                                  240, 244};

    const int scala_blues[32] = { 0,  12,  20,  24,  28,  40,
                                  48,  60,  68,  72,  76,  88,
                                  96, 108, 116, 120, 124, 136,
                                 144, 156, 164, 168, 172, 184,
                                 192, 204, 212, 216, 220, 232,
                                 240, 248};
```

Un semplice esempio di algoritmo da implementare in un modulo affinché generi informazioni di Volt/ottava è riportato di seguito, per ragioni di praticità si è scelta un'unica scala e due potenziometri, ma il principio è estendibile a qualsiasi altra scala esistente o da codificare secondo gusto, accessibile con ogni tipologia di controllo che si ritenga opportuno. In questo caso, qualsiasi sia l'interazione esterna con il programma, si avrà la certezza di muoversi in 5 ottave, quantizzate secondo la scala esatonale.

```
const int tonica [12] = { 4,  8, 12, 16,
                          20, 24, 28, 32,
                          36, 40, 44, 48};

const int scala_esatonale[32] = { 0,   8,  16,  24,  32,  40,
                                  48,  56,  64,  72,  80,  88,
                                  96, 104, 112, 120, 128, 136,
                                 144, 152, 160, 168, 176, 184,
                                 192, 200, 208, 216, 224, 232,
                                 240, 248};
int indice_tonica;
 int indice_scala;
 int pwm_out;

void setup() {

  pinMode(11, OUTPUT); //out pwm quantizzata
}

void loop() {

 indice_tonica = map(analogRead(A0), 0, 1023, 0, 11);
  indice_scala = map(analogRead(A1), 0, 1023, 0, 31);

pwm_out = tonica[indice_tonica] + scala_esatonale[indice_scala];
 analogWrite(11, pwm_out);

}
```

Sarebbe insufficiente adottare un design del genere senza implementare un metodo di decodifica, necessario per tradurre queste informazioni in valori di frequenza interpretabili da

un oscillatore digitale. Il principio che si è ritenuto utile sviluppare è concettualmente molto simile a quello utilizzato dal protocollo MIDI:

$$\text{Frequenza} = 440 * 2 \wedge ((\text{segnale} * 0.25 - 136) / 48) \qquad (\textit{Equazione 3.2})$$

Dove:

- 440 sono gli Herz corrispondenti al LA centrale del pentagramma
- segnale * 0.25 è l'informazione letta dal pin analogico e riscalata da 0-1023 a 0-255
- 136 è il valore corrispondente al LA secondo il protocollo che si è costruito[82], ciò permette di poter spaziare da 61.74 a 2453 Hz, dominio ulteriormente estendibile via software
- 48 è il range di valori con il quali si è scelto di esprimere un'ottava, da cui è quindi possibile ottenere al massimo micro-intervalli di un quarto di semitono

Di seguito è presentato un esempio della logica con cui un modulo è capace di acquisire l'informazione di Volt/ottava da un ingresso dedicato; viene inoltre sfruttata la funzione *tone* di Arduino per generare un'onda quadra, con duty cycle fisso al 50%, in banda audio.

```
int in_volt_ott; //lettura da ingresso
 double frequenza; //valore di frequenza
  double arr_cambio_ott [5] = {0.25, 0.5, 1.00, 2.00, 4.00};
   int indice_ott;

void setup() {
  pinMode(11, OUTPUT); //out segnale
}

void loop() {

// leggo dall'ingresso dedicato per volt/ott
  in_volt_ott = analogRead(A0) * 0.25;
   indice_ott = map(analogRead(A1), 0, 1023, 0, 4);

//converto il valore letto nella frequenza corrispettiva
   frequenza = (double) (440.00 * pow(2, (in_volt_ott - 136.00) / 48.00))
                        * arr_cambio_ott[indice_ott]; //posso cambiare software l'ottava
//funzione per generare onda quadra con 50% duty cycle
  tone(11, frequenza);
}
```

3.5 Arduino e sintesi sonora: la libreria Mozzi

Tutti i segnali descritti fino a questo punto riescono a formalizzare un linguaggio di comunicazione funzionale alla gestione di un sistema modulare, agevolmente implementabile in ambiente Arduino; nonostante i vantaggi nel generare informazioni di controllo, non è trascurabile l'ostacolo più grande introdotto dalla scelta del Pro Mini: la sintesi di segnali in

[82] Come osservabile nel primo array presentato in questo sotto-paragrafo

banda audio risulta necessariamente limitata dalle possibilità tecniche della scheda, sia dal punto di vista hardware che software. Il problema principale è relativo all'assenza di un convertitore digitale-analogico dedicato, a meno di non voler sintetizzare unicamente onde quadre o introdurre moduli DAC esterni, l'unico approccio possibile è sfruttare un pin PWM agendo sui timer interni di Arduino; in questo paragrafo non verranno approfondite le tecniche di programmazione utili a questo scopo, ma saranno descritti i principi di funzionamento di una libreria che crea un alto livello di astrazione tra il software e il musicista, permettendo di concentrarsi unicamente sugli aspetti artistici legati alla scrittura del codice.

A partire dal 2012, Tim Barrass sviluppa Mozzi[83], una libreria opensource che fornisce ad Arduino tutti gli elementi fondamentali alla gestione e produzione sonora, integrando funzioni intuitive e di semplice implementazione per ottenere oscillatori, inviluppi, LFO, riproduzione di campioni, DSP (filtri, linee di ritardo e riverberi) e algoritmi di sintesi (waveshaping, modulazione in ampiezza, frequenza e fase); nonostante esistano numerose possibilità valide, si è ritenuto che questa libreria sia il giusto compromesso tra risultato sonoro ottenibile e facilità di utilizzo. Mozzi garantisce frequenza di campionamento a 16384 Hz, sfruttando un pin PWM per ottenere segnali con profondità di 8 bit: la qualità audio è piuttosto lontana dal poter essere definita accettabile rispetto agli standard digitali moderni, ma è quanto di meglio si è riusciti a ottenere da un Arduino Pro Mini sprovvisto di DAC; in ogni caso, nell'ultima sezione di questo paragrafo verranno descritti alcuni accorgimenti hardware da adottare per migliorare la qualità del segnale in uscita.

3.5.1 L'anatomia di Mozzi

Mozzi agisce direttamente sui timer della scheda, motivo per cui Arduino non segue la sua normale routine di funzionamento ed è necessario tenere presente i seguenti cambiamenti:

- richiamare funzioni che implicano cicli o loop introdurrà artefatti nell'audio in uscita, sono soprattutto da evitare delay() e delayMicroseconds(), sostituite da EventDelay()
- la funzione analogRead() bloccherebbe l'esecuzione del codice principale, è quindi rimpiazzata da mozziAnalogRead()

[83] https://sensorium.github.io/Mozzi/

- la libreria interferisce con i timer, nello specifico sul Timer1 in modalità STANDARD (pin digitali 9 e 10) e anche sul Timer2 (pin 3 e 11) in modalità HIFI. Su questi pin sarà impossibile richiamare la funzione analogWrite()
- le funzioni millis() e micros() sono disabilitate, sostituite da mozziMicros()
- l'utilizzo del monitor seriale può causare artefatti nell'audio prodotto

L'interfaccia tra Mozzi e l'ambiente Arduino è garantita da 4 funzioni principali, visibili nel seguente template standard di utilizzo della libreria:

```
#include <MozziGuts.h> //includo la libreria Mozzi
#define CONTROL_RATE 64 // frequenza dei segnali di controllo, può essere maggiore di 64
                        // mantenere una potenza di 2

void setup() {
  startMozzi(CONTROL_RATE); //attiva Mozzi
}

void updateControl() {
  //funzione deputata all'aggiornamento delle variabili
  //e alla lettura dai pin analogici e digitali
}

int updateAudio() {
 //aggiornamento del segnale audio, restituisce un intero compreso tra -244 e 243
 //può essere utilizzata una variabile di tipo char
}

