UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Programa de Maestría y Doctorado en Música
Campo en Tecnología Musical

Espacialización Multicanal Algorítmica: Un modelo teórico con implementaciones en Supercollider

Trabajo de Tesis que para optar por
el grado de Maestro en Música
presenta

Edmar Olivares Soria

Tutor: **Dr. Roberto Morales Manzanares**
Programa de Maestría y Doctorado en Música

México D.F a 8 de Febrero de 2014

Índice general

Capítulo 1

Introducción

El presente trabajo propone un planteamiento teórico de espacialización sonora basada en procesos algorítmicos con desarrollos de implementaciones prácticas dentro del contexto del lenguaje de programación Supercollider. El trabajo por lo tanto está dividido en dos partes. La primera parte consta del planteamiento de un modelo categórico y teórico general, basado en teoría de conjuntos, que permita establecer las bases para integrar, analizar y desarrollar creaciones de música electroacústica basada en procesos algorítmicos y en particular a lo referente a la espacialización multicanal. La segunda parte es el desarrollo de un sistema general de espacialización que permita generar percepciones dinámicas tridimensionales en el escucha.

El modelo en su versión de desarrollo funcional, utiliza sistemas dinámicos para generar series de tiempo construidas a partir de las órbitas de cualquier sistema las cuales funcionarán como materia prima de datos (raw data) potencialmente mapeables a un vector de sistema el cual contendrá la información temporal sonora discretizada; desde localización espacial, amplitud, reverberación o algún otro parámetro sonoro deseado.

El uso de ciertos sistemas dinámicos en su versión discreta, como fuente de datos para generar envolventes de control de parámetros sonoros se justifica por su fácil implementación y ejecución en código, por su capacidad de predecir rangos de comportamiento mediante el diagrama de bifurcación que permite al usuario elegir entre series de datos que presenten comportamiento, estable, caótico o cíclico, así como la cantidad ingente de datos que un sólo sistema puede arrojar con una solo conjunto de condiciones iniciales. Se presenta además, toda una estructura categórica basada en la formalidad matemática de la teoría de conjuntos, con el fin de generar un modelo que pueda ser general y que sirva de fundamento para diversos casos específicos de aplicación práctica.

Cuando se utilizan sistemas dinámicos no lineales como elementos centrales en la espacialización multicanal, la intención a priori no es generar una estética perceptual

del caos respecto a la espacialización, [1] sino utilizar las características inherentes de los sitemas dinámicos como generadores de datos en un sentido algorítmicamente económico, controlable y afluente. No obstante, al recurrir a dichos procesos tan específicos desde el punto de vista técnico-matemático, el devenir inevitable converge en una discusión no sólo del uso, sino de la apreciación estética de este proceso en si. Tal discusión no se abordará en el presente trabajo y únicamente se presentará una breve discusión introductoria al respecto.

En el primer capítulo se presenta una revisión general sobre los sistemas dinámicos y se plantea la manera en que el presente trabajo propone trabajar con ellos. Se establecen los conceptos de interacción tecno-científica, flujo y fuente de información y además se describe la forma en que el compositor puede generar envolventes mediante órbitas arbitrarias de sistemas dinámicos para controlar cualesquiera parámetros sonoros.

A continuación se presenta el planteamiento general del modelo, en donde se hace toda la propuesta teórica categórica basada en conjuntos, álgebra y series de tiempo. La finalidad de este capítulo es establecer un marco teórico que pueda funcionar como soporte general del modelo planteado de modo tal que cualquier caso particular de la misma naturaleza, pueda ser no sólo desarrollado sino analizado a partir de dicho modelo. Se proponen diversas definiciones y términos con el fin de establecer categorías y metacategorías de clasificación para generar una estructura teórica funcional y coherente basada en la teoría matemática de conjuntos. Este capítulo establece y desarrolla las ideas de *configuraciones multicanal*, *representación matricial de configuraciones*, *parámetros sonoros* y *flujo de datos* de manera general para que cualquier herramienta teórica matemática y no sólo los sistemas dinámicos, pueda ser utilizada dentro del mismo.

En el siguiente capítulo titulado *Espacialización mediante sistemas dinámicos*, se establece el método para obtener series de datos arbitrarias mediante la generación de órbitas provenientes de sistemas caóticos. Se construye además un método general de mapping que traduzca dicho flujo de información en control real y tangible de cualesquiera parámetros sonoros. Finalmente se presenta la idea de la colocación del sonido mediante planos de campo estéreo que es la base para el desarrollo del paneador $PanW$; una clase escrita en el lenguaje de programación Supercollider.

En el último capítulo se presentan tres clases escritas en Supercollider: *Map_ Chaos*, $PanW$ y *Chaos_ Pan*. La primera es un sistema funcional para generar órbitas para distintos sistemas dinámicos discretos y un caso especial de sistemas Lindenmeyer mediante diccionarios dinámicos estocásticos. La segunda es un sistema de paneo que permite definir espacializaciones mediante trayectorias arbitrarias dentro de una

[1] En el sentido de que no se pretende crear en el espectador una percepción representativa de lo que se pueda entender por caos.

configuración multicanal. El último sistema, *Chaos_Pan* es la conjugación de las dos clases anteriores; permite controlar cinco parámetros sonoros fundamentales en la percepción posicional del sonido mediante envolventes generadas por series de datos arbitrarias y paneadas mediante $PanW$; estos parámetros son: posición de paneo, amplitud de paneo, profundidad local (reverb mix), altura relativa (reverb damp o ecualizador en bandas de frecuencia alta) y profundidad general (reverb room).

Todo el desarrollo de conceptos, definiciones y herramientas matemática, se encuentra detallado en las secciones del apéndice; ahí se presentan los fundamentos matemáticos sobre sistemas dinámicos y ciertos ejemplos específicos que fueron utilizados para desarrollar los algoritmos en el lenguaje de programación mencionado. La intención del capítulo es ofrecer al lector un panorama general de los preeliminares matemáticos de teoría de conjuntos y sistemas dinámicos no lineales, lo más formal y accesible al mismo tiempo, a modo tal que pueda utilizar dichas herramientas con conciencia y de manera práctica en su proceso composicional.

Capítulo 2

Planteamiento

En la actualidad se posee al alcance, gran cantidad de información acerca de la forma en la que el cerebro humano es capaz de localizar espacialmente, fuentes sonoras ante estímulos externos. En base a referencias como el ITD (Iteraural time differece), IID (interaural intense diference),[1] las HRTF (head related transfer funtion), entre muchas otras, representan un enorme campo de investigación en el área de la psicoacústica y acústica y han sido de enorme utilidad para generar sistemas de espacialización como el VBAP, el Ambisonics, sistemas binaurales, la holofonía o el Wave Field Synthesis. Estos sistemas permiten al usuario generar ambientes acústicos virtuales y crear en el escucha la ilusión de fuentes sonoras virtuales y/o localización tridimensional de reproducción de fuentes reales, muchas veces con resultados por demás satisfactorios.

También es cierto que cada sistema posee sus puntos débiles, sobre todo al momento de establecer implementaciones prácticas fuera del laboratorio y sin condiciones ideales tanto de características físicas del recinto como de equipo técnico de sonido utilizado, además de que la correcta instalación de algún sistema multicanal basado en las técnicas descritas, es por demás costoso en muchas ocasiones.

En este sentido, el presente trabajo es un modelo teórico con una implementación práctica definida para la espacialización multicanal basada en procesos algorítmicos y en particular en los sistemas dinámicos que presentan comportamiento caótico. Al generar configuraciones multicanal tridimensionales (esto es donde los arreglos de bocinas no se encuentran únicamente en un sólo nivel como comúnmente sucede en la mayoría de las aplicaciones del mundo de la música electroacústica), sino en dos o más niveles, la distribución del sonido adquiere una forma distinta de percepción por parte del escucha. Por otro lado, se propone que la posición dinámica del sonido respecto al tiempo, esto es, la posición del paneo dentro de la configuración, esté basada no mediante los movimientos tradicionales de paneo entre salidas consecutivas, sino en trayectorias arbitrarias cuyos elementos son precisamente las salidas de la

[1] Estas dos son denominadas *referencias de localización binaural* (Bernstein 1994),

configuración pero en cualquier orden deseado. De este modo, si se tiene una configuración multicanal de 16 salidas, para la trayectoria dada por $[0, 3, 6, 7, 2, 1, 0, 3]$ el paneo tendrá un recorrido con el orden establecido por la bocina 0, luego la 3, luego la 6 y así sucesivamente.

La percepción dinámica de movimiento de la fuente sonora virtual o real dentro del sistema propuesto, está planteada en términos de parámetros sonoros específicos: paneo general multicanal, amplitud de la fuente, reverb mix, reverb damp, reverb room y ecualizador de bandas. Al generar envolventes provenientes de cualquier fuente de información (y en particular de los sistemas dinámicos en general) que controlen cada uno de estos parámetros por separado para cada fuente sonora individual, se generan percepciones espaciales de movimiento detalladas y diferenciables. De este modo, el sistema puede adaptarse al recinto físico y a las necesidades creativas del compositor, permitiendo explorar y usar las características básicas de reverberación y absorción natural del sonido de dicho recinto. Esto ofrece al compositor, una herramienta útil y eficaz basada en una convergencia multidisciplinaria palpable, para ser utilizada en su proceso creativo y cuya implementación práctica no requiere de excesivos o costosos esfuerzos logísticos.

Si bien, cualquier trabajo que aborde el tema de la distribución del sonido en una configuración multicanal está ineludiblemente atado a las consideraciones anteriores, resulta necesario hacer una distinción que el tema que tratan es específicamente sobre la percepción psicoacústica del individuo como objeto de estudio. En el presente trabajo se dan por hecho esos resultados y se propone utilizar estos modelos como una herramienta incorporada al final del sistema planteado, sobre todo si se desea generar efectos específicos en el escucha de creación virtual de ambientes o recintos acústicos. Sin embargo, si por el contrario se desean explotar las características físicas del recinto, el sistema por sí sólo es capaz de generar percepciones audibles de movimiento dinámico de fuentes sonoras.

2.1. Objetivo

Presentar un modelo general teórico de espacialización sonora multicanal algorítmica basada en el mapeo de sistemas dinámicos generados por sistemas de ecuaciones en diferencias y enfocado a la creación y práctica de música electroacústica, generando como consecuencia una aplicación funcional escrita en el lenguaje de programación Supercollider que pueda servir como una herramienta directa y clara al alcance del compositor. Dicho modelo tiene una extensión natural para la implementación del control de otros paramétros comunes en la síntesis por computadora como son: envolventes, modulaciones tímbricas, contenido espectral, etc.

Establecer además, las bases para el planteamiento teórico de una *estética del proceso* basada en el concepto de *interacción tecno-científica*, donde el proceso de percepción

de un objeto artístico y en particular aquel referido al área de la música electroacústica en general, puede ser incluyente respecto a las herramientas teórico-prácticas utilizadas.

Finalmente se pretende generar todo un sistema categórico mediante el planteamiento de conceptos y definiciones claras, que permita establecer las bases de una organización y clasificación general para una referencia explícita de la creación electroacústica basada en procesos algorítmicos y en su extensión natural a cualquier tipo de interacción tecno-científica, mediante una estructuración basada en la teoría matemática de conjuntos.

Capítulo 3

Sistemas dinámicos y Caos para la creación electroacústica.

En este capítulo se mostrarán las características de los sistemas dinámicos no lineales que se usaron dentro de la implementación del modelo general, para el contexto de la composición electroacústica y en particular para la espacialización multicanal. El desarrollo y los conceptos matemáticos de estos sistemas dinámicos caóticos pueden consultarse en el apéndice A3.

Se comenzará definiendo el concepto de *interacción tecno-ceintífica*, como una propuesta teórica que engloba la convergencia transdisciplinar entre arte, ciencia y tecnología y que permite fundamentar categóricamente el uso de procesos algorítmicos dentro de un contexto de composición electroacústica y de live electronics.

Por otro lado, se discutirán las características, ventajas y desventajas en el uso de los sistemas dinámicos en la creación electroacústica y en particular para los elegidos para el propósito actual de este trabajo: la espacialización multicanal.

3.1. El concepto de Interacción Tecno-Científica.

Hoy en día el proceso de la actividad artística y en particular la creación sonora ligada a las áreas de lo que se ha concesado denominar música acusmática, electroacústica, electrónica e interactividad, ha llegado a un punto de inflexión respecto al contexto de la transdisciplina en el sentido de que resulta cada vez más difícil establecer de forma clara los límites de la cooperación interdisciplinaria con otras áreas (en especial aquellas relacionadas con la ciencia y la tecnología) así como del producto final obtenido y por lo tanto de la forma de percepción estética de dicho producto. Día a día se genera material artístico y académico en el que se incorporan de forma cada vez más profunda, herramientas teórico-prácticas derivadas de los sistemas dinámicos, fractales, sistemas complejos, cibernética, autómatas celulares, redes neuronales, inteligencia artificial, genética, probabilidad, estadísitica o teoría

de control por mencionar sólo algunos. De hecho como se ha venido observando desde la década de los años cincuenta del siglo pasado, es virtualmente posible encontrar alguna aplicación sonora para casi cualquier proceso técnico-matemático conocido y esto es, a opinión del autor del presente trabajo, debido a un único concepto: el mapping; *la traducción de la fuente de información al parámetro audible*. El mapping permite conectar mundos tan disconexos como son los sistemas dinámicos caóticos y la música electroacústica, no sólo en concepto sino en señal audible perceptible.

La diferenciación que hace Buxton respecto a la *Música por Computadora* en *composición programada*, *Composición asistida por computadora* y *Generación de señales acústicas* (Buxton 1977), puede ser fácilmente extendida a un campo más general de lo que bien podría denominarse *Interacción técno-científica en la creación sonora* debido a que la clasificación de dicho autor está basada en el reconocimiento explícito de la computadora como sistema que realiza una tarea y del grado de interacción entre el usuario (compositor) y dicho sistema; entendiendo por *Interacción técno-científica*, el amplio proceso transdisciplinar mencionado con anterioridad que comprende la convergencia entre arte, ciencia y tecnología. Desde este punto de vista es directamente consecuente hablar no sólo de composición algorítmica sino de *composición con interacción tecno-científica*, definiendo clasificaciones análogas a las propuestas por Buxton derivadas de la forma y el grado en que ésta interacción tiene lugar en el proceso de composición.

Hasta estos días y tras el legado de Cage, Xenakis, Stockhausen y demás emblemáticas figuras, numerosos autores, académicos y artistas han generado grandes contribuciones enfocadas a esta *interacción tecno-científica* en la composición en las tres categorías mencionadas con anterioridad: el uso de la interacción como proceso de automatización de la composición, la composición asistida por *interacción* y la generación de señales basadas en *interacción* y podría bien agregarse una más; aquella que utiliza la *interacción* para el análisis y/o reconocimiento d patrones. Destacan las publicaciones y los trabajos de Rick Bidlack, Elizabeth Anderson, Agostino DiScipio, Yota Morimoto, Eduardo Reck Miranda, Jeff Pressing, James Harley, Roberto Morales, John Young, Dennis Smalley, KarlHeinz Essl Jr. y Manuel Rocha por mencionar unos pocos pero representantivos iconos de la investigación actual y práctica de la interacción tecno-científica en la composición, que han hecho un uso exitoso y con resultados realmente loables.

Se tienen de este modo, dentro del campo de la música electroacústica, por un lado, aplicaciones de sistemas dinámicos caóticos , autómatas celulares, redes neuronales y demás, a la manipulación del timbre, la creación de patrones composicionales, la generación de señales por medios espectrales, el análisis de obras y de señales, la espacialización multicanal, etc (Di Scipio 1999), (Alpern 1995), (Boon 1994), (R. Miranda 2007), (Bilotta), (Bidlack 1992), (Bolognesi 1987). Por otro lado existen las transformaciones sonoras de modelos físicos relacionados con la dinámica de fluidos,

la astrofísica, procesos biológicos, etc. las cuales se presentan como una forma de interpretación auditiva del fenómeno específico en sí. Más aún, es necesario diferenciar también a todas aquellas obras que involucran el uso de la tecnología para su desempeño performativo sean o no interactivas. Si bien aquí se referencia la eterna discusión sobre lo que es tecnología o nuevos medios, esto permite establecer fronteras claras en el presente trabajo sobre el planteamiento y la postura elegida para abordar este tema. Como simple ejemplo imaginemos un compositor que utiliza un ordenador para escribir una partitura para una obra con instrumentos tradicionales sin ninguna referencia a algún tema o proceso tecno-científico en la creación o desarrollo de la misma. Uno podría preguntarse si existe interacción tecno-científica, dado que el autor está utilizando tecnología o *nuevos medios* para transcribir su pieza, sin embargo, desde la perspectiva propuesta, el grado de interacción tecno-científica sería a un nivel primario y meramente operativo, por lo que tal caso carecería de interés para su estudio. No así para el caso del live electronics, de la obras con interactividad, del net art o del arte electrónico, por ejemplo, puesto que el proceso, el soporte y el planteamiento estético están claramente relacionados con el uso de la tecnología.

El concepto general de *interacción tecno-científica* se puntualiza en el presente trabajo, para el caso particular de *sistemas dinámicos aplicados a la espacialización multicanal*. En este sentido, el mapping consiste en establecer el contexto y la estructura en que la información proveniente de estos modelos matemáticos se pueda traducir y materializar en formas concretas de distribución y colocación del sonido dentro de una configuración multicanal.

El uso de modelos matemáticos relacionados con caos y procesos algorítmicos dentro de la creación electroacústica, tiene ya varias décadas de investigación y aplicación constante por parte de artistas que buscan incluir elementos no lineales dentro de su obra. Como se mencionó anteriormente, la manera de incorporar dichos elementos a la creación o ejecución en tiempo real, puede ser englobada en cuatro vertientes generales: el uso de la interacción como proceso de automatización de la composición, la composición asistida por *interacción*, la generación de señales basadas en *interacción* y la *interacción* para el análisis y/o reconocimiento d patrones.

La postura que el presente modelo adopta es la relacionada a las tres primeras de las cuatro anteriores con respecto al uso específico de ciertos *sistemas dinámicos* que presentan comportamiento caótico. Para consultar definiciones y desarrollo matemático explícito sobre los sistemas dinámicos y herramientas relacionadas, consúltese el apéndice A3.

3.2. Sistemas dinámicos como generadores de información.

Los sistemas dinámicos son modelos matemáticos que surgen debido a la necesidad de estudiar ciertos fenómenos naturales cuya complejidad y comportamiento no era posible predecir con facilidad. A partir de la segunda mitad del siglo pasado, los sistemas dinámicos comenzaron a tener un desarrollo considerable y su uso se fue extendiendo a diversas ramas científicas, hasta que hoy en día existe una aplicación de dichos modelos para virtualmente todas las áreas de investigación no sólo científicas, sino sociales y hasta humanistas: economía, biología, ciencias de la computación, sociología, complejidad, organización industrial, finanzas, etc.

La manera de modelar los sistemas dinámicos es mediante *sistemas ecuaciones diferenciales*, las cuales hacen representaciones matemáticas del modelo deseado cuando el parámetro del tiempo es continuo. La mayoría de las veces no es posible resolver estas ecuaciones por medios analíticos por lo que se utilizan métodos de aproximación numérica para poder obtener resultados con ciertos margenes de error acotados. Muchas veces, con el fin de obtener soluciones analíticas y computables, estos modelos son transformados a expresiones en donde el tiempo toma valores discretos o a pasos, dichas expresiones están conformadas por *ecuaciones en diferencias*, las cuales iteran resultados consecutivamente de la función o modelo en cuestión, obteniendo de este modo, un historial del comportamiento del sistema a lo largo del tiempo, dadas ciertas *condiciones iniciales* específicas; dicho historial es llamado *órbita*.[1] Puesto que los sistemas son iterativos, es necesario que especifiquemos los valores con los que el sistema comenzará su proceso; estas son las llamadas *condiciones iniciales*, y son los valores con que se inicializan los parámetros y las variables del sistema.

Los sistemas dinámicos pueden ser unidimensionales (que constan de una sola ecuación), o multidimensionales (que constan de un sistema de dos o más ecuaciones). Sin embargo, para aplicaciones comunes, la dimensión más alta para dichos sistemas es 2 y en la mayoría de estos casos, para la versión de ecuaciones en diferencias, es posible reducir el sistema a una dimensión mediante sustituciones específicas.

La categorización del comportamiento de los sistemas dinámicos depende frecuentemente del *diagrama de bifucación*, el cual muestra la dinámica de dicho comportamiento respecto a lo que se denomina *parámetro de control*, que es una variable multiplicativa dentro del sistema, la cual define los valores para los que el sistema presenta bifurcaciones; esto es, *cambio de comportamiento*. Este diagrama es de hecho, una de las herramientas más útiles para el compositor que desee hacer uso de los sistemas caóticos en su proceso creativo, ya que el poder interpretar dicho diagrama, le permitirá saber de antemano que conjunto de valores iniciales le arrojará órbitas

[1] Véase el apéndice A3.

con comportamiento estable, periódico o dinámico.

Cada sistema dinámico presentará un comportamiento distinto de acuerdo a la forma en que está definido y al conjunto de condiciones iniciales con que se le dote; si por ejemplo, al variar ligeramente un conjunto de condiciones iniciales, se obtienen órbitas totalmente distintas, se dice entonces que el sistema tiene *sensibilidad ante condiciones inciales* o tropicalmente conocido como *efecto mariposa.* Por otro lado, para poder establecer cuando y bajo que condiciones un sistema presenta comportamiento caótico, se utilizan distintas pruebas matemáticas entre las que sobresalen los *exponentes de Lyapunov*, los cuales caracterizan el grado de separación de dos órbitas distintas en el transcurso del tiempo. Cómo se muestra en el apéndice A3, los valores de estos exponentes con respecto a un parámetro de control, también puede ser utilizado como flujo de información.

Se propone en este modelo, establecer una perspectiva definida de los sistemas dinámicos como *fuentes de información*, generadores de *raw data* para la creación electroacústica y live performance aplicados a la espacialización multicanal. Esto significa, que dentro del presente contexto, los sistemas dinámicos serán considerados como los generadores de la materia prima y elementos constitutivos para establecer la distribución del sonido dentro de una configuración multicanal predefinida. Se utilizarán en la implementación práctica de este trabajo, ciertos modelos discretos de sistemas dinámicos, i.e. definidos mediante ecuaciones en diferencias. Este tipo de modelos discretos presentan un par de ventajas: su diseño algorítmico resulta sencillo y el procesamiento de dicho algoritmo resulta económico computacionalmente hablando.

En este punto es posible que el lector se sienta intrigado tal vez, acerca de las razones puntuales (aparte de lo mencionado respecto a la implementación en el diseño algorítmico) por las que se han elegido los sistemas dinámicos como la *fuente de información* para la creación de *raw data composicional.* Cabe aclarar dos puntos importantes:

- El modelo teórico planteado en el capítulo siguiente está pensado para ser una generalización en la espacialización multicanal de acuerdo al concepto de interacción tecno-científica. Esto significa que de hecho, el modelo está diseñado como una estructura categórica capaz de integrar en ella, cualquier fuente de información. En este sentido, cualquier proceso algorítmico puede funcionar como fuente de información: sistemas dinámicos no lineales, cadenas de Markov, redes neuronales, aprendizaje máquina, complejidad, etc.
- Los sistemas dinámicos en su versión discreta presentan la ventaja de generar una enorme cantidad de datos con múltiples variaciones de acuerdo al conjunto de condiciones iniciales especificas. Es posible además, visualizar y predecir el comportamiento de las órbitas en series de datos de acuerdo al diagrama de bifurcación, lo que permite al compositor elegir rangos de condiciones iniciales

para obtener comportamiento caótico, estable o periódico según se observe en el diagrama de bifurcación.

A continuación se precisa la idea de *fuente de información* y de *flujo de información* y su específica implementación al caso de los sistemas dinámicos.

3.3. La fuente de información y el flujo de información

Dentro del contexto de *interacción tecno-científica* en el arte, es posible distinguir tres etapas o componentes bien definidos que a juicio del autor del presente trabajo, forman la base de dicha convergencia transdisciplinaria :

- **Fuente de información**. El proceso *tecno-científico* elegido por el artista para generar datos de algún tipo y que sean potencialmente útiles en la creación y constitución de su obra.
- **Flujo de información**. La selección y recolección de ciertos tipos de datos provenientes de la fuente y que cumplen con un criterio definido por el artista y sus necesidades para la pieza en proceso.
- **Mapping**. La forma de traducir la información proveniente de la fuente a parámetros específicos dentro de la obra; i.e. el puente entre el proceso tecno-científico y el proceso artístico.

Si se analiza por ejemplo la Illiac Suite (Hiller, Isaacson, 1959), con el planteamiento anterior propuesto, la fuente de información resulta ser el *método de Monte Carlo*, el flujo de información son ciertos conceptos provenientes de *la Teoría de la Información* (Shanon 1948) y el mapping es precisamente el proceso compuesto por las fases de *inciación, generación y verificación* (Di Nunzio 2011).

En este sentido, la fuente de información es básicamente cualquier proceso que sea capaz de proveer datos, puede ser desde un proceso algorítmico (cadenas de markov, sistemas dinámicos, autómatas celulares, etc) hasta bases de datos comunes (índice bursátil, indicadores económicos, datos astronómicos, datos de censado regional, etc.) El flujo de información es la forma en la que el artista discrimina y selecciona los datos provenientes de la fuente con el fin de establecer qué datos le sirven para qué propósito en específico, generando grupos de información filtrada lista para ser utilizada en la obra. Finalmente, el mapping es la forma en la que el artista traduce el flujo de información en parámetros específicos propios del campo artístico.

Obsérvese que el simple hecho de elegir una fuente de información y de definir un flujo de información, conduce inevitablemente a un planteamiento estético que está

ligado inherentemente a dichas fases de interacción tecno-científica y que culmina o se materializa en el mapping y posteriormente en la generación de la obra; esto es precisamente lo que se pretende discutir en futuros trabajos bajo el nombre de *estética del proceso*, bajo la cual se propone generar una apreciación estética que incluya la percepción y aprehensión del material tecno-científico utilizado en la obra y no sólo el producto final obtenido.

3.4. La espacialización sonora mediante sistemas dinámicos

Los sistemas dinámicos, tal y como se plantean en este modelo, son entonces, una herramienta muy útil para poder generar colecciones muy grandes de datos las cuales posteriormente servirán, mediante distintos mapeos, como controladores de diversos parámetros sonoros. En este sentido, las composición algorítimica se vuelve una herramienta al servicio del creador, como fuente primaria en el modelado de su obra.

Como se mencionó anteriormente, las órbitas de los sistemas dinámicos representan el historial de las iteraciones hechas para un conjunto de condiciones iniciales dadas, la idea básica del presente modelo radica en generar series de datos a partir de dichas órbitas, con las cuales puedan generarse después, envolventes que funcionen como control en el tiempo de un cierto parámetro específico dentro de una configuración multicanal. Se propone en el presente modelo que para una configuración multicanal dada, existen cinco parámetros sonoros básicos que definen y permiten al escucha tener una sensación tridimensional de movimiento de una fuente sonora. Estos parámetros son en principio:

- **La posición paneada del sonido** dentro de la configuración multicanal definida por un par o más de bocinas para una fuente sonora específica.
- **La amplitud del sonido** respecto a su posición paneada dentro de la configuración multicanal, para una fuente sonora específica.
- **El parámetro mix del reverb** respecto a la posición paneada dentro de la configuración multicanal, para una fuente sonora específica, el cual genera la sensación de profundidad.
- **El parámetro room del reverb** respecto a la posición paneada dentro de la configuración multicanal, para una fuente sonora específica, el cual genera la sensación de amplitud espacial.
- **El parámetro damp del reverb** respecto a la posición paneada dentro de la configuración multicanal, para una fuente sonora específica, el cual funciona como un filtro suave que permite colocar el sonido de manera vertical.

Se propone entonces que cada uno de estos parámetros esté controlado respecto al tiempo, por una envolvente particular creada por la integración de una o más órbitas provenientes de alguno de los sistemas dinámicos propuestos. De este modo, para la parte de la implementación práctica, como se expondrá a lo largo del presente trabajo, se verá que la fuente de información serán los sistemas dinámicos no lineales seleccionados, el flujo de información será la conformación de las series de datos por medio de las órbitas y finalmente el mapping estará constituido por la generación de envolventes de control de parámetros sonoros de posición, amplitud y reverb dentro de una configuración multicanal.

Este acercamiento permite al compositor usar libremente cualquier proceso algorítmico como una herramienta para su proceso creativo, evitando en cierta medida condicionarse a la misma y aprovechar las ventajas algorítmicas ya mencionadas de los sistemas dinámicos no lineales como generadores de datos potencialmente mapeables a parámetros sonoros.

3.5. Sistemas caóticos discretos en práctica.

Para poder generar series de datos de manera práctica y eficiente, se escribió una clase en el lenguaje de programación que incluye un catálogo de seis sistemas dinámicos implementados en su versión para ecuaciones en diferencias. El algoritmo escrito en Supercollider arroja las órbitas para el número de iteraciones y condiciones iniciales dadas y permite plotear dicha serie para que el compositor se de una idea de los elementos con los que podría generar las envolventes de control.[2]

Cada uno de estos sistemas posee características bien definidas que permiten al compositor diferenciar entre rangos de condiciones iniciales para obtener series de datos con comportamiento caótico, estable o periódico mediante el diagrama de bifurcación, además de representar los modelos estándar de sistemas caóticos para aplicaciones foráneas a la teoría matemática.

La clase se denomina *Chaos_Map* y está conformada por las implementaciones del mapeo logísitco, el mapeo de Henon, el mapeo cúbico, el mapeo Duffing, el mapeo tent (tienda), el mapeo caótico seno y una versión puntual de sistemas lindenmeyer con diccionarios dinámicos estocásticos. Con esta clase el compositor podrá construir envolventes y series de tiempo basadas en órbitas provenientes de sistemas caóticos, de una manera fácil y rápida, definiendo las condiciones iniciales junto con el número de iteraciones deseado para cada sistema.

En el apéndice A3 se muestran distintas gráficas de series de datos arrojadas por cada uno de los sistemas elegidos para condiciones iniciales dadas. La idea básica

[2]Véase el apéndice A3.

es considerar estas series de datos como raw data o flujo de información para ser mapeado a los parámetros sonoros de espacialización antes descritos. [3] La ventaja de trabajar con estas series de datos, es que es posible realizar transformaciones sobre ellas obteniendo nuevas series que resultan ser variaciones de las primeras, para ello se proponen dos tipos de transformaciones: las isometrías euclideanas[4] y la convulución o composición de funciones[5]. Al transformar las series de datos mediante alguno de estos procesos, se obtienen series nuevas e independientes pero que mantienen la estructura de la serie original; esto ofrece al compositor, la posibilidad de obtener material creativo que mantenga coherencia y cohesión a pesar de tratarse de elementos distintos.

Una vez que se ha generado el flujo de información deseado, el siguiente paso consiste en definir los parámetros sonoros objetivo, a los cuales se les desea aplicar control mediante las series de datos obtenidas por las órbitas de los sistemas dinámicos, que en este caso serían los cinco parámetros de espacialización mencionados con anterioridad. Una serie de datos es una colección de valores o puntos graficados, provenientes de alguna fuente con respecto al paso del tiempo. Una envolvente puede pensarse como la manera en que se unen estos puntos, uno después de otro; esto es a grandes rasgos lo que se denomina interpolación. Cuando la interpolación es lineal, los elementos que unen los puntos son precisamente lineas rectas, en cambio cuando la interpolación es caudrática o cúbica, los elementos que unen los puntos se vuelven curvas. Ambos casos se muestran en la figura 3.1, la cual hace el llamado de la clase *Map_ Chaos* para el mapeo cúbico. En este sentido, las envolventes muestran de manera gráfica, el cambio en el tiempo, del valor de un parámetro dentro de cierto rango con un método de interpolación definido.

