



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y SISTEMAS DE TELECOMUNICACIÓN

PROYECTO FIN DE GRADO

TEL
E
L
E
C
T
R
O
M
U
N
I
C
A
C
I
ÓN

Campus Sur
POLITÉCNICA

TÍTULO: Diseño e implementación de un instrumento virtual de Harpejji mediante síntesis por modelado físico

AUTOR: Alberto Barrera Herrero

TITULACIÓN: Grado en Ingeniería de Sonido e Imagen

TUTOR: Lino García Morales

DEPARTAMENTO: Ingeniería Audiovisual y Comunicaciones

VºBº

Miembros del Tribunal Calificador:

PRESIDENTE: Ignacio Antón Hernández

TUTOR: Lino García Morales

SECRETARIO: Antonio Mínguez Olivares

Fecha de lectura:

Calificación:

El Secretario,

Resumen

Diseño e implementación de un instrumento virtual de Harpejji mediante síntesis por modelado físico

El Harpejji es un instrumento de relativamente nueva creación que combina una construcción similar a la de una guitarra con una ejecución más parecida a la de un piano. En este proyecto se desarrolla un sintetizador de Harpejji polifónico utilizando síntesis por modelado físico para modelar la vibración de las cuerdas del instrumento, que es un tipo de instrumento que no existe todavía en el mercado.

El resultado es un plugin de instrumento virtual en formato VST3 compatible con la mayoría de las estaciones de audio digital (DAW) actuales, tanto en sistemas Windows como MacOS. El plugin recibe mensajes MIDI del DAW huésped y reproduce a su salida el sonido producido por la combinación de la vibración de todas las cuerdas virtuales del instrumento. Para su desarrollo se ha empleado la biblioteca de código libre JUCE y como lenguaje de programación, C++.

Se ha utilizado como método de modelado físico para modelar la vibración de las cuerdas del instrumento la aproximación de la ecuación de onda mediante diferencias finitas. No es el método más eficiente, pero es relativamente sencillo de implementar y se ha conseguido un buen rendimiento con un número de voces suficiente para la mayoría de las aplicaciones.

Se considera que los resultados del proyecto son satisfactorios, aunque existe margen de mejora. Se han cumplido todos los objetivos y restricciones de diseño establecidos y se plantean algunas propuestas de mejora que harían del instrumento una herramienta realmente útil para su uso en ámbitos de producción musical.

Abstract

Design and implementation of an Harpejji virtual instrument using physical modeling synthesis

The Harpejji is a relatively newly created instrument that combines a guitar-like construction with a more piano-like execution. In this project, a polyphonic Harpejji synthesizer is developed using physical modeling synthesis to model the vibration of the instrument's strings, which is a kind of instrument that does not yet exist on the market.

The result of the project is a virtual instrument plugin in VST3 format compatible with most current digital audio workstations (DAW), both on Windows and MacOS systems. The plugin receives MIDI messages from the host DAW and plays at its output the sound produced by the combination of the vibration of all the virtual strings of the instrument. For its development, the JUCE free code library and C++ programming language has been used.

The finite difference approximation of the wave equation has been used as the method for the physical modeling of the vibration of the instrument strings. It is not the most efficient method, but it is relatively simple to implement, and good performance has been achieved with enough voices for most applications.

The results of the project are considered satisfactory, although there is room for improvement. All the established objectives and design restrictions have been met and some improvements are proposed that would make the instrument a very useful tool for music production.

Contenido

Lista de acrónimos	7
Definiciones	8
1. Introducción.....	11
1.1. El Harpejji	12
1.2. Métodos de síntesis musical.....	15
1.2.1. Síntesis aditiva	15
1.2.2. Síntesis subtractiva.....	15
1.2.3. Síntesis por modulación de frecuencia	16
1.2.4. Síntesis mediante tabla de ondas	17
1.2.5. Síntesis granular.....	18
1.2.6. Síntesis mediante modelado físico.....	18
1.3. Métodos de síntesis por modelado físico	19
1.3.1. Guía de ondas.....	19
1.3.2. Aproximación de la ecuación diferencial	19
1.4. Ventajas y desventajas de la síntesis por modelado físico	20
1.5. Especificaciones de diseño.....	21
2. Vibraciones en cuerdas.....	23
2.1. La cuerda ideal	24
2.2. Condiciones de contorno.....	25
2.3. Soluciones armónicas de la ecuación de onda	26
2.4. Pérdidas	27
2.5. Rigidz	28
3. Desarrollo del plugin	31
3.1. Virtual Studio Technology (VST).....	32
3.2. Librería JUCE	32
3.3. Funcionamiento general del plugin	33
3.4. Aproximación de la ecuación de onda mediante diferencias finitas	36
3.4.1. Término de la segunda derivada	36
3.4.2. Término de la cuerda ideal.....	37
3.4.3. Término de atenuación lineal.....	37
3.4.4. Término de la atenuación dependiente de la frecuencia.....	38

3.4.5.	Ecuación de onda completa.....	39
3.4.6.	Término de la rigidez	42
3.5.	Excitación de la cuerda.....	44
3.5.1.	Velocidad inicial constante en toda la cuerda	46
3.5.2.	Parábola.....	47
3.5.3.	Rampa.....	47
3.5.4.	Coseno alzado desplazado.....	49
3.5.5.	Función trayectoria de proyectil con rozamiento.....	51
3.6.	Corrección de la amplitud.....	53
3.7.	Pérdidas	54
3.8.	Detector de nivel.....	55
3.9.	Gestión de la entrada MIDI	56
3.10.	Control de tono	57
3.11.	Control de ganancia	58
3.12.	Interfaz gráfica de usuario	58
4.	Resultados	63
4.1.	Validación del modelo.....	64
4.2.	Funcionamiento del plugin.....	66
4.3.	Evaluación de los objetivos	66
5.	Presupuesto	69
5.1.	Mano de obra	70
5.2.	Material.....	70
5.3.	Total.....	70
6.	Conclusiones	71
6.1.	Dificultades.....	72
6.2.	Trabajo futuro	73
6.3.	Aportaciones	74
7.	Referencias.....	75
8.	Bibliografía	77
	Anexo: Guía de instalación	78

Lista de acrónimos

- CPU** Unidad central de procesamiento; *Central Processing Unit*.
- DAW** Estación de trabajo de audio digital; *Digital Audio Workstation*.
- DSP** Procesado de señales digitales; *Digital Signal Processing*.
- IIR** Respuesta infinita al impulso; *Infinite Impulse Response*.
- MIDI** Interfaz digital de instrumentos musicales; *Musical Instrument Digital Interface*.
- VST** Tecnología de Estudio Virtual; *Virtual Studio Technology*.
- VST3** VST Versión 3.
- VSTfx** Efecto de audio VST.
- VSTi** Instrumento VST.

Definiciones

- α Parámetro del detector de nivel. Inverso del número de muestras que utiliza.
- μ Densidad lineal de la cuerda.
- ρ Densidad volumétrica.
- σ_0 Constante de atenuación lineal.
- σ_1 Constante de atenuación dependiente de la frecuencia.
- ω Frecuencia angular de vibración.
- ω_n Frecuencia angular de vibración del modo propio número n .
- a Radio de la cuerda.
- b_n Amplitud de la vibración del modo propio número n .
- c Velocidad de propagación de la vibración en la cuerda.
- c_n^2 Valor del detector de nivel RMS.
- d Distancia entre la cejuela del instrumento y el traste número n .
- E Módulo de Young o módulo de elasticidad longitudinal de la cuerda.
- f_s Frecuencia de muestreo.
- k Número de onda.
- K Radio de giro.
- L Longitud de la cuerda.
- R Coeficiente de reflexión.
- s Longitud de escala. Distancia entre la cejuela y el puente del instrumento.
- S Área de la sección transversal de la cuerda.
- T Tensión en la cuerda.
- T_{60} Tiempo de decaimiento. Tiempo que tarda el nivel de sonido en caer 60 dB.
- v_0 Velocidad inicial en la cuerda.

X Longitud de la cuerda en número de puntos.

Z Impedancia en el extremo de la cuerda.

Z_0 Impedancia característica del medio.

1. Introducción

Se realiza en este capítulo una introducción al Harpejji, que incluye una breve historia del instrumento y la revisión en detalle de sus características. Se explican también los diferentes métodos de síntesis y más detalladamente la síntesis por modelado físico, así como sus ventajas y desventajas y por qué se ha elegido. Finalmente, se incluyen las restricciones de diseño u objetivos que se establecieron para el proyecto en la fase de anteproyecto.

1.1. El Harpejji

El Harpejji es un instrumento musical de cuerda inventado por Tim Meeks en el año 2007 y fabricado por Marcodi Musical Products, según [1]. Combina características de la guitarra con una ejecución y distribución de las notas similar a la del piano. Tiene una construcción en distintos tipos de madera (abedul, arce o bambú), formando una superficie sobre la que se colocan los trastes. La ejecución se realiza pulsando la cuerda con los dedos, como si se pulsaran las teclas de un piano. El método de interpretación del Harpejji se observa en la Figura 1.



Figura 1. Intérprete tocando el Harpejji.

Los trastes se distribuyen, como los de una guitarra, en semitonos y entre una cuerda y la siguiente hay una diferencia de un tono. Esta distribución genera un teclado isomórfico, que permite que tanto los acordes como las escalas que se ejecuten en una determinada tonalidad tengan la misma disposición al transponerse a otra.

En el momento del desarrollo del proyecto existen tres modelos, vendidos en la página del fabricante [2], de tres, cuatro y cinco octavas, con 12, 16 y 24 cuerdas respectivamente. La vibración de las cuerdas se recoge mediante una pastilla piezoeléctrica situada bajo la selleta de cada cuerda. Cuando las cuerdas no se están pulsando se silencian, empleando un sistema magnético que evita las "vibraciones simpáticas" al pulsar otras cuerdas.

Se basa en otros instrumentos similares como el *Chapman Stick* o el *StarrBoard*, cuya ejecución también se realiza pulsando las cuerdas sobre los trastes. Algunos de los intérpretes más famosos de este instrumento son Stevie Wonder, Jacob Collier, Justin-Lee Schultz o Cory Henry.

En este proyecto se simula el funcionamiento de un modelo *G16* de Harpejji, que tiene 16 cuerdas y las notas distribuidas como en la Figura 2.

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
F#3	G#3	A#3	C4	D4	E4	F#4	G#4	A#4	C5	D5	E5	F#5	G#5	A#5	C6
F3	G3	A3	B3	C#4	D#4	F4	G4	A4	B4	C#5	D#5	F5	G5	A5	B5
E3	F#3	G#3	A#3	C4	D4	E4	F#4	G#4	A#4	C5	D5	E5	F#5	G#5	A#5
D#3	F3	G3	A3	B3	C#4	D#4	F4	G4	A4	B4	C#5	D#5	F5	G5	A5
D3	E3	F#3	G#3	A#3	C4	D4	E4	F#4	G#4	A#4	C5	D5	E5	F#5	G#5
C#3	D#3	F3	G3	A3	B3	C#4	D#4	F4	G4	A4	B4	C#5	D#5	F5	G5
C3	D3	E3	F#3	G#3	A#3	C4	D4	E4	F#4	G#4	A#4	C5	D5	E5	F#5
B2	C#3	D#3	F3	G3	A3	B2	C#4	D#4	F4	G4	A4	B4	C#5	D#5	F5
A#2	C3	D3	E3	F#3	G#3	A#2	C4	D4	E4	F#4	G#4	A#4	C5	D5	E5
A2	B2	C#3	D#3	F3	G3	A2	B2	C#4	D#4	F4	G4	A4	B4	C#5	D#5
G#2	A#2	C3	D3	E3	F#3	G#2	A#2	C4	D4	E4	F#4	G#4	A#4	C5	D5
G2	A2	B2	C#3	D#3	F3	G2	A2	B2	C#4	D#4	F4	G4	A4	B4	C#5
F#2	G#2	A#2	C3	D3	E3	F#2	G#2	A#2	C4	D4	E4	F#4	G#4	A#4	C5
F2	G2	A2	B2	C#3	D#3	F3	G3	A3	B3	C#4	D#4	F4	G4	A4	B4
E2	F#2	G#2	A#2	C3	D3	E3	F#3	G#3	A#3	C4	D4	E4	F#4	G#4	A#4
D#2	F2	G2	A2	B2	C#3	D#3	F3	G3	A3	B3	C#4	D#4	F4	G4	A4
D2	E2	F#2	G#2	A#2	C3	D3	E3	F#3	G#3	A#3	C4	D4	E4	F#4	G#4
C#2	D#2	F2	G2	A2	B2	C#3	D#3	F3	G3	A3	B3	C#4	D#4	F4	G4
C2	D2	E2	F#2	G#2	A#2	C3	D3	E3	F#3	G#3	A#3	C4	D4	E4	F#4

Figura 2. Representación esquemática de la distribución de las notas en el Harpejji. En blanco, las posiciones que se corresponden con notas naturales y, en negro, con notas con alteraciones

1. Introducción

Dada la dificultad para obtener información directamente del fabricante ha sido necesario suponer algunas de las características del instrumento, como su longitud de escala, que se ha estimado en 27", o, lo que es lo mismo, 68,58 cm. El resto de los parámetros de cada una de las cuerdas se han medido o calculado a partir de un conjunto de cuerdas de Harpejji G16. Los resultados se han recogido en la Tabla 1.

N. ^º	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Calibre ["]	.64	.56	.48	.40	.34	.30	.26	.22	.20	.18	.16	.14	.12	.10	.09	.08
Diámetro [m]	1,626 E-03	1,422 E-03	1,219 E-03	1,016 E-03	8,636 E-04	7,620 E-04	6,604 E-04	5,588 E-04	5,080 E-04	4,572 E-04	4,064 E-04	3,556 E-04	3,048 E-04	2,540 E-04	2,286 E-04	2,032 E-04
Nota más grave	C2	D2	E2	F#2	G#2	A#2	C3	D3	E3	F#3	G#3	A#3	C4	D4	E4	F#4
Frecuencia min [Hz]	65,4	73,4	82,4	92,5	103,8	116,5	130,8	146,8	164,8	185,0	207,7	233,1	261,6	293,7	329,6	370,0
Densidad lineal [kg/m]	1,30 E-02	1,03 E-02	7,64 E-03	5,26 E-03	3,92 E-03	2,86 E-03	2,24 E-03	1,63 E-03	1,03 E-03	1,12 E-03	9,57 E-04	6,73 E-04	5,00 E-04	4,00 E-04	3,00 E-04	2,60 E-04
Tensión [N]	104,3	104,5	97,6	84,6	79,5	73,0	72,3	66,2	68,5	72,3	77,6	68,8	64,4	64,9	61,3	67,0
Velocidad del sonido [m/s]	89,71	100,70	113,03	126,87	142,41	159,85	179,42	201,39	257,75	253,74	284,82	319,70	358,85	402,79	452,12	507,48

Tabla 1. Parámetros de las cuerdas de un Harpejji G16.

1.2. Métodos de síntesis musical

La síntesis de sonido es la técnica de generar sonido a partir de medios no acústicos, empleando para ello equipos electrónicos o *software*. Según la técnica empleada para generar el sonido se distinguen varios tipos de síntesis que se explican a continuación para justificar el método de síntesis elegido.

1.2.1. Síntesis aditiva

Es el conjunto de técnicas en las que el sonido se genera sumando sinusoides de distinta frecuencia. Como se explica en [3], Para generar un timbre concreto se suman a la frecuencia fundamental una cantidad determinada de sus armónicos. Se aplica a cada armónico una envolvente de amplitud distinta, de forma que el tiempo de decaimiento de cada armónico sea distinto, como ocurre con los instrumentos reales y conseguir un sonido más realista.

Otra estrategia para conseguir mayor realismo consiste en modular lentamente las frecuencias de las sinusoides para variar ligeramente la afinación de la nota sintetizada y simular las imperfecciones de un instrumento o intérprete real. Para generar el sonido de forma analógica es necesario utilizar un banco de sinusoides con frecuencia y envolvente de amplitud configurable. En la Figura 3 se incluye un diagrama de bloques esquemático de un sintetizador aditivo, que se ha tomado de [4].

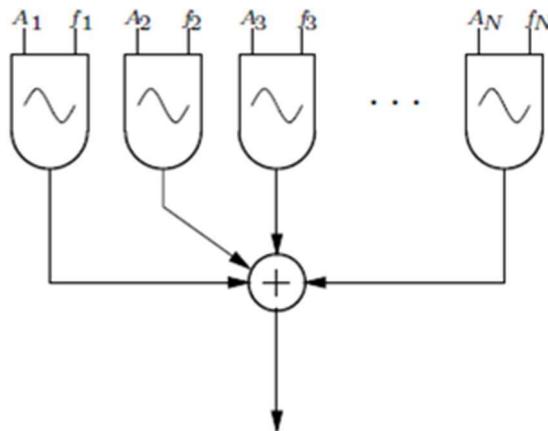


Figura 3. Diagrama de bloques de un sintetizador aditivo.

1.2.2. Síntesis subtractiva

El sonido se genera mediante el filtrado y amplificación de la señal de un oscilador con diferentes formas de onda. La síntesis subtractiva parte de una señal rica en armónicos y

se filtra según convenga. Es la técnica de síntesis más común empleada en sintetizadores *hardware* analógicos.

La señal generada por uno o varios osciladores (normalmente una señal triangular, dientes de sierra o cuadrada) se combina y se filtra [5]. La señal de los osciladores se introduce en un filtro paso bajo, paso alto o paso banda, que elimina cierto contenido armónico de la mezcla original. Finalmente se introduce en un amplificador que controla el volumen del sonido.

Según la implementación que se utilice, los parámetros (frecuencia de corte de los filtros, frecuencia fundamental del oscilador, ganancia del amplificador...) se pueden controlar con generadores de envolvente, que se suelen configurar para iniciarse al pulsar una nota del teclado, o con osciladores de baja frecuencia si se quieren hacer variar estos parámetros lentamente.

En la Figura 4 se observa un posible diagrama de bloques de un sintetizador subtractivo.

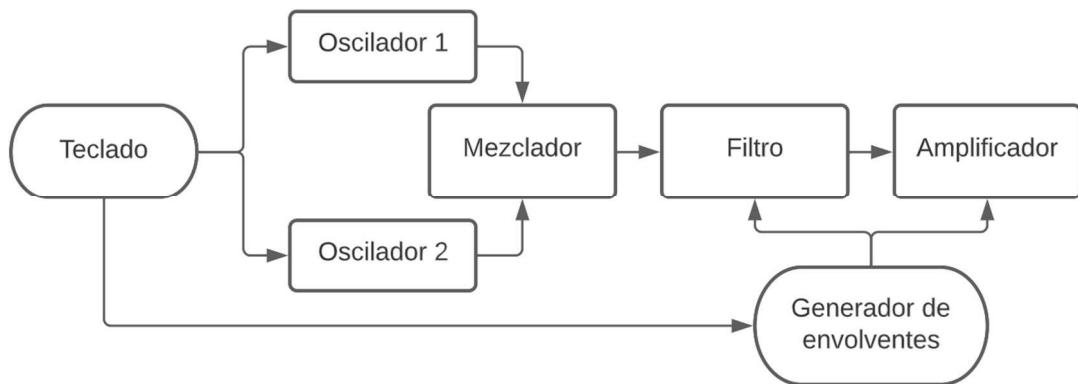


Figura 4. Diagrama de bloques típico de un sintetizador subtractivo.

1.2.3. Síntesis por modulación de frecuencia

El sonido se genera variando la frecuencia de una señal portadora según una segunda señal (moduladora). Mediante esta modulación se generan armónicos tanto por encima como por debajo de la frecuencia de la portadora separados un mismo intervalo [6]. Utilizando de dos a seis osciladores se consiguen generar señales complejas que se asimilan mucho a sonidos de instrumentos reales.

Uno de los sintetizadores más famosos que utiliza este tipo de síntesis es el Yamaha DX7, con un característico sonido de los años ochenta y que dispone de seis osciladores (operadores) que se combinan y modulan entre sí. En la Figura 5 se muestran los algoritmos que utiliza el sintetizador Yamaha DX7, tomada de [7]. En azul se indican los osciladores moduladores, mientras que en verde se representan las portadoras, que producen la señal que se reproduce a la salida del sintetizador [8]. El inconveniente de

este método de síntesis es la falta de correlación entre los parámetros modificables por el usuario y el sonido producido.