void loop() {
  audioHook(); //funzione che riempie il buffer audio
}
```

- startMozzi(CONTROL_RATE) va richiamata nel setup() e permette l'attivazione della gestione dei timer per segnali audio e di controllo
- updateControl() è dove vengono aggiornate le variabili associate alla lettura dei pin digitali e analogici, oltre a tutti i possibili cambiamenti relativi a segnali non in banda audio, come LFO e inviluppi
- updateAudio() è la funzione deputata alla gestione degli algoritmi di sintesi audio, si aggiorna mediamente 16384 volte al secondo, richiede una variabile in uscita necessariamente compresa tra -244 e 243 nella modalità STANDARD o tra -8192 e 8191 nella modalità HIFI
- audioHook() è l'unica funzione da richiamare all'interno del loop() di Arduino. Decodifica il buffer riempito dalla funzione updateAudio() e permette la generazione del segnale in uscita dal pin digitale 9

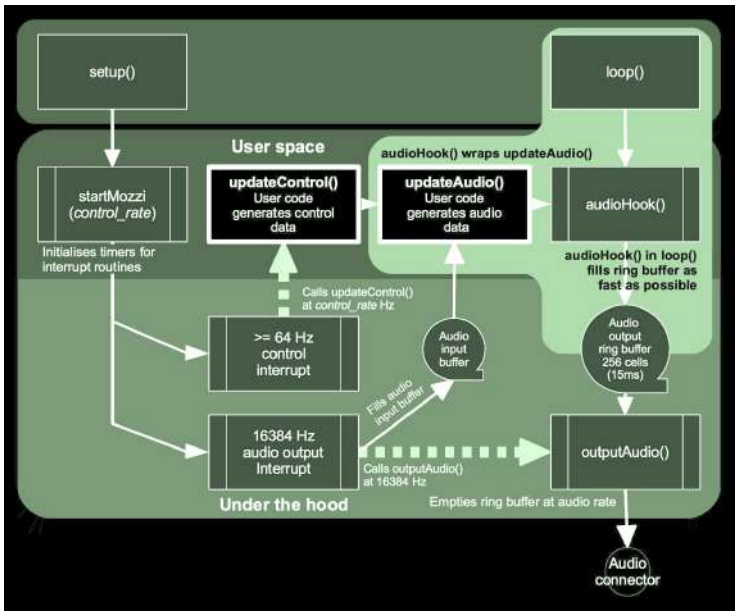


Figura 3.4 Schema a blocchi del principio di funzionamento della libreria Mozzi (Tim Barrass - *https://sensorium.github.io/Mozzi/learn/under-the-hood/)*

Per garantire una produzione audio scevra di artefatti indesiderati è utile mantenere il codice quanto più snello possibile, grazie ai seguenti accorgimenti:

- eseguire tutti i calcoli necessari, quando possibile, all'interno delle funzioni di setup() e updateControl(); le funzioni loop() e updateAudio() devono rimanere agili
- evitare le divisioni
- utilizzare, dove possibile, numeri in potenza di 2 e l'aritmetica di bit-shifting al posto delle moltiplicazioni
- dichiarare variabili e costanti con tipi necessari al proprio scopo, se possibile non segnati
- evitare numeri a virgola mobile
- utilizzare il monitor seriale solo in fase di debugging e all'interno della funzione updateControl()

3.5.2 Accorgimenti hardware: circuiti di uscita audio

Una migliore resa sonora è ottenibile utilizzando Mozzi in modalità HIFI con il seguente circuito all'uscita dei pin digitali 10 e 9[84]:

[84] Dual PWM: http://www.openmusiclabs.com/learning/digital/pwm-dac/dual-pwm-circuits/index.html

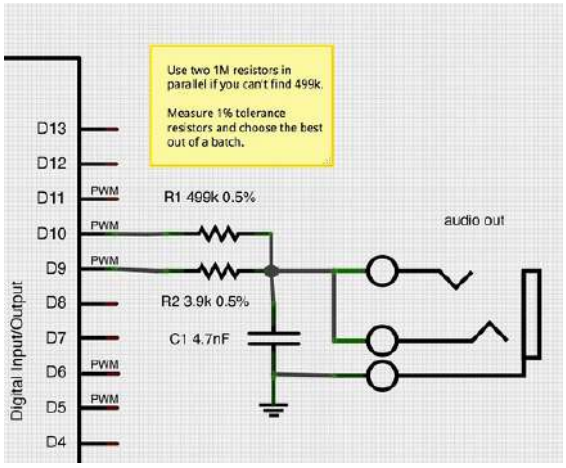
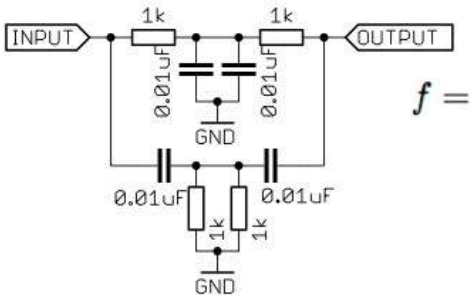


Figura 3.5 Circuito di uscita per la modalità HIFI (Tim Barrass - *https://sensorium.github.io/Mozzi/learn/output/)*

Qualora non si volesse tenere occupato il secondo timer della scheda, perché necessario ad altri processi, va considerato che in modalità STANDARD la frequenza portante della PWM ricade in banda audio; un filtro notch passivo[85] come quello proposto di seguito attenua la porzione di spettro indesiderata.



$$f = \frac{1}{2\pi RC} = \frac{1}{2\pi 10^3 10^{-8}} \approx 15.9\text{kHz}$$

Figura 3.6 Filtro notch per la modalità STANDARD della libreria (Andrew McPherson - *https://sensorium.github.io/Mozzi/learn/output/)*

In entrambe le modalità, a causa dell'aliasing, sono riscontrabili frequenze indesiderate nella parte acuta dello spettro; un semplice filtro RC passa-basso passivo[86] migliora sensibilmente la qualità del segnale audio prodotto.

[85] https://sim.okawa-denshi.jp/en/TwinTCRtool.php

[86] https://sim.okawa-denshi.jp/en/PWMtool.php

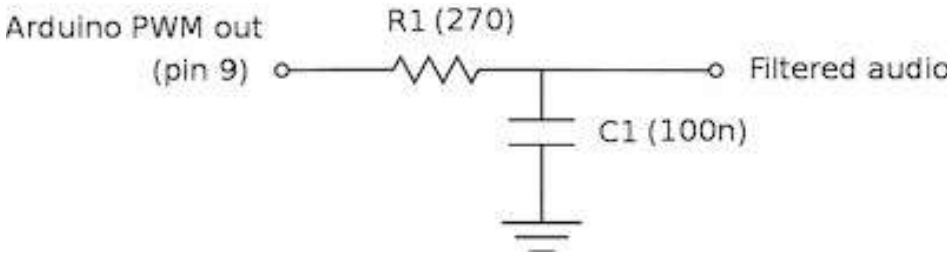


Figura 3.7 Filtro RC passa-basso con frequenza di taglio intorno ai 6 kHz. Per la modalità HIFI è posto dopo il nodo che combina i due segnali provenienti dai pin digitali. (Tim Barrass - *https://sensorium.github.io/Mozzi/learn/output/)*

3.6 Design interfaccia strumentista

Quanto descritto finora rappresenta il metodo utilizzato dal sistema per gestire e produrre suono, in questo ultimo paragrafo verrà esposta la logica progettuale con cui si è scelto di definirne l'aspetto fisico: come ogni altro strumento musicale deve agevolare quanto più possibile l'esecuzione rispetto alle intenzioni dello strumentista e al risultato sonoro potenzialmente ottenibile, con il vantaggio che l'emissione sonora non è acustica, quindi assolutamente svincolata dalle dimensioni fisiche del sistema. Piuttosto che definire dimensioni arbitrarie si è ritenuto vantaggioso basarsi su uno standard costruttivo già esistente in ambito modulare, così da poter eventualmente integrare i moduli progettati personalmente all'interno di strumenti già presenti sul mercato e viceversa.

3.6.1 Lo standard eurorack

Il formato eurorack si basa sul sistema dimensionale rack internazionale da 19 pollici[87], introdotto in ambito musicale da Dieter Doepfer alla fine degli anni Ottanta: le unità previste dallo standard definiscono in U[88] l'altezza del pannello frontale del modulo e in HP[89] la sua larghezza.

[87] DIN 41494 / IEC 297-3 / IEEE 1001.1

[88] 1 U = 7/4 di pollice = 44.45 mm

[89] 1 HP = 1/5 di pollice = 5.08 mm

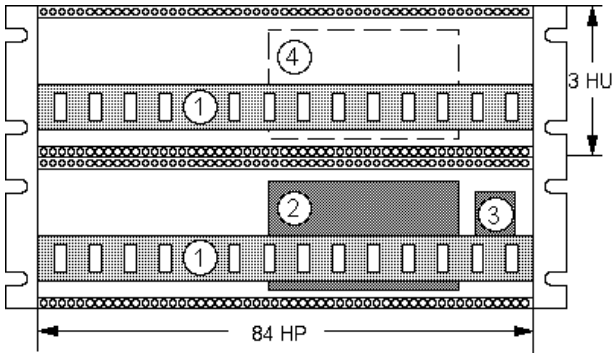


Figura 3.8 Esempio di un telaio di montaggio che rispetta lo standard eurorack. I numeri cerchiati fanno riferimento a elementi deputati all'alimentazione del sistema, argomento non considerato in questo lavoro di tesi. (*http://www.doepfer.de/a100_man/a100m_e.htm*)

Tra due *rail*, i binari dove vengono fissati i moduli, è definita un'altezza fissa di 3 U (133.4 mm) e considerando il loro spessore l'altezza per modulo è pari a 128.5 mm; la larghezza in HP (Horizontal Pitch) è invece variabile secondo necessità, avendo come riferimento il seguente schema:

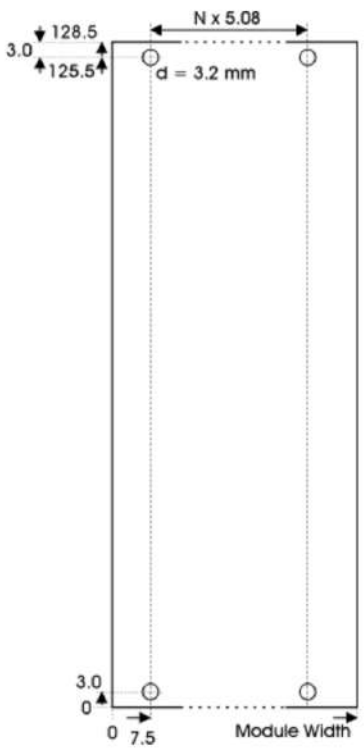
Position of the mounting holes	Module width [HP]	calculated module width [mm] (= multiples of 5.08 mm)	actual module width [mm]
	1	5.08	**5.00**
	1.5	7.62	**7.50**
	2	10.16	**9.80**
	4	20.32	**20.00**
	6	30.48	**30.00**
	8	40.64	**40.30**
	10	50.80	**50.50**
	12	60.96	**60.60**
	14	71.12	**70.80**
	16	81.28	**80.90**
	18	91,44	**91,30**
	20	101.60	**101.30**
	21	106.68	**106.30**
	22	111.76	**111.40**
	28	142.24	**141.90**
	42	213.36	**213.00**

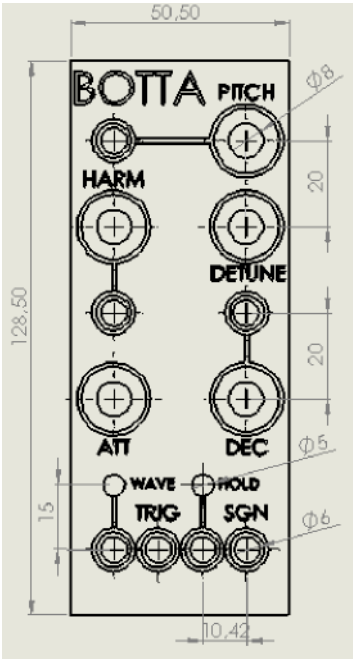
Figura 3.9 Direttive dimensionali per il design di un pannello eurorack. (*http://www.doepfer.de/a100_man/a100m_e.htm*)

I fori di fissaggio sono usualmente 2, uno per rail, per i moduli che non eccedono i 10-12 HP, altrimenti è consigliato prevederne almeno 4. Le viti di bloccaggio sono a testa ovale con incavo a croce M3x6[90] e relativo dado filettato.

Il materiale utilizzato per i pannelli è indicato come alluminio anodizzato da 2 mm di spessore; personalmente si è invece optato per un multistrato in legno di pioppo da circa 4 mm, così da assecondare gusto estetico e maggiore facilità di lavorazione.

Oltre alle dimensioni fornite dallo standard si è ritenuto valido definire esplicitamente, a partire da prove empiriche, le distanze minime tra ogni controllo presente sul pannello per agevolare quanto più possibile l'interazione manuale:

- la distanza tra il centro del foro di un potenziometro (diametro 8 mm) e il centro di qualsiasi altro componente deve essere al minimo pari a 20 mm
- la distanza tra il centro del foro di un jack femmina (diametro 6 mm) e il centro di un altro ingresso jack deve essere al minimo pari a 10 mm
- la distanza tra il centro del foro di un pulsante (diametro 5 mm) e il centro di un altro pulsante o jack deve essere al minimo pari a 15 mm
- le zone del pannello adiacenti ai rail non devono essere ingombrate da componenti per almeno 10 mm



Sebbene non sia un obbligo dettato dallo standard, la maggior parte dei moduli e dei telai di montaggio in commercio sono proporzionati per un numero pari di HP; a meno di non inserire due moduli con HP dispari dello stesso valore, per saturare completamente e con certezza lo spazio disponibile di un rail si è scelto di dimensionare ogni modulo affinché abbia HP in multipli di 2.

[90] DIN7985

3.6.2 Agevolare l'esecuzione dal vivo

Ogni modulo ha mediamente una dozzina di parametri che possono o devono essere controllati in tempo reale dallo strumentista, associati a potenziometri, pulsanti, interruttori e soprattutto collegamenti jack con le altre unità; il sistema progettato è formato da più di 20 moduli diversi e costituisce perciò una mole notevole di interazioni che il musicista deve poter gestire intuitivamente e senza ostacoli, con la possibilità di memorizzare facilmente posizione e scopo di ogni controllo. Il primo accorgimento in questo verso è relativo al layout di ogni pannello, dove ogni funzione ha una parola chiave che la identifica in modo univoco e simboli grafici che la connettono ad altri parametri logicamente associabili a essa. Si è inoltre cercato di discretizzare cromaticamente le manopole di ogni potenziometro, differenziandole per scopo e usando lo stesso colore per controlli simili presenti su più moduli: per gli oscillatori, ad esempio, il rosso è correlato alla variazione del pitch e il bianco alla modifica timbrica.