Dentro de Supercollider es posible definir envolventes con interpolación variable e independiente para cada punto, definiendo la excentricidad de la curvatura a cada paso. Lo anterior se logra con el método *.xyc* de la clase *Env*. La figura 3.2 muestra una envolvente creada a partir de la clase *Map_ Chaos*. Primeramente se hace la llamada al mapeo tent (tienda) y luego al mapeo cúbico ambos instanciados con 50 iteraciones. Posteriormente se define una función moduladora o transformadora para que afecte por convolución a la serie de datos del mapeo tent; esta función es la conocida $f(x) = xcos(x)$. Finalmente se define otra serie de datos, pero ahora proveniente del mapeo Duffing normalizada al intervalo $[-4.0, 1.5]$, para ser asignado a la curvatura individual dinámica de la envolvente (que es un rango apropiado para dicho parámetro de interpolación).

[3] En el capítulo 5 se detalla el proceso mediante el cual es posible formar series de datos arbitrarias proveninetes de cualquier fuente de información.

[4] Véase la sección 5.2.2

[5] Véase la sección 6.4

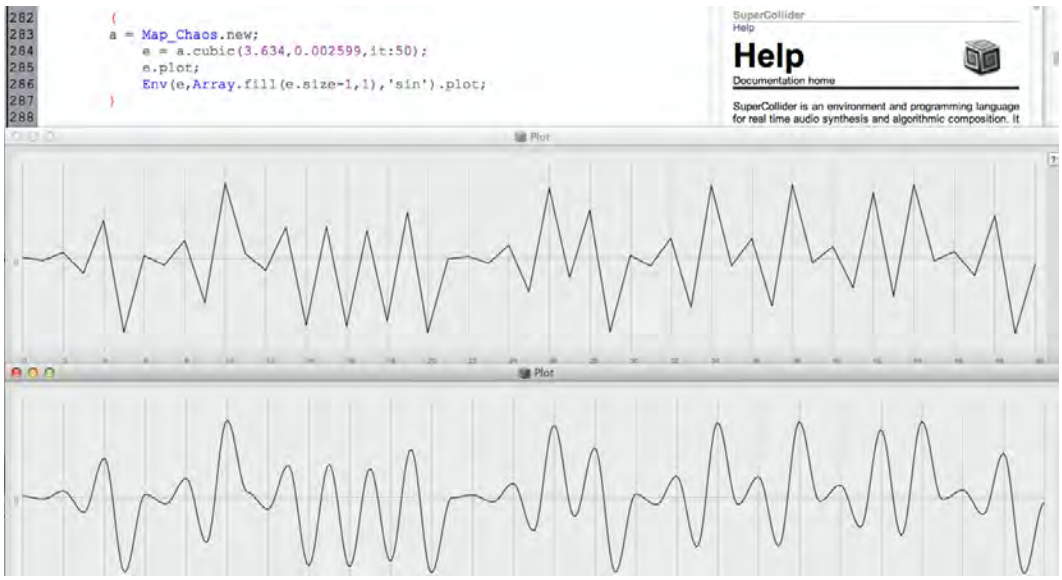


Figura 3.1: Envolvente creada a partir del mapeo cúbico con interpolación lineal (arriba) e interpolación basada en la función seno (abajo).

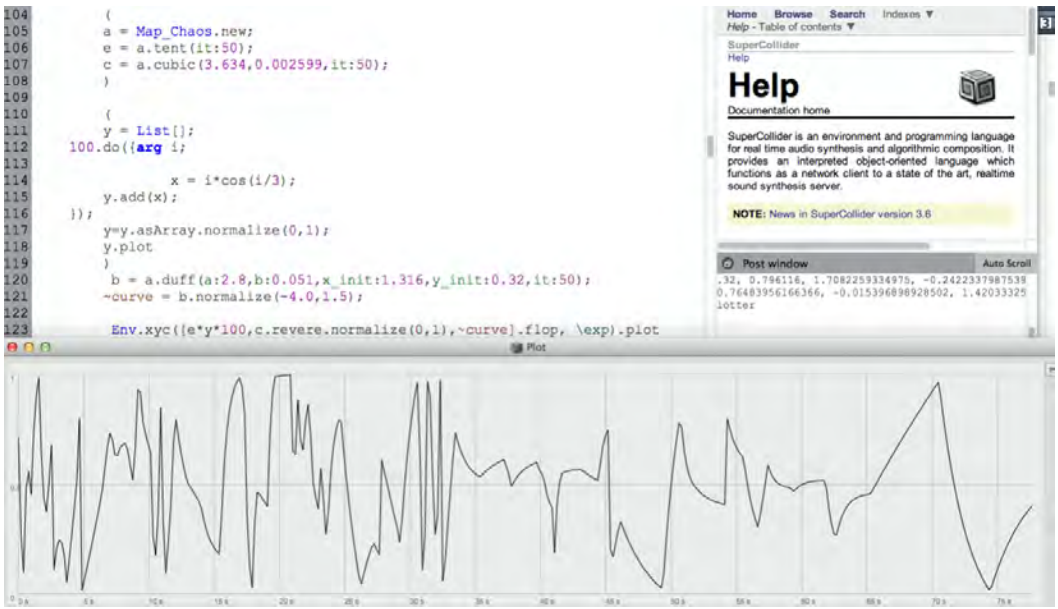


Figura 3.2: Envolvente creada a partir del mapeo tent, el mapeo Duffing, el mapeo cúbico y la función moduladora $xcos(x)$.

Esta envolvente puede ser mapeada entonces, a cualquier parámetro sonoro objetivo, como por ejemplo la amplitud o la posición de paneo de alguna fuente sonora. Observando la morfología gráfica de la envolvente, el compositor puede darse una

idea del comportamiento audible que dicho parámetro tendrá al ser controlado por la envolvente en cuestión.

Así, el compositor es capaz de crear distintas envolventes a partir de series de datos arbitrarias para después mapearlas a cualquier parámetro sonoro deseado tanto en el ámbito de la espacialización como en cualquier otro contexto de composición electroacústica o live electronics. Todos los detalles técnicos así como los algoritmos de programación de Supercollider al respecto, están desarrollados en el anexo A3 y en el capítulo 6 respectivamente.

Capítulo 4

Planteamiento del modelo General

Este capítulo estará dedicado al planteamiento teórico y categórico de un marco general para la espacialización de distintas funtes sonoras afectadas por distintos parámetros mediante trayectorias creadas por series de datos construidas mediante sistemas dinámicos. El objetivo de este modelo de espacialización no está enfocado en la búsqueda de un modelo de localización psicoacústica exacta de fuentes virtuales dentro de un espacio sonoro como los sistemas VBAP, Ambisonics o Binaural, sino en la incorporación de estos métodos dentro de un marco teórico-práctico basado en trayectorias arbitrarias predefinidas dentro de una configuración multicanal cualquiera y cuya distribución del sonido a través de cada una de las salidas componentes de dicha configuración, esté gobernada por un marco de interpretación general de los sistemas dinámicos no lineales. En este sentido, se definen conceptos y categorías que permitan establecer un modelo lo más general, formal y coherente posible, que utilice como línea medular, los sistemas dinámicos caóticos tanto conceptual como pragmáticamente, además de generar una base para el desarrollo algorítmico del manejo de datos y el flujo de información en implementaciones prácticas desarrolladas en el lenguaje de programación Supercollider.

Se da por hecho que el lector está familiarizado con la extensa literatura acerca de las características físicas y psicoacústicas involucradas en el proceso de localización de fuentes sonoras por el oído humano. Se hace énfasis de nuevo, en que el objetivo de este trabajo no está orientado en la investigación psicoacústica de la localización de fuentes sonoras que hace el sistema auditivo humano, en cambio, se busca establecer un marco teórico y práctico enfocado a la composición de música electroacústica y en particular mediante la espacialización del sonido dentro de una configuración multicanal. Para efectos de consulta más detallada sobre el mecanismo y características del proceso de localización de fuentes sonoras por el sistema auditivo, se recomienda (Wang Brown 2005).

4.1. Configuración multicanal de bocinas.

Una parte muy importante para la espacialización es la ubicación del escucha respecto a las bocinas. La configuración tradicional estéreo ofrece principalmente, la posibilidad de localización de fuentes virtuales sobre una línea de horizonte L-R (left, right). Localizaciones de fuentes virtuales en ubicación *up-down* y *front-back* (profundidad) tienen un rango reducido de acción dentro de esta configuración. En este sentido, el número de fuentes de reprodución o bocinas así como la forma en que están dispuestas a través del espacio físico cobran importancia al momento de generar espacialización sonora y un ambiente sonoro inmersivo.

Dentro del campo de la música electroacústica, gran parte de el uso de la espacialización multicanal está enfocado a la generación de ambientes envolventes que funcionen a modo de paisaje sonoro, como un soporte de creación y percepción de la obra. Se tienen entonces, ciertos elementos muy claros para dicho contexto de espacialización; el número de bocinas, su disposición físíca sobre el espacio de performance[1], la ubicación del escucha dentro de esta configuración y por supuesto el algoritmo utilizado para distribuir el sonido dentro de la configuración.

Se establecerá en la siguiente sección, un método de generación de configuraciones o *arrays* primitivos basados en la obtención de raíces complejas de la unidad. A partir de estas configuraciones se obtendrán variaciones en el diseño mediante deformaciones obtenidas por anamorfósis. Se plantearán también alternativas de disposiciones físicas del espacio ocupado por la audiencia dentro de las configuraciones de bocinas.

4.1.1. Configuraciones simétricas de bocinas basadas en polígonos

Las proyecciones en el diagrama de Argand de las raíces enésimas de la unidad, forman los vértices de un polígono regular de n lados inscrito en el círculo unitario centrado en el origen. Dado $z \in \mathbb{C}$ un número complejo, y $n \in \mathbb{N}$, el polinomial $z^n = 1$ tiene exactamente n raíces las cuales forman los vértices del polígono regular inscrito dentro de la círculo unitario. Los números complejos que satisfacen la ecuación anterior son llamados raíces enésimas de la unidad.

Sea $z = r(cos\theta + isin\theta) = re^{i\theta}$ un número complejo en su forma polar. Utilizando la fórmula de Moivre, es fácil calcular las ráices unitarias de dicho número.[2] De este modo resulta sencillo y algorítmicamente rápido, generar representaciones simétricas de configuraciones de bocinas multicanal dentro del modelo.

Toda vez que se ha obtenido dicho polígono regular inscrito dentro del círculo

[1] A esta disposición física se le nombrará a partir de este momento como configuración o array de bocinas.

[2] Véase (Haaser 1987).

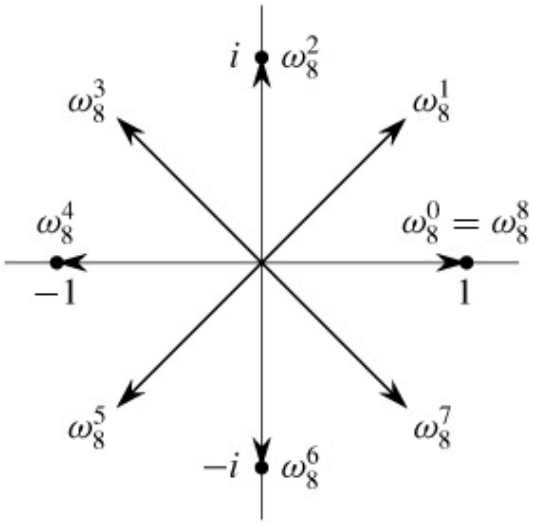


Figura 4.1: Vértices de un polígono regular por raíces complejas

unitario, es posible escalar el radio del círculo por un factor determinado k que sirva para propósitos de cálculos de implementaciones prácticas. A partir de este momento, a dicho círculo escalado se le denominará *círculo de transformación.*

Considérese una configuración primaria simétrica de n vértices con una bocina situada en cada uno de los vértices. Esta configuración delimita un espacio dentro del cual se encuentra situado el espectador. En este punto vale la pena mencionar que se hará uso de la palabra *espectador* para referirse a un único escucha y *audiencia* para el caso de un grupo de escuchas. El espacio físico delimitado por la configuración de bocinas se denominará *espacio audible* mientras que el espacio físico ocupado por la audiencia se denotará como *espacio audiencia*. En la figura 4.2 se ilustran estas ideas.

La configuración de bocinas puede ser considerada como un plano de forma poligonal inmerso en $\mathbb{R}^3$. De esta forma, la configuración puede ser orientada en cualquier ángulo respecto al plano $x-y$, $x-z$ o $y-z$. Lo anterior brinda la oportunidad dentro del modelo, de diseñar arrays de bocinas de una manera totalmente libre permitiendo crear espacios audibles de distintas características.

Debe tenerse en cuenta que hasta este momento los arrays de bocinas están planteados dentro del modelo teórico, las implementaciones prácticas dependerán de factores específicos tanto de logísitca como de adecuación desde el punto de vista composicional. Se propone la siguiente:

Definición 4.1. *La* ***configuración individual*** *o* ***componente*** *de n bocinas colocadas en los vértices del polígono obtenido por la resolución de la ecuación $z^n = 1, z \in \mathbb{C}$, se denota $\mathcal{S}^n = \{s_i : i = \overline{1,n}\}$ y se denomina* ***configuración primaria*** *; ésta puede tener dos orientaciones básicas:*

- *Cuando la configuración se encuentra en un ángulo $\theta = 0$ respecto al plano*

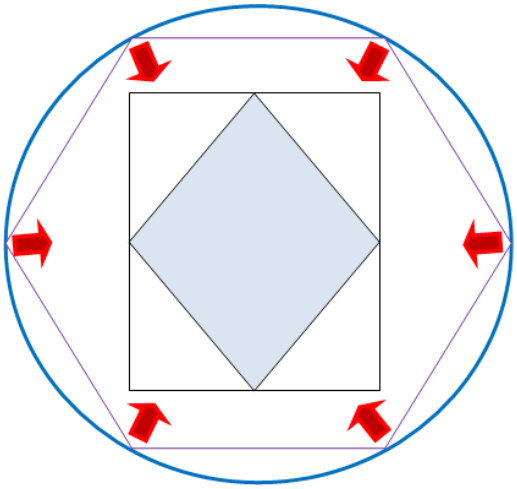


Figura 4.2: Espacio audible y espacio audiencia (rombo), delimitados dentro de una configuración hexafónica

horizontal x-y se le denotará como $\hat{\mathcal{S}}^n_{x,y}$.

- *Cuando la configuración se encuentra en un ángulo* $\phi = 0$ *respecto al plano vertical y-z se le denotará como* $\hat{\mathcal{S}}^n_{x,y}$.

Para $\mathcal{S}^n_{x,y}$ queda implícito el hecho de que se tomará en cuenta únicamente la mitad simétrica del polígono regular la cual estaría inscrita en el semicírculo que comenzaría en un punto ubicado en el piso enfrente del espectador a una distancia d y terminaría en otro punto en el piso detrás del espectador a una distancia d, suponiendo que el espectador se encuentra en el centro de dicho semicírculo. En algunas ocasiones se hará referencia a estas *configuraciones individuales* como *subarrays*.

Nótese que es relativamente fácil calcular la posición espacial respecto a algún origen de cada bocina s_i en alguna de las dos configuraciones primarias anteriores utilizando la conocida norma euclideana o mediante coordenadas polares. De hecho utilizando las segundas, dada alguna configuración $\mathcal{S}^n_{x,y}$ inscrita en algún *círculo de formación* de radio k_2, cualquier $s_i \in \mathcal{S}^n_{x,y}$ tendrá una posición (k_1, ψ) donde ψ representa el conocido parámetro *azimuth*. De igual modo, dada $\mathcal{S}^n_{x,z}$ inscrita en algún *círculo de formación* de radio k_2, cualquier $s_i \in \mathcal{S}^n_{x,z}$ tendrá una posición (k_2, ϕ) donde ϕ representa el conocido parámetro *elevación*.

Dadas estas dos configuraciones primarias, es posible generar configuraciones arbitrarias con cualquier orientación (inclinación) respecto a los planos horizontal y vertical ampliando las posibilidades de inmersión sonora desde el punto de vista composicional y perceptual. Lo anterior puede obtenerse dentro del modelo con rotaciones de copias de cualesquiera de las configuraciones primarias respecto al azimuth y a la elevación. Para poder lograr lo anterior se requieren únicamente tres transforma-

ciones: rotación respecto al plano x-y, rotación respecto al plano xz y traslación. A continuación se explican de forma más detallada dichos movimientos en el espacio los cuales, cabe la pena resaltar, no son otra cosa que movimientos basados en las isometrías o transformaciones rígidas en el espacio; de hecho cualquier movimiento que se desee realizar en el espacio, puede ser expresado matemáticamente mediante rotaciones, traslaciones o reflexiones.[3]

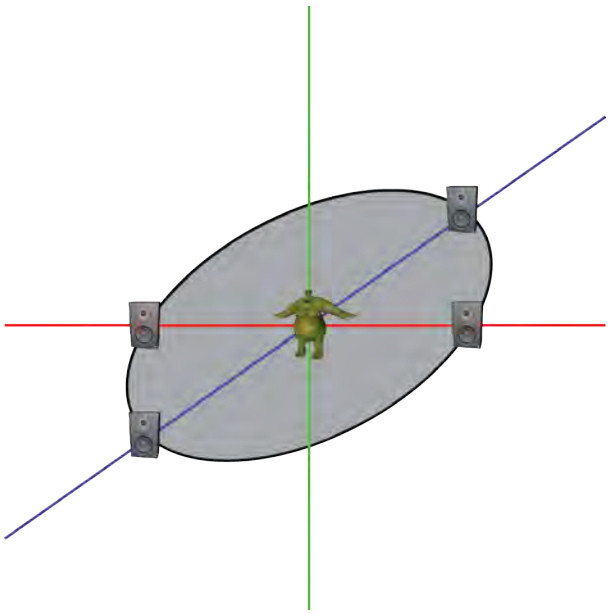


Figura 4.3: Espectador dentro de una configuración horizontal cuadrafónica. Imagen cortesía de Roberto Cabezas

Para la rotación respecto al plano xz, considérese un arreglo primario $\mathcal{S}^n(x, z)$. Supóngase que la primer bocina s_1 se encuentra frente al espectador a una distancia k_1 mientras que la última bocina s_n se encuentra justo detrás del espectador a una distancia k_2. Esto significa que la proyección horizontal del array coincide exactamente sobre el eje x. Considérese ahora un ángulo α respecto a dicho eje y rótese el arreglo entero sobre el plano horizontal x-y con éste ángulo, el resultado será entonces que la primer bocina estará desplazada una cierta distancia del eje x, mientras que la última bocina conservará la misma posición. El desplazamiento anterior puede visualizarse como la proyección de un triángulo isosceles sobre el plano horizontal x-y de lados iguales al radio k_2 y base cuya magnitud estará dada por $b = 2k_2 sen(\alpha/2)$. Para llegar al resultado anterior basta con trazar una bisectriz que divida a la mitad el ángulo de rotación original α, esto provocará que el triángulo isósceles se divida en dos triángulos rectángulos, uno de los cuales puede ser resuelto mediante el teorema de Pitágoras. La rotación respecto al plano xy resulta totalmente análoga.

[3]Para más detalles respecto a las isometrías euclideanas consúltese (Seade Kuri 2005).

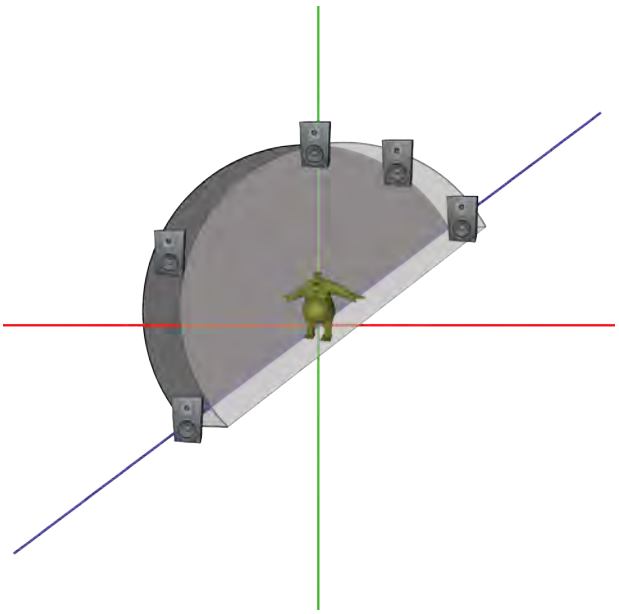


Figura 4.4: Espectador dentro de una configuración vertical cuadrafónica. Imagen cortesía de Roberto Cabezas

Para el caso de la traslación únicamente se elige cualquier punto p del plano poligonal y se traslada por un vector T y de este modo se traslada todo el plano completo (Lang 1988), (Seade Kurri 2005).

Toda vez que se han definido todas las *configuraciones componentes* o *subarrays*, la integración simultánea de todas ellas en el espacio físico o recinto representará la *configuración general o final multicanal* denotada por:

$$S(\rho, \xi) = \{S_i^{n_i} : i = \overline{1, \rho}, \sum_{i=1}^{a} n_i = \xi\}$$

Donde ρ representa el número total de subarrays y ξ el número total de bocinas sumado de todos los subarrays.

4.1.2. Extendiendo las formas de las configuraciones de bocinas

En la sección anterior se generaron arrays de bocinas basados en polígonos regulares inscritos en círculos. Claramente estas configuraciones representan el caso ideal teórico cuya implementación no siempre puede resultar práctica, sobre todo en espacios físicos con características arquitectónicas especiales. Tomando esto en cuenta y con el objetivo de aprovechar el vasto conocimiento actual sobre espacialización, se propone extender las configuraciones circulares a configuraciones arbitrarias basadas

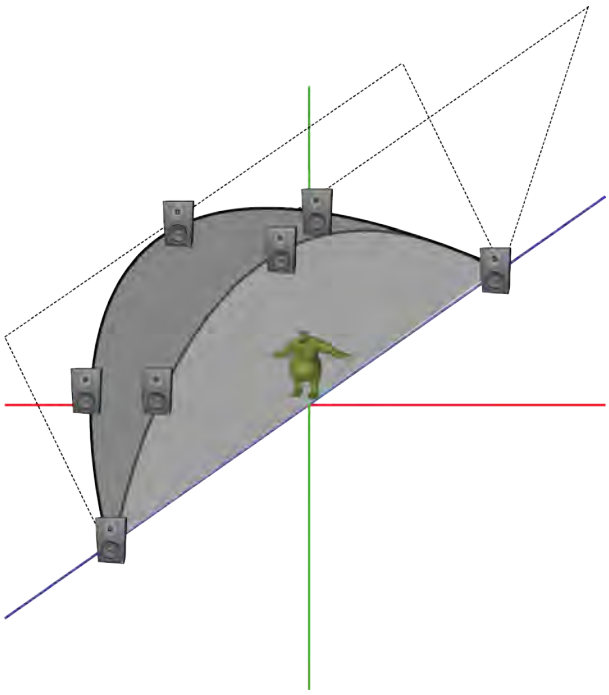


Figura 4.5: Espectador dentro de una configuración compuesta por dos planos rotados respecto al eje vertical. Imagen cortesía de Roberto Cabezas.

en la generación de la forma deseada para el *espacio audible*. En este sentido, las configuraciones de bocinas deberían de ser capaces de ajustarse a la arquitectura del recinto y ofrecer de este modo una alternativa de expansión creativa sonora al compositor o al intérprete. Toda vez que el *espacio audible* se ha definido, el *espacio audiencia* estará a su vez delimitado por él en ciertas características geométricas que funcionarían como límites.[4]

Para generar el espacio audible se propone especificar primero la forma espacial general que debiera tener y en base a ello definir los arrays de bocinas. Si por ejemplo se desea un espacio audible en forma piramidal con base triangular, entonces claramente se deberían colocar tres arrays lineales de bocinas. La primera alternativa que surge para poder ajustar el array teórico a los requerimientos del espacio físico es el llamado proceso de *anamorfósis del círculo en elipse* (Spickler Bergner 2011). Ajustando el semieje mayor de la recién obtenida elipse (junto con el polígono inscrito dentro de ella) a la profundidad total deseada del espacio audiencia y el semieje menor al ancho del mismo, se obtiene una configuración mucho más flexible para con las características reales del recinto o espacio físico.

Lo anterior puede aterrizarse en base a cuatro características de construcción del

[4]El caso inverso es análogo; se diseña primero la forma deseada del espacio audiencia y en base a ello se construye el espacio audible.

espacio audiencia que valdría la pena definir claramente: *proyección horizontal inferior o de piso*, *proyección horizontal superior o de techo*, *proyección vertical frontal o de pared frontal* y *proyección vertical trasera o de pared trasera.* Manejando individualmente estos cuatro parámetros, la creación de configuraciones se amplia de manera considerable permitiendo una mediana generalización tanto conceptual como práctica de la forma simétrica del *array de bocinas.*

Se denominará *altura base* o $h_1(xy)$ a la distancia entre el nivel de suelo del recinto y la proyección horizontal inferior; esto es, el nivel respecto del piso al que se encuentran las bocinas del array horizontal cuando el array correspondiente sea paralelo a éste. De forma correspondiente $h_2(xy)$ representará el nivel sobre el suelo al que se encontrará la proyección de techo. Para el caso en que se deseen inclinaciones específicas para alguno de los dos arrays horizontales correspondientes a las proyecciones de piso o de techo, basta con establecer el ángulo de elevación tal y como se realizó en la sección pasada o definiendo dos alturas donde la diferencia de las mismas generará dicho ángulo, en tal caso $I = I(h_1, h_2, \phi)$, i.e. la inclinación es una función de las dos alturas y del ángulo.

De lo anterior puede observarse que conviene trabajar el diseño de cualquier array $\mathcal{S}^n$ mediante la generación de semiplanos poligonales individuales, no sólo desde el punto de vista formal, teórico y categórico sino al momento de generar algoritmos propios del diseño. Para efectos prácticos y siempre que no exista peligro de confusión, se hará referencia a $\mathcal{S}^n$ tanto a el arreglo físico de n bocinas (ya sea circunscrito o deformado por anamorfósis) como a el semiplano individual definido por el mismo.

4.1.3. Configuración basadas en prismas.

Recuérdese que un prisma es un tipo de poliedro el cual consta de dos polígonos paralelos y congruentes como bases y paralelogramos como caras[5] . La *altura* del prisma está definida por la distancia entre las bases tal y como se muestra en la figura 4.6.

Tomando como punto de partida estas figuras geométricas, es posible hacer modificaciones para ajustar las configuraciones de bocinas a los requerimientos específicos de los recintos o espacios físicos en donde se llevará a cabo la espacialización. Considérese el caso más básico de un sistema cuadrafónico usual. Para dicho ejemplo la *proyección de piso* es exactamente igual a *la proyección de techo*, siendo para ambos casos un paralepípedo, sin embargo la *proyección de pared frontal y trasera* es nula, en este sentido el espacio audible es bidimensional únicamente.[6]

[5]Los lados de los polígonos que funcionan como bases se denominan *arsitas básicas*, mientras que los lados de los paralelogramos que funcionan como caras son llamados *aristas laterales.*

[6]Recuérdese que el *espacio audiencia* tal y como se ha definido en el presente trabajo, es el espacio físico delimitado al interior de la configuración de bocinas dentro del proceso de modelación.

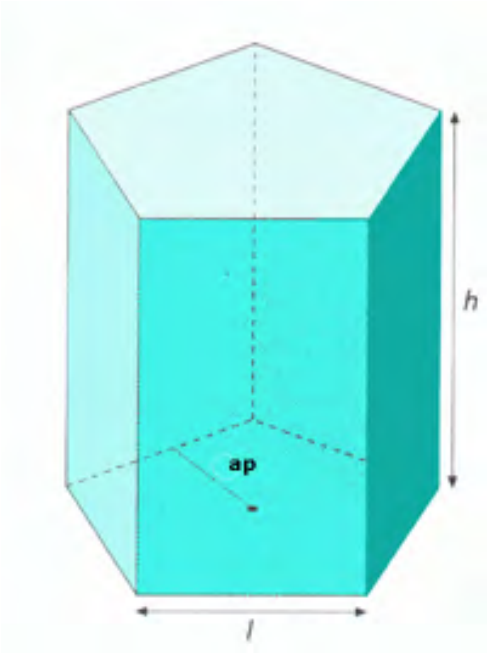


Figura 4.6: Prisma.

Considérese de nuevo la configuración anterior y supóngase que se desea extender el espacio audible para que tenga una forma prismática rectangular. Se define entonces una altura d y se construye dicho cuboide mediante esa altura, como resultado se habrá generado entonces un sistema de espacialización octafónico con proyección de pared frontal y trasera idénticas con forma de cuadriláteros de base igual a la distancia entre s_1 y s_2 y altura d.

Considérese ahora un polígono regular de 8 lados inscrito en una *circunferencia de formación* de radio k_1 y otro polígono regular de de 8 lados inscrito en otra *circunferencia de formación* de radio k_2 con $k_1 > k_2$. Supóngase que debido a los requerimentos del recinto, es necesario ajustar dichos polígonos circunscritos por anamorfósis, a polígonos inscritos en elipses e_1 y e_2 respectivamente, donde e_1 posee un semieje mayor de radio $k_1 + \delta_1 k_1$ y semieje menor de valor $k_1 - \delta_2 k_1$, para $\delta_1 k_1, \delta_2 k_1 > 1$ valores de dilatación y contracción respectivamente. Para el caso de la segunda elipse e_2, se tendrán un semieje mayor de radio $k_2 + \delta_1 k_2$ y un semieje menor de radio $k_1 - \delta_2 k_2$, para $\delta_2 k_1, \delta_2 k_2 > 1$. A continuación se establece que el polígono inscrito en e_1 defina la proyección de piso a una altura base $h_1(xy) = m$, mientras que la segunda elipse e_2 definirá la proyección de techo con una altura $h_2(xy) = s$. Uniendo cada vértice del polígono de proyección de suelo en con su correspondiente en la proyección de techo se obtendrá un espacio audible en forma de pirámide trunca de caras trapezoidales y base en forma de octágono alargado.

Si bien podría parecer innecesario el proceso de diseño de los arrays utilizando ele-

mentos matemáticos como las raíces complejas, las isometrías, la anamorfósis y el uso de poliedros, esto resulta de enorme utilidad computacional al momento de modelar las configuraciones de bocinas para la implementación práctica de acuerdo tanto a parámetros físicos propios del recinto, como perceptuales, composicionales y estéticos. Más aún, el hecho de generar todas las configuraciones en base a polígonos circunscritos en circunferencias (de las cuales algunas podrían ser transformadas a elipses) ofrece la ventaja al momento organizar la información de la distribución sonora como se verá en la siguiente sección.

4.2. Representación matricial de las configuraciones de bocinas.

Una vez que se ha modelado la forma espacial del array de bocinas, para propósitos de tratamiento de la información de espacialización algorítmica, se propone generar una proyección bidimensional de las bocinas en forma matricial para poder estructurar una implementación de flujo de datos como puente entre el sistema dinámico utilizado y la distribución sonora multicanal real.