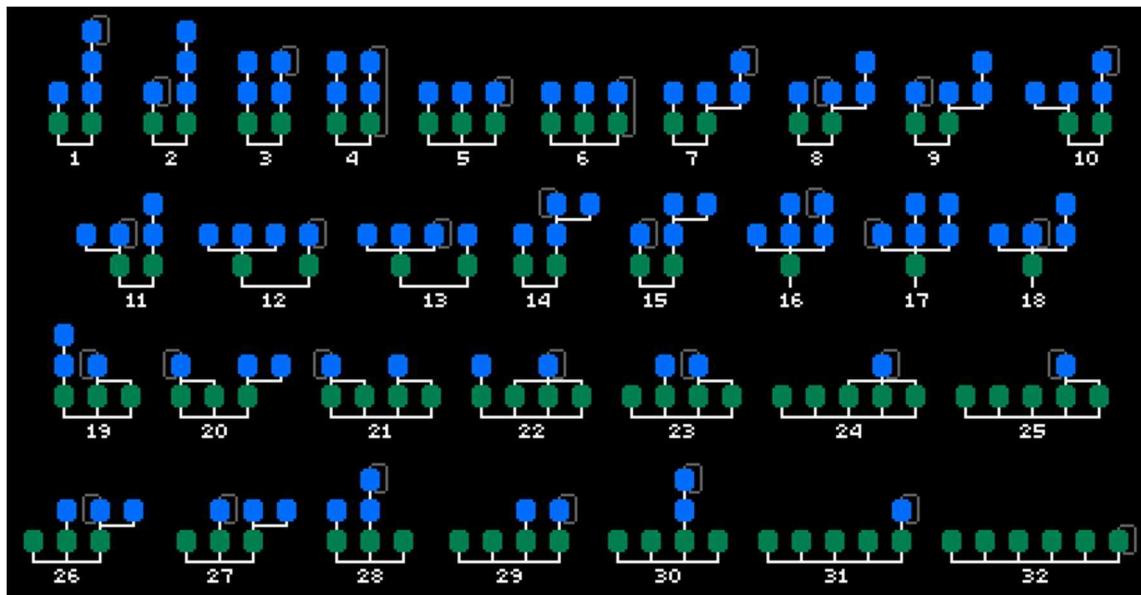


Figura 5. Algoritmos utilizados en el sintetizador Yamaha DX7. Cada algoritmo está compuesto por la combinación de seis operadores.

1.2.4. Síntesis mediante tabla de ondas

Es la implementación digital más común para osciladores [4]. La forma de onda de un periodo del oscilador se almacena en una “tabla de ondas”, en lugar de calcularse en tiempo real. Para reproducir el sonido del oscilador se lee la información almacenada en la tabla a una velocidad concreta, lo que determina la frecuencia fundamental del oscilador. Es común reproducir varias tablas de ondas a la vez o comenzar reproduciendo una forma de onda al pulsar una nota para poco a poco transformarla en otra. Se consiguen así sonidos que evolucionan con el tiempo.

Según la frecuencia de la nota deseada puede ser necesario interpolar las muestras contenidas en la tabla de ondas para generar una nota de diferente frecuencia [9]. En la Figura 6, tomada de [4] se representa este proceso de lectura e interpolación.

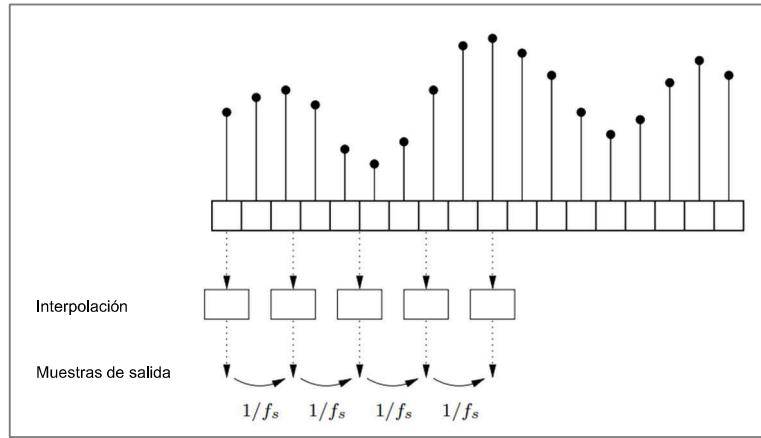


Figura 6. “Un buffer, que almacena una señal, se lee a intervalos de $1/f_s$, donde f_s es la frecuencia de muestreo. Se emplea interpolación.”

1.2.5. Síntesis granular

El sonido se genera mediante procesos que utilizan “gránulos” (pequeños segmentos enventanados) de la señal de audio. El audio se divide en pequeñas porciones (de 1 a 100 ms), que se reproducen alterando su orden, velocidad de reproducción o reproduciendo varios gránulos a la vez [10].

En [11] se distinguen dos tipos de síntesis granular. El primer tipo, sincrónico, se basa en la repetición de los gránulos a una frecuencia regular, con el fin de reproducir sonidos con una altura tonal determinada. El segundo tipo, asincrónico reproduce los gránulos de forma aleatoria para producir “texturas sonoras”.

1.2.6. Síntesis mediante modelado físico

La síntesis por modelado físico se puede definir, según [12], como el método de generación del sonido en el que se emplea un modelo matemático para simular la fuente física del sonido.

En general, se utiliza un conjunto de ecuaciones diferenciales que describen la presión del aire, el desplazamiento de cuerdas o membranas, etc. Para calcular el sonido producido por el instrumento, se resuelve el conjunto de ecuaciones aproximándolas a una solución numérica para obtener una señal de salida dada una excitación de entrada.

Para evitar resolver la ecuación diferencial en tiempo real es posible emplear técnicas como la guía de ondas que reflejan el comportamiento de la ecuación diferencial mediante el uso de *buffers*, filtros y amplificadores.

1.3. Métodos de síntesis por modelado físico

1.3.1. Guía de ondas

Una de las implementaciones más comunes de la síntesis por modelado físico es mediante la utilización de *guías de ondas*, que están formadas por líneas de retardo y filtros, para simular la transmisión y reflexión del sonido en el aire, en una cuerda, una barra, etc. En la Figura 7 se muestra la configuración propuesta en [13] para la síntesis de la vibración en una cuerda ideal. Esta estructura se puede obtener a partir de la solución de D'Alembert para la ecuación de onda, que se explica con más detalle en el Capítulo 2 y que supone una onda viajera hacia cada extremo de la cuerda.

Las condiciones iniciales de la cuerda se introducen en las dos líneas de retardo y se desplazan hacia cada uno de los extremos. En cada uno de los extremos de la cuerda se produce una inversión de fase (ganancia -1). El resultado se obtiene sumando en un punto de la cuerda la salida de las dos líneas de retardo.

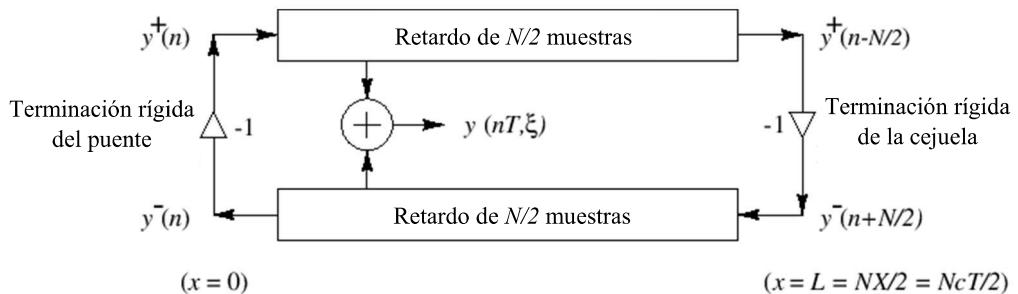


Figura 7. Implementación de la ecuación de onda ideal en una cuerda mediante guía de ondas.

Para simular la absorción del aire (constante a todas las frecuencias) se reduce la ganancia en el extremo de la cuerda por debajo del valor unitario, manteniendo el cambio de fase.

Se puede introducir, además, un filtro en uno de los extremos de la cuerda (paso bajo en este caso) que provoca que las pérdidas sean mayores a altas frecuencias, de modo que las frecuencias graves permanezcan durante más tiempo vibrando en la cuerda.

1.3.2. Aproximación de la ecuación diferencial

Aunque es más costoso computacionalmente que utilizar guías de onda se puede implementar directamente la ecuación diferencial, discretizándola mediante diferencias finitas. Si se le añaden todos los términos de la ecuación diferencial da lugar a una vibración amortiguada cuyas pérdidas dependen de la frecuencia. Una vez implementada la ecuación diferencial sólo es necesario aplicar unas condiciones de contorno y unas condiciones iniciales a la cuerda y tomar la vibración en un punto de la cuerda para generar el sonido.

El proceso de implementación mediante diferencias finitas de la ecuación diferencial se explica con detalle en el apartado 3.4.

1.4. Ventajas y desventajas de la síntesis por modelado físico

La principal ventaja de este método es el de la calidad de sonido que se consigue en comparación con otros métodos de síntesis. Una vez conocidos los elementos básicos de un instrumento (cuerdas, membranas, placas...) es sencillo extrapolarlo a otro sin realizar demasiadas modificaciones.

Mediante la síntesis por modelado físico es posible emular tanto instrumentos acústicos como instrumentos eléctricos e incluso electrónicos aplicando, por ejemplo, modelos en tiempo real de filtros, amplificadores y osciladores para modelar sintetizadores analógicos clásicos. Otra ventaja de este método es que permite generar instrumentos con características físicas imposibles: dimensiones y proporciones extraordinarias, materiales inusuales, etc.

En la actualidad existen numerosos ejemplos de instrumentos virtuales en el mercado que utilizan este tipo de síntesis para reproducir sonidos de instrumentos reales (como el plugin de emulación de bajo MODO Bass de IK Multimedia, o SWAM Strings, de Audio Modeling que emula instrumentos de cuerda frotada), así como algunos ejemplos de sintetizadores experimentales que no buscan simular el sonido de ningún instrumento en concreto, sino que permiten combinar elementos mecánicos, acústicos y eléctricos para generar sonidos nuevos (un buen ejemplo de este tipo de sintetizadores puede ser el sintetizador de percusión Chromaphone 3 de Applied Acoustics Systems).

En muchos casos, como en los ejemplos mencionados en el párrafo anterior se consiguen sonidos indistinguibles de los del sonido real emulado, permitiendo además en ocasiones alterar las propiedades del instrumento, pudiendo producir sonidos que no son posibles en los instrumentos reales debido a las limitaciones físicas de los materiales o del medio de transmisión del sonido.

Uno de los problemas de este tipo de síntesis es el elevado coste computacional que conlleva. Como explica Marc-Pierre, de Applied Acoustics Systems en [14], al ejecutarse el programa no existen muestras pregrabadas, sino que el sonido debe generarse en ejecución, lo que requiere cierto uso de la CPU. Sin embargo, si se optimiza bien, es posible conseguir muy buenos resultados con un coste computacional aceptable.

1.5. Especificaciones de diseño

El objetivo del proyecto es el desarrollo de un sintetizador software en forma de plugin que emule el sonido del Harpejji mediante modelado físico de las cuerdas del instrumento.

El plugin se desarrolla en VST3 (Virtual Studio Technology versión 3) para que sea posible ejecutarlo en cualquier estación de trabajo de audio digital estándar, tanto en Windows como en MacOS.

En la fase de anteproyecto se establecieron las siguientes especificaciones de diseño para el sintetizador:

- El instrumento virtual tendría la forma de un plugin multiplataforma que se podría ejecutar en cualquier estación de trabajo de audio digital estándar compatible con VST, tanto en Windows como en MacOS.
- El plugin emularía el sonido de un Harpejji mediante síntesis por modelado físico de las cuerdas del instrumento.
- El plugin tomaría a su entrada instrucciones MIDI provenientes del programa huésped y proporcionaría en su salida de audio el sonido emulado del instrumento en formato mono.
- El plugin funcionaría en tiempo real, sin latencia apreciable por el usuario.
- El instrumento virtual sería sensible a la velocidad de las notas MIDI que se le introduzcan.
- El usuario podría modificar los parámetros físicos del instrumento mediante la interfaz gráfica del plugin.
- El instrumento sería polifónico con un número de voces a determinar según el rendimiento.

2. Vibraciones en cuerdas

Con el objetivo de modelar las cuerdas del Harpejji primero es necesario conocer las propiedades de las cuerdas en general y cómo se propagan las vibraciones en ellas. Se explica en este capítulo cómo se produce la vibración en las cuerdas, siguiéndose en la investigación el desarrollo teórico de [15].

Se consideran únicamente movimientos transversales, perpendiculares a la dirección de la cuerda. Aunque existen también movimientos longitudinales y torsionales en las cuerdas, su importancia en el sonido puede ser considerada despreciable en comparación con la aportación de los transversales.

2.1. La cuerda ideal

Se comienza por considerar una cuerda como un conjunto de masas y resortes, como se observa esquemáticamente en la Figura 8. Se añade un número de masas tal que la distancia entre ellas se reduce a dx y el número total de masas es $n = L/dx$.

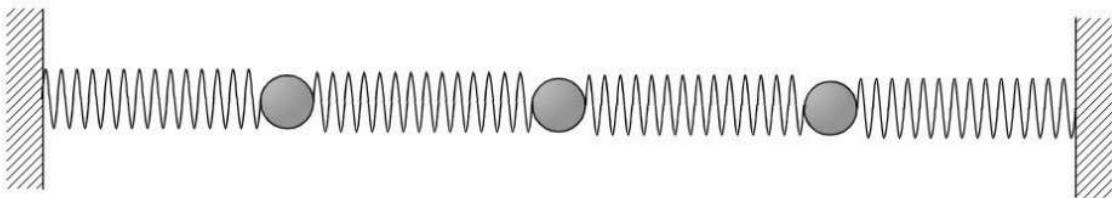


Figura 8. Representación de una cuerda como un sistema de masas y resortes.

Por cada masa que se añade al sistema aparece en la cuerda un nuevo modo propio transversal, como se representa en la Figura 9.

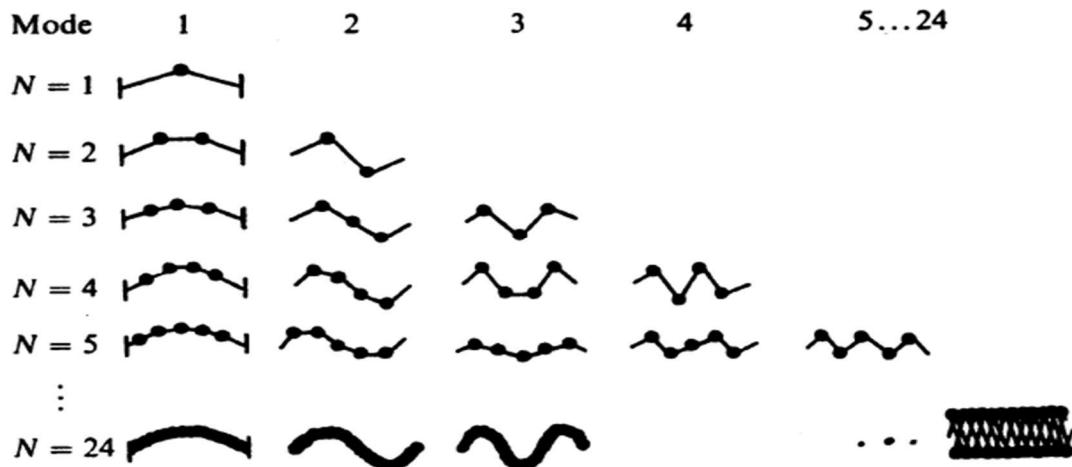


Figura 9. Modos de vibración de la cuerda a medida que se introducen masas en el sistema.

A partir de la expresión de la tensión en la cuerda y aplicando la serie de Taylor y la segunda ley de Newton (el desarrollo paso a paso se puede encontrar en [15]) se llega a la ecuación (1), que representa el movimiento de las ondas transversales en una onda vibrante.

$$\frac{\partial^2 y}{\partial t^2} = \frac{T}{\mu} \frac{\partial^2 y}{\partial x^2} \quad (1)$$

Donde x es la dirección paralela al eje longitudinal de la cuerda, y es la dirección transversal, T es la tensión en Newtons y μ es la densidad lineal de la cuerda, en kg/m.

La solución general para esta ecuación de onda, propuesta por D'Alembert, es la de dos ondas f_1 y f_2 viajando a la misma velocidad en sentidos opuestos de la cuerda.

$$y = f_1(ct - x) + f_2(ct + x) \quad (2)$$

Donde $c = \sqrt{T/\mu}$ es la velocidad de propagación de las ondas en la cuerda.

2.2. Condiciones de contorno.

En el caso de las cuerdas del Harpejji, a diferencia de otros instrumentos con puentes móviles (el violín o el banjo, por ejemplo), se pueden aproximar las condiciones de contorno a las de una cuerda con los extremos fijos, de forma idéntica a la de una guitarra eléctrica. Esto significa que el desplazamiento en una cuerda de longitud L es nulo tanto en el punto $x = 0$ como en el $x = L$ y, por lo tanto, la impedancia en los dos extremos, Z , se aproxima a infinito.

Utilizando la solución de D'Alembert y aplicando la condición de extremo fijo para el punto $x = 0$ se obtiene:

$$y = 0 = f_1(ct - 0) + f_2(ct + 0) \quad (3)$$

Por lo que, en el extremo de la cuerda:

$$f_1(ct) = -f_2(ct) \quad (4)$$

Esto es fácilmente comprobable mediante el cálculo del coeficiente de reflexión, R a través de la ecuación (5).

$$R = \frac{Z_0 - Z}{Z_0 + Z} \quad (5)$$

Donde Z_0 , es la impedancia característica de la cuerda, Z es la impedancia del extremo de la cuerda y $R = A_r/A_i$ es la relación entre la amplitud de la onda incidente y la reflejada.

Para impedancia del extremo infinita se obtiene un coeficiente de reflexión $R = -1$.

$$R = \lim_{Z \rightarrow \infty} \frac{Z_0 - Z}{Z_0 + Z} = \frac{-\infty}{\infty} = -1 \quad (6)$$

Esto significa que se producirá una inversión de fase al reflejarse la onda en el extremo fijo. Este proceso de reflexión se representa en la Figura 10.

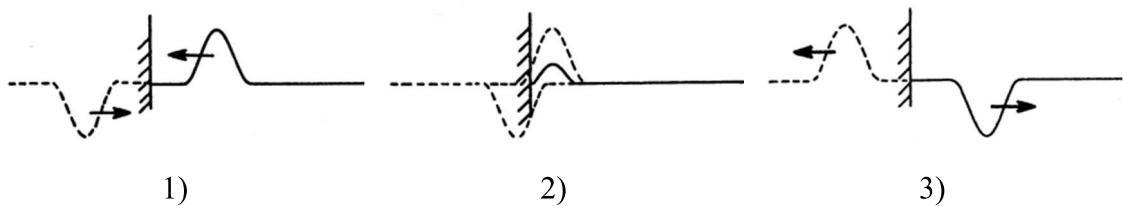


Figura 10. Reflexión de una onda en uno de los extremos fijos de una cuerda.

2.3. Soluciones armónicas de la ecuación de onda

Para estudiar la propagación de movimientos armónicos por la cuerda se consideran las funciones f_1 y f_2 que se desplazan hacia cada lado de la cuerda formadas por un término seno y uno coseno.

$$y(x, t) = A \operatorname{sen}(\omega t - kx) + B \cos(\omega t - kx) + C \operatorname{sen}(\omega t + kx) + D \cos(\omega t + kx) \quad (7)$$

Donde $k = \omega/c$ es el número de onda.

Si se aplican las condiciones de contorno para una cuerda de longitud L y extremos fijos en $(x = 0)$ y $(x = L)$ se obtiene la ecuación (8).