Un altro fattore di feedback visivo è rappresentato da led colorati collegati in parallelo alla quasi totalità delle uscite di ogni modulo, così da permettere allo strumentista di interpretare rapidamente le informazioni prodotte prima di inviarle ad altre unità senza avere la necessità di un riscontro sonoro.

Questo è l'esempio di layout di un modulo deputato a eseguire operazioni booleane su due segnali binari in ingresso. Il funzionamento specifico dell'unità verrà descritto nel prossimo capitolo, ora è da osservare come si è cercato di assecondare in maniera intuitiva il percorso del segnale: dagli input superiori, le informazioni binarie vengono valutate rispetto a una soglia, impostabile da potenziometro, restituendo 1 o 0; quindi i primi due output sono relativi all'inversione di questa condizione (!A, !B). Le restanti 6 uscite combinano le due informazioni secondo gli operatori booleani indicati sul pannello mentre il potenziometro etichettato con *ruota* permette di variare il routing di ogni segnale sulle varie uscite; come si può notare dal segmento grafico, a questo controllo è associato un ingresso per la variazione automatica tramite CV. Per garantire l'interpretazione visiva dei segnali prodotti, ogni uscita è dotata di un led adiacente al rispettivo jack di uscita.

3.6.3 Trasportabilità

Un altro punto che si è ritenuto considerare durante la progettazione è relativo alla trasportabilità dello strumento, in modo che peso e dimensioni non lo releghino al solo uso in studio, ma ne favoriscano la mobilità per le esecuzioni dal vivo. Vista la natura e conformazione dello strumento, è lo stesso telaio di fissaggio a fare da involucro di trasporto e quanti più moduli si sceglie di voler inserire all'interno, tanto maggiore sarà il volume necessario da considerare. In questo senso si è scelto di costruire due case: il primo ha capienza maggiore (18 U con 100 HP per binario, per un totale di 600 HP) a discapito di una trasportabilità non favorevole in determinate situazioni; il secondo telaio è più piccolo (6 U con 84 HP a binario e 168 HP complessivi), riesce a ospitare meno moduli, ma acquista un ottimo grado di mobilità e riesce a rientrare, ad esempio, nelle specifiche della maggior parte delle compagnie aeree per essere imbarcato come bagaglio a mano. In particolare, questo case è stato disegnato in modo che il coperchio, una volta chiuso, non interferisca con il routing tra i cavi dei moduli, così che lo strumentista abbia la facoltà di trasportare lo strumento con una patch già impostata.

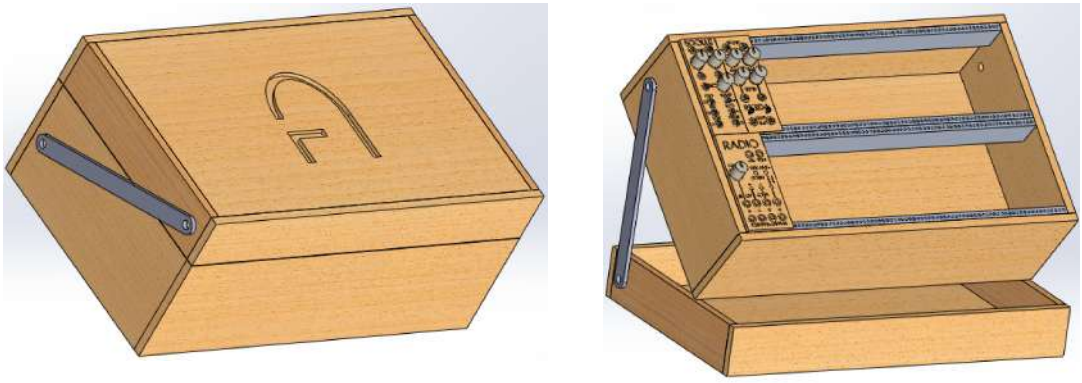


Figura 3.10 Design in 3D del case più piccolo da 168 HP. Il sistema di apertura e chiusura è progettato in modo che il coperchio possa fungere da base della struttura, la cui inclinazione è quindi regolabile in funzione delle necessità dello strumentista.

4 RUMORE BINARIO

Legge di Hofstadter: Per fare una cosa ci vuole sempre più tempo di quanto si pensi, anche tenendo conto della Legge di Hofstadter

In questo capitolo sono descritte le funzioni specifiche di ogni modulo, principalmente soffermandosi sullo scopo musicale cui è deputata ogni unità e sul ruolo che svolge rispetto alla totalità del sistema; la divisione in paragrafi asseconda la tipologia di informazioni prodotte, individuando, per ogni modulo, otto categorie principali di appartenenza.

Figura 4.1 Design del sistema completo *Rumore Binario*

4.1 Generazione ed elaborazione di informazioni temporali

I moduli appartenenti a questa famiglia sono progettati per produrre ed elaborare segnali di clock, garantendo il corretto funzionamento temporale dell'intero sistema e, se desiderato o necessario, il sincronismo tra le azioni di più unità.

4.1.1 Orologio

Orologio è il master clock dello strumento, ovvero la prima e principale sorgente di informazioni sulla gestione del tempo: tutte le unità che necessitano di una scansione temporale si trovano in una posizione subordinata rispetto a questo modulo, secondo un rapporto di master e slave. Orologio è l'unico modulo che prevede di usare delay() per tempi maggiori di 15 millisecondi, sgrava tutte le altre unità del sistema dalle latenze associate alla finestra di buio introdotta da questa funzione e favorisce, quindi, il massimo grado di reattività allo strumento.

Il modulo ha dimensioni di 8 HP e assorbe circa 65 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

0) Schermo LCD per visualizzare informazioni di BPM e figurazione ritmica impostate dai pulsanti

1) Pulsante decremento BPM (-1 BPM)
2) Pulsante incremento BPM (+1BPM)
3) Pulsante FINE BPM, se premuto in combinazione con + o - permette una regolazione più precisa del clock (+/- 0.25 BPM)
4) Pulsante RITMO, se premuto in combinazione con + o - permette di cambiare la figurazione ritmica prodotta in uscita
5) Premendo contemporaneamente RITM e FINE e agendo sui pulsanti + e -, è possibile impostare con più scarto la velocità di variazione dei BPM (+/- 10 BPM)
6) Ingresso trigger o gate esterno per variare casualmente la figurazione ritmica
7) Pulsante RND per variazione figura ritmica secondo un principio basato sulla passeggiata aleatoria
8) Indicatore led del clock prodotto
9) Output per il segnale di clock

Impostato il BPM desiderato è dunque possibile variare a piacere la figura ritmica prodotta in uscita, moltiplicando i millisecondi di attesa tra un trigger e il successivo per i seguenti fattori: 6, 4, 3, 2.666667, 2, 1.5, 1.333333, 1, 0.75, 0.6666667, 0.5, 0.375, 0.333333, 0.25, 0.1875, 0.166667, 0.125, 0.09375, 0.083333, 0.0625.

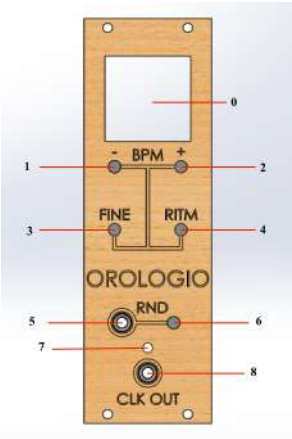


Figura 4.2 Design del pannello per il modulo Orologio

4.1.2 Tappo

Tappo rappresenta l'alternativa a Orologio in contesti di performance dal vivo con altri strumentisti, laddove, non avendo precise informazioni di BPM, un classico master clock risulterebbe macchinoso o inservibile. Questo modulo rende disponibile all'esecutore la funzione di *tap tempo* e permette di rendere ciclici più trigger consecutivi provenienti da altre unità. Il modulo ha dimensioni di 6 HP e assorbe circa 25 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Output segnale clock coerente con l'intervallo di interazione del pulsante o trigger
2) Ingresso trigger esterno
3) Pulsante per il tap tempo manuale, sono necessarie almeno due interazioni

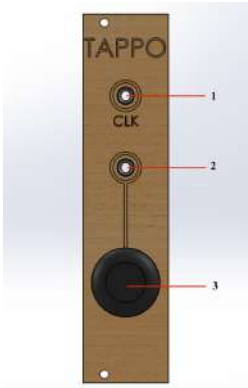


Figura 4.3 Design del pannello per il modulo Tappo

4.1.3 Ingranaggi

Ingranaggi è il clock multiplier dello strumento: accetta un'informazione di clock in ingresso, ne analizza la ciclicità, e produce in uscita 8 segnali la cui frequenza è un suo multiplo pari o dispari. Il modulo è stato progettato ispirandosi idealmente a un insieme di ingranaggi con specifici rapporti tra i relativi diametri e messi in rotazione da un albero comune, dove a ogni velocità angolare è associata una diversa frequenza di clock in output; questo modello ha agevolato l'implementazione di una funzione di rotazione tra i segnali generati, rendendo possibile la manipolazione in tempo reale del ruoting logico di ogni clock rispetto alle uscite fisiche del modulo. Sebbene garantisca risultati precisi solo entro un certo limite frequenziale, è anche possibile presentare all'input un'onda quadra in banda audio, ottenendo in uscita le relative 8 armoniche superiori.

Il modulo ha dimensioni di 6 HP e assorbe circa 60 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Ingresso clock esterno da analizzare
2) Ingresso CV per modificare automaticamente il routing logico delle uscite
3) Potenziometro per la modifica manuale del routing logico delle uscite
4) Matrice dei clock prodotti in uscita, con led associati per feedback visivo

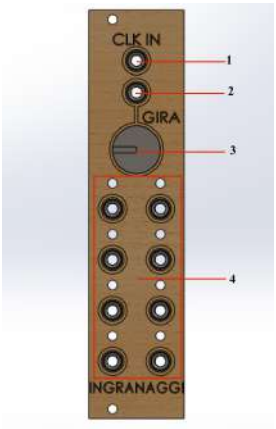


Figura 4.4 Design del pannello per il modulo Ingranaggi

4.1.4 Battiti

Battiti è la controparte di Ingranaggi, deputata alla decelerazione di segnali temporali: analizza un clock in ingresso e ne restituisce 6 versioni aventi frequenze ridotte secondo divisori pari. Anche in questo caso, con risultati spesso imprecisi, è possibile introdurre in input un'onda quadra nello spettro udibile per ottenere le relative sub-armoniche pari. Il

modulo è fornito di un input per azzerare l'algoritmo di suddivisione interno, così da agevolare la creazione di ritmi irregolari e in *levare* rispetto al clock ricevuto.