El propósito de esta matriz es generar una herramienta matemática útil para el compositor al momento de organizar la distribución del sonido para cada bocina. Teniendo esto en mente resulta provechoso si la matriz fuera capaz de representar bidimensionalmente (en la medida de lo posible) la relación espacial de la ubicación de cada bocina.

Sea una matriz $A \in \mathcal{M}_{m\times n}$ con:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & \ddots & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

En el presente contexto cada fila $(a_{k1}a_{k2}\dots a_{kn})$ representará las bocinas que componen cada uno de los semiplanos $\mathcal{S}_k^{n_i}(x,y,z,I) \in S(\rho,\xi)$ que componen la *configuración multicanal general*.[7] Como cada semiplano $\mathcal{S}_i^{n_i}(x,y,z,I)$ contiene un número n_i distinto de bocinas, entonces es necesario definir a n dentro de la matriz del siguiente modo:

$$n = max\{n_i : i = \overline{1,\rho}\}$$

[7]Nótese que dado que ρ es el número de subarrays o configuraciones individuales, entonces necesariamente, tal y como se ha definido la matriz, $m = \rho$.

Por lo tanto, una configuración general con tres subarrays, de cuatro, ocho y nueve bocinas cada uno respectivamente, tendrá la siguiente representación matricial:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & 0 & \dots & 0 \\ a_{21} & a_{22} & \dots & \dots & a_{27} & 0 & 0 \\ a_{31} & a_{32} & \dots & \dots & \dots & \dots & a_{39} \end{pmatrix}$$

A partir de este momento, la matriz anterior será de enorme utilidad dentro del modelo, pues en base a ella se podrán organizar tanto el flujo de datos como el mapping del sistema dinámico. Usualmente dentro de las matrices, los elementos a_{ij} son números de algún tipo, sin embargo dado que cada elemento de la matriz representa en sí una bocina individual, dentro del presente trabajo se propone que cada a_{ij} sea representada por una función. Está función deberá reflejar el comportamiento dinámico de cada salida respecto a dos conceptos fundamentales: *las fuentes sonoras* y *los parámetros sonoros.*

Se consideran dentro de este modelo como *fuentes sonoras* a todos aquellos eventos audibles individuales que componen una estructura definida en el tiempo de acuerdo a las intenciones composicionales del autor. Los parámetros sonoros son todas aquellas características que afectan las fuentes como por ejemplo: amplitud, filtrado, variaciones tímbricas, envolventes, efectos, etc. Una manera sencilla de organizar matemáticamente este concepto de fuentes sonoras, es definir un vector que caracterice la información de dichos eventos audibles; se denominará a este concepto, el *vector de fuentes sonoras.* De este modo considérese cada entrada como una función tal que:

$$a_{ij} = f_{ij}(x) = (f_{ij}^1(\vec{x}), f_{ij}^2(\vec{x}, \dots, f_{ij}^3(\vec{x}))$$

Donde $\vec{x} = (x_1, x_2, \dots, x_n) \in \mathbb{R}^m$ es el vector de fuentes sonoras. Construyamos la función f hacia atrás:

$$\begin{array}{rccc} h_i: & \mathbb{R}^m & \to & \mathbb{R} \\ & \vec{x} & \to & x_i \end{array}$$

Donde h_i es la función proyección que a cada vector de fuentes sonoras le asigna su i-ésima componente. Por otro lado sea:

$$\begin{array}{rccc} g_i: & \mathbb{R} & \to & \mathbb{R}^k \\ & x_i & \to & (y_1, y_2, \dots, y_k) \end{array}$$

La función que a cada fuente sonora le asigna su vector de parámetros sonoros. Para los parámetros sonoros se hace abuso de notación dándose por hecho que se sobreentiende que $y_i = y_i(t)$, i.e. que cada parámetro sonoro es una función del tiempo. Si ahora definimos que cada f_i sea del siguiente modo:

$$f_i = g_i \circ h_i$$

Entonces, f_i es una composición de las funciones g_i y h_i lo que implica que su efecto final sobre el vector de fuentes sonoras será del modo que sigue:

$$f_i(\vec{x}) = g_i(h_i(\vec{x})) = g_i(x_i) = (y_1, y_2, \ldots, y_k)$$

Por lo tanto, cada $f_i(\vec{x})$ representará *la función de espacialización componente* para cada fuente sonora individual la cual reflejará el comportamiento de todos los parámetros que afectan dicha fuente a través del tiempo. Obsérvese entonces que cada bocina es una función vectorial cuyos componentes son funciones anidadas que representan cualquier número de fuentes sonoras afectadas por *cualquier número de parámetros sonoros*. Todo el desarrollo anterior establece una formalidad breve que supone una base firme en la estructuración del modelo generalizable si no a cualquier caso de creación o composición, sí a una amplia gama de ellos.

Para ejemplificar la idea anterior supóngase que se tiene una configuración general que consiste de un único subarray de ocho bocinas colocado a 1 metro del nivel del suelo y paralelo al mismo:

$$S(1, 8) = \{\mathcal{S}^8(0, 0, 1, 0)\}$$

Supóngase además que el compositor desea espacializar ruido blanco de una forma más o menos periódica y para tal efecto elige que la función de espacialización para todas las bocinas sea $f(x, t) = sin(t)$; función con la que controlará la amplitud de su fuente sonora en cada bocina. Dado que $sin(t) \in [-1, 1]$ para $t \in [0, 2\pi]$, el compositor realiza un sencillo escalamiento tal que $sin(t) \in [0, 1]$ para $t \in [0, 2\pi]$. Su función de espacialización para cada bocina (i, j) y una fuente sonora $x_1 = x$ tendrá la forma:

$$f_i(x) = f(x) = sin(t)$$

Ahora bien, el compositor realiza implementaciones en su DAW para que cada canal de salida (bocina) tenga como fuente sonora el ruido blanco variando en amplitud de acuerdo a la función antes mencionada. Para realizar esto necesita establecer una escala de línea de tiempo sobre la cual la función de espacilización esté operando y que sea apropiada para sus intereses composicionales; esto no es otra cosa que establecer la frecuencia mediante modulación. $sin(t)$.[8] Si la función moduladora está

[8]Recuérdese que la frecuencia es simplemente la unidad que mide los ciclos por segundo a los que vibra el medio que transmite ese sonido, entonces el proceso descrito es lo que comunmente se conoce como modulación de amplitud o AM.

dada por:

$$\chi(t) = Asin(\omega t)$$

Entonces la función de espacialización con amplitud modulada estará dada por:

$$f_i(x) = f(x) = \chi(t)$$

Hasta este punto, el compositor parece no estar del todo satisfecho con su pieza y decide incluir nuevos parámetros sonoros en su función de espacialización por lo que comienza aplicando síntesis substractiva a cada canal por separado de modo tal que para cada canal define un BPF (Band Pass Filter) con un ancho de banda específico que además es dinámico en el tiempo, es decir, cuyos límites inferior y superior se van moviendo en el tiempo. Su función de espacialización tiene ahora la siguiente forma:

$$f_i(x) = g_i(h_i(\vec{x})) = g_i(x_i) = (y_1(t), y_2(t))$$

donde $y_1(t) = \chi(x,t)$ y $y_2(t) = BPF(x,t)$. Establecido todo lo anterior, es posible reescribir la *matriz de espacialización* a una forma más general en la que cada entrada de la matriz corresponda a las funciones de espacialización:

Definición 4.2. *Sea* $S(\rho,\xi) = \{S_i^{n_i} : i = \overline{1,\rho}, \sum_{i=1}^{a} n_i = \xi\}$ *una configuración general multicanal. Sea* $n = max\{n_i : i = \overline{1,\rho}\}$ *y* $\vec{x} \in \mathbb{R}^k$ *un vector de k fuentes sonoras, entonces la* ***matriz de espacialización*** $A \in \mathcal{M}_{\rho\times n}$ *es de la forma:*

$$A = \begin{pmatrix} f_{11}(\vec{x}) & f_{12}(\vec{x}) & \dots & f_{1n}(\vec{x}) \\ f_{21}(\vec{x}) & \ddots & & f_{2n}(\vec{x}) \\ \vdots & & \ddots & \vdots \\ f_{m1}(\vec{x}) & f_{m2}(\vec{x}) & \dots & f_{mn}(\vec{x}) \end{pmatrix}$$

donde $f_{ij}(\vec{x})$ *son las* ***funciones de espacialización*** *para cada salida individual y cumplen que:*

$$f(x) = \begin{cases} 0 & j > n_i \\ (f_1(\vec{x}), \dots, f_m(\vec{x})) & j \leq n \end{cases}$$

Todo el desarrollo matemático y formal de esta sección estuvo orientado a la construcción de una base teórica de lo que se podría denominar *mapeo individual multicanal*, en el cual cada bocina es tratada como una función independiente de los parámetros y de las fuentes sonoras. Esta perspectiva puede ser muy útil cuando se desea generar trayectorias espaciales sonoras muy específicas para cada canal y pueden resultar útiles en configuraciones relativamente pequeñas. Sin embargo, para

configuraciones generales multicanal con mayor número de componentes tal acercamiento podría tornarse muy complicado de manejar debido al elevado número de especificidad requerido que está dado directamente por el número total de bocinas dentro de la configuración. En tales casos, definir una sóla función de espacialización que controle parámetros y fuentes sonoras de manera general dentro del arreglo, puede convertirse en una solución óptima; la siguiente sección estará dedicada a desarrollar dicho concepto.

4.3. Mapeo general centralizado.

En esta sección se describirá una segunda forma de definir la espacialización, la cual será mediante una única función centralizada que controle toda el flujo de datos dentro de la configuración general. La idea básica es que esta función tome valores de cualquier fuente de información y los mapee dentro de la configuración para que distribuya el comportamiento de las fuentes sonoras y sus parámetros a través de cada bocina.

Para construir dicha función comencemos con un caso simple: una única fuente sonora para ser espacializada en una configuración octafónica de un nivel. Para poder generar una única función de espacialización centralizada dentro de este contexto, son necesarios tres parámetros: tiempo, posición dentro de la configuración y amplitud. Formemos un vector tres-dimensional a partir de lo anterior: $w(x) = (t, z_t(x), A_t(x))$ para $t \in \mathbb{Z}$. Este vector nos dará la información sobre la posición donde se encuentra y la amplitud a la que está sonando la fuente sonora en cada instante de tiempo.

Se propone el siguiente procedimiento para generar las localizaciones dentro de la configuración general multicanal. Considérese un intervalo cerrado $I \subset \mathbb{R}^+$ de donde z_t tome valores para indicar la posición de la fuente sonora en el instante de tiempo t. A continuación es necesario establecer una forma adecuada de utilizar el intervalo anterior como representativo de la posición z_t. Se propone que dicho intervalo se divida en partes iguales de acuerdo al número de bocinas y que cada bocina esté representada por la subdivisión exacta relativa. Para ser más claros al respecto, sea $I = [0, s], \quad s \in \mathbb{N}$ donde s es un número adecuado de acuerdo a la cantidad total de bocinas n. Entonces $s = s\frac{n}{n}$, lo que nos indica la forma en que quedará dividido el intervalo I:

$$I = \bigcup_{i=0}^{n-1} [s\frac{i}{n}, s\frac{i+1}{n}] = [0, s\frac{1}{n}] \cup [s\frac{1}{n}, s\frac{2}{n}] \cup \ldots \cup [s\frac{n-1}{n}, s\frac{n}{n}]$$

Los puntos $\{s\frac{i}{n} : i = \overline{1, n}\} = \{s\frac{1}{n}, s\frac{2}{n}, \ldots, s\frac{n}{n}\}$ representan las posiciones actuales de las bocinas consideradas como vértices del polígono, mientras que puntos intermedios dentro de los intervalos anteriores no son otra cosa que la posición relativa sobre las

arsitas del polígono; todo esto dentro del modelo y para propósitos de cálculo y desarrollo algorítmico. Denominamos a I como el *intervalo de localización espacial*. Entonces, para una configuración de 5 bocinas, el *intervalo de localización espacial* tendrá la siguiente representación:

$$I = [0, 1] = [0, 1/5] \cup [1/5, 2/5] \ldots [4/5, 5/5]$$

Para una configuración de 13 canales el *intervalo de localización espacial* estará formado por:

$$I = [0, 3] = [0, 3\frac{1}{13}] \cup [3\frac{1}{13}, 3\frac{2}{13}] \ldots [3\frac{12}{13}, 3\frac{13}{13}]$$

El objetivo de aumentar el rango del intervalo conforme aumenta el número total de bocinas es con fines de evitar subdivisiones demasiado pequeñas que surgirían de una homegenización del intervalo unidad.

Si asumimos que la amplitud toma valores en el intervalo $[0, 1]$ como normalmente sucede, entonces puede describirse más formalmente el vector $w(s, t)$ como:

$$w(s, t) = (t, z_t(s), A_t(s)) \in I \times T \times [0, 1]$$

donde $I \subset \mathbb{R}^+$ y $T \subset [0, 4] \subset \mathbb{R}$. Recalcamos el hecho de la discretización del tiempo para fines de implementación algorítmica, sin embargo se aclara de igual modo que en el contexto del modelo se generaliza la posibilidad de tener cuantizaciones de tiempo no enteras así como pasos o retardos menores que 1 e irregulares. Desde el punto de vista computacional, cada componente del vector $w(s, t)$ está formada por una serie de tiempo alojada en un array de longitud determinada cuya información proviene de algún flujo de datos establecido.

Hasta ahora se tiene una modelación clara sobre el movimiento espacial de una fuente sonora a través de una configuración compuesta por un único subarray. Dentro de este caso especial, las trayectorias del sonido viajan a lo largo de las aristas del polígono definido por la configuración, la cual a su vez delimita el espacio audible.

Se añade ahora un tercer parámetro sonoro el cual establece la idea de *profundidad*, permitiendo de este modo que las trayectorias viajen a lo largo de las aristas, así como dentro y fuera del espacio audible para un cierto rango límite; por supuesto esto se obtiene de la forma más básica mediante aplicación de *reverberación* a la fuente sonora.

De entre los distintos parámetros que pueden afectar el efecto de percepción del sonido por reverberación, el primordial es el llamado *mix* entre la señal original y

la señal afectada (*dry* y *wet*, respectivamente) ya que es este el que proporciona en mayor medida al escucha, la sensación de profundidad o lejanía y de espacio acústico. Otros dos parámetros importantes son el *room* y el *damping*; el primero proporciona el nivel de reflexión o eco de la fuente procesada generando sensaciones específicas del tamaño del recinto acústico virtual, mientras que el segundo tiene que ver con aspectos de discriminación de bandas de frecuencia respecto al desvanecimiento y presencia en el ratio *dry/wet*.[9]. Por el momento bastará con añadir reverberación con único control; el mix:

$$w(s,t) = (t, z_t(s), A_t(s), RevMix_t(s)) \in I \times T \times [0,1] \times [0,1]$$

Una vez planteada la idea particular anterior y de manera análoga, es posible ir incluyendo cualesquiera parámetros sonoros deseados para cualquier número de fuentes sonoras y de este modo hacer una generalización para establecer el concepto de *mapeo centralizado* de una manera más firme:

Definición 4.3. *Sea* $S(\rho,\xi) = \{S_i^{n_i} : i = \overline{1,\rho}, \sum_{i=1}^{a} n_i = \xi\}$ *una configuración general multicanal. Sea* $n = max\{n_i : i = \overline{1,\rho}\}$ *y* $\vec{s} \in \mathbb{R}^k$ *el vector de* k *fuentes sonoras,* $y(\vec{s},t) = (y_1, y_2, \ldots, y_m) \in \mathbb{R}^m$ *el vector de parámetros sonoros sobre el vector de fuentes sonoras. Para cada* $i \in [1,m]$ *se define el* ***vector de estado*** *como aquel que tiene la siguiente forma:*

$$w(t, s_i,) = (t, z_i(s,t), A_i(s,t), y_{1(t)}, y_2(t), \ldots, y_m(t))$$

donde $z_i(s,t)$ *representa la posición lineal de la fuente sonora sobre las aristas del polígono definido por cada configuración componente o subarray en el instante de tiempo* t *y* $A_i(s,t)$ *la amplitud de dicha fuente en ese mismo instante de tiempo.*

Obsérvese que la principal diferencia entre la estructuración de la sección anterior y la presentada aquí, radica en los componentes que se consideran primordiales o básicos para generar la espacialización. En el primer caso dicho componente era cada bocina individual considerada como una función de espacialización, mientras que en el caso actual ese papel lo juegan las fuentes sonoras descritas por un arreglo matricial multidimensional.

4.4. El flujo de datos.

Se ha construido hasta ahora una sustentación teórica sobre configuraciones, arrays y mapeos, basada en conceptos y herramientas matemáticas que serán de enorme utilidad no sólo desde el punto de vista categórico y formal sino al momento de generar algoritmos y código fuente en algún lenguaje de programación, puesto que todo este trabajo representa los pilares y la estructura fundacional de todo el modelo.

[9]Para mayor detalle al respecto consúltese (Errede 2002)

En esta sección se describirá la manera en que se trabajara el flujo de información o data streaming dentro del modelo, la cual estará basada en análisis de series de tiempo. Se generará además una formulación detallada sobre la manera de definir el mapping entre la fuente de información y el modelo de espacilalización.

4.4.1. Series de tiempo.

Se mencionó anteriormente con cierta frecuencia el concepto de series de tiempo el cual es una colección de valores en pares. Para cada instante de tiempo t dado, existe un valor específico asociado x_t, esto genera pares de valores (t, x_t) los cuales pueden ser fácilmente graficados bidimensionalmente. El análisis de series de tiempo incluye entre sus más sobresalientes características, la búsqueda de correlación entre los términos, temporalidad, estacionareidad, etc. sobre todo cuando se busca estudiar el fenómeno o el proceso que las genera, proceso que además es modelado mediante ecuaciones en diferencias. Cada valor dado aparece en un instante de tiempo específico y por lo general los instantes de tiempo tienen diferencias iguales, i.e. $t_1 - t_0 = t_1 - t_2 = \ldots = t_i - t_{i+1} = \Delta t$, estas diferencias son llamadas *retardos* o pasos. Sin embargo, desde el punto de vista de la composición y para el objetivo de generalización del presente trabajo, resulta más útil poder ampliar esta visión y ofrecer la posibilidad de manejar una perspectiva basada en una línea de tiempo independiente de tal modo que las series de tiempo puedan ser construidas usando pasos de tiempo o retardos irregulares. Si además estos retardos sobre la línea de tiempo se mantienen con diferencias irregulares y menores a 1 en valor absoluto, la serie mantiene su forma general pero presenta variaciones internas que pueden ser muy bien aprovechadas para fines composicionales.

La forma más directa de obtener el comportamiento descrito es manejar la línea de tiempo como una serie de datos independiente de tal modo que la serie de tiempo estará compuesta por dos series de datos independientes, pero con ciertas restricciones sobre la línea de tiempo (p.e. que sus elementos tengan valor absoluto menor a 1). De este modo podemos definir que una serie de tiempo es:

$$\{\eta_i = [t_i, x_i(t_i)] : i \in \mathbb{Z}, t_i \in [0, 1], x_i(t_i) \in \mathbb{R}\}$$

4.4.2. El índice de tiempo.

La indización es un aspecto importantísimo cuando se trabaja con series de tiempo. Se comentó en la sección anterior que una serie de tiempo en el presente contexto, podía ser construida a partir de dos órbitas cualesquiera, analicemos con un poco más de detalle esta característica.

Una serie de tiempo puede ser vista básicamente como una sucesión de pares or-

denados que describen el valor del fenoméno a observar en instante de tiempo dado, donde por lo general los intervalos de tiempo son regulares e idénticamente espaciados. En este sentido la serie:

$$\{[t_0, x(t_0)], [t_1, x(t_1)], \ldots, [t_n, x(t_n)]\}$$

denota información sobre el comportamiento que un fenómeno tuvo en cada uno de los instantes t_i. Cuando se dice que los eventos están idénticamente espaciados esto significa únicamente que:

$$|t_i - t_{i+1}| = h, \quad \forall i \in \mathbb{R}^+$$

Es decir, que la diferencia de tiempo entre un suceso y otro es constante[10]. Ahora bien, dentro del presente contexto, los intervalos temporales o *pasos* pueden ser considerados como el lapso que tarda el sistema de cambiar de un estado a otro y de este modo ampliamos la restricción anterior permitiendo ahora que los pasos sean irregulares y distintos entre ellos. Esto por supuesto ofrece una perspectiva composicional más flexible dotando a los datos de un significado tangible. De este modo aparecen numerosas posibilidades para trabajar con el flujo de datos de cualquier fuente de información para formar series de tiempo con un significado preciso:

$$|t_i - t_{i+1}| = h_i, \quad \forall i, h_i \in \mathbb{R}^+$$

Ahora, cada paso[11] puede ser distinto y tener el valor que el(la) compositor(a) decidan. La forma más básica de variar el índice de tiempo sería mapeando de forma escalada, la misma órbita utilizada como flujo de datos con la siguiente lógica:

$$\begin{array}{rcc|} t_0 & = & f_0(x_0) & |0 \\ t_1 & = & 1 + f_1(x_0) & |1 \\ \vdots & & \vdots & \vdots \\ t_n & = & n + f_n(x_0) & |n \end{array}$$

donde la tercer columna indica el número del paso. Por supuesto nada impide que ajustemos el índice temporal a cualquier capricho algorítmico o matemático a nuestro gusto:

$$\begin{array}{rcc|} t_0 & = & f_0(x_0) & |0 \\ t_1 & = & p(x_0) + f_1(x_0) & |1 \\ \vdots & & \vdots & \vdots \\ t_n & = & p(x_{n-1}) + f_n(x_0) & |n \end{array}$$

[10] Por lo general y a modo de simplificación, se toman diferencias o *pasos* unitarios; $h = 1$.

[11] i.e. el tiempo que tarda el sistema en pasar de un estado al siguiente.

donde $p(x_i)$ puede ser cualquier función de probabilidad derivada de algún proceso estocástico, obteniendo como serie de datos final:

$$\begin{array}{cc} f_0(x_0), & f_0(x_0) \\ p(x_0) + f_1(x_0), & f_1(x_0) \\ \vdots & \vdots \\ p(x_n) + f_n(x_0), & f_n(x_0)|n \end{array}$$

Obsérvese que si bien se tiene plena libertad para formar la serie del índice temporal, existe una única restricción: *el tiempo no perdona*. Entonces, la sucesión que denota la indización siempre debe ir aumentando, por lo que se debe tener especial cuidado de escalar apropiadamente cualesquiera flujos de datos a utilizar. De este modo, dentro del vector de estado, el parámetro t puede generalizarse a una función cualquiera que depende directamente del paso o índice temporal: $T(t)$, a esta función se le denotará en lo sucesivo como *la función de indización de tiempo* o simplemente *función de indización* .[12]

Obsérvese además, la distinción entre *iteración* e *instante de tiempo*. La primera alude al número de veces que la función que define el sistema dinámico ha sido evaluada, mientras que la segunda hace referencia precisamente a la función de indización, esto es, la duración temporal del cambio del sistema dinámico de un estado a otro en relación a cada iteración. De este modo, $t = 1, 2, \ldots$ representa el número de iteración y $T(t)$ el instante de tiempo en que dicha iteración es llevada a cabo.

Nótese también que es necesario asociar una *función de indización* distinta a cada parámetro sonoro si se pretende establecer una generalización y una flexibilidad de manipulación de los mismos, de tal modo que para cada fuente sonora, cada parámetro pueda tener un comportamiento independiente.

La idea anterior conlleva a generar series de tiempo independientes para cada parámetro dentro del vector de estado construidas mediante indizaciones distintas. Se construirá a continuación dicho proceso para propósitos de clarificación. Sea $z_i(t)$ el vector de posición espacial de la fuente sonora i en la iteración t, y sea además $T_{z_i}(t)$ una función de indización asociada a dicho parámetro, i.e. $T_{z_i}(t)$ gobierna la forma en que $z_i(t)$ va cambiando con respecto al tiempo. Por lo tanto es necesario construir una serie de datos de los dos flujos anteriores:

$$\mathcal{Z}_i = \{[T_{z_i}(0), z_i(0)], [T_{z_i}(1), z_i(1)], [T_{z_i}(2), z_i(2)], \ldots, [T_{z_i}(m), z_i(m)], \ldots\}$$

Del mismo modo se construyen las series para los parámetros restantes:

$$\mathcal{A}_i = \{[T_{A_i}(0), A_i(0)], [T_{A_i}(1), A_i(1)], [T_{A_i}(2), A_i(2)], \ldots, [T_{A_i}(m), A_i(m)], \ldots\}$$

[12]En el ejemplo anterior dicha función está dada por: $T(t) = tp(x) + f_t(x_0)$.

$$\mathcal{Y}_{1_i} = \{[T_{y_{1_i}}(0), y_{1_i}(0)], [T_{y_{1_i}}(1), y_{1_i}(1)], [T_{y_{1_i}}(2), y_{1_i}(2)], \ldots, [T_{y_{1_i}}(m), y_{1_i}(m)], \ldots\}$$

$$\vdots$$

$$\mathcal{Y}_{n_i} = \{[T_{y_{n_i}}(0), y_{n_i}(0)], [T_{y_{n_i}}(1), y_{n_i}(1)], [T_{y_{n_i}}(2), y_{n_i}(2)], \ldots, [T_{y_{n_i}}(m), y_{n_i}(m)], \ldots\}$$

Nótese que estas series de tiempo están denotando para cada iteración, el flujo de información respecto al parámetro en cuestión y su dependencia a la función de indización. Por ejemplo, $\mathcal{A}_i$ es la serie de datos del parámetro de la amplitud cuyo primer miembro es el par ordenado $[A_i(0), T_{A_i}(0)(0)]$; este elemento ofrece la información de: *la amplitud de la fuente sonora i en el instante de tiempo $T_{A_i}(0)$ para la primer iteración del sistema dinámico asociado a cada elemento, o en su defecto, la primer muestra de la fuente de información.* Entonces, suponiendo que el flujo de información de los valores de $A_i(t)$ y de $T_{A_i}(t)$ provienen de fuentes distintas, el único eslabón en común y que mantiene coherencia en la serie es precisamente el contador o número de iteración t

Con todo lo anterior en mente, el vector de estado para la fuente sonora i, en su forma general, estará dado por:

Definición 4.4. *Sea $\vec{s} = (s_1, \ldots, s_n)$ el vector de n fuentes sonoras, $z_i(t)$ el vector dinámico de posición espacial, $A_i(t)$ el vector dinámico de la amplitud e $y_{i_1}(t), \ldots, y_{i_m}(t)$ los vectores dinámicos de los m parámetros sonoros restantes para cada fuente sonora $i = \overline{1, n}$. Sean $T_{z_i}(t)$, $T_{A_i}(t)$, $T_{y_{i_1}}(t), \ldots, T_{y_{i_m}}(t)$, las funciones de indización asociadas a cada parámetro respectivo, entonces la* ***forma general del vector de estado*** *para cada iteración t y cada fuente sonora i, estará dada por la siguiente definición:*

$$w(t, s_i,) = ([T_{z_i}(t), z_i(t)], [T_{A_i}(t), A_i(t)], [T_{y_{i_1}}(t), y_{i1}(t)], [T_{y_{i_2}}(t), y_{i2}(t)], \ldots, [T_{y_{im}}(t), y_{im}))$$

Para el caso particular en que se utilice una sóla función de indización para gobernar simultáneamente todos los parámetros, la forma anterior se reduce a:

$$w(t, s_i,) = (T(t), z_i(t), A_i(t), y_{1_i}, y_{2_i}, \ldots, y_{k_i}))$$

Obsérvese que en tal caso, todos los parámetros van a cambiar de valor al mismo tiempo, lo que puede resultar en efectos sonoros interesantes aunque posiblemente predecibles después de cierto tiempo significativo.

Capítulo 5

Espacialización mediante sistemas dinámicos

Ahora que se ha desarrollado todo la estructura matemática y categórica, el siguiente paso consiste en formular el mapping de los flujos de datos desde el sistema dinámico hacia el sistema de espacialización. De hecho, como se mencionó anteriormente, es vitualmente posible mapear cualquier información proveniente de cualquier fuente de datos hacia el sistema de espacialización estableciendo una estructura general de mapping bien fundamentada.

Todo este capítulo estará dedicado a mostrar el proceso teórico que se propone para establecer el nivel del modelo correspondiente a la tercer etapa: *flujo de datos* entre la *fuente de información* y el sistema de espacialización; por supuesto se puntualizarán dichos conceptos.

5.1. Construcción de las órbitas sonoras.

En esta sección se desarrollará la forma en que se construirán las órbitas provenientes de los sistemas dinámicos que después serán mapeadas al sistema de espacialización. Para ello se comenzará retomando la forma general de un sistema dinámico[1] :

$$x'(t) = \alpha x(t)$$

o en términos discretos:

$$x_n = \alpha f(x_n)$$

Para $x \in \mathbb{R}^n$ y $\alpha, t \in \mathbb{R}$. Dado que uno de los objetivos del presente trabajo es la generalidad, de modo tal que el modelo pueda aplicarse a cualquier sistema dinámico o incluso a cualquier fuente de información, la estructuración teórica resulta necesaria

[1] Véase el apéndice A3

de este modo. Consíderese la órbita hacia adelante de este sistema para un punto inicial x_0:

$$\mathcal{O}_f(x_0) = \{f^k(x_0) : k \in \mathbb{N}\}$$

Esta órbita representa el *flujo de datos* y el sistema dinámico previamente expuesto es precisamente la *fuente de información*. Con esta idea en mente, cualquier *modelo teórico* puede verse como la *fuente de información* y su *output* como el *flujo de datos*. Existen dos formas de trabajar con dicha órbita: la primera consiste en un forma de *espacialización algorítmica en tiempo real* y la segunda está orientada a una manera de *espacialización algorítmica predeterminada*.

En el caso de *espacialización algorítmica predeterminada* es necesario considerar un número fijo de iteraciones para generar las órbitas por lo que se tomará en cuenta sólo un subconjunto finito de dicha órbita:

$$\overline{\mathcal{O}_f(x_0)} = \{f^k(x_0) : k = \overline{1, n}, n \in \mathbb{N}\} \subset \mathcal{O}_f(x_0)$$

Se denominará a este conjunto la *semiórbita hacia adelante* o siempre que no haya lugar a confusión únicamente se empleará el término *semiórbita*.

La órbita anterior representa el comportamiento del sistema para condiciones iniciales dadas y para un punto inicial en específico. Una de las virtudes de los sistemas dinámicos es que es posible generar una enorme cantidad de información distinta con una eficiencia computacional relativamente económica toda vez que se ha generado un algoritmo base. Entonces, conociendo el comportamiento del sistema, ya sea por métodos análiticos o por simple inspección del diagrama de bifurcación, es posible especificar las condiciones iniciales ideóneas para que nuestras órbitas convertidas en series de datos, tengan características definidas respecto a sus valores.