Para extremo fijo en $(x = 0)$:

$$y(0, t) = 0 = A \operatorname{sen}(\omega t) + B \cos(\omega t) + C \operatorname{sen}(\omega t) + D \cos(\omega t) \quad (8)$$

Necesariamente se debe cumplir que $A = -C$ y $B = -D$, por lo que:

$$y = A[\operatorname{sen}(\omega t - kx) - \operatorname{sen}(\omega t + kx)] + B[\cos(\omega t - kx) - \cos(\omega t + kx)] \quad (9)$$

Mediante las fórmulas del coseno y el seno de la suma y la diferencia se puede reducir a la ecuación (10).

$$y = 2[A \cos \omega t - B \sin \omega t] \sin kx \quad (10)$$

Para el extremo ($x = L$) se debe cumplir la segunda condición $y(L, t) = 0$. Para ello es necesario que $\sin(kL) = 0$, o lo que es lo mismo, $\omega L/c = n\pi$. Esta condición restringe los valores de ω tal que $\omega_n = n\pi c/L$, o en términos de frecuencia, $f_n = n(c/2L)$.

Estas frecuencias representan los modos propios de vibración de la cuerda, que responden a la expresión:

$$y_n(x, t) = (A_n \sin \omega_n t + B_n \cos \omega_n t) \sin k_n x \quad (11)$$

Estos modos son armónicos porque cada frecuencia f_n es n veces $f_1 = c/2L$.

La solución general de una cuerda vibrante con extremos fijos se puede escribir como la suma de sus modos propios:

$$y(x, t) = \sum_n (A_n \sin \omega_n t + B_n \cos \omega_n t) \sin k_n x \quad (12)$$

2.4. Pérdidas

Hasta ahora se ha considerado la cuerda como un sistema ideal, sin pérdidas de energía. En una cuerda real, sin embargo, se producen pérdidas de energía debida a distintos factores, que se traducen en una atenuación de las vibraciones. Los tres factores principales que provocan pérdidas en las cuerdas son el aire, el material de la cuerda y la transmisión de energía a otros sistemas vibrantes en los extremos de la cuerda.

Para reproducir el amortiguamiento viscoso, como es el del aire, es común incluir un término de atenuación que se opone al movimiento en la ecuación diferencial y que depende de una constante σ_0 , que determina cuán rápido se atenúan las ondas en la cuerda.

$$\frac{\partial^2 y}{\partial t^2} = \frac{T}{\mu} \frac{\partial^2 y}{\partial x^2} - \sigma_0 \frac{\partial y}{\partial t} \quad (13)$$

La solución a esta nueva ecuación es también armónica pero multiplicada por un factor exponencial negativo dependiente de σ_0 .

$$y(x, t) = e^{-\sigma_0 t} \sum_{n=1}^{\infty} (A_n \sin \omega_n t + B_n \cos \omega_n t) \sin k_n x \quad (14)$$

Este nuevo término provoca una atenuación constante a todas las frecuencias. En una cuerda real, sin embargo, los armónicos más agudos se atenúan antes que los más graves. Para reproducir este comportamiento se introduce en la ecuación de onda un nuevo término de atenuación dependiente de la frecuencia. Se elige el cambio de curvatura de la cuerda de forma que a frecuencias más altas el cambio de curvatura es mayor y se produzca una mayor atenuación.

La ecuación completa con los dos términos de atenuación se incluye en (15).

$$\frac{\partial^2 y}{\partial t^2} = \frac{T}{\mu} \frac{\partial^2 y}{\partial x^2} - \sigma_0 \frac{\partial y}{\partial t} + \sigma_1 \frac{\partial}{\partial t} \frac{\partial^2 y}{\partial x^2} \quad (15)$$

El tiempo de decaimiento varía con la frecuencia como se observa en la Figura 11, tomada de [4] y se controla con los parámetros σ_0 y σ_1 .

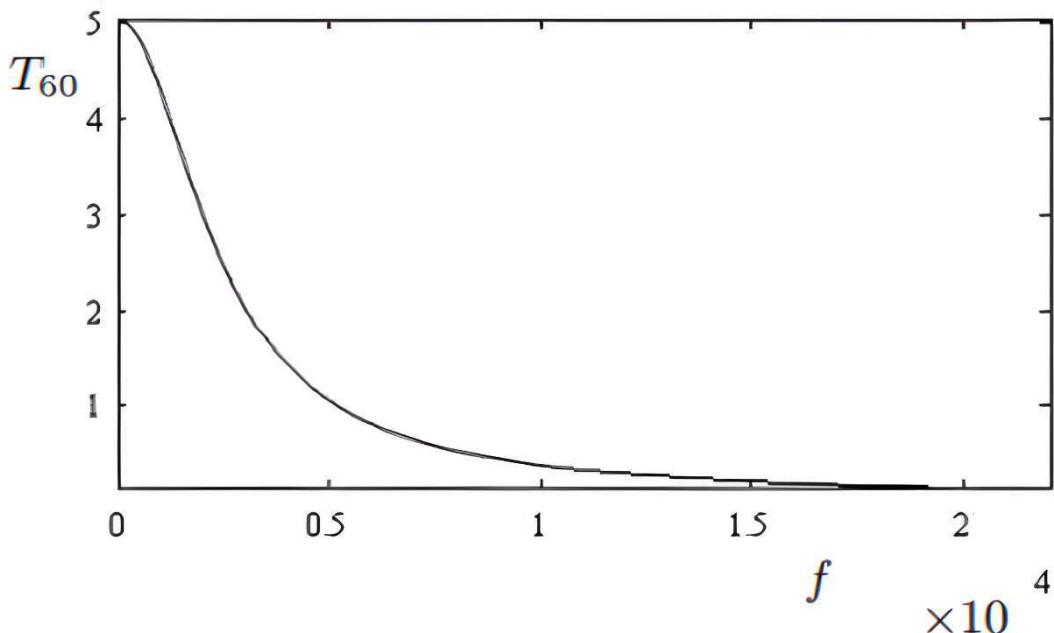


Figura 11. Gráfica del T_{60} respecto a la frecuencia, con $T_{60} = 3s$ a 2500 Hz y $T_{60} = 1s$ a 5 kHz.

2.5. Rigidez

En cierta medida las cuerdas de los instrumentos musicales se comportan como barras rígidas. Cuando una barra se dobla, la parte exterior se alarga mientras que la parte interna se comprime. A partir de la fuerza en la cuerda [15], representada en la ecuación (16) se puede obtener el término de la ecuación diferencial para la rigidez (17).

$$F = \frac{\partial M}{\partial x} = -ESK^2 \frac{\partial^3 y}{\partial x^3} \quad (16)$$

$$\frac{\partial^2 y}{\partial t^2} = -\frac{EK^2}{\rho} \frac{\partial^4 y}{\partial x^4} \quad (17)$$

Donde:

- $K = a/2$ es el radio de giro para formas cilíndricas y a es el radio de la cuerda o barra.
- E es el Módulo de Young o módulo de elasticidad longitudinal, que caracteriza el comportamiento elástico de un material [16].
- S es el área de la sección transversal de la cuerda.
- ρ , la densidad volumétrica de la cuerda o barra.

Se añade este término a la ecuación diferencial y se obtiene así la ecuación diferencial completa para la vibración de una cuerda.

$$\frac{\partial^2 y}{\partial t^2} = \frac{T}{\mu} \frac{\partial^2 y}{\partial x^2} - \sigma_0 \frac{\partial y}{\partial t} + \sigma_1 \frac{\partial}{\partial t} \frac{\partial^2 y}{\partial x^2} - \frac{EK^2}{\rho} \frac{\partial^4 y}{\partial x^4} \quad (18)$$

Este término introduce una diferencia en la velocidad de propagación de las ondas en la cuerda o barra para las diferentes frecuencias, es decir, introduce dispersión.

3. Desarrollo del plugin

En este capítulo se expone paso a paso el proceso de desarrollo del plugin: se explica la elección de la tecnología utilizada para el desarrollo y el formato final del plugin y el funcionamiento general del programa. Se detalla el proceso de aproximación de la ecuación de onda mediante diferencias finitas y su traslado a código en C++. A continuación, se enumeran los distintos tipos de excitación que se han considerado para la cuerda y su efecto en el sonido del instrumento. Se detallan finalmente todos los elementos que conforman el plugin, tanto del *backend* como de la interfaz de usuario y se explica su funcionamiento.

3.1. Virtual Studio Technology (VST)

Se ha optado por desarrollar el plugin en formato VST (Virtual Studio Technology) por su amplia compatibilidad con la mayoría de las estaciones de trabajo con audio digital (DAW, por sus siglas en inglés).

VST [17] es una interfaz que permite la comunicación entre sintetizadores de audio y plugin de efectos y software de edición de audio, secuenciación y grabación. Fue desarrollado por Steinberg y lanzado en el año 1996, transformando por completo los procesos de grabación y edición de audio. En la actualidad los plugin han reemplazado casi por completo a los equipos hardware en los estudios de grabación por su practicidad y menor coste.

La mayoría de los plugin VST se pueden clasificar como instrumentos VST (VSTi) o efectos de audio (VSTfx), aunque también existen plugin de efectos MIDI, medidores, analizadores de espectro, etc. Los plugin VST se controlan mediante una interfaz de usuario gráfica, que se presenta en ventanas.

La principal ventaja de la tecnología VST es que permite conectar software de diferentes compañías y funcionalidades en único sistema de grabación dentro del ordenador personal. Esto la convierte en una tecnología muy versátil y flexible.

3.2. Librería JUCE

Se ha decidido utilizar la librería JUCE para desarrollar el plugin empleando C++ como lenguaje de programación. Esta librería de código abierto para uso no comercial se encarga de la mayoría de los procesos de entrada, salida y muestreo, simplificando muchos procesos y limitando las operaciones que son necesarias realizar muestra a muestra.

Se configura JUCE para generar una plantilla de aplicación sintetizador, utilizando la clase *Synthesiser* de JUCE [18], que genera las funciones básicas para recibir a su entrada instrucciones MIDI y reproducir a su salida lo que se coloque en el buffer de salida del plugin. El programa se divide en cuatro *scripts* y en cada uno de ellos se escriben las instrucciones relativas a una parte del programa: *PluginProcessor*, *PluginEditor*, *SynthesizerVoice* y *SynthesizerSound*.

- *PluginProcessor*: en este *script* se programa el *backend* del plugin. Se incluyen en él los procesos de audio, se crean los parámetros controlables por el usuario y, en el caso de un sintetizador, se generan los objetos de tipo *SynthesizerVoice* (cada una de las voces del sintetizador). También se realiza en él el procesado de audio que afecta a todas las voces; en este caso se realiza la ganancia y el filtrado del conjunto de todas las voces.
- *PluginEditor*: en él se incluyen las instrucciones relativas a la interfaz de usuario y a los aspectos gráficos del plugin. En el caso del sintetizador se crean en este

script los potenciómetros que controlan los parámetros creados previamente en PluginProcessor y se establece su posición dentro de la ventana del plugin. En este caso también se incluye en este documento las instrucciones necesarias para visualizar la vibración de las cuerdas y representarlas sobre un *Harpejji* virtual.

- SynthesiserVoice: se realizan aquí el grueso de las operaciones relacionadas con el modelado físico de la cuerda del Harpejji. Al recibir una nota MIDI, el plugin crea una cuerda con las características adecuadas (longitud, tensión y densidad lineal) para que se corresponda con la frecuencia de la nota pulsada y establece las condiciones iniciales en ella, dependiendo de la velocidad de pulsación del teclado. Se calcula la posición de la cuerda para las muestras siguientes hasta que el nivel caiga por debajo de un umbral. Después, se apaga esa voz.
- SynthesiserSound: en este *script* únicamente se establecen algunos parámetros necesarios para el funcionamiento del programa.

3.3. Funcionamiento general del plugin

Al abrirse el plugin en una estación de audio digital éste se inicializa, creándose los parámetros a los que el usuario tendrá acceso desde la interfaz gráfica, así como el filtro paso bajo que servirá como control de tono y que se explica con detalle en el punto 3.10. A continuación, se crea un objeto de tipo SynthesiserSound y se añaden cada una de las voces del sintetizador, de tipo SynthesiserVoice, quedando el plugin con la estructura de la Figura 12.

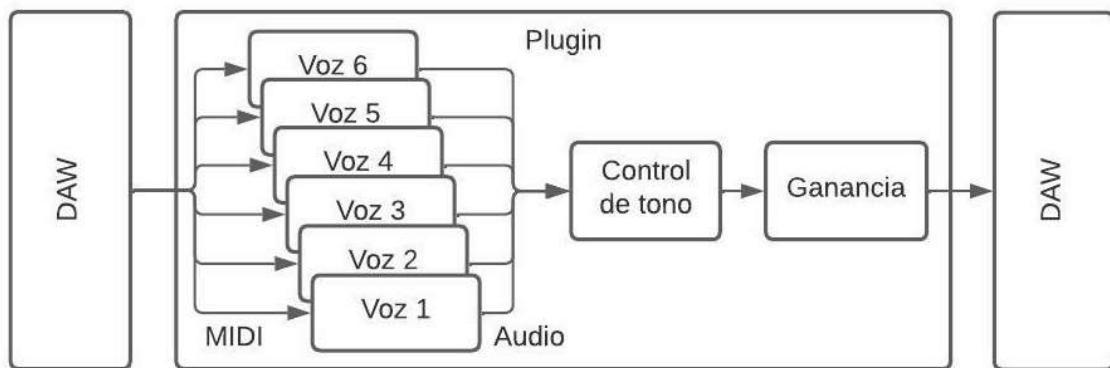


Figura 12. Diagrama de bloques del plugin.

Antes de comenzar a reproducir audio es necesario establecer algunos parámetros generales como son la frecuencia de muestreo, el tamaño de *frame* y el número de canales de salida con el que se va a trabajar. El programa huésped le proporciona esta información al plugin y se almacena para generar correctamente señal de audio que el DAW pueda reproducir.

Una vez inicializado el proceso que se encarga del procesado del audio, se inicializa la interfaz gráfica de usuario, que es un proceso secundario con menor prioridad que el

proceso principal de audio. Se establece su frecuencia de refresco y se muestra la imagen de fondo del plugin, sobre la que se dibujan las cuerdas y los controles rotatorios en ejecución. Todos estos procesos relacionados con la interfaz gráfica de usuario se explican con más detalle en el apartado 3.12.

Con el proceso de audio y la interfaz de usuario inicializados el plugin queda a la espera de recibir mensajes MIDI. Al recibir una nota MIDI, JUCE se encarga de buscar una voz libre (que no esté reproduciendo una nota en ese momento) y realiza una llamada al método startNote() de esa voz. Dentro de este método se comprueba que la nota se encuentra en el rango del Harpejji G16 (C2 – C6) y, si es así, se genera el modelo de la cuerda y se establecen en ella las condiciones iniciales. De lo contrario, se libera la voz.

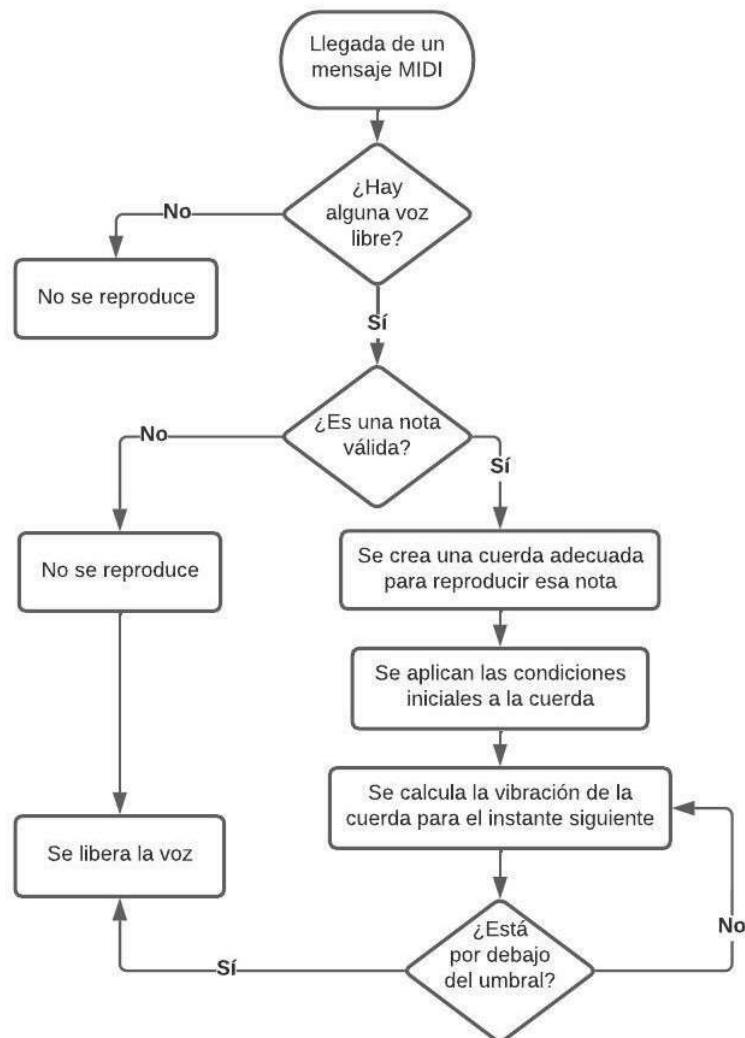


Figura 13. Diagrama de flujo de cada voz del sintetizador.

Para generar el modelo de la cuerda se calcula el paso de muestreo espacial a partir de la frecuencia de la nota recibida. Según la posición (combinación cuerda/traste) en la que se vaya a “tocar” la nota se genera una cuerda de una tensión y longitud determinadas. Una vez calculada la longitud en puntos de muestreo que debe tener la cuerda se crean tres vectores de este tamaño que contendrán la posición de la cuerda en el instante actual (y), en el instante anterior de ejecución (yPrev) y en el instante siguiente (yNext).

Se aplican a continuación las condiciones iniciales en la cuerda. Con este fin se genera una de las funciones de excitación que se explican en el apartado 3.5 adaptada a la longitud de la cuerda calculada. Esta función supondrá la velocidad inicial en la cuerda al pulsarse la nota, que se corrige en amplitud para que dependa de la velocidad MIDI de la nota recibida (detallado en el punto 3.6).

Conociendo la velocidad inicial en la cuerda y sabiendo que el desplazamiento en el momento de su pulsación es nulo en toda la cuerda, se calcula muestra a muestra la posición de la cuerda en el instante siguiente de ejecución. Para ello se emplea la ecuación diferencial de la onda aproximada mediante diferencias finitas. En ella se introducen la posición de toda la cuerda en el instante actual y en el instante previo de ejecución y, a partir de ellas se calcula la posición en el instante siguiente. El proceso de aproximación de la ecuación de onda mediante diferencias finitas se explica en profundidad en el apartado 3.4.

Se toma entonces la posición de la cuerda en el instante actual, en la posición de lectura de la cuerda y se introduce en el buffer de salida del plugin. Se repite el proceso muestra a muestra hasta que se completa el *frame* correspondiente a cada una de las voces activas y se suman las contribuciones de todas las voces a la salida del plugin.

Es importante tener en cuenta que la nota MIDI no tiene por qué llegar en un instante que coincida con el inicio de un *frame*. A la llegada de una nota MIDI, JUCE proporciona el número de muestra en el que ésta se produce. Sólo se procesan las muestras entre ésta y el tamaño de *buffer* para el primer *frame*. Los *frames* siguientes se procesan completos.

Se sigue procesando todas las voces activas hasta que el nivel del detector de nivel RMS de alguna de ellas (que se desarrolla en el apartado 3.8) caiga por debajo del umbral. En ese momento se deja de reproducir la nota y se libera la voz.

Antes de aplicarse el *buffer* de salida del plugin se procesa, filtrándolo y aplicándole la ganancia correspondiente a los controles de tono y de ganancia (apartados 3.10 y 3.11).

La interfaz gráfica se dibuja 60 veces por segundo mediante una llamada al método *paint()*; esto incluye el fondo (imagen .JPG), las cuerdas del instrumento y los controles rotatorios con los que puede interactuar el usuario. Estos controles rotatorios están ligados a unos parámetros (*tension*, *sustain*, *tone* y *gain*) que se pasan de la interfaz al proceso principal de audio que, a su vez, los pasa a cada una de las voces para que se realicen los cambios necesarios en las características de las cuerdas.

Si alguna de las voces está activa se dibuja la vibración de la cuerda sobre la interfaz gráfica en la posición de la cuerda correspondiente. Para ello, cada una de las voces proporciona la vibración de la cuerda al proceso principal de audio aproximadamente una vez cada 5 ms y éste se la pasa a la interfaz gráfica para dibujarla cuando se llame al método *paint()* (cada 16 ms).