Il modulo ha dimensioni di 4 HP e assorbe circa 35 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Input RST per azzerare il conteggio delle sottodivisioni
2) Input per clock esterno da analizzare
3) Uscite per i segnali di clock suddivisi, con led associati

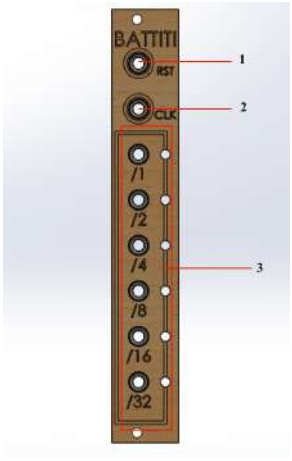


Figura 4.5 Design del pannello per il modulo Battiti

4.2 Generazione di informazioni di controllo

Questo insieme di moduli rappresenta la principale fonte di control voltages del sistema, grazie alle quali è possibile ottenere variazioni automatiche dei parametri di ogni altra unità, così come l'innesco di funzioni ed eventi sonori: è qui che vengono prodotti la maggior parte dei trigger, gate, CV e informazioni di Volt/ott che circolano all'interno dello strumento, motivo per cui rappresenta una sezione cardine nell'ottica della musica generativa.

4.2.1 Trigga

Trigga è un sequencer di trigger e gate a 16 step, programmabile per gestire polimetricamente e poliritmicamente 8 linee temporali: ogni sequenza può essere pilotata singolarmente da clock esterni e un numero qualsiasi di step può essere disattivato per ogni linea.

Il modulo ha dimensioni di 30 HP e assorbe circa 95 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Switch per selezione modalità clock interno o esterno. Nel primo caso tutte le linee avranno lo stesso clock
2) Ingresso clock esterno per pilotare tutte le linee contemporaneamente
3) Potenziometro per impostare BPM in modalità di clock interno, o per suddividere la frequenza del clock esterno presente all'ingresso ALL
4) Ingressi dedicati ai clock esterni, per coppie di due linee
5) Potenziometro per impostare il numero di step soggetti alle funzioni randomiche
6) Uscita clock per la modalità clock interno
7) Uscita SYNC, questa funzione restituisce un gate positivo ogni volta che tutte le linee si trovano sullo stesso step, cioè quando le linee, procedendo polimetricamente, ritornano in fase. Se il clock è comune e nessuno step è stato disattivato, genererà un segnale costantemente alto
8) Ingresso RST per far ritornare tutte le linee allo step iniziale
9) Uscite per i trigger o gate di ogni linea
10) Potenziometro per impostare la durata del gate, minimo 15 millisecondi (trigger) e massimo 5000 millisecondi
11) Pulsante RND che, associato a uno dei pulsanti per la selezione delle linee, varia casualmente il valore degli step (0 o 1) del pattern selezionato
12) Pulsante MUT, per silenziare l'uscita della linea selezionata
13) Pulsante REV, inverte il valore logico degli step del pattern selezionato
14) Pulsante FNC, per accedere a funzioni diverse combinandolo con altri tasti
15) Pulsanti per la selezione e visualizzazione dei singoli pattern o per le funzioni
16) Led RGB per feedback visivo sui controlli impostati, ogni pattern ha un colore dedicato
17) Pulsanti per impostare il valore dello step o per attivarlo e disattivarlo
18) Pulsante START/STOP per attivare o disattivare il sequencer

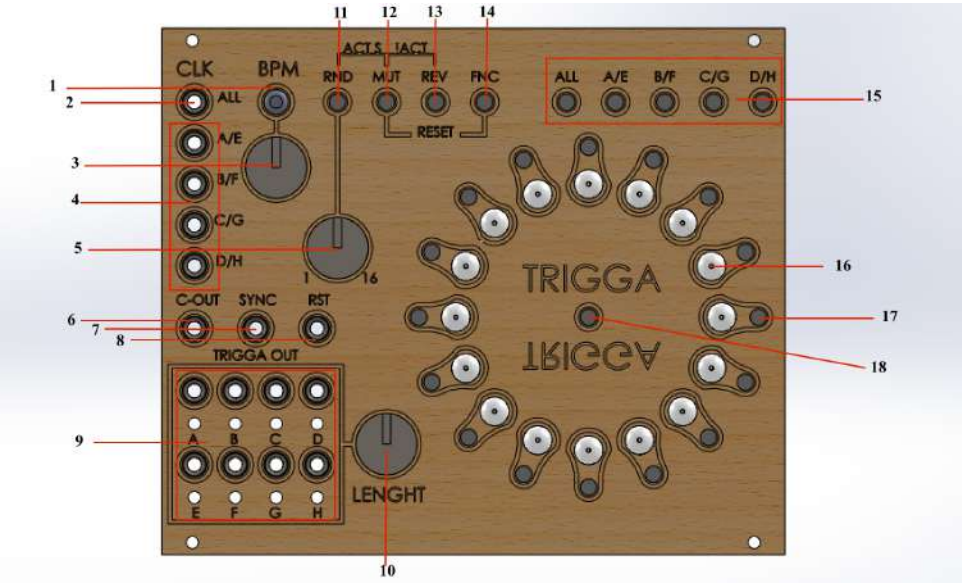


Figura 4.6 Design del pannello per il modulo Trigga

4.2.2 Cartesio

Cartesio è un sequencer a 16 step per la generazione di segnali di controllo, la cui principale peculiarità risiede nella modalità cartesiana di gestione delle informazioni; tutta la struttura software e hardware è sviluppata intorno a una matrice 4x4: 16 potenziometri permettono di impostare il valore da associare a ogni step (0 - 255), alla cui posizione è possibile accedere grazie a 2 ingressi clock indipendenti. Il modulo è stato infatti progettato pensando a un sistema bidimensionale di assi cartesiani, un clock fa avanzare da 0 a 3 il valore ascissa e l'altro sull'ordinata; la combinazione dei due clock, quindi delle due coordinate, permette di accedere a ognuna delle 16 posizioni della matrice, restituendo in output il valore impostato sullo step dal potenziometro.

L'unità è inoltre fornita di una tastiera a 16 tasti per accedere manualmente a ogni step, con uscite gate dedicate e protocollo Volt/ott: il modulo è quindi utilizzabile anche come tastiera, con la possibilità di impostare il valore di pitch di ogni tasto e la scala musicale in cui si preferisce agire.

Il modulo ha dimensioni di 46 HP e assorbe circa 105 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Incrementa coordinata X da trigger esterno
2) Decrementa coordinata X da trigger esterno
3) Varia la modalità di ricezione trigger dell'ascissa
4) Incrementa coordinata Y da trigger esterno

5) Decrementa coordinata Y da trigger esterno
6) Varia la modalità di ricezione trigger dell'ordinata
7) Azzera la posizione dello step al punto di coordinate (0, 0), corrispondente al potenziometro posto in basso a sinistra
8) Varia randomicamente la tonica dell'uscita quantizzata
9) Varia randomicamente la scala dell'uscita quantizzata
10) Attiva e disattiva il sequencer con trigger esterno
11) Uscite dei gate corrispondenti a ogni posizione della matrice
12) Led RGB per feedback visivo, ogni tonica e scala ha un colore dedicato
13) Matrice dei potenziometri per impostare il valore dello step
14) Uscita CV per il valore di tensione corrispondente allo step attualmente selezionato
15) Uscita CVQ per il protocollo Volt/ott
16) Uscita gate generale
17) Pulsante Play/Stop per attivare e disattivare la sequenza
18) Pulsante FCN per accedere alle funzioni extra associato a ogni tasto
19) Pulsante per mutare/saltare uno step
20) Pulsante per selezionare la scala/tonica
21) Pulsante per variare la modalità di ricezione per l'asse X/Y
22) Pulsanti per accedere manualmente a ogni step, per variare la scala, la tonica o la modalità di ricezione dei trigger esterni per l'ascissa e l'ordinata

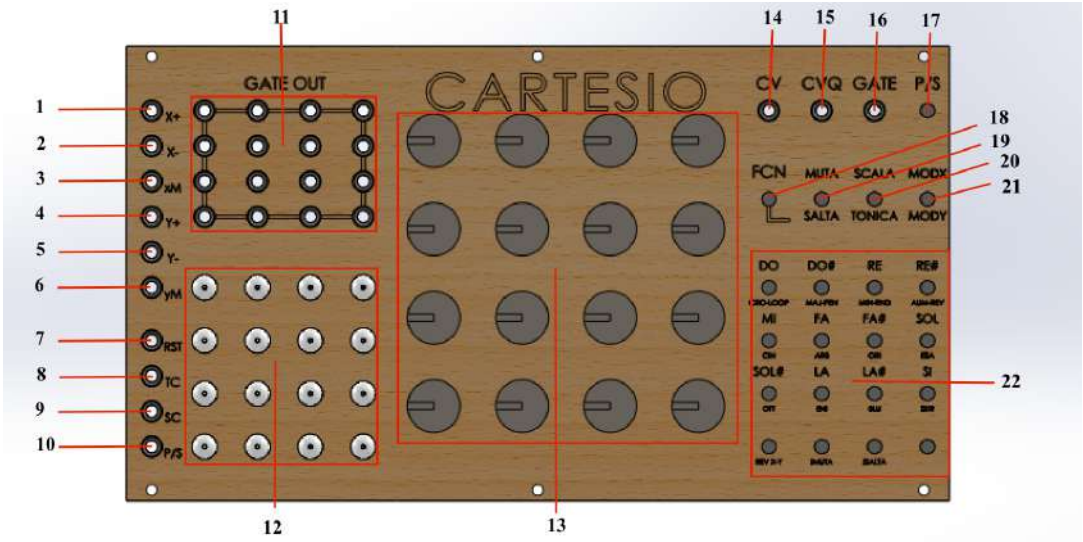


Figura 4.7 Design del pannello per il modulo Cartesio

4.2.3 Block

Block è un semplice generatore di segnali randomici a gradino e, non avendo una particolare logica produttiva, rappresenta la fonte di massima casualità presente all'interno dello strumento. Il modulo può sintetizzare 3 segnali casuali a partire da trigger esterni o secondo un proprio clock interno, in ambedue i casi con la capacità di entrare in banda audio per ottenere rumore digitale. Il modulo ha dimensioni di 12 HP e assorbe circa 35 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Switch per impostare il range del clock interno
2) Ingresso clock esterno
3) Potenziometro frequenza clock interno o suddivisioni clock esterno
4) Potenziometro per impostare il valore casuale massimo producibile dall'uscita A
5) CV per impostare il valore massimo di A
6) Uscita clock casuale
7) Uscita segnale casuale A
8) Uscita segnale casuale compreso tra il massimo impostato dal potenziometro A e B
9) Switch per selezione clock interno/esterno

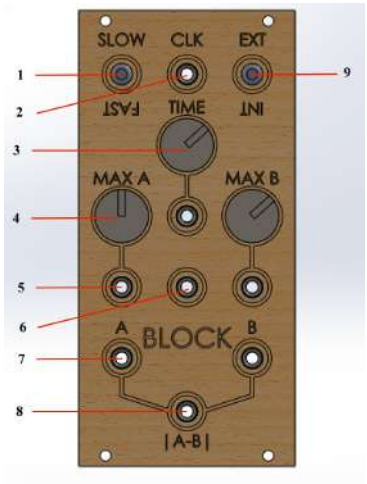


Figura 4.8 Design del pannello per il modulo Block

4.2.4 Tocca

Tocca è una tastiera capacitiva pensata con la specifica intenzione di agevolare lo strumentista nell'esecuzione dal vivo, è infatti uno dei pochi moduli del sistema che non prevede modifiche del proprio comportamento da parte di altre unità, ma solo tramite interazione umana. L'interfaccia è composta da 4 pulsanti con uscite gate indipendenti, associati a

generatori di CV il cui valore è determinato dalla posizione di 4 potenziometri; è inoltre presente un controllo per impostare la sensibilità di attivazione di ogni tasto, così da rendere possibile un controllo della tastiera senza contatto, con distanza della mano dalla superficie anche superiore ai 5 centimetri.

Il modulo ha dimensioni di 14 HP e assorbe circa 40 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Uscite gate per i singoli tasti premuti e uscita gate generale
2) Tastiera capacitiva
3) Potenziometro sensibilità tastiera
4) Uscita segnale di controllo
5) Uscita segnale Volt/ott
6) Potenziometri associati ai tasti per impostare il valore del segnale di controllo in uscita

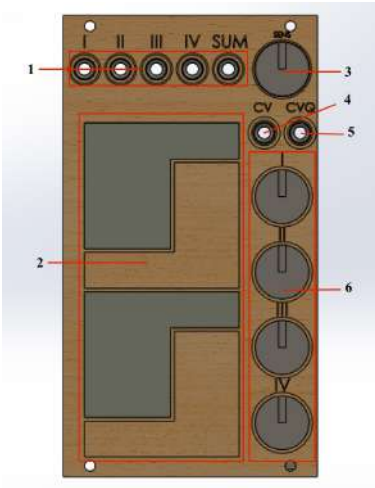


Figura 4.9 Design del pannello per il modulo Tocca

4.3 Manipolazione di informazioni

Questa sezione dello strumento è deputata alla ricezione di informazioni e alla loro conseguente trasformazione in altre forme di segnali; nessuno dei seguenti moduli possiede algoritmi interni di generazione e i propri output dipendono unicamente dall'elaborazione dei dati provenienti da altre unità.

4.3.1 Boolle

Il modulo *Boolle* agisce su due segnali continui in ingresso, li converte in segnali binari con valore logico unitario se superano una certa soglia e quindi applica i classici operatori