Recuérdese que en el capítulo anterior se estableció que las series de tiempo estaban compuestas por dos series de datos independientes en donde la línea del tiempo podía ser formada a partir de cualquier fuente de información siempre y cuando cumpliera ciertas características. Entonces, los elementos de las órbitas forman series de datos individuales que cuando se ordenan en pares forman una serie de tiempo específica:

Definición 5.1. *Sean $\mathcal{O}_f(x_a, \alpha_a)$ y $\mathcal{O}_f(x_b, \alpha_a)$ dos órbitas generadas por el sistema:*

$$x_n = \alpha f(x_n)$$

con condiciones iniciales:

$$x = x_a, \alpha = \alpha_a$$

$$\&$$

$$x = x_b, \alpha = \alpha_b$$

La serie de tiempo generada por estas dos órbitas es la sucesión de pares ordenados dada por:

$$\{(f^k(x_a), f^k(x_b)) : k = 0, 1, \ldots, n, \ldots\}$$

$$= \{(f^0(x_a), f^0(x_b)), (f^1(x_a), f^1(x_b)), \ldots, (f^k(x_a), f^k(x_b)), \ldots\}$$

Análogamente sean $\overline{\mathcal{O}_f(x_a)}$ *y* $\overline{\mathcal{O}_f(x_b)}$ *dos semiórbitas formadas por el sistema anterior bajo condiciones iniciales dadas, etonces la* ***serie de tiempo*** *generada por estas dos órbitas es la sucesión de pares ordenados dada por:*

$$\{(f^k(x_a), f^k(x_b)) : k = \overline{1, n}\}$$

$$= \{(f^0(x_a), f^0(x_b)), (f^1(x_a), f^1(x_b)), \ldots, (f^n(x_a), f^n(x_b)), \}$$

De acuerdo a la definición anterior una serie de tiempo está formada por cualesquiera dos series de datos independientes obtenidas de la evaluación del sistema con cualesquiera condiciones iniciales y como se ha venido mencionando, dichas series de datos pueden de hecho, provenir no sólo de algún sistema dinámico sino de cualquier fuente de información.

En este punto las isometrías vuelven a entrar en juego pero ahora con el fin de transformar las series de datos obtenidas de una manera rápida y coherente. Recuérdese que las isometrías son transformaciones rígidas, esto significa que preservan la distancia por lo que cuando se aplican sobre algún objeto matemático, éste no se deforma sino que únicamente cambia su posición respecto a algún eje coordenado en el plano o en el espacio. Dicha característica aplicada a las series permite obtener series de datos secundarias con variaciones coherentes e interesantes respecto a la serie original.[2] Se denotará como:

$$T_{\vec{a}}[\mathcal{O}_f(x_0)]$$

a la serie de tiempo transformada por traslación por un vector $\vec{a}$. El vector $\vec{a}$ es el que definirá la naturaleza de la traslación la cual puede ser horizontal (si el vector tiene componente $y = 0$), vertical (si el vector tiene componente $x = 0$) o mixta (si el vector tiene ambas componentes distintas de cero).

$R_x[\mathcal{O}_f(x_0)]$. Es la serie transformada mediante reflexión respecto al eje x.

[2]Véase el capítulo siguiente.

$R_y[\mathcal{O}_f(x_0)]$. Es la serie transformada mediante reflexión respecto al eje y.

$S_{a,b}[\mathcal{O}_f(x_0)]$. Es la serie transformada por escalamiento con límite inferior de valor a y límite superior igual a b.

Ahora bien, el compositor necesitará toda una gama de opciones en el estrato de la fuente de información para poder establecer mappings de acuerdo a sus necesidades, en este sentido se define el *conjunto orbital general* como aquel que contiene todas las órbitas y/o semiórbitas originales y/o transformadas y que son potencialmente útiles para el proceso de espacialización:

$$\mathfrak{O} = \{\overline{\mathcal{O}_f^i(x_0)} :\} \bigcup \{\mathcal{O}_f^i(x_0)\}$$

5.2. El mapping hacia la matriz de espacialización.

Ya que se han construido las series de tiempo a partir de las series de datos obtenidas por alguna fuente de información que en este caso resulta ser cualquier sistema dinámico, entonces es necesario poder mapear este flujo de datos a la matriz de espacialización previamente construida, de tal forma que se genere un puente informacional entre ambos estratos; la fuente y el sistema de espacialización.

Una afirmación que realiza el autor de este trabajo es que a final de cuentas, la creación de cualquier obra basada en medios algorítmicos es posible gracias a un sólo elemento: el mapping; que es precisamente este puente informacional. Entonces el tipo de mapping variará de acuerdo a las necesidades, planteamientos estéticos o hasta limitaciones del compositor, por lo que no resulta tan descabellado afirmar que existen tantos tipos de mappings como piezas realizadas y por realizar. Se proponen aquí ciertas opciones de mapping que sólo son algunas de entre muchos posibles. Pero antes de presentar la propuesta de mapping, es necesario resaltar aspectos importantes sobre los índices de tiempo.

5.2.1. Mapping mediante una órbita única.

En este método la idea es generar una órbita única ya sea para espacialización en tiempo real o predeterminada y formar el vector de estado a partir de los elementos de la misma. Sea:

$$\mathcal{O}_f(x_0)$$

una órbita generada a partir del sistema dinámico

$$x_n = \alpha f(x_n)$$

con condiciones iniciales:

$$x = x_0, \alpha = \alpha_0$$

Sea además $\vec{s} = (s_1, s_2, \ldots, s_m) \in \mathbb{R}^m$ el vector de fuentes sonoras. Recuérdese que para cada $i \in [1, m]$, la función que a cada fuente sonora le asigna su vector de parámetros dinámico temporal es:

$$\psi : \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}^n$$

tal que $\psi(s_i, t) = \psi_t(s_i) = (y_{1_i}(t), \ldots, y_{k_i}(t))$. Recuérdese además que $z_i(\vec{s}, t) = (r_i, \theta_i, \phi_i)_t$ es la función que asigna a cada fuente sonora i del vector de fuentes sonoras $\vec{s}$, su posición espacial dentro de la configuración de bocinas en coordenadas esféricas, entonces, el vector de estado es:

$$w(t, s_i,) = (T(t), z_i(t), A_i(t), y_{1_i}, y_{2_i}, \ldots, y_{k_i}))$$

tal y como se definió en la sección anterior para una configuración general de bocinas $\mathcal{S}(\rho, n)$ con representación matricial $A \in \mathcal{M}_{\rho \times n}$. Obsérvese que se tienen a lo más $m + 3$ parámetros donde:

$$m = max\{m_i : i \in [1, n]\}$$

para n el mayor número de bocinas presentes en los subarrays. Recuérdese que se mencionó con anterioridad que la característica más importante al momento de la creación electroacústica mediante medios algorítmicos es precisamente *el mapping*, y y de hecho existen innumerables formas de mapear el flujo de datos de una fuente de información específica a cualquier conjunto de paramétros sonoros. Es de hecho este mapping el que ofrece en cierta medida el sello característico del trabajo personal del compositor. Teniendo en cuenta este punto, resulta clara la imposibilidad de establecer en este trabajo un método general de mapping que pretenda ser de validez categórica, en cambio, se presentan propuestas específicas que sin duda podrán ser aprovechadas por el lector, ya sea que se apliquen literalmente o se generen variaciones en base a éstas. Se comenzará pues, tal y como dice el título de la sección, con la forma más inmediata y básica que es la de utilizar una sóla órbita de la fuente de información como flujo de datos y mapear la misma órbita a todos los componentes de los vectores de estado de todas las fuentes, con escalamientos apropiados para cada uno de dichos componentes según corresponda. Á continuación, se construye paso a paso el caso anterior a modo de ejemplificación.

Sea $\mathcal{O}_{f(x_0)} = \{f_1(x_0), \ldots, f_n(x_0), \ldots\}$ la órbita obtenida del sistema dinámico con las condiciones iniciales dadas. Observando el vector de estado, tenemos que la primer componente es el *índice de tiempo*, por lo que se define la sucesión de valores para este parámetro obtenida mediante la función de indización:

$$\mathcal{T} = \{T(t) : t \in \mathbb{N}\}$$

donde para el presente ejemplo $T(t) = T(t, f_t(x_0))$, i.e. que la función de indización es cualquier mapeo arbitrario que depende del número de iteración y del valor del sistema en esa iteración, con la única condición de que la función sea monótona estrictamente creciente, i.e. $T(t) > T(t+1)$.

El siguiente parámetro en el vector de estado es el vector de posición espacial $z_i(t)$ que denota la ubicación dentro de la configuración de bocinas de la $i-$ésima fuente sonora en el paso t. Recordemos que este vector está expresado en coordenadas esféricas de tal modo que: $z_i(t) = (r_i, \theta_i, \phi_i)_t$, donde r es el radio del origen al punto, θ el ángulo respecto al eje x de la proyección del punto en el plano xy y ϕ el ángulo respecto al eje z de la proyección del punto en el plano yz. Nótese que el valor de r puede ser arbitrario positivo ya que es la distancia del origen al punto de ubicación espacial dentro del recinto, por lo que puede ser expresado en metros o alguna otra unidad apropiada. Para los dos ángulos se tiene la restricción de que $\theta, phi \in [0, 2\pi]$, por lo que es necesario hacer un correcto escalamiento de los valores provenientes de la órbita del sistema dinámico, de acuerdo a la manera en que quiera usarse tal información.[3]. Una idea primaria es formar bloques de tres elementos consecutivos provenientes de la órbita y mapearlos directamente a cada componente del vector de posición espacial, de tal modo que:

$$z_i(t) = (f_{3t}(x_0), \alpha_i f_{3t+1}(x_0), \beta_i f_{3t+2}(x_0))$$

donde $\alpha_i, \beta_i \in \mathbb{R}$ son constantes apropiadas para escalar los valores del flujo de datos original de la órbita a los valores de los ángulos respectivos.
Los siguientes parámetros pueden de hecho ser mapeados de forma análoga de tal modo que:

$$\begin{array}{rcl} A_i(t) & = & f_{3t+3}(x_0) \\ y_{1_i}(t) & = & f_{3t+4}(x_0) \\ \vdots & & \vdots \\ y_{1_m}(t) & = & f_{3t+(m+4)}(x_0) \end{array}$$

De este modo se obtiene se cubren todos los parámetros del vector de estado con un sólo proceso algorítmico activo base. Obsérvese que a partir de esta sencilla modelación, es posible extender las opciones a cualquier forma de mapping imaginable y con el detalle y complejidad matemática que se desee. Como contraparte al mapping mediante una órbita única, se propone el ampping con órbitas múltiples, pero antes de profundizar en dicho modelo, vale la pena detallar la forma en que las isometrías euclideanas pueden ser aplicadas al flujo de datos para obtener transformaciones coherentes de una misma órbita.

[3]Se estableció desde un principio, el uso de coordenadas esféricas por la facilidad que ofrecen para fines de localización, sin embargo, éstas pueden ser transformadas a coordenadas rectangulares sencillamente para fines de cálculo y procesamiento de datos.

5.2.2. Isometrías aplicadas a las órbitas sonoras

Las isometrías euclideanas son básicamente transformaciones rígidas; esto es, mapeos que conservan la distancia original entre puntos. En este apartado se mostrará como puede aplicarse esta idea a series de tiempo obtenidas mediante alguna fuente de información, con el propósito de obtener órbitas distintas pero que mantengan coherencia y simetría con las originales, permitiendo además una economía algorítmica al no ser necesaria una nueva evaluación o iniciación del sistema dinámico en cuestión. Se vió además que cualquier isometría en el plano es producto de la composición de una traslación y una transformación ortogonal. Con el objetivo de clarificación, retómese las isometrías del plano euclideano aplicadas a un punto $\vec{x} = (x_1, x_2) \in \mathbb{R}^2$ del siguiente modo:

1. Traslación por un vector $\vec{a} = (a_1, a_2)$. $T(\vec{x}) = T(x_1, x_2) = (x_1 + a_1, x_2 + a_2)$.
2. Rotación alrededor del origen por un ángulo θ en dirección al sentido antihorario y cuya expresión está dada por:

   $$R_\theta(x_1, x_2) = \begin{pmatrix} cos\theta & -sen\theta \\ sen\theta & cos\theta \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

3. Reflexiones alrededor de una semirecta paralela al eje x_1. $E(x_1, x_2) = (x_1, -x_2)$.
4. Reflexión alrdedor de una semirecta paralela al eje x_2. $E(x_1, x_2) = (-x_1, x_2)$.
5. Deslizamiento y reflexión (glide reflection). $G_{\vec{a}}(x_1, x_2) = (a_1 + x_1, a_2 - x_2)$.

Las isometrías corresponden de hecho, a los movimientos básicos de voces en el plano musical compuesto por el eje horizontal en tiempo y el eje vertical en altura, y pueden ser consideradas de este modo si se restringen los movimientos simétricos dentro de dicho plano a movimientos horizontales y verticales únicamente.

En el plano euclideano existen estos tres tipos de transformaciones que preservan distancias entre los puntos: traslaciones, reflexiones y rotaciones. En el plano musical, todas las transformaciones deber afectar cada una de las diemensiones por separado, dejando estas posibles isometrías: traslación horizontal (repetición), traslación vertical (transposición), reflexión horizontal (movimiento retrógrado), reflexión vertical (inversión) y cualesquiera combinaciones de los anteriores (Vi Hart, 2009).

De lo anterior se desprende una manera simple de aplicar las isometrías directamente a las series de tiempo obtenidas por el flujo de datos provenientes de cualquier fuente de información, que es la que precisamente se propone a continuación.

Lo primero que se debe tener en cuenta es el rango de valores adecuados del flujo de datos y de los parámetros a los que se van a mapear. Por ejemplo, la función de índice de tiempo $T(t)$ sólo acepta valores positivos si se asume que el inicio es

en el instante $t = 0$[4]. A su vez, el vector de posición espacial $s_i(t)$ toma valores por arriba del plano xy por lo que la componente en z siempre es positiva. El parámetro de amplitud $A_i(t)$ restringe sus valores por lo general, al intervalo cerrado $[0, 1]$. Aún en el caso en que estos rangos sean libremente manipulados por el(la) compositor(a), los mismos deben ser especificados de manera predeterminada. Una vez hecho lo anterior, la sutileza radica en insertar la serie de tiempo en el plano euclideano y considerar a cada elemento de la serie como un punto de dicho plano de tal modo que las isometrías puedan construirse a partir de semirectas o puntos aislados. Esto permitirá crear versiones transformadas de una misma órbita que puedan ser adaptables de acuerdo a las necesidades específicas del creador.

Traslación.

Se vió con anterioridad que la traslación en el plano se define mediante un vector $\vec{a}$ cuya magnitud, dirección y sentido definen el movimiento final del objeto geométrico al cual le fue aplicada la transformación rígida.

Sea una serie de datos

$$A = \{(t, y(t)) : t \in \mathbb{N}, y \in \mathbb{R}^+\} = \{(0, y(0)), (1, y(1)), \dots (n, y(n))\}$$

de tal modo que su función de indización de tiempo es constante y de paso $h = 1$. Además los valores que toma la serie de tiempo en cada índice son positivos únicamente, i.e. la serie se encuentra localizada en el primer cuadrante del plano euclideano.

Defínase el vector $\vec{a} = (1, 3)$ y supóngase que se desea aplicar la traslación a la serie de datos por este vector. Primeramente, grafíquese el vector dado. El proceso ahora consiste en trasladar cada punto de la serie individualmente por el vector $\vec{a}$. El resultado será que toda la serie se habrá desplazado en el plano de acuerdo a la magnitud dirección y sentido del vector $\vec{a}$, como se corroborará más adelante, las demás isometrías pueden ser expresada este tipo de traslaciones.

De nuevo, el punto importante que no se debe descuidar es el rango final que ocurre después de aplicar la transformación y si dicho rango satisface nuestras necesidades algorítmicas, ya que de lo contrario se deberá establecer otro vector para guiar el proceso. Conocer el rango después de la transformación es sumamente sencillo; primero localícense los valores mínimo y máximo de la serie[5]:

[4]Si bien es posible asumir pasos de tiempo negativos de modo tal que se defina la órbita hacia atrás, por lo general es posible expresar dichos elementos de la órbita en términos positivos.

[5]Obsérvese que en este caso se asume que la serie ya ha sido generada previamente y está almacenada en alguna lista o pila, sin embargo con elementos como el diagrama de bifurcación resulta relativamente sencillo conocer el rango de valores de la órbita original del sistema dinámico de acuerdo a las condiciones iniciales específicas que se apliquen.

$$\bar{y} = min\{y(t) : t \in \mathbb{N}, y \in \mathbb{R}^+\}$$
$$\breve{y} = max\{y(t) : t \in \mathbb{N}, y \in \mathbb{R}^+\}$$

Entonces, el rango de la serie original obviamente estará dado por el intervalo $[\bar{y}, \breve{y}]$ y por consecuencia lógica el rango de la serie transformada después de aplicada la transformación resulta ser el intervalo $[\bar{y} + a_2, \breve{y} + a_2]$, que para el ejemplo anterior resultaría en $[\bar{y} + 3, \breve{y} + 3]$
Nótese además que la traslación mediante el vector $\vec{a}$ también desplaza la serie de tiempo horizontalmente, lo que se traduce en un desplazamiento en el tiempo de la ocurrencia de los cambios del sistema. En el ejemplo anterior este desplazamiento tiene un valor idéntico a 1. Nótese además de la figura.. que el efecto final que tiene la traslación, es el de *elevar y desplazar* la serie original pero manteniéndola en el primer cuadrante.

El ejemplo presentado fue el más básico y por supuesto se pueden hacer traslaciones de cualquier tipo para obtener cualquier número de series transformadas a partir de una órbita predeterminada, más aún, puesto que las isometrías trabajan en el plano, es posible utilizar como rangos, subconjuntos no sólo del primer cuadrante sino de cualquiera de los otros tres cuadrantes.

Rotación.

La rotación consiste en desplazar un punto u objeto geométrico alrededor del origen por un ángulo θ dado en el sentido antihorario. Como se vió con anterioridad, esta rotación tiene una expresión matemática dada por:

$$R_\theta(x_1, x_2) = \begin{pmatrix} cos\theta & -sin\theta \\ cos\theta & sin\theta \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

En la rotación la distancia del punto al origen permanece fija y sólo cambia su posición respecto a una circunferencia de radio igual a esa distancia tal y como se observa en la figura. De este modo, con una rotación se puede mover la serie alrededor del origen en sentido horario y ubicarla enalguna posición específica. Debe tenerse en cuenta al igual que con todas las isometrías, el rango final después de efectuar la transformación.

El siguiente código muestra el algoritmo implementado en Supercollider para realizar rotaciones de acuerdo a las características antes descritas. ...

Reflexiones.

Sea $\mathcal{L}_\phi$ una recta que pasa a través del origen con una inclinación de un ángulo ϕ respecto al origen. Las reflexiones de un punto $(x, y) \in \mathbb{R}^2$ alrededor de una recta arbitraria como la anterior, pueden expresarse del modo siguiente:

$$Re_{\phi}(x,y) = \begin{pmatrix} cos2\phi & sin2\phi \\ sen2\phi & -cos2\phi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Si bien es posible y útil reflejar las series de tiempo respecto a una semirecta arbitraria en el plano, para fines prácticos de variaciones algorítmicas respecto a alguna órbita base, resulta útil definir las reflexiones de las series de datos respecto a semirectas paralelas ya sea al eje horizontal o al eje vertical, de este modo la reflexión conserva el rango de valores del sistema e incluye además implícitamente una traslación.

Sea $x = c$, con $c > 0$, una semirecta paralela al eje y ubicada en el punto c sobre el eje x tal y como se muestra en la figura.... Entonces, la reflexión de la serie de datos alrededor de esta recta invertirá el orden de la serie de tal modo que el último elemento ahora pasará a ser el primero como se muestra en la figura..

Para el caso de la reflexión alrdededor de una semirecta paralela al eje x, sea $y = c'$ con $c' > 0$ ubicada en el punto c' sobre el eje y tal y como se muestra en la figura. El resultado es completamente análogo al caso anterior sólo que ahora los datos son invertidos horizontalmente.

De este modo aparece más clara la utilidad que en general respresentan las isometrías como transformaciones a las series de datos, pues es posible cambiar rangos de manera inmediata sin alterar la serie, o generar escalamientos de la misma, manteniendo la misma estructura, tomando en cuenta el hecho de que algorítmicamente, el procesamiento de datos mediante isometrías es sencillo, rápido y económico.

La figura... muestra el algoritmo implementado en Supercollider, de una sencilla clase diseñada para realizar reflexiones de cualquier tipo a series de datos, de acuerdo a las características antes descritas. En la figura .. se muestran las isometrías básicas aplicadas

5.3. Mapping mediante múltiples órbitas.

La siguiente opción general de mapping es utilizar, al contrario de una órbita única como en la sección precedente, distintas órbitas para cada grupo de parámetros del vector de estado. Por supuesto esto también es a total discreción y necesidad del(la) compositor(a). Se ilustra esta idea a continuación.

Supóngase con fines de generalización, que se desean utilizar órbitas distintas del sistema dinámico para cada parámetro y para cada fuente sonora. Recuérdese que el vector de estado tiene como primer componente el índice temporal (el cual a su vez posee una sólo parámetro), como segunda componente el vector de posición (con tres parámetros), como tercera componente la amplitud (con un parámetro) y finalmente

el vector de parámetros sonoros con a lo más m elementos[6]. De lo anterior resulta que la parametrización del vector de estado requiere $5+m$ elementos. Si se tienen en total n fuentes sonoras, se necesitán por lo tanto $n(5+m)$ órbitas distintas para cubrir todo el esquema sonoro. Si bien es posible generar cada una de estas órbitas aplicando condiciones iniciales distintas, resulta muy práctico utilizar en este punto las isometrias euclideanas para reciclar órbitas previamente generadas y conservar además una coherencia estructural general[7].

De hecho las fuentes de información, tal y como se ha venido insistiendo a lo largo del trabajo, pueden ser de cualquier naturaleza, desde un sistema dinámico, el índice bursátil, o los datos de algún proceso electoral. Por supuesto, esta naturaleza involucra un contexto estético específico que bien puede ser explotado en la pieza a realizar.

Recuérdese que en la sección.. se definió el conjunto orbital general como aquel que contiene todas las órbitas generadas y transformadas a modo de paleta de opciones composicionales:

$$\mathfrak{O} = \{\overline{\mathcal{O}_f^i(x_0)}\ :\} \bigcup \{\mathcal{O}_f^i(x_0)\}$$

Por supuesto, este conjunto contendrá las $n(5+m)$ órbitas que el(la) compositor(a) necesita para cubrir sus necesidades sonoras. Y tal y como se mencionó en la sección anterior, el paso siguiente es el mapping hacia el vector de estado. Puesto que la intención de esta opción particular es la de mapear órbitas distintas para cada componente del vector de estado, este mapeo resulta directo del modo siguiente:

$$\begin{array}{ccc} T(t) & \rightarrow & \mathcal{O}^1_{f_1}(x_0,i) \\ r_i & \rightarrow & \mathcal{O}^1_{f_2}(x_0,i) \\ \theta_i & \rightarrow & \mathcal{O}^1_{f_3}(x_0,i) \\ \phi_i & \rightarrow & \mathcal{O}^1_{f_4}(x_0,i) \\ A_i & \rightarrow & \mathcal{O}^1_{f_5}(x_0,i) \\ y_{1_i} & \rightarrow & \mathcal{O}^1_{f_6}(x_0,i) \\ \vdots & \vdots & \vdots \\ y_{n_i} & \rightarrow & \mathcal{O}^1_{f_{6+n_i}}(x_0,i) \end{array}$$

La numeración anterior debe entenderse así. Para cada fuente sonora i, cada uno de los parámetros estará controlado en el tiempo, por alguna de las órbitas $\mathcal{O}^1_{f_k}(x_0,i), k = \overline{1, 6+n_i}$, donde el subíndice k de la función f, únicamente denota una enumeración arbitraria de selección de las órbitas del conjunto orbital general, y la expresión $f_k(x_0,i)$ hace referencia a que dicha órbita[8] fue generada con condiciones iniciales

[6]Esto significa que m es el número máximo de parámetros disponibles y que otros vectores de estado pueden tener un número igual o menor de componentes.

[7]Véase la sección...

[8]Por supuesto esta órbita pudo haber sido generada por un mismo sistema dinámico con condiciones iniciales distintas o por trasformación mediante isometría o por un sistema dinámico distinto.

dadas por el vector x_0, para la fuente sonora i.

De este modo puede observarse que una de las ventajas de utilizar órbitas distintas para mapear cada parámetro sonoro radica por supuesto, en la ampliación de la gama de recursos de flujo de datos y en la automatización directa para cada parámetro, pero su implementación resulta mucho más conpleja y con un mayor gasto computacional con respecto al mapping mediante una órbia única.

Claro está que combinaciones de ambos métodos podrían ajustarse de forma más flexible a situaciones de composición reales y de acuerdo a las necesidades específicas del uso de información de la pieza.

5.3.1. Colocación del sonido por planos de campo estéreo.

Este es un modelo teórico de diseño de espacialización y mapeo algorítmico de parámetros sonoros. La intención del mismo radica en crear, una categorización y metacategorización estructural en base a la herrameinta teórica matemática para poder establecer una generalización que pueda ser válida en cualquier tipo de espacialización multicanal, tanto si es algorítmica, automatizada o hecha a mano. Si bien el presente trabajo no incluye detalles acerca de los efectos físicos de la colocación espacial del sonido,[9] si se mencionarán ciertas características fundamentales respecto a la dinámica del sonido como fenómeno físico al momento de realizar las implementaciones prácticas.

Esta es una propuesta que surge como extensión general al sonido estéreo dentro de una configuración de bocinas arbitraria. La idea consiste en generar campos estéreo seleccionando cualesquiera pares de bocinas individuales dentro de la configuración y hacer un tratamiento de paneo del sonido como normalmente se haría en cualquier configuración estéreo habitual, con la salvedad del carácter dinámico de dichas configuraciones; esto es, generar distintos campos estéreo a lo largo de la pieza.

Existen dos características particulares al respecto: simultaneidad y dinamismo. La simultaneidad se refiere al número de campos estéreo internos generados al mismo tiempo, o mejor dicho que operan durante un mismo periodo de tiempo, mientras que el dinamismo se refiere a la creación y disolución de múltiples campos estéreo a lo largo de la pieza.

Recuérdese el funcionamiento básico del paneo dentro de una configuración estéreo es la energía del sonido; para colocar el sonido en algún extremo es necesario que dicho sonido esté sonando únicamente en ese canal (i.e. la energía se concentra en

[9] Ya que no es la intención ahondar en ello, puesto que tal temática correspondería a una extensión práctica del modelo y pertenecería más al área de acústica y psicoacústica que al de composición algorítmica en el área de música electroacústica, que es el enfoque principal del presente trabajo.

ese lado), mientras que para el caso en que se desea colocar el sonido en el centro, es necesario que ambos lados están reproduciendo el sonido en cuestión con la misma intensidad:

La ley seno-coseno usada en el paneo tiene la ventaja de mantener la energía de la señal contante mientras la posición aparente es variada, además de que ha sido usada durante mucho tiempo. Como ejemplo, si:.

$$Left_output = \cos(p) \cdot input$$
$$Right_output = \sin(p) \cdot input$$

donde p se asume que varía de 0º degrees (full left) a 90º (full right), con centro en $p = 45^o$. *Nótese entonces que*

$$(Left_output)^2 + (right_output)^2 = (input)^2$$

lo que indica niveles contantes en ambas salidas (Griesinger 2002).

Sea

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & \ddots & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

La representación matricial de la configuración general de bocinas. Defínase el *conjunto dinámico de campos estéreo* como:

$$\mathfrak{A}[t_i, t_{i+1}] = \{(a_{ij}, a_{kl}) : i, k = \overline{1, m}, j, l = \overline{1, n}, \}, t \in \mathbb{N}$$

Como el conjunto que contiene todos los campos estéreo obtenidos de formar pares ordenados de bocinas (a_{ij}, a_{kl}) que están activos simultáneamente dentro de un intervalo de tiempo arbitrario y bien definido $t_i, t_{i+1}]$. De este modo es posible definir distintos conjuntos estéreo a través del tiempo haciendo distintas elecciones de pares ordenados. Si se establece un *intervalo de duración total o general* medido en segundos y dado por:

$$[t_0, t_n] = \sum_{i=0}^{n-1} t_i, t_{i+1}] = h$$

Entonces se ve claramente que la duración total será de h segundos y que dicho intervalo general está compuesto por $n - 1$ intervalos de duración arbitraria pero de tal modo que la suma de todos ellos sea igual a h. En el mismo espíritu de formalización que se ha venido desarrollando en este trabajo se define el *conjunto general de campos estéreo* de duración h como:

$$\mathfrak{A}[t_0, t_f] = \bigcup_{i=0}^{n-1} \mathfrak{A}[t_i, t_{i+1}]$$

En este sentido, dado cualquier campo estéreo formado por cualesquiera par de bocinas, la idea del tratamiento radica en colocar el sonido mediante paneos definidos para ese horizonte en particular, con lo que se obtendrán distintos tratamientos simultáneos a lo largo de la pieza. Entonces, la lógica en el diseño sonoro de la espacialización deberá pensarse como trayectorias lineales a lo largo de los planos horizontales definidos para cada intervalo de tiempo.

Para poder operar cualitativamente sobre cada uno de estos planos, se propone utilizar segmentos cerrados de línea, los cuales constituyen la herramienta básica para demostrar la convexidad de un conjunto dentro del campo del Análisis matemático. Un segmento cerrado de línea está definido por la siguiente expresión:

$$[x, y] = \{\alpha x + (1 - \alpha)y : \alpha \in [0, 1]\}$$

[10]

Conforme α va tomando valores desde 0 hasta 1, la función $\alpha x + (1 - \alpha)y$ va dibujando punto a punto, una línea recta que comienza en x y termina en y. Obsérvese que $x, y \in \mathbb{R}^n$ y dependiendo de las necesidades de representación podemos utilizar vectores en el plano ($n = 2$) o en el espacio ($n = 3$). Claramente el punto x corresponderá a la primer bocina a_{ij} que conforma un plano estéreo específico, mientras que y corresponderá a la segunda bocina a_{kl} que pertenece a dicho campo; de este modo resulta directo y relativamente sencillo, operar tanto abstracta como algorítmicamente con éstos planos estéreo dinámicos.

Definición 5.2. *Sea*

$$\mathfrak{A}[t_i, t_{i+1}] = \{(a_{ij}, a_{kl}) : i, k = \overline{1, m}, j, l = \overline{1, n}, \}, t \in \mathbb{N}$$

un conjunto de campos estéreo dinámicos, entonces para cada $(a_{ij}, a_{kl}) \in \mathfrak{A}[t_i, t_{i+1}]$ *se define el* ***horizonte dinámico*** *o representación geométrica del campo estéreo dinámico como el segmento cerrado de línea recta que une* a_{ij} *con* a_{kl}*:*

$$[a_{ij}, a_{jl}] = \{\alpha a_{ij} + (1 - \alpha)a_{jl} : \alpha \in [0, 1]\}$$

donde $a_{ij} = (x_{ij}, y_{ij}, z_{ij}), a_{jl} = (x_{jl}, y_{jl}, z_{jl}) \in \mathbb{R}^3$

En la definición anterior se estableció que la representación vectorial de las bocinas y de los planos estéreo es en el espacio y mediante coordenadas cartesianas, sin embargo se puede utilizar sin ningún problema representaciones en el plano o

[10] Véase la sección..

mediante coordenadas esféricas o cilindrícas según convenga. Nótese que esta *representación geométrica del campo estéreo dinámico* será de gran utilidad al momento de operar algorítmicamente con flujos de datos y para crear el diseño de espacialización sonora en general.