3.4. Aproximación de la ecuación de onda mediante diferencias finitas

El código del programa calcula la posición de la cuerda para el instante de tiempo siguiente a partir de la posición de la cuerda en el instante actual y el instante anterior. Para ello, es necesario adaptar la ecuación de onda, discretizándola y posteriormente traducirla a código en C++. Se emplean las aproximaciones por diferencias finitas siguientes para discretizar la ecuación diferencial tomadas de [19].

Derivada primera. Diferencia hacia atrás:

$$\frac{\partial f}{\partial g} \approx \frac{f_{i-1} - f_i}{\Delta g} \quad (19)$$

Derivada primera. Diferencia hacia delante:

$$\frac{\partial f}{\partial g} \approx \frac{f_i - f_{i+1}}{\Delta g} \quad (20)$$

Derivada segunda. Diferencia central:

$$\frac{\partial^2 f}{\partial g^2} \approx \frac{f_{i+1} - 2f_i + f_{i-1}}{(\Delta g)^2} \quad (21)$$

Utilizando estas expresiones se discretizan individualmente cada uno de los términos de la ecuación diferencial para posteriormente combinarlos en una única expresión.

$$\frac{\partial^2 y}{\partial t^2} = \frac{T}{\mu} \frac{\partial^2 y}{\partial x^2} - 2\sigma_0 \frac{\partial y}{\partial t} + \sigma_1 \frac{\partial}{\partial t} \frac{\partial^2 y}{\partial x^2} \quad (22)$$

El objetivo es calcular la posición de la cuerda en el instante de muestreo siguiente, que se denota en las ecuaciones como $y(x, t + 1)$. Con este fin, se calculan individualmente todos los términos de la ecuación y una vez obtenidas las expresiones de todos ellos se halla la expresión de $y(x, t + 1)$, combinándolas.

3.4.1. Término de la segunda derivada

Para discretizar el primer término, a la izquierda de la ecuación, $\left(\frac{\partial^2 y}{\partial t^2}\right)$ se aplica directamente la expresión para la segunda derivada empleando diferencias centrales (21), que da lugar a la ecuación (23).

$$\frac{\partial^2 y}{\partial t^2} \approx \frac{y(x, t+1) - 2y(x, t) + y(x, t-1)}{(\Delta t)^2} \quad (23)$$

3.4.2. Término de la cuerda ideal

El segundo término $\left(\frac{T}{\mu} \frac{\partial^2 y}{\partial x^2}\right)$, que representa la ecuación de onda de una cuerda ideal, se aproxima en primer lugar aplicando la misma expresión de la ecuación (21) a la segunda derivada de y respecto de x . Se obtiene así la ecuación (24).

$$\frac{\partial^2 y}{\partial x^2} \approx \frac{y(x+1, t) - 2y(x, t) + y(x-1, t)}{(\Delta x)^2} \quad (24)$$

Que al añadirle el factor $\frac{T}{\mu}$ queda como en la ecuación (25).

$$\frac{T}{\mu} \frac{\partial^2 y}{\partial x^2} \cong \frac{T}{\mu} \frac{y(x+1, t) - 2y(x, t) + y(x-1, t)}{(\Delta x)^2} \quad (25)$$

3.4.3. Término de atenuación lineal

El tercer término $\left(-2\sigma_0 \frac{\partial y}{\partial t}\right)$, que se corresponde con la atenuación lineal, igual a todas las frecuencias, se puede aproximar empleando cualquiera de las dos expresiones (19) y (20) para la primera derivada. Al aplicarlas a las variables de la ecuación de onda, resultan respectivamente en las ecuaciones (26) y (27).

$$\frac{\partial y}{\partial t} \approx \frac{y(x, t-1) - y(x, t)}{\Delta t} \quad (26)$$

$$\frac{\partial y}{\partial t} \approx \frac{y(x, t) - y(x, t+1)}{\Delta t} \quad (27)$$

Dado que $y(x, t+1)$ no es conocido (es la muestra en el instante siguiente de ejecución y el objetivo de todos estos cálculos), la segunda expresión (27) no es útil en este caso. Se utiliza la aproximación por diferencias finitas hacia atrás, que emplea la muestra anterior. Con ello y añadiendo el factor $-2\sigma_0$, la expresión para el término de atenuación lineal queda como:

$$-2\sigma_0 \frac{\partial y}{\partial t} \cong -2\sigma_0 \frac{y(x, t-1) - y(x, t)}{\Delta t} \quad (28)$$

3.4.4. Término de la atenuación dependiente de la frecuencia.

Para aproximar el tercer término $\left(\sigma_1 \frac{\partial}{\partial t} \frac{\partial^2 y}{\partial x^2}\right)$, que se corresponde con la atenuación dependiente de la frecuencia; en primer lugar, se utiliza la expresión de la segunda derivada empleando diferencias centrales (21), de la que se obtiene la ecuación (29). A continuación, se aplica a esta la expresión de la primera derivada y diferencias finitas hacia atrás (19), obteniéndose la ecuación (30).

$$\frac{\partial^2 y}{\partial x^2} \approx \frac{y(x+1, t) - 2y(x, t) + y(x-1, t)}{(\Delta x)^2} \quad (29)$$

Se han apaisado las siguientes páginas para mejorar la legibilidad de las ecuaciones.

$$\frac{\partial}{\partial t} \frac{\partial^2 y}{\partial x^2} \approx \left[\frac{y(x+1,t) - 2y(x,t) + y(x-1,t)}{(\Delta x)^2} - \frac{y(x+1,t-1) - 2y(x,t-1) + y(x-1,t-1)}{(\Delta x)^2} \right] / \Delta t \quad (30)$$

Finalmente se añade el factor de atenuación, σ_1 , y la expresión completa para este término queda como:

$$\sigma_1 \frac{\partial}{\partial t} \frac{\partial^2 y}{\partial x^2} \approx \sigma_1 \left[\frac{y(x+1,t) - 2y(x,t) + y(x-1,t) - y(x+1,t-1) + 2y(x,t-1) - y(x-1,t-1)}{(\Delta x)^2 \Delta t} \right] \quad (31)$$

3.4.5. Ecuación de onda completa

Para obtener la posición de la cuerda en el instante siguiente de ejecución se combinan las expresiones obtenidas para cada uno de los términos de la ecuación de onda (22).

$$\frac{\partial^2 y}{\partial t^2} = \frac{T}{\mu} \frac{\partial^2 y}{\partial x^2} - 2\sigma_0 \frac{\partial y}{\partial t} + \sigma_1 \frac{\partial}{\partial t} \frac{\partial^2 y}{\partial x^2} \quad (22)$$

Se combinan las expresiones de las ecuaciones (25), (28) y (31) para obtener la ecuación (32).

$$\begin{aligned} & \frac{y(x,t+1) - 2y(x,t) + y(x,t-1)}{(\Delta t)^2} \\ & \approx \frac{T}{\mu} \frac{y(x+1,t) - 2y(x,t) + y(x-1,t)}{(\Delta x)^2} - 2\sigma_0 \frac{y(x,t-1) - y(x,t)}{\Delta t} \\ & + \sigma_1 \left[\frac{y(x+1,t) - 2y(x,t) + y(x-1,t) - y(x+1,t-1) + 2y(x,t-1) - y(x-1,t-1)}{(\Delta x)^2 \Delta t} \right] \end{aligned} \quad (32)$$

Finalmente, se puede despejar $y(x, t + 1)$ para obtener la expresión de la posición siguiente de la cuerda a partir de las muestras anteriores:

$$\begin{aligned}
 y(x, t + 1) = & \frac{(\Delta t)^2}{(\Delta x)^2} \frac{T}{\mu} (y(x + 1, t) - 2y(x, t) + y(x - 1, t)) + 2y(x, t) - y(x, t - 1) \\
 & - 2\sigma_0 \Delta t [y(x, t - 1) - y(x, t)] \\
 & + 2\sigma_1 \frac{\Delta t}{(\Delta x)^2} [y(x + 1, t) - 2y(x, t) + y(x - 1, t) - y(x + 1, t - 1) + 2y(x, t - 1) \\
 & - y(x - 1, t - 1)]
 \end{aligned} \tag{33}$$

Esta expresión se puede traducir directamente a código en C++ como se muestra en el Código 1. Para ello se utilizan tres vectores de tamaño X , igual a la longitud en muestras de la cuerda, que contienen el desplazamiento en cada punto de la cuerda en el instante actual (vector y), en el instante de ejecución anterior (y_{Prev}) y en el siguiente (y_{Next}). dt es el periodo de muestreo temporal, igual a $1 / f_s$ y dx , el periodo de muestreo espacial, o la distancia entre una posición de la cuerda y la siguiente.

Se calcula el desplazamiento de la cuerda en todas sus posiciones, de $(x = 1)$ a $(x = X - 1)$ ya que los extremos son fijos y su desplazamiento siempre es nulo. Una vez calculada el desplazamiento en toda la cuerda se lee la muestra en la posición de lectura, x_{Read} , cercana a uno de los extremos fijos porque es donde se encuentran las pastillas piezoelectricas en el instrumento real, y se introducen en el *buffer* de salida de esa voz. Una vez leída la muestra se actualizan los tres vectores: la posición actual, y , pasa a ser y_{Prev} y la posición en el instante de ejecución siguiente pasa a ser la posición actual.

Este proceso se repite para cada muestra hasta que se completa el *buffer* de la voz del sintetizador.

Se incluye el código relativo al cálculo de la ecuación diferencial en el Código 1.

```
for (int s = 0; s < synthBuffer.getNumSamples(); s++) {  
    // Cálculo de la posición de la cuerda en el sample siguiente  
  
    for (int x = 1; x < X-1; x++) {  
  
        // Ecuación de onda de la cuerda  
  
        yNext[x] = (T/mu) * (y[x - 1] - 2.0f * y[x] + y[x + 1]) * (powf(dt, 2) / powf(dx, 2)) + 2 * y[x] - prev[x]  
  
        // Atenuación lineal  
        - 2.0f * s0 * (y[x] - yPrev[x]) * dt  
  
        // Atenuación dependiente de la frecuencia  
        + 2.0f * s1 * (y[x + 1] - 2.0f * y[x] + y[x - 1] - yPrev[x + 1] + 2.0f * yPrev[x] - yPrev[x - 1])  
        * dt / powf(dx, 2);  
    }  
  
    // Se añade al buffer el desplazamiento de la cuerda en el punto de lectura.  
    synthBuffer.addSample(0, s, y[xRead]);  
    yPrev = y;  
    y = yNext;  
}
```

Código 1. Implementación de la ecuación de onda mediante diferencias finitas.

3.4.6. Término de la rigidez

A estos términos se puede añadir, por último, el término dependiente de la rigidez de la cuerda (34). Como se ha explicado en el apartado 2.5 este término provoca una dependencia en la velocidad de propagación de las ondas en la cuerda según su frecuencia, es decir, introduce dispersión. Este término no tiene demasiada relevancia en cuerdas con tensión y masa normales, pero se ha incluido para aportar realismo a los casos extremos fuera de las características físicas reales de los materiales.

$$\frac{\partial^2 y}{\partial t^2} = - \frac{E K^2}{\rho} \frac{\partial^4 y}{\partial x^4} \quad (34)$$

Para aproximarla por diferencias finitas se utilizan las siguientes expresiones para la cuarta derivada:

Central:

$$\frac{\partial^4 y}{\partial x^4} \cong \frac{y(x+2,t) - 4y(x+1,t) + 6y(x,t) - 4y(x-1,t) + y(x-2,t)}{(\Delta x)^4} \quad (35)$$

Hacia delante:

$$\frac{\partial^4 y}{\partial x^4} \cong \frac{y(x+4,t) - 4y(x+3,t) + 6y(x+2,t) - 4y(x+1,t) + y(x,t)}{(\Delta x)^4} \quad (36)$$

Hacia atrás:

$$\frac{\partial^4 y}{\partial x^4} \cong \frac{y(x-4,t) - 4y(x-3,t) + 6y(x-2,t) - 4y(x-1,t) + y(x,t)}{(\Delta x)^4} \quad (37)$$

Se usa la expresión para diferencias finitas central para todas las posiciones de la cuerda excepto las de los extremos, en los que no existe una posición siguiente ($x = x + 1$) o anterior ($x = x - 1$). En estos casos se utilizan diferencias hacia atrás o hacia delante según en qué extremo de la cuerda nos encontramos.

La expresión completa para el término de la rigidez queda como en las ecuaciones (38), (39) y (40) utilizando diferencia central, hacia delante y hacia atrás respectivamente.

Central:

$$-\frac{EK^2}{\rho} \frac{\partial^4 y}{\partial x^4} \cong -\frac{EK^2}{\rho} \frac{y(x+2,t) - 4y(x+1,t) + 6y(x,t) - 4y(x-1,t) + y(x-2,t)}{(\Delta x)^4} \quad (38)$$

Hacia delante:

$$-\frac{EK^2}{\rho} \frac{\partial^4 y}{\partial x^4} \cong -\frac{EK^2}{\rho} \frac{y(x+4,t) - 4y(x+3,t) + 6y(x+2,t) - 4y(x+1,t) + y(x,t)}{(\Delta x)^4} \quad (39)$$

Hacia atrás:

$$-\frac{EK^2}{\rho} \frac{\partial^4 y}{\partial x^4} \cong -\frac{EK^2}{\rho} \frac{y(x-4,t) - 4y(x-3,t) + 6y(x-2,t) - 4y(x-1,t) + y(x,t)}{(\Delta x)^4} \quad (40)$$

En la versión final del plugin se ha eliminado el cálculo de este término de la rigidez dado su elevado coste computacional y su impacto despreciable en el sonido.

3.5. Excitación de la cuerda

Una vez que se ha programado el comportamiento de la cuerda, basta con aplicarle unas condiciones iniciales cada vez que se reciba un mensaje MIDI. Para ello, una vez que se ha decidido en qué cuerda se va a tocar la nota, se calcula la velocidad del sonido en ella (dependiente de la densidad lineal y de la tensión de la cuerda y, por tanto, diferente para cada cuerda) y se “genera” una cuerda con la longitud adecuada para reproducir la frecuencia de la nota con esa velocidad de transmisión en la cuerda, utilizando la expresión (42), que a su vez se obtiene de (41).

$$c_0 = L_{esc} \cdot 2f_{min} \quad (41)$$

$$L_{cuerda} = \frac{c_0}{2 \cdot f_{nota}} \quad (42)$$

Tras obtener la longitud de la cuerda, se divide entre el paso de muestreo espacial (dx) para obtener el número de puntos que se consideran de la cuerda (43). A continuación, se establecen las condiciones iniciales en la cuerda aplicándose un desplazamiento y velocidad iniciales a cada punto de la cuerda.

$$X = \frac{L_{cuerda}}{dx} \quad (43)$$

En el caso de los instrumentos de cuerda pulsada (como la guitarra, la mandolina o el arpa, por ejemplo), las condiciones iniciales que se aplican a la cuerda se pueden aproximar a un desplazamiento inicial triangular, como en la Figura 14 y una velocidad inicial nula provocadas al actuar sobre la cuerda el dedo o la púa.

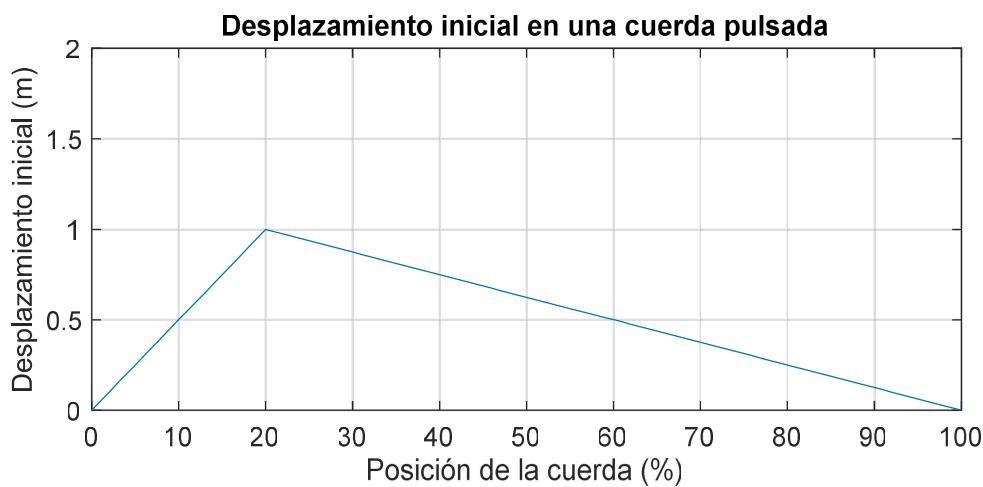


Figura 14. Desplazamiento inicial en instrumentos de cuerda pulsada.

En el caso del Harpejji, su ejecución es distinta: se toca pisando las cuerdas sobre los trastes, sin pulsarlas, confiriendo a las cuerdas una velocidad inicial no nula. Si se considera como condición inicial el momento en que la cuerda entra en contacto con el traste, en ese momento existe en la cuerda, además, un desplazamiento nulo. Esta excitación es más parecida a un instrumento de cuerda percutida como el piano, lo que se refleja en el sonido del instrumento.

En la Figura 15 se incluye un ejemplo de la velocidad inicial en una cuerda percutida con un martillo desde abajo en un punto al 30% de su longitud. Solamente la zona que entra en contacto con el martillo adquiere velocidad en el instante inicial. El desplazamiento inicial es nulo en toda la cuerda.

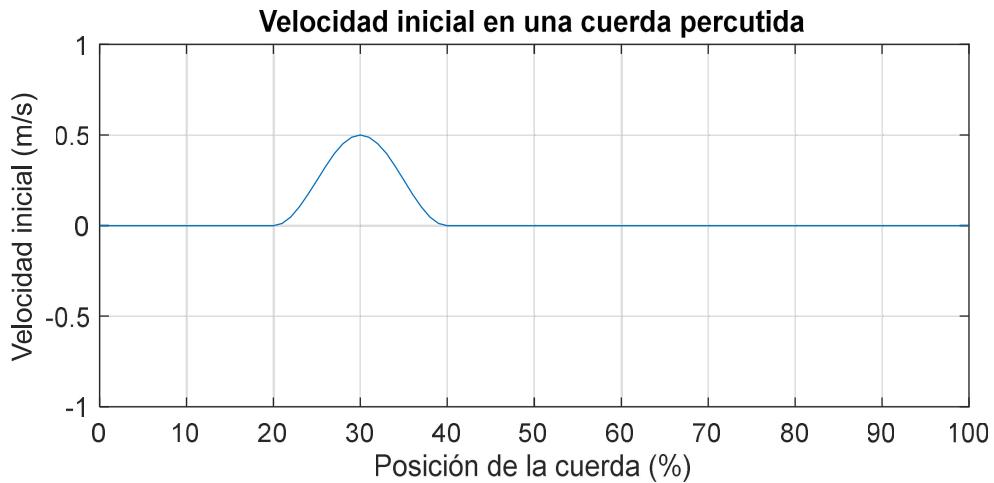


Figura 15. Velocidad inicial en una cuerda percutida desde abajo en el punto al 30% de su longitud.

A diferencia de los instrumentos de cuerda percutida, sin embargo, al pisar las cuerdas del Harpejji se confiere velocidad inicial a toda la cuerda. Se han propuesto varias posibles excitaciones que podrían reproducir la del instrumento real. Para cada una de ellas se ha estudiado el espectro que producen y se ha comparado con sonidos grabados de un Harpejji G16 real. Se incluyen a continuación todas las excitaciones propuestas.

Se han normalizado todas las excitaciones con valor máximo igual a 1 m/s para que sea más fácil trabajar con ellas. Las diferentes excitaciones no producen notas de la misma amplitud, ni la misma función de excitación produce notas de la misma amplitud a diferentes frecuencias por lo que es necesario normalizar su volumen. Este proceso se detalla en el apartado 3.6.

3.5.1. Velocidad inicial constante en toda la cuerda

Como primera idea se propuso que la velocidad inicial en la cuerda fuera constante en toda la cuerda, a excepción de en sus extremos fijos.



Figura 16. Velocidad inicial constante en toda la cuerda.

Para comprobar la evolución del espectro en frecuencia se realiza el spectrograma de la señal de salida del sintetizador, excitado con velocidad constante en toda la cuerda y reproduciendo una nota C3, con una frecuencia fundamental de 130,81 Hz. El spectrograma obtenido se incluye en la Figura 17.