dell'algebra booleana per produrre 8 output distinti. L'unità permette inoltre di variare in tempo reale il routing logico di ogni segnale prodotto sulle uscite fisiche; grazie ai due potenziometri soglia è possibile utilizzare, con l'opportuna regolazione, una delle due linee per impostare un valore costante senza avere necessariamente un secondo segnale in ingresso. Il modulo ha dimensioni di 8 HP e assorbe circa 30 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Input segnali, possono essere binari o continui
2) Potenziometri per impostare il valore oltre il quale il segnale assume valore logico unitario
3) Uscite per segnale logico invertito
4) Controllo manuale ed esterno per ruotare il routing delle uscite
5) Uscite per i valori prodotti dagli operatori booleani, con led associati per feedback visivo

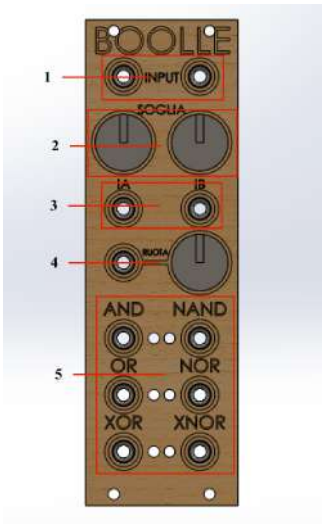


Figura 4.10 Design del pannello per il modulo Boolle

4.3.2 Rikeda

Rikeda è un convertitore probabilistico di trigger in gate: un impulso in ingresso ha una certa possibilità di essere presente all'output sotto forma di gate, con lunghezza massima di 5000 millisecondi. Il modulo ha dimensioni di 8 HP e assorbe circa 35 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Ingresso trigger da convertire in gate
2) Potenziometri per impostare la durata del segnale in uscita, da 15 a 5000 millisecondi
3) Potenziometri e ingressi CV esterni per impostare la probabilità di generare un segnale in uscita, da 6.25% a 100%

4) Ingressi CV esterni per variare automaticamente la lunghezza del segnale prodotto in uscita
5) Uscite per i gate, con led associati

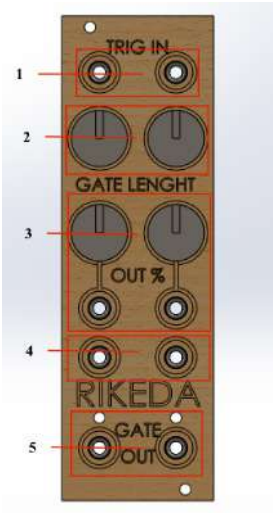


Figura 4.11 Design del pannello per il modulo Rikeda

4.3.3 Taighete

Taighete svolge la funzione opposta rispetto a Rikeda: accetta come input un gate e trasforma la sua fase positiva in uno o più trigger consecutivi, distanziati nel tempo secondo un segnale di clock interno o esterno. Il modulo ha dimensioni di 4 HP e assorbe circa 25 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Ingresso gate esterno
2) Pulsante gate manuale
3) Controlli per impostare il tempo di ripetizione in modalità clock interno o per suddivisioni clock esterno
4) Switch per passaggio da clock interno a clock esterno
5) Ingresso clock esterno
6) Uscita trigger
7) Uscita per generazione di un trigger nell'istante di passaggio dalla fase positiva alla fase negativa del gate

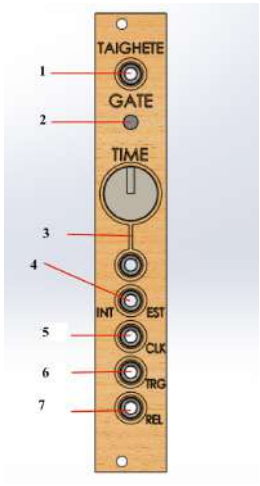


Figura 4.12 Design del pannello per il modulo Taighete

4.3.4 Cancelli

Cancelli è un modulo che permette di azzerare l'ampiezza di qualsiasi segnale presente ai suoi ingressi accoppiando una fotoresistenza con un led[91] controllato da Arduino: in questo modo il percorso dell'informazione rimane analogico, evitando sia perdite causate da una conversione AD che problemi di click nelle fasi di transizione dell'ampiezza; dei 6 circuiti vactrol disponibili, 3 hanno l'ulteriore aggiunta di un condensatore in serie alla fotoresistenza, costituendo un low pass gate nello stile dei sintetizzatori modulari Buchla.

Per questo modulo si è scelta una particolare gestione software dei segnali in ingresso: possono essere interrotti manualmente tramite interruttore o con un segnale di controllo binario esterno, invertendo il valore logico associato alla posizione dello switch; si ritene che questo design sia versatile nella maggior parte dei routing che prevedono l'azione congiunta tra strumentista e generatività del sistema, motivo per cui è stato implementato in altre unità presenti all'interno dello strumento e verrà identificato con il termine *switch adattivo*. *Cancelli* ha l'ulteriore capacità di gestire informazioni di controllo con frequenza superiore ai 20 Hz che, modificando l'ampiezza di un segnale in ingresso in banda audio, permettono di ottenere modulazioni a onda quadra in ampiezza (AM). Il modulo ha dimensioni di 10 HP e assorbe circa 55 mA dalla linea di 5 V, la sua interfaccia strumentista è così composta:

1) Ingressi per le informazioni che si desidera interrompere, possono essere sia segnali audio che di controllo

[91] In lingua inglese identificato con il termine vactrol, foto-accoppiatore in italiano.

2) Ingressi gate di controllo, per tutta la durata della loro fase negativa verrà invertito il valore logico associato alla posizione dello switch, aprendo o chiudendo il flusso di segnale
3) Uscite segnale
4) 6 mini-interruttori per aprire o chiudere i rispettivi cancelli, permettendo o meno il passaggio del segnale

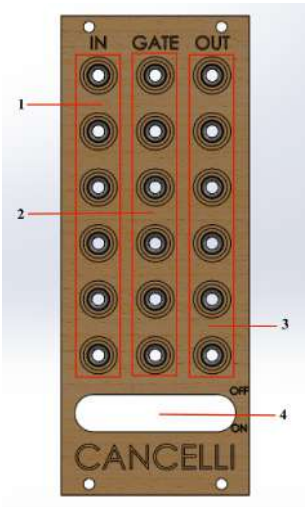


Figura 4.13 Design del pannello per il modulo Cancelli

4.3.5 Disco

Disco è costituito da un unico grande encoder rotativo ricavato dalla riconversione di un hard disk drive (HDD) presente su un computer dei primi anni del 2000. Oltre che per una questione legata all'utilizzo di materiali normalmente non riciclabili, si è ritenuto interessante sfruttare questo apparato per le sue qualità meccaniche: a differenza di un normale encoder e grazie all'inerzia del sistema, la rotazione può durare per un periodo anche superiore al tempo di interazione dello strumentista, permettendo gestualità non ottenibili con un normale potenziometro. I 3 segnali provenienti dalle fasi del motore associato all'HDD sono sfruttati per acquisire dati sul verso di rotazione (orario o antiorario) e sulla velocità angolare dell'albero, quindi convertiti in informazioni interpretabili dallo strumento modulare.

4.4 Acquisizione di informazioni da fonti esterne al sistema

Questo insieme di moduli fa guadagnare al sistema la capacità di acquisire informazioni dall'ambiente che lo circonda: un pattern di sensori con caratteristiche diverse permette di convertire un certo numero di variabili fisiche in segnali interpretabili dallo strumento.

In questa categoria rientrano anche tutti i moduli che permettono di ottenere informazioni dal corpo dello strumentista, considerato come una fonte esterna da cui è possibile ricavare dati relativi alla posizione degli arti, battito cardiaco ed emissione di fiato.

4.4.1 Luce

Luce è un modulo che converte l'intensità luminosa presente all'esterno dello strumento in segnali gestibili e interpretabili dal sistema. La luce è percepita da 4 fotoresistenze collegate al modulo tramite cavo jack, il cui segnale è convertito e riscalato da Arduino tra 0 e 5 V: la tensione minima e massima producibile dal modulo può essere impostata dallo strumentista grazie a due potenziometri, così da garantire la migliore resa sonora nell'associare percettivamente la luce al fenomeno sonoro. L'unità è dotata di uscite per la generazione di gate, i quali transitano nella fase positiva ogni qualvolta che l'intensità luminosa eccede, in positivo o negativo, gli estremi di un certo intervallo; è inoltre presente un algoritmo di taratura automatica delle fotoresistenze, così da ottenere risultati quanto più lineari possibili in ogni condizione di luce ambientale.
Il modulo è stato pensato per dotare il sistema di un primordiale *apparato visivo,* che si ritiene essere interessante per molte applicazioni generative, soprattutto in caso di ambienti con luminosità fortemente variabile; in più, lo strumentista può influenzare attivamente la luce che impatta sulle fotoresistenze del modulo, per controllare il comportamento di qualsiasi altra unità e costituendo così una sorta di theremin luminoso.

4.4.2 Vento

Vento sfrutta un anemometro WS1080 per acquisire informazioni su velocità istantanea e media del vento, mentre un encoder rotativo a 16 posizioni restituisce dati relativi alla sua direzione. Con questo modulo lo strumento ha un'ulteriore possibilità di feedback dall'ambiente esterno, le informazioni di velocità e direzione vengono riscalate all'interno del range 0 - 5 V e rese disponibili a tutte le altre unità. Il modulo ha un'uscita dedicata per convertire la frequenza di rotazione dell'albero dell'anemometro in informazioni coerenti con il protocollo Volt/ott: soprattutto per configurazioni generative all'aperto, si è ritenuto interessante poter associare la frequenza di uno o più oscillatori all'intensità del vento.

4.4.3 Iannis

