FACULTAD DE INGENIERÍAS Y CIENCIAS AGROPECUARIAS
ESCUELA DE INGENIERÍA EN SONIDO Y ACÚSTICA

"DISEÑO Y PROGRAMACIÓN DE UN SINTETIZADOR VIRTUAL DE SONIDO POR COMPONENTES ESPECTRALES"

Trabajo de titulación presentado en conformidad a los requisitos establecidos para optar por el título de Ingeniero de Sonido y Acústica

Profesor Guía:

Ing. Marcelo Lazzati

Autores:

Nicolás Víctor Aráuz Soria
Carlos Patricio Jácome Zambrano

2012

DECLARACIÓN DEL PROFESOR GUÍA

“Declaro haber dirigido este trabajo a través de las reuniones periódicas con los estudiantes, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”.

Marcelo Lazzati

Ingeniero en Ejecución de Sonido

CI: 171163573-8

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

"Declaramos que este trabajo es original, de nuestra autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes".

Nicolás Víctor Aráuz Soria

CI: 171717585-3

Carlos Patricio Jácome Zambrano

CI: 171959068-7

AGRADECIMIENTOS

Yo, Nicolás Aráuz agradezco profundamente a las siguientes personas por la ayuda brindada para que este proyecto llegue a buen término:

A mis padres por todo el apoyo brindado a este proyecto, y a lo largo de toda la carrera.

Marcelo Lazzati por la buena idea, y por todos los aportes brindados para concretarla.

A todos mis amigos por todo el respaldo y tiempo brindado, pero principalmente por su amistad.

A todos muchas gracias por aportar a la consecución de este proyecto en buenos términos.

AGRADECIMIENTOS

Yo, Carlos Jácome agradezco a las siguientes personas por el aporte y ayuda brindada:

Marcelo Lazzati por su guía y consejo el cual ayudó a culminar el proyecto en buenos términos con los mejores resultados posibles.

Hugo Jácome por su guía y enseñanza en el entorno de programación utilizado.

A mis padres y hermanos, por el apoyo y las sugerencias brindadas a lo largo del periodo de este proyecto.

Gabriela Leiva por su aporte en la parte visual, gráfica y de color.

Muchas gracias por ser parte importante en los distintos aspectos que presenta el proyecto.

RESUMEN

El presente proyecto plantea la posibilidad de diseñar y programar una aplicación de un sintetizador virtual de sonido por componentes espectrales, con una interface gráfica que permita al usuario visualizar el espectro de frecuencias de los sonidos a sintetizarse.

La idea general del proyecto surge de la comprensión del análisis de Fourier, el cual dice que todo sonido complejo puede ser descompuesto en una suma de funciones seno y coseno. Así mismo, mediante la suma de tonos puros, es posible sintetizar sonidos complejos, es por esto que cobra sentido la idea de un sintetizador por componentes espectrales.

El diseño del sintetizador parte de las necesidades de este, combinadas con las capacidades del entorno de programación a usarse. Para la programación del sintetizador, se usa al programa Max/MSP. Este *software* con programación orientada a objetos permite la conexión de objetos ya creados para hacer estructuras complejas, en este caso un sintetizador aditivo.

Para evaluar las capacidades del sintetizador se usan un método cuantitativo y cualitativo. El primero fue usado para obtener datos técnicos del funcionamiento del sintetizador y de los sonidos sintetizados, el segundo para identificar las cualidades sonoras de estos.

Este proyecto permite comprender de mejor manera a la síntesis aditiva, cómo esta trabaja, y sus limitaciones y ventajas. Por otro lado permite también entender cómo se relacionan los distintos componentes espectrales para generar un sonido.

ABSTRACT

The present project proposes the possibility to design and program an application of a virtual sound synthesizer by spectral components, with a graphic interface that allows the user to view the frequency spectrum of the sound being synthesized.

The main idea of the project comes from the understanding of the Fourier analysis, which says that every complex sound can be decomposed in a sum of sine and cosine functions. Likewise, by the sum of pure tones, it's possible to synthesize complex sounds, is by this that the idea of a synthesizer by spectral components has sense.

The design of the synthesizer comes from its needs, combined with the capabilities of the used programing environment. For the programing of the synthesizer, the software Max/MSP it's used. This objects oriented programing software allows connecting created objects and make complex structures, in this case an additive synth.

To evaluate the capabilities of the synthesizer a quantitative and qualitative methods are used. The first one was used to get technical data of the synthesizer performance and the synthesized sounds, the second one to identify the sonic qualities of them.

This project allows a better understanding of the additive synthesis, how it works, and its limitations and advantages. By other side it allows too to understand how the different spectral components generate a sound.

ÍNDICE

1. Introducción

1.1. Antecedentes

Los primeros dispositivos electrónicos concebidos como instrumentos musicales fueron inventados en la década de 1920, entre los cuales destacan el Theremin (1923), el cual permite generar sonidos variando su frecuencia y amplitud; y el Martenot (1928), primer sintetizador en incorporar un teclado para su ejecución.

Con el paso del tiempo, el segundo de los anteriores se iría desarrollando y aparecerían otros modelos de sintetizadores entre los que se destacan el Voder (1939), primer sintetizador pensado en la reproducción de la palabra hablada; y Mark I y Mark II (1965), diseñados bajo el patrocinio de la RCA.

Si bien los dispositivos anteriores ya realizaban síntesis sonora, no es sino hasta la década de 1960, gracias a la masificación de los transistores, que es posible la construcción de generadores de sonido más precisos. Es así que en 1963 Robert Moog desarrolla los primeros VCO (osciladores controlados por voltaje), y VCA (amplificadores controlados por voltaje), y un año más tarde los VCF (filtros controlados por voltaje) pasa alto y pasa bajo, y los primeros ADSR (generadores de envolvente). Como resultado de todo este desarrollo tecnológico, en octubre de 1964 Robert Moog presenta en una convención de la AES al sintetizador con mayor repercusión en el desarrollo histórico musical, el Moog Modular.

Una vez inventado el Moog Modular, comenzó la comercialización del mismo, y en 1968 quedó demostrada la posibilidad de utilizar a los sintetizadores como instrumentos musicales gracias a la afamada obra musical Switched On Bach del músico Walter Carlos, en la cual se ejecutan piezas musicales de J.S. Bach con el Moog Modular, dando como resultado un rotundo éxito.

Posteriormente a la aceptación generalizada del sintetizador sonoro como instrumento musical, se siguieron inventando distintos métodos de síntesis así como diferentes modelos de sintetizadores, los cuales aplicando nuevas

tecnologías se volvían más asequibles cada vez, masificando su uso en el ámbito musical.

Con la aparición de la tecnología digital, los sintetizadores alcanzaron mayores prestaciones, ya que si bien los métodos utilizados eran los mismos que en los sintetizadores electrónicos, los componentes usados en la fabricación de los sintetizadores digitales eran más exactos y estables. Así mismo, los sintetizadores ganaron en versatilidad gracias a la aparición de la tecnología MIDI, inventada por Dave Smith en 1982 y estandarizada a principios de 1983.

Finalmente, los sintetizadores llegaron al mundo de la computación personal apareciendo los denominados sintetizadores virtuales, los cuales ocupan las capacidades de procesamiento de los computadores personales para realizar los procesos de generación de sonidos y de síntesis sonora, lo que aumentó de forma increíble los tipos de sintetizadores y métodos de síntesis disponibles, convirtiendo al sintetizador en una herramienta indispensable de creación y emulación de sonidos.

1.2. Marco Referencial

El presente proyecto se basa en el diseño e implementación de un sintetizador virtual por componentes espectrales, para lo cual se utilizará un método de síntesis muy conocido denominado método aditivo, el que funciona principalmente con tres componentes básicos de síntesis: los VCO (osciladores controlados por voltaje), los VCA (amplificadores controlados por voltaje) y los ADSR (generadores de envolvente).

Una parte fundamental del proyecto vendría a ser la elección del medio de programación para realizar el sintetizador, para lo cual se tiene como opciones Max/MSP y SuperCollider, que serán definidos en un estudio más especializado para conocer cuál presenta mayores ventajas para este proyecto.

El problema planteado para este proyecto es el desconocimiento de una interface comercial de programación que permita modelar cuantitativamente

(en tiempo real) en detalle al espectro sonoro, y a la vez incorpore una interface de visualización del espectro frecuencial sintetizado.

1.3. Alcance

Para la realización del proyecto se vuelve necesario tener conocimientos en las áreas de síntesis sonora (métodos de síntesis y componentes involucrados en dichos métodos) y programación, para poder implementar la aplicación del sintetizador espectral.

Como finalidad ulterior, se pretende crear una aplicación *stand alone* del sintetizador virtual, en la cual se incorpore una interfaz gráfica con todos los componentes de síntesis y sus controles, una sección de control que permita la ejecución del sintetizador, y un analizador de espectro. Adicionalmente se desea dar la opción de manejar el sintetizador por medio de superficies de control externas.

1.4. Justificación

Mediante el presente proyecto se pretenden conocer algunos métodos de síntesis (enfocándose más profundamente en el método aditivo) y parte de la tecnología que estos procesos involucran, para así aprender de sus fundamentos y de su correcto manejo dada la importancia que estos tienen en la emulación y creación de sonidos.

Si bien en el presente proyecto no se pretende inventar nuevos métodos de síntesis, sino por el contrario utilizar los previamente conocidos y enfocarlos desde un nuevo punto de vista (el cual se centra en la visualización de los espectros armónicos de las señales sintetizadas), el cual entregaría al usuario una forma de evaluación comparativa cuantitativa-cualitativa (espectro-timbre) del sonido sintetizado. La herramienta propuesta (el sintetizador espectral) puede resultar de interés para músicos profesionales y aficionados, para los productores musicales, y especialmente para sonidistas en formación.

1.5. Objetivos

1.5.1. Objetivo General

- Diseñar y programar una aplicación de un sintetizador aditivo virtual de sonido por componentes espectrales.

1.5.2. Objetivos Específicos

- Determinar los componentes de síntesis sonora que serán parte del sintetizador.
- Determinar un lenguaje de programación que permita la programación del sintetizador.
- Permitir el manejo de los componentes del sintetizador, por medio de superficies de control externas MIDI.
- Evaluar cualitativamente las capacidades de emulación de sonidos del sintetizador virtual.

1.6. Hipótesis

Como hipótesis del presente proyecto de titulación se asume que resulta factible el diseñar e implementar un sintetizador virtual aditivo de sonido, que permita la emulación de sonidos realistas y la creación de nuevos sonidos, incorporando una interface de programación y visualización gráfica en tiempo real del espectro armónico de la señal.

2. Marco Teórico

2.1. Propiedades del Sonido

Se define al sonido como una perturbación en un medio elástico, el cual genera una variación de presión y movimiento de partículas. También puede ser considerado como una percepción psicoacústica generada por los impulsos nerviosos del cerebro, ante la reacción mecánica producida en el oído.

2.1.1. Amplitud, Frecuencia y Fase

2.1.1.1. Amplitud

La amplitud de una onda es la desviación máxima que alcanza una oscilación en un ciclo, respecto a su punto de equilibrio.

Los valores de amplitud de un sonido vienen asociados a la presión sonora generada por la perturbación causante de este. Dado que los valores de presión perceptibles por el oído humano oscilan en un rango muy amplio, yendo desde 20 µPa a 20 Pa, se introduce una escala logarítmica que comprime este rango con la finalidad de volverlo más manejable.

Los nuevos valores que toma la presión para ajustarse a esta escala, son los valores de Nivel de Presión Sonora, los cuales vienen dados en decibeles (dB). Este nivel se calcula mediante la fórmula:

$$NPS = 20 \log_{10}\left(\frac{P}{Pref}\right) \ [dB]$$

Donde: P es la presión sonora medida.
Pref es la presión sonora de referencia igual a 20 µPa.

Ec. 1

Fórmula del nivel de presión sonora o NPS.

2.1.1.2. Frecuencia

La frecuencia es el número o la cantidad de ciclos por segundo, asociado al sonido producido por una fuente.

La frecuencia se mide en Hertz (Hz) en honor a Heinrich Hertz (1857-1894), científico alemán que descubrió las ondas de radio. Esta unidad es equivalente al ciclo por segundo.

La frecuencia es un parámetro que se relaciona con el periodo, el cual se define como el tiempo transcurrido entre una perturbación y la siguiente, la cual viene dada por la siguiente fórmula:

$$T = \frac{1}{f}[\text{s}]$$

Donde: T es periodo medido en segundos (s).
f es frecuencia medida en Hz.

Ec. 2

Relación entre periodo y frecuencia.

Otros dos parámetros vinculados estrechamente con la frecuencia son la longitud de onda y la velocidad del sonido.

La longitud de onda es la distancia que recorre una onda para realizar un ciclo completo de oscilación, se la representa con la letra griega lambda (λ), y es medida en metros.

La velocidad del sonido determina que tan rápido viaja una onda por el medio de propagación, y se la representa por la letra c. Esta velocidad es bastante constante, en el caso del aire varía dependiendo de la temperatura que se tenga. Dos valores de referencia muy utilizados son la velocidad de 344 m/s a 21°c, y de 331 m/s a 0°C.

Estos tres parámetros se relacionan mediante la siguiente fórmula:

$$\lambda = \frac{c}{f}[\text{m}]$$

Ec. 3

Relación matemática entre longitud de onda, frecuencia y velocidad del sonido.

2.1.1.3. Fase

Si bien la fase es una propiedad del sonido, sus efectos son más perceptibles cuando se analiza la manera en que una forma de onda interactúa con otra.

Se puede decir que una forma de onda está completamente en fase con otra cuando tienen 0° de separación temporal, es decir que las dos señales se encuentran sumando sus componentes armónicas; el otro caso extremo es cuando las dos señales tienen un desfase de 180°, si las señales son exactamente las mismas la señal resultante será una cancelación completa de ambas, mientras que si las señales son ligeramente distintas existirá una cancelación grande de componentes armónicos de las estas teniendo como resultado una señal pobre en componentes armónicas. También pueden existir ligeros desfases menores a 180° que de igual manera produce una cancelación de armónicos, la cual va aumentando a medida que se acerca a un desfase de 180 °.

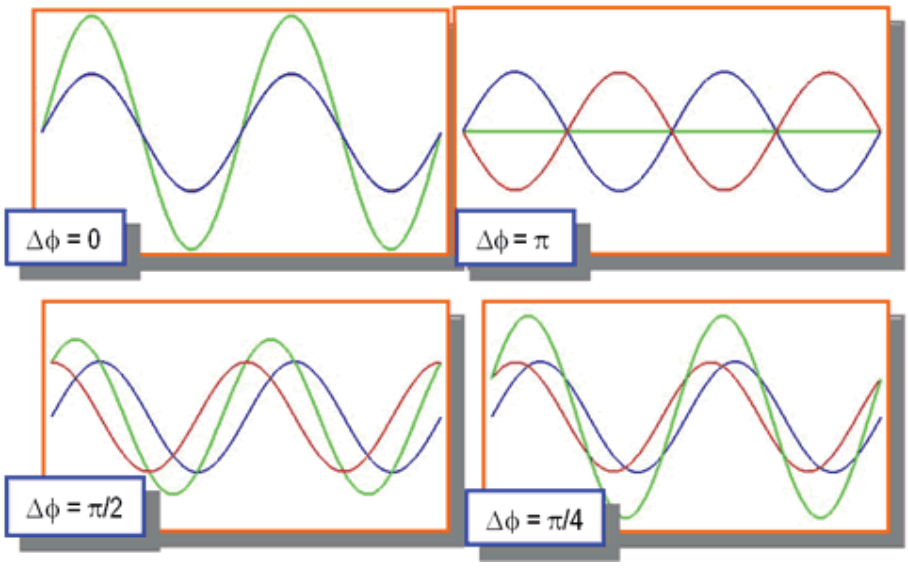


Fig. 1

Desfases de señales a distintos ángulos.

Tomado de: http://www.inacap.cl/data/sede_virtual/Ayudantias/Optica/Html/2_6.html

2.1.1.4. Espectro de Frecuencia y Timbre

El espectro de frecuencia es la combinación de varias ondas sonoras que se perciben como un sonido único, por lo tanto se puede analizar la relación armónica de una señal determinada. El espectro de frecuencia indica la manera como interactúa la fundamental de una señal con sus demás armónicos de forma cuantitativa, mientras que el timbre es de tipo cualitativo y hace referencia a cómo ese sonido es percibido por el oyente. Gracias al timbre es que el ser humano puede diferenciar entre distintos instrumentos musicales, por ejemplo, si hay varios instrumentos ejecutando la misma nota, a la misma altura tonal, y al mismo tiempo, se puede identificar a cada uno de los instrumentos, esto se debe a que cada uno tiene su propio timbre.

Gracias a que la relación armónica de las señales es distinta es que se pueden diferenciar distintos instrumentos y sonidos. Los sonidos armónicos están formados por frecuencias múltiplo enteros de las fundamental, denominados armónicos. Dependiendo de la amplitud del armónico con respecto a la fundamental se obtendrá un timbre diferente, aunque también existen señales que no son múltiplos enteros denominados inarmónicos.

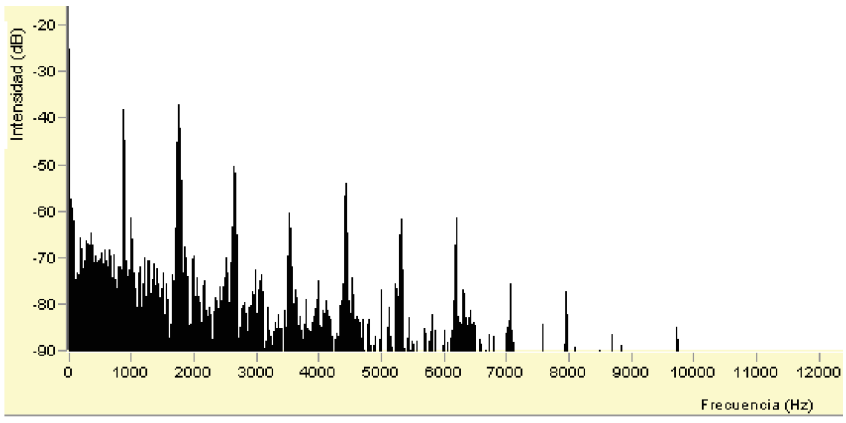


Fig. 2

Espectro de frecuencia nota A de una flauta.

Tomado de: http://www.uhu.es/ondas/proguia/ondasuper/ondasuper.html

En los diferentes sonidos que existen, los armónicos no se comportan de igual manera, de hecho en ciertas señales predominan unos más que otros. En algunos casos son los armónicos pares (2, 4, 8,16, etc.) los que predominan mientras que en otros son los armónicos impares (3, 5, 7, 9, etc.), en el caso de la música occidental se los estudia como Armónicos Naturales, los cuales toman una relación con respecto a la nota fundamental. A continuación se presenta un ejemplo tomando como nota fundamental a C1.

Tabla 1

Serie natural de armónicos.

Armónico N°	Frecuencia (Hz)	Nota	Intervalo
1	32.7	C1	Tono fundamental
2	65.4	C2	1 octava (consonante)
3	98.1	G2	1 octava + 1 quinta (consonante)
4	130.8	C3	2 octavas consonante
5	163.5	E3	2octavas + 1 tercera mayor (consonante)
6	196.2	G3	2 octavas + 1quinta (consonante)
7	228.9	Bb3	2 octavas + 1séptima menor (disonante)
8	261.4	C4	3 octavas (consonante)
9	293.7	D4	3 octavas + 1 segunda mayor disonante
10	329.6	E4	3 octavas + 1 tercera mayor (consonante)
11	369.9	F#4	3 octavas +1 cuarta aumentada (disonante)
12	392	G4	1 quinta justa (consonante)
13	440	A4	3 octavas + 1 sexta mayor (consonante)
14	466.1	Bb4	3 octavas + 1 séptima menor (disonante)
15	490.5	B4	3 octavas +1 séptima mayor (disonante)
16	523.3	C5	4 octavas (consonante)

Es importante no solo conocer los armónicos que hay, sino también saber que percepción subjetiva causan en el ser humano. La nota fundamental da la

percepción de altura tonal, los armónicos pares 2, 4, 8, 16 ayudan a definir mejor la percepción de altura tonal, los armónicos 3, 6, 9, 12 ayudan a definir sonidos de tipo nasal y de mayor calidez, los armónicos 5 y 10 ayudan con la idea de profundidad y redondez, finalmente los armónicos 7, 11, 13, 14 y 15 producen disonancia creando sonidos más duros y estruendosos.

2.1.1.5. Envolvente Temporal

Un factor muy importante en el sonido es la envolvente temporal de las señales, que es el comportamiento temporal de la intensidad de cada una de las componentes armónicas de una señal. Eso en el caso de intensidad, pero en un caso más general la envolvente temporal es una propiedad en tiempo de algún parámetro deseado.

Se puede separar en dos partes a la envolvente temporal, la primera es de tipo global y hace referencia al comportamiento de la señal en términos de amplitud a lo largo de un tiempo determinado y está ligado a conocer cómo se produce el sonido, la fuente que lo produce y cómo a esta se la manipula para producir dicho sonido, la segunda es de tipo específico ya que analiza el comportamiento de cada una de las componentes de la señal y analiza el decaimiento de cada una de estas a lo largo del tiempo, ya que este decaimiento se produce rápido en altas frecuencias y lento en bajas frecuencias.

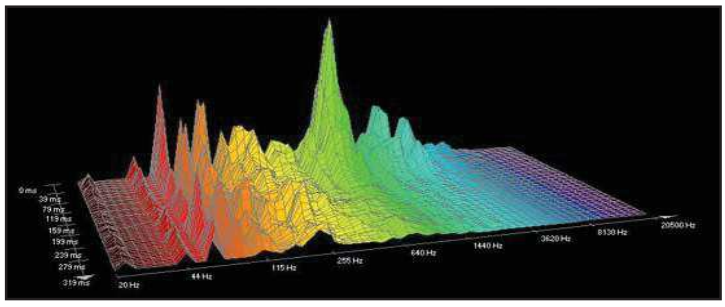


Fig. 3

Espectro de frecuencia y envolvente temporal del sonido de “madera” de tambores.

Tomado de: http://www.eumus.edu.uy/docentes/maggiolo/acuapu/tbr.html

2.2. Audio

2.2.1. Audio Analógico

El audio analógico es la representación eléctrica del sonido, donde la amplitud es análoga al voltaje. La señal obtenida de un sonido es de tipo continua, esto quiere decir que es una representación exacta del sonido que se encuentra en el entorno acústico y con todas las variaciones que esta presenta.

Este proceso de transformar energía acústica en voltaje se lo realiza a través de un transductor electro-mecano-acústico, en este caso el micrófono, que transforma las variaciones de presión existentes en el medio acústico en variaciones de voltaje. A esta señal se la puede registrar en distintos medios, posterior a esto se puede realizar el proceso contrario utilizando un transductor electro-mecano-acústico, en este caso un altavoz para reproducir la señal que se ha grabado.

2.2.2. Audio Digital

El audio digital es la representación del sonido en términos de números y/o dígitos. Los sistemas digitales utilizan la numeración binaria que viene dada por el uso de los números 0 y 1. La razón de utilizar lenguaje binario es que eléctricamente resulta fácil codificar los 0 y 1, dando así un valor de 1 a niveles de tensión de 5V y un 0 para 0V, lo que ayuda a evitar de mejor manera el ruido que se produce en sistemas análogos.

En el audio digital el principal punto a considerar es el muestreo de la señal, proceso en el cual se remplaza la señal original por una señal formada por una serie de muestras tomadas en intervalos determinados. En el dominio acústico y en el audio analógico las señales varían de forma continua, y si bien el oído humano es una herramienta que distingue muy bien las diferencias de amplitud y frecuencia, existen variaciones que no es capaz de distinguir, por esta razón tiene sentido el proceso de muestreo de una señal.

La frecuencia con la cual se toma las muestras se la denomina frecuencia de muestreo, la cual representa la cantidad de muestras que se tomarán en un tiempo determinado, como por ejemplo si se tiene una frecuencia de muestreo de 44100 Hz entonces se tomará este número de muestras en un segundo. Mientras más alta sea la frecuencia de muestreo mayor fidelidad tendrá el sonido de la señal digital. Existe un criterio obligatorio al momento de muestrear audio, y va ligado al teorema de Nyquist el cual dice que: La frecuencia de muestreo debe ser mayor que el doble de la máxima frecuencia presente en la señal a muestrearse.

Luego de un proceso de muestreo de la señal se debe digitalizar a esta, esto quiere decir almacenar a cada muestra como un número binario, para lo cual se necesita un convertidor análogo-digital el cual transforma valores de voltaje en números binarios. Mientras mayor sea la cantidad de *bits* se utilicen en la digitalización de la señal se podrá representar de mejor manera los cambios de amplitud de la señal deseada. Para saber el rango dinámico que se obtiene con una cantidad de bits determinada se multiplica al número de *bits* utilizados por 6, por ejemplo si se usa 16 bits el rango dinámico que se tiene es de 96 dB.

2.3. Síntesis Sonora

2.3.1. Introducción a la Síntesis Sonora

Según el diccionario síntesis se define como: “composición de un todo mediante la reunión de sus partes”. [10]

En dicho caso, un sintetizador vendría a ser un equipo que logre realizar este proceso de síntesis, que como indica la definición, genera un sonido complejo mediante la combinación de elementos simples. En el caso de la síntesis sonora, estos elementos simples son las componentes de síntesis que se encuentran dentro del circuito del sintetizador, o de manera virtual en un software determinado. La síntesis permite por un lado generar sonidos nuevos y por otro emular sonidos existentes, como el de los instrumentos musicales.

2.3.2. Componentes Eléctricos de la Síntesis de Sonido

Los componentes eléctricos de la síntesis de sonido son cinco: VCO (osciladores controlados por voltaje), VCF (filtros controlados por voltaje), VCA (amplificadores controlados por voltaje), LFO (osciladores de baja frecuencia), ADSR (generadores de envolvente).

2.3.2.1. Osciladores Controlados por Voltaje (VCO)

El VCO es el componente de la síntesis que genera señales periódicas. Este componente permite ajustar parámetros como la frecuencia, que en algunos casos se puede expresar en Hz, o en otros este parámetro puede venir expresado en notas musicales con sus respectivas subdivisiones de tono, semitono y en casos más especiales centésimas de tono. También como parámetro se puede ajustar la forma de onda de la oscilación, la cual puede ser cuadrada, sinusoidal, triangular, y hasta señales más complejas como ruido rosa o ruido blanco.

2.3.2.2. Filtros Controlados por Voltaje (VCF)

Como su nombre lo indica, los VCF son filtros, componentes muy utilizados para modelar sonidos a partir de señales complejas. La función de este elemento es detener o permitir el paso de ciertas componentes de frecuencias de una señal determinada. Estos filtros también presentan varios parámetros como la frecuencia o frecuencias en la cual actuará, el tipo de filtraje (filtro de tipo pasa altos, pasa bajos, rechaza banda, etc.), la pendiente del filtro, y la resonancia de este.

2.3.2.3. Amplificadores Controlados por Voltaje (VCA)

Estos componentes son aquellos que permiten dar una ganancia a la señal originada por los VCO. Presentan un único parámetro que es la ganancia de la señal, en algunos métodos de síntesis se llegan a usar varios VCA para controlar el nivel de varios VCO para la generación de sonidos.

2.3.2.4. Osciladores de Baja Frecuencia (LFO)

Como lo indica su nombre, los LFO son osciladores que producen frecuencias menores a 20 Hz. Este dispositivo es utilizado en la síntesis para modular parámetros de los demás componentes, por ejemplo si se modula a la amplitud se obtiene un efecto de tremolo, en el caso de modular frecuencia se tiene un *vibrato*. Este componente presenta algunos parámetros como son la forma de onda de oscilación (sinusoidal, triangular, cuadrada, etc.), la frecuencia de la oscilación, y la profundidad de la modulación (*depth*).

2.3.2.5. Generadores de Envolvente (ADSR)

Los ADSR son componentes que modelan la evolución en el tiempo de algún parámetro. La mayoría de los generadores de envolvente presentan cuatro parámetros importantes que son: ataque, decaimiento, sostenimiento y relevo, de ahí sus siglas ADSR. El primero determina el tiempo inicial de la curva temporal del parámetro requerido, el segundo es el tiempo en que el parámetro va de su punto máximo de ataque a un punto de estabilidad, el tercero es el tiempo que se mantiene estable y el cuarto y último es el tiempo donde pasa de su punto estable a su punto final, esta curva que se forma se puede utilizar en cualquier opción de la síntesis sonora aunque se lo utiliza en mayor medida en la amplitud.

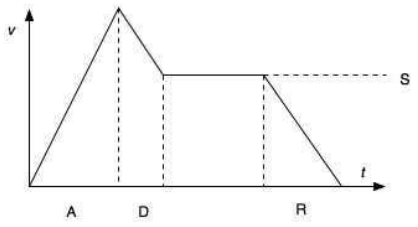


Fig. 4

Curva de Envolvente (ADSR).

Tomado de: http://hackmeopen.com/category/rockit-synth/programming/

2.3.3. Principales Métodos de Síntesis Sonora

En este apartado se conocerá sobre los métodos de síntesis más conocidos, con una explicación de cada uno de ellos.

2.3.3.1. Síntesis Aditiva

El método de síntesis aditiva se basa en la mezcla o suma de varios sonidos para generar uno nuevo. La síntesis aditiva más simple se basa en la suma de osciladores senoidales, donde cada uno de estos representa a una componente armónica de la señal que se desea generar. En este tipo de síntesis se utiliza un VCO un VCA y un ADSR por componente armónica para así modelar cada una, sumarlas y generar el sonido que se desea.

Los parámetros de los componentes de síntesis que se establecen en este tipo de método, pueden ser dados de dos maneras. La primera es de forma empírica en la cual el usuario ajusta los parámetros mediante sondeo hasta encontrar el timbre que se desea, esta forma es muy trabajosa pero se obtienen resultados con una gran calidad, además existe la posibilidad de generar una infinita variedad de sonidos nuevos. El otro método es basado en el análisis de señales ya existentes, si el sonido sintético y el real son idénticos el análisis funciona y se consigue el objetivo.

La síntesis aditiva se basa en el análisis de Fourier, gracias a esto se puede definir a un sonido en sus parciales o componentes armónicas y así encontrar un modelo adecuado para realizar una re síntesis. Lo interesante de este tipo de síntesis es la gran cantidad de nuevos sonidos que se pueden crear al variar sus parámetros basándose en sonidos ya existentes y estudiados.

Un caso particular pero también utilizado en síntesis aditiva es la que se realiza con ondas complejas, de esta manera a través de formas de ondas cuadradas, triangulares, entre otras, se consiguen espectros ricos en frecuencia con pocos VCO.

2.3.3.2. Síntesis Substractiva

Este método de síntesis parte de una forma de onda compleja, la cual es pasada por un filtro para modificar su espectro, atenuando ciertos componentes de frecuencia y reforzando otros. La señal de partida debe ser lo más rica en contenido armónico posible, por eso se recomienda ruido blanco, ruido rosa u otros. La ventaja de este método de síntesis es que a diferencia del anterior utiliza pocos VCO, ya que en su arquitectura puede existir uno solo. Lo que si es requerido en este tipo de síntesis es varios VCF para poder filtrar a la señal compleja y modelar la señal deseada, además posee un VCA para amplificar la señal y puede también incorporar un ADSR para una síntesis más fina.

2.3.3.3. Síntesis por Modulación de Frecuencia

En la síntesis por modulación de frecuencia existen dos tipos que se deben conocer, estos son el FM y el AM.

2.3.3.3.1. Síntesis por Modulación de Frecuencia FM

En este tipo de síntesis se tienen dos señales, una se la denomina moduladora y a la otra se la denomina portadora. La frecuencia de la señal portadora es modulada por la amplitud de la señal moduladora. En este caso la señal resultante presenta variaciones de frecuencia, donde se forman componentes espectrales por encima y debajo de la señal original. Pero pueden existir síntesis más complejas como que una señal moduladora afecte a varias portadoras o el caso contrario, este tipo de síntesis es muy útil para conseguir timbres innovadores más que para la simulación de sonidos de instrumentos musicales.

2.3.3.3.2. Síntesis por Modulación de Frecuencia AM

En este caso también existe la señal moduladora y portadora, la diferencia es que la señal resultante es afectada en su amplitud y no en la frecuencia, por eso la señal moduladora afecta con su frecuencia a la amplitud de la portadora. Dentro de este tipo de síntesis existe un método denominado modulación de

anillo, en el cual se obtiene una señal resultante a través de una modulación AM común, pero se saca una copia de este resultado el cual vuelve a ser modulado de vuelta y el resultado de esta pasa por el mismo proceso y así sucesivamente.

2.3.3.4. Síntesis Granular

Este tipo de síntesis se basa en dividir a una señal en fragmentos cortos de tiempo y reacomodar a estos de forma diferente. También se pueden generar sonidos muy cortos (de unos 25 milisegundos), y ubicarlo en distintas posiciones para generar un sonido nuevo, siendo este la suma de todos estos "granos". Este método de síntesis plantea sonidos interesantes y nuevos, y no es muy útil para la simulación de sonidos, sino más bien para la creación sonora.