Con todo lo anterior en mente resulta más claro el como será posible colocar el sonido a lo largo de un horizonte definido por un plano estéreo dinámico; basta con seleccionar puntos coordenados a lo largo de la representación geométrica de dicho campo estéreo y dado que para cada campo en particular, cualquier punto estará dado por la función que define el segmento cerrado de línea recta, se obtiene de este modo una forma de localización directa y sencilla en el paneo; un ejemplo ilustrará mejor esta idea.

Supóngase que se tiene un campo estéreo definido por dos bocinas a y b, las cuales tienen coordenadas rectangulares $a = (1, 4, 1)$ y $b = (3, 1, 3)$. Entonces, el horizonte definido por estas dos bocinas queda expresado mediante:

$$[a, b] = \{\alpha a + (1 - \alpha)b : \alpha \in [0, 1]\} = \{\alpha(1, 4, 1) + (1 - \alpha)(3, 1, 3) : \alpha \in [0, 1]\}$$

y para realizar paneos en este horizonte basta con variar la posición de la fuente sonora sobre esta precisa línea recta como si se tratase de un campo estéreo normal utilizando además la noción de la distribución proporcional de la energía citada al principio de la sección.

Capítulo 6

Implementaciones en Supercollider

En éste capítulo se describirán los algoritmos que se diseñaron para poder generar una implementación práctica de un sistema de espacialización multicanal en el lenguaje de programación supercollider. Se comenzará describiendo el algoritmo del paneador multicanal, así como la estructura algorítmica de control basada en ProxySpace. A continuación se mostrará la estructura algorítmica de mapeos caóticos y la generación de series de tiempo a partir de éstos. Finalmente se planteará la clase mediante la cual es posible obtener envolventes basadas en sistemas dinámicos para controlar cada parámetro sonoro deseado.

Este capítulo no pretende ser un tutorial de Supercollider, sino que se da por hecho que el lector está familiarizado con el lenguaje y con ciertos proceso básicos de programación inherentes al mismo. De cualquier modo se recomienda al lector poco familiarizado con el tema, consultar (Cottle 2006).

6.1. La clase PanW

Puesto que la idea principal del presente proyecto estuvo basada desde el principio, en la idea de generar trayectorias de espacialización basadas en sistemas caóticos, la primera característica necesaria era la de proveer al sistema con la posibilidad de desplazar el sonido entre cualesquiera pares de salida **arbitrarias**.

La mayoría de los sistemas de paneo con los que cuenta Supercollider (Pan4, PanAz, PanX, PanX2D, etc.) están restringidos al desplazamiento del sonido entre bocinas **adyacentes**; esto significa que dentro de una configuración multicanal de n salidas por ejemplo, para que el sonido vaya de la salida 1 a la salida 5, éste necesita recorrer la salida 2, 3 y 4 previamente. Dicha restricción enmarca la forma general de la espacialización, en un movimiento circular per se, desde el punto de vista funcional.

Ante la necesidad de poder crear trayectorias entre cualesquiera salidas arbitrarias no necesariamente adyacentes, la solución más inmediata y a la que recurren algunos

artistas usuarios de Supercollider del área electroacústica o acusmática, es la de generar una *base de datos* de configuraciones de paneos de dos, cuatro o n salidas (Pan2, Pan4 y PanX respectivamente) que incluyan la mayor cantidad de opciones que generen los arreglos multicanal deseados. Por supuesto, este acercamiento resulta poco efectivo desde el punto de vista tanto de programación como funcional en tiempo real (consumo de CPU por proceso) ya que el número de procesos individuales por programar y por ejecutar es proporcional al número de variaciones o configuraciones deseadas. Por ejemplo, supongase que se tiene una configuración general octafónica y que se desean generar trayectorias de espacialización mediante el proceso anterior utilizando un paneador de cuatro canales (Pan4). Si se desearan cubrir todas las posibilidades, el número de procesos a programar y a ejecutar equivaldría a la permutación de cuatro elementos escogidos de un grupo de ocho: $nPr = 5040$, número que resulta bastante elevado.

Por esta razón se decidió diseñar desde cero, un algoritmo que pudiera distribuir el sonido dentro de una configuración multicanal arbitraria basado en la idea primordial de *trayectoria caótica*, entendiendo por éste término, lo que se definió en capítulos anteriores: *la espacialización multicanal cuyos parámetros sonoros están controlados por series de datos obtenidas a partir de sistemas dinámicos caóticos.*

El primer prototipo consistió en escribir una clase en Supercollider basada en un sencillo algoritmo para dos salidas arbitrarias utilizando la ley de seno-coseno y la idea de *planos de campo estéreo* . La figura 6.1 muestra el código diseñado para tal propósito:

```
1
2
3 PanD {
4     *ar {|in, pos, out1, out2, level|
5         var out, chan1, chan2, input;
6
7         input = in;
8
9         chan1 = (cos(pos).abs)*input;
0         chan2 = (sin(pos).abs)*input;
1
2         out = [Out.ar(out1,chan1*level),Out.ar(out2,chan2*level)];
3         ^out;
4     }
5 }
6
7
```

Figura 6.1: Algoritmo básico de paneo de dos canales con salidas arbitrarias.

Esta clase (y la última versión del paneador general) hace uso de la llamada *ar por lo que básicamente se trata de un Pseudo Ugen dentro del servidor de Supercolli-

der.[1] La clase toma como argumentos la señal de entrada (in), la posición del paneo (pos), el primer bus de salida (out1), el segundo bus de salida (out2) y la amplitud de control general de la señal (level). Esta clase consta básicamente de un arreglo bidimensional con un Ugen *Out.ar* en cada una, que en este caso haría la función del *OutputProxy* dentro de *MultiOutUgen*.[2] Con esta clase es posible extender la funcionalidad del Ugen *Pan2* permitiendo elegir cualesquiera canales arbitrarios de salida no necesariamente adyacentes.

El siguiente paso consistía en poder extender este prototipo básico de modo tal que pudiera manejar cualquier número de canales de salida simultáneamente dada una trayectoria predefinida. Considérese el caso por ejemplo, que se tuviera una configuración de doce bocinas y que se deseara generar las siguientes trayectorias de espacialización: $[0, 1, 10, 11, 2, 3]$ y $[5, 2, 7, 9, 12, 1]$. Ante tal planteamiento, la idea más inmediata y natural era la de generar arrays que encadenaran los elementos de las trayectorias usando la clase *PanD*; i.e. se tendría un *PanD* para encadenar el canal 0 y el canal 1, otro *PanD* para encadenar el canal 1 y el canal 10, y así sucesivamente. Sin embargo dicho acercamiento era muy parecido al primer proceso de espacialización comparado y al igual que éste resultaba poco funcional desde el punto de vista de programación ya que se necesitaban $n-1$ objetos *PanD* corriendo simultánemente para una trayectoria de n canales.

Se necesitaba entonces plantear un proceso algorítmico que fuera muy general y que al mismo tiempo fuera eficiente y funcional para realizar la tarea de distribuir el sonido mediante campos estéreo arbitrarios.

Utilizando la ley de coseno-seno para el paneo, la amplitud máxima de cada canal se alcanza en los extremos izquierdo y derecho respectivamente debido a que:

$$cos(0) = 1, sen(0) = 0$$
$$\&$$
$$cos(1) = 0, sen(1) = 1$$

Lo anterior puede observarse claramente en la figura 6.2 donde el eje x representa el ángulo y el eje y la amplitud asociada a dicha posición.

La idea con la que se trabajó a continuación fue la de usar las propiedades básicas de periodicidad de las funciones sinusoidales y sus propiedades de desfase. Recuérdese que debido a dicha propiedad de periodicidad se cumple que:

[1] Para más información acerca del tema de clases, ugens y pseudougens dentro de Supercollider véase (Stowell 2011)

[2] OutputProxy es básicamente un slot que Supercollider usa para almacenar o darle salida dentro del servidor a alguna señal de audio. MultiOutUgen no es otra cosa que la versión multicanal de Ugen, o mejor dicho, la *expansión* multicanal de éste

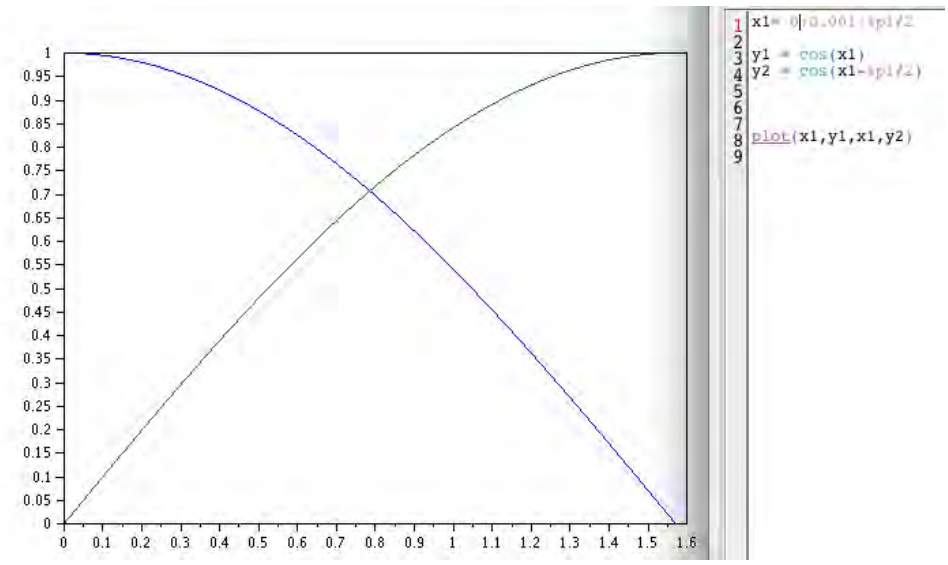


Figura 6.2: Gráfica en Scilab de dos ondas coseno con desfase $\pi/2$

$$cos(x) = sen(x + \frac{\pi}{2})$$

Esto implica que es posible expresar todos las salidas como combinaciones de funciones seno o coseno, i.e., para el caso de dos canales, la amplitud de cada salida estará dada por:

$$Out_1 = \sin(x) \cdot input$$
$$Out_2 = \mid \sin(x - \frac{\pi}{2}) \mid \cdot input$$

El signo negativo en la expresión anterior indica un desplazamiento hacia la izquierda de la función sinusoidal, que es precisamente lo que se desea cada vez que se introduce un nuevo canal.

6.2. Versión con función seno.

Como primer propuesta se decidió utilizar precisamente la función seno como base generadora del paneo multicanal, teniendo en mente la posibilidad de experimentar con una forma alternativa de paneo a la usual cambiando el comportamiento de la amplitud de los extremos, permitiendo hacer un fade in y un fade out completo de la señal al variar la posición entre cada extremo, en vez de tener amplitud máxima como en el caso anterior. Esto significaba que se tendría amplitud cero en cada extremo y que se seguiría cumpliendo la relación de proporcionalidad en el cambio de la amplitud.

Debido a que en el caso de las sinusoidales se usan secciones completas de longitud π, [3] la extensión al caso multicanal resulta bastante sencilla. De hecho, tras un análisis detallado del comportamiento de dicha función para varios casos, se pudieron inferir las siguientes características:

- Los rangos de acción para cada canal son todos de longitud π y están relacionados linealmente con el ángulo o posición.
- Cada nuevo canal que se añade puede ser expresado mediante una función sinusoidal cuyo ángulo está defasado proporcionalmente al número total de canales.

Lo anterior puede resumirse en la siguiente expresión matemática:

$$\begin{array}{ccc} f_i : I_i & \longrightarrow & [0,1] \\ x & \longrightarrow & sen(x - i\frac{\pi}{2}) \end{array}$$

donde $I_i = [i\frac{\pi}{2}, i\frac{\pi}{2} + \pi]$.

Dentro de la clase *PanAz* existe un parámetro denominado *width* el cual básicamente distribuye el paneo entre el número de canales especificado. Por ejemplo para una configuración de n canales, un *width* igual a 2 distribuye el paneo entre dos canales adyacentes, para un width igual a 3 el paneo se distribuirá en tres canales y así sucesivamente. Para poder implementar una funcionalidad parecida al paneador que se estaba construyendo, se buscó sin éxito, información respecto al algoritmo de fondo que controlaba tal parámetro; ante tal situación se decidió resolver dicho problema de manera personal.

La idea básica con la que se comenzó estaba basada en el factor multiplicativo del ángulo de desfase. Dada la ecuación sinusoidal de onda $cos(x - \alpha)$, como es bien sabido, el valor de α es la cantidad que se desplaza dicha onda a la izquierda sobre el eje x. Por otro lado, cuando $\alpha = \frac{\pi}{2}$ y para cualquier número de canales, se cumple que cuando la amplitud de algún canal es máxima (esto es igual a 1), la amplitud del canal anterior y posterior son mínimas (esto es igual a 0), sin embargo si el 2 se sustituye por valores incrementales sucede que estos máximos y mínimos se van desplazando hacia la izquierda generando situaciones en las que se tengan varias ondas con amplitud mayor a cero simultáneamente cuyos valores de dicha amplitud son proporcionales precisamente al valor del desplazamiento. Tomando en cuenta estas dos características es posible visualizar un método para distribuir el sonido en más de dos canales preservando la equipotencia del paneo.

[3] Puesto que el periodo completo es de 2π y para el caso de la amplitud se busca trabajar con media onda únicamente.

Véase por ejemplo la siguiente figura en la que se ilustra el caso de cinco ondas sinusoidales con ángulo de desfase equivalente a múltiplos de $\frac{\pi}{3}$.

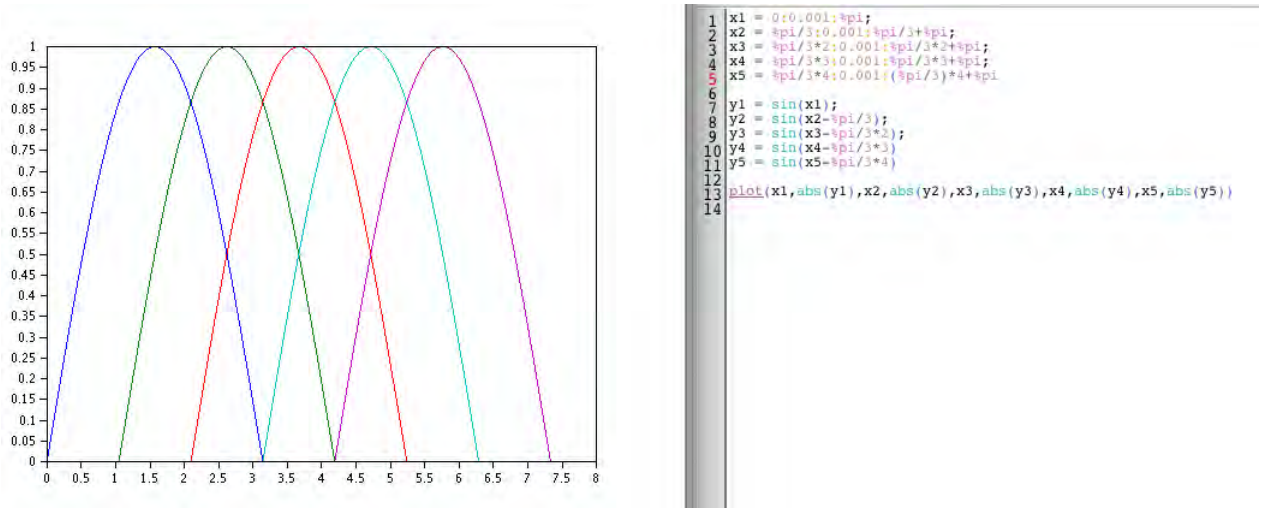


Figura 6.3: Gráfica en Scilab de cinco ondas sinusoidales con desfase de $\pi/3$

Obsérvese por ejemplo que cuando la amplitud de la segunda onda es igual a 1, la primera y la tercera onda tienen amplitudes positivas, lo que equivaldría a a que los tres primeros canales estén sonando simultáneamente con amplitudes distintas, i.e. que el paneo se distribuya en tres canales exactamente. Lo anterior conduce a una natural conclusión general:

$$\begin{array}{rcl} f_i : I_i & \longrightarrow & [0,1] \\ x & \longrightarrow & sin(x - i\frac{\pi}{k}) \end{array}$$

donde $I_i = [i\frac{\pi}{k}, i\frac{\pi + k\pi}{k}]$, para $k \in \mathbb{N}$ que representa precisamente el número de canales entre los que se distribuye el paneo; denominamos a éste parámetro *spread* y es equivalente al parámetro *width* del *PanAz*.

Ahora bien el siguiente paso consistía en diseñar el algoritmo anterior dentro del lenguaje de programación de Supercollider. El problema principal como en todo algoritmo que se desee sirva de manera general para todos los casos y que al mismo tiempo sea funcional, es que sea ejecutable mediante una expresión única lo más simplificada posible. Para el caso presente se tenían tres componentes principales a considerar: las funciones con desfases incrementales, los rangos asociados a cada una de ellas y el valor del desfase. Obviamente resultaba poco funcional definir explícitamente dentro del algoritmo cada una de estas características por lo que la solución más óptima era hacer uso de la función escalón de Heaviside pero adaptada al contexto de los rangos de cada una de las componentes utilizando un único parámetro de desplazamiento y por lo tanto una única función sinusoidal dentro del algoritmo:

$$\mu_i(x) = \begin{cases} 1 & x \in I_i \\ 0 \quad j & x \notin I_i \end{cases}$$

Dentro de Supercollider existe un Ugen que es capaz de representar esta funcionalidad; *InRange.kr*. Un vistazo rápido al documento de ayuda de dicho Ugen aclara que los parámetros que maneja son el input, la cota mínima y la cota superior; ambas, cota mínima y superior forman el intervalo de ejecución. *InRange.kr* retornará el valor de 0 siempre que el input se encuentre fuera del intervalo definido por las cotas y el valor de 1 cuando dicho valor se encuentro dentro de este intervalo. Para poder aplicar esta herramienta al contexto deseado, se optó por definir un rango general sobre el cual variara la posición del siguiente modo:

$$I_x = [0, \kappa(x)]$$

donde $\kappa(x) = \dfrac{n\pi + kpi}{k}$, para n el número total de canales dentro de la configuración general. De este modo, la entrada de InRange.kr es la posición sobre el rango general y los límites inferior y superior son definidos por los respectivos rangos individuales para cada canal. Esto permite que se pueda expresar el algorítmo como un array sencillo de iteraciones incrementales, donde el contador define tanto la función con su defase como los rangos de evaluación o dominios de cada función.

Habilitando a la clase para que además reciba un array como argumento para la definición de la trayectoria de espacialización, se dedujo finalmente el algoritmo mostrado en la figura 6.4:

```
1
2  Pan_S {
3      *ar{|in, pos, level=0.5,offset=2,path|
4          var chan,input,dps,max;
5
6          input = in;
7          dps = (pi/offset);
8          max = path.size*dps+pi;
9
10         chan = Array.fill(path.size,{arg i;
11             Out.ar(path[i],(InRange.kr(pos*max,dps*i,
   (dps*i)+pi)*sin((pos*max)-(i*dps))).abs*input*level);
12         });
13             ^chan;
14     }
15 }
```

Figura 6.4: Algoritmo en Supercollider del paneador básico

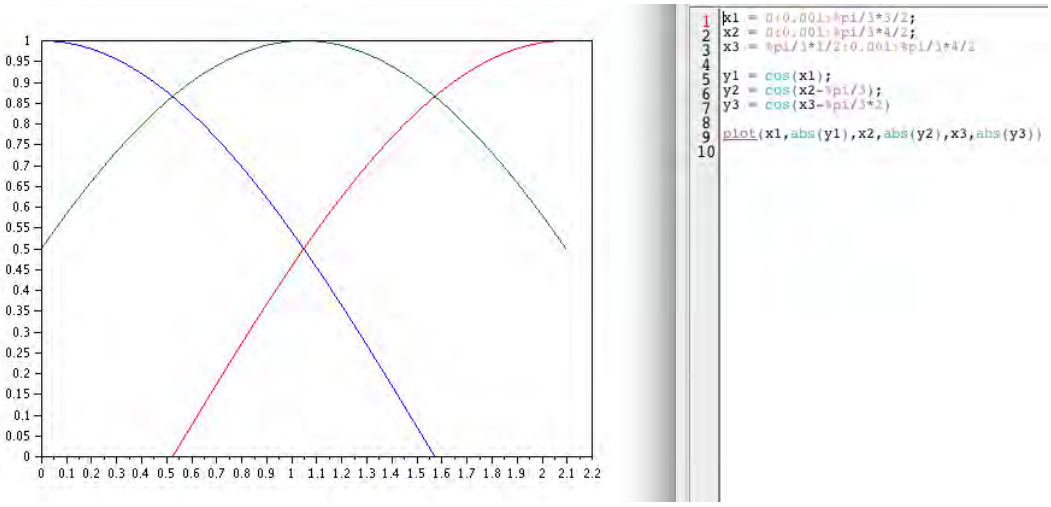
Si bien este algoritmo trabaja correctamente y según lo esperado, dentro de la implementación práctica existe una deficiencia audible en el momento en que la señal de audio hace cruce entre cada canal. Esta deficiencia radica en que existe un pequeño

intervalo, justo en la intersección de las amplitudes, dentro del cual se perciben las señales como ondas separadas y no como un cambio en la posición relativa de una misma señal de audio.

6.3. Versión con función coseno

Con el fin de resolver el detalle de funcionamiento descrito anteriormente, se optó por construir el paneador pero ahora en base a cosenos. Con esta función, ahora los extremos tendrían amplitudes máximas al contrario del caso en el que se usaban funciones senos para un número n de canales deseados en la trayectoria de espacialización.

Sin embargo, la complejidad algorítmica para definir los rangos para el proceso multicanal utilizando funciones cosenos resultó ser mucho más elevada que en el sencillo caso de la función seno. El problema principal radicaba en el hecho de que los canales extremos (primero y último) debían de tener amplitud máxima y al mismo tiempo mantener proporcionalidad variable entre los canales intermedios. Debido a esto los rangos de cada componente varían y la cuestión era descifrar la forma en que lo hacían o mejor dicho, establecer un modelado para expresar dicho comportamiento. Las siguientes gráficas muestra distintos casos para distintos valores de factor de *spread* y número de canales:

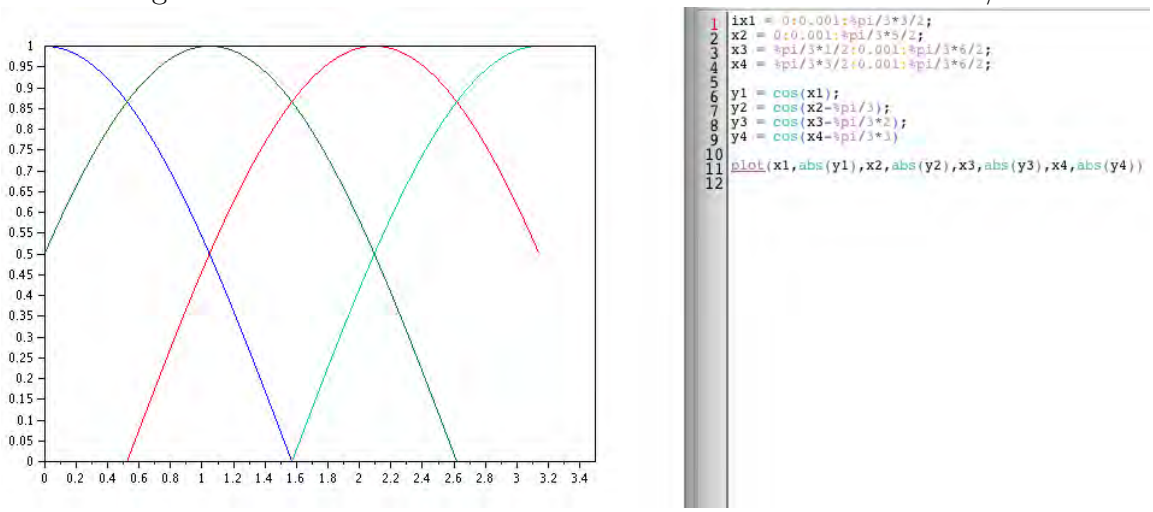


```
x1 = 0:0.001:%pi/3*5/2;
x2 = 0:0.001:%pi/3*4/2;
x3 = %pi/3*1/2:0.001:%pi/3*4/2

y1 = cos(x1);
y2 = cos(x2-%pi/3);
y3 = cos(x3-%pi/3*2)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3))
```

Figura 6.5: Gráfica en Scilab de tres ondas sinusoidales con desfase de $\pi/3$

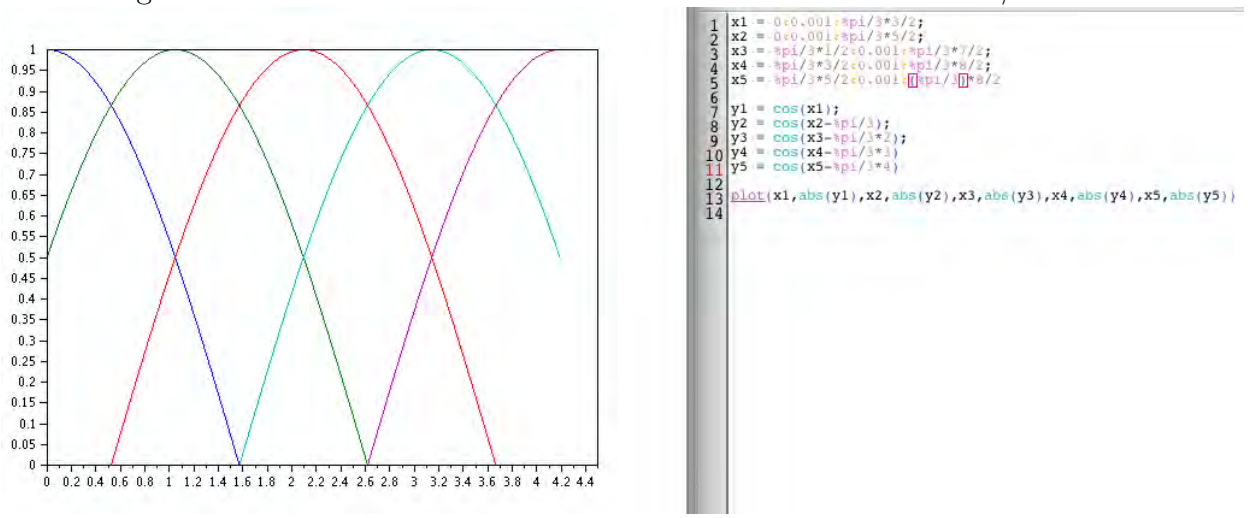


```
ix1 = 0:0.001:%pi/3*3/2;
x2 = 0:0.001:%pi/3*5/2;
x3 = %pi/3*1/2:0.001:%pi/3*6/2;
x4 = %pi/3*3/2:0.001:%pi/3*6/2;

y1 = cos(x1);
y2 = cos(x2-%pi/3);
y3 = cos(x3-%pi/3*2);
y4 = cos(x4-%pi/3*3)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4))
```

Figura 6.6: Gráfica en Scilab de cuatro ondas sinusoidales con desfase de $\pi/3$

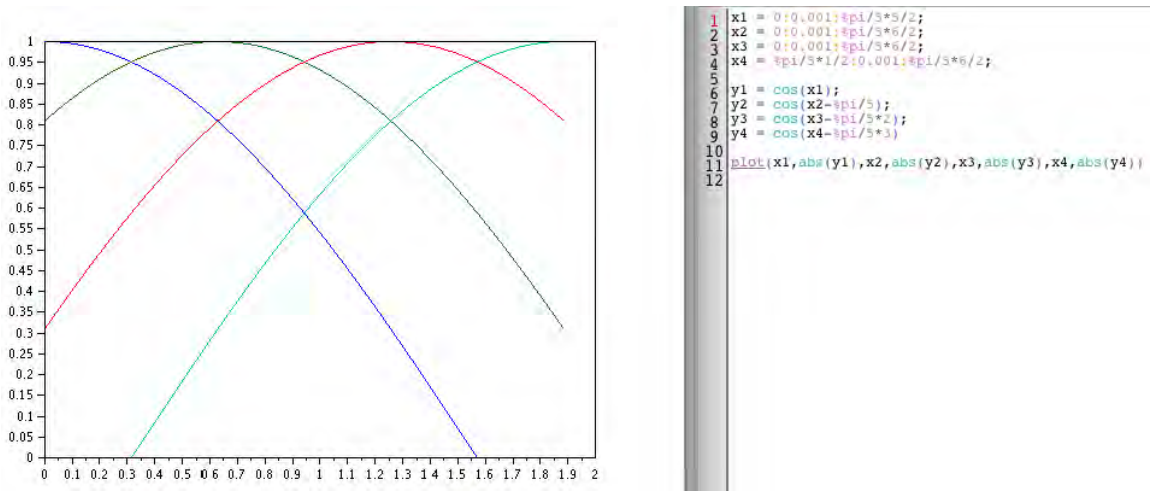


```
x1 = 0:0.001:%pi/3*3/2;
x2 = 0:0.001:%pi/3*5/2;
x3 = %pi/3*1/2:0.001:%pi/3*7/2;
x4 = %pi/3*3/2:0.001:%pi/3*8/2;
x5 = %pi/3*5/2:0.001:%pi/3*8/2

y1 = cos(x1);
y2 = cos(x2-%pi/3);
y3 = cos(x3-%pi/3*2);
y4 = cos(x4-%pi/3*3)
y5 = cos(x5-%pi/3*4)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4),x5,abs(y5))
```

Figura 6.7: Gráfica en Scilab de cinco ondas sinusoidales con desfase de $\pi/3$

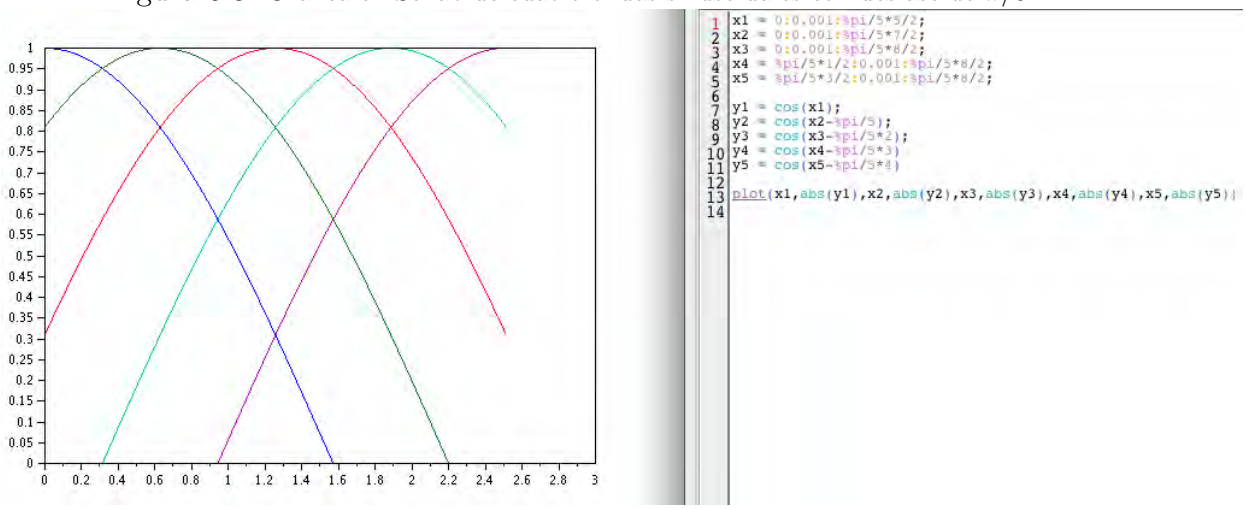


```
x1 = 0:0.001:%pi/5*5/2;
x2 = 0:0.001:%pi/5*6/2;
x3 = 0:0.001:%pi/5*6/2;
x4 = %pi/5*1/2:0.001:%pi/5*6/2;

y1 = cos(x1);
y2 = cos(x2-%pi/5);
y3 = cos(x3-%pi/5*2);
y4 = cos(x4-%pi/5*3)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4))
```