Questo modulo è sviluppato sul chip ClosedCube BME680, un insieme di più sensori integrati che permettono di monitorare pressione atmosferica, temperatura e umidità. Queste variabili, a meno di un intervento invasivo da parte dello strumentista, sono difficilmente caratterizzate da variazioni repentine, associarle a un parametro musicale implica un'evoluzione temporale tanto dilatata da risultare pressoché impercettibile; la ragione principale per cui si è ritenuto comunque interessante sviluppare *Iannis*, è il suo possibile utilizzo in un contesto installativo piuttosto che performativo: lo strumento può essere configurato per suonare all'aperto continuativamente durante l'arco di una o più giornate, proponendo materiale sonoro variabile in funzione delle condizioni meteorologiche e ambientali, anche grazie all'aiuto dei moduli Luce e Vento.

4.4.4 Guanto

Guanto è un modulo progettato con lo scopo di interfacciare un giroscopio[92] allo strumento musicale modulare. Il sensore, vincolato a un guanto, monitora l'inclinazione della mano rispetto ai piani x e y, e la sua accelerazione perpendicolare al suolo: queste informazioni possono essere sfruttate per associare un gesto dello strumentista a qualsiasi parametro disponibile all'interno del sistema. Sul palmo del guanto è presente anche una fotoresistenza sfruttata per generare sia un gate positivo, ogni volta che la mano è aperta, che un segnale di controllo continuo all'incirca coerente con la percentuale di apertura della mano.
Il guanto è dotato di un emettitore wireless, per trasmettere tutti i dati necessari al modulo evitando qualsiasi impedimento fisico allo strumentista, che può allontanarsi dallo strumento fino a circa 3.5 metri.

4.4.5 Trentatré

Trentatré è un modulo che racchiude al suo interno un sensore di battito cardiaco[93], tensione muscolare[94] e uno spirometro[95]. Come l'unità Guanto, tutti i sensori sono collegati a un

[92] GY-521 MPU-6050

[93] Haljia Pulse Sensor

[94] MyoWare Muscle Sensor

[95] Honeywell ASDX Series Silicon - sensore differenziale di pressione

emettitore wireless, da applicare al corpo dello strumentista o di un esecutore esterno, impegnato, per esempio, in una danza o un'arte marziale e da cui sarà quindi possibile ottenere dati utili per una conversione musicale. Nello specifico l'associazione a parametri musicali è organizzata in questo modo:

- a ogni battito cardiaco è associato in uscita un trigger, quindi, analizzata una finestra temporale, è generato un segnale di clock
- il sensore di tensione muscolare genera segnali di controllo continui, più o meno ampi in funzione dell'intensità di contrazione. Un gate è generato al superamento di una certa soglia di tensione
- lo spirometro produce un segnale continuo coerente con la velocità del flusso di fiato emesso, mentre un gate è generato ogni qualvolta che l'intensità supera una certa soglia

4.5 Attuatori

I moduli appartenenti a questa famiglia permettono al sistema di agire attivamente su determinati elementi del mondo fisico che lo circonda, rappresentando una categoria complementare rispetto alle unità deputate alla sensoristica presentate nel paragrafo precedente. In questo senso, lo schema proposto nel paragrafo 3.2.5 può essere aggiornato come segue:

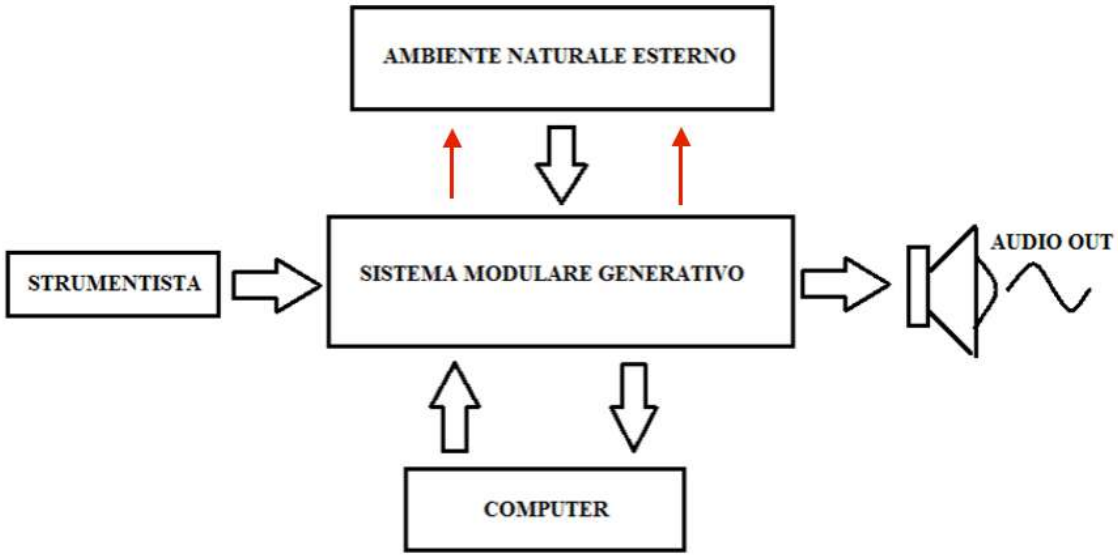


Figura 4.14 Schema logico di produzione sonora, integrato rispetto alla sezione di moduli attuatori

4.5.1 Sole

Sole permette di controllare l'attivazione di 2 solenoidi manualmente o tramite segnali di controllo esterni. Questi attuatori permettono di percuotere qualsiasi superficie: dal punto di vista sonoro risulta interessante ricercare timbri diversi variando materiali e geometrie, soprattutto se a questo elemento è associato un microfono piezoelettrico.

4.5.2 Motore

Motore permette di controllare la rotazione di 3 motori DC manualmente o tramite segnali di controllo esterni al modulo. Un motore può essere utile per mettere in vibrazione molti materiali diversi, personalmente si ritiene particolarmente interessante se sfruttato per eccitare una o più corde di strumenti tradizionali, permettendone in un certo senso l'automazione.

4.5.3 Nonservo

Nonservo permette di variare l'angolo dell'albero di 2 motori passo-passo manualmente o tramite segnali di controllo esterni al modulo. Questa tipologia di attuatori è utile per compiere spostamenti piccoli, lenti e precisi; sebbene non si sia trovata, attualmente, una precisa applicazione musicale per questo modulo, risulta comunque piuttosto interessante per lo sfregamento automatizzato di superfici di diverso materiale.

Una possibile applicazione futura per questo modulo prevede la creazione di un braccio robotico a due gradi di libertà, cui è vincolato un pungolo: l'idea è quella di creare due piastre metalliche gemelle, una utilizzata da uno strumentista umano e l'altra da una rete neurale che, controllando il braccio robotico, ripropone i gesti appresi dalla prima piastra.

4.6 Gestione e condizionamento di informazioni

A questa famiglia appartengono tutti quei moduli il cui design è direttamente ispirato agli argomenti esposti nel secondo capitolo di questo lavoro di tesi: il concetto di generatività è qui implementato esplicitamente.

4.6.1 PID

Il modulo *PID* codifica al proprio interno un semplice sistema di controllo attivo su determinate variabili, suggerito dai processori utilizzati nell'ambito dell'ingegneria

dell'automazione. Può essere considerato come una sorta di quantizzatore di informazioni in ingresso: il modulo le valuta, confrontandole con dei valori di riferimento, e restituisce in uscita un segnale modificato coerentemente rispetto alle azioni proporzionali, integrative e derivative tipiche di un controllo PID.

Con il corretto posizionamento dei 3 potenziometri associati a P, I e D è possibile:

- interpolare con curve più morbide dei segnali in ingresso, binari o a gradino (*slew limiter*)
- rendere dei segnali continui, binari o a gradino (*discretizzazione*)
- ottenere effetti di *glide* e *portamento* su informazioni di Volt/ott
- quantizzare segnali continui rispetto a una determinata scala musicale (*quantizer*)
- introdurre fluttuazioni e imprevedibilità nell'andamento di un segnale (*jittering*)
- ottenere informazioni randomiche in uscita a partire da segnali prevedibili come LFO e inviluppi (una sorta di *sample and hold*)

4.6.2 Neurone

Neurone è capace di valutare fino a 8 segnali in ingresso, binari o continui, e di produrre in uscita un unico segnale, ottenuto grazie alla media pesata tra tutti gli input. A ogni ingresso è associato un coefficiente di pesatura diverso, non è possibile variare questi parametri se non nel routing del segnale: come per il modulo Ingranaggi, i valori di pesatura possono essere ruotati in senso orario o antiorario al fine di riconfigurare i collegamenti logici di ogni ingresso, così da ottenere, a parità di input, informazioni variabili in uscita.

4.6.3 I Ching

I Ching è un modulo che codifica nella propria logica di funzionamento le informazioni contenute nell'omonimo libro. Alla ricezione di un trigger o un gate è associata la simulazione di un lancio di una moneta, raggiunti 3 valori consecutivi viene restituito in uscita il *responso* associato all'esagramma ottenuto dal lancio delle monete; l'intervallo compreso tra 0 e 5 Volt è suddiviso in 64 valori, uno per esagramma, quindi proposto in output come segnale utile ad altri moduli. Saranno inoltre generati 6 gate coerenti con l'esagramma selezionato, positivi per le linee continue e negativi per le linee spezzate.

4.7 Sintesi sonora

Questo insieme di moduli è deputato alla generazione sonora vera e propria dello strumento, così da finalizzare il lavoro svolto da tutte le altre unità descritte fino a questo punto.

4.7.1 Piastra

Piastra è l'unico elemento completamente acustico presente all'interno dello strumento, formato da una piastra di ferro di 2 mm di spessore e lunga 68 HP, su cui sono fissate 8 molle con diverso spessore e coefficiente elastico. Allo stato attuale il modulo non ha circuiti attivi, per cui la produzione di segnale è affidata semplicemente a 3 microfoni piezoelettrici disposti equidistantemente sulla superficie e dotati di uscite indipendenti per riprodurre un panorama stereo; è inoltre presente un ingresso collegato a uno speaker passivo da 8 Ohm, pensato per mettere in vibrazione il metallo grazie al fenomeno del feedback acustico.
Questa unità è stata progettata per lavorare in simbiosi con i moduli Sole, Motore e Nonservo: sulla sua superficie sono fissati dei supporti per vincolare in maniera non permanente qualsiasi tipo di attuatore, così che buona parte dello strumento modulare possa partecipare all'automatizzazione dei suoni prodotti dalla piastra metallica e dalle sue molle.

4.7.2 Radio