2.3.3.5. Síntesis por Tabla de Onda

Este método de síntesis es diferente a los vistos anteriormente, ya que a diferencia de utilizar VCO y otros componentes de síntesis para generar sonidos, este utiliza un circuito que reproduce muestras de sonidos (*samples*), con distinta frecuencia dependiendo de la altura tonal que se desee interpretar. Este tipo de síntesis ayuda a tener un gran realismo sonoro de instrumentos existentes.

2.3.3.6. Otros Métodos de Síntesis

En este apartado solo se enlistarán otros métodos de síntesis, que si bien son utilizados no son tan comunes como los vistos anteriormente.

2.3.3.6.1. Síntesis por Distorsión de Fase

Método de síntesis que se basa en el desfase de dos o más señales para producir distorsión y generar un nuevo timbre.

2.3.3.6.2. Síntesis Cruzada

Método de síntesis que se basa en analizar la evolución de ciertos parámetros de un sonido y aplicárselos a otro.

2.3.3.6.3. Síntesis por Modelado Físico

Método de síntesis que consiste en el análisis del comportamiento físico del instrumento, y en la emulación de este por medio de ecuaciones matemáticas.

2.4. Sintetizadores

En este apartado conoceremos más sobre dos grandes divisiones de los sintetizadores, por un lado los sintetizadores analógicos y por otro los digitales.

2.3.4. Sintetizadores Analógicos

Los sintetizadores analógicos salieron al mercado en los años ‘60, pero su acogida y gran difusión empezó en la década de 1970. Estos dispositivos presentaban una gran ventaja ya que todos sus parámetros de generación y modificación sonora podían ser controlados por medio de voltaje. Gracias a esto, por ejemplo, se podía controlar a un oscilador denominado VCO a través de presionar una tecla, y así generar una onda con una frecuencia determinada.

Los sintetizadores analógicos constan de varios módulos donde cada uno cumple una función, además ciertos módulos pueden controlar a otros a través de conexiones entre estos. Por lo general la cadena de un sintetizador empieza con un VCO el cual produce una onda con una frecuencia determinada, pueden existir varios VCO para sumar estas señales y obtener una señal más rica en su espectro armónico. Luego se la conecta a un VCF para modelar el espectro obtenido con los VCO. Un segmento importante de los sintetizadores es el generador de envolvente, ya que es muy útil para tener un mejor resultado de la síntesis. Otro de los módulos es el LFO el cual como se ha visto anteriormente puede controlar a parámetros de los demás componentes de síntesis y como último modulo el VCA utilizado para amplificar las señales.

Los sintetizadores analógicos son inestables debido a sus circuitos. Si bien esto no es lo óptimo debido a que produce desafinación y cambia el resultado según las condiciones del ambiente en el que se encuentre, es una característica apetecible, propia y muy utilizada en la composición musical.

Fig. 5

Sintetizador analógico Korg PS-3100.

Tomado de: http://www.kimoibento.com/sintetizador-korg-ps-3100/

2.3.5. Sintetizadores Digitales

A la tecnología de control por medio de voltaje le siguió otra, en la cual todos los parámetros trabajan y se los controla de forma digital y almacena en una memoria RAM, aunque también existe aquellos que sus componentes son analógicos pero controlados digitalmente, a esto se los denomina sintetizadores híbridos.

En la actualidad la mayoría de los sintetizadores son digitales tanto en generación como en control, lo que ayudado a realizar nuevos métodos de síntesis sonora. Debido a que los componentes en este caso son digitales sus siglas se modificaron como son el DCO (oscilador controlado digitalmente),

DCF (filtro controlado digitalmente) y DCA (amplificador controlado digitalmente).

Gracias a esto se pueden utilizar nuevas formas de ondas y aumentar la gama de sonidos que se pueden generar, así como el uso de sonidos previamente grabados y reproducidos posteriormente como en la síntesis por tabla de onda.

Fig. 6

Sintetizador Digital MOTIF XF8.

Tomado de: http://www.musisol.com/sintetizadores-yamaha/3590-yamaha-motif-xf8-sintetizador.html

2.3.6. Sintetizadores Virtuales

Lo visto anteriormente tanto en el sintetizador analógico y digital, ocurre con el caso de los sintetizadores virtuales. En estos se presentan componentes de síntesis como VCO, VCA, VCF, LFO y ADSR, con la diferencia que estos son incorporados en un computador a través de la programación de los mismos. Este tipo de síntesis ocupa el procesador propio de la máquina para su funcionamiento, por lo que la calidad del audio generado por este, en muchos casos será dependiente de las capacidades de procesamiento del computador en que se los use.

En los últimos años los sintetizadores virtuales han tenido gran acogida, ya que presentan grandes ventajas frente a los sintetizadores no virtuales, ya que permiten incorporar dentro de un mismo dispositivo (un solo computador) gran

cantidad de sintetizadores, ahorrando espacio en *hardware* y con un mejor resultado económico para los consumidores al no tener que adquirir gran número de dispositivos físicos.

Fig. 7

Sintetizador Virtual Hybrid.

Tomado de: http://www.avid.com/es/products/Hybrid

2.5. Max/MSP

Max/MSP nace a manos de Miller Puckette, donde su aplicación se basaba en formar un entorno gráfico para la composición de música por computadora. En sus inicios el programa era conocido solo como Max, actualmente su nombre completo es Max/MSP/Jitter, el cual fue adquirido por la empresa Cycling ’74, propiedad de David Zicarelli en el año 1999.

El comienzo de este programa se remonta a los años ‘80, cuando Miller Puckette desarrolla un programa denominado “Patcher”, este software fue diseñado para un computador Macintosh del organismo francés IRCAM (Instituto de Investigación y Coordinación Acústica/Música), el programa era

una herramienta capaz de ayudar en la composición musical por medio de ordenadores. Más adelante en el año 1989 el IRCAM desarrolla una versión de Max para sistema Linux, llamado Max/FTS y precursor del Max/MSP actual, esta denominación MSP se debe a una serie de extensiones que amplían el campo de aplicación del programa, ya que de solo utilizarse protocolo MIDI se podía trabajar en el dominio del audio.

En sus inicios Max/MSP fue diseñado como un sistema de control para *samplers*, sintetizadores, efectos, entre otros, pero gradualmente fue incorporando mayor cantidad de herramientas y accesorios los cuales permitían procesar de manera poderosa señales en tiempo real. Posteriormente, al programa se le añade en el año 2003 una sección de procesamiento de matrices, imágenes y videos denominada Jitter, con lo cual este programa se convierte en un software capaz de manejar datos MIDI, información de audio y además video, todo esto simultáneamente y en tiempo real.

Este software se desarrolló bajo el lenguaje C, se programa a través de objetos ya incorporados o que se pueden programar o modificar dependiendo del uso que se les dé. Muchas de las opciones de cada objeto pueden ser modificadas en el denominado *inspector*, donde se pueden realizar cambios de funcionamiento como de apariencia del objeto deseado, además si son necesarios cambios más relevantes se pueden programar o variar la programación de objetos existentes en el leguaje C++. Estos objetos pueden conectarse entre sí para realizar estructuras u operaciones más complejas.

Estos objetos se los puede agrupar en los llamados *patcher* de Max/MSP/Jitter, los cuales serán archivos de tipo *patch*, donde se pueden conectar objetos entre si por medio de una conexión virtual de cables, lo que conforma un flujo determinado de la señal. Los cables en la interfaz gráfica del programa son de distintos colores dependiendo del tipo de señal que este circulando por ellos, los cables de color negro se utilizan para conducir información solo de datos numéricos o de caracteres, los cables amarillos se utilizan para conducir señales de audio, los cables de color verde transmiten información visual o de

matrices relacionadas con Jitter, y los cables de color blanco y negro intercalado son utilizados para información MIDI. También existen los *subpatch*, en los cuales se pueden realizar las mismas operaciones que en un *patch*, y es utilizado para formar estructuras mayores dentro de ellos y realizar operaciones complejas y así condensar el número de objetos hasta llegar al *patch* final.

Además de permitir la creación de estructuras mayores con gran capacidad de procesamiento, Max/MSP/Jitter puede añadir *externals* o *plug-ins* para su funcionamiento, las cuales son extensiones que mejoran y amplían las funciones del programa sin afectar a las internas del mismo. Se pueden utilizar *plug-ins* de aplicaciones de grabación o edición de audio, de igual forma en video, así como *externals* programados por usuarios con especificaciones determinadas.

2.6. Herramientas utilizadas de Max/MSP

En el siguiente apartado se listará y explicará de forma concreta, los objetos utilizados en el trabajo de construcción del sintetizador espectral, el uso y conexión de los mismos será definido más adelante en la sección de diseño y programación.

- ***~:** Objeto que realiza la operación de multiplicación entre señales de audio.

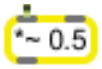


Fig. 8

Objeto *~.

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/times~.html

- **+~:** Objeto que realiza la operación de adición entre señales de audio.

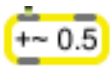


Fig. 9

Objeto +~.

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/plus~.html

- ***/~:*** Objeto que realiza la operación de división entre datos de audio.

Fig. 10

Objeto /~.

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/div~.html

- **>=:** Objeto que permite comparar dos números. Si el número en la entrada izquierda del objeto es mayor o igual al de la derecha en la salida del objeto se envía un mensaje de 1, caso contrario un mensaje de 0.

Fig. 11

Objeto >=.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/greaterthaneq.html

- ***poly:*** Objeto que permite manejar datos de polifonía, es decir permite la ejecución de varias notas musicales en tiempo simultáneo.

Fig. 12

Objeto *poly*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/poly.html

- ***pack*:** Objeto que permite tomar datos separados y combinarlos en una lista.

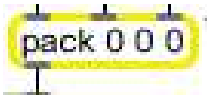


Fig. 13

Objeto *pack*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/pack.html

- ***route*:** Objeto que permite tomar varios mensajes y enviarlos hacia otros objetos en un orden específico.

Fig. 14

Objeto *route*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/route.html

- ***unpack*:** Objeto que permite separar elementos de una lista y enviar a estos por una salida independiente.

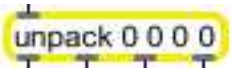


Fig. 15

Objeto *unpack*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/unpack.html

- ***mtof*:** Objeto que convierte datos de nota MIDI en datos de frecuencia en Hz.

Fig. 16

Objeto *mtof*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/mtof.html

- ***ctlin*:** Objeto que es asignado a un control y canal MIDI específico, el cual puede recibir valores de control y enviarlos hacia su salida.

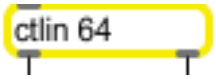


Fig. 17

Objeto *ctlin*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/ctlin.html

- ***adsr~*:** Objeto que genera una envolvente temporal en función de datos de ADSR ingresados.

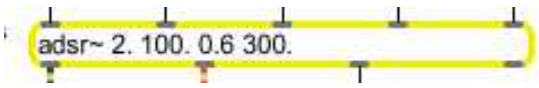


Fig. 18

Objeto *adsr~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/adsr~.html

- ***urn*:** Objeto que permite generar datos numéricos aleatorios con un rango de límite específico.

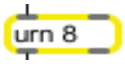


Fig. 19

Objeto *urn.*

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/urn.html

- ***gate*:** Objeto que permite el paso o no en función de un dato de control, de un mensaje recibido por una salida determinada del mismo.

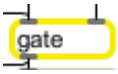


Fig. 20

Objeto *gate.*

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/gate.html

- ***gswitch:*** Objeto gráfico que sirve como *switch* controlado por un dato de control en su entrada, y permite seleccionar qué dato de sus dos entradas rutear hacia su única salida.

Fig. 21

Objeto *gswitch.*

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/gswitch.html

- ***gswitch2:*** Objeto gráfico que sirve como *switch* controlado por un dato de control en su entrada, y permite seleccionar a cual de sus dos salidas rutear la información que llega a su única entrada.

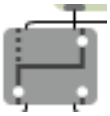


Fig. 22

Objeto *gswitch2.*

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/gswitch2.html

- ***cycle~:*** Objeto que se encarga de generar ondas senoidales.

Fig. 23

Objeto *cycle~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/cycle~.html

- ***tri~:*** Objeto que se encarga de generar ondas triangulares.

Fig. 24

Objeto *tri~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/tri~.html

- ***saw~:*** Objeto que se encarga de generar ondas tipo diente de sierra.

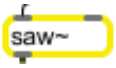


Fig. 25

Objeto *saw~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/saw~.html

- ***rect~:*** Objeto que se encarga de generar ondas cuadradas.

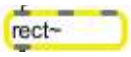


Fig. 26

Objeto *rect~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/rect~.html

- ***makenote:*** Objeto que permite generar un mensaje de tipo MIDI el cual lleva información de: mensaje de nota encendida (*note on*), mensaje de velocidad (*velocity*) y mensaje de nota apagada (*note off*).

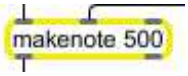


Fig. 27

Objeto *makenote.*

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/makenote.html

- ***biquad~:*** Objeto que implementa un filtro utilizando la siguiente ecuación:

$$y[n] = a0 * x[n] + a1 * x[n-1] + a2 * x[n-2] - b1 * y[n-1] - b2 * y[n-2]$$

Ec. 4

Ecuación utilizada por el objeto *biquad~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/makenote.html

Se puede especificar los coeficientes A0, A1, A2, B1 y B2 como señales o como números.

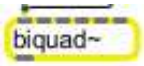


Fig. 28

Objeto *biquad~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/biquad~.html

- ***filtergraph~:*** Objeto gráfico que si bien no procesa señales de audio por si solo permite al usuario generar valores de coeficiente para el filtro del objeto *biquad~.*

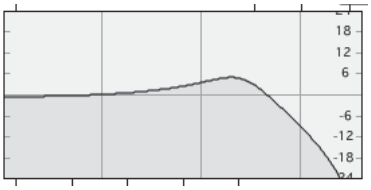


Fig. 29

Objeto *filtergraph~.*

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/filtergraph~.html

- ***pipe:*** Objeto que es utilizado para generar un *delay* en los datos (números o lista de números).

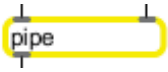


Fig. 30

Objeto *pipe*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/pipe.html

- ***spectroscope~:*** Objeto gráfico que es utilizado como espectrograma o sonograma para el análisis de señales.

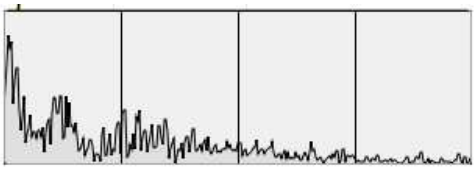


Fig. 31

Objeto *spectroscope~*.

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/spectroscope~.html

- ***lcd:*** Objeto gráfico que permite dibujar formas básicas mediante comandos de entrada o por medio de la interface gráfica.

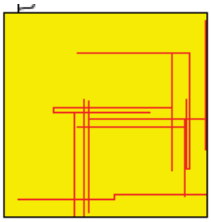


Fig. 32

Objeto *lcd*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/lcd.html

- ***ezdac~:*** Objeto gráfico con apariencia de botón. Enciende o apaga la salida de audio.

Fig. 33

Objeto *ezdac~*.

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/ezdac~.html

- ***gain~:*** Objeto gráfico que a través de un control deslizante permite ajustar la ganancia o nivel de una señal.

Fig. 34

Objeto *gain~.*

Tomado de: http://cycling74.com/docs/max5/refpages/msp-ref/gain~.html

- ***preset:*** Objeto gráfico que puede guardar y cargar parámetros de cualquier objeto especificado.

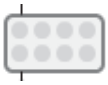


Fig. 35

Objeto *preset*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/preset.html

- ***pattrstorage****:* Objeto usado para guardar y cargar conjuntos de datos. Usado con el objeto *preset*, sirve para guardar y cargar archivos que incluyen datos de los parámetros de cualquier objeto especificado.

Fig. 36

Objeto *pattrstorage*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/pattrstorage.html

- ***autopattr:*** Objeto que al ser incluido dentro de un *patch* de Max/MSP , causa que automáticamente una gran cantidad de objetos sean especificados para guardar y cargar sus parámetros mediante el objeto *preset*.

Fig. 37

Objeto *autopattr*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/autopattr.html

- ***button:*** Objeto gráfico usado para gatillar otros mensajes o procesos.

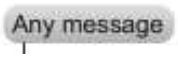


Fig. 38

Objeto *button*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/button.html

- ***kslider:*** Objeto gráfico de un teclado musical que permite recibir información de *velocity* y *pitch.* Así mismo permite al usuario ejecutar notas musicales sobre si, enviando mensajes de *velocity* y *pitch* en función de lo ejecutado.

Fig. 39

Objeto *kslider*.

http://cycling74.com/docs/max5/refpages/max-ref/kslider.html

- ***number:*** *O*bjeto gráfico de una caja de número, usado para mostrar, recibir o enviar números enteros.

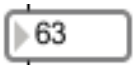


Fig. 40

Objeto *number*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/number.html

- ***flonum:*** *O*bjeto gráfico de una caja de número, usado para mostrar, recibir o enviar números de coma flotante.

Fig. 41

Objeto *flonum*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/flonum.html

- ***pictslider:*** Objeto gráfico que permite crear un control deslizante con una apariencia de una imagen guardada en la computadora.

Fig. 42

Objeto *pictslider*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/pictslider.html

- ***live.gain~:*** Objeto gráfico que a través de un control deslizante permite ajustar la ganancia o nivel de una señal, y visualmente muestra el nivel del sonido en una escala de dB.

Fig. 43

Objeto *live.gain~*.

Adaptado de: http://cycling74.com/docs/max5/refpages/m4l-ref/live.gain~.html

- ***live.dial:*** Objeto gráfico que sirve como un control deslizante circular, que envía un dato de control en función de su grado de rotación.

Fig. 44

Objeto *live.dial*.

Adaptado de: http://cycling74.com/docs/max5/refpages/m4l-ref/live.dial.html

- ***live.slider:*** Objeto gráfico que sirve como un control deslizante vertical.

Fig. 45

Objeto *live.slider*.

Adaptado de: http://cycling74.com/docs/max5/refpages/m4l-ref/live.slider.html

- ***live.menu:*** Objeto gráfico de menú que muestra una lista de palabras, las cuales son asociadas a una lista de números (empezando en cero), permitiendo así que el usuario seleccione una de las opciones y enviando un mensaje control asociada a esta.

Fig. 46

Objeto *live.menu*.

Adaptado de: http://cycling74.com/docs/max5/refpages/m4l-ref/live.menu.html

- ***inlet:*** Objeto que sirve dentro de un *patcher,* para recibir mensajes desde un *patch* exterior.

Fig. 47

Objeto *inlet*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/inlet.html

- ***outlet:*** Objeto que sirve dentro de un *patcher,* para enviar mensajes hacia un *patch* exterior.

Fig. 48

Objeto *outlet*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/outlet.html

- ***notein:*** Permite recibir mensajes de *note on* y *note off* de un controlador MIDI externo.

Fig. 49

Objeto *notein*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/notein.html

- ***tab:*** Objeto gráfico utilizado para crear interfaces de múltiple botón y de múltiple ventana.

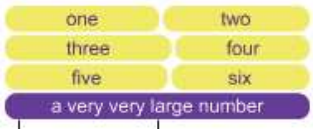


Fig. 50

Objeto *tab*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/tab.html

- ***fpic:*** Objeto gráfico usado para mostrar imagines guardadas en archivos gráficos externos al *patch.*

Fig. 51

Objeto *fpic*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/fpic.html

- ***bpatcher:*** Objeto gráfico usado para incluir un *subpatcher* dentro de una caja visible.

Fig. 52

Objeto *bpatcher*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/bpatcher.html

- ***panel:*** Objeto gráfico que permite crear fondos de forma rectangular con color, usado para crear interfaces gráficas.

Fig. 53

Objeto *panel*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/panel.html

- ***message:*** Objeto que es usado para mostrar y enviar mensajes hacia otros objetos.

Fig. 54

Objeto *message*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/message.html

- ***comment:*** Objeto que sirve para mostrar cualquier texto ingresado en el mismo.

Fig. 55

Objeto *comment*.

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/comment.html

- ***selector~:*** Objeto que tiene la capacidad de seleccionar una de las señales de audio presentes en sus varias entradas, hacia su única salida, en función de un dato de control ingresado.

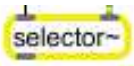


Fig. 56

Objeto *selector~*.

Adaptado de: http://cycling74.com/docs/max5/refpages/msp-ref/selector~.html

- ***maximum:*** Objeto que selecciona el dato máximo de una lista de datos ingresados.

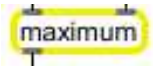


Fig. 57

Objeto *máximum.*

Adaptado de: http://cycling74.com/docs/max5/refpages/max-ref/maximum.html

3. Diseño y Programación

Para el desarrollo del proyecto, resultó necesario en cada etapa de diseño, hacer un balance entre las necesidades a cubrirse con la aplicación a programarse, las capacidades del entorno de programación que se estaba usando, y la facilidad de manejo del sintetizador; balance que buscaba cubrir los tres aspectos de forma equitativa para tener un resultado satisfactorio.

Si bien fue necesario cubrir los aspectos antes mencionados, el principal, y el que fue utilizado como punto de partida para el diseño, fueron las necesidades a cubrirse con la aplicación.

Tomando en cuenta que lo que se necesitaba programar era un sintetizador virtual de sonido por componentes espectrales, surge la necesidad de crear dichas componentes espectrales, y dentro de los métodos de síntesis conocidos, el más útil para este propósito es el de las síntesis aditiva. Es por este motivo que el diseño del sintetizador, así como su programación parte de la necesidad de incorporar un determinado número de osciladores con parámetros de control independientes en el sintetizador.

Una vez encontrado un punto de partida para el desarrollo del sintetizador (su necesidad principal), así como una idea clara del objetivo final al que se pretendía llegar, el siguiente paso fue conjugar estos dos puntos con las características del entorno de programación Max/MSP.

Conociendo la forma de funcionamiento de Max/MSP, la cual se basa en la programación orientada a objetos, se determinó desde un principio que el sintetizador sería diseñado y programado desde sus componentes más pequeños y sencillos, para que así, una vez logrados los componentes deseados y requeridos, mediante la interconexión de estos dentro de una interfaz gráfica amigable al usuario, se cumplan los objetivos de diseño y programación.

En función de toda la aclaración previa, se explicará el proceso de creación del sintetizador, a partir de los componentes más básicos y necesarios para este, es decir, los bloques de generación de sonido.

3.1. Bloques de Generación de Sonidos

El nombre de bloque de generación de sonido hace referencia a cada uno de los osciladores del sintetizador, junto con todos sus parámetros de control. La parte más importante de estos es su oscilador, precisamente el encargado de la generación de audio.

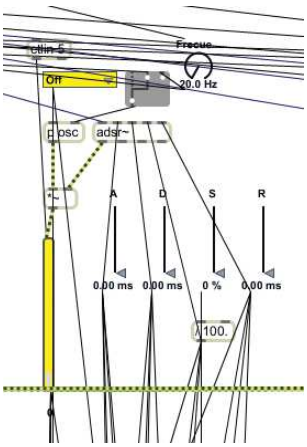


Fig. 58

Bloque de generación de sonidos.

Como se puede ver en la imagen, dentro de estos bloques de generación de sonidos, hay un *patch* encapsulado, el cual contiene a los verdaderos osciladores.

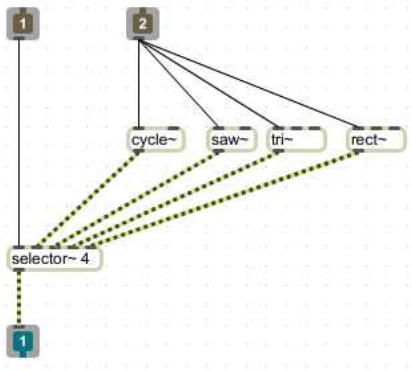


Fig. 59

Patch "p osc".

Dentro de este denominado *patch* "*p osc*", se encuentran los objetos encargados verdaderamente de la generación de señales, estos son *cycle~*, *saw~, tri~*, y *rect~*, los cuales producen una señal con forma de onda senoidal, diente de sierra, triangular y rectangular, respectivamente.

Si bien la síntesis aditiva típicamente combina varias señales de tipo senoidal, (es decir tonos puros que una vez sumados dan como resultado el espectro sintetizado), después de considerar que esta es una característica pensada principalmente para la función de emular fielmente espectros sonoros, y que esta es una aspiración del sintetizador pero no la única, sino que también se espera que sirva para la creación de nuevos sonidos, se decide agregarle la posibilidad de generar señales de tipo diente de sierra, triangular y rectangular, ampliando así el abanico de herramientas brindadas al usuario.

El *patch c*onsta también de dos *inlets* y un *outlet*. El primero de sus *inlets* debe ingresar una señal de control para que el objeto selector~ determine que señal de audio de las que ingresan en sus entradas debe ser enviada a la salida. El segundo *inlet* ingresa el dato de frecuencia hacia los bloques *cycle~*, *saw~, tri~*, y *rect~*, indicándoles la frecuencia a la que deben generar las señales. Por último el *outlet* encargado de sacar la señal generada hacia afuera del *patch.*

Dentro del bloque de generación de sonidos también se encuentra la sección de generación de envolventes de tiempo para la amplitud, el objeto denominado *adsr~*.

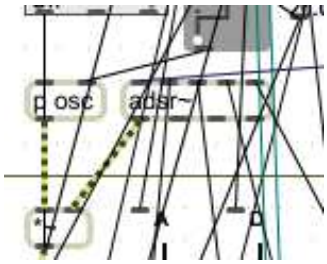


Fig. 60

Objeto *adsr~*.

Los parámetros de ataque, decaimiento, sostenimiento, y relevo son ingresados al objeto ***adsr~*** mediante los cuatro objetos *live.slider* con nombre A, D, S, y R, respectivamente, y la determinación de los rangos de valores entre los cuales oscilan estos *sliders* será explicada más adelante.

Como ya se menciono anteriormente, este bloque está siendo usado para aplicarle una envolvente de tiempo a la amplitud, y trabaja de la siguiente manera. El valor de *velocity*, correspondiente a dicho bloque de generación de sonidos es enviado a la primera entrada del objeto ***adsr~***, y de esta forma se gatilla el proceso de generación de envolvente con los parámetros ingresados al bloque. El valor de *velocity* una vez que gatilla el proceso, es tomado para que se le aplique la determinada envolvente de tiempo, y es esta la señal que sale del objeto ***adsr~***. Una vez esta señal sale de ***adsr~***, pasa al objeto ***~**, donde es multiplicada con la señal entregada por el *outlet* del *patch "p osc*".

En resumen, la señal entregada por este objeto ***~**, es la señal de audio generada con la forma de onda seleccionada en el *patch* "*p osc*", y aplicada una envolvente de tiempo que considera tanto parámetros de ataque, decaimiento, sostenimiento, y relevo, que pueden ser ingresados por el usuario, así como el *velocity* generado en la ejecución musical del usuario.

Todo esto pensado para brindar facilidad de ejecución, y opciones de control del sintetizador al usuario.

La señal antes mencionada, luego es enviada a un objeto *gain~*, mediante el cual el usuario puede controlar la amplitud dicha señal, y así tener control de la amplitud de cada bloque de generación de sonidos en la señal final generada por el sintetizador.

Dentro del bloque de generación de sonido también se encuentran los objetos: *live.menu*, los cuales permiten al usuario seleccionar la forma de onda a generarse, y envían una señal de control al *patch "p osc*" para este propósito; el objeto *live.dial*, el cual permite al usuario ingresar el dato de frecuencia para el bloque de generación de sonidos; y un objeto *gswitch*, cuyo propósito será explicado más adelante.

La determinación de los rangos de valores de los parámetros del sintetizador se desprenden en algunos casos de análisis de datos realizados, en otros, de aspectos que brindan facilidad de uso del sintetizador, y en otros, de aspectos de diseño de la interface gráfica.

En el caso de los valores permitidos en los parámetros de ataque, decaimiento, sostenimiento y relevo, estos surgen de un análisis de espectro tridimensional realizado a algunos instrumentos musicales. Este análisis se lo realizó usando el espectrograma tridimensional encontrado en la función de *Meter Bridge* del *plug-in* Ozone 5 Advanced, de la compañía Izotope, el cual permite analizar el espectro de los sonidos deseados en tiempo real en tres dimensiones.

Este análisis se lo aplicó a varios instrumentos, pero de ellos el más útil fue el realizado con *samples* de notas largas de un violín. Esto dado que el violín tiene una envolvente de tiempo natural bastante compleja. Para el análisis se usaron diferentes notas musicales de diferentes octavas, pero la considerada como más representativa fue la de A4, es decir la nota A de 440 Hz.

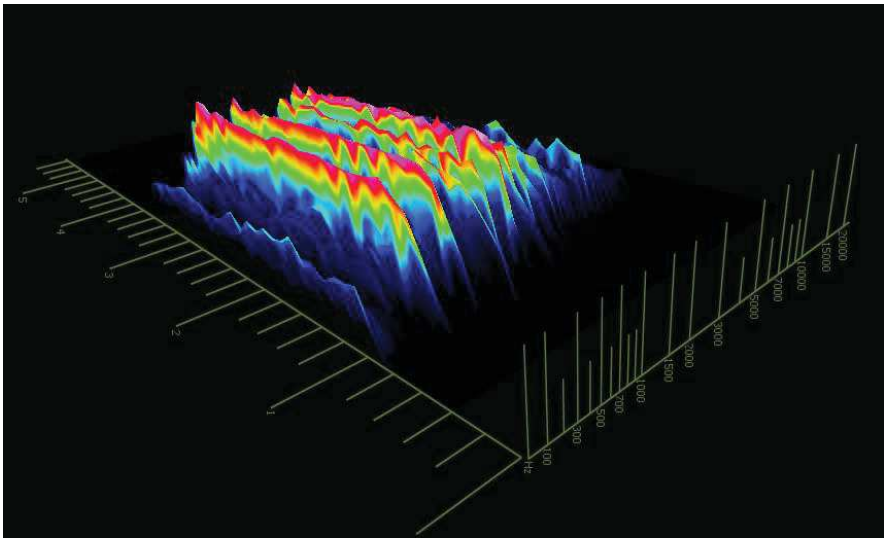


Fig. 61

Vista diagonal del espectro de violín ejecutando nota A4.

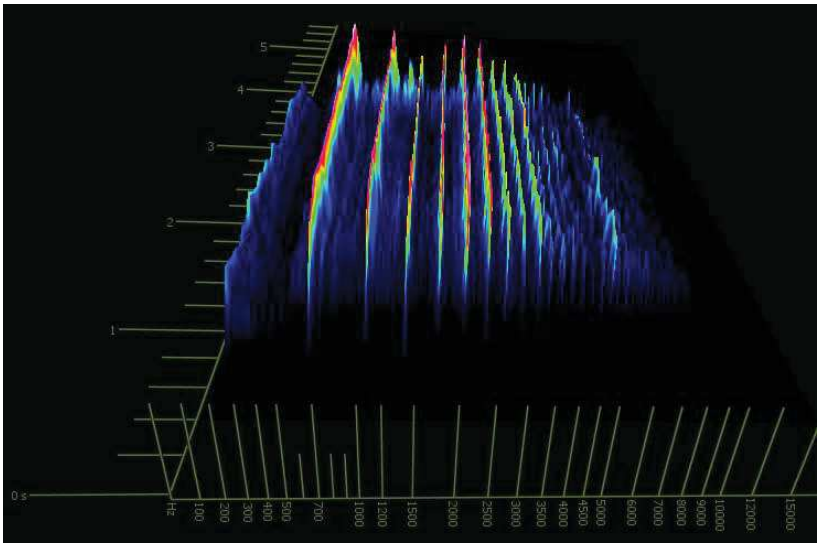


Fig. 62

Vista frontal del espectro de violín ejecutando nota A4.

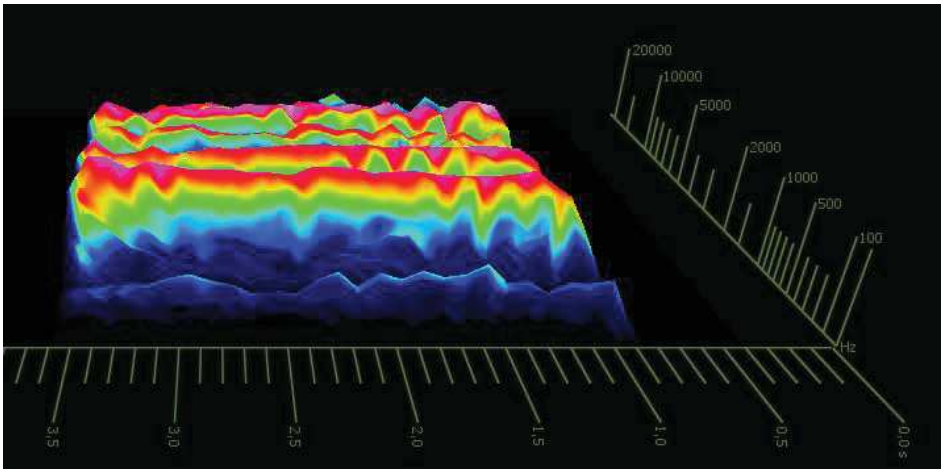


Fig. 63

Vista lateral del espectro de violín ejecutando nota A4.

Del análisis realizado se desprende la siguiente tabla:

Tabla 2.

Valores de armónicos de la muestra de violín.