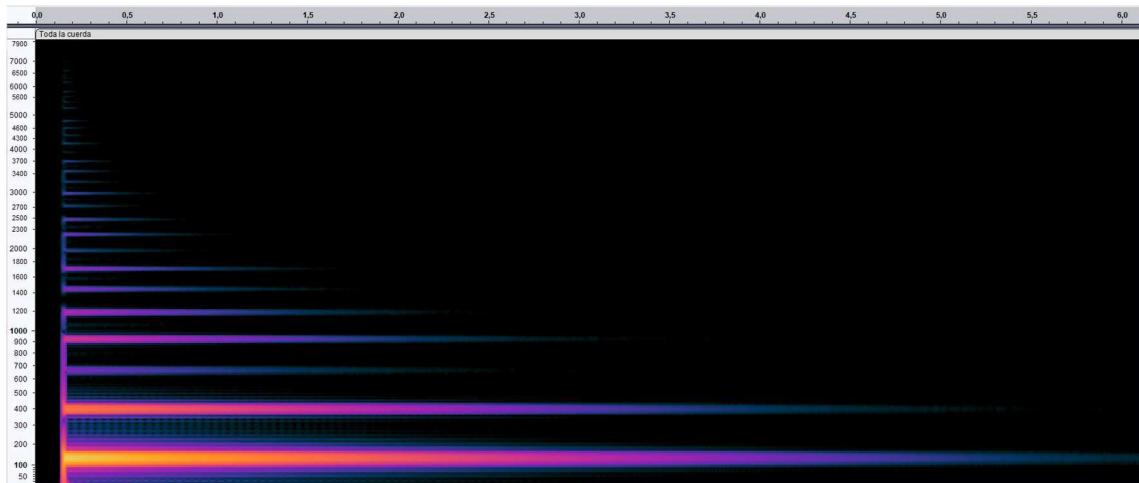


Figura 17. Espectrograma de una nota C3 reproducida con una excitación igual en toda la cuerda.

Esta excitación provoca un espectro en el que sólo aparecen los armónicos impares de la frecuencia fundamental.

3.5.2. Parábola

Se propone seguidamente una excitación de la cuerda con una velocidad inicial tipo parábola a lo largo de la cuerda, utilizando la expresión (44). La velocidad inicial en la cuerda tiene la forma que se observa en la Figura 18 y genera el espectrograma de la Figura 19.

$$v_0(x) = x \cdot (L - x) \quad (44)$$

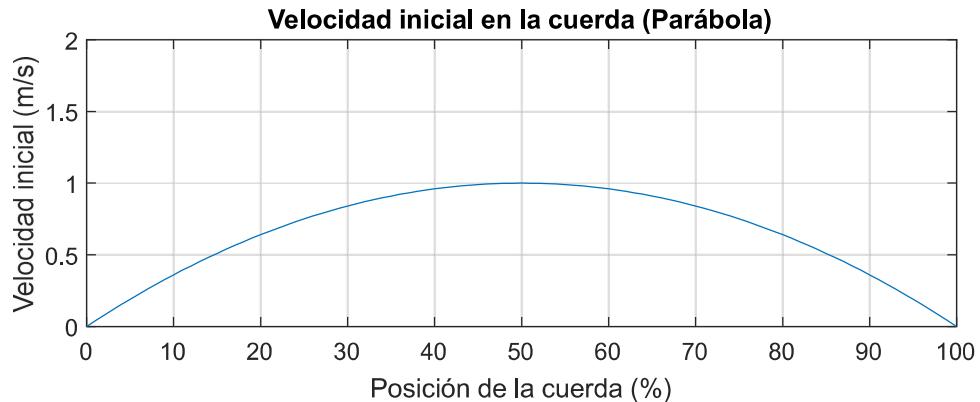


Figura 18. Velocidad inicial con forma de parábola.

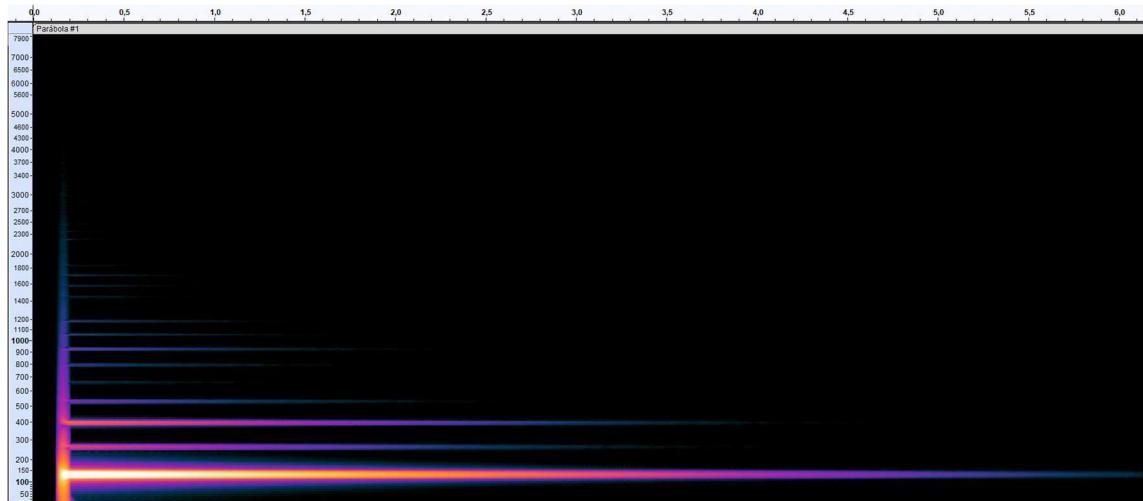


Figura 19. Espectrograma de una nota C3 con una excitación con forma de parábola.

Con esta excitación se generan tanto armónicos pares como impares, pero el contenido espectral a altas frecuencias es muy pobre.

3.5.3. Rampa

Dado que la cuerda se pisa desde uno de los extremos se ha considerado una excitación en la que la velocidad máxima ocurre en el punto más cercano al extremo de la cuerda.

3. Desarrollo del plugin

Como en el resto de los casos los extremos tienen velocidad inicial y desplazamiento nulos.

Se utiliza la expresión de la ecuación (45) con una discontinuidad en el último punto de la cuerda para pasar del máximo de velocidad al extremo fijo. La forma de la velocidad inicial en la cuerda se puede observar en la Figura 20.

$$v_0(x) = \frac{x}{L} \quad (45)$$

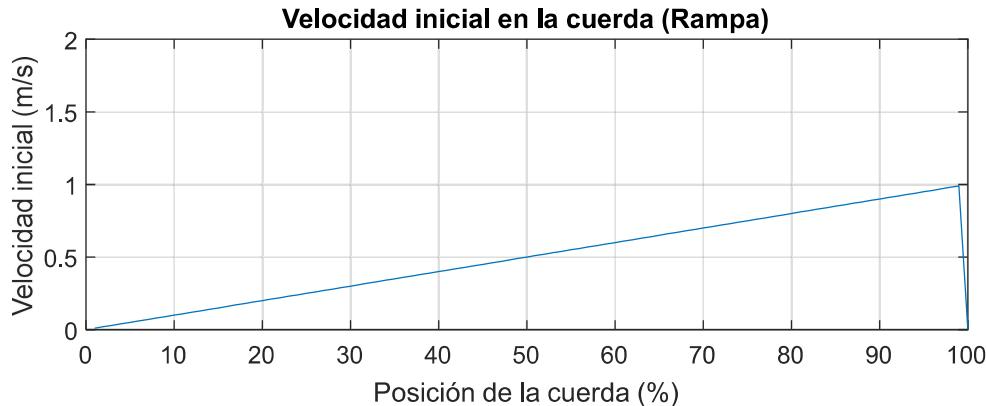


Figura 20. Velocidad inicial con forma de rampa.

Se realiza el espectrograma del instrumento reproduciendo una nota C3 con la excitación en forma de rampa, que se incluye en la Figura 21.

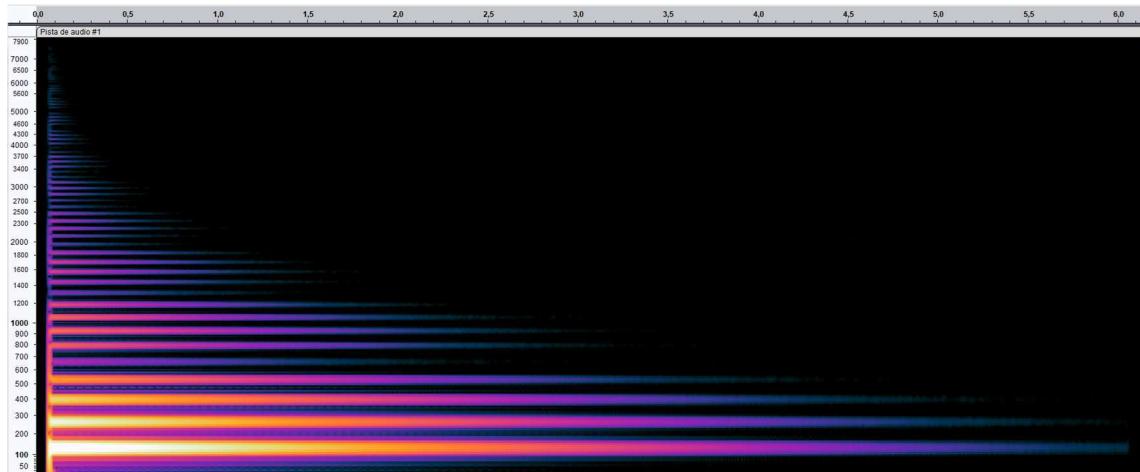


Figura 21. Espectrograma de una nota C3 reproducida con excitación en forma de rampa.

Con esta excitación se generan tanto armónicos pares como impares y un contenido espectral muy rico, extendiéndose los armónicos hasta casi los 8 kHz.

3.5.4. Coseno alzado desplazado

La excitación con forma de rampa considerada en el apartado anterior genera una gran cantidad de armónicos, que se pueden reducir empleando una forma sin discontinuidades, con una transición progresiva entre el punto de mayor velocidad y el extremo fijo.

Se ha empleado una función coseno alzado en la que el primer semiperíodo se ha alargado hasta el punto de velocidad máxima y el segundo se ha encogido para ocupar desde el punto de máxima velocidad hasta el extremo fijo. Para representar matemáticamente esta forma de onda se utiliza la expresión de la ecuación (47).

$$v_0(x) = \begin{cases} 0.5 - 0.5 \cos\left(\frac{x}{x_{max}}\pi\right) & \text{si } x \leq x_{max} \\ 0.5 + 0.5 \cos\left(\frac{x - x_{max}}{L - x_{max}}\pi\right) & \text{si } x > x_{max} \end{cases} \quad (46)$$

Las figuras Figura 22 y Figura 23 incluyen la velocidad inicial en la cuerda y el espectrograma producido al excitar la cuerda con ella reproduciendo un C3. El punto de máxima velocidad, x_{max} , se sitúa al 90% de la cuerda.

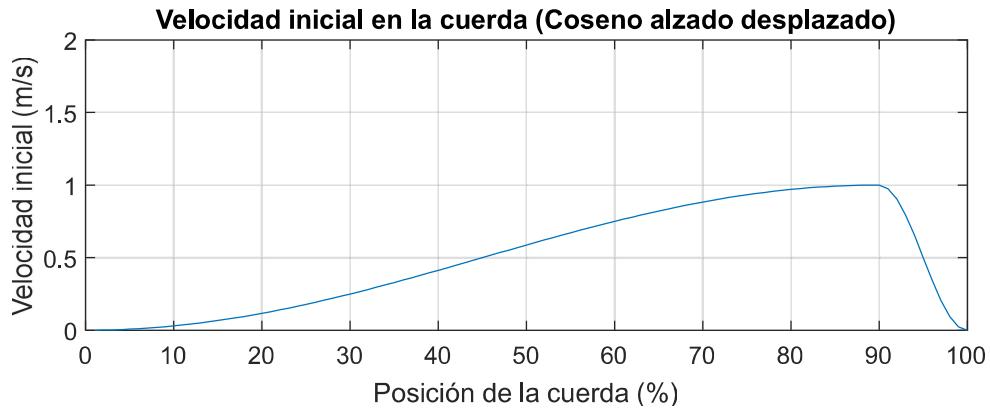


Figura 22. Velocidad inicial con forma de coseno alzado desplazado al 90% de la cuerda

3. Desarrollo del plugin

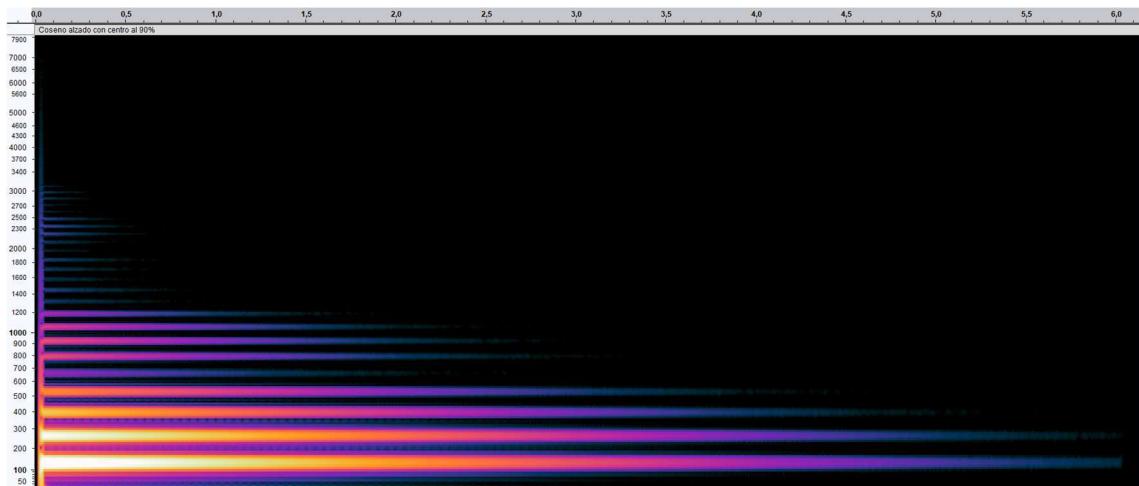


Figura 23. Espectrograma de una nota C3 con una excitación coseno alzado con centro al 90% de la cuerda.

Con esta excitación se consigue un buen número de armónicos a altas frecuencias, tanto pares como impares y un sonido relativamente similar al del Harpejji para los registros graves y medios.

Si se desplaza el punto de máxima velocidad hacia el centro de la cuerda el contenido espectral se reduce. En las figuras Figura 24 y Figura 25 se muestra la velocidad inicial en la cuerda y su espectrograma respectivamente.

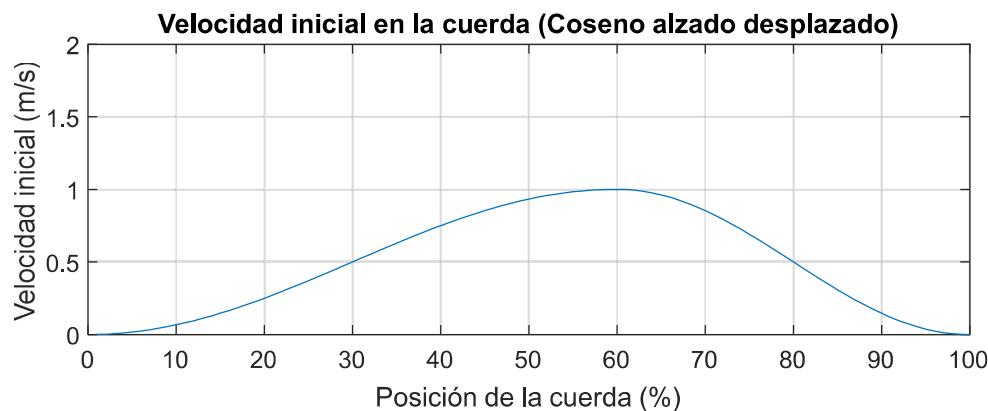


Figura 24. Velocidad inicial con forma de coseno alzado desplazado al 60% de la cuerda.

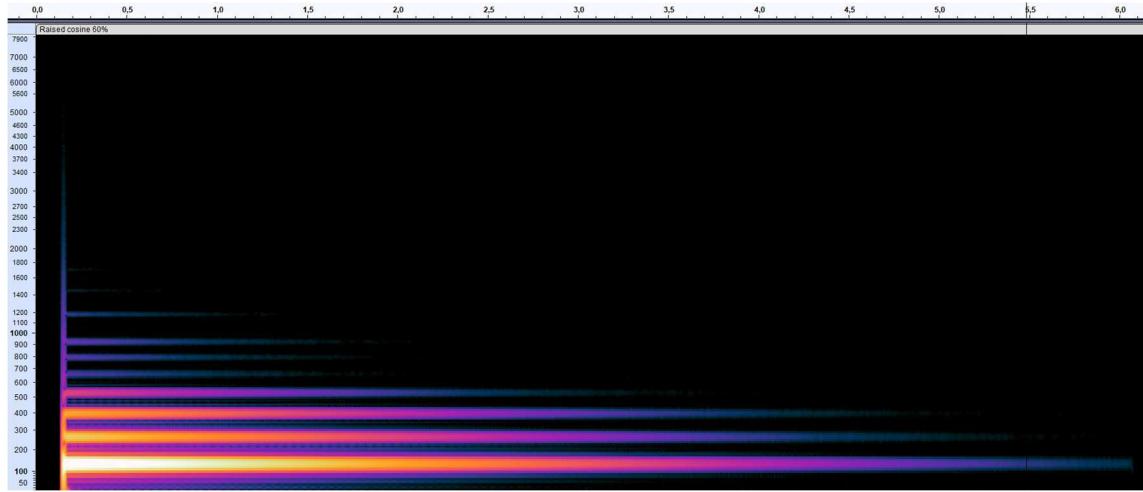


Figura 25. Espectrograma de una nota C3 con una excitación coseno alzado con centro al 60% de la cuerda.

3.5.5. Función trayectoria de proyectil con rozamiento

Se busca posteriormente una función similar a la del coseno alzado modificado del apartado 3.5.4. pero sin las discontinuidades que se pueden observar en la Figura 22 y que no ocurrirían en la cuerda real.

Se utiliza para ello la función de la trayectoria de un proyectil con resistencia del aire y rozamiento de Stokes, [20]. Se modifican los valores de masa, velocidad y dirección inicial del proyectil, así como el rozamiento del aire para conseguir funciones similares a las obtenidas utilizando el coseno alzado. Las funciones obtenidas se incluyen en las figuras Figura 26 y Figura 28 y en las figuras Figura 27 y Figura 29 sus respectivos espectrogramas al reproducir con esta excitación una nota C3.

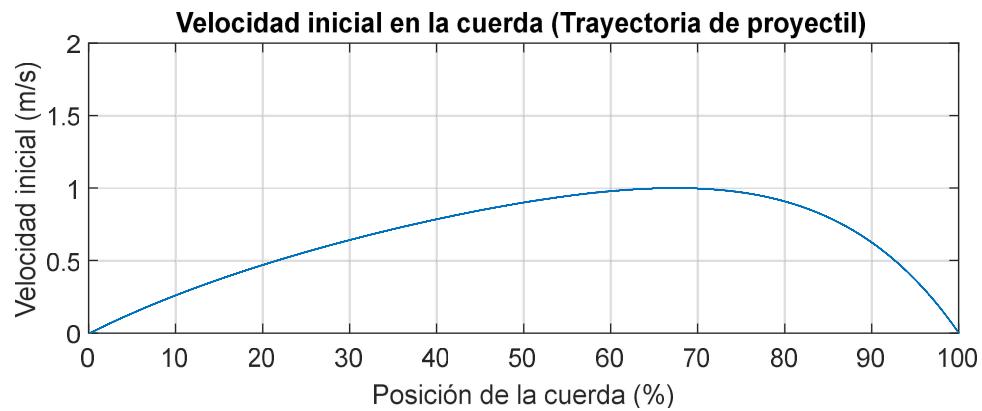


Figura 26. Velocidad inicial con forma de trayectoria de proyectil 1.