Radio è un modulo basato sul chip TEA5767N che permette di captare onde radio FM da 85.00 a 105.00 MHz, range al cui interno ricadono le principali trasmissioni radiofoniche regionali e nazionali. Il design è stato ispirato principalmente da Immaginary Landscape No. 4 di Cage[96]: si ritiene interessante associare questo modulo a diverse catene di effetti e looper per accumulare materiali sonori assolutamente casuali e svincolati tra loro, riproponendoli fusi insieme grazie all'uso dell'elettronica. Il modulo è in oltre un'ottima fonte di rumore con timbri di colore diverso, grazie alla possibilità di variare microscopicamente la frequenza di sintonizzazione della radio.

4.7.3 N-Dronato

N-Dronato genera 3 onde quadre e le somma in un unico output grazie a un circuito analogico integrato con 3 *switch adattivi*. Questo modulo non prevede la gestione singola delle forme

[96] Paragrafo 2.2.2

d’onda prodotte: sono presenti solo 3 macro-controlli che permettono di variare la frequenza media degli oscillatori, la loro differenza in centesimi di tono e il tempo di inviluppo. Si è ritenuto che un design di questo tipo, seppur atipico per un oscillatore, sia particolarmente efficace per la generazione di bordoni che evolvono lentamente nel tempo, scopo principale per cui è stato progettato il modulo.

4.7.4 Botta

Botta è un sintetizzatore di eventi sonori timbricamente associabili alla percussione di una membrana. L’algoritmo di sintesi del modulo è gestito grazie ai seguenti blocchi logici:

- oscillatore portante sinusoidale
- oscillatore modulante, di cui è possibile selezionare la forma d’onda
- indice di modulazione variabile della sintesi FM
- detuning tra l’oscillatore portante e modulante, per ottenere spettri più o meno armonici
- inviluppo esponenziale Attack-Hold-Decay, azionabile tramite trigger (AD) o gate (AHD) esterni

4.8 Utility

Questi moduli, per lo più passivi e quindi non gestiti da un Arduino, hanno il compito di facilitare il routing e la gestione dei segnali che circolano all’interno dello strumento.

4.8.1 Molti

Molti accetta un segnale sbilanciato in ingresso e ne propone in uscita 7 copie all’incirca identiche. Questo modulo passivo connette semplicemente in parallelo più jack femmina: è principalmente adatto alla moltiplica di segnali binari, mentre potrebbe risultare poco efficace per i segnali di controllo, a causa di una leggera e possibile perdita di intensità nel segnale duplicato.

Uno switch permette di scegliere se avere un’unica linea di copie identiche (7 copie) o due circuiti indipendenti (3 copie per circuito).

4.8.2 Adatta

Adatta è un semplice modulo passivo, utile per cambiare lo standard dimensionale dei jack evitando l'utilizzo di adattatori esterni: permette di collegare facilmente i cavi sbilanciati dello strumento da 3.5 mm a cavi sbilanciati da 6.3 mm, tipicamente più comuni nelle schede audio e mixer.

4.8.3 Bivio

Bivio propone 4 circuiti passivi identici che accettano un segnale in ingresso e lo inviano a due uscite distinte, o viceversa.

4.8.4 Squarepush

Questo modulo permette di sommare, grazie a un insieme di operatori logici OR, 8 segnali binari differenti in un unico output; è utile per inviare in un unico input più trigger provenienti da varie fonti.

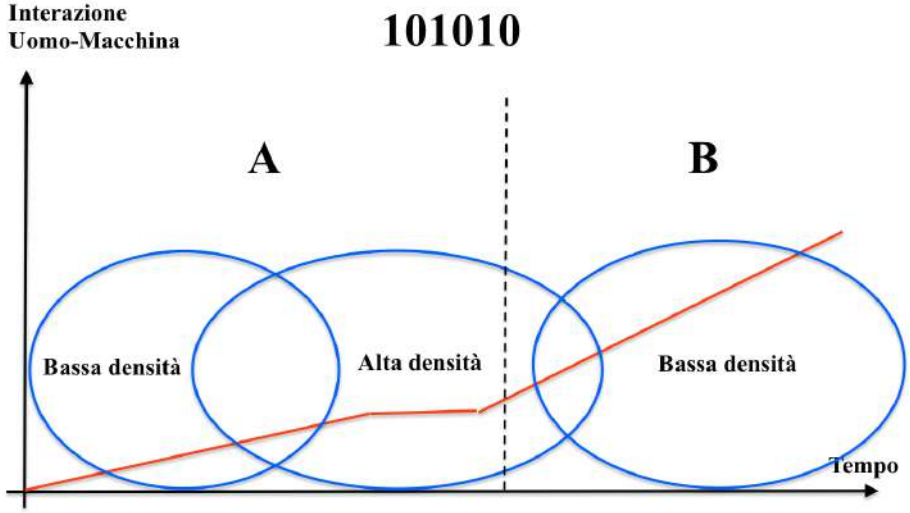
4.9 101010: dallo strumento alla performance

Come esposto nel paragrafo 3.2.5, le possibili combinazioni tra ogni unità sono pressoché infinite: si è ritenuto, quindi, più pratico limitarsi alla sola spiegazione delle funzioni specifiche di ogni modulo, delegando l'esposizione delle potenzialità del sistema al brano presentato come progetto finale per questo lavoro di tesi, dal titolo *101010:* un live electronics quadrifonico per esecutore e strumento elettroacustico generativo, della durata di circa 6-8 minuti. La performance mira a indagare il concetto di dualità tra strumento e strumentista, entrambe le parti hanno la possibilità di agire in maniera autonoma, pur mantenendo un costante feedback sulle reciproche direzioni musicali; lo scopo è quello di influenzarsi vicendevolmente per raggiungere l'ottenimento della forma finale: la componente umana da una parte e la generatività dall'altra, tendono a bilanciare quanto più possibile i propri comportamenti per raggiungere uno stato di simbiosi e armonia tra strumentista e strumento.
Il titolo stesso ha lo scopo di rendere esplicita la componente duale del brano, ulteriormente sottolineata dalla divisione in sole due macrosezioni timbricamente speculari, in cui il parametro della densità sonora avrà maggiore rilevanza:

- sezione A: il sistema è appena stato avviato, l'arrangiamento iniziale tra i collegamenti di ogni modulo identifica un comportamento piuttosto casuale caratterizzato da eventi impulsivi e dilatati nel tempo. Lo strumentista comincia ad agire su un certo numero di parametri, concentrandosi principalmente su una densità sempre crescente del suono e su micro-variazioni timbriche; raggiunto uno stato di equilibrio soddisfacente, addensando al massimo gli eventi e saturando lo spettro si procede verso un punto di climax, all'estinzione del quale si apre piuttosto bruscamente, come in un segnale binario da 1 a 0, la sezione successiva.
- sezione B: la massa *caotica* della prima sezione si distende, l'interazione tra uomo e macchina è più consapevole e pacifica, gli eventi percussivi lasciano spazio a fasce e bordoni. Rimane una componente di imprevedibilità nei reciproci comportamenti, ma è voluta e ben accetta, considerando naturale qualsiasi prodotto dato dall'unione tra i due sistemi complessi *strumento* ed *esecutore*. Il brano si chiude con una rarefazione sempre maggiore fino al *silenzio*.

Il sistema ha la responsabilità di tutta la produzione sonora, mentre la manipolazione audio e la diffusione quadrifonica sono gestite dallo strumentista grazie alla DAW Ableton, con dispositivi personalmente sviluppati in Max For Live.

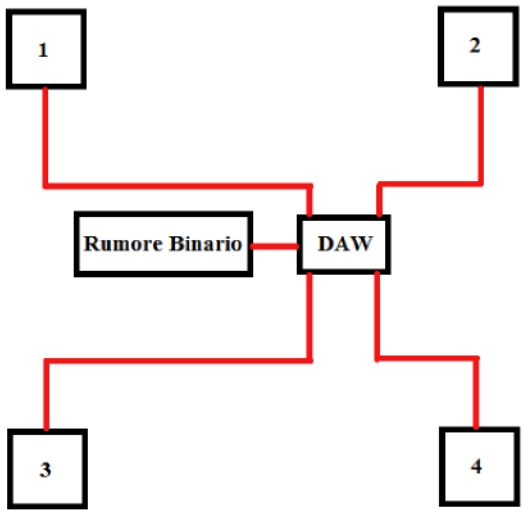
L'esecuzione è caratterizzata da una fortissima componente improvvisativa e aleatoria, soprattutto considerando la particolare tipologia di interazione e gestione dello strumento; quanto segue è ciò che si è ritenuto sufficiente e necessario presentare come *partitura* per questo progetto performativo.



Orologio (clock out)	>	in Molti	>	in clock Ingranaggi, Battiti, Trigga
VIII out Ingranaggi	>	I in Rikeda	>	II in Ndronato
VII out Ingranaggi	>	II in Rikeda	>	III in Ndronato
III out Battiti	>	II in Boolle		
IV out Tocca	>	I in Boolle		
out OR Boolle	>	gate in Radio		
out NOR Boolle	>	clock in Block		
out A Block	>	in freq Botta		
out B Block	>	in freq Radio		
I out Tocca	>	in hold Botta		
Trigger out I Disco	>	I in Ndronato		
I out Trigga	>	I in Sole		
II out Trigga	>	II in Sole		
III out Trigga	>	I in Motore		
IV out Trigga	>	II in Motore		

Lo schema proposto è il routing minimo tra i moduli al fine di identificare univocamente lo stato iniziale dello strumento, fermo restando che ha sola validità nell'*istante zero* immediatamente successivo all'accensione del sistema: non è possibile descriverne ulteriormente e con certezza gli stati successivi, così come la sua evoluzione temporale.

4.9.1 Processi di spazializzazione



La performance si avvale di un sistema di diffusione quadrifonico, dove specifici algoritmi implementati in Max For Live permettono una spazializzazione quanto più coerente possibile con le intenzioni esecutive. Considerata la mole di parametri che deve gestire lo strumentista in tempo reale, si è scelto di creare diversi paradigmi di movimento facilmente accessibili grazie al routing logico di Ableton: ogni processo spaziale è associato a un *send* della DAW, l'esecutore può quindi scegliere quanta percentuale di un dato suono deve essere essere soggetta a movimento all'interno dello spazio quadrifonico.

Di seguito sono illustrati i paradigmi che si è scelto di definire:

- STATICO, il suono è diffuso uniformemente su tutti gli speaker
- ALEA, il suono è spazializzato casualmente e puntualmente su una singola cassa alla volta. Questo paradigma è sfruttato nella sezione A per enfatizzare ulteriormente il concetto di imprevedibilità
- SEGMENTO, il suono segue traiettorie nello spazio che randomicamente possono coinvolgere altoparlanti adiacenti o distanti (secondo movimenti diagonali) con un certo tempo di interpolazione. E' utilizzato principalmente nella prima sezione, con maggiore enfasi nel punto di massima densità
- ROTAZIONE SPOT, il suono può ruotare in senso orario o antiorario presentandosi su una cassa alla volta. E' usato in piccola misura e con tempi di rotazione rapidi nella sezione A,