Orden del Armónico	Frecuencia (Hz)	Amplitud (dB)	Ataque (ms)	Decaimiento (ms)	Sostenimiento (s)	Relevo (ms)
1	440	69,2	110	190	1,82	520
2	880	66,5	150	70	2,08	300
3	1320	52,1	130	310	2,02	90
4	1760	59,3	150	220	1,82	370
5	2200	66,2	170	140	1,85	370
6	2640	63	260	60	1,88	440
7	3080	51	470	170	1,88	380
8	3520	51,6	170	180	1,79	440
9	3960	45,3	260*	60*	1,88*	440*
10	4400	37,1	470*	170*	1,88*	380*
11	4840	36,6	470*	170*	1,88*	380*

12	5280	35,9	260*	60*	1,88*	440*
13	5720	30,8	----	----	----	----
14	6160	24	----	----	----	----
15	6600	26	----	----	----	----
16	7040	34	----	----	----	----

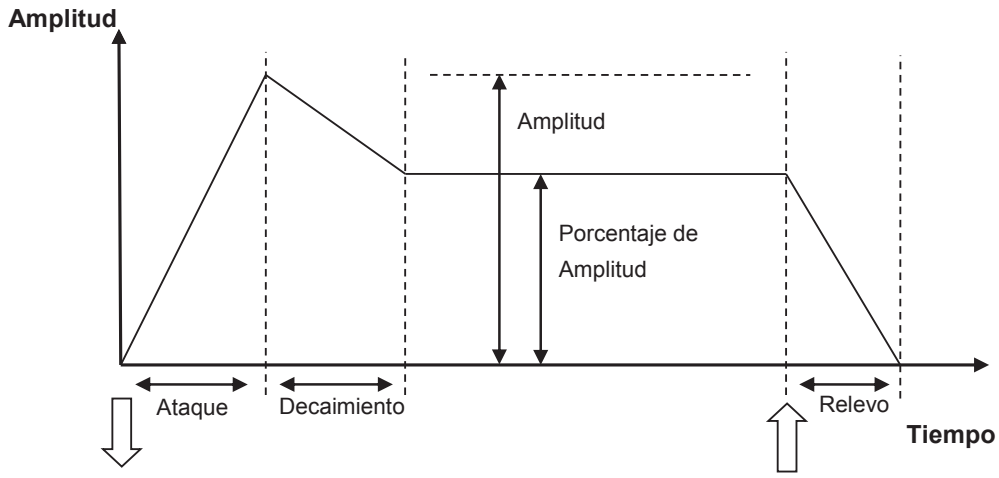
Observaciones: * Valores Aproximados, ---- Valores No Obtenidos

Se debe considerar que las muestras de violín tomadas para el análisis son producto de una interpretación musical de arco frotado, obteniendo la máxima duración posible con un solo movimiento de frotación del arco.

Tomando en cuenta que las muestras que se usaron para el análisis eran de un tipo de instrumento que se encuentra entre los más difíciles de sintetizar de forma realista (cuerda frotada), y que dichas muestras eran de muy larga duración, se consideró que utilizar valores cercanos a los más altos obtenidos en el análisis, como máximos en los parámetros de ataque, decaimiento, sostenimiento y relevo, para los generados de envolvente de tiempo sería una opción adecuada.

Por otro lado se debe considerar que el bloque *adsr~*, para el parámetro de sostenimiento no considera valores de tiempo sino que es un dato de porcentaje expresado en valores de 0 a 1.

En el siguiente gráfico se presenta una explicación más detallada del comportamiento del objeto *adsr~* de Max/MSP.



Donde: ↓ es el momento en que se presiona una tecla en un controlador MIDI.
↑ es el momento en que se deja de presionar dicha tecla del controlador MIDI.

Fig. 64

Comportamiento del ADSR.

Cuando se presiona una tecla, se envía la información de *velocity* al objeto *adsr~* y empieza a correr el tiempo de ataque. En este caso el tiempo de ataque es el tiempo que se demora la señal en alcanzar la amplitud correspondiente al máximo *velocity* con el que se haya presionado la tecla. Posterior a esto empieza el tiempo de decaimiento, y este es el tiempo que le toma a la señal descender hasta la amplitud correspondiente al sostenimiento. Como se puede ver en el gráfico, el sostenimiento no es un tiempo sino un porcentaje de la máxima amplitud alcanzada. Finalmente una vez que el usuario deje de ejecutar la nota, empieza a correr el tiempo de relevo, que es el tiempo que demorara el bloque *adsr~* hasta extinguir la señal.

Por ejemplo, si se tiene una calibración de parámetros del siguiente tipo: ataque = 10 ms, decaimiento = 5 ms, sostenimiento = 0,75 (75%), relevo = 10 ms; y suponiendo que el usuario ejecute una nota musical con un *velocity* que genere una señal de 100 dB, se tendrá el siguiente comportamiento:

Una vez que el usuario presione una tecla en el controlador MIDI empezará a correr el tiempo de ataque, es decir que en 10 ms la señal alcanzará una amplitud de 100 dB. Posterior a esto empezará el tiempo de decaimiento, y a la señal le tomara 5 ms llegar a un nivel de 75 dB. Este valor de 75 dB porque la máxima amplitud alcanzada por la señal llega a ser de 100 dB, y el porcentaje de sostenimiento es de 75%. La señal seguirá sonando con una amplitud de 75 dB hasta que el usuario deje de presionar la tecla en el controlador MIDI. En cuanto el usuario haga esto, empezará a correr el tiempo de relevo, y en este caso demorará 10 ms hasta que el objeto *adsr~* lleve la señal a una amplitud de $-\infty$.

Del análisis de datos realizado, así como de la comprensión del funcionamiento del objeto *adsr~*, se desprenden los siguientes rangos de valores para los parámetros del objeto *adsr~*:

- Ataque: 0 - 500 ms
- Decaimiento: 0 - 250 ms
- Sostenimiento: 0 - 100%
- Relevo: 0 - 500 ms

Una vez terminado de diseñar y programar el denominado bloque de generación de sonidos, el siguiente paso fue determinar cuál era el número necesario de estos bloques para el funcionamiento óptimo del sintetizador.

Este número de bloques viene dado en función de dos factores, por un lado el número de armónicos considerado como importantes y/o escuchados en los instrumentos acústicos; y por otro lado por criterios de diseño, pensando en la interface gráfica del sintetizador y en su facilidad de manejo.

Inicialmente se tomó como dato referencial la cantidad de armónicos promedio que puede escuchar el oído humano, o a su vez, que pueden ser generados por un instrumento musical. Si bien este dato varía entre fuentes bibliográficas, se puede considerar como promedio un número de dieciséis armónicos.

Si bien el número anterior es alto, otra vez a partir del análisis espectral realizado a diferentes muestras de instrumentos musicales, en este caso particular al del violín, surge la cuestión de, ¿cuantos armónicos realmente entregan información importante para la constitución del timbre propio de un instrumento?, y también, ¿cuántos armónicos realmente pueden ser escuchados por el oído humano?

De dicho análisis, el primer dato relevante que salta a la vista, es la diferencia en amplitud entre el primer armónico, y los armónicos de orden superior al noveno. En este caso particular, el primer armónico tiene un valor de casi 70 dB y el decimo armónico tiene un nivel de casi 40 dB, habiendo una diferencia entre estos de casi 30 dB.

Si bien la diferencia en decibeles para que se produzca enmascaramiento por nivel entre un sonido y otro, varía dependiendo de las característica de estos y de su distribución temporal, de esta diferencia en nivel (30 dB) encontrada en este caso particular surge la cuestión de, ¿los armónicos de índice superior a diez, son realmente escuchados por el oído humano y aportan a la creación del timbre en un instrumento musical?

Si bien la anterior es una cuestión difícil de resolver, y la respuesta variaría en función del instrumento musical que se analice, y de las condiciones auditivas de quien intente resolverla, se cree que para el presente proyecto, sintetizar sonidos con diez armónicos es suficiente.

Por otro lado surge también la cuestión de que en el caso de análisis de muestras del violín, los datos de ADSR son realmente difíciles de obtener en los armónicos a partir del noveno, esto debido a que sus formas de onda se dibujan pequeñas en el analizador espectral. Para tener un dato cercano en estos armónicos, se hizo una comparación con las envolventes temporales de armónicos de menor orden hasta encontrar el más parecido y asumir un comportamiento similar, método que sirvió hasta el decimosegundo armónico, pues en los de orden superior ni si quiera se pudo realizar esto.

Si a los resultados arrojados por el análisis realizado a las muestras de violín, se le agrega el hecho de que en el momento de desarrollar la interface gráfica para el sintetizador, incorporar más de diez bloques de generación de sonido daba como resultado un panel de control poco atractivo, y a que la calibración de más de diez de estos bloques (con siete parámetros cada uno) resultaría muy engorroso, se terminó decidiendo que tener una cantidad de diez bloques de generación de sonidos resultaba suficiente para emulación y creación de timbres, sin que el manejo del sintetizador sea demasiado complicado para los usuarios.

En la siguiente imagen se pueden ver a los diez bloques de generación de sonido para el sintetizador.

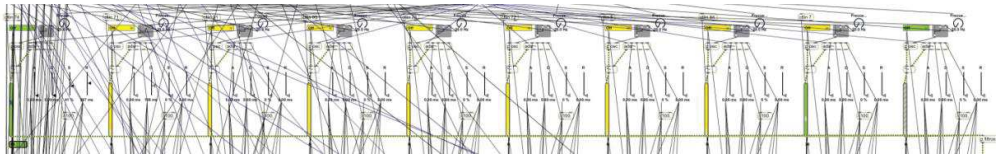


Fig. 65

Bloques de generación de sonido.

Posterior a la determinación de la cantidad de bloques de generación de sonido necesarios, surgirían cuestionamientos respecto a las capacidades de polifonía interpretativa que debería brindar el sintetizador.

3.2. Polifonía Interpretativa

Para entender la polifonía interpretativa de este sintetizador, es necesario comprender que cuando éste sintetiza una nota musical, está utilizando un juego de diez bloques de generación de sonido, por ende, si se desea que éste sintetice dos notas musicales al mismo tiempo, necesariamente se debe duplicar el número de bloques de generación de sonido. Es decir, para que el usuario pueda ejecutar dos notas musicales al mismo tiempo, el sintetizador debe tener veinte bloques de generación de sonido funcionando al mismo tiempo.

Si se considera además, que cada bloque de generación de sonido tiene funcionando al mismo tiempo a cuatro osciladores, y que cada oscilador consume una determinada capacidad de procesamiento del computador en el que se lo esté utilizando, resulta evidente que no se le puede otorgar capacidad de polifonía interpretativa infinita al sintetizador, y que es necesario determinar hasta dónde deben llegar estas capacidades.

La determinación de esta capacidad polifónica interpretativa del sintetizador surge de la armonía de la música occidental, en la cual un acorde formalmente está compuesto por tres notas musicales superpuestas, o más.

En resumen la mínima unidad armónica formalmente se compone de tres notas musicales, por lo cual el sintetizador debe contar con por lo menos una capacidad polifónica interpretativa de tres notas musicales. Considerando que hay acordes más complejos, se decide que el sintetizador finalmente tenga la capacidad de sintetizar cuatro notas musicales al mismo tiempo, para que así este sea más flexible sin tener un consumo de recursos demasiado elevado.

Considerando lo anterior, se deben cuadruplicar el número de bloques de generación de sonido, dando como resultado lo siguiente:

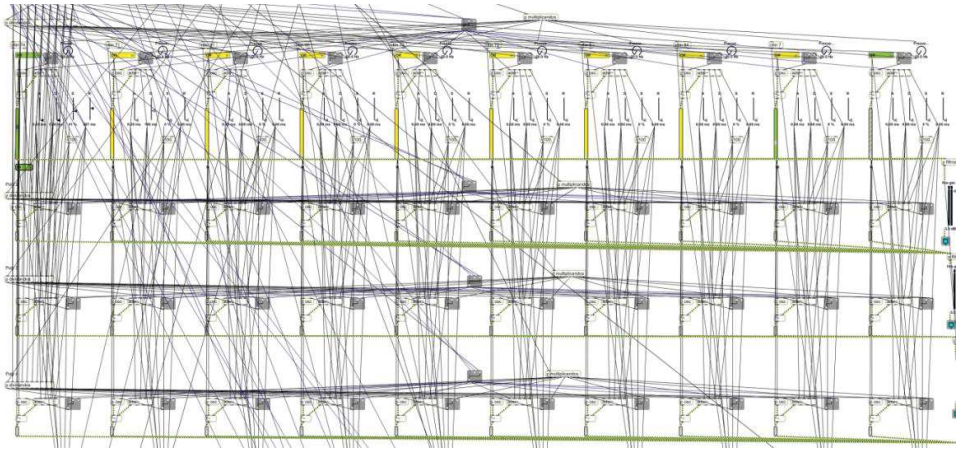


Fig. 66

Conjunto de bloques de generación de sonidos utilizados para una polifonía interpretativa de cuatro notas musicales.

Como se puede ver, las copias de los bloques de generación de sonido para la polifonía interpretativa son más sencillas que las originales, ya que no cuentan con objetos de calibración de parámetros.

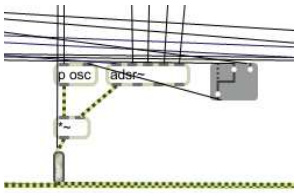


Fig. 67

Bloque de generación de sonidos utilizados para la polifonía interpretativa.

Estos bloques constan únicamente de objetos de control (explicados posteriormente), el *patch "p osc*", el objeto *adsr~*, la operación *~ y el objeto gain~; cuyos parámetros son los mismos que el usuario calibra para los bloques de generación de sonidos de la primera polifonía. Esto gracias a que los objetos que permiten enviar valores de calibración a los objetos de los primeros bloques, pueden enviar las mismas señales a los segundos, tercero, y cuartos bloques, fenómeno que se puede evidenciar en el siguiente gráfico.

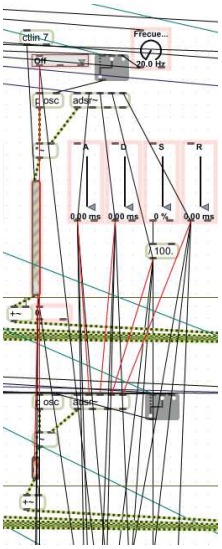


Fig. 68

Conexiones entre objetos de control de los bloques de generación de sonidos.

3.3. Creación de Zonas de Teclado

Una vez que se programó toda la sección de generación de sonidos con polifonía interpretativa, surge la idea de incorporar en el sintetizador la función de zonas de teclado, es decir, que el sintetizador tenga la capacidad de sintetizar un sonido con un determinado timbre para una determinado rango de notas musicales, y otro sonido con diferente timbre para otro rango de notas musicales.

La función anterior surge para brindar la posibilidad de emular sonidos de instrumentos cuyo timbre varía en función de la altura tonal que se ejecuta, por ejemplo el piano, en el cual sus notas de tesitura baja se caracterizan por tener un mayor sostenimiento y generar mayor contenido armónico que sus notas de tesitura alta.

Para añadir esta función al sintetizador, fue necesario duplicar toda la sección dedicada a la generación de sonidos con polifonía interpretativa, quedando el *patch* como se indica a continuación.

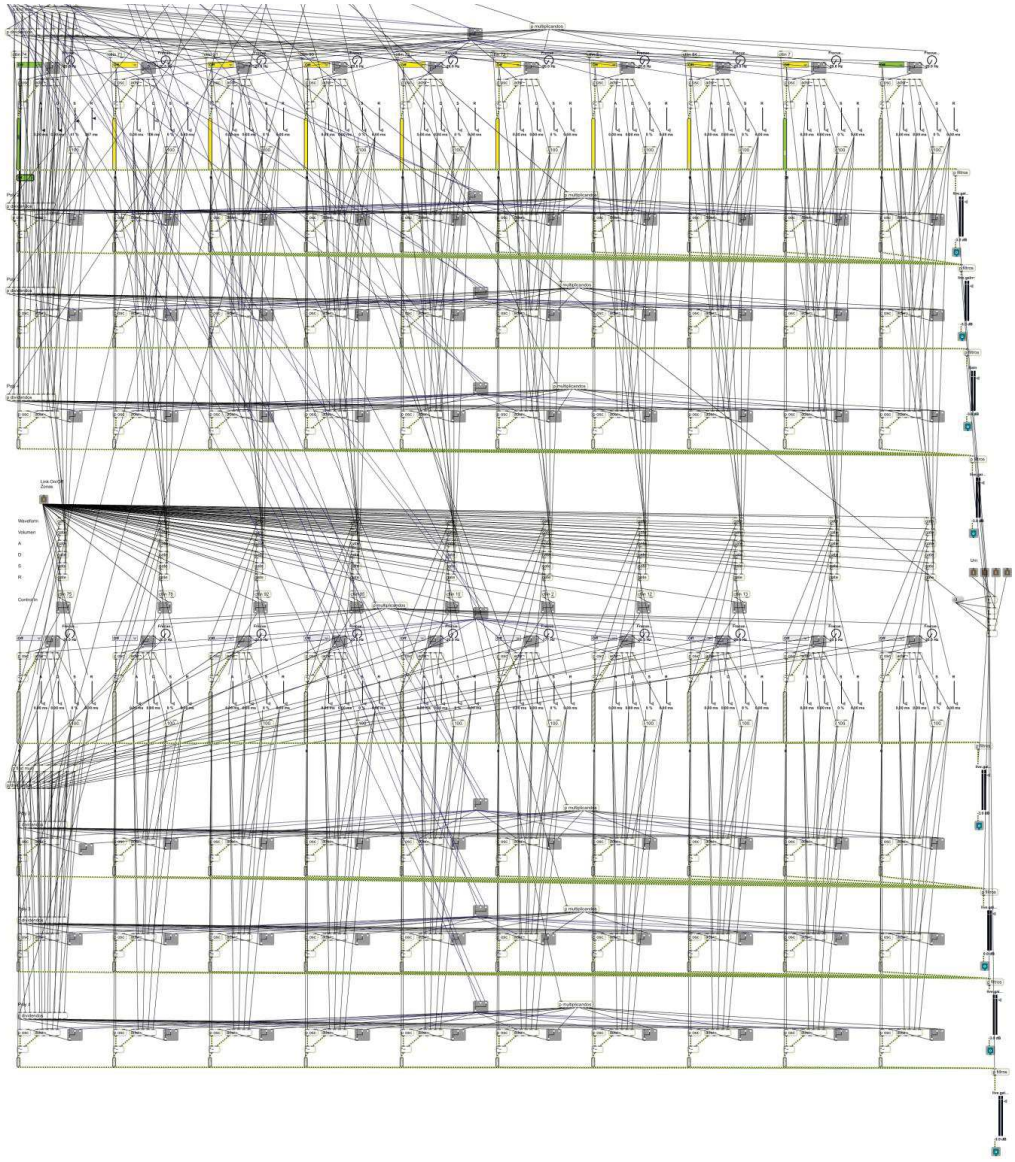


Fig. 69

Bloques de generación de sonido para las dos zonas del sintetizador.

Una vez realizada la duplicación se vio la necesidad de permitirle al usuario realizar un enlace de la calibración de parámetros entre las zonas uno y dos en las que se decidió dividir al sintetizador, esto en caso de que se desee utilizar un solo timbre y no tener que programar individualmente cada zona. Para esto se agregaron objetos de control manejados por un botón en la interface gráfica, permitiendo o no, que los parámetros de los bloques de generación de sonido

que se van calibrando en la zona uno, instantáneamente sean iguales para la calibración en los de la zona dos.

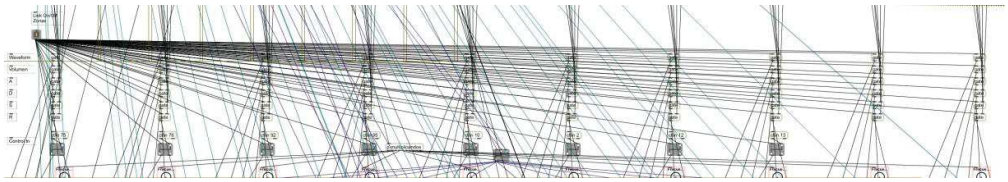


Fig. 70

Objetos de control agregados para la función de enlace de zonas.

Se debe considerar que para esta función de enlace, se tomó en cuenta a la zona uno como principal, es decir que, cuando la función está encendida, si se cambia un parámetro en los objetos correspondientes a la zona uno, este parámetro se modificará en igual medida en los objetos de la zona dos, pero que si aún con la función de enlace encendida, se cambia un parámetro en los objetos de la zona dos, esto no generará ningún cambio en ningún parámetro de la zona uno.

Además de estos mecanismos de control para la función de enlace, fue necesario incorporar objetos de control destinados a determinar a qué zona corresponde la nota que está ejecutando el usuario, y posterior a esta identificación, rutear la información de altura tonal y de *velocity* a los correspondientes bloques de generación de sonidos.

Para esto fueron necesarios dos sistema de ruteo de información, el primero, destinado al ruteo de información de altura tonal de la nota que se esté ejecutando, toma la información de número de nota MIDI que ingresa al sintetizador y lo envía al objeto >= donde se realiza una comparación con un número de nota MIDI que puede ingresar el usuario desde la interface gráfica (por ejemplo el número de nota MIDI 60 correspondiente a C3, también conocido como C central), y en caso de que el número de nota MIDI sea mayor al realizar la comparación, el objeto *gswitch2* lo rutea a la zona dos, y de resultar menor al realizar la comparación, lo rutea a la zona uno. De esta

manera el usuario puede ingresar un número de nota MIDI y así determinar en qué nota se divide al sintetizador en dos zonas.

Una vez que la información de número de nota MIDI es ruteada hacia la zona correspondiente, esta pasa a los objetos *mtof* correspondientes a cada zona, los cuales se encargan de convertir la información de numero de nota MIDI a su correspondiente frecuencia expresada en Hz. Por ejemplo, si el usuario ejecuta una nota A4, al bloque *mtof* le llegará como dato el número 69 (correspondiente a A4 en la norma MIDI), este la transformará en frecuencia, y así, a su salida entregará el dato de 440 Hz.

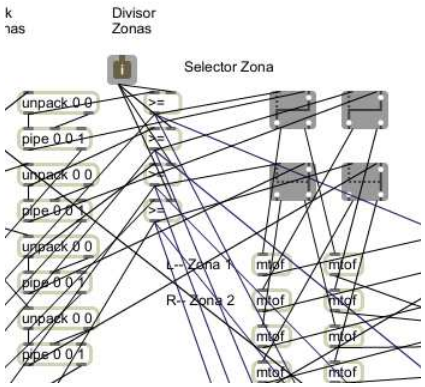


Fig. 71

Objetos encargados de rutear la información de altura tonal por zonas.

Para realizar el ruteo de datos de *velocity* correspondientes para cada zona se utilizan inicialmente los mismos bloques de comparación >= que se usan para rutear la información de número de nota MIDI. Por tanto, estos son los encargados de indicarles también a los objetos *gswitch2* correspondientes al ruteo de *velocity* hacia qué zona enviar esta información.

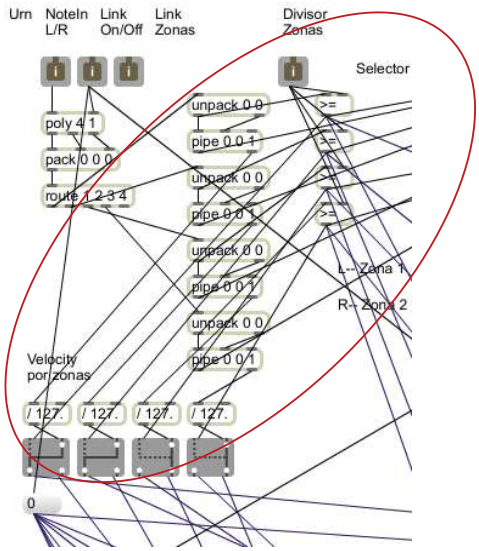


Fig. 72

Objetos encargados de rutear la información de *velocity* por zonas.

Los datos de *velocity* en la norma MIDI pueden oscilar en un rango de 128 pasos, siendo 0 su mínimo valor posible y 127 su máximo valor posible, pero se debe considerar que los datos de amplitud en Max/MSP, pueden oscilar en un rango de valores que va de 0 a 1, motivo por el cual hay que escalar los valores de *velocity* que ingresan al sintetizador. Es este el motivo por el cual la información de *velocity* pasa por los objetos / con argumento 127. Antes de llegar al objeto *gswitch2* para ser ruteados a la zona correspondiente.

Antes de que este dato escalado de *velocity* llegue al correspondiente bloque de *ads~*, pasa por otra sección de ruteo, esto con la finalidad de asegurar que la amplitud sea la correcta para cada zona. Es decir que, mientras el sonido se sintetice en los bloques de generación de sonido correspondientes a la zona en ejecución, que haya un sistema que evite de forma rotunda la generación de sonidos en los bloques correspondientes a la zona que no esta en ejecución.

Esto fue logrado mediante el uso de los objetos *gswitch*, siendo utilizados de forma inversa para la zona uno y la dos. Es decir, en los *gswitch* correspondientes a la zona dos, hay un mensaje constante de valor cero en su primera entrada de datos y el valor de *velocity* en su segunda entrada de datos;

mientras que en los *gswitch* correspondientes a la zona uno, el mismo mensaje de cero constante está conectado a la segunda entrada de datos, y el valor de *velocity* está conectado a su primera entrada de datos. Todos estos objetos *gswitch* son controlados por el bloque de comparación >= de la selección de zona. Por tanto, si el usuario está ejecutando notas en la zona uno, el bloque de comparación enviará un mensaje de cero a los bloques *gswitch* indicando que dejen pasar los valores que están ingresando a su primera entrada de datos, en el caso de los correspondientes a la zona uno, el valor de *velocity,* y en los correspondientes a la zona dos, el valor de cero constante. Si por el contrario el usuario ejecuta notas en la zona dos, el bloque de comparación >= enviará un valor de “uno”, indicando a los *gswitch* que permitan pasar los valores que están ingresando a su segunda entrada de datos, en los correspondientes a la zona uno, el mensaje de “cero” constante, y en los correspondientes a la zona dos, el valor de *velocity*.

Por medio de este método de ruteo, se evita que el sintetizador genere sonidos con los bloques correspondientes a la zona uno mientras se ejecutan notas en la zona dos, y viceversa.

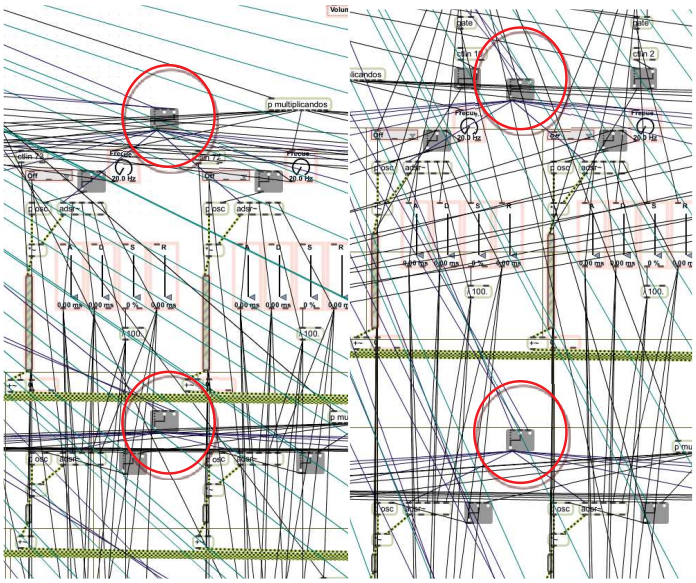


Fig. 73

Objetos de ruteo de valores de *velocity* y cero constante.

3.4. Enlace Armónico Encendido/Apagado

Otra parte muy importante en el diseño del sintetizador, son los mecanismos que permiten relacionar de dos diferentes formas a los bloques encargados de generar los sonidos a sintetizarse. Es decir la forma en que las frecuencias a generarse por los diferentes bloques de generación de sonidos se relacionan entre sí y con la frecuencia correspondiente a la nota ejecutada por el usuario.

Estas dos funciones son la de enlace y desenlace de armónicos. La primera función, la de enlace, es de utilidad cuando se quiere crear un timbre con contenido espectral armónico, es decir, cuando los sobre tonos de la frecuencia fundamental son múltiplos enteros de ésta. Por otro lado la función de desenlace sirve cuando se quiere sintetizar un timbre cuyo contenido espectral es no armónico, es decir, cuando los sobre tonos de la frecuencia fundamental no son necesariamente múltiplos enteros de ésta.

En ambos casos, la información de nota MIDI una vez convertida en dato de frecuencia por el objeto *mtof*, es enviada a un *gswitch2*, donde puede ser ruteada hacia los objetos encargados de realizar los cálculos de enlace o desenlace. Estos objetos *gswitch* rutean los datos en función de un dato de control que ingresa al *patch* por un *inlet*, el cual indica si el sintetizador tiene esta función encendida o apagada.

Cuando la función de enlace está encendida, el dato de frecuencia es enviado al *patch* “p multiplicandos”.

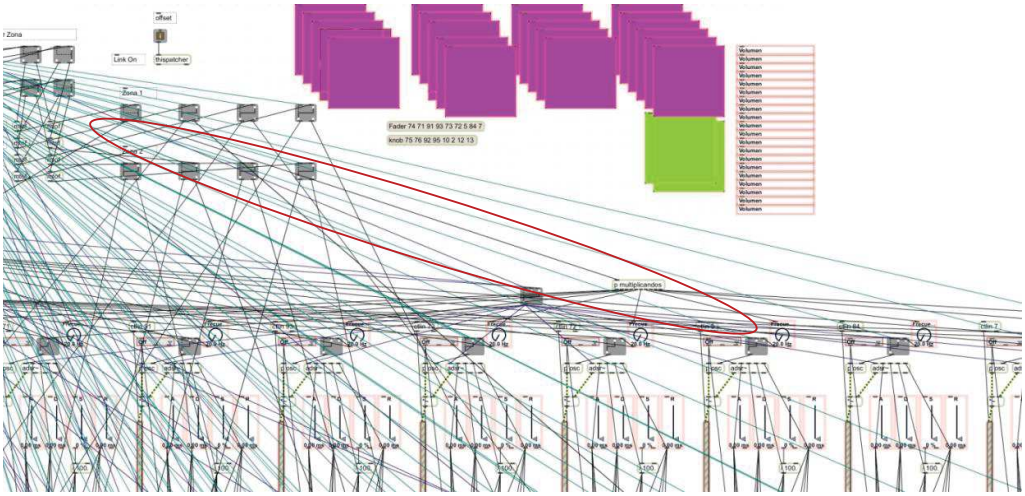


Fig. 74

Ruteo de información al *patch* "*p multiplicandos*".

En este *patch*, el valor de la frecuencia fundamental es multiplicado por números enteros del uno al diez, para así obtener los valores de frecuencia correspondientes a los diez primeros armónicos de la nota que se esté ejecutando.

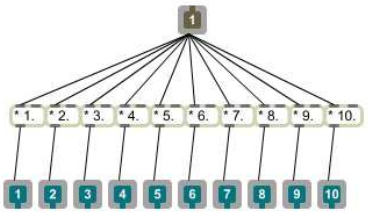


Fig. 75

Operaciones matemáticas dentro del *patch* "*p multiplicandos*"

Después estos valores de frecuencia son enviados a los objetos *live.dial* de cada bloque de generación de sonidos, para después ir a un objeto *gswitch*, el cual, en caso de estar la función de enlace encendida, envía estos datos de frecuencia hacia el *patch* "*p osc*" donde se generaran las señales con la frecuencia correspondiente.

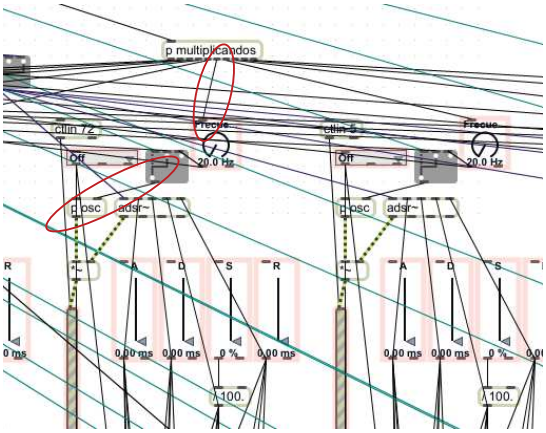


Fig. 76

Ruteo de datos con el enlace encendido.

Se debe tomar en cuenta que todo este proceso de ruteo y multiplicación de datos y señales, funciona igual para las cuatro polifonías interpretativas de cada zona. A pesar de esto, como ya se indicó antes, solo los bloques de generación de sonido correspondientes a la primera polifonía interpretativa (en ambas zonas) tienen interface gráfica por motivos de diseño. Por tanto, si bien el proceso funciona igual en todos los casos, solo cuando se toque notas cuya información vaya a la primera polifonía interpretativa, se verá como los objetos *live.dial* se mueven y muestran los valores de frecuencia correspondiente en la interface gráfica.

Gracias a esta función el usuario no tiene que calcular y calibrar los parámetros de las frecuencias correspondientes a los armónicos, y en caso de querer desactivar esta función lo puede hacer fácilmente con la modificación de un parámetro en la interface gráfica.