3. Desarrollo del plugin

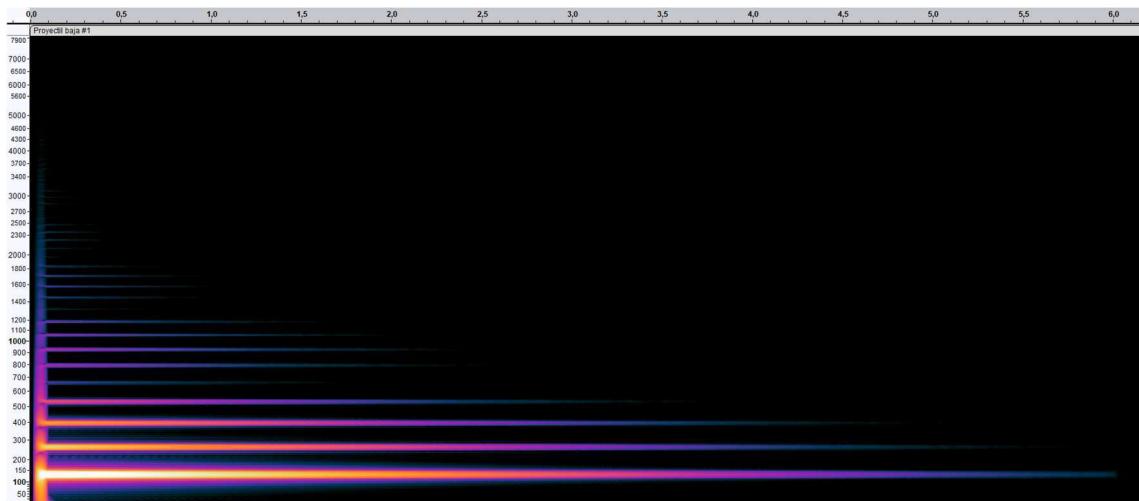


Figura 27. Espectrograma de una nota C3 con una excitación con forma de trayectoria de proyectil 1.

En el primer caso, con el punto de máxima velocidad cercano al 70% de la cuerda se obtiene un espectro con menor contenido de armónicos a alta frecuencia.

Al desplazar el punto de máxima velocidad al 90% de la cuerda se consigue un espectro más rico en armónicos.

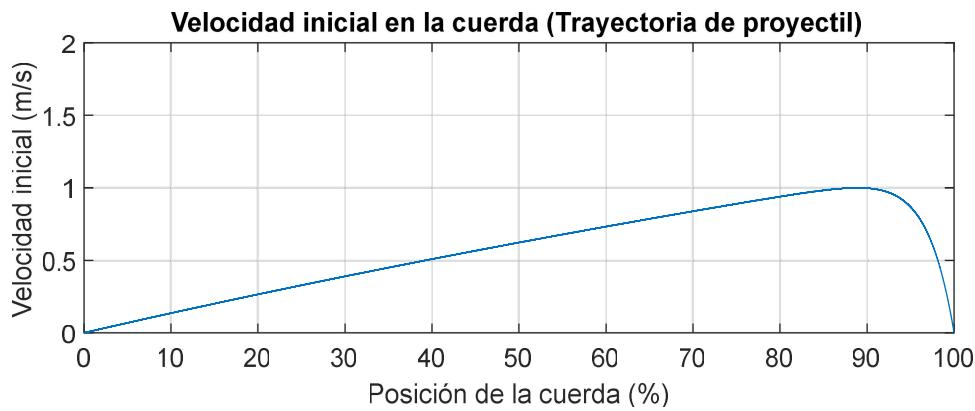


Figura 28. Velocidad inicial con forma de trayectoria de proyectil 2.

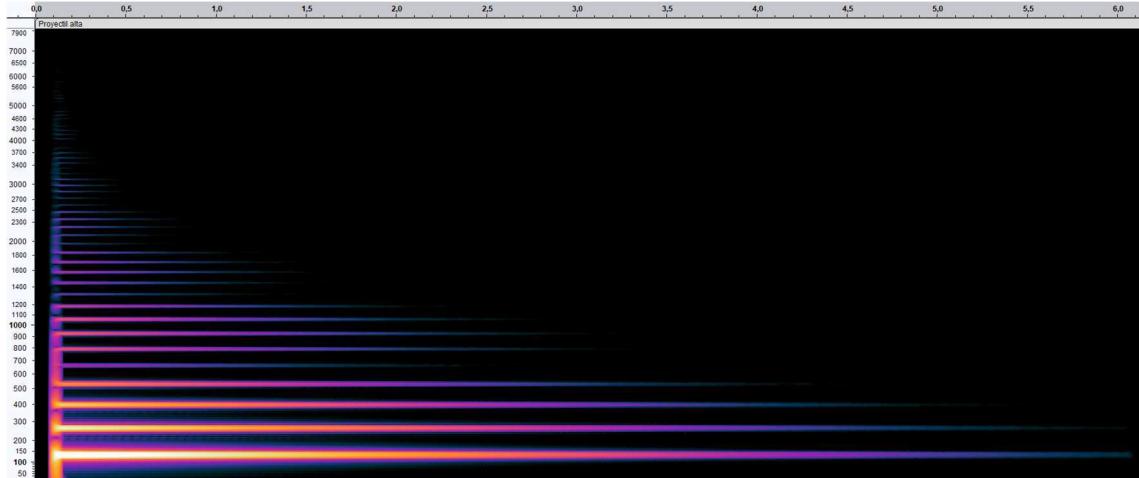


Figura 29. Espectrograma de una nota C3 con una excitación con forma de trayectoria de proyectil 2.

En la versión final del plugin se ha utilizado una combinación ponderada de las funciones de las figuras Figura 26 y Figura 28 según la velocidad de las notas MIDI recibidas.

Para aquellas notas que se reciben con velocidad MIDI mayor se utiliza la función de la figura 26, que produce una nota con mayor contenido espectral y ataque más pronunciado al comenzar a sonar la nota. Para aquellas con menor velocidad se utiliza la función de la figura 24, que produce menos armónicos y un sonido más suave. Para todos los casos intermedios se utiliza una suma ponderada de las dos funciones según la velocidad MIDI recibida.

Este proceso se realiza en el Código 2, donde `vAlta` es la excitación de la figura 26 y `vBaja`, la de la figura 24.

```
for (int x = 1; x < X; x++) {
    // Forma para velocity altas -> sonido más percusivo
    v0[x] = velocity * vAlta;
    // forma para velocity baja -> sonido más suave
    v0[x] = v0[x] + (1 - velocity) * vBaja;
}
```

Código 2. Ponderación de las dos excitaciones de la cuerda según la velocidad MIDI.

3.6. Corrección de la amplitud

La amplitud de la vibración de la cuerda (y por tanto el volumen de la salida de audio del plugin) varía en función de la longitud de la cuerda, la frecuencia de la oscilación y la velocidad inicial de la cuerda y se puede obtener mediante el método de Fourier [21]:

$$b_n = \frac{2}{L\omega_n} \int_0^L v_0 \sin(k_n x) dx \quad (47)$$

Para corregir esta desviación y conseguir que todas las notas tengan el mismo volumen se calcula para la excitación inicial empleada en cada caso la integral en toda la cuerda; empleando para ello integración discreta:

$$\int_0^L v_0 \sin(k_n x) dx \cong \sum_0^L v_0 \sin(k_n x) \Delta x \quad (48)$$

La expresión para el cálculo de la amplitud a partir de la velocidad inicial en cada punto discreto de la cuerda queda como:

$$b_n = \frac{2}{L\omega_n} \sum_0^L v_0 \sin(k_n x) \Delta x \quad (49)$$

Una vez calculada la amplitud b_n , según (49), se divide la velocidad inicial por este valor de forma que todas las notas tengan el mismo volumen. Este proceso de normalización de la amplitud se puede trasladar a código en C++ como se muestra en el Código 3.

```
// Se inicializa la integral con valor 0
float integral = 0;

// Se recorren todas las posiciones de la cuerda sumando la velocidad
// inicial en ese punto ponderado por sen(kx) a la integral
for (int x = 0; x < X; x++) {
    integral = integral + v0[x] * sin(kn * x * dx) * dx;
}

// Se calcula la amplitud de la vibración resultante de esa velocidad
// inicial en la cuerda
b_n = (2 * integral / (L * 2 * float_Pi * frequency));

// Se normaliza la velocidad inicial en la cuerda
for (int x = 0; x < X; x++) {
    v0[x] = v0[x] / b_n;
}
```

Código 3. Corrección de la amplitud de la vibración de la cuerda.

Una vez normalizada la amplitud, la velocidad inicial en la cuerda se multiplica por la velocidad de la nota MIDI recibida que JUCE proporciona como un valor de tipo *float* entre 0 y 1.

3.7. Pérdidas

A partir del tiempo de decaimiento, T_{60} , a dos frecuencias cualesquiera se pueden calcular los parámetros σ_0 y σ_1 que definen las pérdidas lineales y las dependientes de la frecuencia respectivamente.

Para ello se utilizan las ecuaciones (50) y (51), tomadas de [4].

$$\sigma_0 = \frac{6 \ln(10)}{\xi(\omega_2) - \xi(\omega_1)} \left(\frac{\xi(\omega_2)}{T_{60}(\omega_1)} - \frac{\xi(\omega_1)}{T_{60}(\omega_2)} \right) \quad (50)$$

$$\sigma_1 = \frac{6 \ln(10)}{\xi(\omega_2) - \xi(\omega_1)} \left(-\frac{1}{T_{60}(\omega_1)} + \frac{1}{T_{60}(\omega_2)} \right) \quad (51)$$

Se modifica el valor de los T_{60} hasta la consecución de un espectrograma similar al que se obtiene del instrumento real. Concretamente se utiliza un T_{60} de 9 segundos a 200 Hz y uno de 4 segundos a 10 kHz.

3.8. Detector de nivel

Dado que el movimiento de la cuerda tiene una amplitud exponencial negativa, nunca alcanza el cero hasta que se supera la precisión de las variables utilizadas (float). Mientras la voz que está reproduciendo la nota no haya acabado de reproducirla, no se libera para que se pueda reproducir una nueva nota en ella.

Para liberar la voz tan pronto como deje de escuchar, se ha implementado un detector de nivel RMS, que toma como entrada el punto de lectura de la cuerda y detecta cuándo la vibración cae por debajo de cierto umbral. El detector de nivel tiene la forma de la ecuación (52).

$$c_n^2 = \alpha x(n)^2 + (1 - \alpha)c_{n-1}^2 \quad (52)$$

Se elige un parámetro α , tal que el tiempo de integración del detector de nivel sea de 1 ms, que se ha comprobado experimentalmente que consigue una respuesta rápida del detector tanto al mantener la nota pulsada como al liberarla.

$$\alpha = \frac{1000}{1(ms)} \frac{1}{f_s} \quad (53)$$

Se ha elegido un umbral de -90 dBFS, que se ha comprobado que es suficiente para que no se perciba el corte al dejar de reproducir la nota en esa voz.

$$-90 \text{ dBFS} = 20 \log \left(\frac{\text{valor}}{\text{valor fondo de escala}} \right) \quad (54)$$

Dado que el valor a fondo de escala en el entorno de trabajo es igual a 1, se obtiene de la ecuación (54) un valor de $10^{-4.5}$ que se ha aproximado a 5×10^{-5} . El código referente al detector de nivel se ha incluido en el Código 4.

```
// Para cada muestra del buffer
for (int s = 0; s < synthBuffer.getNumSamples(); s++) {
    // Se calcula el desplazamiento de la cuerda y
    // se añade al buffer (omitido)
    {...}
    // Se introduce la muestra actual en el detector de
    // nivel RMS (c2n)
    c2n = alfa * powf(y[xRead], 2) + (1 - alfa) * c2n;

    // Si el nivel está por debajo del umbral (-90dbFS) se apaga la
    // nota
    if (c2n < 0.00005f) {
        {...}
        clearCurrentNote();
        return;
    }
    {...}
}
```

Código 4. Detector de nivel.

3.9. Gestión de la entrada MIDI

El plugin VST recibe mensajes MIDI desde el DAW huésped. Se ha programado el instrumento virtual para reaccionar únicamente a mensajes Note On y Note Off, que contiene la información relativa a la altura y la velocidad de las notas. El resto de los mensajes MIDI (*Aftertouch*, *pitch bend*, cambio de programa, etc. [18]) se ignoran.

Al recibirse un mensaje Note On el sintetizador busca una voz libre y ejecuta en ella la función `startNote()`, que se ha programado para llamar a la función `setInitialConditions()`, que establece las características y la velocidad inicial de la cuerda. Para ello se extraen del mensaje MIDI la frecuencia (en hercios), utilizando la función `getMidiNoteInHertz()` incluida en la librería JUCE [22] y la velocidad de la nota (normalizada entre 0 y 1), que la clase proporciona como argumento a la función.

```

void SynthVoice::startNote(int midiNoteNumber, float velocity,
juce::SynthesiserSound* sound, int currentPitchWheelPosition) {

    float frequency = getMidiNoteInHertz(midiNoteNumber);

    // Se comprueba que la nota está en el rango de notas que pueden
    // ejecutarse en el Harpejji (C2 - C6)
    if (frequency > 65.40f && frequency < 1047)
        setInitialConditions(velocity, frequency);
    else
        clearCurrentNote();
}

```

Código 5. Gestión de los mensajes MIDI.

3.10. Control de tono

Se ha incluido en el plugin un control de tono como el del instrumento real. Dado que no se ha tenido acceso a un Harpejji “real” y no existe documentación a este respecto se ha supuesto que la circuitería es idéntica a la del control de tono de una guitarra eléctrica. Normalmente consiste en un filtro paso bajo RC, de primer orden, formado por un condensador y una resistencia variable que regula la frecuencia de corte del filtro. El esquema del control de tono se incluye en la Figura 30.

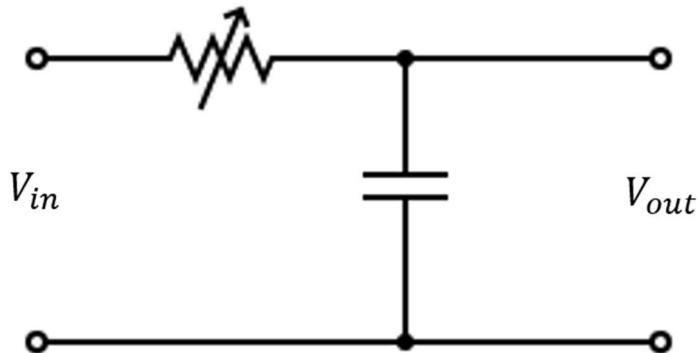


Figura 30. Filtro RC paso bajo de primer orden.

Para su implementación en el programa se ha utilizado un filtro paso bajo IIR predefinido de la librería DSP de JUCE [23]. El filtro se inicializa con frecuencia de corte a 20 kHz y se permite al usuario controlar este parámetro desde la interfaz gráfica, pudiendo variarse la frecuencia de corte de 20 kHz a 20 Hz. Este tipo de filtros presenta una respuesta en frecuencia como la de la Figura 31, tomada de [24], que produce una caída de 3 dB por octava desde la frecuencia de corte del filtro.

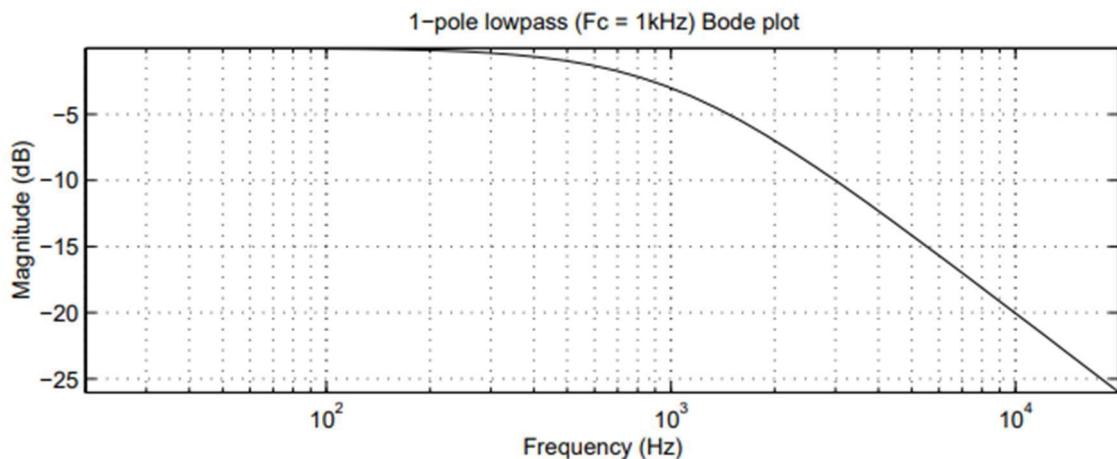


Figura 31. Respuesta en frecuencia del filtro paso bajo de primer orden.

3.11. Control de ganancia

A la señal de salida del filtro se le aplica una ganancia según el valor del control rotatorio de la interfaz gráfica. Se utiliza para ello también una estructura de la librería DSP de JUCE que permite aplicar una ganancia tanto en dB como en unidades lineales a una muestra o a un bloque de audio completo.

3.12. Interfaz gráfica de usuario

Se ha diseñado una interfaz gráfica que muestra al usuario qué notas y en qué posiciones se están reproduciendo en cada momento y que permite el control de los parámetros del instrumento mediante cuatro potenciómetros virtuales rotatorios que posibilitan al usuario controlar la tensión de las cuerdas, el tiempo de decaimiento, la frecuencia de corte del filtro paso bajo y la ganancia de salida del plugin respectivamente. Los controles del plugin se muestran en la Figura 32.

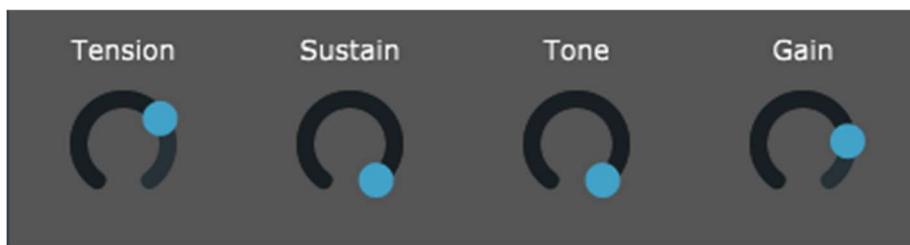


Figura 32. Controles rotatorios de la interfaz del plugin.

Los cuatro controles rotatorios se crean, utilizando la clase Slider de Juce y se dibujan sobre la ventana del plugin. Se establece en cada potenciómetro sus márgenes relativos al área del plugin de forma que al redimensionarse la ventana del plugin se modifique el tamaño de los potenciómetros consecuentemente. Así, cada potenciómetro ocupa un sexto

del ancho de la ventana, tanto de alto como de ancho. Se establece la posición del primer potenciómetro como un octavo del ancho de la ventana menos la mitad del ancho del potenciómetro y el resto se desplazan por una cantidad igual al ancho de la ventana entre 4. Se ha incluido el código relativo a este proceso en el Código 6.

```
void SynthAudioProcessorEditor::resized()
{
    const auto bounds = getLocalBounds();
    const auto sliderWidth = bounds.getWidth() / 6;
    const auto sliderHeight = bounds.getHeight() / 6;
    const auto sliderStartX = bounds.getWidth() / 8 - (sliderWidth / 2);
    const auto sliderStartY = 15 + 92 / 2 - (sliderHeight / 2);

    tensionSlider.setBounds(sliderStartX, sliderStartY, sliderWidth,
    sliderHeight);
    sustainSlider.setBounds(sliderStartX + bounds.getWidth() / 4,
    sliderStartY, sliderWidth, sliderHeight);
    toneSlider.setBounds(sliderStartX + 2 * bounds.getWidth() / 4,
    sliderStartY, sliderWidth, sliderHeight);
    gainSlider.setBounds(sliderStartX + 3 * bounds.getWidth() / 4,
    sliderStartY, sliderWidth, sliderHeight);
}
```

Código 6. Posiciones de los controles rotatorios.