Figura 6.8: Gráfica en Scilab de cuatro ondas sinusoidales con desfase de $\pi/5$

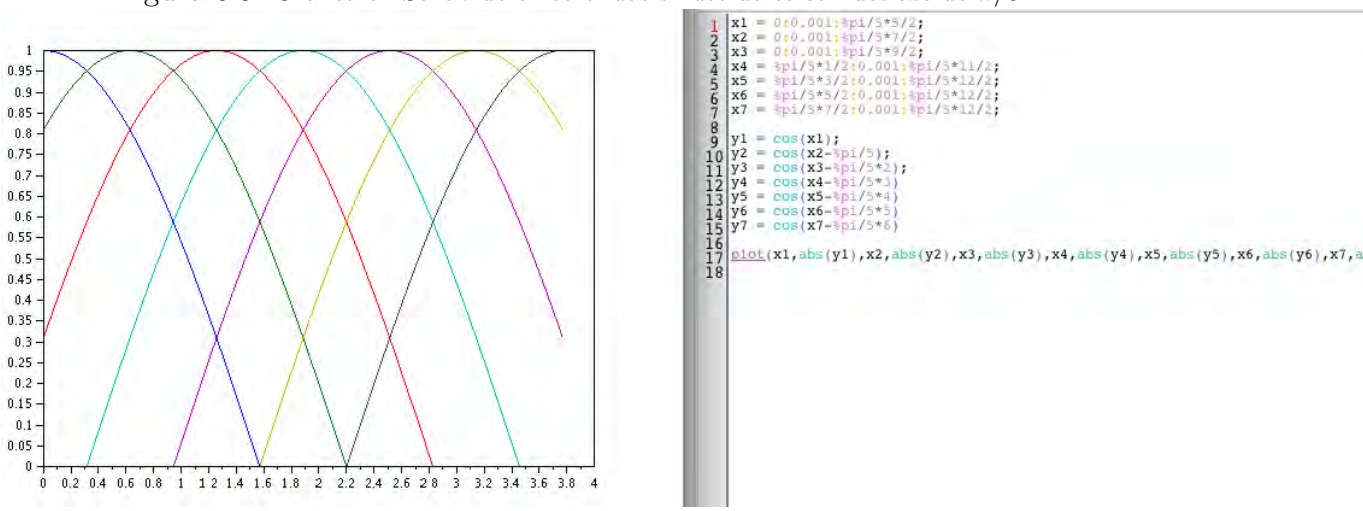


```
x1 = 0:0.001:%pi/5*5/2;
x2 = 0:0.001:%pi/5*7/2;
x3 = 0:0.001:%pi/5*8/2;
x4 = %pi/5*1/2:0.001:%pi/5*8/2;
x5 = %pi/5*3/2:0.001:%pi/5*8/2;

y1 = cos(x1);
y2 = cos(x2-%pi/5);
y3 = cos(x3-%pi/5*2);
y4 = cos(x4-%pi/5*3)
y5 = cos(x5-%pi/5*4)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4),x5,abs(y5))
```

Figura 6.9: Gráfica en Scilab de cinco ondas sinusoidales con desfase de $\pi/5$

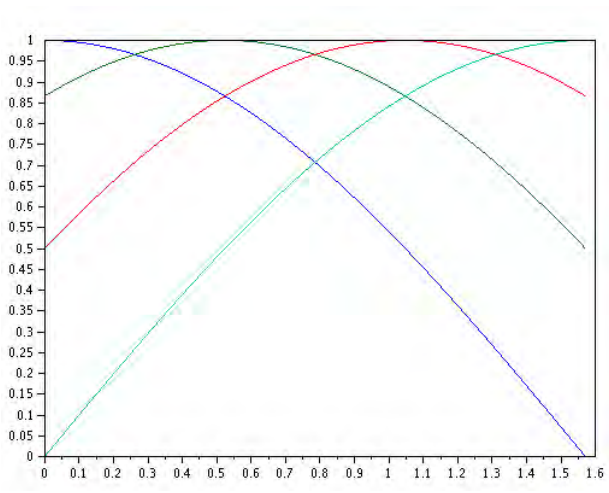


```
x1 = 0:0.001:%pi/5*5/2;
x2 = 0:0.001:%pi/5*7/2;
x3 = 0:0.001:%pi/5*9/2;
x4 = %pi/5*1/2:0.001:%pi/5*11/2;
x5 = %pi/5*3/2:0.001:%pi/5*12/2;
x6 = %pi/5*5/2:0.001:%pi/5*12/2;
x7 = %pi/5*7/2:0.001:%pi/5*12/2;

y1 = cos(x1);
y2 = cos(x2-%pi/5);
y3 = cos(x3-%pi/5*2);
y4 = cos(x4-%pi/5*3)
y5 = cos(x5-%pi/5*4)
y6 = cos(x6-%pi/5*5)
y7 = cos(x7-%pi/5*6)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4),x5,abs(y5),x6,abs(y6),x7,a
```

Figura 6.10: Gráfica en Scilab de siete ondas sinusoidales con desfase de $\pi/5$

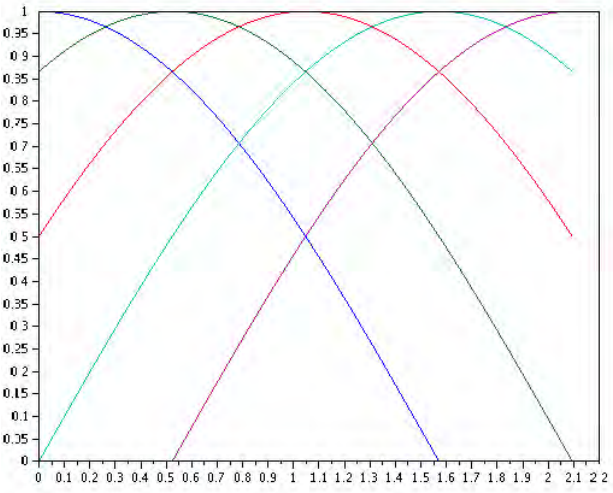


```
x1 = 0:0.001:%pi/6*6/2;
x2 = 0:0.001:%pi/6*6/2
x3 = 0:0.001:%pi/6*6/2
x4 = 0:0.001:%pi/6*6/2

y1 = cos(x1);
y2 = cos(x2-%pi/6);
y3 = cos(x3-%pi/6*2);
y4 = cos(x4-%pi/6*3)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4))
```

Figura 6.11: Gráfica en Scilab de cuatro ondas sinusoidales con desfase de $\pi/6$

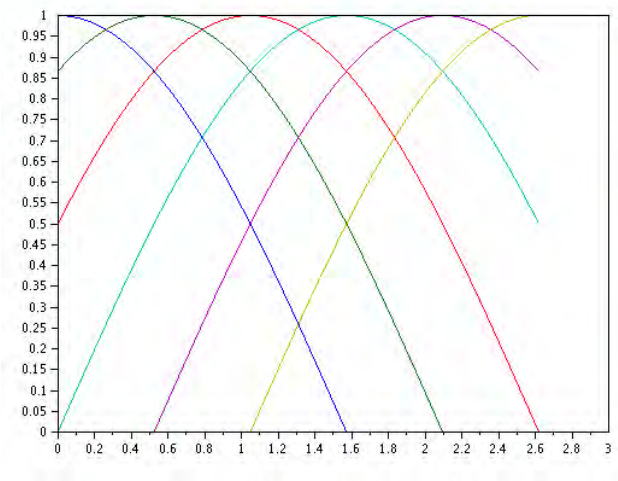


```
x1 = 0:0.001:%pi/6*6/2;
x2 = 0:0.001:%pi/6*8/2;
x3 = 0:0.001:%pi/6*8/2;
x4 = 0:0.001:%pi/6*8/2;
x5 = %pi/6:0.001:%pi/6*8/2;

y1 = cos(x1);
y2 = cos(x2-%pi/6);
y3 = cos(x3-%pi/6*2);
y4 = cos(x4-%pi/6*3)
y5 = cos(x5-%pi/6*4)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4),x5,abs(y5))
```

Figura 6.12: Gráfica en Scilab de cinco ondas sinusoidales con desfase de $\pi/6$



```
x1 = 0:0.001:%pi/6*6/2;
x2 = 0:0.001:%pi/6*8/2;
x3 = 0:0.001:%pi/6*10/2;
x4 = 0:0.001:%pi/6*10/2;
x5 = %pi/6*2/2:0.001:%pi/6*10/2;
x6 = %pi/6*4/2:0.001:%pi/6*10/2;

y1 = cos(x1);
y2 = cos(x2-%pi/6);
y3 = cos(x3-%pi/6*2);
y4 = cos(x4-%pi/6*3)
y5 = cos(x5-%pi/6*4)
y6 = cos(x6-%pi/6*5)

plot(x1,abs(y1),x2,abs(y2),x3,abs(y3),x4,abs(y4),x5,abs(y5),x6,abs(y6))
```

Figura 6.13: Gráfica en Scilab de seis ondas sinusoidales con desfase de $\pi/6$

Después de análisis cuantitativos y teóricos se puntualizaron las siguientes características respecto al planteamiento descrito:

- Dependiendo del valor del desplazamiento o spread ($\frac{\pi}{k}$) definido existe un número específico de componentes cosenoidales que intersectan al eje y. Para valores mayores, las funciones cosenoidales recorridas a la izquierda ya no intersectan al eje y y se van desplazando sobre el eje x en intervalos de longitud proporcional a dicho spread. De hecho, dependiendo de si k es par o impar, este desplazamiento tiene un comportamiento definido. Se decidió denominar *nodo de transición* al número de canales a partir del cual, para un k definido, las componentes cosinoidales ya no intersectan al eje y. Como se puede observar de las figuras anteriores, este *nodo de transición* varía de acuerdo al valor de k; por lo que se definió una función para poder calcularlo:

$$\varsigma(k) = \lfloor(\frac{k+2}{2}\rfloor$$

  donde $\lfloor x \rfloor = max\{m \in \mathbb{N} : m \leq x\}$ es la *función parte entera*.[4] La función anterior arroja el número de canales a partir del cual, dado el valor k que define el spread, las componentes sinusoidales dejan de intersectar al eje y.

- Para $n \leq \varsigma(k)$, los rangos de las componentes poseen un comportamiento lineal que depende únicamente de el número de canales n, dicho comportamiento se modeló del siguiente modo:

$$I_i = [0, i\frac{\pi}{k}], \quad \forall i = \overline{1, n}$$

  para funciones componentes con incrementos proporcionales en al ángulo de defase:

$$f_i(x) = sen(x - i\frac{\pi}{k})$$

- Para $n > \varsigma(k)$ los rangos de las componentes comienzan a depender de múltiplos del spread ($\frac{\pi}{k}$)) pero de una manera no tan directa. En primer lugar, la primer componente recorre un cuarto de ciclo completo por lo que el factor de multiplicación debe ser tal que su rango sea $[0, \frac{\pi}{2})]$; dicho factor es obviamente: $\frac{k}{2}$). En el proceso del diseño del algoritmo se decidió definir el mecanismo de cada límite de los rangos por separado, por lo que primero se definieron los límites inferiores del siguiente modo:

$$\mathcal{A}(k,n) = \{a_i = 0 : i = \overline{0, \varsigma(k) - 1}\} \bigcup \{a_i = \varrho + (i - \varsigma(k)) : i = \overline{\varsigma(k), n}\}$$

[4]Esta función básicamente redondea un valor al menor número entero más cercano; por ejemplo $\lfloor 1.56 \rfloor = \lfloor 1.89 \rfloor = \lfloor 1.9 \rfloor = 1$, Véase...

donde ϱ es un factor de proporcionalidad del cual se hablará más adelante. Para el caso de los límites superiores de cada rango se obtuvo el siguiente algoritmo:

$$\mathcal{B}(k,n) = \{b_i = \frac{\pi}{k} + \frac{2i}{2} : i = \overline{0, \alpha - 1}, \alpha = n - \varsigma(k) = \} \bigcup \{b_i = (n-1) : i = \overline{\alpha, n}\}$$

- Cuando k es impar los límites inferiores poseen un factor de proporcionalidad que actúa del siguiente modo:

$$\frac{\pi}{k} * (\frac{1}{2} + i), \quad i = \overline{0, n-1}$$

Cuando k es par el factor de proporcionalidad es:

$$\frac{\pi}{k} * (1 + i), \quad i = \overline{0, n-1}$$

Ahora bien, dentro del programa esta función puede ser resuelto de dos modos:

$$if(k.even, \varrho = 1, \varrho = 1/2);$$

o con el fin de evitar el test booleano se formó la siguiente alternativa:

$$\varrho = (\frac{k}{2} - \lfloor(\frac{k - 0.5}{2}\rfloor)$$

- Toda vez que se han formado los conjuntos de límites inferiores y superiores $\mathcal{A}(k,n)$ y $\mathcal{B}(k,n)$ el siguiente paso consiste en definir los rangos finales formando pares ordenados con elementos de cada uno de estos conjuntos:

$$I_i = [a_i, b_i], \quad a_i \in \mathcal{A}(k,n), b_i \in \mathcal{B}(k,n)$$

El agoritmo final para definir los rangos quedó descrito por la expresión en Supercollide mostrada en la figura..:

```
        |in,pos=0,offset=2,path,level=0.5|
        var
beta,n_channels,max,transition_node,range,data,data_x,alpha,factor,dummy,input,signal;

        data = List[];
        data_x = List[];
        range = List[];
        input = in;

        beta = pi/offset;
        n_channels = path.size;
        max = (path.size-1)*beta;
        transition_node = ((offset+2)/2).floor;
        alpha = n_channels-transition_node;

        dummy = (((offset-0.5)/2).floor);
        factor = ((offset/2)-dummy);

       //if(offset.even,{factor=1},{factor=1/2});


    if(n_channels<=transition_node,{
    n_channels.do({
                range.add([0,beta*(n_channels-1)])});
        },{
    n_channels.do({arg i;
    if(i<alpha,{data.add(offset+(2*i)/2)},{data.add(n_channels-1)});
    if(i<transition_node,{data_x.add(0)},{data_x.add(factor+(i-transition_node))});
          });
range = [data_x*beta,data*beta].flop;
            });

```

Figura 6.14: Algoritmo en Supercollider para definir los rangos generales de acción

Para concluir, solamente restaba integrar los valores de los rangos de una forma adecuada a las funciones componentes con ángulos de defase incrementales a modo tal que la señal de audio de entrada pudiera ser modulada apropiadamente para la distibución multicanal de paneo. Usando de nuevo el Ugen *InRange.kr* y la salida *Out.ar*, se generó el siguiente segmento de código:

```
50
51          signal = Array.fill(n_channels,{arg j;
52
53              Out.ar(path[j],(InRange.kr(pos*max,
54              range[j][0],range[j][1]))*cos((pos*max)-(beta*j)).abs*input*level)
55                              });
56
57                  ^signal;
58              }
59          }
60
```

Figura 6.15: Algoritmo en Supercollider para definir el comportamiento de las salidas individuales

De este modo el algoritmo final es capaz de realizar paneo multicanal para un número arbitrario de canales con trayectorias arbitrarias predefinidas de manera totalmente libre por el usuario, con un barrido de paneo correcto, un diseño estructural simplificado y sintetizado, así como una funcionalidad denominada *spread* que permite distribuir el paneo entre algún cierto número de canales deseados. Este algoritmo no presenta el problema de la versión realizada con funciones seno y resulta altamente estable en funcionamiento dentro de Supercollider además de que el gasto en CPU es bajo.

PanW.ar tiene como argumentos:

- Input. La señal de audio de entrada.
- Pos. La posición del paneo de acuerdo a la trayectoria predefinida ; 0 corresponde al primer canal y 1 al último.
- Level. El nivel de amplitud general del paneo; varía entre 0 y 1.
- Spread. El número de canales entre los cuales se distribuye el paneo (el parámetro k). Varía entre 2 y n, donde n es el número total de salidas.

Por ejemplo, una rápida comparación del presente algoritmo *PanW.ar* con *PanAz.ar* para una salida de ruido blanco (*WhiteNoise.ar*) de 18 canales y un valor de *spread* (*width*, en *PanAz.ar*) de 4 y un control de posición mediante *MouseX.kr*, se obtuvó el registro del siguiente rendimiento del Servidor:

Algoritmo	CPU max	CPU promedio	Canales arbitrarios de salida	spread	spread en tiempo real
PanW	1.91	0.52	sí	sí	no
PanAz	1.83	0.43	no	sí	sí
PanX	1.80	0.41	no	sí	sí

Como se observa de la tabla anterior, el algoritmo PanW presenta un incremento en el uso de CPU del 0.08 % y del 0.11 % con respecto a PanAz y PanX en rendimiento máximo y un incremento del 0.09 % y del 0.11 % con respecto a éstos en el rendimiento promedio. Por otro lado, PanW permite establecer de manera arbitraria y cómoda (mediante un simple array que defina la trayectoria deseada) los canales de salida para los que se desea realizar el paneo, mientras que ni PanAz ni PanX limitan al usuario a una configuración preestablecida por canales consecutivos. Si bien los tres algoritmos permiten distribuir el paneo en un número arbitrario de canales (spread o width), la actual implementación de PanW no soporta el cambio de este parámetro en tiempo real. Lo anterior es debido a una sencilla razón; por motivos de simplificación de código se utilizaron test booleanos dentro de las iteraciones para la definición de los rangos. Un mejoramiento posterior del código se realizará con el fin de extender la funcionalidad del *spread* en tiempo real. A continuación se muestra la clase final construida:

```
Pan_W {
     *ar{

        |in,pos=0,offset=2,path,level=0.5|
        var beta,n_channels,max,transition_node,range,data,data_x,alpha,factor,dummy,input,signal;

        data = List[];
        data_x = List[];
        range = List[];
        input = in;

        beta = pi/offset;
        n_channels = path.size;
        max = (path.size-1)*beta;
        transition_node = ((offset+2)/2).floor;
        alpha = n_channels-transition_node;

        dummy = (((offset-0.5)/2).floor);
        factor = ((offset/2)-dummy);

    if(n_channels<=transition_node,{
    n_channels.do({
                range.add([0,beta*(n_channels-1)])});},
            {
    n_channels.do({arg i;
    if(i<alpha,{data.add(offset+(2*i)/2)},{data.add(n_channels-1)});
    if(i<transition_node,{data_x.add(0)},{data_x.add(factor+(i-transition_node))});
          });
    range = [data_x*beta,data*beta].flop;
            });

        signal = Array.fill(n_channels,{arg j;
            Out.ar(path[j],(InRange.kr(pos*max,
            range[j][0],range[j] [1]))*cos((pos*max)-(beta*j)).abs*input*level)
                              });
               ^signal;
            }
         }
```

Figura 6.16: Algoritmo final en Supercollider del paneador multicanal.

6.4. La clase Map_Chaos

Con el fin de poder generar series de tiempo de manera práctica a partir de sistemas dinámicos de acuerdo a lo descrito en la sección... se diseñó una clase dentro de Supercollider cuya funcionalidad radica en la integración algorítmica de los siguientes sistemas dinámicos: sistemas Lindenmeyer con diccionarios dinámicos estocásticos, ecuación logística, mapeo Henon, mapeo seno caótico, mapeo cúbico, el mapeo tent y el mapeo Duffing.

Para el caso del sistema Lindenmeyer con diccionarios dinámicos estocásticos, se definió un algoritmo que permite generar diccionarios con componentes de intervalos numéricos aleatorios. Dependiendo del número de elementos del diccionario, el algoritmo generará series de datos dentro de los intervalos crecientes:

```
23 //create Dictionary
24 dict {
25 arg number;
26
27     var dict_0 = Dictionary(number);
28     var offs1 = {|i| ("base"++i).asSymbol};
29     var offs2 = {|i| ("sub_base"++i).asSymbol};
30
31     number.do({arg i;
32
33         dict_0.add((offs1.(i) -> Prand(Array.fill(12,{Array.fill(rrand(5-(2*i),7-(2*i)),
34             {rrand((0.3+i),(0.5+i))})}),inf).asStream));
35
36         dict_0.add((offs2.(i) -> Prand(Array.fill(12,{Array.fill(rrand(4-(i+1),6-(i+1)),
37             {rrand((0.5+i),(0.8+i))})}),inf).asStream));
38     });
39     dictionary = dict_0;
40     }
41
```

Figura 6.17: Algoritmo en Supercollider para definir diccionarios dinámicos estocásticos

A continuación se definieron siete sistemas Lindenmeyer que hacen uso del mismo diccionario dinámico estocástico, teniendo por consiguiente siete reglas de producción distintas para una misma base de datos, las cuales son ejecutadas por la clase *Prewrite* de Supercollider.

Como resultado final, se genera un array que contiene el resultado en serie de datos de cada uno de los siete sistemas distintos. Si se sustituyen las entradas numéricas de los diccionarios por secciones de envolventes previamente construidas, es posible generar envolventes híbridas a partir de distintas fuentes de información y que pueden presentar características de variación dinámica interesantes; la anterior idea se detalla en la siguiente sección.

Los demás casos de sistemas dinámicos son sencillas implementaciones dentro de Supercollider de los mapeos ya mencionados en el capítulo 3, los cuales arrojan como resultado, las órbitas para un conjunto de condiciones iniciales dadas, estas órbitas forman por supuesto, el raw data en la generación de las *trayectorias caóticas*, pero

```
        //first rule->
        this.dict(dict_num);
        iterations.do({arg i;
a = Prewrite(Pseq([\base0]), // start with half
                (     \base0: #[base2,sub_base1,base1,sub_base0],
                       \base1: #[sub_base2,sub_base0],
                       \base2: #[base0,sub_base1,base0],
                       \sub_base0: #[sub_base2,sub_base1],
                       \sub_base1: #[base1,sub_base0],
                      \sub_base2:#[sub_base0,sub_base1,base0]

            ), (i+1));

y = a.asStream.all;
y = y.collect({|i| dictionary[i].next});
            lin_data_a.add(y.flat);


//second rule ->
b = Prewrite(Pseq([\base1]), // start with half
                (     \base0: #[sub_base1,base2],
                       \base1: #[base0,sub_base1,sub_base0,base2],
                       \base2: #[base0,base2],
                       \sub_base0: #[sub_base0,sub_base2],
                       \sub_base2: #[sub_base0,base0],
                      \sub_base1:#[sub_base0,sub_base1]

            ), (i+1));
z = b.asStream.all;
z = z.collect({|i| dictionary[i].next});
            lin_data_b.add(z.flat);


    //third rule->
c = Prewrite(Pseq([\base2]), // start with half
                (     \base2: #[base1,sub_base0,base1,sub_base2],
                       \base0: #[sub_base2,sub_base0],
                       \base1: #[base0,sub_base1,base0],
                       \sub_base2: #[sub_base2,base1],
                       \sub_base0: #[base1,sub_base0],
                      \sub_base1:#[sub_base0,base0]

            ), (i+1));
w = c.asStream.all;
w = w.collect({|i| dictionary[i].next});
```

Figura 6.18: Algoritmo en Supercollider para definir las reglas de iteración de los Sistemas Lindenmeyer.

pueden ser usadas con cualquier otro propósito deseado. El llamado a la clase se realiza como se muestra en la figura:

```
//basic chaotic panning system

(//create envelopes from data series (logistic equation);
x = Map_Chaos.new; //create a new Map_Chaos object
x = x.logistic(10,0.5,3.8); //iterations, initial condition, control parameter.
y = x.reverse.keep(y.size-1);
)


(//create envelopes from data series (lindenmeyer);
x = Map_Chaos.new; //create a new Map_Chaos object
x = x.linden(3,1.6,4); //number of dictionaries, ratio, lindenmeyer iterations.
y = x.reverse.keep(y.size-1);
)
```

Figura 6.19: Algoritmo en Supercollider para definir los rangos generales de acción

6.4.1. Búsqeda de la gestualidad mediante hibridación.

Una de las características deseables al momento de generar procesos automatizados que controlen parámetros sonoros, es el evitar la monotonía o percepción de un cierto patrón fijo durante un lapso de tiempo demasiado prolongado, que es de hecho una de las virtudes de la ejecución humana sobre la automatizada: *la dinámica gestual*. Si bien el uso directo de las órbitas generadas por los sistemas dinámicos presentados, es capaz de producir material muy prolífico e interesante para el desarrollo composicional o de performance, también es cierto que aún a pesar de que dichas órbitas tengan comportamiento caótico, la percepción audible de un cierto parámetro controlado por éstas, podría llegar a estancarse dinámicamente. Con el fin de aumentar más posibilidades variacionales y sobre todo riqueza dinámica, son necesarios cambios de cierto modo bruscos y que sean perceptualmente diferenciables por el escucha, como si se tratase de la manipulación en tiempo real de un dispositivo interactivo. Se propone aquí, que uno de los factores decisivos en la búsqueda de la gestualidad algorítmica es la serie de datos que gobierna la línea de ocurrencia temporal o como se denominó en el capítulo 4, *el índice de tiempo*.

Aunque se tenga una órbita mapeada al valor del parámetro sonoro deseado (eje y) rica en variaciones, si el índice temporal no presenta cambios significativos a lo largo de su desarrollo, esta propiedad puede llegar a perderse. Estos cambios deben formar clusters o subgrupos de valores contrastantes para que el efecto audible llegue a tener ciertas propiedades gestuales. Un ejemplo clarificará la idea mencionada. Supongase que se generó una envolvente que controla la amplitud del paneo a partir de la integración en serie de datos de una órbita del mapeo duffing (para el valor del parámetro) y otra del mapeo logístico (para el índice de tiempo), ambas con condiciones iniciales que se sabe generan comportamiento caótico en los sistemas correspondientes. Dicha envolvente se muestra en la figura..:

En la figura anterior se muestra en la parte superior la envolvente en serie de tiempo del mapeo Duffing con retardo constante e igual a 1 y en la parte inferior la envolvente generada mediante la integración en el índice de tiempo de la ecuación logística. A simple vista la envolvente conserva su forma original y permanece casi inalterada, sin embargo si se observa con cuidado las variaciones se presentan de forma significativa en la escala de tiempo que es precisamente lo que se desea alterar.

Si bien se observa una envolvente realmente interesante con variaciones útiles en su estructura, también es cierto que los cambios a lo largo del índice temporal son suaves a lo largo de toda la envolvente. Sin embargo, en la búsqueda de la gestualidad aquí propuesta es posible explorar variaciones aún más expresivas. Existen innumerables formas de realizar lo anterior, cada compositor puede decidir su particular método; desde construcciones totalmente a mano hasta métodos basados en cualquier proceso algorítmico, como redes neuronales, aprendizaje máquina, cadenas de Markov, autómatas finitos, etc. Por supuesto, la idea más lógica e inmediata para generar índices temporales con las características mencionadas sería la concatenación

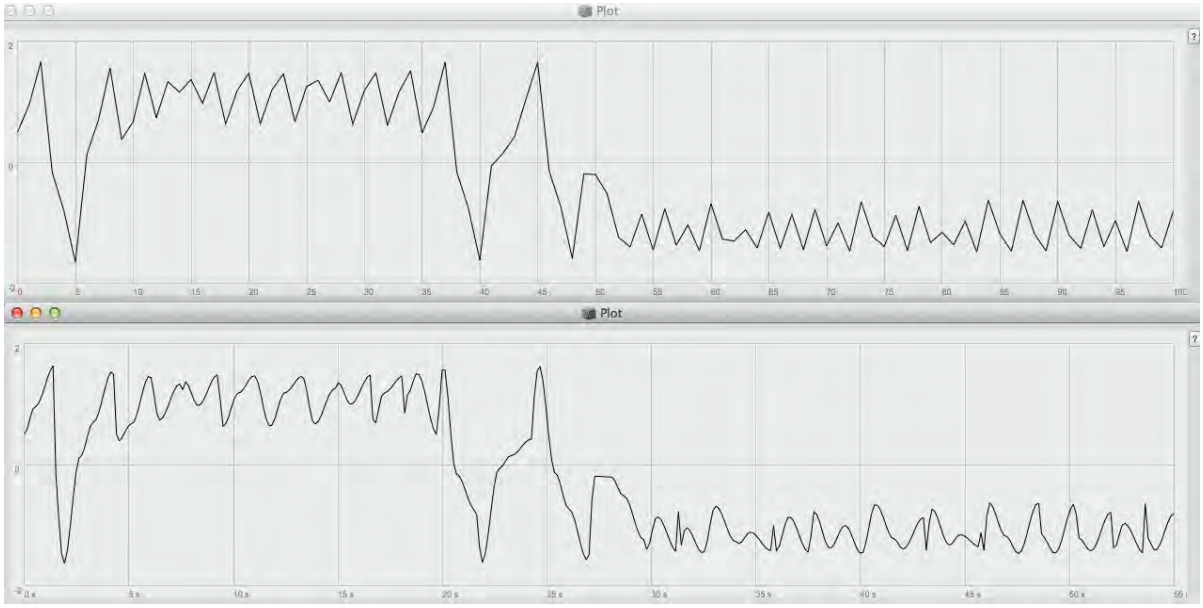


Figura 6.20: Diagrama de bifurcación, mapeo cúbico.

de distintas órbitas con diferentes escalamientos, por ejemplo observése la figura... que es la envolvente resultante de introducir en el índice temporal la concatenación de cinco órbitas de mapeos distintos con distintos valores de escalamiento:

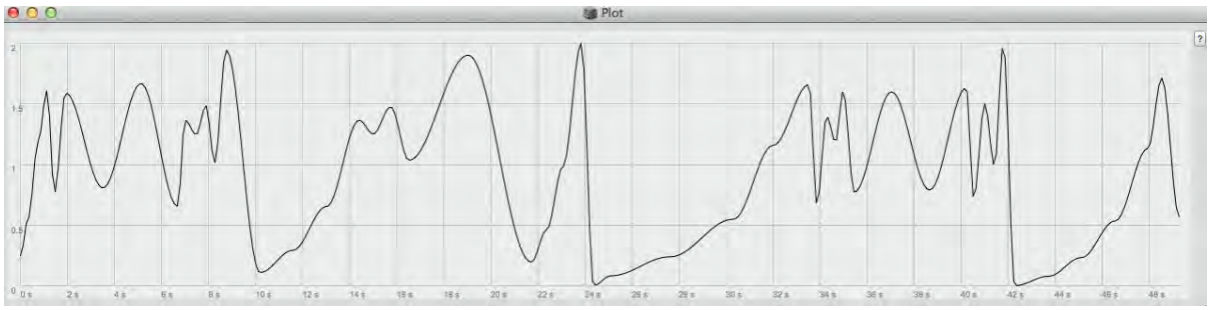


Figura 6.21: Diagrama de bifurcación, mapeo cúbico.

De hecho, basados en esta idea básica de concatenación se desarrolló la instancia de los sistemas lindenmeyer de la clase *Map_ Chaos*. Se generan diccionarios estocásticos y cada elemento del mismo es sustiuido iterativamente por el sistema lindenmeyer, dotando al diccionario de grupos de valores contrastantes y definidos, se generan clusters alternantes de forma fractal tras la ejecución del sistema Lindenmeyer. Esta es una de las formas primarias que se proponen aquí en la exploración de la gestualidad.