Cuando la función de enlace no está encendida, el usuario deberá modificar las frecuencias de los armónicos a generarse mediante los objetos *live.dial* (con nombre frecuencia) de la interface gráfica de cada bloque de generación de sonidos. Debe calibrar estos parámetros considerando que en la práctica el parámetro que estará modificando indirectamente será una relación entre los armónicos a generarse. Esto con la finalidad de que el sintetizador tenga

keyboard tracking con la función de enlace apagada, es decir que cambien la afinación de los sonidos sintetizados en función de la altura tonal de las notas que se ejecutan.

En esta función de enlace apagado, una vez que el usuario ha seleccionado las frecuencias en los objetos *live.dial*, estos datos de frecuencia en lugar de ser enviados a los *patch "p osc*", son enviados a los *patch* "*p find multiplicandos*".

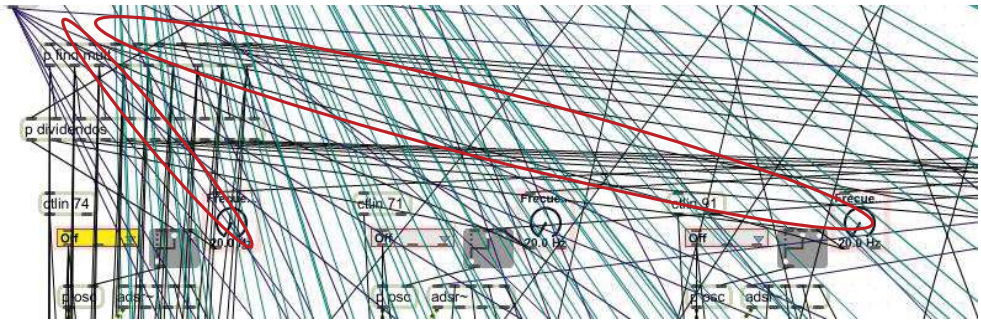


Fig. 77

Información de frecuencia conectada hacia el *patch* "*p find multiplicandos*".

En este *patch* se realiza una división entre las frecuencias correspondientes a los armónicos de orden dos al diez, para la frecuencia fundamental, es decir, se divide cada una de las frecuencias calibradas para los bloques de generación de sonido del dos al diez, para la frecuencia calibrada para el bloque número uno.

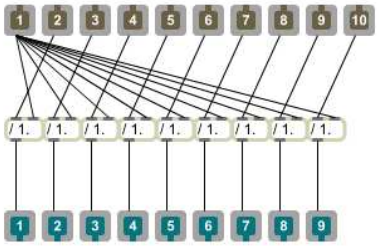


Fig. 78

Operaciones matemáticas dentro del *patcher* "*p find multiplicandos*".

Una vez realizada esta división, el dato correspondiente a la relación entre los armónicos superiores y la frecuencia fundamental, es enviada a todos los *patch* "*p dividendos*", uno para cada polifonía interpretativa.

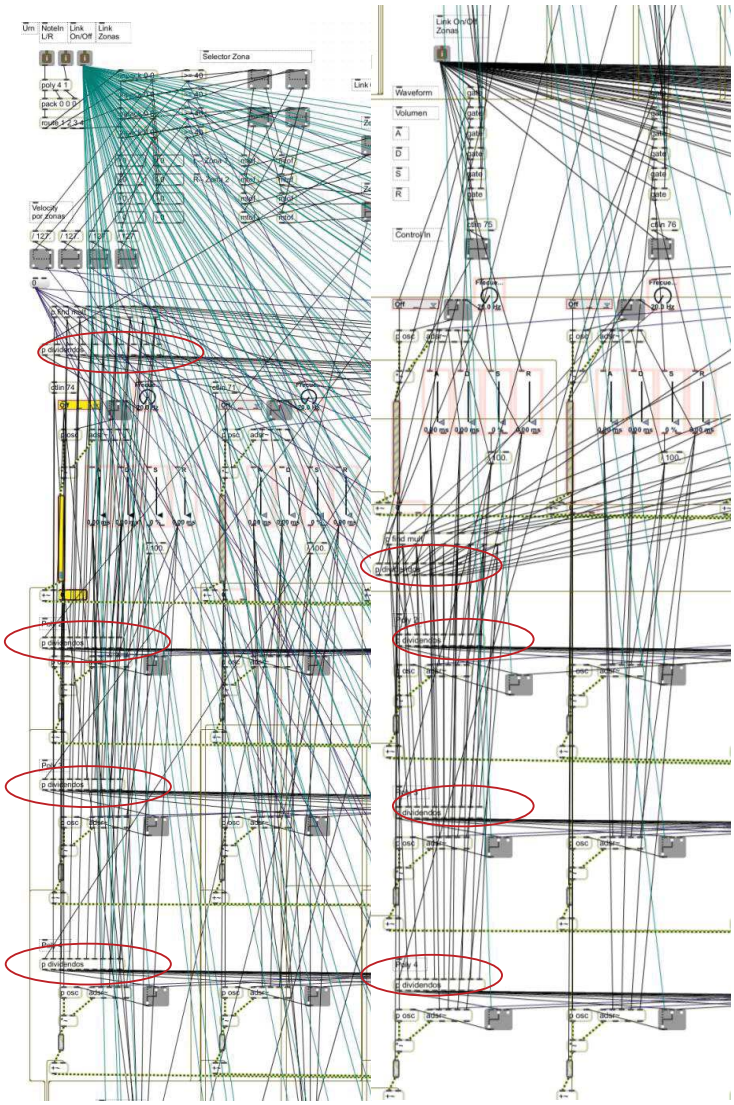


Fig. 79

Patchs "p dividiendos".

En estos *patchs* "*p dividendos*", en la primera entrada ingresa el dato de frecuencia correspondiente a la nota musical que se esté ejecutando, y en las otras nueve entradas restantes ingresan los resultados de las divisiones calculadas en los *patchs* "*p find multiplicandos*". Dentro del *patch* se multiplica

el dato de frecuencia por las relaciones obtenidas previamente. Luego los resultados de las multiplicaciones son enviados a cada uno de los diez bloques de generación de sonidos, que conforman cada una de las cuatro polifonías interpretativas, para cada una de las dos zonas.

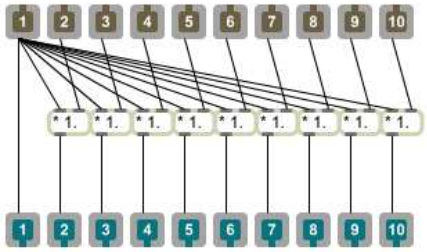


Fig. 80

Operaciones matemáticas realizadas dentro del *patch* "*p dividendos*".

Estos datos llegan a los objetos *gswitch*, los cuales permiten pasar estos datos calculados en los *patchs* "p dividendos" hacia los *patchs* "*p osc*" solo cuando recibe el dato de control de que la función de enlace está apagada.

Gracias a esta función el usuario del sintetizador puede determinar una cierta relación entre la frecuencia fundamental y el resto de frecuencias que conforman el timbre del sonido sintetizado, relación que se aplicará a cada nota que el usuario ejecute. En resumen, todas las notas ejecutadas tendrán la misma relación entre su frecuencia fundamental y sus sobre tonos y/o armónicos, pero los sonidos sintetizados irán cambiando su afinación para que sea la correspondiente a las notas musicales que se ejecutan.

3.5. Ingreso de Datos de Notas Ejecutadas

Para que el sintetizador tenga la capacidad de recibir información de cuatro notas siendo ejecutadas al mismo tiempo por el usuario, es decir, que tenga polifonía interpretativa, lo más importante es la forma en que el sintetizador obtiene la información de la ejecución.

En el sintetizador, en el *patch* dedicado a la interface gráfica, se encuentra el objeto *notein*, el cual tiene la función de obtener información de notas siendo

ejecutadas en una superficie de control conectada al computador. Las salidas de este objeto: *pitch* y *velocity*, van conectadas al objeto *poly*. En conjunto los objetos *notein* y *poly* son los que permiten que el sintetizador sea polifónico en términos de interpretación.

Si bien la información de polifonía ingresa al sintetizador por estos objetos, estos no tienen las capacidades de rutear la información de forma adecuada. Es por esto que los datos de polifonía son empacados por el objeto *pack* para luego ser ruteados por el objeto *route* hacia cuatro salidas diferentes. Cada uno de estos paquetes de datos ruteados tiene empacados los datos de *pitch* y *velocity* obtenidos por los objetos *notein* y *poly*. Luego de ser ruteados los datos son desempacados por el objeto *unpack* para poder enviar a los objetos correspondientes los datos de *pitch* y *velocity* por separado.

Posterior al desempaquetado de datos, fue necesario agregar el objeto *pipe* en el camino de los datos hacia los objetos encargados de rutear la información. Este objeto con la finalidad de retardar su llegada en un milisegundo. Fue necesario realizar esto dado que los mismos datos de notas ejecutadas son los que luego se comparan en el objeto >=, por tanto se debía dar ese milisegundo de ventaja al comparador para que envíe los datos de control a las compuertas antes de que los datos a ser ruteados lleguen a estas.

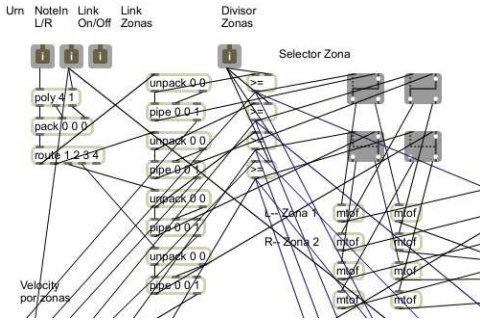


Fig. 81

Ruteo de datos de notas.

Al sintetizador también se le ha agregado en su interface gráfica un objeto de control, el cual permite ejecutar el sintetizador de forma monofónica en caso de

no tener un controlador MIDI para conectar al computador. Este objeto es el denominado *kslider*, que en este caso está configurado para permitir la ejecución desde la nota MIDI 36, es decir la nota musical C3, hasta la nota MIDI 84 la cual corresponde a C5.

3.6. Función Aleatoria de Nivel

Una vez que todas las señales de audio han sido generadas, se les agregó una sección de filtrado, dado el alto ruido inherente del programa al momento de generar señales. Esta sección de filtrado consta de dos filtros pasa altos con una frecuencia de corte de 30 Hz, y un Q de 0,7, y se encuentran dentro del *patch* "*p filtros*".

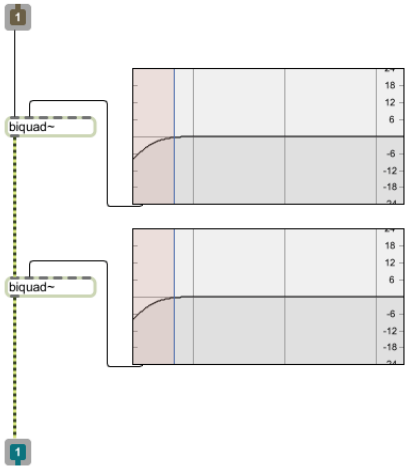


Fig. 82

Sección de filtrado dentro del *patch* "*p filtros*".

Ya que las señales que han sido filtradas deben ir del *patch* de generación de sonidos al *patch* de la interface gráfica. Hay que considerar que el *patch* de generación de sonidos estará incluido dentro del de la interface gráfica, por lo que pasa a convertirse en un *subpatch* de éste.

Para lograr este propósito se agregó un objeto *live.gain~* para cada polifonía interpretativa, es decir, cuatro por zona dando un total de ocho controles de nivel. Estos objetos posteriormente son conectados a un *outlet*.

Si bien estos objetos *live.gain~* cumplen su función de control de nivel, no lo hacen para brindar al usuario más opciones de control de las señales, sino para agregarle al sintetizador una función aleatoria de nivel.

Esta función se presenta como una respuesta a una cuestión de diseño surgida del análisis realizado previamente a las muestras de instrumentos musicales.

Una de las cosas más evidentes al analizar las envolventes temporales de amplitud que se producen en instrumentos acústicos, es la complejidad de estas.

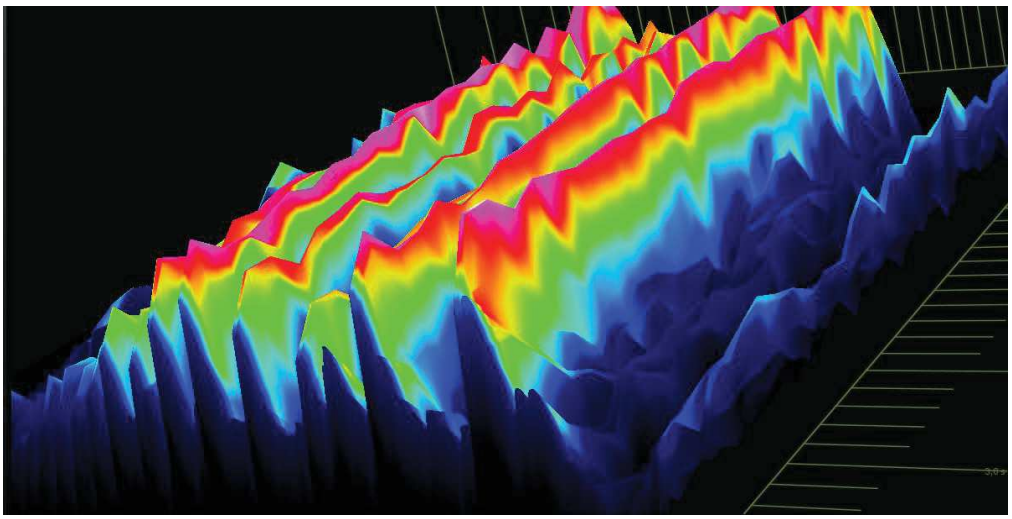


Fig. 83

Captura del análisis espectral realizado.

Como se puede ver en la imagen, la envolvente de tiempo de todos los componentes espectrales de la muestra de violín tienen un comportamiento imposible de emular con un generador de envolvente ya que presentan grandes cantidades de variaciones pequeñas, mientras que los generadores de envolvente tienen un comportamiento lineal en el tiempo, es decir, en los generadores de envolvente los incrementos o decrementos de amplitud son constantes, mientras que en los instrumentos reales, se presentan cambios.

La función aleatoria pretende darle al sintetizador la posibilidad de generar señales con envolventes cambiantes. Esto mediante pequeñas variaciones de nivel causadas por los objetos *live.gain~*, variaciones en función de números generados de forma aleatoria por el objeto *random*.

Los números aleatorios generados, pasan a sumarse con un valor de -3 dB, y luego llegan a los objetos *live.gain~* para indicarle las variaciones de nivel que debe generar. El dato de -3 dB es para asegurarse que las señales no causarán distorsión en los objetos *live.gain~* puesto que el número más alto que se puede generar aleatoriamente es de 5 dB, en dicho caso el máximo valor de *live.gain~* alcanzado sería 2 dB, dejando un *headroom* de 4 dB, esto considerando que el equivalente a 0 dB *Full Scale,* en el objeto *live.gain~* es representado con un valor de 6 dB.

Fig. 84

Objetos *live.gain~* usados para la función aleatoria de nivel.

Los controles de manejo de esta función aleatoria se encuentran en el *patch* destinado a la interface gráfica, así como el objeto destinado a crear los números aleatorios.

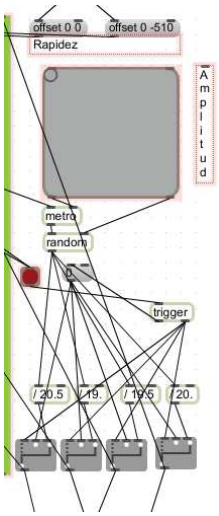


Fig. 85

Objetos usados para el ruteo de datos randómicos generados.

En este caso el objeto *pictslider* funciona como control de la función de generación de datos aleatorios, con la característica de que en su eje "x" controla la velocidad con la que los datos aleatorios son generados, y en su eje "y" controla el rango de valores entre los que pueden ser generados.

Para el control de la velocidad de generación de los datos, el objeto *pictslider* le envía valores entre 60 y 500 milisegundos al objeto *metro*. Este objeto *metro* por su parte, detecta cuándo hay en ejecución una nota musical y empieza a generar pulsos en intervalos de tiempo de acuerdo al valor entregado por el objeto *pictslider* (entre 60 y 500 milisegundos). Estos pulsos son enviados al objeto *random*, el encargado de generar datos aleatorios cada vez que le llega un pulso enviado por el objeto *metro*.

Para el control del tamaño de los valores generados aleatoriamente, el objeto *pictslider* envía al objeto *random* valores entre 25 y 95, indicándole que los

datos aleatorios que genere deben estar en el caso mínimo en un rango de entre 0 y 25, y en el caso máximo entre 0 y 95.

Posteriormente estos datos aleatorios generados son divididos para 19; 19,5; 20 y 20,5; dependiendo de la polifonía, y luego ruteados por un objeto *gswitch*, el cual permite pasar el dato hacia los objetos + cuando la función aleatoria está activa. Por el contrario, cuando la función está inactiva, envía un valor de cero constante a los objetos + en cuyo caso mantiene a los objetos de *live.gain~* en su valor de -3 dB de forma constante.

3.7. *Patch* de Interface Gráfica

Una vez programados los componentes de generación de sonidos, lo siguiente fue realizar un *patch* que los incorpore y al mismo tiempo permita desarrollar una interface gráfica agradable al usuario.

Para esto se creó un nuevo *patch* en el cual mediante un objeto *bpatcher* se le incorporó al *patch* de generación de sonidos (de ahora en adelante llamado *patch "osciladores")* para tener todas sus funciones y poder conectarlo con los objetos de control necesario.

Dentro de este nuevo *patch* además de desarrollar la interface gráfica, también se le agregaron algunas funciones al sintetizador, las cuales son explicadas a continuación.

Lo primero a realizarse fue permitir la salida de audio del *patch "osciladores*" hacia este nuevo *patch.* Para esto se conectaron las salidas del objeto *bpatcher* correspondientes al audio a dos objetos *live.gain~*, uno para control de nivel de cada una de las dos zonas de teclado creadas en el sintetizador.

Luego estos objetos fueron conectados a un nuevo objeto *live.gain~*, el cual pasa a convertirse en el control de nivel *master* de todo el sintetizador.

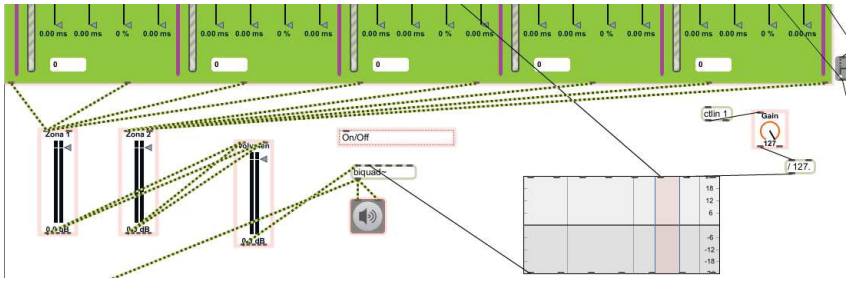


Fig. 86

Objeto *bpatcher* conectado a los objetos *live.gain~*.

Una vez conectadas las salidas de audio del *patch* osciladores al nuevo *patch,* se crearon los objetos encargados de enviar los mensajes de *pitch* y *velocity* hacia el *patch "*osciladores".

Para esto se creó el objeto *notein*, el cual va conectado al objeto *kslider* y a las entradas de *pitch* y *velocity* del *bpatcher*. A su vez las salidas del objeto *kslider* van conectadas al objeto *makenote* (con una calibración de duración de nota de 100 milisegundos, de tal manera que las notas ejecutadas desde el objeto *kslider* tengan dicha duración) y las salidas de este objeto a las entradas del objeto *bpatcher* correspondientes a *pitch* y *velocity*.

Ya realizado esto se observa también la necesidad de crear un sistema que sirva en caso de que hayan notas estancadas. Para esto se creó un mensaje de *stop*, el cual puede ser accionado mediante un objeto *button* o mediante la tecla Q del teclado de la computadora, enviando así un mensaje de *all notes off* al objeto *poly* que se encuentra en el *patch* "*osciladores"*.

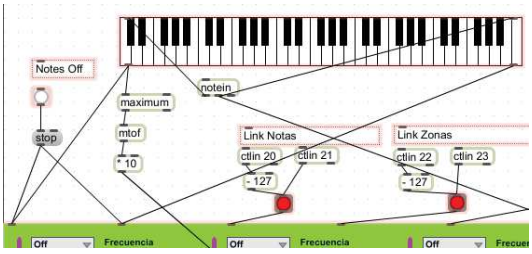


Fig. 87

Objetos *notein, kslider*, y mensaje *stop*.

Después se desarrolló un sistema de filtrado para complementar al sistema de zonas de teclado en su función de emulación de timbres de instrumentos cuyo timbre varíe en función de su altura tonal. Este filtro se encuentra conectado después del control master de nivel del sintetizador.

Este es un filtro de tipo pasa bajos, y tiene la particularidad de que su frecuencia de corte cambia automáticamente y está relacionada con la nota musical de mayor altura tonal que se esté ejecutando.

Para definir esta frecuencia de corte, se agregó un objeto *maximum* a la salida de *pitch* del objeto *kslider*, el cual se encarga de calcular cuál es la nota MIDI más alta que se está ejecutando en un determinado momento. Este objeto *maximum* va conectado a un objeto *mtof*, que convierte el dato de nota MIDI a frecuencia.

Una vez es calculado el dato de frecuencia de la nota más alta que se está ejecutando, este dato es multiplicado por diez y pasa a ser la frecuencia de corte del filtro pasa bajos, logrando así que la frecuencia de corte dependa de la altura tonal que se está ejecutando.

El control de ganancia del filtro es controlado por un objeto *live.dial*, el cual en su mínimo valor calibra al filtro para tener una ganancia de -40 dB y en su máximo valor una ganancia de 0 dB.

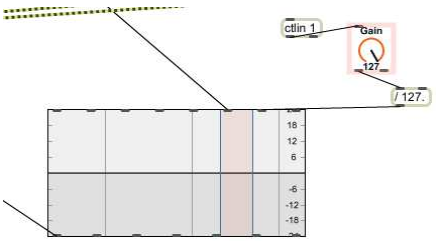


Fig. 88

Filtro pasa bajos.

Es la señal sintetizada y posteriormente filtrada la que va conectada a la salida de audio del sintetizador y hacia el analizador de espectro.

Para la salida de audio se uso un objeto *ezdac~*, el cual cumple una doble función. Por un lado sirve de salida del sintetizador encargándose de enviar la señal sintetizada hacia las salidas de audio del computador donde se utilice el sintetizador, y por otro lado actúa como un botón el cual permite activar o desactivar la generación de audio del sintetizador.

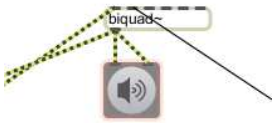


Fig. 89

Objeto *ezdac~*

Para el analizador de espectro se creó un nuevo *patch* llamado "*visor*", el cual está incorporado al *patch* de la interface gráfica mediante un objeto *bpatcher*.

Este *patch* "*visor"* tiene la función de mejorar la interface gráfica del analizador de espectro, puesto que el objeto *spectroscope~* como tal es capaz de realizar un análisis de espectro, pero su interface gráfica no permite interpretar los resultados con mucha facilidad y claridad.

En el *patch "visor"* se incorporó al objeto *spectroscope~*, y por encima de este se agregó al objeto *lcd*, para así poder dibujar líneas divisorias en los ejes de frecuencia y amplitud.

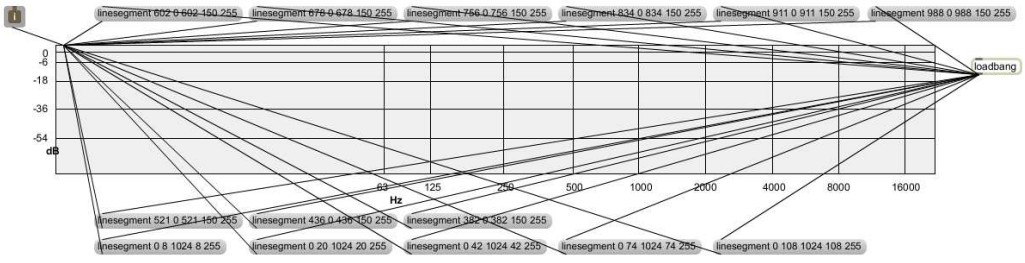


Fig. 90

Patch visor.

Gracias al objeto *lcd*, ahora se puede saber con más exactitud qué representan los gráficos entregados por el analizador de espectro, ya que el eje de frecuencia está dividido en nueve bandas de octava, y el eje de amplitud tiene cinco divisiones.

3.8. Sistema de Guardado y de Carga

Al sintetizador también se le agregó un sistema de guardado y carga de *presets* para así poder guardar los valores que tienen los objetos más importantes, y volver a ellos cuando sea necesario.

Así, una vez que se logre sintetizar un sonido agradable o emular fielmente el timbre de un instrumento musical, los valores de los parámetros de los objetos pueden ser guardados para así poder regresar a la misma calibración de parámetros y al mismo sonido sintetizado mediante un sencillo sistema de carga. Este sistema consta de dos funciones.

Por un lado se pueden guardar y cargar hasta un total de ocho programas mediante el objeto *preset* incorporado en el *patch* de la interface gráfica. Esta función está pensada para un guardado y cargado rápido de programas, para así pasar de uno a otro rápidamente.

Por otro lado, se pueden guardar las calibraciones de parámetros de cada uno de los ocho *presets*, en un archivo externo. Para esto se presiona un botón que corresponde al guardado, e inmediatamente se abre una ventana que permite poner un nombre y una ubicación determinada para el archivo que se va a guardar. Al hacer esto se crea un archivo con extensión *.json, el cual tiene los datos correspondientes a la calibración de los objetos en cada uno de los ocho programas.

Cuando se desea realizar la carga de los archivos guardados, se presiona el botón correspondiente a la carga de archivos, y esto abre inmediatamente un explorador que permite seleccionar el que se desee cargar. Una vez cargado, el objeto *preset* del sintetizador carga las ocho *programas* guardados en él previamente, para así poder cargarlos.

3.9. Controlador MIDI

Para implementar la opción control del sintetizador desde un controlador MIDI externo, se utilizó al controlador Axiom 49 II, de la empresa M-Audio.

Utilizando un controlador Axiom 49 II, configurado en su *preset* número uno, se pueden controlar los siguientes parámetros: con sus controles deslizantes, se puede controlar el volumen de los primeros nueve bloques de generación de sonido de la primera zona; con sus controles *encoders*, se pueden controlar los primeros ocho bloques de generación de sonido de la segunda zona; con el botón *loop* de la zona de transporte se puede desactivar la función aleatoria de nivel, y con el botón *play* se la puede activar; con el botón *record* de la zona de transporte se puede enviar un mensaje de *all notes off* al objeto *poly*; y con la rueda de modulación se puede controlar el parámetro de ganancia del filtro pasa bajos incorporado en el sintetizador.

3.10. Interface Gráfica

Una vez programadas todas las funciones de generación de sonido, así como todos los métodos de control de las mismas, finalmente se diseñó una interface gráfica agradable y fácil de usar, y que incorpore los objetos gráficos que permiten el uso del sintetizador.

Ya terminada la interface gráfica, lo único restante fue darle un nombre al sintetizador, y se opto por el siguiente: POLARIS 89, dando como resultado de toda la etapa de diseño y programación el siguiente sintetizador:

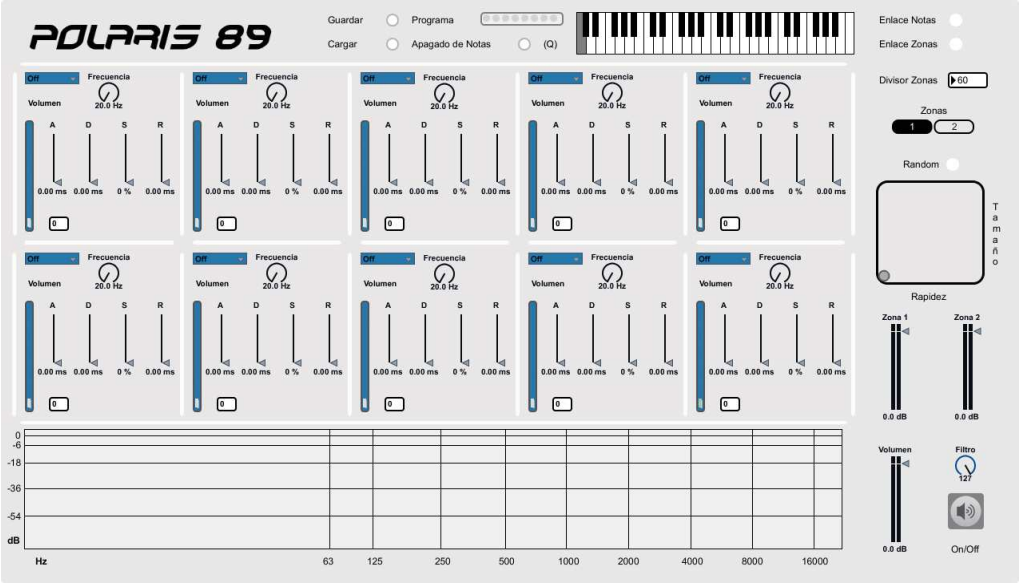


Fig. 91

Sintetizador POLARIS 89.

4. Presentación y Análisis de Resultados

Para tener una idea clara del desempeño del sintetizador, es necesario evaluarlo en los siguientes aspectos.

Por un lado el funcionamiento correcto de sus componentes individualmente y una vez ya integrados, es decir la verificación de que cada objeto esté cumpliendo su función correctamente; por otro lado el desempeño del sintetizador al momento de ser usado; y finalmente las características de los sonidos que pueden ser sintetizados por medio de éste, es decir, sus cualidades sonoras, su grado de similitud cuando se intenta emular instrumentos musicales, o la facilidad de creación de nuevos sonidos.

4.1. Funcionalidad del Sintetizador

Inicialmente se somete al sintetizador a pruebas de funcionamiento, verificando que su operación, y la interacción entre sus componentes sea correcta.

De las pruebas resulta evidente que todos los parámetros de todos los bloques de generación de sonido en ambas zonas de teclado funcionan correctamente, así mismo sus funciones de guardado y carga de archivos, y la carga de programas, la función de apagado de notas, el teclado incorporado en la interface gráfica, las funciones de enlace de notas y enlace de zonas, el divisor de zonas, la función aleatoria de nivel, el filtro incorporado, los controles de nivel, el botón de encendido o apagado, y el analizador de espectro.

En resumen, todas las funciones pensadas e incorporadas en el sintetizador funcionan correctamente sin presentar problemas.

4.2. Pruebas de Desempeño del Sintetizador.

Las pruebas de desempeño realizadas, fueron encaminadas a determinar el consumo de recursos computacionales del sintetizador al ser ejecutado utilizando todas sus funciones. Estas pruebas resultan necesarias de realizar dadas las características de generación de señales en el entorno de programación Max/MSP.

Por un lado es de sobra conocido que algunos programas realizados en Max/MSP pueden resultar muy demandantes en lo que respecta a recursos computacionales; y por otra lado se sabe también que Max/MSP suele tener problemas para generar señales con frecuencias que se encuentren en la parte más alta del espectro (comprobado más adelante con las pruebas realizadas), que es el caso del sintetizador desarrollado.

Para realizar estas pruebas se utilizaron dos computadoras, una con sistema operativo Windows 7, y otra con sistema operativo OS X 10.6; ambas con capacidades en *hardware* por debajo del estándar de capacidad disponibles actualmente en el mercado, esto con la finalidad de someter al sintetizador a pruebas en donde decididamente consuma gran porcentaje de la capacidad de procesamiento de las computadoras en las que se lo está ejecutando.

Igualmente las pruebas fueron realizadas con una calibración de DSP en Max/MSP promedio, de tal manera que no exista latencia.

A continuación se detallan los resultados encontrados en las pruebas realizadas con la primera computadora:

- **Toshiba Satellite A305-S6837**

Sistema Operativo:	Windows 7 Ultimate de 64 bits
Procesador:	Intel CoreDuo @ 1.83 GHz
Memoria RAM:	4 GB DDR2

- **Max/MSP 5.19**

I/O Vector Size:	1024
Signal Vector Size:	1024
Sampling Rate:	44.1 kHz
Scheduler:	Apagado

Vector Optimization: Apagado

Porcentaje de DSP consumido por la aplicación corrida desde Max/MSP con la función de Enlace de Notas encendida, en diferentes circunstancias:

- *Stand By* (sin ejecución de notas musicales): 27%
- Al ejecutar la máxima nota MIDI posible: 90%
- Mínimo consumo ejecutando notas musicales: 40%
- Promedio de consumo: 65%

A continuación se detallan datos adicionales con respecto al rango de notas MIDI capaz de ser ejecutadas.

- Máxima nota MIDI posible de ejecutar: Nota MIDI 90
- Mínima nota MIDI posible de ejecutar: Nota MIDI 0
- Nota MIDI con menor consumo de DSP: Nota MIDI 20

Porcentaje de DSP consumido por la aplicación corrida desde Max/MSP con la función de Enlace de Notas apagada, en diferentes circunstancias.