mentre caratterizza l'andamento spaziale della seconda sezione grazie a movimenti più pacati e rilassati

- MOVIMENTO MASSA, questo paradigma permette di spostare il suono tra due coppie di casse alla volta, garantendo movimenti avanti-dietro (da 1-2 a 3-4 e viceversa) o sinistra-destra (da 1-3 a 2-4 e viceversa). Particolarmente presente e con tempi di spostamento sempre maggiori nella sezione B
- SUB, qualora fosse disponibile un sub e l'acustica della sala lo permetta o richieda, tutti i suoni verranno inviati costantemente a questo *send,* associato all'uscita fisica relativa dello speaker; su questa mandata è utile introdurre un filtro passa-basso, i cui parametri sono da impostare secondo le esigenze dell'esecutore

Lo strumentista gestisce in tempo reale il routing logico di ogni segnale proveniente dal sistema su ciascuno di questi paradigmi, così come la velocità con cui ogni movimento è eseguito nello spazio; la variazione di questi parametri deve assecondare ed enfatizzare quanto più possibile ciò che viene prodotto dall'interazione uomo-macchina.

CONCLUSIONI

Questo lavoro di tesi *chiude* circa un anno di studi, ricerche ed esperimenti, musicali e non, condotti dall'autore: se da un lato si è riuscito a ottenere uno strumento che asseconda la quasi totalità delle personali esigenze musicali ed espressive pensate per live electronics, dall'altro, però, non si è ottenuto il grado di generatività e autonomia che ci si era prefissato nelle fasi iniziali del progetto. Lo strumento creato è capace di generare texture, paesaggi sonori che evolvono all'infinito, bordoni, ma non ha la possibilità di agire autonomamente per creare architetture complesse o eventi sonori funzionali alla gestione di una struttura finita nel tempo. Questo limite è considerato personalmente marginale, visto l'interesse artistico per tale direzione musicale che, con ogni probabilità, ha inconsciamente guidato verso un design del genere. Tuttavia, va osservato che lavorare su questo tipo di sistemi implica un approccio molto particolare alla musica, che ha stimolato un interessante cambio di prospettiva: passare dall'acquisizione di metodi di lavoro codificati, che rappresentano ormai la prassi nell'approccio alla musica elettroacustica, alla ricerca vera e propria di una personale via creativa. Tutto ciò può essere riassunto da queste 3 domande:

Che cos'è la musica?
Che cos'è la bellezza?
Come si può *spiegarlo* a circa 30 Arduino?

Non si ha la presunzione di aver risposto ad alcuna di queste domande, ma è stato fondamentale osservare come il semplice atto di prenderne coscienza con tale e nuova prospettiva abbia fatto maturare nuove idee e ne ha confermate razionalmente altre che sarebbero altrimenti rimaste impressioni relegate al subconscio. Da un punto di vista di crescita personale, ci si è resi conto che l'aver ottenuto uno strumento, generativo o meno, un prodotto finito, è un aspetto assolutamente marginale: molto più importante è stato l'investire circa un anno in ricerche, letture, ascolti e prove con circuiti e codici. In questo lasso di tempo le fondamenta musicali che erano instabili, sono diventate un po' più solide, focalizzate in una direzione definita: ne è nato un percorso di ricerca personale.

Il lavoro svolto necessita di ulteriore ricerca e sviluppo negli anni a venire: lo strumento modulare ha incredibili potenzialità musicali ed espressive; l'obbiettivo sarà quello di *cucire* su se stessi un sistema con cui ci si senta perfettamente a proprio agio, che sia una vera estensione del corpo e della mente del musicista, uno strumento che assecondi il gesto e l'idea.

Bibliografia

BAGGI D. L., *The Role of Computer Technology in Music and Musicology*, 1998

BEAMENT J., *The violin explained: Componets, Mechanism and Sound*, Oxford University Press, 1997.

BODE H., *Sound Synthesizer creates new musical effects*, Electronics magazine, Dicembre 1961.

BONFANTE L. *Etruscan Life and after Life: A handbook of etruscan studies*, Wayne State University Press, 1986.

CAGE J., *Silenzio*, Shake, 2008.

CAPITONI F., *In C, opera aperta. Guida ai capolavori di Terry Riley*, Arcana, 2016

CASELLA A., *Il Pianoforte*, Ricordi, 1984.

CHADABE J., *New Approaches to analog-studio design,* Perspectives of New Music, 1967.

CHOMSKY N., *Three Models for the Description of Language*, MIT Press, 1956.

COPE D., *Computer Models of Musical Creativity,* MIT Press, 2005.

CREMASCHI A. e GIOMI F., *Rumore bianco. Introduzione alla musica digitale*, Zanichelli, 2008.

CRNKOVIC G. e GIOVAGNOLI R., *Computing nature: Turing centenary perspective*, Springer, 2013.

DIANA G., FOSSATI F. e RESTA F., *Elementi di controllo di sistemi meccanici,* Edizioni Spiegel, 1998.

DUNN D., *A history of electronic music pioneers*, Ars Electronica, 1992.

EMMERSON S., *Living Electronic Music*, Routledge, 2007.

GIOMI F., *L'intelligenza artificiale nella musicologia cognitiva: approcci ed applicazioni*, Il Mulino, 1995.

HOFSTADTER D., *Gödel, Escher, Bach: Un'eterna ghirlanda brillante*, Adelphi,1984.

HOLMES T., *Electronic and experimental music: Technology, music and culture*, Scribner, 1985.

HOPCROFT J. e ULLMAN J. *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, 1979.

KANT I., *Kritik der reinen Vernunft*, 1781.

KUNEJ D. e TURK I., *New perspectives on the beginnings of music: archaeological and musicological analysis of a Middle Palaeolithic bone "flute",* MIT Press, 2000.

LAURSON M., KUUSKANKARE M. e SANDRED O., *Revisiting the Illiac Suite*, researchgate.net, 2009.

LEIBNIZ W., *Spiegazione dell'aritmetica binaria*, 1705.

LERDAHL F. e JACKENDOFF R., *A Generative Theory of Tonal Music*, MIT Press, 1983.

LINDENMAYER A. e PRUSINKIEWICZ P., *The Algorithmic Beauty of Plants*, Springer-Verlag, 1990.

MACHOVER T., *Hyperinstruments: A progress Report 1987 - 1991*, MIT Media Laboratory, 1992.

MANNING P., *Computer and Electronic Music*, Oxford Univ. Press, 1993.

MATHEWS M., *The Technology of Computer Music,* MIT Press, 1969.

MC LUHAN M., *Gli strumenti del comunicare*, Il saggiatore, 1967.

MENZIES R., *New Electronic Performance Instruments for Electroacoustic Music*, University of York, 1999.

MOORE R., *Elements of computer music*, Prentice-Hall Inc, 1969.

PASCERI C.,	*Tecnolgia Musicale: La rivelazione della musica,* Aracne Editrice, 2011.
PLINIO IL VECCHIO,	*Storia Naturale*, 77 a.C., tradotto da Harris Rackham, Harvard University Press, 1938.
POOLE D., MACKWORTH A. e GOEBEL R.,	*Computational Intelligence: A Logical Approach*, Oxford University Press, 1998.
RECK M. E. e BILES J. A.,	*Evolutionary Computer Music*, Springer, 2007.
RICCIARDI C.,	*L'albero che canta – Il didgeridoo*, Wondermark, 2011.
RUSSOLO L.,	*L'arte dei rumori*, Edizioni futuriste di "poesia", 1913.
SACHS C.,	*Storia degli strumenti musicali*, Arnoldo Mondadori, 1980.
SCHAEFFER P.,	*Traité des objets musicaux*, Seuil, 1966.
SCHAFER R. M.,	*The Tuning of the World*, Random House Inc, 1977.
WEBERN A.,	*Verso la nuova musica*, Bompiani, 1963.
WINTER-JENSEN A.	*Automates et musiques pendules*, Musee De L'Horlogerie et De L'émaillerie, 1987.
XENAKIS I.,	*Musiques formelles: nouveaux principes formels de composition musicale*, Editions Richard-Masse, 1963.

Sitografia

GREGORI G.,	Archivio della liuteria cremonese. http://www.archiviodellaliuteriacremonese.it
CRAB S.,	120 Years of Electronic Music. http://120years.net
BARRASS T.,	Mozzi - Audio synthesis library for Arduino. https://sensorium.github.io/Mozzi/
LAPORTE S. e MAUMOUD H.,	Listen to Wikipedia. http://listen.hatnote.com/#en
PARVIAINIEN T.,	Trams of Helsinki. https://codepen.io/teropa/full/mBbPEe/
PARVIAINIEN T.,	How generative music works - A prespective. https://teropa.info/loop/#/systemdefinition
DOEPFER D.,	A100 system DIY. www.doepfer.de/a100_man/a100m_e.htm
ENO B.,	Brian Eno about generative music. http://www.inmotionmagazine.com/eno1.html
REICH S.,	Steve Reich on *Pendulum Music*. http://www.furious.com/perfect/ohm/reich.html
TYKA M.,	*The art of neural networks*. TEDx Talks, 2015. https://www.youtube.com/watch?v=0qVOUD76JOg
JI-SUNG K.,	Using Keras & Theano for deep learning driven jazz generation. https://deepjazz.io
LIANG F., GOTHAM M., TOMCZAK M., JOHNSON M. e SHOTTON J.,	The BachBot Challenge http://bachbot.com/#/?_k=39izvg
MAGENTA TEAM	Magenta - Make Music and Art Using Machine Learning https://magenta.tensorflow.org
VITELLI M.,	GRUV https://github.com/MattVitelli/GRUV
OKAWA-DENSHI S.,	Strumenti di calcolo per dimensionamento di filtri analogici. http://sim.okawa-denshi.jp/

Ringraziamenti