La imagen de fondo del plugin se ha diseñado utilizando el software GIMP, de forma que represente la superficie del Harpejji sobre la que se tocan las notas. Se ha representado a escala la distancia entre los trastes. Para ello se ha obtenido de [25] la fórmula para la distancia, d , entre el traste número n y la cejuela. La imagen generada se incluye en la Figura 33.

$$d = s \cdot (1 - e^{-kn}) \quad (55)$$

Donde s es la longitud de escala y $k = 0.05716$

Se toma como longitud de escala la altura de la imagen en píxeles y se calcula la posición de los 19 trastes con la fórmula (55). La imagen generada de la Figura 33 se introduce en el proyecto de JUCE, que permite cargarla como fondo del plugin.

3. Desarrollo del plugin

Sobre esta superficie se dibujan, en ejecución, las 16 cuerdas del instrumento, así como los controles rotatorios para los cuatro parámetros tal como aparecen en la Figura 34, utilizando las clases Path y Graphics de JUCE ([26] y [27]), que permiten dibujar líneas entre dos puntos de la ventana del plugin.

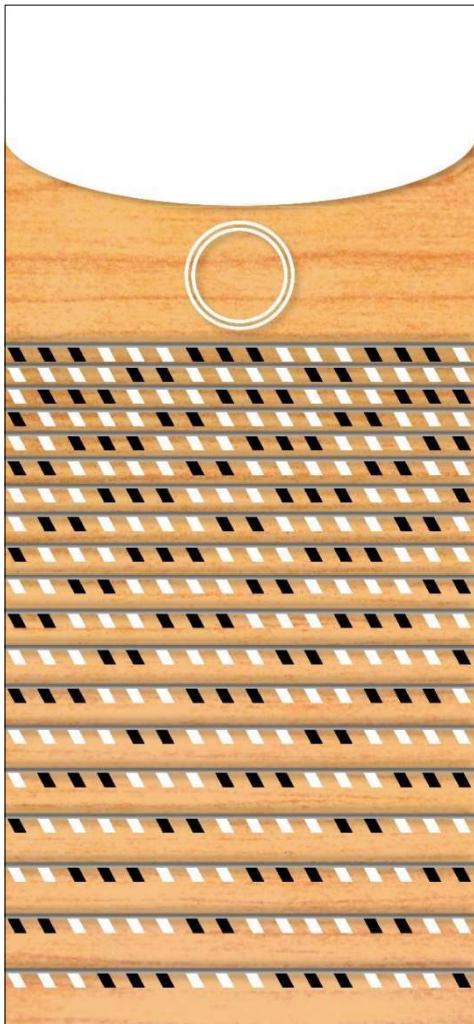


Figura 33. Fondo del plugin.

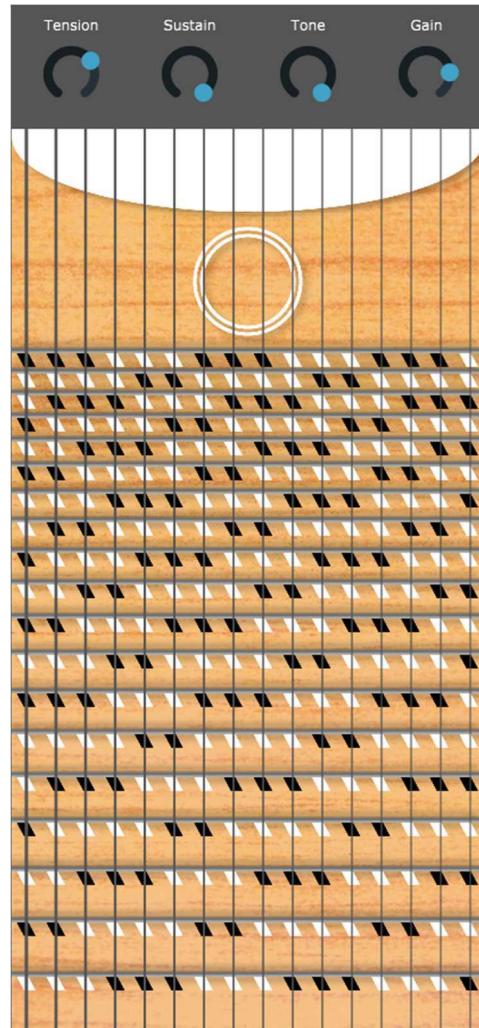


Figura 34. Interfaz de usuario del plugin.

Al pulsarse una nota se dibuja un círculo negro sobre la posición del instrumento en la que se colocaría el dedo en el instrumento real. Además, se representa la vibración de la cuerda que se pasa como un vector a la clase PluginEditor. La vibración de la cuerda se escala utilizando de nuevo la ecuación (55) para representarla desde el traste correspondiente hacia arriba hasta el extremo superior de la ventana.

Se incluye en la Figura 35 una imagen del plugin reproduciendo varias notas. La vibración de las cuerdas no se refleja bien, al tratarse de una imagen estática.

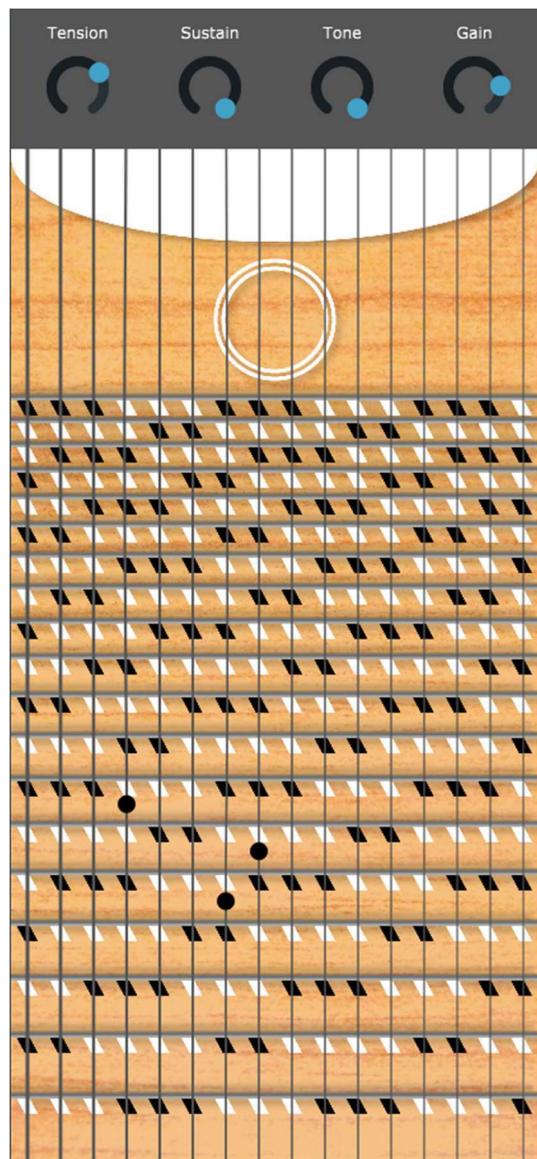


Figura 35. Interfaz del plugin reproduciendo varias notas.

4. Resultados

En este capítulo se analizan los resultados del proyecto: se compara el sonido del plugin con el del instrumento real, se explican los resultados obtenidos en cuanto a funcionamiento y rendimiento del programa y se comprueba cuáles de los objetivos establecidos para el proyecto se han cumplido.

4.1. Validación del modelo

Se utiliza el espectro máximo y el spectrograma del sonido del plugin y de grabaciones reales del Harpejji para analizar la similitud conseguida entre el plugin y el instrumento real, tomada de [28]. En las figuras Figura 36 y Figura 37 se incluyen los máximos del espectro al reproducir una nota C3 en el instrumento real y en el plugin respectivamente.

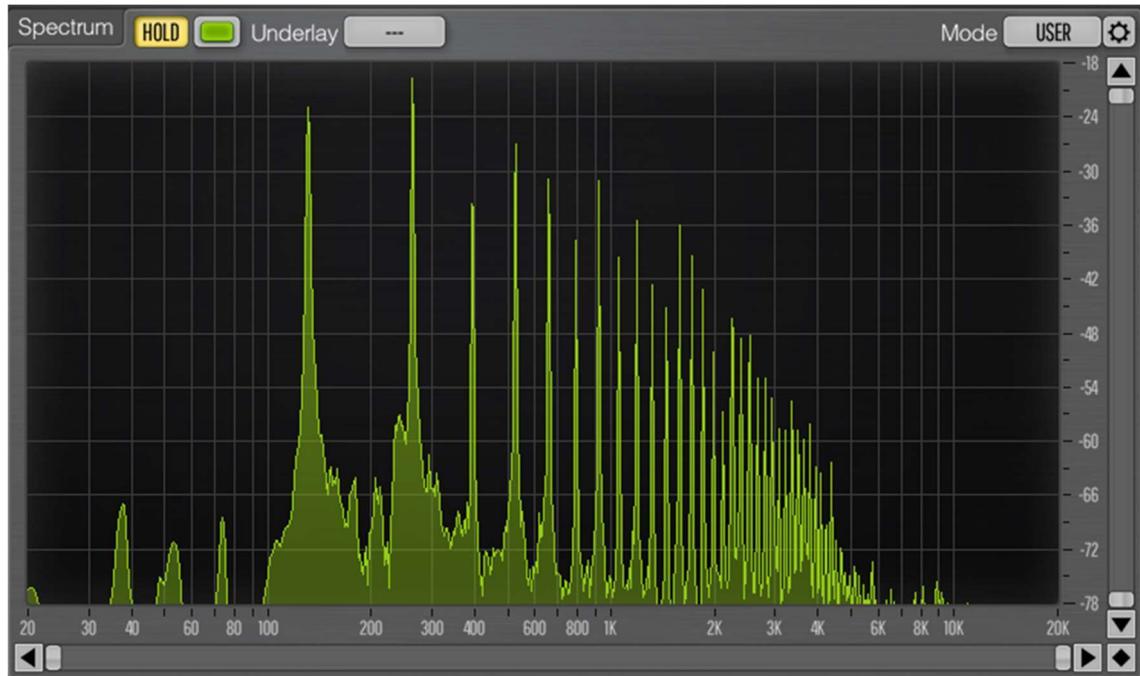


Figura 36. Espectro máximo de una nota C3 tocada en un Harpejji real.

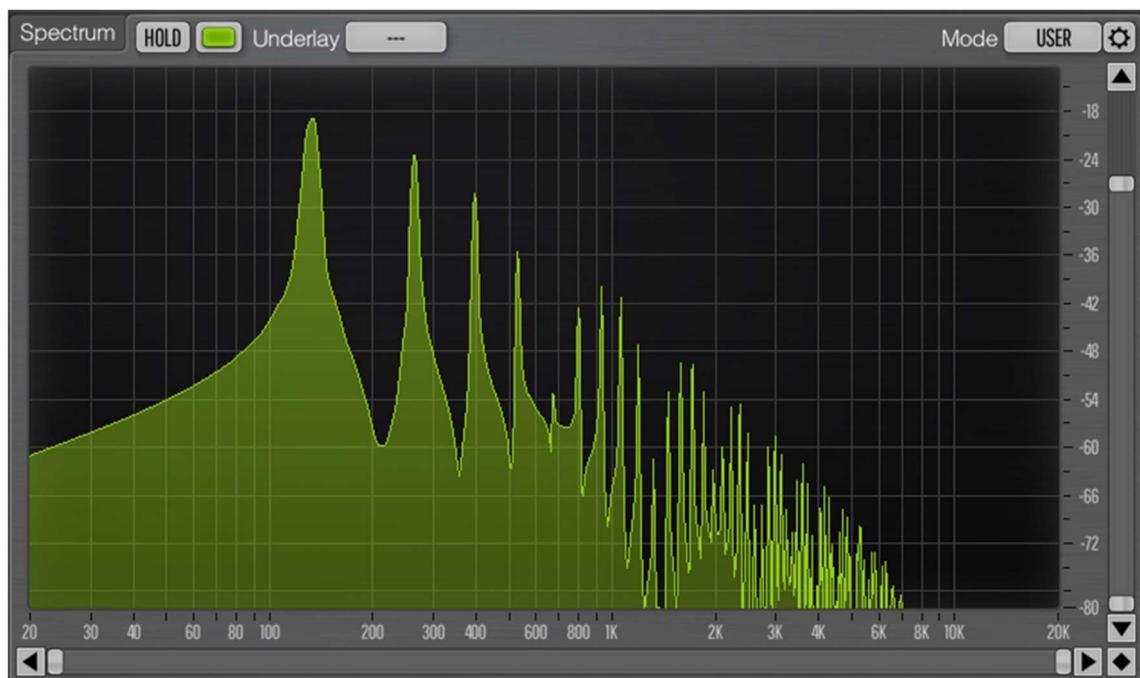


Figura 37. Espectro máximo del plugin reproduciendo una nota C3.

Se observa que en general se consigue bastante similitud en cuanto a densidad de armónicos y los armónicos aparecen exactamente a las mismas frecuencias en el plugin y en el instrumento real. Algunas frecuencias aparecen atenuadas en el espectro del plugin debido al punto de lectura elegido en la cuerda y la forma de la excitación inicial utilizada.

Se comparan a continuación los espectrogramas de las dos fuentes. En este caso tocando una nota B3, ya que es la nota aislada más larga que ha sido posible encontrar; tomada también de [28]. Se incluyen los espectrogramas en las figuras Figura 38 y Figura 39.

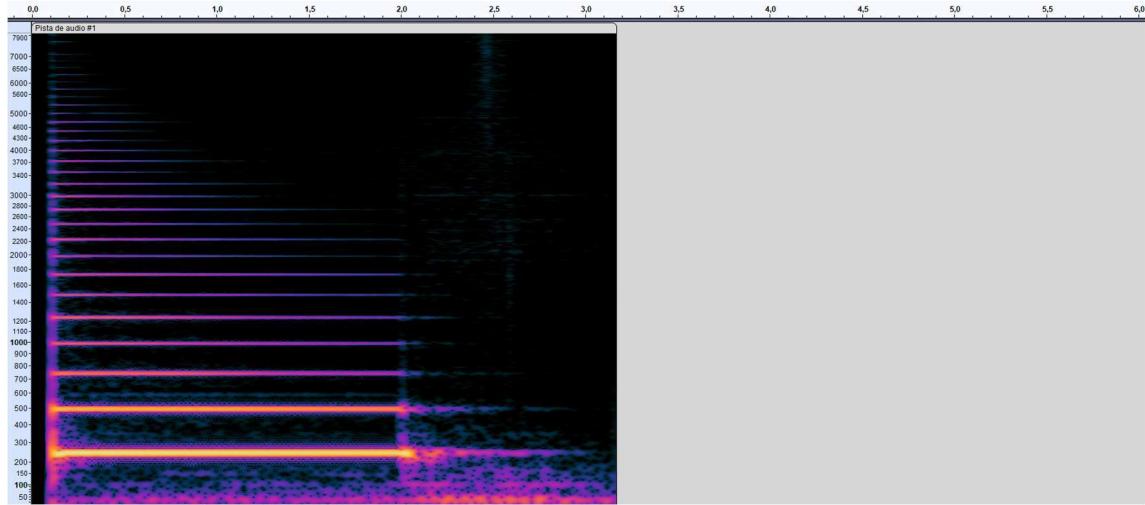


Figura 38. Espectrograma de una nota B3 reproducida en el instrumento real.

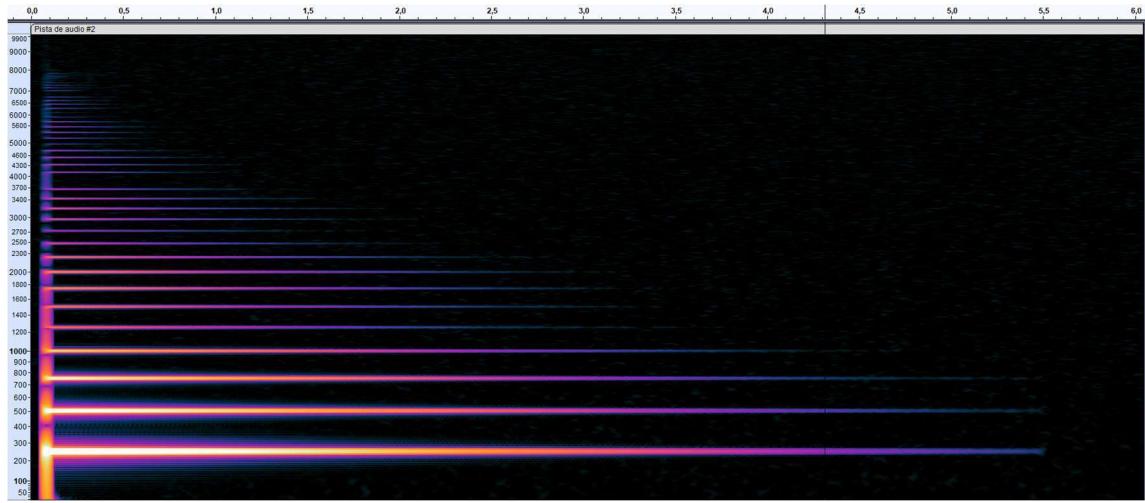


Figura 39. Espectrograma de una nota B3 reproducida en el plugin.

Se observa de nuevo la atenuación que aparece en algunos de los armónicos. Tanto en los espectros como en los espectrogramas se observa que los armónicos a altas frecuencias aparecen con menor amplitud en el instrumento virtual que en el real. El tiempo de decaimiento se aproxima mucho al del instrumento real a todas las frecuencias.

Se comprueba también el sonido del instrumento mediante su escucha y se compara con el sonido del instrumento real observándose gran similitud entre ambos.

4.2. Funcionamiento del plugin

En la versión final del plugin se han utilizado seis voces para el sintetizador, siendo este número el máximo número de voces posibles sin que aparezcan problemas de clics y cortes de audio debido al elevado número de operaciones que el procesador tiene que realizar simultáneamente.

Los controles rotatorios no producen cortes ni clics en el audio, pudiéndose manipular en tiempo real mientras se reproduce sonido en el plugin. Los controles de los parámetros relativos al modelado físico de la cuerda no tienen efecto sobre notas que ya se están ejecutando, sino que tienen efecto al “generar” una nueva cuerda al recibir una nueva nota MIDI.

Se ha prescindido del cálculo del término de la rigidez para poder introducir un número mayor de voces sin que se produzcan errores de audio. Se han realizado pruebas y la eliminación de este término no tiene ningún impacto en el sonido en la mayoría de los casos, a excepción de aquellos casos extremos, con tensión muy baja o masa lineal muy alta de la cuerda.

La interfaz gráfica muestra la vibración de las cuerdas en tiempo real e indica con un punto aquellas posiciones en las que se está pisando la cuerda.

La salida de audio es ajustable para evitar saturación al pasar la señal del plugin al DAW.

4.3. Evaluación de los objetivos

Una vez completado el desarrollo del proyecto se considera cuáles de aquellos objetivos y restricciones de diseño detalladas en 1.5 se ha conseguido satisfacer.

Aunque ha sido necesario tener acceso a un equipo con sistema operativo MacOS y uno con Windows para generar el plugin VST3 compatible con cada sistema se ha conseguido que el plugin funcione de manera idéntica en ambos sistemas.

Como se ha detallado en el capítulo 4.1, se ha conseguido emular consiguiendo una buena similitud el sonido del Harpejji, aunque existen mejoras que todavía se pueden incluir en el plugin para conseguir aproximar su sonido aún más al instrumento real y que se explican en el apartado 6.2.

El plugin responde a los mensajes de entrada MIDI que le proporciona el DAW, y reproduce las notas correspondientes en tiempo real, sin latencia apreciable por el usuario y sin cortes de audio si se configura correctamente el tamaño de buffer del adaptador de audio del equipo. La señal de salida del plugin es monofónica tal como se especificaba en las restricciones de diseño.

Según la velocidad con la que se toquen o se introduzcan en el plugin las notas MIDI el instrumento varía tanto el volumen de salida como el contenido armónico de los sonidos que se reproducen a su salida.

Se permite al usuario controlar los parámetros físicos de las cuerdas del instrumento (tensión de las cuerdas y el tiempo de decaimiento de las cuerdas, relacionado con el amortiguamiento del aire y de la propia cuerda) además de controlar el volumen de salida del plugin y el control de tono del instrumento.

En cuanto a la polifonía del instrumento se ha elegido un número de seis voces como máximo número de voces sin encontrar problemas de cortes de audio provocados por el elevado número de operaciones que no da tiempo a ejecutar en el tiempo entre que se lee un *frame* a la salida del plugin y el siguiente.