En el caso particular en que se use el sintetizador con la función de enlace de notas apagada, se ve un comportamiento que depende específicamente de la calibración de parámetros que se tenga, es así que con ciertas calibraciones se puede ejecutar hasta la nota MIDI mas alta reconocida por Max/MSP (nota MIDI 120) con un consumo de DSP de 60%; y en otras ocasiones al ejecutar notas musicales mucho más bajas (nota MIDI 70 o inferiores) se da una sobrecarga en el DSP.

A continuación se detallan los resultados encontrados en las pruebas realizadas con la segunda computadora:

- **MacBook Pro 7.1**

Sistema Operativo: OS X 10.6

Procesador: Intel CoreDuo @ 2.4 GHz

Memoria RAM: 4 GB DDR3

- **Max/MSP 5.19**

I/O Vector Size: 512

Signal Vector Size: 512

Sampling Rate: 44.1 kHz

Scheduler: Apagado

Vector Optimization: Apagado

Porcentaje de DSP consumido por la aplicación corrida desde Max/MSP con la función de Enlace de Notas encendida, en diferentes circunstancias:

- *Stand By* (sin ejecución de notas musicales): 20%
- Al ejecutar la máxima nota MIDI posible: 93%
- Mínimo consumo ejecutando notas musicales: 30%
- Promedio de consumo: 50%

A continuación se detallan datos adicionales con respecto al rango de notas MIDI capaz de ser ejecutadas.

- Máxima nota MIDI posible de ejecutar: Nota MIDI 108
- Mínima nota MIDI posible de ejecutar: Nota MIDI 0
- Nota MIDI con menor consumo de DSP: Nota MIDI 67

Porcentaje de DSP consumido por la aplicación corrida desde Max/MSP con la función de Enlace de Notas apagada, en diferentes circunstancias.

Igual que en las pruebas realizadas con la primera computadora, este consumo de DSP va ligado a la calibración de parámetros usados, específicamente a la relación entre las frecuencias usadas en los bloques de generación de sonido. Cuando los bloques de generación de sonido superiores tienen calibradas frecuencias muy altas en relación a la fundamental, al momento de ejecutar

notas musicales superiores, las señales a generarse tienen frecuencias muy por encima de los 20 kHz, lo que presenta un conflicto al programa.

4.3. Pruebas de Síntesis de Sonido

Una vez realizadas las pruebas de funcionamiento y desempeño del sintetizador, se realizaron las pruebas enfocadas a las capacidades de síntesis como tal de este, enfocándose en la facilidad de obtener sonidos realistas o de creación de nuevos sonidos, y si las diferentes funciones incorporadas son útiles o no, si generan cambios notorios o no, si aportan sonoramente o van en desmedro de los sonidos sintetizados, etc.

En su mayoría estas pruebas fueron realizadas cuantitativamente y cualitativamente, es decir, enfocadas desde aspectos técnicos, y de aspectos estéticos. Las primeras pruebas arrojan datos exactos y comparables técnicamente, mientras que las otras son meramente estéticas y son el reflejo de lo percibido por parte de los sujetos de prueba.

De manera general las pruebas técnicas consistieron en comparación de espectro sonoros entre diferentes muestras de audio, mientras que las pruebas estéticas en comparación auditiva de las mimas muestras

Para las pruebas técnicas se utilizó también a la función de *Meter Bridge* del *plug-in* Ozone 5 Advanced, de la compañía Izotope, mientras que las pruebas estéticas fueron realizadas por los desarrolladores del proyecto.

4.3.1. Pruebas de Emulación de Sonido (Prueba 1)

Esta prueba tuvo como finalidad ver las capacidades del sintetizador para emular sonidos de instrumentos acústicos, y consistió de dos partes. Por un lado se intentó sintetizar los sonidos de forma analítica y por otro lado de forma experimental.

Para la prueba analítica, se utilizaron los datos obtenidos del análisis de la muestra de violín, y se calibró al sintetizador con éstos. Para la prueba

experimental, se sintetizó un sonido que se asemeje al de un violín mediante la manipulación de los parámetros del sintetizador desde cero.

En ambos casos se compararon los espectros de frecuencia de los sonidos sintetizados con los de la muestra original de violín, y posteriormente se realizó la misma comparación auditivamente.

Para tener más resultados de las pruebas de emulación de sonido, se decidió hacer la misma prueba descrita previamente, pero sintetizando un sonido de bombo, realizando previamente un análisis espectral de una muestra de este instrumento musical.

A continuación se muestran los espectros obtenidos de la muestra sonora del violín de referencia (Violín de referencia.wav).

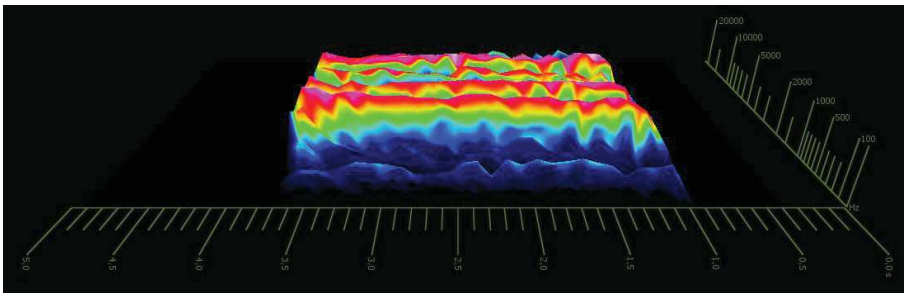


Fig. 92

Vista frontal del espectro de frecuencias de la muestra de violín de referencia.

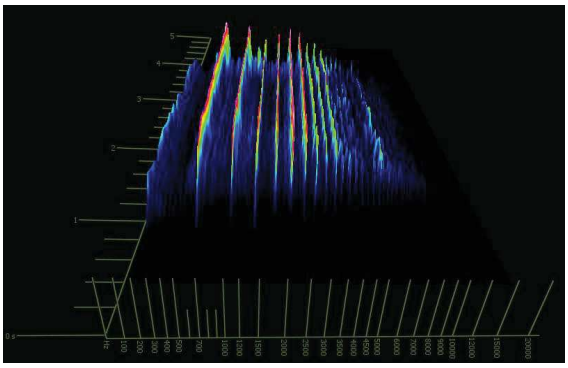


Fig. 93

Vista lateral del espectro de frecuencias de la muestra de violín de referencia.

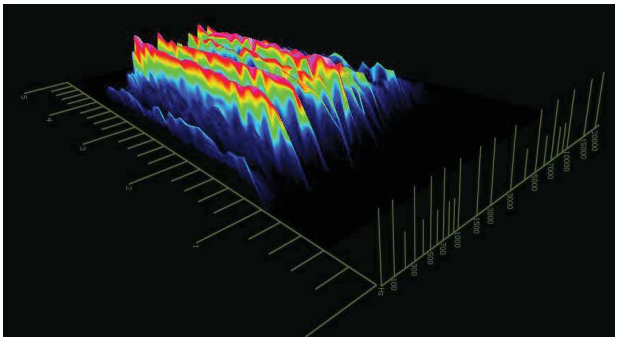


Fig. 94

Vista diagonal del espectro de frecuencias de la muestra de violín de referencia.

A continuación se muestran los datos obtenidos del análisis de la muestra del violín de referencia (obtenidos previamente para la sección de diseño y programación).

Tabla 3.

Valores de armónicos de la muestra de violín.

Orden del Armónico	Frecuencia (Hz)	Amplitud (dB)	Ataque (ms)	Decaimiento (ms)	Sostenimiento (s)	Relevo (ms)
1	440	69,2	110	190	1,82	520
2	880	66,5	150	70	2,08	300
3	1320	52,1	130	310	2,02	90
4	1760	59,3	150	220	1,82	370
5	2200	66,2	170	140	1,85	370
6	2640	63	260	60	1,88	440
7	3080	51	470	170	1,88	380
8	3520	51,6	170	180	1,79	440
9	3960	45,3	260*	60*	1,88*	440*
10	4400	37,1	470*	170*	1,88*	380*

11	4840	36,6	470*	170*	1,88*	380*
12	5280	35,9	260*	60*	1,88*	440*
13	5720	30,8	----	----	----	----
14	6160	24	----	----	----	----
15	6600	26	----	----	----	----
16	7040	34	----	----	----	----

Observaciones: * Valores Aproximados, ---- Valores No Obtenidos

A continuación se muestran los espectros obtenidos de la muestra sonora del violín sintetizada analíticamente (Violín síntesis analítica.wav).

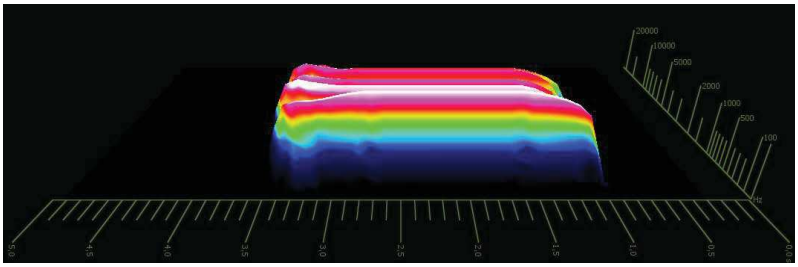


Fig. 95

Vista frontal del espectro de frecuencias de la muestra de violín sintetizada analíticamente.

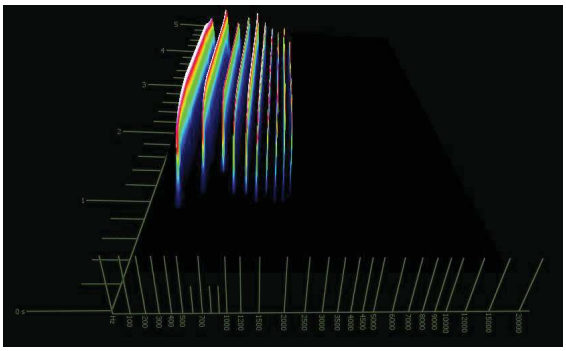


Fig. 96

Vista lateral del espectro de frecuencias de la muestra de violín sintetizada analíticamente.

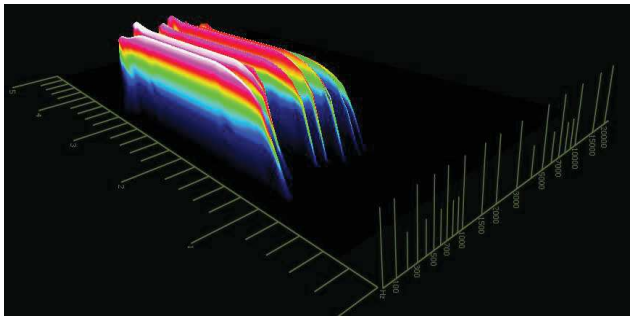


Fig. 97

Vista diagonal del espectro de frecuencias de la muestra de violín sintetizada analíticamente.

A continuación se muestran los espectros obtenidos de la muestra sonora del violín sintetizada experimentalmente (Violín síntesis experimental.wav)

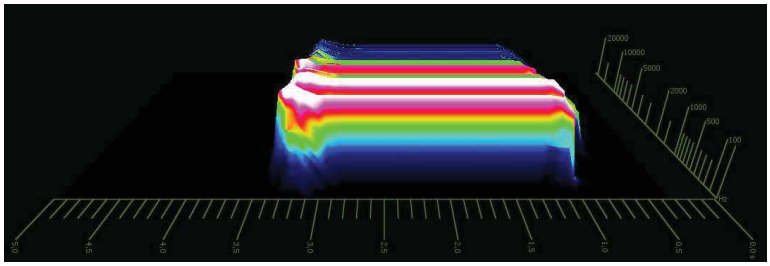


Fig. 98

Vista frontal del espectro de frecuencias de la muestra de violín sintetizada experimentalmente.

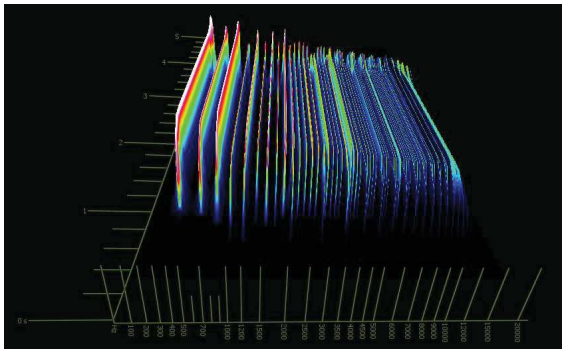


Fig. 99

Vista lateral del espectro de frecuencias de la muestra de violín sintetizada experimentalmente.

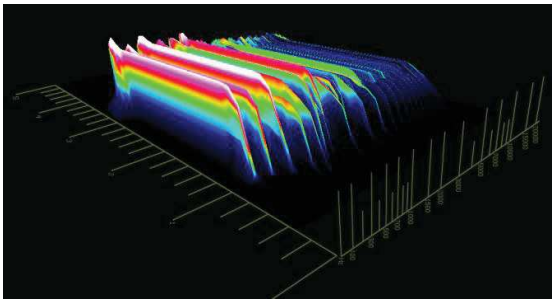


Fig. 100

Vista diagonal del espectro de frecuencias de la muestra de violín sintetizada experimentalmente.

A continuación se muestran los espectros obtenidos de la muestra sonora del bombo de referencia (Bombo de referencia.wav).

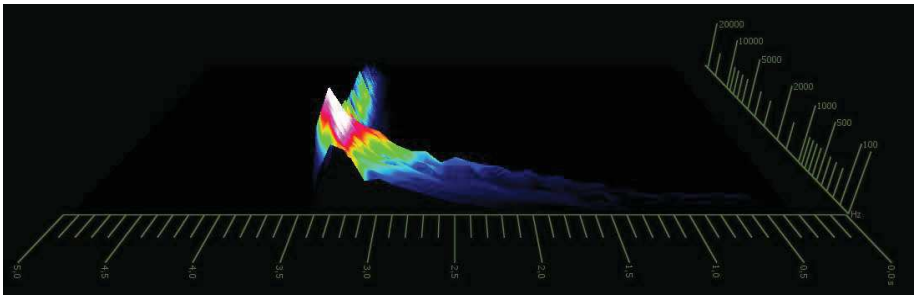


Fig. 101

Vista frontal del espectro de frecuencias de la muestra de bombo de referencia.

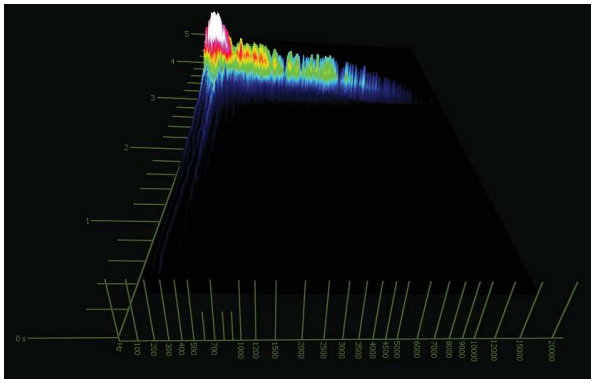


Fig. 102

Vista lateral del espectro de frecuencias de la muestra de bombo de referencia.

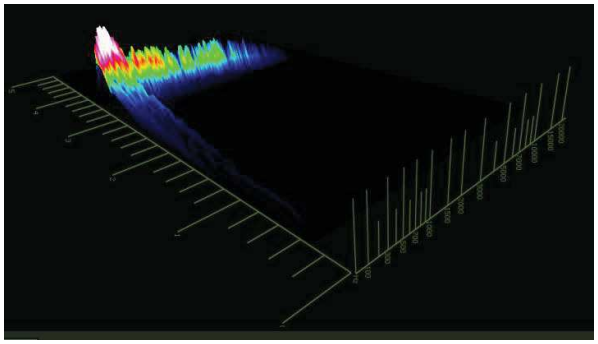


Fig. 103

Vista diagonal del espectro de frecuencias de la muestra de bombo de referencia.

A continuación se muestran los datos obtenidos del análisis de la muestra de bombo de referencia.

Tabla 4.

Valores de sobretonos de la muestra de bombo de referencia.

Orden del Sobretono	Frecuencia (Hz)	Amplitud (dB)	Ataque (ms)	Decaimiento (ms)	Sostenimiento (s)	Relevo (ms)
1	56,1	87	240	150	----	500
2	124	90	150	100	----	140
3	274	83	30	50	----	125*
4	384	79	40	45*	----	133*
5	470	76	50	48*	----	125*
6	523	72	40*	54*	----	100*
7	572	72	45*	36*	----	110*
8	606	72	60*	35*	----	98*
9	683	70	50*	40*	----	90*
10	850	71	30*	37*	----	90*

Observaciones: * Valores Aproximados, ---- Valores No Obtenidos

A continuación se muestran los espectros obtenidos de la muestra sonora del bombo sintetizado analíticamente (Bombo síntesis analítica.wav).

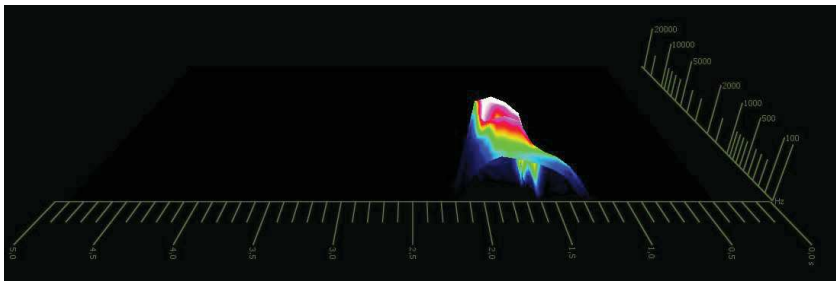


Fig. 104

Vista frontal del espectro de frecuencias de la muestra de bombo sintetizado analíticamente.

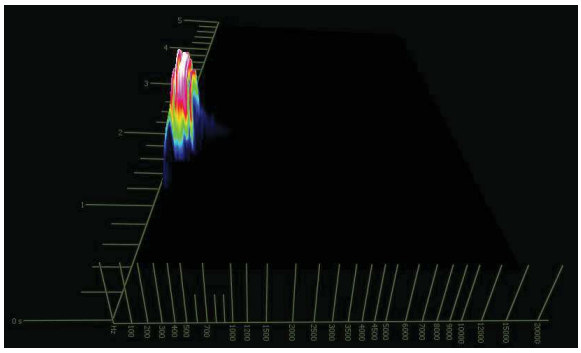


Fig. 105

Vista lateral del espectro de frecuencias de la muestra de bombo sintetizado analíticamente.

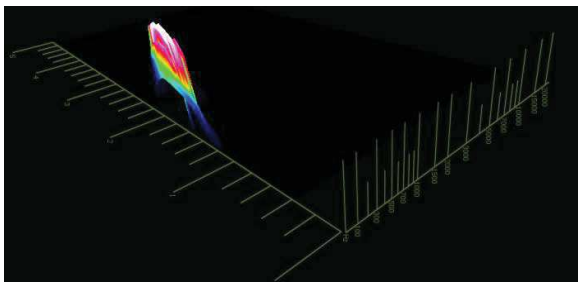


Fig. 106

Vista diagonal del espectro de frecuencias de la muestra de bombo sintetizado analíticamente.

A continuación se muestran los espectros obtenidos de la muestra sonora del bombo sintetizado experimentalmente (Bombo síntesis experimental.wav)

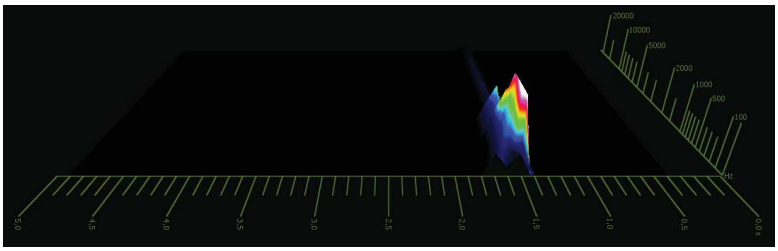


Fig. 107

Vista frontal del espectro de frecuencias de la muestra de bombo sintetizado experimentalmente.

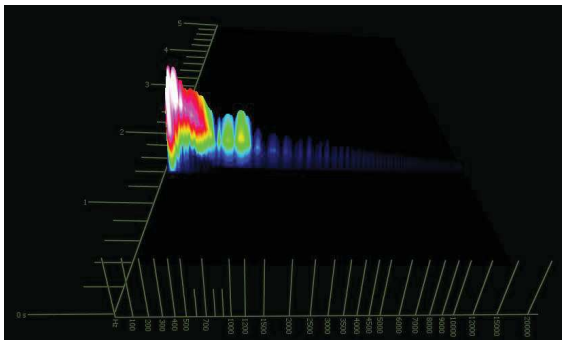


Fig. 108

Vista lateral del espectro de frecuencias de la muestra de bombo sintetizado experimentalmente.

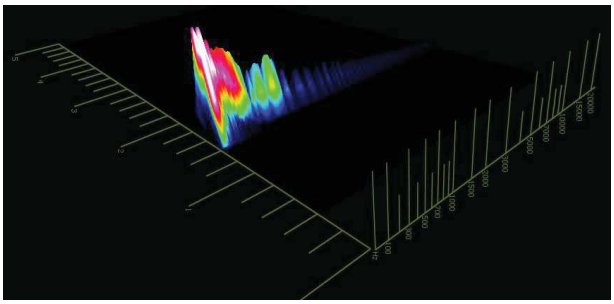


Fig. 109

Vista diagonal del espectro de frecuencias de la muestra de bombo sintetizado experimentalmente.

4.3.1.1. Análisis de Resultados Técnicos

Al comparar los espectros obtenidos del violín, queda en evidencia una vez más que las envolventes temporales de amplitud de un instrumento acústico presentan grandes variaciones y son muy complejas, mientras que las creadas mediante ADSR son mucho más simples y no pueden emular el comportamiento aleatorio y complejo de la envolvente temporal de un instrumento acústico.

Además en el espectro de la muestra de referencia se ve que sus componentes espectrales tienen un determinado ancho de banda, es decir que no son tonos totalmente puros como los generados por el sintetizador. También que sus espectros están compuestos por una gran cantidad de inarmónicos, los cuales no son considerados en la elaboración del timbre de los sonidos sintetizados.

Por otra parte, al comparar los sonidos sintetizados analíticamente con los sintetizados experimentalmente y las muestras de referencia, se ve que los primeros tiene un rango de frecuencia mucho menor. Esto dado que los primeros se componen únicamente de diez señales senoidales sumadas como se puede ver en las figuras 96 y 105; mientras que los sonidos sintetizados experimentalmente, al usar formas de onda más complejas (diente de sierra, triangular o cuadrada), tienen mayor cantidad de armónicos y/o sobretonos, tal como las muestras de referencia.

4.3.1.2. Análisis de Resultados Auditivos

En el caso particular del violín, aun cuando técnicamente las muestras de los sonidos sintetizados distan mucho de la muestra del sonido de referencia, auditivamente se encuentran similitudes en el timbre de las señales. Por ejemplo el sonido sintetizado experimentalmente emula aceptablemente el sonido característico de la cuerda frotada del violín, y por el otro lado el sonido sintetizado analíticamente emula mejor la relación entre armónicos y fundamental del sonido de referencia, aunque ninguno de los dos llegue a emular satisfactoriamente al instrumento completo.

Por otro lado en el caso de las muestras de bombo los sonidos sintetizados analíticamente y experimentalmente son muy diferentes. El sonido sintetizado analíticamente carece totalmente de definición y tiene una característica sónica muy típica en efectos de sub graves generados electrónicamente para bombos. El sonido sintetizado experimentalmente, por el contrario tiene mayor definición y se acerca más al sonido de un bombo acústico; además, a pesar de ser diferente a la muestra de referencia, tiene un sonido más agradable que ésta.

4.3.2. Pruebas de la Función Aleatoria de Nivel (Prueba 2)

Estas pruebas tuvieron como finalidad medir el impacto sonoro de la función aleatoria de nivel, y consistieron en la comparación auditiva y de espectros de muestras sintetizadas de un bombo y un violín, en un caso con la función aleatoria de nivel apagada, y en otro caso con la función encendida.

En el caso de estas pruebas, se usó como muestra de violín y bombo sin función aleatoria de nivel, a los sonidos de estos sintetizados experimentalmente (Violín sin función aleatoria de nivel.wav y Bombo sin función aleatoria de nivel.wav).

A continuación se muestran los espectros obtenidos de la muestra sonora del violín con función aleatoria de nivel (Violín con función aleatoria de nivel.wav).

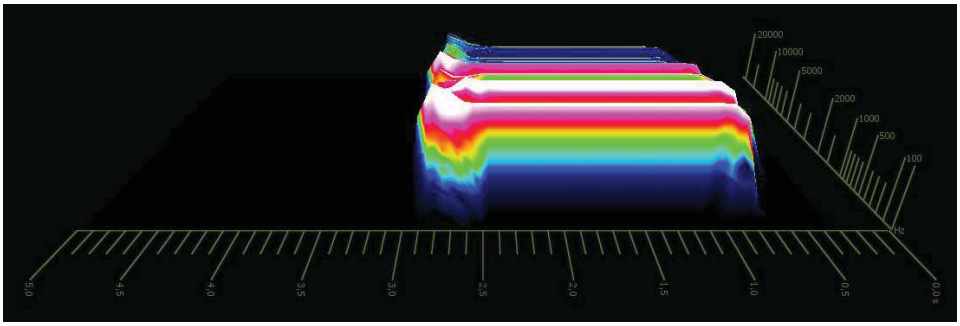


Fig. 110

Vista frontal del espectro de frecuencias de la muestra de violín con función aleatoria de nivel.

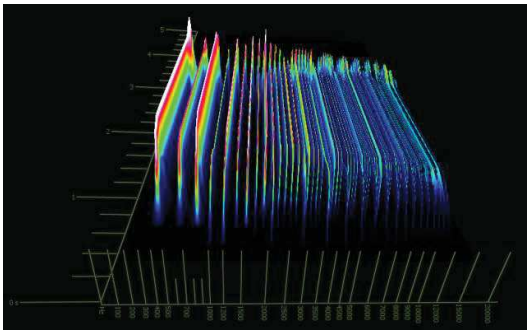


Fig. 111

Vista lateral del espectro de frecuencias de la muestra de violín con función aleatoria de nivel.

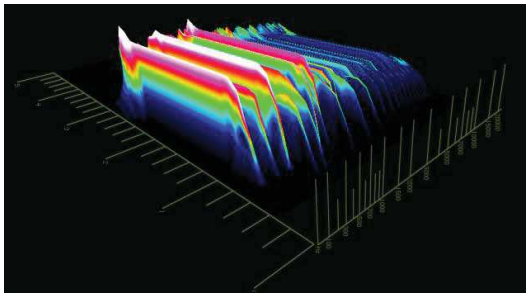


Fig. 112

Vista diagonal del espectro de frecuencias de la muestra de violín con función aleatoria de nivel.

A continuación se muestran los espectros obtenidos de la muestra sonora del bombo con función aleatoria de nivel (Bombo con función aleatoria de nivel.wav).

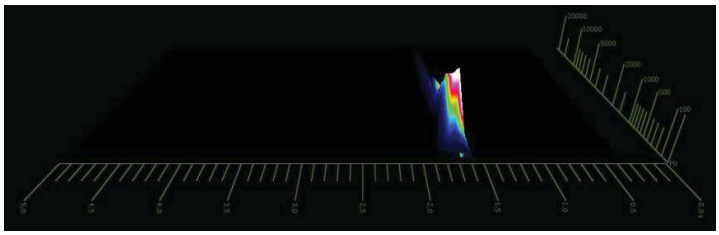


Fig. 113

Vista frontal del espectro de frecuencias de la muestra de bombo con función aleatoria de nivel.

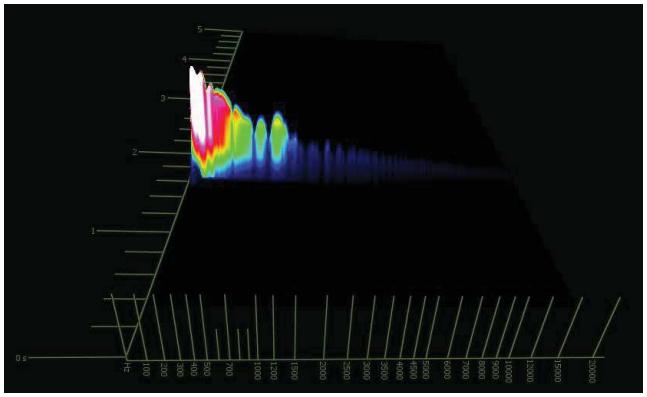


Fig. 114

Vista lateral del espectro de frecuencias de la muestra de bombo con función aleatoria de nivel.

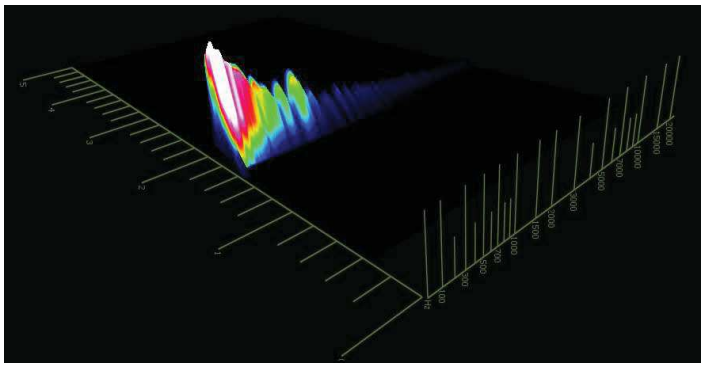


Fig. 115

Vista diagonal del espectro de frecuencias de la muestra de bombo con función aleatoria de nivel.

4.3.2.1. Análisis de Resultados Técnicos

Los resultados técnicos de esta prueba, muestran que aún cuando la función aleatoria de nivel trabaja correctamente y se encuentra activada en el sintetizador, esta no genera ningún cambio en los espectros obtenidos, más que un ligero aumento de nivel de toda la muestra.

4.3.2.2. Análisis de Resultados Auditivos

Los resultados auditivos obtenidos al realizar una escucha crítica de las muestras de audio, muestran que a pesar de que la función aleatoria de nivel se encuentre activada no se escucha un cambio auditivo perceptible.

4.3.3. Pruebas del Filtro Pasa Bajos (Prueba 3)

Estas pruebas tuvieron como finalidad medir el impacto sonoro causado por el filtro incorporado en el sintetizador, y consistieron en la comparación entre un sonido de violín y de bombo sintetizado sin ningún filtro aplicado, con los mismos sonidos sintetizados aplicando la configuración de filtro que entregue un sonido más realista.

En el caso de estas pruebas, se uso como muestra de violín y bombo sin filtro aplicado, a los sonidos de estos sintetizados experimentalmente (Violín sin filtrar.wav. y Bombo sin filtrar.wav).

A continuación se muestran los espectros obtenidos de la muestra sonora del violín filtrado (Violín filtrado.wav).

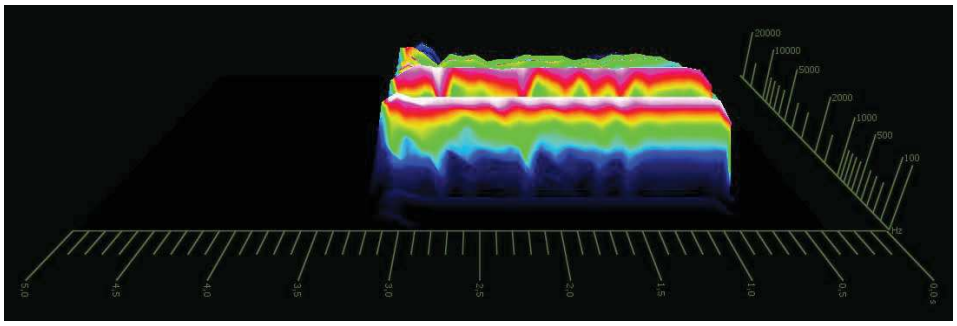


Fig. 116

Vista frontal del espectro de frecuencias de la muestra de violín filtrado.

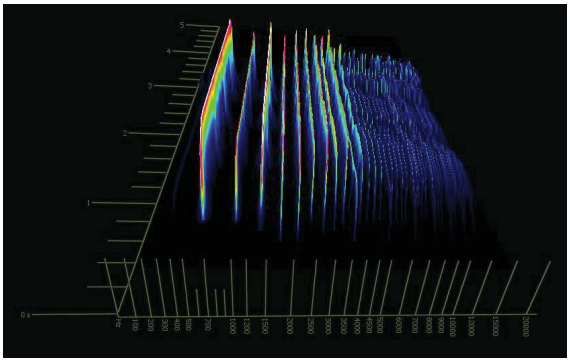


Fig. 117

Vista lateral del espectro de frecuencias de la muestra de violín filtrado.

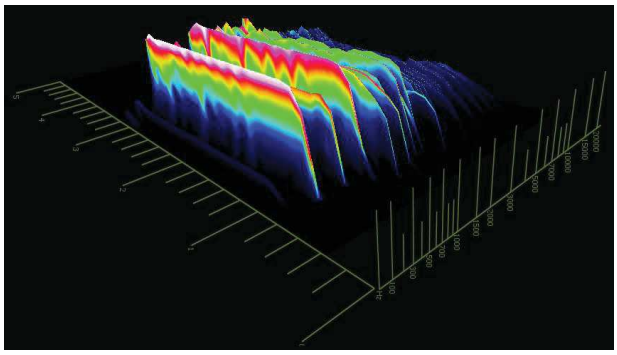


Fig. 118

Vista diagonal del espectro de frecuencias de la muestra de violín filtrado.