Como segunda opción se propone la exploración de la transformación de las series de datos originales mediante aplicación de isometrías o concatenación básica con

funciones. En la figura... se muestra una envolvente generada mediante la multiplicación de una órbita caótica del mapeo tent con la función $xcos(x)$ en el índice de tiempo y una variación obtenida mediante la multiplicación de la órbita del mapeo tent consigo misma.

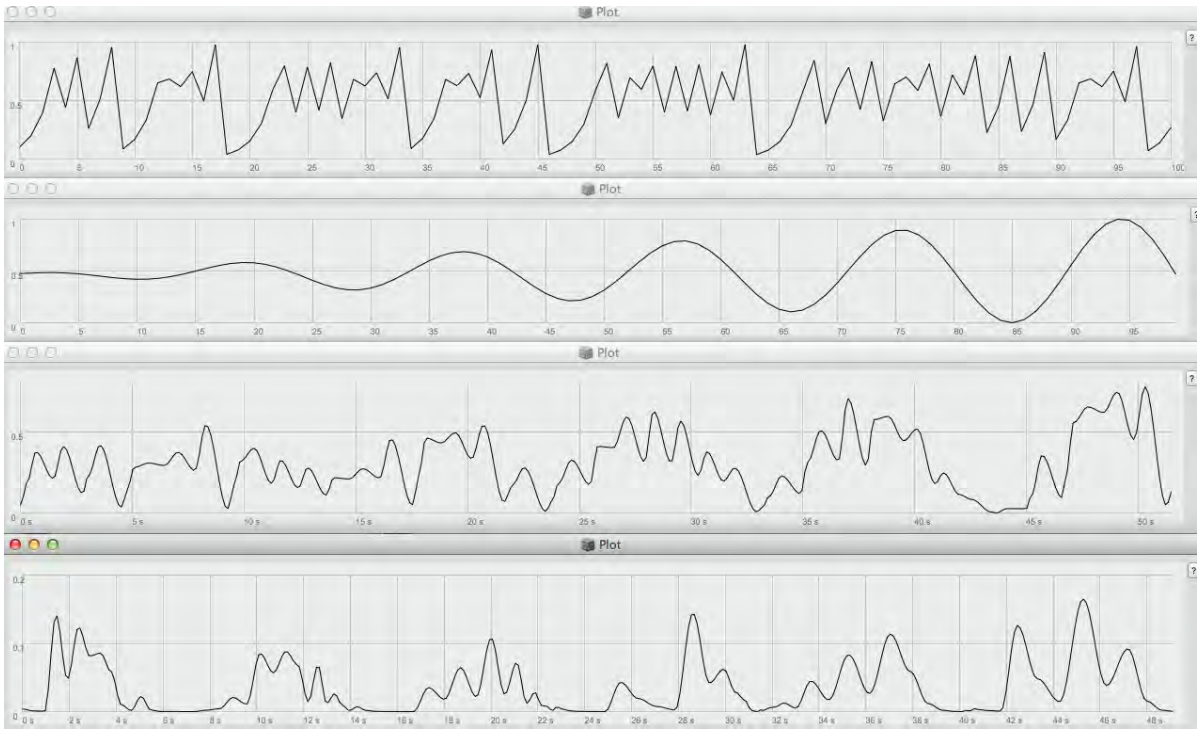


Figura 6.22: Envolvente generada por modulación de transformaciones de las órbitas del mapeo Tent y la función $xcos(x)$

Como puede observarse el método anterior está claramente basado y es análogo al concepto de *modulación* de señales, como por ejemplo el am o el fm. A partir de este principio pueden generarse infinidad de transformaciones dada una órbita única primaria. Esto puede establecerse fácilmente del modo siguiente.

Sean $\{x_n\}$, $\{y_n\}$, series de tiempo obtenidas por cualesquiera fuentes de información, sean además $f^1_m(x_n)$ y $f^2_m(y_n)$, funciones moduladoras cuyo propósito es realizar una transformación de cualquier tipo sobre cada una de las series correspondientes, entonces las series resultantes tras las transformaciones respectivas estarán dadas por:

$$g(x) = f[^1_m(x_n) \quad \& \quad g(y) = f^2_m(y_n)$$

y la envolvente generada será el resultado de la integración de cada serie a un parámetro objetivo; el índice de tiempo y el valor asociado.

En la figura ... se muestran algunas variaciones usando potencias de la misma serie

y transformaciones por polinomios, además se utilizan distintos grados de curvatura para la interpolación entre cada punto:

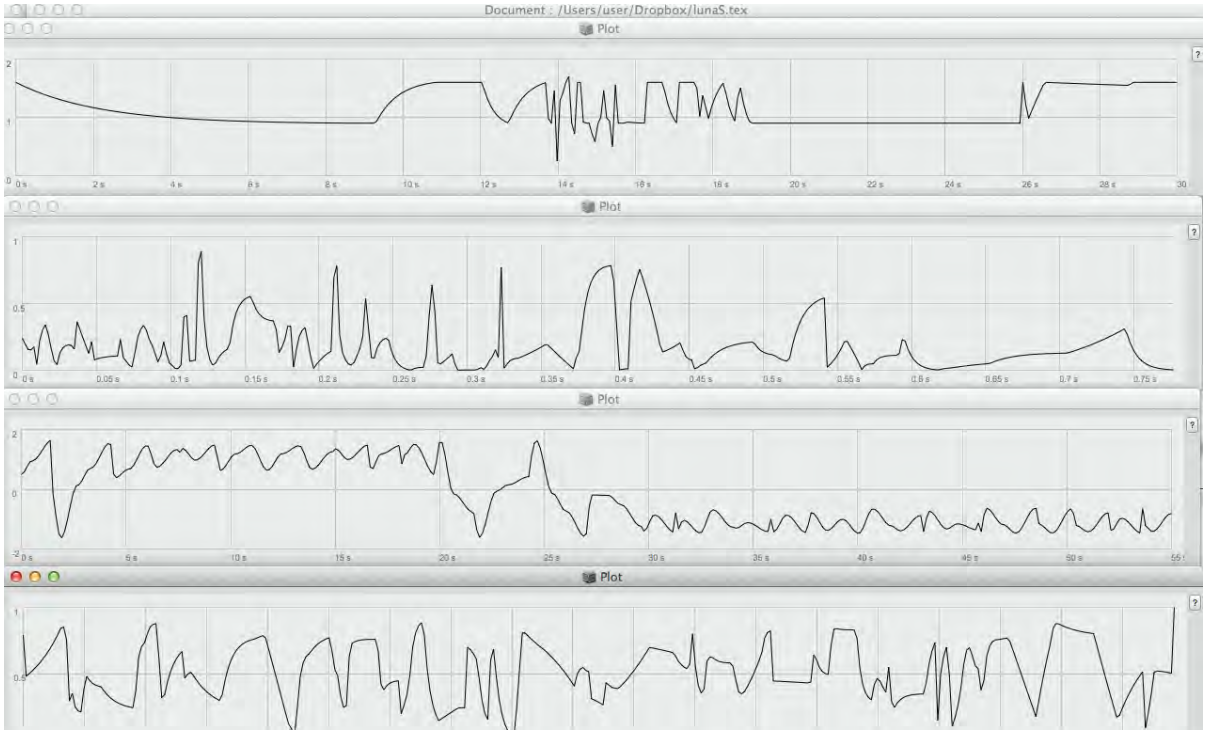


Figura 6.23: Envolvente generada por modulación de transformaciones de las órbitas del mapeo Tent y la función $xcos(x)$

6.5. La clase Chaos Pan

Para concluir el sistema de paneo multicanal, se construyó una clase final que genera envolventes a partir de las series de datos de los sistemas caóticos (o de cualquier otra fuente), permitiendo controlar los parámetros sonoros: posición del paneo, amplitud general del paneo, mix del reverb, room del reverb, damp del reverb. Como reverb, en este caso se utilizó el Ugen FreeVerb.ar implementado en la distribución normal de Supercollider, pero la clase puede modificarse para utilizar cualquier otro Ugen; más aún para controlar cualquier otro efecto o parámetro deseado.

El sistema se pensó para ser utilizado tanto dentro de un contexto de performance en vivo, como una herramienta de composición. Para ello, el sistema *Chaos_Pan*, permite al usuario elegir entre trayectorias definidas por él mismo o trayectorias generadas algorítmicamente. En el último caso las trayectorias que se generan poseen la cualidad que ningún canal se repite. Con fines de eficiencia de ejecución se estableció un test booleano que elegía alguna de las dos inicializaciones mencionadas como se muestra en la figura..:

```
Chaos_Pan {
    var <>proxy, <>path_proxy,<>pos,<>room,<>level,<>mix,<>damp,<>path,<>low_index,<>high_index,
    <>num_channels,<>bus_chain,<>bus_input,<>s,<>time_min,<>time_max,<>average,<>synth_fade_time;

    *new {
    arg
num_channels,bus_chain,path_choice,low_index=nil,high_index=nil,path=nil,synth_fade_time=nil;
        if(path_choice == 0,{
^super.new.init_pre(num_channels,bus_chain,path,synth_fade_time)},
            {^super.new.init_calc(num_channels,bus_chain,low_index,high_index,synth_fade_time)});
    }

    init_pre {
        arg num,bus,path_p,fade;

        bus_chain = bus;
        num_channels = num;
        path_proxy = path_p;
        synth_fade_time = fade;
        s = Server.default;


    }

    init_calc {
        arg num,bus,low,high,fade;

        bus_chain = bus;
        num_channels = num;
        low_index = low;
        high_index = high;
        synth_fade_time = fade;
        s = Server.default;
        this.path_calc;
            }
```

Figura 6.24: Algoritmo en Supercollider para la inicialización de la clase Chaos_Pan

El siguiente paso consiste en alojar al paneador en una estructura de audio del servidor. En este caso se eligió el *NodeProxy* por encima del *SynthDef* debido a que se ajustaba de mejor modo para fines de procesos en tiempo real y su llamada al servidor resulta un poco más sencilla y directa desde el punto de vista de eficiencia computacional. Esta inicialización denominada *set_source*, comienza haciendo una llamada a otra inicialización: *env_group*, la cual genera las envolventes a partir de las series de datos definidas como argumentos en *set source*. Recuérdese que como se mencionó en la sección... las series de datos se construyen de fuentes independientes para la línea de tiempo y para la linea de valor en sí, por lo tanto, si se desean controlar m parámetros sonoros, se necesitarán $2m$ series de datos para generar las m envolventes necesarias. El actual estado del sistema está configurado para manejar los cinco parámetros sonoros mencionados al inicio de la sección, sin embargo es muy sencillo extender e implementar nuevos parámetros sonoros, como filtros de banda, reverbs con algoritmos más complejos, delays, etc. En la figura ...se observa la llamada de clase *set_source*:

Cuando se hace la llamada *set_source*, el NodeProxy es creado, su fuente, servidor y número de canales son definidos, y se inicializa; todo ésto es realizado simultáneamente con la instrucción **.audio(s,num_channels).source_ { }.play** . El bus de entrada es definido por el argumento *bus_chain* por lo que la salida de cualquier instrumento, SynthDef, o señal de audio que quiera sera paneada mediante *Chaos_Pan*,

```
48      set_source {
49          arg pos_data,pos_indx,level_data,level_indx,mix_data,mix_indx,room_data,
50              room_indx,damp_data,damp_indx,loop=false;
51       if(loop==false,{
52          this.env_group(pos_data,pos_indx,level_data,level_indx,mix_data,mix_indx,room_data,
53                  room_indx,damp_data,damp_indx)},
54      {this.env_group_loop(pos_data,pos_indx,level_data,level_indx,mix_data,mix_indx,room_data,
55                  room_indx,damp_data,damp_indx)});
56
57
58          proxy = NodeProxy.audio(s,num_channels).source_{
59          arg t_gate=0,doneAction=0,spread=1;
60          var input,mix_env,room_env,damp_env,pos_env,level_env;
61
62              mix_env = EnvGen.kr(mix,gate:t_gate,doneAction:doneAction);
63              room_env = EnvGen.kr(room,gate:t_gate,doneAction:doneAction);
64              damp_env = EnvGen.kr(damp,gate:t_gate,doneAction:doneAction);
65              pos_env = EnvGen.kr(pos,gate:t_gate,doneAction:doneAction);
66              level_env = EnvGen.kr(level,gate:t_gate,doneAction:doneAction);
67
68          input = In.ar(bus_chain);
69          input = FreeVerb.ar(input,mix_env,room_env,damp_env,0.9);
70
71          PanW.ar(input,pos_env,level_env,spread,path_proxy);
72          }.play
73          ^proxy;
```

Figura 6.25: Algoritmo en Supercollider para definir los rangos generales de acción

deberá ser ruteada a través de dicho parámetro. Si bien con la llamada *set_source* el NodeProxy se define y se activa, como puede observarse, todas las envolventes tienen el parámetro *gate* encadenado a un sólo trigger: *t_gate*, por lo que el sistema comenzará la ejecución de todas las envolventes (y por consiguiente del paneo general final) al momento de que dicho argumento pase del valor 0 (que es su estado predefinido) a 1. Esto se realiza mediante un llamado extra denominado *go*, el cual cuenta con tres funcionalidades de disparo de ejecución distintas:

- **go_loop**. Permite definir nodos generales de loop para que todas las envolventes se repitan un número definido de veces. Posee como argumentos: *repetitions* (número de repeticiones del loop), loop_choice y define. Para loop_choice se tienen las siguientes opciones:
  - 0. Toma la duración mínima de todas las envolventes y asigna este valor al intervalo del loop.
  - 1. Toma la duración máxima de todas las envolventes y asigna este valor al intervalo del loop.
  - 2. Toma la duración promedio de todas las envolventes y asigna este valor al intervalo del loop.
  - 3. Permite al usuario establecer de manera manual el valor del nodo mediante el parámetro define.

- **go_keep**. Inicializa las envolventes y deja el NodeProxy activo.
- **go_once**. Inicializa las envolventes y libera el NodeProxy al término de la envolvente selccionada (usando las mismas opciones que en *go_loop*); esta duración es sumada al parámetro *synth_fade_time* predefinido por el usuario el cual representa el valor de la duración del fade out tras la finalización del instrumento del usuario.

Con el fin de conocer los valores de tiempos máximo, mínimo y promedio de las envolventes, la llamada *time_info* los calcula y los muestra en el post window. La llamada a Chaos_Pan puede realizarse tal y como se muestra en la figura:

```
34 //option 1..manually define path
35 a = Chaos_Pan.new(bus_chain:10,path_choice:0,path:[0,1,2,3],synth_fade_time:2.6)
36
37 //option 2...let the computer define the path (non repeating channel´s index).
38 a = Chaos_Pan.new(num_channels:4,bus_chain:10,path_choice:0,low_index:0,high_index:10,synth_fade_time:2)
39
40 //in both cases define envelopes for reverb mix, reverb room, reverb damp, panner position and panner
   level...there are requeried in total 10 data series a pair --> data-index.
41 a.set_source(x1,y1,x2,y2,x3,y3,x4,y4,x5,y5);
42 //this instruction also returns the NodeProxy which contains the panner.
43
44 //start and clear the panner...several options
45 a.go_loop(2,2) //options for loop nodes are: 0->min_time (the algorithm calculates the total duration of
   each envelope and returns the minimum time, 1->max_time, 2->average time, 3->arbitrary time(user
   defined).
46
47 a.go_keep //starts the panner and keeps the signal flowing after the envelope is done(doneAction=0);
48
49 a.go_once(1) //starts the panner and release the NodeProxy in a time that matches user´s synth
   fadeout...this time is defined in the .new call.
```

Figura 6.26: Algoritmo en Supercollider para definir los rangos generales de acción

Como se observa en la figura anterior, Chaos Pan admite los argumentos:

- **num channels**. El número de canales en el caso en que se desee que el ordenador defina algorítmicamente la trayectoria de espacialización.
- **bus_chain**. El número del bus para rutear entre el instrumento y el paneador. Valores de buses arriba de 100 son buena opción para diferenciarlos de los buses de salida física.
- **path_choice**. 0 si el usuario desea definir la trayectoria manualmente mediante un array. 1 si se desea que la trayectoria sea definida por el ordenador.
- **low_index**. Cuando se deja que el ordenador defina la trayectoria, este valor define el número de bus de salida mínimo.
- **high_index**. Cuando se deja que el ordenador defina la trayectoria, este valor define el número de bus de salida máximo.
- **path**. El array que define la trayectoria de espacialización definida por el usuario.

- **synth_fade_time**. El tiempo agregado que espera el servidor para liberar el NodeProxy del sistema, que debe ser igual a la duración del fade out del instrumento del usuario en caso de que exista.

6.6. El sistema Chaos_Map dentro de la Ópera Donahi

El 28 y 29 de noviembre tuvo lugar en Oaxaca, el estreno mundial de la Ópera multimedia multicanal en zapoteco, Donahi, de el renombrado compositor mexicano, el Dr. Roberto Morales Manzanares. Además de ser de las pocas propuestas actuales cuyo material melódico y orquestal está realizado en su mayoría en base a procesos algorítmicos, en esta ópera, Morales Manzanares hace un uso extensivo de live electronics basado en procesamientos en tiempo real de distintas fuentes. A petición del Dr. Morales, algunos de estos procesamientos debían ser espacializados en tiempo real dentro de una configuración multicanal fija en base a trayectorias caóticas.

Donahi, una ópera de poco más de 110 minutos, fue el primer ejemplo donde el sistema *Chaos_Map* se probó de manera aterrizada y formal. El sistema funcionó satisfactoria y correctamente durante las casi dos horas de cada una de las dos presentaciones de la ópera mencionada. De hecho una de las razones en el rediseño general del sistema fue precisamente la necesidad de que se ocupara la menor cantidad de CPU ante procesos de espacialización simultáneos, puesto que la cantidad de instrumentos que el Dr. Morales usa en tiempo real, son considerables.

El segundo punto respecto a la espacialización radicaba en una búsqueda de la gestualidad de la distribución del sonido, de tal modo que el movimiento automatizado general de la señal no fuera lineal o continuo sino que presentara una morfología dinámica como si se tratase de un parámetro controlado por un usuario. Para ello y debido a la urgencia del tiempo, fue que se diseñó la primera de las opciones en la clase *Map_Chaos*; *sistemas Lindenmeyer con diccionarios dinámicos estocásticos* combinados con los mapeos caóticos.

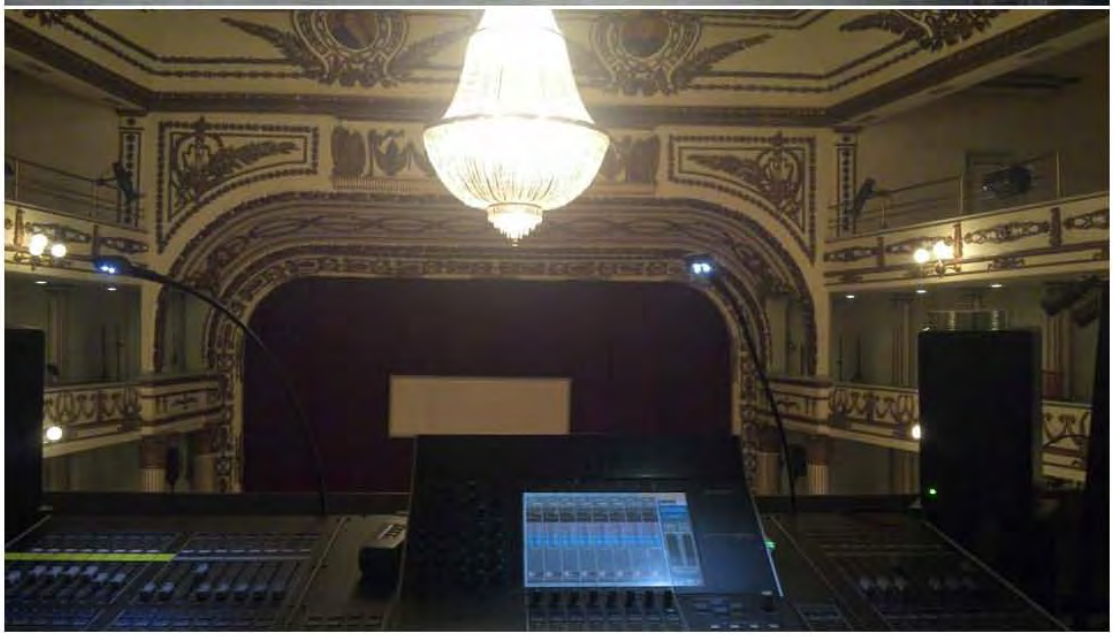
Como se mencionó en la sección anterior, el efecto de la gestualidad depende en gran medida de la serie de datos mapeada al tiempo, por lo tanto, variaciones dinámicas sobre la línea de tiempo permiten generar percepciones gestuales más concretas. Para este propósito se utilizaron los sistemas Lindenmeyer por ser de fácil implementación y por permitir generar, mediante la sustitución iterada, bloques de valores interpuestos de manera fractal.

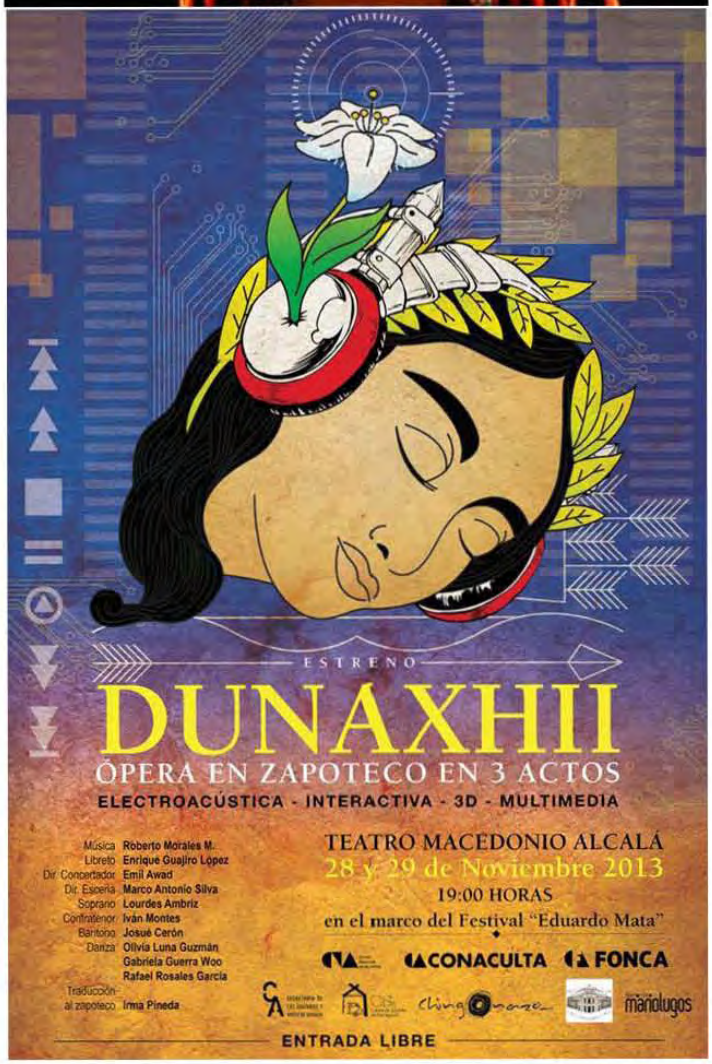
Cada vez que se hace el llamado $.linden()$, se genera un array bidimensional $linden[i][j]$.La primer componente i, hace referencia al sistema lindenmeyer utilizado, mientras que el segundo índice j es el resultado de iterar dicho sistema $j+1$ veces. Entonces cada

llamada tendrá un número de componentes lindenmeyer $7 * q$ donde q es el total de iteraciones definidas por el usuario.

Adapatándose al equipo disponible del lugar, se montó un sistema hexafónico en el paraiso del Teatro Macedonio Alcalá, con bocinas de 10". Dado que la distancia entre paraiso y la salida más alta del PA era de poco más de 4 metros, se pudo generar un movimiento espacial definido respetando los niveles de la orquesta y cantantes. Aplicando normalización a las envolventes se pueden establecer los rangos correctos y apropiados para cada parámetro sonoro específico deseado, por lo que la implementación de las envolventes a los parámetros sonoros resultó ser de manera directa y aunado al hecho de la colocación del sistema canal, en ningún momento de la ópera se tuvo problema alguno de feedback y el control con trayectorias caóticas sobre cada parámetro sonoro tanto de reverb como del paneador se pudo apreciar correctamente.


DUNAXHII




ESTRENO
DUNAXHII
ÓPERA EN ZAPOTECO EN 3 ACTOS
ELECTROACÚSTICA - INTERACTIVA - 3D - MULTIMEDIA
Música Roberto Morales M.
Libreto Enrique Guajiro López
Dir. Concertador Emil Awad
Dir. Escena Marco Antonio Silva
Soprano Lourdes Ambriz
Contratenor Iván Montes
Barítono Josué Cerón
Danza Olivia Luna Guzmán
Gabriela Guerra Woo
Rafael Rosales García
Traducción al zapoteco Irma Pineda
TEATRO MACEDONIO ALCALÁ
28 y 29 de Noviembre 2013
19:00 HORAS
en el marco del Festival "Eduardo Mata"
CONACULTA
FONCA
mariolugos
ENTRADA LIBRE

6.7. Conectividad con Renders y control por OSC

Una de las grandes ventajas de las aplicaciones computacionales es el portabilidad en el uso del OSC dentro de Supercollider para poder controlar cualquier aplicación de dicho lenguaje con cualquier aplicación o dispositivo compatible.[5]

De las más sencillas formas de controlar *PanW* mediante OSC, es a través de *TouchOSC* en algún dispositivo touch de Apple. Se construye la interfaz gráfica en el *Touch OSC Editor*, despúes se carga en el dispositivo y tras haber escrito el código de enlace de mensjes OSC en Supercollider se puede tener control en tiempo real sobre los parámetros deseados mediante una conexión WiFi.

Gracias a la versatilidad de Supercollider se puede tener el mismo control pero usando MIDI, por lo que el paneador puede ser controlado por cualquier dispositivo o aplicación midi. Renders como Spatium que generan espacios acústicos virtuales basados en VBAP o Ambisonics pueden ser utilizados, conectando la salida de Supercoliider a la entrada de los mismos mediante ruteadores como JackPilot o SoundFlower y de este modo explorar en las posibilidades de la espacialización.

[5]Véase *OSC Communication* dentro del archivo de ayuda de Supercollider

Capítulo 7

Conclusiones

El presente trabajo es un modelo teórico de espacialización multicanal basada en procesos algorítmicos con una implementación práctica desarrollada en Supercollider. Supone un planteamiento formal desde el punto de vista categórico-matemático e intenta mostrar cómo los sistemas dinámicos pueden servir como herramientas concretas para generar raw data en el proceso de composición y en un ambiente de live performance dentro del contexto de la música electroacústica.

Este trabajo pretende ser una estructura firma y bien definida cuya aportación radique en la formación de ese marco referencial bien definido en base al cual sea posible utilizar ciertos conceptos y herramientas matemáticas bien definidas para su aplicación en el campo de la composición y ejecución de la música electroacústica. Se incita al lector a no entender ni valorar el presente trabajo únicamente como la generación de un algoritmo escrito dentro de un lenguaje de programación, sino como el planteamiento general de una propuesta creativa basada en ciertos procesos matemáticos que de manera incluyente, y como resultado, ofrece una implementación práctica en SuperCollider.

Los algoritmos de espacialización generados poseen una función distinta a los sistemas como VBAP o Ambisonics; de hecho, dado que el paneador aquí propuesto realiza colocaciones dinámicas del sonido mediante configuraciones arbitrarias predefinidas cuyos parámetros sonoros son controlados mediante trayectorias caóticas creadas a partir de la incorporación de distintos sistemas dinámicos, los sistemas como VBAP o Ambisonics, pueden ser usados al final de procesamiento de la señal tanto para emular espacios acústicos definidos, como para trabajar de acuerdo a las características físicas propias del lugar.

Si bien existe un planteamiento estético discreto de fondo inherente al proceso empleado asociado con el concepto del caos y de los objetos matemáticos utilizados como herramientas, en este trabajo no se pretende ahondar en ello, sino únicamente hacer mención del mismo tal y como se presentó en la introducción de la presente

tesis. Futuros trabajos profundizarán en este tópico de gran importancia para el autor.

Respecto al uso de sistemas dinámicos caóticos como generadores de flujo de datos se pudo apreciar que funcionan satisfactoriamente y además, gracias al concepto de hibridación, es posible generar envolventes con un cierto grado de gestualidad algorítmica respecto al control sobre el parámetro específico que se mapea. Se plantea que el compositor no necesita estar preparado en las formalidades y tecnicismos matemáticos para adentrarse en este modelo, pero sí comprender de manera general la sustentación de los sistemas dinámicos para su posterior uso como herramienta composicional. En este sentido, la asimilación de los conceptos aquí presentados al respecto, pueden ayudar significativamente en este proceso.

Al generar todo un marco categórico compuesto de estructuras conceptuales y definiciones, el modelo es capaz de incluir gran cantidad de casos particulares de interacción tecno-científica en el ámbito de la música electroacústica, tanto para análisis como para desarrollo. Este marco teórico permite, además de una formalización, una estructuración coherente y estable de métodos y procesos utilizados comúnmente por los artistas para su creación composicional.

7.0.1. Futuros trabajos

.

Para futuros trabajos se consideran los siguientes puntos específicos:

- Extensión del catálogo de mapeos caóticos dentro de la clase *Map_ Chaos*.
- Exploración de trayectorias generadas con el uso de redes neuronales, aprendizaje máquina, autómatas celulares y autómatas finitos.
- Desarrollo más a fondo del concepto de *Hibridación* usando los procesos algorítmicos anteriores.
- Exploración y experimentación detallada de la *gestualidad algorítmica* haciendo uso de distintos mapeos y procesos algorítmicos.
- Mejora en el diseño del algoritmo *PanW* de modo que sea posible cambiar el valor del spread en tiempo real. Esta mejora radica en cambiar los test booleanos en la definición de los rangos, por procesos iterativos independientes.
- Extensión en el diseño de la clase *Chaos_ Pan* para poder incluir un mayor número de parámetros sonoros a controlar.
- Diseño de una interfaz gráfica basada en GUI para el algoritmo *PanW*. Si bien esta opción no es tan necesaria debido a que de acuerdo a como está diseñado el

algoritmo, la manera de operarlo resulta sencilla y rápida, dicha funcionalidad extendería las capacidades del mismo desde el punto de vista operativo por parte del usuario.

- Desarrollo del concepto de *estética del proceso* e *interacción tecno-científica.*

Capítulo 8

Apéndices

8.1. A1. Notación matemática.

En el presente trabajo se usará la notación matemática usual, además de conceptos básicos en los temas de: funciones, sucesiones, álgebra lineal, Topología básica de $\mathbb{R}^3$, geometría, isometrías euclideanas y números complejos. Se da por supuesto el hecho de que el lector está familiarizado con conceptos básicos de matemáticas relacionados a teoría de conjuntos, álgebra lineal, ecuaciones en diferencias y sistemas dinámicos. Si bien los conceptos y el material matemático utilizado no es demasiado especializado, se recomienda al lector consultar (Lang 1986), (Seade Kuri 2005), (Haaser 1987), (Lang 1988), (Weiss 2008), (Gonze 2011), (Alligood 200), (Roitman 2011) y (Lang 2004) para mayor referencia.