Para conseguir reducir la carga de operaciones y así aumentar el número de voces que se pueden procesar se ha eliminado el término de la rigidez de la cuerda de la ecuación de onda, que se explica en detalle en los apartados 2.5 y 3.4.6. Este término tiene un impacto despreciable en el sonido del instrumento para los casos “normales”, dentro de las características físicas reales de las cuerdas, pero se había incluido para aportar realismo a los casos extremos, en los que la cuerda se comporta más como una barra rígida. En la versión final del plugin ha sido necesario eliminarlo.

Se ha conseguido, por lo tanto, satisfacer todos los objetivos establecidos en la fase de anteproyecto. Aun así, existen mejoras que se pueden realizar en el plugin para conseguir un mejor sonido, funcionamiento e interfaz y que se explican en el apartado 6.2.

5. Presupuesto

Se incluyen en este apartado los gastos derivados del desarrollo del proyecto divididos entre mano de obra y materiales.

5.1. Mano de obra

Concepto	Unidad de medida	Cantidad de Unidades	Precio Unitario	Importe
Ingeniería	Horas	300	€ 20,00	€ 6.000,00
			Total	€ 6.000,00

Total mano de obra: SEIS MIL EUROS

5.2. Material

Concepto	Precio	Valor residual	Vida útil (años)	Amortización
Ordenador	€ 750,00	€ 100,00	10	€ 65,00
Periféricos	€ 250,00	€ 0,00	5	€ 50,00
Teclado MIDI	€ 100,00	€ 25,00	5	€ 15,00
Licencia DAW Cubase	€ 300,00	€ 0,00	10	€ 30,00
Licencia JUCE no comercial	€ -	€ -	-	€ -
Licencia Microsoft Visual Studio	€ -	€ -	-	€ -
		Total amortización		€ 160,00

Concepto	Cantidad de Unidades	Precio Unitario	Importe
Juego de cuerdas de Harpejji	1	€ 50,00	€ 50,00
Báscula de precisión	1	€ 30,00	€ 20,00
	Total compras		€ 80,00

Total recursos materiales: DOSCIENTOS CUARENTA EUROS

5.3. Total

Concepto	Importe
Mano de obra	€ 6.000,00
Material	€ 250,00
Total	€ 6.250,00

Total: SEIS MIL DOSCIENTOS CUARENTA EUROS

6. Conclusiones

En este capítulo se exponen las conclusiones obtenidas tras el desarrollo del proyecto, así como las dificultades encontradas y el trabajo futuro que se podría realizar para mejorar el sintetizador. Se aclaran también las aportaciones realizadas por el autor.

6.1. Dificultades

Se menciona en este apartado las dificultades que se han encontrado tanto en la fase de anteproyecto como en el desarrollo del proyecto y cómo se han superado.

Para modelar fielmente el instrumento es necesario conocer las características físicas. En primera instancia se recurrió al fabricante del instrumento, al que se le solicitó información acerca de la longitud de escala del instrumento, así como el calibre de las cuerdas que utilizaba. Tras explicar el objetivo del proyecto y asegurar que se trata de un proyecto académico y no comercial se recibe la negativa del fabricante a aportar ninguno de estos datos.

Dadas las circunstancias ha sido necesario estimar la longitud de escala del instrumento en 27" o 68,58 cm, basándose en imágenes del instrumento. En cuanto al calibre de las cuerdas se obtuvieron de una publicación en el perfil en redes sociales de un intérprete de Harpejji [29], que se incluye en la Figura 40.



Figura 40. Publicación en la red social Instagram en la que se muestra un juego completo de cuerdas de Harpejji.

Tras conseguir la información sobre el calibre de las cuerdas se realizó un pedido de cuerdas individuales de los calibres correspondientes sobre los que se realizaron las medidas incluidas en la Tabla 1.

Otra dificultad que se ha encontrado se relaciona con la forma de la velocidad en la cuerda para este tipo de instrumento. Aunque existe extensa bibliografía sobre la excitación para instrumentos de cuerda pulsada, percutida o frotada, el Harpejji no se puede clasificar en ninguna de estas categorías. El método de ejecución, pisando las cuerdas directamente

sobre los trastes, sin pulsarlas, provoca una excitación inicial en la cuerda en la que el desplazamiento inicial es nulo, pero existe velocidad inicial no nula en toda la cuerda.

Ha sido necesario, por tanto, inferir la forma de la velocidad inicial en la cuerda, probando varias excitaciones hasta dar con una que produce un sonido y un espectrograma similares al del instrumento real.

La última dificultad que se ha planteado es relativa al lenguaje de programación utilizado, C++. Durante el grado se enseña programación en C, Java y MATLAB. C++ tiene muchas similitudes con C, pero ha sido necesario familiarizarse con la forma de trabajar en C++, así como con los métodos y estructuras propios de la librería JUCE.

Para resolver esta dificultad han sido muy útiles, en especial, los conocimientos obtenidos de la asignatura Ingeniería de Audio III; en la que se ha trabajado con procesadores de audio en tiempo real, utilizando MATLAB. Se debe mencionar también en relación a esta dificultad a The Audio Programmer, en YouTube [30] y su serie de vídeos en los que explica cómo crear un sintetizador básico usando JUCE, así como los proyectos de ejemplo de JUCE [31], que ayudaron mucho al comienzo del desarrollo del plugin.

6.2. Trabajo futuro

Como se ha mencionado anteriormente existen muchos aspectos del plugin en los que se pueden realizar mejoras. Se proponen algunas de ellas a continuación.

- Excitación de la cuerda: dado que la velocidad inicial en la cuerda se ha tenido que inferir, cabe esperar que existan diferencias entre la función de velocidad inicial propuesta y la excitación real que ocurre al pisar las cuerdas del Harpejji. Si se tuviera acceso al instrumento real se podría medir esta excitación e incluirla en el plugin de forma que el sonido producido fuera aún más similar al del instrumento real.
- Tiempos de caída según la frecuencia: tampoco ha sido posible encontrar una nota completa de Harpejji aislada de la que se puedan obtener los tiempos de decaimiento a todas las frecuencias. Con grabaciones de notas completas del instrumento real se podría establecer los tiempos de caída fielmente a como ocurren en el instrumento real.
- *Slides*: una técnica habitual al tocar el Harpejji consiste en la realización de *slides* o deslizamientos entre dos notas de la misma cuerda. No existe una forma fácil de implementar este comportamiento mediante modelado físico ya que para ello habría que modificar la longitud de la cuerda en la que se está reproduciendo una nota mientras se calcula en ella el desplazamiento en las muestras siguientes. Cambiar la longitud de la cuerda daría lugar a problemas de estabilidad. Una opción para implementar esta técnica podría ser mediante el uso de un efecto de *pitch-bend*, que modifica la altura del sonido de entrada.

- Función de transferencia de la pastilla piezoeléctrica: en el Harpejji, la vibración de la cuerda se recoge mediante una pastilla piezoeléctrica situada en uno de los extremos de la cuerda. Para reproducir fielmente el sonido del instrumento sería necesario filtrar la señal de salida de las cuerdas del plugin utilizando para ello la respuesta en frecuencia de las pastillas piezoeléctricas.
- Mejora del algoritmo para decidir en qué cuerda/traste se toca una nota: en la versión final del plugin se ha utilizado un algoritmo muy sencillo para determinar en qué posición se toca una nota. Esto afecta tanto al sonido, ya que cada cuerda tiene características distintas, como a la interfaz gráfica que representa en qué posición se está tocando la nota. Con el algoritmo actual es posible tocar dos notas distintas en la misma cuerda, aunque sólo una de ellas se representa en la interfaz gráfica. Un algoritmo más sofisticado para decidir en qué posición se toca la nota de entrada sería una mejora sencilla del plugin que no se ha implementado por falta de tiempo.
- Mejora de la estabilidad del plugin: se conocen algunos *bugs* relacionados con la interfaz gráfica que provocan que el plugin se detenga. No ha sido posible solucionarlos y es una mejora que debería introducirse para dar por terminado el plugin.

6.3. Aportaciones

Se ha conseguido implementar un instrumento virtual de Harpejji utilizando síntesis por modelado físico, que es un producto que no existe en el mercado. Si se realizasen las mejoras mencionadas en 6.2 se podría considerar su venta como producto comercial.

Ha sido necesario inferir por completo la excitación de la cuerda basándose en el método de ejecución, ya que no existe bibliografía al respecto de síntesis por modelado físico de este instrumento ni de otros con una técnica de ejecución similar.

Aunque existen otros ejemplos en la bibliografía el código relativo a la solución de la ecuación de onda mediante diferencias finitas es original.

7. Referencias

- [1] «Harpejji Model Comparison», *Marcodi Musical Products*.
<https://www.marcodi.com/pages/harpejji-model-comparison> (accedido 30 de abril de 2022).
- [2] «Harpejji», *Wikipedia, la enciclopedia libre*. 27 de agosto de 2020. Accedido: 30 de abril de 2022. [En línea]. Disponible en:
<https://es.wikipedia.org/w/index.php?title=Harpejji&oldid=128799314>
- [3] «Síntesis aditiva», *Wikipedia, la enciclopedia libre*. 6 de enero de 2020. Accedido: 24 de mayo de 2022. [En línea]. Disponible en:
https://es.wikipedia.org/w/index.php?title=S%C3%ADntesis_aditiva&oldid=122555029
- [4] S. D. Bilbao, *Numerical sound synthesis: finite difference schemes and simulation in musical acoustics*. Chichester: Wiley, 2009.
- [5] «Síntesis substractiva», *Wikipedia, la enciclopedia libre*. 19 de febrero de 2020. Accedido: 24 de mayo de 2022. [En línea]. Disponible en:
https://es.wikipedia.org/w/index.php?title=S%C3%ADntesis_substractiva&oldid=123690639
- [6] «Síntesis por modulación de frecuencias», *Wikipedia, la enciclopedia libre*. 14 de agosto de 2021. Accedido: 24 de mayo de 2022. [En línea]. Disponible en:
https://es.wikipedia.org/w/index.php?title=S%C3%ADntesis_por_modulaci%C3%B3n_de_frecuencias&oldid=137669380
- [7] bryc, «Algorithms.md», *Gist*.
<https://gist.github.com/bryc/e997954473940ad97a825da4e7a496fa> (accedido 24 de mayo de 2022).
- [8] «JonDent - Exploring Electronic Music: Yamaha DX7 - Operators & Algorithms». <https://djjondent.blogspot.com/2019/10/yamaha-dx7-algorithms.html> (accedido 24 de mayo de 2022).
- [9] «Síntesis mediante tabla de ondas», *Wikipedia, la enciclopedia libre*. 13 de septiembre de 2021. Accedido: 24 de mayo de 2022. [En línea]. Disponible en:
https://es.wikipedia.org/w/index.php?title=S%C3%ADntesis_mediante_tabla_de_ondas&oldid=138322253
- [10] White Noises, *Granular Synthesis EXPLAINED*, (1 de junio de 2018). Accedido: 24 de mayo de 2022. [En línea Video]. Disponible en:
<https://www.youtube.com/watch?v=ftDLRYnRYZQ>
- [11] «Síntesis granular», *Wikipedia, la enciclopedia libre*. 6 de enero de 2022. Accedido: 24 de mayo de 2022. [En línea]. Disponible en:
https://es.wikipedia.org/w/index.php?title=S%C3%ADntesis_granular&oldid=140772965

7. Referencias

- [12] «Síntesis por modelado físico», *Wikipedia, la enciclopedia libre*. 3 de marzo de 2022. Accedido: 24 de mayo de 2022. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=S%C3%ADntesis_por_modelado_f%C3%A1sico&oldid=142040193
- [13] «Digital Waveguide Modeling Elements». https://ccrma.stanford.edu/~jos/pasp/Digital_Waveguide_Modeling_Elements.html (accedido 24 de mayo de 2022).
- [14] «AAS- Tech Talk - Physical modeling». <https://www.applied-acoustics.com/techtalk/physicalmodeling/> (accedido 24 de mayo de 2022).
- [15] N. H. Fletcher y T. D. Rossing, *The Physics of Musical Instruments*. New York, NY: Springer New York, 1998. doi: 10.1007/978-0-387-21603-4.
- [16] «Módulo de Young», *Wikipedia, la enciclopedia libre*. 26 de septiembre de 2021. Accedido: 25 de mayo de 2022. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=M%C3%B3dulo_de_Young&oldid=138590261
- [17] «VST (Virtual Studio Technology)», *ingeniatic*. <http://ingeniatic.euitt.upm.es/index.php/tecnologias/item/659-vst-virtual-studio-technology> (accedido 25 de mayo de 2022).
- [18] «JUCE: Synthesiser Class Reference». <https://docs.juce.com/master/classSynthesiser.html> (accedido 25 de mayo de 2022).
- [19] J. Walsh, «Finite-Difference and Finite-Element Methods of Approximation», *Proc. R. Soc. Lond. Ser. Math. Phys. Sci.*, vol. 323, n.º 1553, pp. 155-165, 1971.
- [20] «Projectile motion», *Wikipedia*. 31 de mayo de 2022. Accedido: 3 de junio de 2022. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Projectile_motion&oldid=1090776713
- [21] Physics Learning With Dr. Shaw, *Solution of wave equation for struck string | Struck String | Vibration of String | Part 4*, (29 de junio de 2020). Accedido: 6 de junio de 2022. [En línea Video]. Disponible en: <https://www.youtube.com/watch?v=FfIcsCAyg3E>
- [22] «Essentials of the MIDI protocol». <https://ccrma.stanford.edu/~craig/articles/linuxmidi/misc/essenmidi.html> (accedido 29 de mayo de 2022).
- [23] «JUCE: dsp::IIR::Coefficients< NumericType > Struct Template Reference». https://docs.juce.com/master/structdsp_1_1IIR_1_1Coefficients.html#a70e856c050b1fa38659b1b67469f567a (accedido 29 de mayo de 2022).
- [24] Max Kamenetsky, «Filtered Audio Demo». Accedido: 29 de mayo de 2022. [En línea]. Disponible en: https://web.stanford.edu/~boyd/ee102/conv_demo.pdf
- [25] R. M. Mottola, «Liutaio Mottola Lutherie Information Website», *Liutaio Mottola Lutherie Information Website*. <https://www.liutaiomottola.com/> (accedido 31 de mayo de 2022).

- [26] «JUCE: Path Class Reference». <https://docs.juce.com/master/classPath.html> (accedido 31 de mayo de 2022).
- [27] «JUCE: Graphics Class Reference». <https://docs.juce.com/master/classGraphics.html> (accedido 31 de mayo de 2022).
- [28] Harpejji by Marcodi, *Harpejji Playing Basics*, (17 de noviembre de 2020). Accedido: 5 de junio de 2022. [En línea Video]. Disponible en: <https://www.youtube.com/watch?v=AUnvq2zRU6k>
- [29] «Lance Hoeppner en Instagram: “Getting ready for a few shows this month. All cleaned up with new strings! #harpejjig16 #harpejji #fortwaynemusicscene #fortwaynemusic...”», *Instagram*. <https://www.instagram.com/p/CP3VCEJLCbh/> (accedido 8 de junio de 2022).
- [30] The Audio Programmer, *Let’s Build a Synth with Juce Part 0 - Oscillator*, (3 de diciembre de 2020). Accedido: 26 de marzo de 2022. [En línea Video]. Disponible en: <https://www.youtube.com/watch?v=ADG6Rsd3ekg>
- [31] «JUCE: Tutorial: Build a MIDI synthesiser». https://docs.juce.com/master/tutorial_synth_using_midi_input.html (accedido 26 de marzo de 2022).

8.Bibliografía

- [32] JUCE, *Martin Shuppius - Physical modelling of guitar strings (ADC’17)*, (20 de noviembre de 2017). Accedido: 26 de marzo de 2022. [En línea Video]. Disponible en: https://www.youtube.com/watch?v=sxt5rxF_PdI
- [33] E. Hayes, *BatSynth*. 2022. Accedido: 26 de marzo de 2022. [En línea]. Disponible en: <https://github.com/Emmet-Hayes/BatSynth>
- [34] J. W. S. Rayleigh, *The theory of sound. Vol. 1*, Repr. of 2. ed. rev. and enl. 1894., vol. 1. New York: Dover Publ, 1969.

Anexo: Guía de instalación

Para instalar el plugin en el equipo basta con copiar el archivo *HarpejVST.vst3* de la carpeta Mac o Windows (según el sistema operativo del equipo de destino) a la carpeta VST3 del equipo. Por defecto se encuentra en *C:\Archivos de programa\Common Files\VST3* en el caso de sistemas Windows y en *Library/Audio/Plug-ins/VST3* en MacOs.

Para realizar alguna modificación en el código del plugin es necesario tener instalado tanto el editor de JUCE, Projucer como un entorno de desarrollo (Normalmente Visual Studio en Windows y XCode en MacOs). Abriendo el archivo *Synth.jucer* con Projucer y a continuación pulsando en el símbolo de Visual Studio o de Xcode (Save and open on IDE) se abre el proyecto en el entorno de desarrollo elegido desde el que se puede compilar dando lugar a un nuevo archivo .vst3 que se puede introducir en la carpeta VST3 del equipo para que el DAW lo reconozca automáticamente.

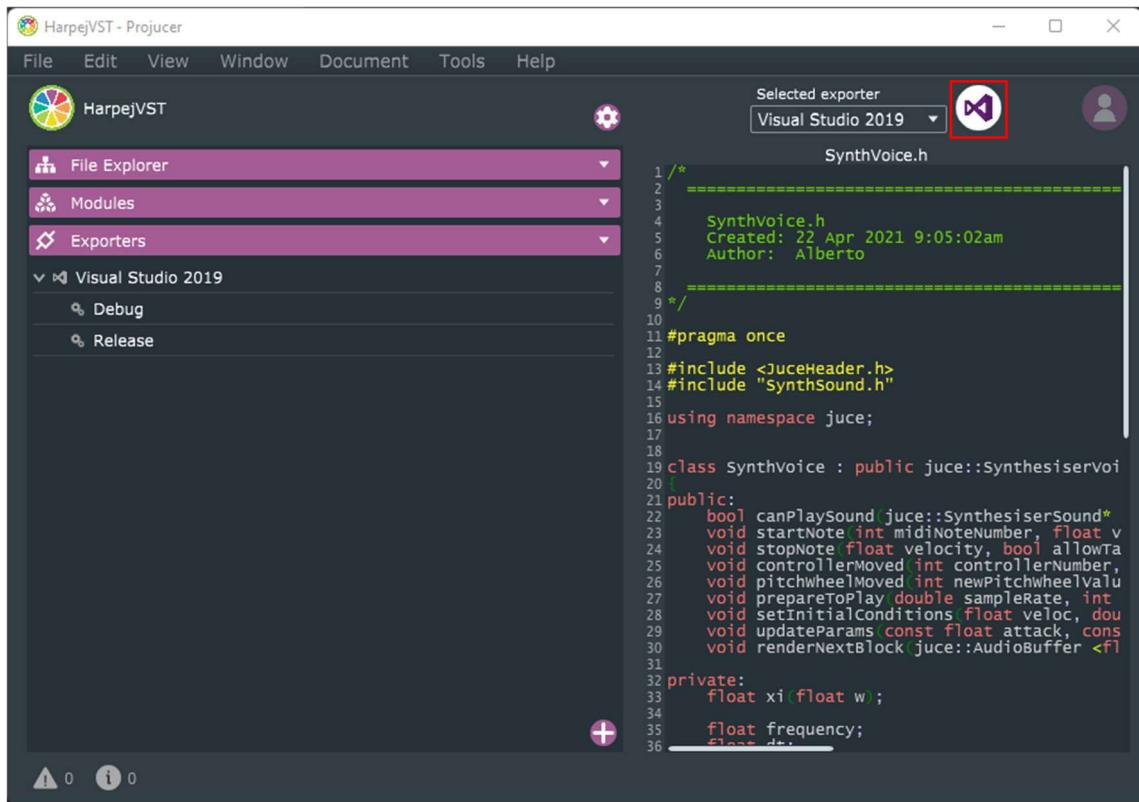


Figura 41. Ventana de Projucer con el proyecto abierto. Recuadrado en rojo el botón para abrir el proyecto en el entorno de desarrollo Visual Studio.

Se debe tener en cuenta que los archivos .vst3 generados en Xcode desde MacOs solamente funcionarán en equipos con este mismo sistema operativo. Lo mismo ocurre con Visual Studio y Windows.