A continuación se muestran los espectros obtenidos de la muestra sonora del bombo filtrado (Bombo filtrado.wav).

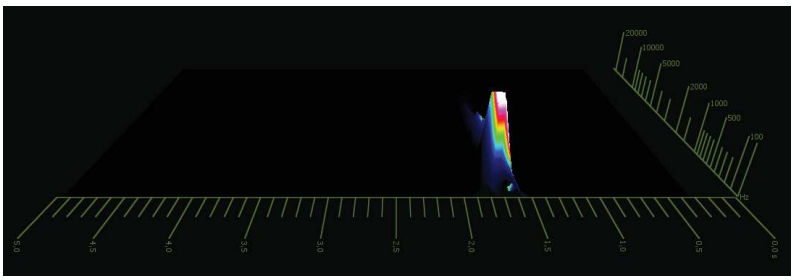


Fig. 119

Vista frontal del espectro de frecuencias de la muestra de bombo filtrado.

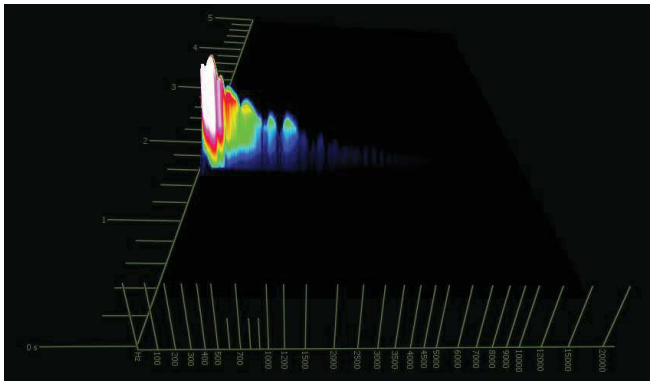


Fig. 120

Vista lateral del espectro de frecuencias de la muestra de bombo filtrado.

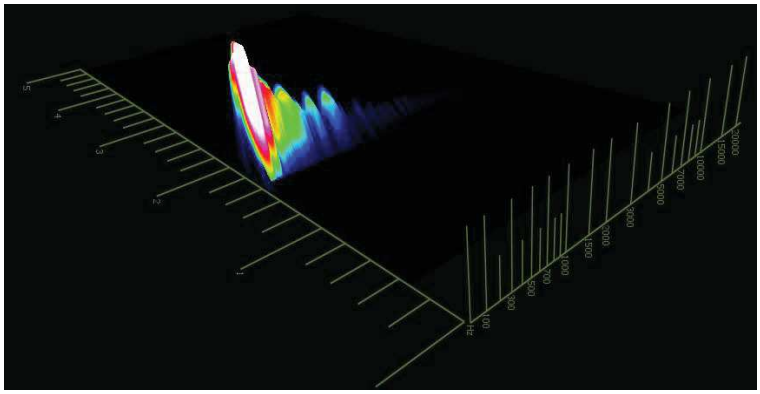


Fig. 121

Vista diagonal del espectro de frecuencias de la muestra de bombo filtrado.

4.3.3.1. Resultados Técnicos

De la comparación entre los espectros obtenidos de las seis muestras presentadas en los anteriores gráficos, resulta evidente que el filtro se comporta como era de esperarse, restringiendo el paso de altas frecuencias de acuerdo a los parámetros establecidos.

4.3.3.2. Resultados Auditivos

En el caso del violín, se utilizó al filtro de forma interpretativa para simular el sonido natural característico del instrumento generado por el frotamiento del

arco sobre las cuerdas, dando un resultado aceptable. En el caso del bombo, el filtro es percibido auditivamente, pero sin mejora significativa del sonido sintetizado.

4.3.4. Pruebas del Analizador de Espectro (Prueba 4)

De todas las pruebas realizadas, esta es la única conformada únicamente por una parte técnica, y consistió en la comparación de los espectros entregados por el analizador incorporado en el sintetizador con los espectros entregados por el analizador utilizado para todas las pruebas, estos espectros se presentan bidimensionalmente a diferencia de los anteriores, esto debido a que el sintetizador programado no dispone de un analizador espectral en tres dimensiones para su comparación.

Para prueba se utilizó las siguientes muestras: el sonido de violín sintetizado previamente pero ejecutado en la nota musical A3 (Violin prueba analizador espectral.wav), el sonido de bombo sintetizado previamente (bombo prueba analizador espectral.wav), y un sonido nuevo creado en el sintetizador (Sonido nuevo prueba analizador espectral.wav).

A continuación se muestran los espectros obtenidos de la muestra de violín.

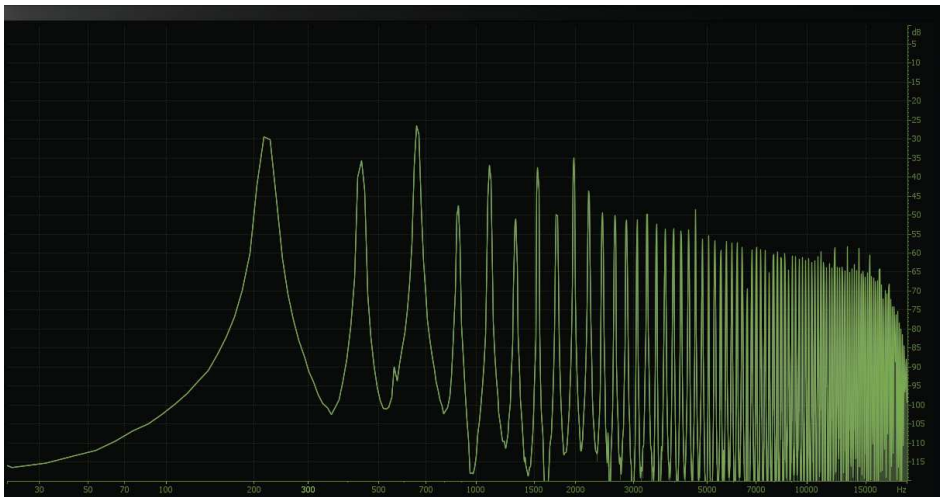


Fig. 122

Análisis espectral de la muestra de violín, por el analizador de prueba.

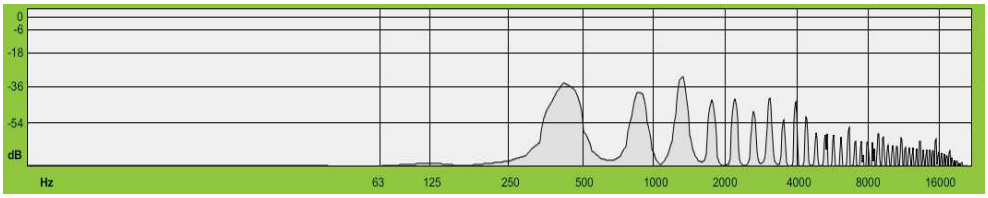


Fig. 123

Análisis espectral de la muestra de violín, por el analizador incorporado en el sintetizador.

A continuación se muestran los espectros obtenidos de la muestra bombo.

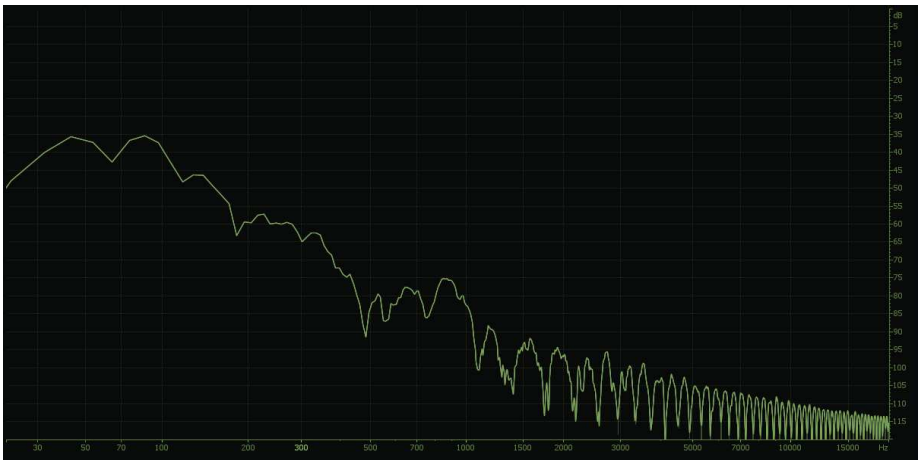


Fig. 124

Análisis espectral de la muestra de bombo, por el analizador de prueba.

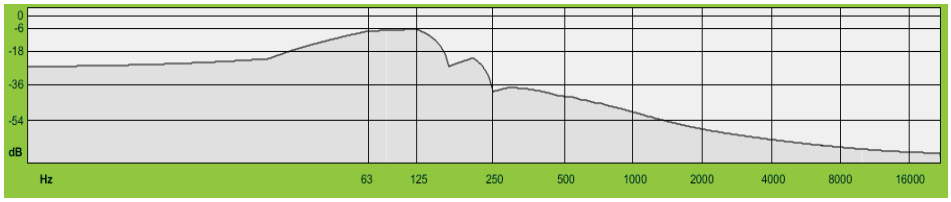


Fig. 125

Análisis espectral de la muestra de bombo, por el analizador incorporado en el sintetizador.

A continuación se muestran los espectros obtenidos del nuevo sonido creado en el sintetizador.

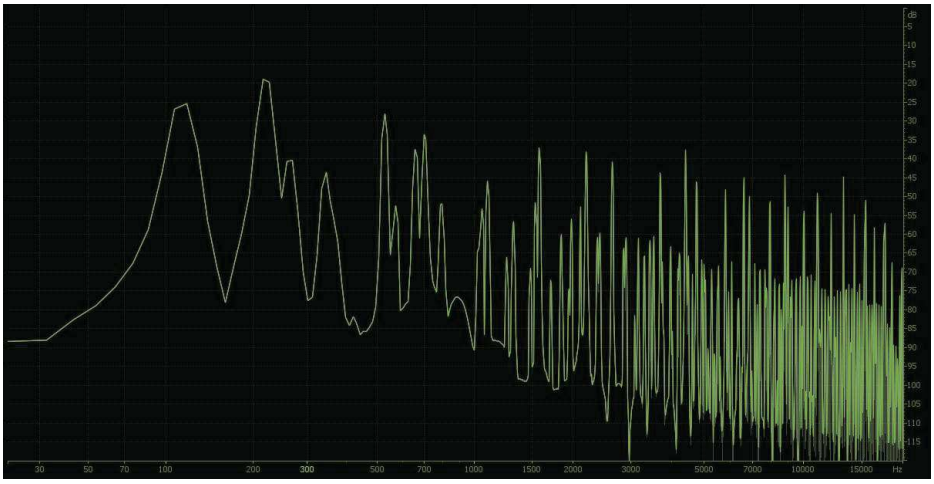


Fig. 126

Análisis espectral de la muestra del nuevo sonido creado, por el analizador de prueba.

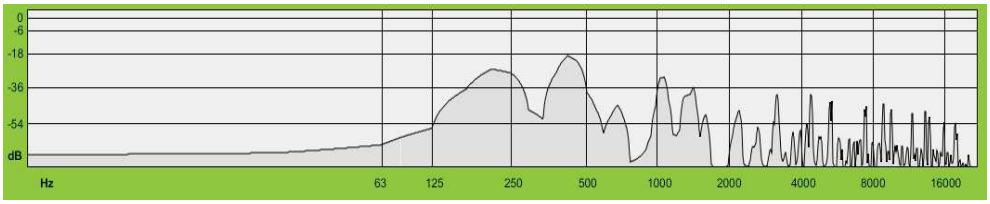


Fig. 127

Análisis espectral de la muestra del nuevo sonido creado, por el analizador incorporado en el sintetizador.

Previo a los resultados, hay que considerar la gran calidad del analizador espectral bidimensional de la función *Meter Bridge* del *plug-in* Ozone 5 Advanced de la compañía Izotope, dado que este analizador es dedicado para este fin, mientras que el analizador incorporado en el sintetizador es una función extra del mismo.

Si bien en rasgos generales los espectros mostrados en ambos analizadores son similares, lo primero que salta a la vista es la calidad y exactitud del analizador de prueba en relación al incorporado en el sintetizador. Esta diferencia de precisión y fidelidad es mucho más evidente en el análisis de bajas frecuencias, particularmente en las muestras de bombo.

Por otro lado, el analizador de prueba tiene una interface gráfica más útil, ya que brinda datos mucho más específicos gracias a su mejor discretización de ejes. Además, dado el tamaño de la ventana en la que se presentan los resultados del análisis espectral, estos pueden ser mejor visualizados e interpretados.

A pesar de todo lo previo, el analizador espectral incorporado en el sintetizador cumple satisfactoriamente el trabajo para el que fue diseñado, ya que está pensado para dar una idea del sonido que se está construyendo al sintetizar, mas no ser una herramienta de análisis espectral dedicado.

4.3.5. Pruebas de Creación de Sonidos (Prueba 5)

Estas pruebas tenían como finalidad determinar la capacidad del sintetizador para crear sonidos nuevos, y la facilidad, o no, asociada a este proceso. De todas las pruebas realizadas esta fue la única que consistió solamente en una parte estética.

Para esta prueba se propuso la creación de tres nuevos sonidos sintetizados, y en someterlos a una apreciación auditiva.

A continuación se presentan las configuraciones realizadas en el sintetizador para generar los tres nuevos sonidos (sonido nuevo 1 .wav. sonido nuevo 2 .wav, sonido nuevo 3 .wav).

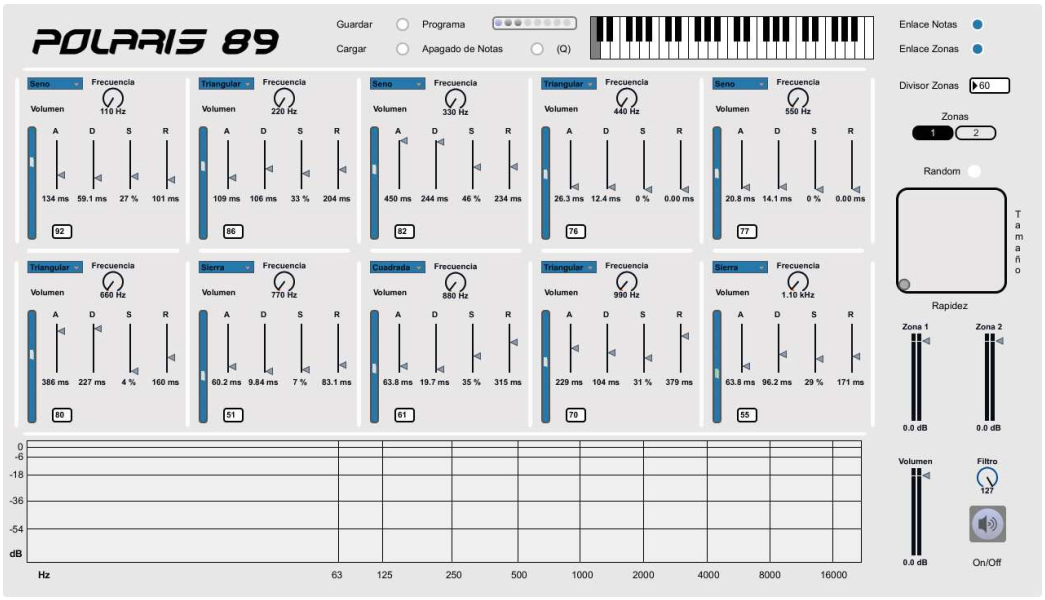


Fig. 128

Configuración de parámetros del sintetizador para el sonido nuevo 1.

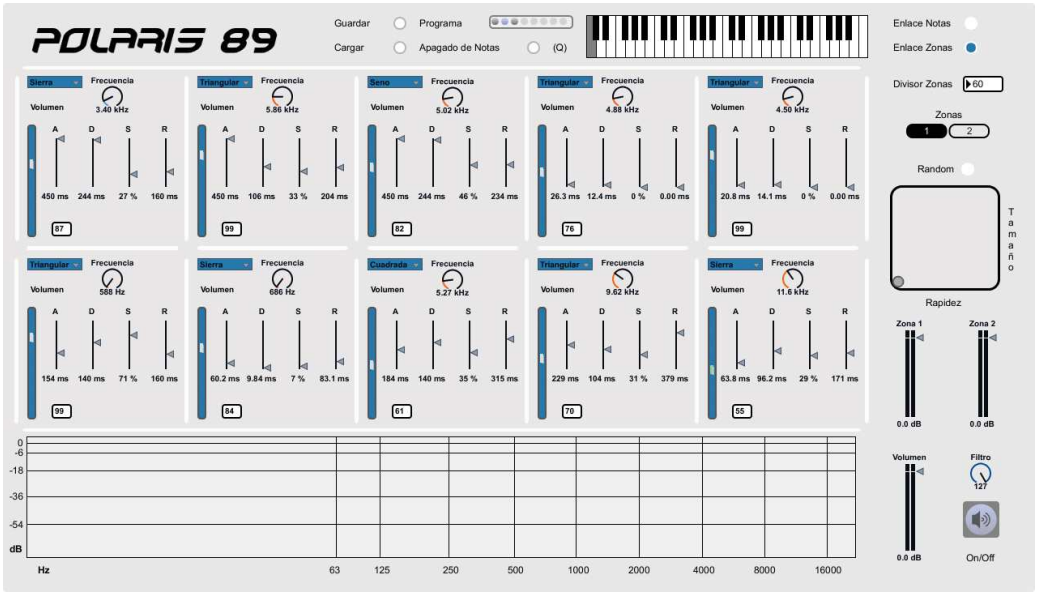


Fig. 129

Configuración de parámetros del sintetizador para el sonido nuevo 2.

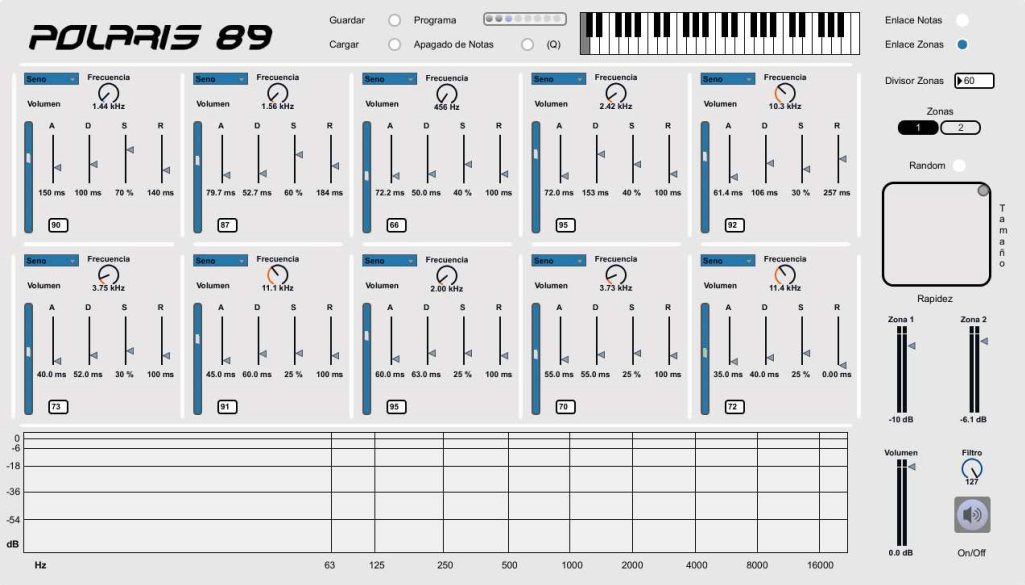


Fig. 130

Configuración de parámetros del sintetizador para el sonido nuevo 3.

Finalmente se decidió realizar una última prueba con el sintetizador, que consistió en la ejecución de una melodía sencilla con los tres sonidos nuevos sintetizados (melodía sonido nuevo 1.wav, melodía sonido nuevo 2.wav, melodía sonido nuevo 3.wav).

De la primera parte de esta prueba, se puede decir que el sintetizador presenta una gran facilidad al momento de crear sonidos nuevos y muy diferentes entre sí, ya que la variación pequeña de parámetros en algunos casos genera diferencias sonoras abismales. Además dados la cantidad de parámetros disponibles en el sintetizador, las posibilidades de creación de sonidos nuevos son muy amplias.

También hay que recalcar que al momento de usar el sintetizador sin la función de enlace de notas activada, este se convierte en una gran herramienta de creación de nuevos sonidos, considerando que con la función activada, se restringe ligeramente esta posibilidad, volviéndose el sintetizador más adecuado para la emulación sonora de instrumentos con alturas definidas.

5. Estudio Económico

5.1. Estudio de Costos

Dada la naturaleza tecnológica del proyecto, los principales costos de este se derivan de los equipos y del *software* que fueron necesario para su desarrollo.

Si bien en términos generales la tesis fue desarrollada en un computador, y en un entorno de programación específico, para los análisis de obtención de datos, y para las pruebas de funcionamiento del sintetizador, fue necesario ocupar otros programas dedicados, así como equipos externos que permitan obtener mejores resultados. También fue necesaria una conexión a Internet permanente para tener acceso a información de gran necesidad.

Tabla 5.

Tabla de Costos.

Rubro	Unidad	Cantidad	Precio	Total
Computador con sistema operativo Windows	u	1	600.00	600.00
Computador MacBook Pro	u	1	800.00	800.00
Conexión a internet	mes	10	19.90	199.00
Capacitación en entorno Max/MSP	hora	8	20.00	160.00
Software Max/MSP 5.1.9.	u	1	399.00	399.00
Controlador MIDI M-Audio Axiom 49	u	1	299.99	299.99
Software Pro Tools 10	u	1	700.00	700.00
Plug-in Izotope Advanced 5	u	1	999.00	999.00
Gastos Varios	glb	2	100.00	200.00
Total				4356.00

A continuación se detalla la información referente a los rubros:

- Computador con sistema operativo Windows: En cada aspecto del desarrollo de este proyecto está involucrado el uso de un computador, motivo por el que resulta imprescindible para su ejecución.
- Computador MacBook Pro: Si bien el proyecto podía ser desarrollado únicamente en una computadora con sistema operativo Windows, a lo largo del proyecto se usó también una computadora con sistema operativo OS X, con la finalidad de crear una aplicación *stand alone* tipo *.app, compatible con computadoras tipo Mac, no solo una aplicación tipo *.exe, compatible con computadoras PC.
- Conexión a Internet: La mayor fuente disponible de información relacionada con audio, *software*, instrumentos musicales, etc., hoy en día es Internet, por lo cual resultó necesario en todo momento tener una conexión a esta red.
- Capacitación en entorno Max/MSP: Dado que los encargados de desarrollar este proyecto no tenían capacitación previa en este entorno de programación, fue necesario que se capaciten en el uso básico de este programa. Si bien en gran término la capacitación fue de un tipo autodidacta, sin una capacitación externa inicial, resulta muy difícil conocer el funcionamiento del entorno Max/MSP, así como los aspectos fundamentales para su uso.
- Software Max/MSP 5.1.9.: Se considera la versión más actual, y la única disponible para adquisición en el momento de desarrollo del proyecto.
- Controlador MIDI M-Audio Axiom 49: Dentro de uno de los objetivos del proyecto estaba incluido el de permitir el manejo de los componentes del sintetizador, por medio de superficies de control externas MIDI; motivo por el cual resulta necesario tener a disposición una superficie de control externa MIDI.
- Software Pro Tools 10: Para tener un correcto manejo de las muestras de audio, y para poder ejecutar el *plug-in* Izotope Advanced 5, es necesaria una *DAW* (*Digital Audio Workstation*).

- *Plug-in* Izotope Advanced 5: Plug-in utilizado para obtener los datos de análisis espectral tridimensional.
- Gastos Varios: Dentro de este apartado de gastos varios se consideran los costos de: impresión, movilización, alimentación, etc., que surgieron durante el desarrollo del proyecto.

5.2. Análisis Costo-Beneficio

En el costo de desarrollo del proyecto se incluyen rubros que en la actualidad una persona dedicada profesionalmente al audio puede poseer o tener a disposición. Por ejemplo: las computadoras, la conexión a Internet, el software Pro Tools (o cualquier *DAW* a disposición), el *plug-in* Izotope Advanced 5 (o cualquier otra opción de *plug-in* con similares características), y un controlador MIDI externo.

A pesar de lo anterior, el análisis de costos se lo realizó partiendo desde cero, y suponiendo que ese sea el caso, se considera que el costo de desarrollo de este proyecto es muy bajo frente a los posibles beneficios que este puede implicar.

En la actualidad sobran los ejemplos de casos en los que una buena idea o el prototipo de un programa innovador, son adquiridos por grandes empresas encargadas de desarrollar tecnología, a precios considerablemente altos.

6. Proyecciones

Proyecciones de posibles expansiones del sintetizador.

A continuación se detallan posibles opciones de expansión del sintetizador:

- Aumento del número de bloques de generación de sonidos.
- Incorporación de un LFO al sintetizador.
- Ampliación del filtro incorporado en el sintetizador, para que incluya otros tipos de filtraje y otros parámetros como resonancia.
- Incorporación de controles de paneo en el sintetizador en cada bloque de generación de sonido.
- Posibilidad de usar al sintetizador con salidas monofónicas, estereofónicas, o de sonido envolvente (5.1).
- Incorporación de efectos de sonido tales como: reverberación, retardo, compresión, distorsión, etc.
- Incorporar un ecualizador, que permita modificar a los sonidos sintetizados.
- Incorporar la función de síntesis por distorsión de fase al sintetizador.
- Aumentar las opciones de formas de onda usadas para sintetizar sonidos.
- Permitir el uso de sonidos externos al sintetizador, para la generación sonora en los bloques de generación de sonido.

Proyecciones de posibles proyectos

Estudio psicoacústico de componentes espectrales escuchadas por el oído humano, enfocado a determinar la importancia de los armónicos e inarmónicos en el timbre percibido por los seres humanos.

Análisis y obtención de datos de envolventes temporales de amplitud de las componentes espectrales de diferentes instrumentos musicales, y sonidos reales.

Análisis cualitativo y cuantitativo entre uno o varios ADSR implementado en hardware, frente a uno o varios ADSR implementados en sintetizadores virtuales.

Mejoramiento del objeto *adsr~* disponible en Max/MSP.

Creación de un analizador espectral tridimensional en el entorno de programación Max/MSP.

Creación de un ADSR virtual que incluya los parámetros de porcentaje de amplitud al finalizar el tiempo de decaimiento y sostenimiento.

Creación de una superficie de control MIDI dedicada para el sintetizador POLARIS 89.

7. Conclusiones y Recomendaciones

7.1. Conclusiones

Se concluye que gracias al conocimiento en la materia de audio; el estudio de: el método de síntesis aditiva, los diferentes componentes de síntesis, y la arquitectura utilizada en la construcción de sintetizadores; y a los datos obtenidos de los análisis de espectros de frecuencias de diversas muestras; es posible diseñar un sintetizador que cumpla con los requerimientos propuestos para el proyecto. Además, utilizando el entorno de programación Max/MSP, se puede implementar un sintetizador virtual aditivo de sonido, que incorpore una interface de programación y visualización gráfica en tiempo real del espectro armónico de la señal siendo sintetizada, el cual permite la creación de nuevos sonidos, y la emulación de sonidos realistas con ciertas limitaciones.

Ya que el método de síntesis utilizado es aditivo, se puede concluir que: es posible sintetizar sonidos en esta aplicación mediante la manipulación y suma de sus componentes espectrales, para así conformar el timbre deseado.

Los componentes de síntesis utilizados, así como su configuración e interconexión, están condicionados a las posibilidades brindadas por el entorno de programación utilizado, así como al método de síntesis implementado.

El entorno de programación Max/MSP, por medio de su sistema de programación orientada a objetos, permitió la programación del sintetizador, cumpliendo con las necesidades propuestas para este.

Gracias a las capacidades de manejo de datos de control MIDI incorporadas en Max/MSP, fue posible implementar el control mediante superficies de control externas MIDI, de los diferentes componentes del sintetizador.

Mediante la escucha y análisis crítico fue posible realizar una comparación cualitativa entre las señales sintetizadas que pretendían emular a instrumentos acústicos, y los sonidos generados por estos dichos instrumentos, encontrando así diferencias y similitudes entre estas.

Dado que las formas de onda senoidales se componen de un tono puro y no tienen armónicos, cuando se las utiliza para emular espectros complejos, resultan insuficientes diez bloques de generación de sonido generando ondas senoidales. Considerando por ejemplo que para sintetizar el espectro de un bajo eléctrico las capacidades del sintetizador son suficientes, pero para sintetizar espectros como por ejemplo el del piano, las capacidades del sintetizador son insuficientes.

Para poder sintetizar sonidos realistas mediante un método analítico (análisis de muestras de sonido y extracción de datos para la síntesis), se necesitan mayor cantidad de datos (todos los parámetros de ADSR, y las amplitudes exactas de todos los armónicos con sus frecuencias), y estos deben ser de mayor exactitud (no valores aproximados) para que el sonido sea una emulación más fiel del instrumento acústico.

Debido a que el parámetro de sostenimiento del objeto *adsr~* en Max/MSP es expresado en porcentaje y no en tiempo, tiene la ventaja de que al ejecutar el sintetizador, la duración del sonido generado está ligada al tiempo que permanezca la tecla presionada. Pero por otro lado esta característica dificulta la emulación de sonidos realistas ya que en el entorno acústico, el sostenimiento de los sonidos no depende de este factor sino de las propiedades acústicas de la fuente de sonido.

Para agregar mayor realismo a sonidos sintetizados que emulen sonidos reales, dentro de los generadores de envolvente deberían existir por lo menos dos parámetros más aparte del ataque, decaimiento, sostenimiento y relevo. El primero debería ser un porcentaje de amplitud al que llegue el sonido una vez se concluya el tiempo de decaimiento, y el segundo un porcentaje de amplitud al que llegue el sonido una vez se concluya el tiempo de sostenimiento, teniendo en cuenta que el cien por ciento de la amplitud es el nivel alcanzado una vez transcurre el tiempo de ataque.

Dadas las características de diseño del sintetizador este presenta el mismo inconveniente de los sintetizadores desarrollados en hardware, ya que para

tener mayor polifonía interpretativa hay que duplicar todos los generadores de sonido cada vez que se quiera aumentar una voz.

Una desventaja que presenta Max/MSP es que el rango de amplitud que maneja es de cero como mínimo y uno como máximo, lo que obliga al usuario a escalar los valores que ingresa al programa para que no se generen distorsiones, lo que en muchos casos genera inexactitud o problemas con los objetos de interface gráfica.

El sintetizador como tal tiene un alto consumo de recursos computacionales debido a la construcción del mismo, esto ya que en todo momento tiene a trescientos veinte generadores de señales funcionando.

Max/MSP es poco eficiente al momento de generar señales con altas frecuencias ya que al hacerlo consume demasiados recursos de la computadora donde se lo ejecute, principalmente DSP. Este es un gran inconveniente ya que puede llegar a sobrecargar el DSP, haciendo que funcione mal la aplicación en todos sus aspectos.

Gran parte del sonido percibido por un oyente de un instrumento musical, viene dado por las características del entorno acústico en el que este se esté ejecutando (resonancias, reflexiones, condiciones meteorológicas, etc.), características que no son posibles de emular en el sintetizador, lo que resta realismo al sonido generado.

Las características sonoras de una ejecución musical están influenciadas por muchas variantes (técnica de ejecución del instrumento, estado del instrumento, calidad del instrumento y componentes), las cuales se pierden al momento de ejecutar un sintetizador. Es por esto que la interpretación de un músico jamás va a sonar igual cuando ejecuta un instrumento, que cuando ejecuta en un sintetizador que emula al mismo.

El límite de realismo que se puede alcanzar en los sonidos sintetizados, depende en gran medida de la persona que calibre los parámetros del sintetizador usado para poder obtener los resultados deseados con un

sintetizador, es indispensable conocer el funcionamiento de los diferentes componentes del sintetizador usado, y una noción de las características sonoras de lo que se desea sintetizar.

Si bien el filtro incorporado en el sintetizador funciona correctamente, el impacto que este tiene en el resultado estético es leve, y no aporta de forma significativa.

Aun cuando el analizador de espectro incorporado en el sintetizador cumple satisfactoriamente con su función, sería conveniente que se pueda visualizar a este en una ventana emergente y de mayor tamaño, para tener una mejor interpretación de los datos.

Cuando los bloques de generación de sonido están calibrados para producir formas de onda tipo senoidal, las variaciones en los parámetros de ADSR generan cambios más notorios en sonido sintetizado, que las relaciones de volumen entre los diferentes bloques.

Cuando el sintetizador está con la función de enlace de notas apagada y las frecuencias calibradas en los bloques de generación de sonido no son múltiplos enteros entre sí, cualquier cambio de parámetros (ADSR, volumen o forma de onda) produce cambios muy notorios, esto ya que al no ser las frecuencias armónicas entre sí, sino disonantes, son más fáciles de diferenciar entre sí. Es debido a esto que el sintetizador con la función de enlace de notas apagada permite la creación de sonidos nuevos más fácilmente, ya que los instrumentos musicales melódicos y armónicos conocidos típicamente no tienen espectros disonantes.

Con una mala configuración en los parámetros de ADSR, se producen *clips* digitales. Esto especialmente en el tiempo de decaimiento y tiempo de relevo debido a que, cuando estos parámetros son cero, la señal no tiene un intervalo de tiempo para reducir su amplitud, produciéndose un cambio muy brusco de esta.

Concordando con la información generalizada sobre síntesis aditiva, se puede concluir que la calibración de parámetros en un sintetizador aditivo para la emulación de sonidos reales es trabajosa y lenta, esto debido a la gran cantidad de parámetros que se deben tomar en cuenta.

Max/MSP en su versión 5.1.9, presenta un mal manejo de datos de notas MIDI, esto debido a que el identificador de estas en el programa, tienen un error de calibración ya que está desfasado una octava completa, es decir, que cuando se ejecuta una nota de A4 que corresponde a la nota MIDI 69 el programa la identifica como si correspondiera a la nota A3.

El rango de notas que se pueden ejecutar con el sintetizador está directamente ligado a las capacidades de procesamiento del computador en que se lo esté usando. A pesar de esto, con una computadora estándar (como las usadas para las pruebas), se tiene un buen rango de notas ejecutables.

Si bien las zonas de teclado fueron implementadas para permitir la emulación detallada de sonidos cuyo timbre varía en función a la altura tonal, en la práctica la mayor ventaja que estas brindan, es la de poder emular dos sonidos totalmente diferentes al mismo tiempo con un mismo sintetizador.

Aunque el entorno de programación SuperCollider tiene un mejor manejo de datos y permite realizar procesamientos digitales de señales más robustos que Max/MSP, fue seleccionado este segundo para el desarrollo del proyecto ya que no es necesaria una capacitación muy extensa para manejarlo en términos generales.

Una gran ventaja que presenta el entorno de programación Max/MSP, es la capacidad de crear interfaz gráfica en el mismo entorno de programación, lo que en otros ambientes de programación se debería realizar en un *software* distinto.

A pesar de que la versión *stand alone* del sintetizador consume muchos menos recursos que el caso en que se ejecuta el *patch* desde Max/MSP, hacerlo de este modo presenta gran cantidad de ventajas frente a la otra opción. Esto principalmente porque utilizando a *Rewire* es posible conectar al sintetizador con todos los programas que aceptan este modo de interconexión, por ejemplo Pro Tools.

Si bien el sintetizador cumple con los objetivos propuestos para este proyecto, este tiene una gran posibilidad de expansión, lo que lo convertiría en un instrumento virtual mucho más apetecible.

Dadas las características de audición del oído humano, aun cuando dos sonidos tienen espectros frecuenciales muy diferentes entre si, al escucharlos, estos sonidos pueden ser percibidos como de timbres similares.

7.2. Recomendaciones

En los anexos de este trabajo de titulación, se encuentra un *patch* llamado "*Polaris 89.maxpat"*, el cual es el *patch* de desarrollo del sintetizador. Si se desea ver la programación detallada de la implementación del mismo, se recomienda usar al programa Max/MSP/Jitter versión 5.1.9. Además, si se desea abrir los *subpatches* de este, se debe tener en cuenta que el *patch* de nombre "*Osciladores.maxpat"* es en el que se encuentran todos los objetos de generación de sonidos, mientras que en el *patch* "*Visor.maxpat*", se encuentra el analizador espectral.

En caso de no tener a disposición una licencia del programa Max/MSP/Jitter, y querer abrir los diferentes *patchs* que conforman este proyecto, se recomienda usar la versión de prueba gratuita del programa, con una duración de treinta días, disponible en la página oficial de la empresa *Cycing74*, comercializadora del *software.*

Dentro de los anexos, también está incluida una sesión del Pro Tools en la cual se encuentran las muestras de audio organizadas en función de las pruebas realizadas. En esta sesión está cargado en todas las pistas el *plug-in* Ozone

Advanced 5, que fue usado para el análisis. En caso de querer repetir el mismo experimento de análisis de las muestras realizado, se recomienda adquirir este *plug-in.* En caso de no tener acceso a este, se recomienda buscar otra opción de *plug-in* disponible en el mercado con las mismas funciones.

En caso de querer sintetizar los sonidos usados para las pruebas a las que fue sometido el sintetizador, dentro de los anexos están incluidos los archivos: *"PresetPrueba1.json"* (incluye la calibración de parámetros usados para sintetizar los sonidos de las pruebas uno y dos), *"PresetPrueba2.json"* (incluye la calibración de parámetros usados para sintetizar los sonidos de las pruebas tres y cuatro), y *"PresetPrueba3.json"* (incluye la calibración de parámetros usados para sintetizar los sonidos de la prueba cinco).

Además, dentro de los anexos del proyecto están incluidas las muestras de audio grabadas utilizadas para todas las pruebas a las que se sometió al sintetizador, en caso de querer escucharlas.

Dentro de los anexos, están incluidas las versiones *stand alone* del sintetizador, para los sistemas operativos Windows y Mac. En caso de no tener acceso a una licencia de Max/MSP, es posible utilizar estas versiones con las funciones de generación de sonido del sintetizador, pero no se tiene acceso a las diferentes opciones de calibración de DSP que permite Max/MSP.

Debido al gran tamaño de las imágenes de los *patch* del sintetizador, estas no pueden ser incluidas dentro del documento en un tamaño que permita la visualización correcta y cómoda de todas sus partes, por esto, dentro de los anexos virtuales están incluidas las imágenes de los *patch* en tamaño completo para su visualización.

Al momento de utilizar el sintetizador, un posible caso por el cual este no genere ningún sonido, es que si el ADSR tiene todos sus parámetros calibrados en cero, este no tiene capacidad alguna de generar señales.

Cuando se abre el sintetizador y se calibra algún parámetro desde cero, hay que considerar que si no está la función de enlace de notas activada, esta

calibración solo será válida para la zona uno del sintetizador, este es un posible motivo por el cual no suene el sintetizador en su segunda zona.

Un proyecto de estas características con mayor asesoría, mayores recursos monetarios, y con recursos humanos con mayor capacitación, bien podría ser comprado por una empresa dedicada a crear instrumentos virtuales, o *plug-ins* tipo VST o RTAS. Inclusive dada la gran difusión en la actualidad de los teléfonos celulares inteligentes con amplias capacidades de procesamiento (cada vez en aumento), bien podría desarrollarse un prototipo de este proyecto para un sistema operativo tipo Android o iOS.

Si bien no hay pruebas de recursos mínimos exactos necesarios para correr al programa, y ha funcionado en computadoras con recursos relativamente bajos para el estándar actual, se recomienda las siguientes especificaciones para tener acceso a todas las funciones del sintetizador: procesador *IntelCore i5* de 2,5 GHz y 4 GB de memoria RAM.

8. Referencias

8.1. Libros

[1] Ballou, G. (1988). *Handbook for Sound Engineers* (1a. ed.). Indiana, Estados Unidos: Howard W. & Sams Co. a division of Macmillan Inc.

[2] Cycling '74 (2012). *Max 5 Help and Documentation*. Recuperado el 8 de Junio de 2012 de http://cycling74.com/docs/max5/vignettes/intro/docintro.html.

[3] Grupo Editorial OCÉANO. (1998). *Diccionario de la lengua Española* (1988a. ed.). Barcelona, España: OCÉANO-ÉXITO, S.A.

[4] Gutiérrez, E. (2009). *Introducción a la Síntesis Sonora.* Recuperado el 28 Mayo de 2012 de http://www.dtic.upf.edu/~egomez/teaching/sintesi/SPS1/Tema 1-IntroduccionSintesi.pdf

[5] Jácome H. (2009). *Sistema de Generación de Imágenes y Patrones Visuales a partir de una Señal de Audio: considerando Amplitud, Frecuencia y Espectro, destinado principalmente al Refuerzo Sonoro.* (1a. ed.). Quito, Ecuador: "Trabajo de Titulación" UDLA.

[6] Kinsler L. (1992). *Fundamentos de Acústica* (3a. ed.). California, Estados Unidos: LIMUSA S.A. de C.V.

[7] Madisetti, V., Williams, D. (1999). *Digital Signal Processing* Handbook (1a. ed.). Atlanta, Georgia, Estados Unidos: Chapman & Hall/CRCnetBase.

[8] Martínez, D. (2008). *LOS SINTETIZADORES una breve introducción.* Recuperado el 5 de junio de 2012 de http://es.scribd.com/doc/54170499/10/La-sintesis-de-tabla-de-ondas.

[9] Milstead, B. (2001). *Home Recording Power* (1a. ed.). Ohio, Estados Unidos: Muska & Lipman Publishing.

[10] Miyara, F. (2004). *Acústica y Sistemas de Sonido* (4a. ed.). Rosario, Argentina: Universidad Nacional de Rosario.

[11] Núñez A. (1992). *Informática y Electrónica Musical* (2a. ed.). Madrid, España: Editorial Paraninfo.

[12] Quintanilla, M. (2010). *Acústica Musical.* Recuperado el 3 Junio de 2012 de http://cpms-acusticamusical.blogspot.com/2010/05/el-audio-analogico.html.

[13] Smith S. (1999). *The Scientist and Engineer's Guide to Digital Signal Processing* (2a. ed.). San Diego, California, Estados Unidos: California Technical Publishing.

[14] Watkinson, J. (1994). *An Introduction To Digital Audio* (1a. ed.). Gran Bretaña: Clays Ltd, St Ives plc.

ANEXOS

9. Anexos

9.1. Glosario

***.app:** Extensión de archivos ejecutables en el sistema operativo Mac OS.

***.exe:** Extensión de archivos ejecutables en el sistema operativo Windows.

***.wav:** Extensión de un archivo de audio sin compresión de datos.

-∞: Mínimo nivel de amplitud posible en una escale logarítmica.

AES: *Audio Engineering Society* (Sociedad de Ingeniería de Audio).

All notes off: Mensaje MIDI de apagado de todas las voces, utilizado en caso de notas estancadas.

AM: Abreviación de amplitud modulada.

Análisis de Fourier: Estudio matemático que postula la posibilidad de descomponer cualquier señal periódica en una suma de funciones seno y coseno, y viceversa.

Android: Sistema operativo de teléfonos inteligentes de Google.

Binario: En informática, binario hace referencia a un sistema de codificación que usa solamente dos valores (0 o 1) para representar datos.

Bit: Mínima unidad de información digital, representa un dígito del sistema binario.

C: Lenguaje de programación creado en 1972 y que proviene como una evolución del lenguaje B. Lenguaje orientado a la implementación de sistemas operativos.

C++: Lenguaje de programación que aparece a mediados de los años 80, el cual fue creado para mejorar el lenguaje C añadiendo mecanismos que permitan la manipulación de objetos.

Clip: Ruido de audio digital.

Controlador MIDI: Un controlador MIDI es un dispositivo con la capacidad de transmitir mensajes de datos MIDI a dispositivos externos. Entre la vasta cantidad de controladores MIDI existente, quizás los más difundidos son aquellos destinados a permitir ejecuciones musicales (teclados MIDI, guitarras MIDI, controladores de viento MIDI, etc.), estos típicamente tienen una sección similar a la de un instrumento musical y otra compuesta por controladores rotativos y deslizantes.

Cycling ’74: Empresa dedicada al desarrollo de *software*. Comercializadora y parcial creadora de Max/MSP.

dB: Símbolo (abreviatura) del decibel.

Decibel: Unidad convencional asignada a la expresión logarítmica de una magnitud o relación de magnitudes.

Delay: Retardo.

Depth: Profundidad.

DSP: *Digital Signal Processor* (Procesador Digital de Señales). Sistema dedicado a procesar señales de manera digital, parte fundamental en el procesamiento de audio dentro de un computador.

Encoder: Potenciómetro rotatorio típicamente encontrado en superficies de control MIDI.

External: Término que hace alusión a componentes externos que una aplicación puede tener dedicados a un procesamiento adicional. En el caso de Max/MSP, son módulos externos creados por terceras personas para suplir funciones que el programa en si no puede cumplir.

Factor de calidad (Q): En un filtro es la relación entre la frecuencia central de este y su ancho de banda.

FM: Abreviación de frecuencia modulada.

Full Scale: Término que hace referencia a todo el rango dinámico que se puede manejar en audio digital. Hay que considerar que 0 dB Full Scale no representa al valor nominal de la escala sino al máximo valor de esta, punto que al ser alcanzado produce saturación digital.

Ganancia: Coeficiente entre la señal de salida y la de entrada de un amplificador, a menudo expresada en dB. En este caso es la relación entre los niveles de salida y de entrada.

Gigabyte (GB): Aproximadamente mil millones de bytes, representa capacidad de almacenamiento.

GigaHertz (GHz): Mil millones de ciclos por segundo (10^9).

Hardware: Parte física de un sistema digital destinado al procesamiento de información. Complementaria al *software*.

Headroom: Es el exceso de nivel que puede manejar la salida respecto al nivel medio nominal de la señal antes de llegar a la saturación.

Hz: Símbolo o abreviatura de Hertz.

IntelCore i5: Procesador de computadora compuesto por un total de dos procesadores físicos, equivalentes a cuatro procesadores virtuales.

iOS : Sistema operativo del teléfono inteligente iPhone, desarrollado por la empresa Apple.

I/O Vector Size: Tamaño de vector de entrada/salida.

IRCAM: Acrónimo de: *Institut de Recherche et Coordination Acoustique/Musique* (Instituto de Investigación y Coordinación Acústica/Música).

Keyboard tracking: Seguimiento de teclado. Opción en instrumentos electrónicos, que al estar activada permite el cambio de afinación de los sonidos generados en función a la altura tonal que se ejecute.

KiloHertz (kHz): Mil ciclos por segundo.

Loop: Reproducción en bucle.

Master: Principal.

M-Audio: Empresa dedicada al desarrollo de soluciones de audio digital y MIDI para músicos y profesionales del audio.

Meter Brigde: Componente del *plug-in* Ozone 5 Advanced destinado a la medición de niveles de salida de este.

MIDI: Musical Instrument Digital Interface (Interfaz Digital de Instrumentos Musicales). Es un protocolo de comunicación serial, que permite a computadores, sintetizadores, secuenciadores, controladores y otros dispositivos de la música electrónica comunicarse para generar sonidos o controlar parámetros.

ms: Símbolo o abreviatura de milisegundos.

Nivel de Presión Sonora: Veinte veces el logaritmo decimal de la presión sonora dividida por la presión de referencia, es decir, 20 µPa. Es expresado en dB.

Note off: Mensaje MIDI de nota apagada.

Note on: Mensaje MIDI de nota encendida.

NPS: Abreviatura de Nivel de Presión Sonora.

Ozone 5 Advanced: *Plug-in* de la empresa Izotope.

Pa: Símbolo o abreviatura de Pascal.

Pascal: Unidad de presión adoptada internacionalmente, equivalente aproximadamente a la cienmilésima parte de la presión atmosférica.

Patch: Formato de documento de Max/MSP. A partir de la versión 5 adquiere la extensión *.maxpat, en versiones anteriores su extensión era *.pat.

Patcher: Espacio de trabajo de Max/MSP en el cual se crean los patch. Está compuesta por herramientas visuales, y es el lugar donde se crean e interconectan los objetos.

Pitch: Tono.

Play: Reproducir.

Plug in: *Software* desarrollado por terceros, destinados a expandir las funciones de un determinado programa.

Polifonía: Múltiples voces melódicas sonando al mismo tiempo.

Preset: Programa. En un sistema de guardado y carga de archivos, es un determinado conjunto de datos de calibración de parámetros.

Psicoacústica: Ciencia que se encarga de estudiar la percepción del sonido, en otras palabras, cómo el oído y el cerebro reciben e interpretan información que nos llega en forma de sonido.

RAM: *Random Access Memory* (Memoria de Acceso Aleatorio). El lugar de trabajo del software de un computador, pues proporciona almacenamiento para las instrucciones que recibe el procesador, y además, para guardar los resultados de dichos procesos.

Record: Grabación.

Resonancia de filtro: Característica del filtro muy utilizado en síntesis sonora. Es un parámetro que determina la amplificación o atenuación del filtraje en la frecuencia de corte.

Rewire: Protocolo de comunicación entre diferentes *software* dedicados al audio digital, creado por la empresa Propellerheads.

Ruido blanco: Señal cuya densidad espectral es constante con la frecuencia (todas las frecuencias aparecen en la misma proporción).

Ruido rosa: Señal cuya densidad espectral disminuye con la frecuencia. Tiene la particularidad de que su energía es la misma en cada banda de octava, motivo por el cual es usada como señal de prueba en calibración acústica y electroacústica.

Sample: Muestra de audio.

Sampling Rate: Frecuencia de muestreo.

Scheduler: Planificador de procesos en casos de sobrecarga de DSP en Max/MSP.

Señal moduladora: Señal que modula a una señal portadora en los casos de modulación FM o AM.

Señal portadora: Señal que es modulada por una señal moduladora en los casos de modulación FM o AM.

Signal Vector Size: Tamaño de vector de la señal.

Slider: Control deslizante de nivel.

Sobretono: Frecuencias que están por encima de la frecuencia fundamental, y puede o no ser múltiplo entero de esta.

Software: Parte intangible de un sistema digital destinado al procesamiento de información. Su función es la de manejo del hardware de estos sistemas. Complementaria al *hardware.*

Stand Alone: Aplicación ejecutable que solamente necesita de un sistema operativo que lo hospede para su funcionamiento.

Stand by: Estado de reposo.

Sub grave: Sonido cuya frecuencia es menor a 100 Hz.

Subpatch: *Patch* de Max/MSP que adquiere esta denominación cuando forma parte de otro *patch* superior. Para incluir un *patch* dentro de otro se usan determinados objetos que permiten su interconexión.

SuperCollider: Entorno de programación especializado en procesamiento de audio en tiempo real.

Switch: Interruptor.

V: Símbolo o abreviatura de Volt.

Vector Optimization: Optimización de cálculo de vectores en Max/MSP.

Velocity: En la norma MIDI es el mensaje que indica la intensidad con la que se ejecuta una nota musical.

Vibrato: Modulación de frecuencia o de fase de un sonido.

Voces: En un sistema de audio digital, el número de voces indica el número de sonidos distintos que este tiene capacidad de reproducir al mismo tiempo.

Voltaje: Diferencia de potencial eléctrico entre dos puntos de medición.

[µ]: Abreviatura de micro.

9.2. Líneas de Código del Sintetizador


```
{
   "patcher" :    {
      "fileversion" : 1,
      "appversion" :        {
         "major" : 5,
         "minor" : 1,
         "revision" : 9
      }
,
      "rect" : [ -8.0, 50.0, 1280.0, 692.0 ],
      "bglocked" : 0,
      "defrect" : [ -8.0, 50.0, 1280.0, 692.0 ],
      "openrect" : [ 0.0, 0.0, 0.0, 0.0 ],
      "openinpresentation" : 1,
      "default_fontsize" : 12.0,
      "default_fontface" : 0,
      "default_fontname" : "Arial",
      "gridonopen" : 0,
      "gridsize" : [ 15.0, 15.0 ],
      "gridsnaponopen" : 0,
      "toolbarvisible" : 1,
      "boxanimatetime" : 200,
      "imprint" : 0,
      "enablehscroll" : 1,
      "enablevscroll" : 1,
      "devicewidth" : 0.0,
      "boxes" : [           {
            "box" :               {
               "maxclass" : "fpic",
               "id" : "obj-12",
               "patching_rect" : [ 855.0, 915.0, 423.0, 54.0 ],
               "pic" : "Polaris 89.png",
               "presentation" : 1,
               "numinlets" : 1,
               "autofit" : 1,
               "numoutlets" : 0,
               "presentation_rect" : [ 15.0, 30.0, 339.0, 32.0 ]
            }

         }
,          {
            "box" :               {
               "maxclass" : "comment",
               "text" : "Zonas",
               "id" : "obj-63",
               "patching_rect" : [ 1155.0, 143.0, 48.0, 20.0 ],
               "fontsize" : 12.0,
               "presentation" : 1,
               "numinlets" : 1,
               "numoutlets" : 0,
               "fontname" : "Arial",
               "presentation_rect" : [ 1147.0, 128.0, 48.0, 20.0 ]
            }

         }
```

```
,          {
             "box" :                {
                "maxclass" : "newobj",
                "text" : "makenote 100 100",
                "id" : "obj-62",
                "patching_rect" : [ 165.0, 150.0, 110.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 3,
                "numoutlets" : 2,
                "outlettype" : [ "float", "float" ],
                "fontname" : "Arial"
             }

          }
,          {
             "box" :                {
                "maxclass" : "comment",
                "text" : "(Q)",
                "id" : "obj-60",
                "patching_rect" : [ 45.0, 135.0, 27.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 675.0, 45.0, 26.0, 20.0 ]
             }

          }
,          {
             "box" :                {
                "maxclass" : "newobj",
                "text" : "if $i1 == 12 then out1 else out2",
                "id" : "obj-83",
                "patching_rect" : [ 15.0, 30.0, 175.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 2,
                "outlettype" : [ "", "" ],
                "fontname" : "Arial"
             }

          }
,          {
             "box" :                {
                "maxclass" : "newobj",
                "text" : "key",
                "id" : "obj-61",
                "patching_rect" : [ 0.0, 0.0, 59.5, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 0,
                "numoutlets" : 4,
                "outlettype" : [ "int", "int", "int", "int" ],
                "fontname" : "Arial"
             }

          }
,          {
```

```
            "box" :                {
               "maxclass" : "number",
               "id" : "obj-39",
               "patching_rect" : [ 1019.0, 831.0, 50.0, 20.0 ],
               "fontsize" : 12.0,
               "numinlets" : 1,
               "numoutlets" : 2,
               "outlettype" : [ "int", "bang" ],
               "fontname" : "Arial"
            }

         }
,        {
            "box" :                {
               "maxclass" : "newobj",
               "text" : "ctlin 25",
               "id" : "obj-9",
               "patching_rect" : [ 15.0, 105.0, 49.0, 20.0 ],
               "fontsize" : 12.0,
               "numinlets" : 1,
               "numoutlets" : 2,
               "outlettype" : [ "int", "int" ],
               "fontname" : "Arial"
            }

         }
,        {
            "box" :                {
               "maxclass" : "comment",
               "text" : "Apagado de Notas",
               "linecount" : 2,
               "id" : "obj-75",
               "patching_rect" : [ 15.0, 60.0, 67.0, 34.0 ],
               "fontsize" : 12.0,
               "presentation" : 1,
               "numinlets" : 1,
               "numoutlets" : 0,
               "fontname" : "Arial",
               "presentation_rect" : [ 510.0, 45.0, 119.0, 20.0 ]
            }

         }
,        {
            "box" :                {
               "maxclass" : "button",
               "id" : "obj-73",
               "patching_rect" : [ 15.0, 135.0, 20.0, 20.0 ],
               "presentation" : 1,
               "numinlets" : 1,
               "blinkcolor" : [ 0.168627, 0.47451, 0.666667, 1.0 ],
               "numoutlets" : 1,
               "outlettype" : [ "bang" ],
               "presentation_rect" : [ 645.0, 45.0, 20.0, 20.0 ]
            }

         }
,        {
            "box" :                {
```

```
                "maxclass" : "message",
                "text" : "stop",
                "id" : "obj-70",
                "patching_rect" : [ 15.0, 180.0, 33.0, 18.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "panel",
                "id" : "obj-57",
                "patching_rect" : [ 1275.0, 150.0, 128.0, 128.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "presentation_rect" : [ 24.0, 78.0, 1035.0, 4.0 ],
                "bgcolor" : [ 0.988235, 0.988235, 0.968627, 1.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "panel",
                "id" : "obj-52",
                "patching_rect" : [ 1260.0, 135.0, 128.0, 128.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "presentation_rect" : [ 24.0, 525.0, 1035.0, 4.0 ],
                "bgcolor" : [ 0.988235, 0.988235, 0.968627, 1.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "On/Off",
                "id" : "obj-29",
                "patching_rect" : [ 435.0, 750.0, 150.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 1185.0, 675.0, 51.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "live.dial",
                "varname" : "live.dial",
                "id" : "obj-18",
```

```
                "patching_rect" : [ 975.0, 735.0, 44.0, 47.0 ],
                "activedialcolor" : [ 0.031373, 0.282353, 0.584314, 1.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "parameter_enable" : 1,
                "numoutlets" : 2,
                "outlettype" : [ "", "float" ],
                "presentation_rect" : [ 1185.0, 555.0, 44.0, 47.0 ],
                "saved_attribute_attributes" :                 {
                   "valueof" :                    {
                      "parameter_mmin" : 2.0,
                      "parameter_type" : 0,
                      "parameter_initial_enable" : 0,
                      "parameter_shortname" : "Filtro",
                      "parameter_modmax" : 127.0,
                      "parameter_longname" : "live.dial[20]",
                      "parameter_modmin" : 0.0,
                      "parameter_linknames" : 0,
                      "parameter_modmode" : 0,
                      "parameter_info" : "",
                      "parameter_units" : "",
                      "parameter_order" : 0,
                      "parameter_defer" : 0,
                      "parameter_speedlim" : 1.0,
                      "parameter_steps" : 0,
                      "parameter_invisible" : 0,
                      "parameter_exponent" : 1.0,
                      "parameter_annotation_name" : "",
                      "parameter_unitstyle" : 0,
                      "parameter_mmax" : 127.0
                   }

                }

             }

          }
,         {
             "box" :             {
                "maxclass" : "comment",
                "text" : "Programa",
                "id" : "obj-72",
                "patching_rect" : [ 930.0, 135.0, 70.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 510.0, 15.0, 66.0, 20.0 ]
             }

          }
,         {
             "box" :             {
                "maxclass" : "tab",
                "varname" : "zona",
                "id" : "obj-69",
                "bordercolor" : [ 0.0, 0.0, 0.0, 1.0 ],
```

```
                "patching_rect" : [ 1125.0, 165.0, 105.0, 20.0 ],
                "borderoncolor" : [ 0.0, 0.0, 0.0, 1.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "clicktabcolor" : [ 0.0, 0.0, 0.0, 1.0 ],
                "border" : 2,
                "numoutlets" : 3,
                "tabcolor" : [ 0.909804, 0.905882, 0.905882, 1.0 ],
                "tabs" : [ "1", "2" ],
                "htextcolor" : [ 1.0, 1.0, 1.0, 1.0 ],
                "outlettype" : [ "int", "", "" ],
                "fontname" : "Arial",
                "presentation_rect" : [ 1114.0, 148.0, 105.0, 20.0 ],
                "htabcolor" : [ 0.0, 0.0, 0.0, 1.0 ],
                "hovertabcolor" : [ 0.909804, 0.905882, 0.905882, 1.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "T\na\nm\na\nñ\no",
                "linecount" : 6,
                "presentation_linecount" : 6,
                "id" : "obj-68",
                "patching_rect" : [ 1275.0, 315.0, 22.0, 89.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 1237.0, 248.0, 28.0, 89.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "Rapidez",
                "id" : "obj-64",
                "patching_rect" : [ 1110.0, 270.0, 150.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 1135.0, 360.0, 62.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "Cargar",
                "id" : "obj-56",
                "patching_rect" : [ 1035.0, 45.0, 67.0, 20.0 ],
                "fontsize" : 12.0,
```

```
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 405.0, 45.0, 67.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "Guardar",
                "id" : "obj-53",
                "patching_rect" : [ 1035.0, 15.0, 67.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 405.0, 15.0, 67.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "button",
                "id" : "obj-43",
                "patching_rect" : [ 1110.0, 45.0, 20.0, 20.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 1,
                "outlettype" : [ "bang" ],
                "presentation_rect" : [ 480.0, 45.0, 20.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "button",
                "id" : "obj-42",
                "patching_rect" : [ 1110.0, 15.0, 20.0, 20.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 1,
                "outlettype" : [ "bang" ],
                "presentation_rect" : [ 480.0, 15.0, 20.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "Divisor Zonas",
                "id" : "obj-36",
                "patching_rect" : [ 750.0, 135.0, 83.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
```

```
            "numoutlets" : 0,
            "fontname" : "Arial",
            "presentation_rect" : [ 1095.0, 90.0, 83.0, 20.0 ]
          }

        }
,       {
          "box" :           {
            "maxclass" : "number",
            "varname" : "divisorzonas",
            "id" : "obj-25",
            "maximum" : 84,
            "bordercolor" : [ 0.0, 0.0, 0.0, 1.0 ],
            "patching_rect" : [ 750.0, 165.0, 50.0, 20.0 ],
            "fontsize" : 12.0,
            "presentation" : 1,
            "numinlets" : 1,
            "htricolor" : [ 1.0, 1.0, 1.0, 1.0 ],
            "numoutlets" : 2,
            "outlettype" : [ "int", "bang" ],
            "tricolor" : [ 0.0, 0.0, 0.0, 1.0 ],
            "fontname" : "Arial",
            "minimum" : 48,
            "presentation_rect" : [ 1185.0, 90.0, 50.0, 20.0 ]
          }

        }
,       {
          "box" :           {
            "maxclass" : "comment",
            "text" : "Random",
            "id" : "obj-82",
            "patching_rect" : [ 1275.0, 450.0, 150.0, 20.0 ],
            "fontsize" : 12.0,
            "presentation" : 1,
            "numinlets" : 1,
            "numoutlets" : 0,
            "fontname" : "Arial",
            "presentation_rect" : [ 1125.0, 195.0, 58.0, 20.0 ]
          }

        }
,       {
          "box" :           {
            "maxclass" : "newobj",
            "text" : "- 127",
            "id" : "obj-78",
            "patching_rect" : [ 1279.0, 501.0, 38.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 2,
            "numoutlets" : 1,
            "outlettype" : [ "int" ],
            "fontname" : "Arial"
          }

        }
,       {
          "box" :           {
```

```
                "maxclass" : "newobj",
                "text" : "ctlin 21",
                "id" : "obj-79",
                "patching_rect" : [ 1334.0, 474.0, 49.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 2,
                "outlettype" : [ "int", "int" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "ctlin 20",
                "id" : "obj-80",
                "patching_rect" : [ 1273.0, 474.0, 49.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 2,
                "outlettype" : [ "int", "int" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "Enlace Zonas",
                "id" : "obj-71",
                "patching_rect" : [ 538.0, 146.0, 150.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 1095.0, 45.0, 82.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "/ 127.",
                "id" : "obj-59",
                "patching_rect" : [ 1015.0, 788.0, 41.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "float" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
```

```
                "text" : "ctlin 1",
                "id" : "obj-58",
                "patching_rect" : [ 906.0, 719.0, 42.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 2,
                "outlettype" : [ "int", "int" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "* 10.",
                "id" : "obj-55",
                "patching_rect" : [ 285.0, 195.0, 35.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "float" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "mtof",
                "id" : "obj-54",
                "patching_rect" : [ 285.0, 165.0, 34.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 1,
                "outlettype" : [ "" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "maximum",
                "id" : "obj-51",
                "patching_rect" : [ 285.0, 135.0, 63.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 2,
                "outlettype" : [ "int", "int" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "trigger",
                "id" : "obj-50",
```

```
            "patching_rect" : [ 1214.0, 519.0, 45.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 1,
            "numoutlets" : 2,
            "outlettype" : [ "", "" ],
            "fontname" : "Arial"
          }

        }
,       {
          "box" :           {
            "maxclass" : "flonum",
            "id" : "obj-49",
            "patching_rect" : [ 1169.0, 844.0, 50.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 1,
            "numoutlets" : 2,
            "outlettype" : [ "float", "bang" ],
            "fontname" : "Arial"
          }

        }
,       {
          "box" :           {
            "maxclass" : "flonum",
            "id" : "obj-48",
            "patching_rect" : [ 1104.0, 849.0, 50.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 1,
            "numoutlets" : 2,
            "outlettype" : [ "float", "bang" ],
            "fontname" : "Arial"
          }

        }
,       {
          "box" :           {
            "maxclass" : "flonum",
            "id" : "obj-47",
            "patching_rect" : [ 1165.0, 804.0, 50.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 1,
            "numoutlets" : 2,
            "outlettype" : [ "float", "bang" ],
            "fontname" : "Arial"
          }

        }
,       {
          "box" :           {
            "maxclass" : "flonum",
            "id" : "obj-46",
            "patching_rect" : [ 1110.0, 806.0, 50.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 1,
            "numoutlets" : 2,
            "outlettype" : [ "float", "bang" ],
            "fontname" : "Arial"
```

```
            }

        }
,       {
            "box" :             {
                "maxclass" : "preset",
                "id" : "obj-15",
                "patching_rect" : [ 930.0, 165.0, 102.0, 17.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 4,
                "outlettype" : [ "preset", "int", "preset", "int" ],
                "presentation_rect" : [ 600.0, 15.0, 104.0, 17.0 ],
                "pattrstorage" : "sinte"
            }

        }
,       {
            "box" :             {
                "maxclass" : "comment",
                "text" : "Enlace Notas",
                "id" : "obj-33",
                "patching_rect" : [ 373.0, 147.0, 150.0, 20.0 ],
                "fontsize" : 12.0,
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "fontname" : "Arial",
                "presentation_rect" : [ 1095.0, 15.0, 81.0, 20.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "/ 20.",
                "id" : "obj-38",
                "patching_rect" : [ 1230.0, 600.0, 34.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "float" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "/ 19.5",
                "id" : "obj-37",
                "patching_rect" : [ 1185.0, 600.0, 41.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "float" ],
                "fontname" : "Arial"
            }
```

```
        }
,       {
          "box" :               {
            "maxclass" : "newobj",
            "text" : "/ 19.",
            "id" : "obj-35",
            "patching_rect" : [ 1140.0, 600.0, 34.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 2,
            "numoutlets" : 1,
            "outlettype" : [ "float" ],
            "fontname" : "Arial"
          }

        }
,       {
          "box" :               {
            "maxclass" : "gswitch",
            "id" : "obj-31",
            "patching_rect" : [ 1220.0, 652.0, 41.0, 32.0 ],
            "numinlets" : 3,
            "numoutlets" : 1,
            "outlettype" : [ "" ]
          }

        }
,       {
          "box" :               {
            "maxclass" : "gswitch",
            "id" : "obj-30",
            "patching_rect" : [ 1171.0, 654.0, 41.0, 32.0 ],
            "numinlets" : 3,
            "numoutlets" : 1,
            "outlettype" : [ "" ]
          }

        }
,       {
          "box" :               {
            "maxclass" : "gswitch",
            "id" : "obj-27",
            "patching_rect" : [ 1126.0, 654.0, 41.0, 32.0 ],
            "numinlets" : 3,
            "numoutlets" : 1,
            "outlettype" : [ "" ]
          }

        }
,       {
          "box" :               {
            "maxclass" : "message",
            "text" : "0",
            "id" : "obj-22",
            "patching_rect" : [ 1127.0, 481.0, 32.5, 18.0 ],
            "fontsize" : 12.0,
            "numinlets" : 2,
            "numoutlets" : 1,
```

```
                "outlettype" : [ "" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "gswitch",
                "id" : "obj-41",
                "patching_rect" : [ 1081.0, 654.0, 41.0, 32.0 ],
                "numinlets" : 3,
                "numoutlets" : 1,
                "outlettype" : [ "" ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "led",
                "varname" : "led_rand",
                "id" : "obj-40",
                "patching_rect" : [ 1080.0, 480.0, 20.0, 20.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 1,
                "oncolor" : [ 0.168627, 0.47451, 0.666667, 1.0 ],
                "outlettype" : [ "int" ],
                "offcolor" : [ 1.0, 1.0, 1.0, 1.0 ],
                "presentation_rect" : [ 1181.0, 195.0, 20.0, 20.0 ],
                "bgcolor" : [ 0.905882, 0.909804, 0.909804, 1.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "pictslider",
                "varname" : "panel_random",
                "id" : "obj-34",
                "leftvalue" : 500,
                "patching_rect" : [ 1110.0, 285.0, 136.0, 130.0 ],
                "presentation" : 1,
                "numinlets" : 2,
                "topvalue" : 95,
                "numoutlets" : 2,
                "invisiblebkgnd" : 1,
                "outlettype" : [ "int", "int" ],
                "bottomvalue" : 25,
                "presentation_rect" : [ 1097.0, 228.0, 132.0, 124.0 ],
                "rightvalue" : 60
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "/ 20.5",
                "id" : "obj-32",
                "patching_rect" : [ 1095.0, 600.0, 41.0, 20.0 ],
```

```
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "float" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "metro",
                "id" : "obj-26",
                "patching_rect" : [ 1107.0, 424.0, 41.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "bang" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "random",
                "id" : "obj-28",
                "patching_rect" : [ 1108.0, 450.0, 51.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "int" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "filtergraph~",
                "id" : "obj-24",
                "patching_rect" : [ 575.0, 810.0, 256.0, 128.0 ],
                "numinlets" : 8,
                "numoutlets" : 7,
                "outlettype" : [ "list", "float", "float", "float", "float", "list", "int" ],
                "nfilters" : 1,
                "setfilter" : [ 0, 7, 1, 1, 0, 654.063904, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "biquad~",
                "id" : "obj-16",
                "patching_rect" : [ 450.0, 795.0, 86.5, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 6,
                "numoutlets" : 1,
```

```
                "outlettype" : [ "signal" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "message",
                "text" : "read",
                "id" : "obj-23",
                "patching_rect" : [ 990.0, 45.0, 35.0, 18.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "message",
                "text" : "write",
                "id" : "obj-21",
                "patching_rect" : [ 930.0, 45.0, 36.0, 18.0 ],
                "fontsize" : 12.0,
                "numinlets" : 2,
                "numoutlets" : 1,
                "outlettype" : [ "" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "varname" : "u598001384",
                "text" : "autopattr",
                "id" : "obj-19",
                "patching_rect" : [ 930.0, 105.0, 59.5, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 4,
                "outlettype" : [ "", "", "", "" ],
                "fontname" : "Arial",
                "restore" :                 {
                    "divisorzonas" : [ 60 ],
                    "led_link_notas" : [ 0 ],
                    "led_link_zonas" : [ 0 ],
                    "led_rand" : [ 0 ],
                    "live.dial" : [ 127.0 ],
                    "live.gain~" : [ 0.0 ],
                    "live.gain~[1]" : [ 0.0 ],
                    "live.gain~[2]" : [ 0.0 ],
                    "panel_random" : [ 500, 25 ],
                    "zona" : [ 0 ]
                }
```

```
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "varname" : "sinte",
                "text" : "pattrstorage sinte @greedy 1",
                "id" : "obj-11",
                "patching_rect" : [ 930.0, 75.0, 165.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 1,
                "outlettype" : [ "" ],
                "fontname" : "Arial",
                "saved_object_attributes" :                 {
                    "parameter_enable" : 0,
                    "storage_rect" : [ 583, 69, 1034, 197 ],
                    "client_rect" : [ 4, 44, 358, 172 ],
                    "paraminitmode" : 0
                }

            }

        }
,       {
            "box" :             {
                "maxclass" : "live.gain~",
                "varname" : "live.gain~[2]",
                "id" : "obj-20",
                "patching_rect" : [ 300.0, 765.0, 48.0, 136.0 ],
                "presentation" : 1,
                "numinlets" : 2,
                "parameter_enable" : 1,
                "numoutlets" : 5,
                "outlettype" : [ "signal", "signal", "", "float", "list" ],
                "presentation_rect" : [ 1095.0, 555.0, 48.0, 136.0 ],
                "saved_attribute_attributes" :                 {
                    "valueof" :                     {
                        "parameter_mmin" : -70.0,
                        "parameter_initial" : [ 0.0 ],
                        "parameter_type" : 0,
                        "parameter_initial_enable" : 0,
                        "parameter_shortname" : "Volumen",
                        "parameter_modmax" : 127.0,
                        "parameter_longname" : "live.gain~[9]",
                        "parameter_modmin" : 0.0,
                        "parameter_linknames" : 0,
                        "parameter_modmode" : 0,
                        "parameter_info" : "",
                        "parameter_units" : "",
                        "parameter_order" : 0,
                        "parameter_defer" : 0,
                        "parameter_speedlim" : 1.0,
                        "parameter_steps" : 0,
                        "parameter_invisible" : 0,
                        "parameter_exponent" : 1.0,
                        "parameter_annotation_name" : "",
```

```
                    "parameter_unitstyle" : 4,
                    "parameter_mmax" : 6.0
                }

            }

        }

    }
,   {
        "box" :             {
            "maxclass" : "kslider",
            "id" : "obj-17",
            "patching_rect" : [ 255.0, 45.0, 348.0, 53.0 ],
            "presentation" : 1,
            "numinlets" : 2,
            "numoutlets" : 2,
            "outlettype" : [ "int", "int" ],
            "presentation_rect" : [ 720.0, 15.0, 348.0, 53.0 ],
            "range" : 49
        }

    }
,   {
        "box" :             {
            "maxclass" : "live.gain~",
            "varname" : "live.gain~[1]",
            "id" : "obj-14",
            "patching_rect" : [ 150.0, 750.0, 48.0, 136.0 ],
            "presentation" : 1,
            "numinlets" : 2,
            "parameter_enable" : 1,
            "numoutlets" : 5,
            "outlettype" : [ "signal", "signal", "", "float", "list" ],
            "presentation_rect" : [ 1185.0, 390.0, 48.0, 136.0 ],
            "saved_attribute_attributes" :                 {
                "valueof" :                     {
                    "parameter_mmin" : -70.0,
                    "parameter_initial" : [ 0.0 ],
                    "parameter_type" : 0,
                    "parameter_initial_enable" : 0,
                    "parameter_shortname" : "Zona 2",
                    "parameter_modmax" : 127.0,
                    "parameter_longname" : "live.gain~[8]",
                    "parameter_modmin" : 0.0,
                    "parameter_linknames" : 0,
                    "parameter_modmode" : 0,
                    "parameter_info" : "",
                    "parameter_units" : "",
                    "parameter_order" : 0,
                    "parameter_defer" : 0,
                    "parameter_speedlim" : 1.0,
                    "parameter_steps" : 0,
                    "parameter_invisible" : 0,
                    "parameter_exponent" : 1.0,
                    "parameter_annotation_name" : "",
                    "parameter_unitstyle" : 4,
                    "parameter_mmax" : 6.0
```

```
                }

            }

        }

    }
,   {
        "box" :         {
            "maxclass" : "newobj",
            "text" : "sel 0 1",
            "id" : "obj-13",
            "patching_rect" : [ 1125.0, 195.0, 46.0, 20.0 ],
            "fontsize" : 12.0,
            "numinlets" : 1,
            "numoutlets" : 3,
            "outlettype" : [ "bang", "bang", "" ],
            "fontname" : "Arial"
        }

    }
,   {
        "box" :         {
            "maxclass" : "message",
            "text" : "offset 0 -510",
            "id" : "obj-2",
            "patching_rect" : [ 1170.0, 240.0, 75.0, 18.0 ],
            "fontsize" : 11.595187,
            "numinlets" : 2,
            "numoutlets" : 1,
            "outlettype" : [ "" ],
            "fontname" : "Arial"
        }

    }
,   {
        "box" :         {
            "maxclass" : "bpatcher",
            "varname" : "OSC267",
            "id" : "obj-8",
            "name" : "Osciladores.maxpat",
            "patching_rect" : [ -1.0, 236.0, 1078.0, 454.0 ],
            "presentation" : 1,
            "numinlets" : 10,
            "numoutlets" : 8,
            "args" : [  ],
            "outlettype" : [ "signal", "signal", "signal", "signal", "signal", "signal", "signal", "signal"
],
            "presentation_rect" : [ 0.0, 75.0, 1078.0, 454.0 ]
        }

    }
,   {
        "box" :         {
            "maxclass" : "message",
            "text" : "offset 0 0",
            "id" : "obj-10",
            "patching_rect" : [ 1095.0, 240.0, 58.0, 18.0 ],
```

```
            "fontsize" : 11.595187,
            "numinlets" : 2,
            "numoutlets" : 1,
            "outlettype" : [ "" ],
            "fontname" : "Arial"
        }

    }
,   {
        "box" :         {
            "maxclass" : "ezdac~",
            "id" : "obj-7",
            "patching_rect" : [ 450.0, 840.0, 45.0, 45.0 ],
            "presentation" : 1,
            "numinlets" : 2,
            "numoutlets" : 0,
            "presentation_rect" : [ 1185.0, 615.0, 45.0, 45.0 ]
        }

    }
,   {
        "box" :         {
            "maxclass" : "live.gain~",
            "varname" : "live.gain~",
            "id" : "obj-6",
            "patching_rect" : [ 45.0, 750.0, 48.0, 136.0 ],
            "presentation" : 1,
            "numinlets" : 2,
            "parameter_enable" : 1,
            "numoutlets" : 5,
            "outlettype" : [ "signal", "signal", "", "float", "list" ],
            "presentation_rect" : [ 1095.0, 390.0, 48.0, 136.0 ],
            "saved_attribute_attributes" :          {
                "valueof" :             {
                    "parameter_mmin" : -70.0,
                    "parameter_initial" : [ 0.0 ],
                    "parameter_type" : 0,
                    "parameter_initial_enable" : 0,
                    "parameter_shortname" : "Zona 1",
                    "parameter_modmax" : 127.0,
                    "parameter_longname" : "live.gain~[7]",
                    "parameter_modmin" : 0.0,
                    "parameter_linknames" : 0,
                    "parameter_modmode" : 0,
                    "parameter_info" : "",
                    "parameter_units" : "",
                    "parameter_order" : 0,
                    "parameter_defer" : 0,
                    "parameter_speedlim" : 1.0,
                    "parameter_steps" : 0,
                    "parameter_invisible" : 0,
                    "parameter_exponent" : 1.0,
                    "parameter_annotation_name" : "",
                    "parameter_unitstyle" : 4,
                    "parameter_mmax" : 6.0
                }

            }
```

```
            }

        }
,       {
            "box" :             {
                "maxclass" : "led",
                "varname" : "led_link_zonas",
                "id" : "obj-4",
                "patching_rect" : [ 540.0, 180.0, 20.0, 20.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 1,
                "oncolor" : [ 0.168627, 0.47451, 0.666667, 1.0 ],
                "outlettype" : [ "int" ],
                "offcolor" : [ 1.0, 1.0, 1.0, 1.0 ],
                "presentation_rect" : [ 1185.0, 45.0, 20.0, 20.0 ],
                "bgcolor" : [ 0.905882, 0.909804, 0.909804, 1.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "led",
                "varname" : "led_link_notas",
                "id" : "obj-3",
                "patching_rect" : [ 375.0, 180.0, 20.0, 20.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 1,
                "oncolor" : [ 0.168627, 0.47451, 0.666667, 1.0 ],
                "outlettype" : [ "int" ],
                "offcolor" : [ 1.0, 1.0, 1.0, 1.0 ],
                "presentation_rect" : [ 1185.0, 15.0, 20.0, 20.0 ],
                "bgcolor" : [ 0.905882, 0.909804, 0.909804, 1.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "newobj",
                "text" : "notein",
                "id" : "obj-1",
                "patching_rect" : [ 405.0, 0.0, 46.0, 20.0 ],
                "fontsize" : 12.0,
                "numinlets" : 1,
                "numoutlets" : 3,
                "outlettype" : [ "int", "int", "int" ],
                "fontname" : "Arial"
            }

        }
,       {
            "box" :             {
                "maxclass" : "bpatcher",
                "id" : "obj-5",
                "name" : "Visor.maxpat",
                "patching_rect" : [ 15.0, 990.0, 1067.0, 192.0 ],
```

```
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "args" : [  ],
                "presentation_rect" : [ 0.0, 525.0, 1063.0, 183.0 ],
                "offset" : [ -46.0, -50.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "panel",
                "id" : "obj-45",
                "patching_rect" : [ 765.0, 0.0, 128.0, 128.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "border" : 3,
                "numoutlets" : 0,
                "rounded" : 26,
                "presentation_rect" : [ 1095.0, 225.0, 136.0, 130.0 ],
                "bgcolor" : [ 0.909804, 0.905882, 0.905882, 1.0 ]
            }

        }
,       {
            "box" :             {
                "maxclass" : "panel",
                "id" : "obj-44",
                "patching_rect" : [ 1230.0, 0.0, 128.0, 128.0 ],
                "presentation" : 1,
                "numinlets" : 1,
                "numoutlets" : 0,
                "presentation_rect" : [ 0.0, 0.0, 1274.0, 727.0 ],
                "bgcolor" : [ 0.905882, 0.909804, 0.909804, 1.0 ]
            }

        }
],
    "lines" : [         {
            "patchline" :           {
                "source" : [ "obj-17", 0 ],
                "destination" : [ "obj-62", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :           {
                "source" : [ "obj-17", 1 ],
                "destination" : [ "obj-62", 1 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :           {
```

```
            "source" : [ "obj-61", 1 ],
            "destination" : [ "obj-83", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-83", 0 ],
            "destination" : [ "obj-73", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-55", 0 ],
            "destination" : [ "obj-24", 5 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-54", 0 ],
            "destination" : [ "obj-55", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-17", 1 ],
            "destination" : [ "obj-26", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-9", 0 ],
            "destination" : [ "obj-73", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-79", 0 ],
            "destination" : [ "obj-40", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
```

```
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-80", 0 ],
                "destination" : [ "obj-78", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-73", 0 ],
                "destination" : [ "obj-70", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-18", 0 ],
                "destination" : [ "obj-59", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-58", 0 ],
                "destination" : [ "obj-18", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-69", 0 ],
                "destination" : [ "obj-13", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-21", 0 ],
                "destination" : [ "obj-11", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
```

```
            "patchline" :               {
                "source" : [ "obj-23", 0 ],
                "destination" : [ "obj-11", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-37", 0 ],
                "destination" : [ "obj-30", 2 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-28", 0 ],
                "destination" : [ "obj-37", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-35", 0 ],
                "destination" : [ "obj-27", 2 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-32", 0 ],
                "destination" : [ "obj-41", 2 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-28", 0 ],
                "destination" : [ "obj-32", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-28", 0 ],
                "destination" : [ "obj-35", 0 ],
                "hidden" : 0,
```

```
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :           {
            "source" : [ "obj-22", 0 ],
            "destination" : [ "obj-31", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :           {
            "source" : [ "obj-22", 0 ],
            "destination" : [ "obj-30", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :           {
            "source" : [ "obj-22", 0 ],
            "destination" : [ "obj-27", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :           {
            "source" : [ "obj-13", 1 ],
            "destination" : [ "obj-2", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :           {
            "source" : [ "obj-13", 0 ],
            "destination" : [ "obj-10", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :           {
            "source" : [ "obj-24", 0 ],
            "destination" : [ "obj-16", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
```

```
,          {
             "patchline" :              {
                "source" : [ "obj-20", 0 ],
                "destination" : [ "obj-16", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
             }

          }
,          {
             "patchline" :              {
                "source" : [ "obj-6", 0 ],
                "destination" : [ "obj-20", 1 ],
                "hidden" : 0,
                "midpoints" : [  ]
             }

          }
,          {
             "patchline" :              {
                "source" : [ "obj-6", 0 ],
                "destination" : [ "obj-20", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
             }

          }
,          {
             "patchline" :              {
                "source" : [ "obj-14", 0 ],
                "destination" : [ "obj-20", 1 ],
                "hidden" : 0,
                "midpoints" : [  ]
             }

          }
,          {
             "patchline" :              {
                "source" : [ "obj-14", 0 ],
                "destination" : [ "obj-20", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
             }

          }
,          {
             "patchline" :              {
                "source" : [ "obj-16", 0 ],
                "destination" : [ "obj-7", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
             }

          }
,          {
             "patchline" :              {
                "source" : [ "obj-16", 0 ],
                "destination" : [ "obj-7", 1 ],
```

```
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-34", 0 ],
            "destination" : [ "obj-26", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-26", 0 ],
            "destination" : [ "obj-28", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-34", 1 ],
            "destination" : [ "obj-28", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-22", 0 ],
            "destination" : [ "obj-41", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-41", 0 ],
            "destination" : [ "obj-46", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-27", 0 ],
            "destination" : [ "obj-47", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }
```

```
        }
,       {
          "patchline" :       {
             "source" : [ "obj-30", 0 ],
             "destination" : [ "obj-48", 0 ],
             "hidden" : 0,
             "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
             "source" : [ "obj-31", 0 ],
             "destination" : [ "obj-49", 0 ],
             "hidden" : 0,
             "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
             "source" : [ "obj-38", 0 ],
             "destination" : [ "obj-31", 2 ],
             "hidden" : 0,
             "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
             "source" : [ "obj-28", 0 ],
             "destination" : [ "obj-38", 0 ],
             "hidden" : 0,
             "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
             "source" : [ "obj-50", 1 ],
             "destination" : [ "obj-41", 0 ],
             "hidden" : 0,
             "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
             "source" : [ "obj-50", 1 ],
             "destination" : [ "obj-27", 0 ],
             "hidden" : 0,
             "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
             "source" : [ "obj-50", 1 ],
```

```
            "destination" : [ "obj-30", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-50", 1 ],
            "destination" : [ "obj-31", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-50", 0 ],
            "destination" : [ "obj-22", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-40", 0 ],
            "destination" : [ "obj-50", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-1", 0 ],
            "destination" : [ "obj-17", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-1", 1 ],
            "destination" : [ "obj-17", 1 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-17", 0 ],
            "destination" : [ "obj-51", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }
```

```
        }
,       {
          "patchline" :       {
            "source" : [ "obj-51", 0 ],
            "destination" : [ "obj-54", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-78", 0 ],
            "destination" : [ "obj-40", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-42", 0 ],
            "destination" : [ "obj-21", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-43", 0 ],
            "destination" : [ "obj-23", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-16", 0 ],
            "destination" : [ "obj-5", 0 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
            "source" : [ "obj-59", 0 ],
            "destination" : [ "obj-24", 6 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :       {
```

```
            "source" : [ "obj-25", 0 ],
            "destination" : [ "obj-8", 4 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-10", 0 ],
            "destination" : [ "obj-8", 5 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-2", 0 ],
            "destination" : [ "obj-8", 5 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-41", 0 ],
            "destination" : [ "obj-8", 6 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-27", 0 ],
            "destination" : [ "obj-8", 7 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-30", 0 ],
            "destination" : [ "obj-8", 8 ],
            "hidden" : 0,
            "midpoints" : [  ]
          }

        }
,       {
          "patchline" :         {
            "source" : [ "obj-31", 0 ],
            "destination" : [ "obj-8", 9 ],
            "hidden" : 0,
            "midpoints" : [  ]
```

```
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-8", 0 ],
                "destination" : [ "obj-6", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-8", 1 ],
                "destination" : [ "obj-6", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-8", 2 ],
                "destination" : [ "obj-6", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-8", 3 ],
                "destination" : [ "obj-6", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-8", 4 ],
                "destination" : [ "obj-14", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-8", 5 ],
                "destination" : [ "obj-14", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
```

```
            "patchline" :               {
                "source" : [ "obj-8", 6 ],
                "destination" : [ "obj-14", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-8", 7 ],
                "destination" : [ "obj-14", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-3", 0 ],
                "destination" : [ "obj-8", 2 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-4", 0 ],
                "destination" : [ "obj-8", 3 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-70", 0 ],
                "destination" : [ "obj-8", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-70", 0 ],
                "destination" : [ "obj-8", 1 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :               {
                "source" : [ "obj-1", 0 ],
                "destination" : [ "obj-8", 0 ],
                "hidden" : 0,
```

```
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :       {
                "source" : [ "obj-1", 1 ],
                "destination" : [ "obj-8", 1 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :       {
                "source" : [ "obj-62", 0 ],
                "destination" : [ "obj-8", 0 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
,       {
            "patchline" :       {
                "source" : [ "obj-62", 1 ],
                "destination" : [ "obj-8", 1 ],
                "hidden" : 0,
                "midpoints" : [  ]
            }

        }
],
    "parameters" :      {
        "obj-8::obj-360" : [ "live.slider[40]", "R", 0 ],
        "obj-8::obj-375" : [ "live.slider[45]", "S", 0 ],
        "obj-6" : [ "live.gain~[7]", "Zona 1", 0 ],
        "obj-8::obj-16" : [ "live.slider[4]", "R", 0 ],
        "obj-8::obj-404" : [ "live.slider[54]", "D", 0 ],
        "obj-8::obj-401" : [ "live.dial[13]", "Frecuencia", 0 ],
        "obj-8::obj-419" : [ "live.slider[59]", "A", 0 ],
        "obj-8::obj-433" : [ "live.slider[63]", "A", 0 ],
        "obj-8::obj-471" : [ "live.dial[18]", "Frecuencia", 0 ],
        "obj-8::obj-350" : [ "live.menu[9]", "live.menu", 0 ],
        "obj-8::obj-26" : [ "live.menu", "live.menu", 0 ],
        "obj-8::obj-458" : [ "live.slider[68]", "R", 0 ],
        "obj-8::obj-264" : [ "live.slider[14]", "D", 0 ],
        "obj-8::obj-275" : [ "live.dial[4]", "Frecuencia", 0 ],
        "obj-14" : [ "live.gain~[8]", "Zona 2", 0 ],
        "obj-8::obj-345" : [ "live.dial[9]", "Frecuencia", 0 ],
        "obj-8::obj-279" : [ "live.slider[19]", "A", 0 ],
        "obj-8::obj-472" : [ "live.slider[72]", "R", 0 ],
        "obj-8::obj-487" : [ "live.slider[77]", "S", 0 ],
        "obj-8::obj-293" : [ "live.slider[23]", "A", 0 ],
        "obj-8::obj-364" : [ "live.menu[10]", "live.menu", 0 ],
        "obj-8::obj-249" : [ "live.slider[9]", "S", 0 ],
        "obj-8::obj-318" : [ "live.slider[28]", "R", 0 ],
        "obj-8::obj-434" : [ "live.menu[15]", "live.menu", 0 ],
        "obj-20" : [ "live.gain~[9]", "Volumen", 0 ],
```

"obj-8::obj-238" : ["live.menu[1]", "live.menu", 0],
"obj-8::obj-332" : ["live.slider[32]", "R", 0],
"obj-8::obj-347" : ["live.slider[37]", "S", 0],
"obj-8::obj-308" : ["live.menu[6]", "live.menu", 0],
"obj-8::obj-397" : ["Gain", "Gain", 0],
"obj-8::obj-361" : ["live.slider[41]", "S", 0],
"obj-8::obj-376" : ["live.slider[46]", "D", 0],
"obj-8::obj-390" : ["live.slider[50]", "D", 0],
"obj-8::obj-405" : ["live.slider[55]", "A", 0],
"obj-8::obj-235" : ["live.slider[5]", "S", 0],
"obj-8::obj-415" : ["live.dial[14]", "Frecuencia", 0],
"obj-8::obj-485" : ["live.dial[19]", "Frecuencia", 0],
"obj-8::obj-444" : ["live.slider[64]", "R", 0],
"obj-8::obj-250" : ["live.slider[10]", "D", 0],
"obj-8::obj-265" : ["live.slider[15]", "A", 0],
"obj-8::obj-459" : ["live.slider[69]", "S", 0],
"obj-8::obj-289" : ["live.dial[5]", "Frecuencia", 0],
"obj-8::obj-473" : ["live.slider[73]", "S", 0],
"obj-8::obj-8" : ["live.slider[1]", "D", 0],
"obj-8::obj-411" : ["live.gain~[3]", "live.gain~[3]", 0],
"obj-8::obj-488" : ["live.slider[78]", "D", 0],
"obj-8::obj-304" : ["live.slider[24]", "R", 0],
"obj-8::obj-378" : ["live.menu[11]", "live.menu", 0],
"obj-8::obj-319" : ["live.slider[29]", "S", 0],
"obj-8::obj-448" : ["live.menu[16]", "live.menu", 0],
"obj-8::obj-333" : ["live.slider[33]", "S", 0],
"obj-8::obj-252" : ["live.menu[2]", "live.menu", 0],
"obj-8::obj-348" : ["live.slider[38]", "D", 0],
"obj-8::obj-362" : ["live.slider[42]", "D", 0],
"obj-8::obj-377" : ["live.slider[47]", "A", 0],
"obj-8::obj-391" : ["live.slider[51]", "A", 0],
"obj-8::obj-359" : ["live.dial[10]", "Frecuencia", 0],
"obj-8::obj-416" : ["live.slider[56]", "R", 0],
"obj-8::obj-5" : ["live.slider", "A", 0],
"obj-8::obj-430" : ["live.slider[60]", "R", 0],
"obj-8::obj-429" : ["live.dial[15]", "Frecuencia", 0],
"obj-8::obj-251" : ["live.slider[11]", "A", 0],
"obj-8::obj-233" : ["live.dial[1]", "Frecuencia", 0],
"obj-8::obj-445" : ["live.slider[65]", "S", 0],
"obj-8::obj-236" : ["live.slider[6]", "D", 0],
"obj-8::obj-303" : ["live.dial[6]", "Frecuencia", 0],
"obj-8::obj-425" : ["live.gain~[1]", "live.gain~[1]", 0],
"obj-8::obj-276" : ["live.slider[16]", "R", 0],
"obj-8::obj-453" : ["live.gain~[4]", "live.gain~[4]", 0],
"obj-8::obj-474" : ["live.slider[74]", "D", 0],
"obj-8::obj-290" : ["live.slider[20]", "R", 0],
"obj-8::obj-305" : ["live.slider[25]", "S", 0],
"obj-8::obj-392" : ["live.menu[12]", "live.menu", 0],
"obj-8::obj-489" : ["live.slider[79]", "A", 0],
"obj-8::obj-462" : ["live.menu[17]", "live.menu", 0],
"obj-8::obj-9" : ["live.slider[2]", "S", 0],
"obj-18" : ["live.dial[20]", "Filtro", 0],
"obj-8::obj-266" : ["live.menu[3]", "live.menu", 0],
"obj-8::obj-334" : ["live.slider[34]", "D", 0],
"obj-8::obj-349" : ["live.slider[39]", "A", 0],
"obj-8::obj-363" : ["live.slider[43]", "A", 0],
"obj-8::obj-388" : ["live.slider[48]", "R", 0],
"obj-8::obj-402" : ["live.slider[52]", "R", 0],

```
            "obj-8::obj-1" : [ "live.dial", "Frecuencia", 0 ],
            "obj-8::obj-373" : [ "live.dial[11]", "Frecuencia", 0 ],
            "obj-8::obj-417" : [ "live.slider[57]", "S", 0 ],
            "obj-8::obj-443" : [ "live.dial[16]", "Frecuencia", 0 ],
            "obj-8::obj-431" : [ "live.slider[61]", "S", 0 ],
            "obj-8::obj-322" : [ "live.menu[7]", "live.menu", 0 ],
            "obj-8::obj-446" : [ "live.slider[66]", "D", 0 ],
            "obj-8::obj-247" : [ "live.dial[2]", "Frecuencia", 0 ],
            "obj-8::obj-262" : [ "live.slider[12]", "R", 0 ],
            "obj-8::obj-317" : [ "live.dial[7]", "Frecuencia", 0 ],
            "obj-8::obj-439" : [ "live.gain~[2]", "live.gain~[2]", 0 ],
            "obj-8::obj-460" : [ "live.slider[70]", "D", 0 ],
            "obj-8::obj-277" : [ "live.slider[17]", "S", 0 ],
            "obj-8::obj-291" : [ "live.slider[21]", "S", 0 ],
            "obj-8::obj-475" : [ "live.slider[75]", "A", 0 ],
            "obj-8::obj-467" : [ "live.gain~[5]", "live.gain~[5]", 0 ],
            "obj-8::obj-406" : [ "live.menu[13]", "live.menu", 0 ],
            "obj-8::obj-306" : [ "live.slider[26]", "D", 0 ],
            "obj-8::obj-237" : [ "live.slider[7]", "A", 0 ],
            "obj-8::obj-476" : [ "live.menu[18]", "live.menu", 0 ],
            "obj-8::obj-320" : [ "live.slider[30]", "D", 0 ],
            "obj-8::obj-383" : [ "live.gain~", "live.gain~", 0 ],
            "obj-8::obj-280" : [ "live.menu[4]", "live.menu", 0 ],
            "obj-8::obj-335" : [ "live.slider[35]", "A", 0 ],
            "obj-8::obj-234" : [ "live.slider[3]", "R", 0 ],
            "obj-8::obj-374" : [ "live.slider[44]", "R", 0 ],
            "obj-8::obj-389" : [ "live.slider[49]", "S", 0 ],
            "obj-8::obj-403" : [ "live.slider[53]", "S", 0 ],
            "obj-8::obj-418" : [ "live.slider[58]", "D", 0 ],
            "obj-8::obj-387" : [ "live.dial[12]", "Frecuencia", 0 ],
            "obj-8::obj-432" : [ "live.slider[62]", "D", 0 ],
            "obj-8::obj-336" : [ "live.menu[8]", "live.menu", 0 ],
            "obj-8::obj-457" : [ "live.dial[17]", "Frecuencia", 0 ],
            "obj-8::obj-263" : [ "live.slider[13]", "S", 0 ],
            "obj-8::obj-447" : [ "live.slider[67]", "A", 0 ],
            "obj-8::obj-261" : [ "live.dial[3]", "Frecuencia", 0 ],
            "obj-8::obj-331" : [ "live.dial[8]", "Frecuencia", 0 ],
            "obj-8::obj-461" : [ "live.slider[71]", "A", 0 ],
            "obj-8::obj-278" : [ "live.slider[18]", "D", 0 ],
            "obj-8::obj-481" : [ "live.gain~[6]", "live.gain~[6]", 0 ],
            "obj-8::obj-292" : [ "live.slider[22]", "D", 0 ],
            "obj-8::obj-486" : [ "live.slider[76]", "R", 0 ],
            "obj-8::obj-248" : [ "live.slider[8]", "R", 0 ],
            "obj-8::obj-420" : [ "live.menu[14]", "live.menu", 0 ],
            "obj-8::obj-307" : [ "live.slider[27]", "A", 0 ],
            "obj-8::obj-321" : [ "live.slider[31]", "A", 0 ],
            "obj-8::obj-490" : [ "live.menu[19]", "live.menu", 0 ],
            "obj-8::obj-346" : [ "live.slider[36]", "R", 0 ],
            "obj-8::obj-294" : [ "live.menu[5]", "live.menu", 0 ]
        }

    }

}
```