A continuación se enlista la notación matemática que se usará a lo largo de todo el trabajo:

- $\forall$ - para todo.
- $\in$ - pertenece a.
- $\subseteq$ - subconjunto de.
- $\sim$ - equivalente a.
- $\exists$ - existe.
- $\wp()$ - conjunto potencia de.
- $\complement()$ - complemento del conjunto.
- $\mathbb{R}$ - conjunto de los números reales.
- $\mathbb{N}$ - conjunto de los números naturales.
- $\mathbb{Z}$ - conjunto de los números enteros.

- $\mathbb{Q}$ - conjunto de los números racionales.
- $\mathbb{R}^n = \underbrace{\mathbb{R} \times \mathbb{R} \ldots \mathbb{R}}_{\text{n - veces}} = \prod_{i=1}^{n} \mathbb{R}_i$, con $\mathbb{R}_i = \mathbb{R}$, para todo $i = 1, 2, \ldots, n$.
- $A = [a_{ij}]$ - matriz de tamaño $m \times n$, con elementos a_{ij}, $i = \overline{1, n}$, $j = \overline{1, n}$ y $m, n \in \mathbb{N}$.
- A^T - transpuesta de una matriz.
- A^{-1} - inversa de una matriz.
- $|a|$ - módulo o valor absoluto de a, para $a \in \mathbb{R}$.
- $\|\vec{x}\|$ - norma euclidiana del vector $\vec{x}$, para $\vec{x} \in \mathbb{R}^n$.

Para escribir el símbolo matématico de *tal que*, se usará $\pitchfork$. En el caso que se refiera a un índice i variando dentro de un cierto rango de números enteros (generalmente positivos) desde 1 hasta n, se usará $i = \overline{1, n}$.Para escribir la expresión *si y solo si* se utilizará el símbolo $\Leftrightarrow$. La abreviación *p.e.*, se usa para escribir *por ejemplo* y la locución abreviada en latin *i.e.* (id est), para escribir *es decir* o *esto es*.

Para denotar vectores $\vec{x} \in \mathbb{R}^n$, se usan las notaciones $\vec{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$ ó $\vec{x} = (x_1, \ldots, x_n)^T$ y suponiendo que en general se sobreentiende el contexto, se omite el símbolo de matriz transpuesta y la flecha de vector: $x = (x_1, \ldots, x_n)$.

8.2. A2. Preeliminares matemáticos.

A continuación se ofrecen tópicos generales sobre los preeliminares matemáticos más usados en el presente trabajo. Dado que la construcción del modelo general está basada casi en su totalidad en teoría de conjuntos y tópicos de topología y geometría, se recomienda al lector revisar esta sección para poder tener un marco referencial sólido. Así mismo se exhorta al lector no familiarizado a que consulte las fuentes citadas para tal efecto.

8.2.1. Conjuntos

Un conjunto es una colección cualquiera de cualesquiera elementos los cuales generalmente cumplen ciertas condiciones comunes a todos ellos. Si dichos elementos son a su vez conjuntos, el conjunto es denominado *familia de conjuntos*. Las demostraciones de algunos resultados básicos se omiten suponiendo la familiaridad del lector con el tema, sin embargo en cada caso se hará referencia a la bibliografía específica que contiene dichas pruebas.

Definición 8.1. *Sean X y Y dos conjuntos cualesquiera, entonces:*
i) $X = Y$ sii $X \subset Y$ y $Y \subset X$.
ii)$X \setminus Y = \{x \in X : x \notin Y\}$, se denomina el complemento de Y en X
iii)si $\{E_i\}_{i\in I}$ es una colección de conjuntos, entonces:

$$\bigcup_{i\in I} E_i = \{x : x \in E_i \text{ para algún } i \in I\}$$

se denomina unión de los conjuntos E_i.
Y al conjunto:

$$\bigcap_{i\in I} E_i = \{x : x \in E_i \, \forall i \in I\}$$

se le llama intersección de los conjuntos E_i, donde I es un conjunto de índices cualquiera.
iv) $\wp(X) = \{A : A \subset X\}$ es llamado el conjunto Potencia de X.

Proposición 8.1. *Sea $\{E_i\}_{i\in I}$ una familia de subconjuntos de un conjunto X, entonces se cumple lo siguiente:*

i)

$$\bigcup_{i\in I}(X \setminus E_i) = X \setminus \bigcap_{i\in I} E_i$$

ii)

$$\bigcap_{i\in I}(X \setminus E_i) = X \setminus \bigcup_{i\in I} E_i$$

Definición 8.2. *. Dados dos objetos a y b, se llama par ordenado al conjunto $\{\{a\}, \{a, b\}\}$ y se le denota como (a, b).*

Teorema 8.1. *Dos pares ordenados son iguales si y sólo si tienen sus respectivas coordenadas iguales, i.e. :*

$$(a, b) = (a', b') \Leftrightarrow a = a' \text{ y } b = b'$$

Definición 8.3. *Dados dos conjuntos A y B, se llama* ***producto cartesiano*** *al conjunto dado por:*

$$A \times B = \{(a, b) : a \in A \, , \, b \in B\}.$$

Si $B = A$ se suele designar con el símbolo A^2 al producto cartesiano $A \times A$.

Definición 8.4. *Dado un conjunto G, se dice que éste es una gráfica si sus elementos son pares ordenados. Si G es una gráfica y $(x, y) \in G$, se dice que y es el correspondiente de x por G.*

De la anterior definición se reconoce que el conjunto de las primeras coordenadas de G (también llamado *primera proyección*, $pr_1 G$) es el *conjunto de definición* de G y que el conjunto de las segundas coordenadas es el *conjunto de valores de G.*

8.2.2. Relaciones y Correspondencias

Definición 8.5. *Se llama correspondencia o relación entre un conjunto A y un conjunto B, a una terna ordenada dada por: $\Gamma = (G, A, B)$, donde G es una gráfica tal que $pr_1 G \subset A$ y $pr_2 G \subset B$. Se dice que G es la gráfica de Γ.*

A $pr_1 G$ se le suele llamar *dominio de* Γ y a $pr_2 G$ el *conjunto de valores de* Γ o contradominio.

Ejemplo 8.1. *Sean A y B dos conjuntos, entonces $\Gamma = ((A \times B, A, B)$ es una relación entre el conjunto A y el conjunto B.*

Ejemplo 8.2. *Sean $A = \{0, 1\}$ y $B = \{0, 1, 2, 3, \}$. Si G es la gráfica $G = \{(0, 0), (0, 1), (0, 2), (0, 3)\}$, entonces $\Gamma = (G, A, B)$ es una relación entre A y B tal que hace corresponder al elemento 0 de A, los elementos 0, 1 y 2 de B; al mismo tiempo relaciona el elemento 1 de A con el elemento 3 de B.*

Definición 8.6. *Sea A un conjunto. Se llama relación en A a toda correspondencia donde el dominio y el conjunto de valores son subconjuntos de A.*

Si R es una relación en A y si y es correspondiente de x por medio de R, se dice que *x e y están R relacionados*, a lo que se denota como: xRy.

Ejemplo 8.3. *Sea B el conjunto de polígonos de n lados, y G la gráfica dada por:*

$$G = \{(\alpha, \beta) | \alpha, \beta \in B, \alpha \text{ semejante } a\, \beta\}$$

La correspondencia $R = (G, \mathbb{N}, \mathbb{N})$ es una relación en B. Un polígono de n lados está relacionado con todos sus poígonos semejantes.

Definición 8.7. *Sean X y Y dos conjuntos, una función de X a Y es un subconjunto del producto cartesiano $X \times Y$ tal que $(a, b)\,(a, c) \in f$, implica $b = c$.*

Dada una función $f : X \longrightarrow Y$, f induce dos funciones, una tiene por dominio a $\wp(X)$ y contradominio a $\wp(Y)$, definida como sigue:

$$\text{dado } A \subset X,\ f(A) := \{y \in Y :\ y = f(x) \text{para algún } x \in A\}$$

Por otro lado, se tiene una segunda función denotada por f^{-1}, la cual tiene por dominio a $\wp(Y)$ y contradominio $\wp(X)$, definida así:

$$\text{dado } B \subset Y,\ f^{-1}(B) := \{x \in X :\ f(x) \in B\}$$

Al conjunto $f(A)$ se le llama imagen directa de A bajo f y al conjunto $f^{-1}(B)$ se le llama imagen inversa de B bajo f.

Definición 8.8. *Sea $f : X \longrightarrow Y$ una función.*
i) f se dice suprayectiva si $f(X) = Y$
ii)f se dice inyectiva sii $x \neq y \Longrightarrow f(x) \neq f(y)$.
iii)f es biyectiva si es inyectiva y suprayectiva.

Si f es inyectiva entonces existe una única función llamada la inversa de f y denotada por $f^{-1} : f(X) \longrightarrow X$ dada por, $f^{-1}(y) = x$ si $y = f(x)$.

8.2.3. Producto cartesiano y funciones

Definición 8.9. *Dados dos objetos a y b, se llama par ordenado de a y b, al conjunto $\{\{a\}, \{a, b\}\}$ y se le denota como (a, b).*

Teorema 8.2. *Dos pares ordenados son iguales si y sólo si tienen sus respectivas coordenadas iguales, i.e. :*

$$(a, b) = (a', b') \Leftrightarrow a = a' \text{ y } b = b'$$

Definición 8.10. *Dados dos conjuntos A y B, se llama* ***producto cartesiano*** *al conjunto dado por:*

$$A \times B = \{(a, b) : a \in A\,,\, b \in B\}.$$

Si $B = A$ se suele designar con el símbolo A^2 al producto cartesiano $A \times A$.

Definición 8.11. *Dado un conjunto G, se dice que éste es una gráfica si sus elementos son pares ordenados. Si G es una gráfica y $(x, y) \in G$, se dice que y es el correspondiente de x por G.*

De la anterior definición se reconoce que el conjunto de las primeras coordenadas de G (también llamado *primera proyección*, $pr_1 G$) es el *conjunto de definición* de G y que el conjunto de las segundas coordenadas es el *conjunto de valores de G.*

Definición 8.12. *Sean X y Y dos conjuntos, una función $\boldsymbol{f}$ de X a Y es un subconjunto del producto cartesiano $X \times Y$ tal que $(a, b), (a, c) \in f$, implica $b = c$.*

Dada una función $f : X \longrightarrow Y$, f induce dos funciones, una tiene por dominio a $\wp(X)$ y contradominio a $\wp(Y)$, definida como sigue:

$$\begin{array}{rccc} \phi : & \wp(X) & \longrightarrow & \wp(Y) \\ & A & \longrightarrow & f(A) \end{array}$$

$$\text{dado } A \subset X,\ f(A) := \{y \in Y :\ y = f(x) \text{ para algún } x \in A\}$$

Por otro lado,se tiene una segunda función denotada por f^{-1}, la cual tiene por dominio a $\wp(Y)$ y contradominio $\wp(X)$, definida así:

$$\begin{array}{rccc} \phi: & \wp(Y) & \longrightarrow & \wp(X) \\ & B & \longrightarrow & f^{-1}(B) \end{array}$$

$$\text{dado } B \subset Y,\ f^{-1}(B) := \{x \in X :\ f(x) \in B\}$$

Al conjunto $f(A)$ se le llama imagen directa de A bajo f y al conjunto $f^{-1}(B)$ se le llama imagen inversa de B bajo f.

Definición 8.13. *Sea $f : X \longrightarrow Y$ una función.*
i) f se dice suprayectiva si $f(X) = Y$
ii)f se dice inyectiva sii $x \neq y \implies f(x) \neq f(y)$.
iii)f es biyectiva si es inyectiva y suprayectiva.

Si f es inyectiva entonces existe una única función llamada la inversa de f y denotada por $f^{-1} :\ f(X) \longrightarrow X$ dada por, $f^{-1}(y) = x$ si $y = f(x)$.

Definición 8.14. *Se dice que dos conjuntos A y B tienen la misma cardinalidad si existe una función biyectiva entre ellos, a lo que se denota como $A \sim B$. Se dice que A es finito si tiene la misma cardinalidad que $\{1, 2, \ldots, n\}$ para algún $n \in \mathbb{N}$, de otro modo se dice que A es infinito. Dado $S \subset \mathbb{N}$, se dice que A es numerable si tiene la misma cardinalidad que S, de otro modo se dice que A es no numerable.*

Definición 8.15. *. Una relación binaria R en un conjunto X es un conjunto cualquiera tal que $R \subset X \times X$. Generalizando lo anterior, dado $\underbrace{X \times X \ldots \times X}_{n \text{ veces}}$ se tendría una relación n-aria. Del mismo modo podemos considerar:*

$$\prod_{i=1}^{n} X_i$$

donde $X_i \neq X_j$ para al menos un $i \neq j$.

Nótese que una relación binaria R es una función sii dados (x, y), $(x, z) \in R \implies y = z$.

Definición 8.16. *Sea A un conjunto, R una relación binaria en A. Se dice que R es una relación de equivalencia si satisface las siguientes condiciones:*

i) $\forall a \in A$, $(a,a) \in R$, *propiedad reflexiva.*
ii) Si $(a,b) \in R$, *entonces* $(b,a) \in R$, *propiedad de simetría.*
iii) Si $(a,b) \in R$ *y* $(b,c) \in R$, *entonces* $(a,c) \in R$, *propiedad transitiva.*

Definición 8.17. *Una relación binaria* R *en un conjunto* A *se llama de orden parcial si:*

i) $\forall a \in A$, aRa, *propiedad reflexiva.*
ii) Si aRb *y* $bRc \Longrightarrow aRc$, *propiedad transitiva.*
iii) Si aRb *y*$bRa \Longrightarrow a = b$, *propiedad de antisimetría.*

Cuando en un conjunto cualquiera X se define un orden parcial, se dice entonces que X está parcialmente ordenado, a lo que se denota $(X, \preceq)$. Frecuentemente a la propiedad anterior se le suele llamar simplemente *relación de orden* dependiendo de la literatura.

Definición 8.18. *Sea* A *un conjunto. Una operación binaria sobre* A *es una función* $f : A \times A \longrightarrow A$. *A dicha función se le puede llamar suma o multiplicación según sea el contexto. Para el primer caso:* $f(x,y) = xy$; *mientras que para el segundo:* $f(x,y) = x + y$.

8.3. A3. Tópicos de sistemas dinámicos no lineales

Un sistema dinámico es en términos prácticos, una forma matemática de describir el comportamiento a través del tiempo, de todos los puntos de un espacio $\mathcal{S}$. El sistema dinámico en $\mathcal{S}$ nos va a especficar la ubicación para cada punto $x \in \mathcal{S}$ en la unidad de tiempo 1, una unidad de tiempo después (en la unidad de tiempo 2), y así sucesivamente. De este modo se obtiene una sucesión de puntos $\{x_1, x_2, \ldots, x_n\}$, donde cada x_i corresponde al valor del punto x en el instante de tiempo i; esta sucesión se conoce con el nombre de *órbita* o *trayectoria*. Surge de este modo, una forma natural de describir mediante una función, la idea anterior:

$$\begin{array}{rcl} \phi : \mathbf{R} \times \mathcal{S} & \longrightarrow & \mathcal{S} \\ (t,x) & \longrightarrow & x_t \end{array}$$

Esto quiere decir que ϕ es una función que toma un vector cuyas coordenadas son el instante t y el punto x en los que nos queremos fijar y le asigna el valor x_t que es la forma en que el sistema dinámico interactua en tales circunstancias. Formalmente se tiene la siguiente:

Definición 8.19. *Sea* $\phi : \mathbb{R} \times \mathcal{S} \longrightarrow \mathcal{S}$ *una función continuamente diferenciable denotada como* $\phi(x,t) = \phi_t(x)$ *, donde* $\mathcal{S}$ *es un conjunto abierto de un espacio*

Euclideano. Entonces se dice que ϕ genera un sistema dinámico si se cumple que (Bhatia Szego 1970):

- $\phi_0 : \mathcal{S} \longrightarrow \mathcal{S}$ *es la función identidad.*
- $\phi_t \circ \phi_s = \phi_{t+s}$

La definición anterior únicamente esta postulando dos características importantes para que la función ϕ genere un sistema dinámico: primero, que la función evaluada en el tiempo cero (i.e. el comienzo de la órbita) sea el mismo punto (esto significa que en el instante cero el punto x no se mueve) y segundo, que la función evaluada sucesivamente por composición para distintos instantes de tiempo es igual a la función evaluada en la suma de estos instantes; i.e. si se tienen $t, s \in \mathbb{R}$ dos instantes de tiempo tales que $\phi_t(x) = x_t$ y $\phi_s(x) = x_s$, entonces $\phi_t(\phi_s(x)) = \phi_{t+s} = x_{t+s}$. Las condiciones de continuidad, diferenciabilidad y conjunto abierto, implican de manera my general que la función que genera el sistema dinámico sea *bien comportada* y se pueda derivar dentro de un intervalo específico para el cual se toman los valores con los que se va a evaluar dicha función.

Un sistema dinámico puede ser descrito por un sistema de ecuaciones diferenciales (para el caso en que el tiempo es continuo) o por ecuaciones en diferencias (para el caso en que el tiempo es discreto). Si $x \in \mathbb{R}^n$ es un vector n-dimensional y $f : \mathbb{R}^n \to \mathbb{R}^n$ es una función vectorial continua y diferenciable, entonces, un sistema n-dimensional de ecuaciones diferenciales puede ser descrito como:

$$x^{'} = f(x)$$

Puesto que $x \in \mathbb{R}^n$, entonces x es un vector que tiene n componentes, i.e. $x = (x_1, x_2, \ldots, x_n)$ y lo mismo sucede con la función f.

8.4. Ecuaciones en diferencias.

Para propósitos de cálculo computacional de los sistemas dinámicos, comúnmente se recurre a la transformación del sistema expresado originalmente mediante ecuaciones diferenciales por ecuaciones en diferencias[1] . Las ecuaciones en diferencias expresan una característica muy importante: iteratividad discreta. Esto significa que tomando el tiempo no como continuo sino a pasos, la función se evalúa sobre sí misma un número de veces específico. En este sentido, una ecuación en diferencias expresa la dependencia de la variable con sus valores de retardo en el tiempo y con respecto a otras constantes (Gonze 2011):

[1]Cabe señalar que t<mbién se utilizan métodos de aproximación numérica en varios casos en los que el sistema dinámico descrito por ecuaciones diferenciales, no puede ser resuelto por medios análiticos.

$$f(y_t, \delta y_t, \delta^2 y_t, \delta^3 y_t, \ldots, \delta^n y_t)$$

Considérese el siguiente ejemplo:

$$y_{t+1} = y_t + \epsilon_{t+1}$$

La ecuación anterior es un modelo básico muy recurrente en el área de análisis de series de tiempo y define una serie temporal determinista con un componente estocástico. Esta ecuación plantea que el valor de la variable en el tiempo $t+1$ depende lineal y directamente del valor de esa misma variable un instante de tiempo atrás y de un valor estocástico por determinar en el instante $t+1$. La magnitud de la diferencia entre cada instante de tiempo es denominada *paso* o *retardo* y se denota como δy. Dicho retardo por lo general se considera que tiene un valor igual a 1, pero puede variar dependiendo del contexto.

Existe una forma general de ecuaciones en diferencias que incluye a una gran parte de los casos de estudio práctico relacionados con el tema a tratar en este trabajo, se denomina *ecuación en diferencias lineal de orden n con coeficientes constantes* y se denota de la siguiente manera (Enders 2003):

$$y_t = a_0 + \sum_{i=1}^{n} a_i y_{t-i} + x_i$$

El presente trabajo utiliza como material teórico y práctico, únicamente conceptos y características de sistemas dinámicos expresados mediante ecuaciones en diferencias.

8.5. Órbitas, periodicidad, puntos críticos y bifurcaciones.

Una parte de gran interés para el análisis de los sistemas dinámicos es saber o indagar su comportamiento respecto al paso del tiempo. Se tienen tres formas generales respecto a este tópico: punto fijo, periodicidad y caos. A grandes rasgos, los puntos fijos son aquellos valores para los cuales el sistema dinámico permanece inalterabletras un periodo de tiempo considerable. La periodicidad denota que el sistema repite un comportamiento dado cada intervalo de tiempo específico y finalmente el caos denota impredecibilidad total dentro de un rango específico; estos tres aspectos se presentan de acuerdo a los valores iniciales con los que se evalúe el sistema también conocidos como *condiciones iniciales*. Veamos con un poco más de detalle estos conceptos.

Definición 8.20. *Un punto a es un punto periódico de periodo n si se cumple que $f^n(a) = a$ y $f^j(a) \neq a$ para $0 < j < n$. Si $n = 1$ entonces a es llamado punto fijo (Robinson 200).*

Entonces, el punto a es de periodo n si la función n veces iterada y evaluada en ese punto da como resultado el mismo punto pero distinto valor para cualquier número de iteraciones menor que n. Los puntos fijos son aquellos valores para los cuales la función siempre regresa el mismo valor. A continuación se plantean los conceptos de órbita y estabilidad.

Definición 8.21. *Sea f una función continua, entonces la órbita hacia adelante del punto a es el conjunto $\mathcal{O}^+ = \{f^k(a) : k \leq 0\}$ (Robinson 2000).*

Se ve claramente que la *órbita hacia adelante* es el conjunto que recolecta los valores que va arrojando la función conforme se va iterando en tiempos positivos para un punto en específico. Existe una analogía para tiempos negativos la cual es llamada *órbita hacia atrás* pero requiere una condición especial de invertibilidad en la función f; en este trabajo nos bastará la *órbita hacia adelante* por lo que a partir de este momento se hará referencia únicamente como *órbita del sistema*.

Con el propósito de describir y comprender mejor el concepto de órbita para funciones de una dimensión, existe una herramienta gráfica muy útil; es el llamado método de graficación Cobweb. Se elige un punto inicial x_0 y se traza un segmento de línea vertical de (x_0, x_0) al punto $(x_0, f(x_0))$. Posteriormente se traza un segmento de línea horizontal del punto anterior al punto $(f(x_0), f(x_0))$ sobre la diagonal. El proceso anterior representa gráficamente el modo en que se mapea el punto x_0 al punto $x_1 = f(x_0)$, siendo éste último precisamente la primera iteración de la función evaluada en el punto inicial. Repitiendo el proceso anterior se van obteniendo de manera sucesiva segmentos de línea para los puntos $\{(x_1, x_1), (x_1, f(x_1)), (f(x_1), f(x_1)), \ldots, (x_n, x_n), (x_n, f(x_n)), (f(x_n), f(x_n)), \ldots\}$ (Alligood 2000). Nótese que $f(x_i) = x_{i-1}$, por lo que el método Cobweb gráfica visualmente el comportamiento dinámico de la órbita de la función para un cierto punto inicial x_0. En la figura 3.1 se muestra un ejemplo del procedimiento anterior.

Se dice que un punto fijo es *estable* si puntos cercanos a él son movidos aún más cerca conforme transcurre el tiempo dentro del sistema dinámico. De forma inversa, el punto fijo es *inestable* si los puntos cercanos a él se alejan a través del tiempo. En este sentido, los puntos fijos estables corresponden a *estados estacionarios* del sistema, esto es, lapsos de tiempo bajo condiciones iniciales específicas para los cuales el sistema dinámico no cambia o sufre cambios leves. Por otro lado, para un punto fijo inestable, pequeñas variaciones o cambios en el estado generarán perturbaciones considerables en el sistema. En este sentido un *estado estacionario* o un *equilibrio* x_{ss}, se obtiene cuando:

$$x_{n+1} = x_n = x_{ss}$$

Definición 8.22. *Sea $f : \mathbb{R} \to \mathbb{R}$ y $p \in \mathbb{R}$. Si todos los puntos suficientemente cercanos a p son atriados a p, entonces este punto es llamado* ***sumidero*** *o* ***punto***

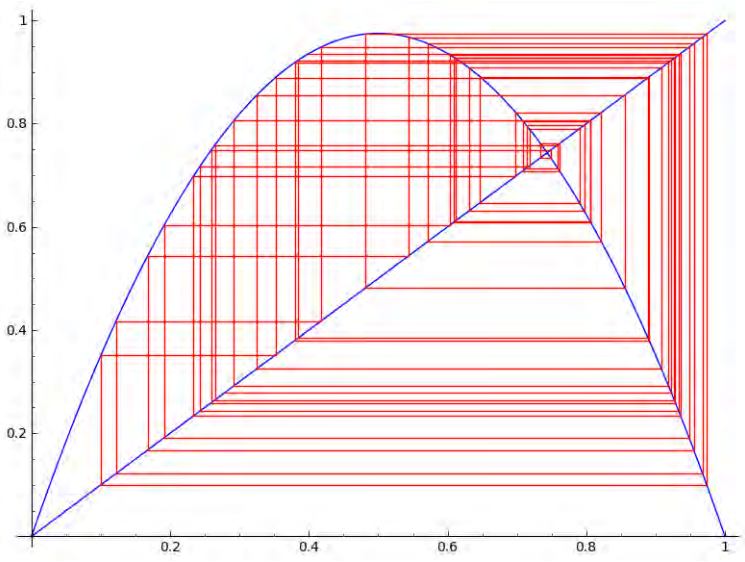


Figura 8.1: e Gráfica Cobweb de la Ecuación Logística

fijo atractor. *Si todos los puntos suficientemente cercanos a p son repelidos de éste, entonces se denomina* ***fuente*** *(Robinson 2000).*

Un atractor es básicamente un conjunto de puntos a los cuales evoluciona el sistema después de un periodo de tiempo si las condiciones iniciales son acordes. Para que el conjunto sea un atractor, las trayectorias que le sean suficientemente próximas han de permanecer próximas incluso si son ligeramente perturbadas. Geométricamente, un atractor puede ser un punto, una curva, una variedad o incluso un conjunto complicado de estructura fractal conocido como atractor extraño. En este sentido, el atractor define cierto grado de estabilidad del sistema permitiendo caracterizar su comportamiento para dichas referencias. Cuando todos los puntos en una vecindad de una trayectoria convergen a la misma órbita, el atractor es un punto fijo o un ciclo límite.

El *diagrama de bifurcación* es una herramienta extremadamente útil a la hora de analizar sistemas dinámicos no lineales puesto que refleja el comportamiento del sistema en relación a las iteraciones y el parámetro de control definido. Es un diagrama en el cual la información gráfica muestra como el sistema depende de cierto parámetro fijo llamado *parámetro de control*. Entonces, una bifurcación ocurre siempre que la dinámica del sistema, como una función de uno o más parámetros, cambia cuantitativamente (Broer, Takens, 2009). El diagrama de bifurcación puede ser utilizado por el compositor para definir las condiciones iniciales con las que debe evaluar al sistema para obtener órbitas estables, periódicas o caóticas.

8.6. Exponentes de Lyapunov

.

Una de las características más sobresalientes para que un sistema se considere caótico es la llamada *sensibilidad a las condiciones iniciales*. Esto significa básicamente que cuando se varían de forma mínima las condiciones iniciales, el sistema presenta un comportamiento totalmente distinto. Más precisamente; si dos trayectorias u órbitas distintas que comienzan suficientemente cerca, se desvían y se alejan más y más al paso del tiempo, entonces es una prueba de que el sistema presenta comportamiento caótico bajo esas condiciones iniciales. La tasa a la cual las trayectorias se desvían una de otra es caracterizada por un parámetro llamado *exponente de Lyapunov* (Cvitanovic 2004).

Si el sistema presenta caos, se puede afirmar por consiguiente que existe un atractor extraño y entonces, dadas $x(t) = f^t(x_0)$ y $x(t)+\delta x(t) = f^t(x_0+\delta x_0)$ dos trayectorias que comienzan muy cercanas,[2] éstas comenzarán a separarse entre sí de manera exponencial y dentro de un timpo finito esta separación será del tamaño de todo el espacio de estado. Dicha sensibilidad a las condiciones iniciales puede ser caracterizada por la siguiente expresión:

$$\parallel \delta x(t) \parallel \approx e^{\lambda t} \parallel \delta x_0 \parallel$$

donde λ es precisamente la tasa de separación de las trayectorias del sistema o *exponeente de Liapunov* (Cvitanović 2004). Su representación general está dada por:

$$\lambda = \lim_{t\to\infty} \frac{1}{t} \sum_i t_i \lambda_i$$

donde $t_i\lambda_i = ln(\frac{\|\delta x(t_i)\|}{\|\delta x_0\|}$ y $t = \sum_i t_i$ (Cvitanović 2004)

Conceptualmente los *exponenetes de Liapunov* poseen un fuerte significado sobre la presencia o no de caos dentro de un sistema dinámico, por lo que pueden resultar ser útiles al momento de generar material conceptual, práctico y estético al momento de la composición. Existe de hecho un diagrama que permite visualizar el comportamiento de los exponentes de Lyapunov para cada sistema en particular y ciertos rangos de iteraciones; se mostrarán para algunos de los ejemplos siguientes.

8.7. La familia de mapeos logísticos.

La ecuación Verhulst fue publicada por primera vez por Pierre Francois Verhulst en 1838 después de haber leído el *Ensayo sobre el principio de población* de Thomas

[2] $\delta x(t)$ es la diferencia o distancia entre ambas trayectorias. En el contexto de las matemáticas, por lo general valores representados por δ y/o ϵ significan incrementos o diferencias mínimas.

Malthus, (Malthus 1798).

Verhulst derivó su ecuación logística para describir el crecimiento auto-limitado de una población biológica. En ocasiones, la ecuación es también llamada ëcuación Verhulst-Pearlp̈or su redescubrimiento en 1920. Alfred J. Lotka obtuvo de nuevo la ecuación en 1925, llamándola *ley del crecimiento poblacional* (Gonze 2013).

Desde una panoramica muy general, la función logística o ecuacion logística modela la función sigmoidea de crecimiento de un conjunto P. El estadio inicial de crecimiento es aproximadamente exponencial; al cabo de un tiempo, aparece la competición entre algunos miembros de P por algún recurso crítico K (cuello de botella) y la tasa de crecimiento disminuye; finalmente, en la madurez, el crecimiento se detiene. En realidad la ecuación logística es una *familia de ecuaciones* asociada al modelo logístico y que está definida por la siguiente expresión:

$$g_a(x) = ax(1-x)$$

Para cada valor distinto de a se obtiene una ecuación distinta[3]. Haciendo un desarrollo analítico matemático se obtiene que cuando $0 \leq a < 1$, el mapeo logísitico tiene un sumidero en $x = 0$; de hecho, una característica importante de esta familia de ecuaciones es que cualquier condición inicial entre 0 y 1 es atraída a este mismo sumidero.

La versión discreta en ecuaciones en diferencias de esta función está dada por la expresión:

$$x_{n+1} = f(x_n) = rx_n(1-x_n)$$

El estado estacionario x_{ss} para esta ecuación se obtiene de acuerdo a (Gonze 2013):

$$x_{ss} = rX_{ss}(1-x_{ss}) \Rightarrow rx_{ss}^2 - x_{ss}(r-1) = 0$$

$$\therefore$$

$$x_{ss} = 0 \quad \& \quad x'_{ss} = 1 - \frac{1}{r}.$$

Dado que el presente trabajo tiene como objetivo ser una herramienta composicional y no un instrumento de tortura, en este punto se abandonarán los desarrollos matemáticos excesivos y se optará por presentar formas prácticas que el lector pueda aplicar en su método creativo. A continuación se muestra el diagrama de bifurcación para la ecuación logística:

[3]De ahí la razón que se le denomine Familia de Ecuaciones Logísticas o Mapeos Logísticos.

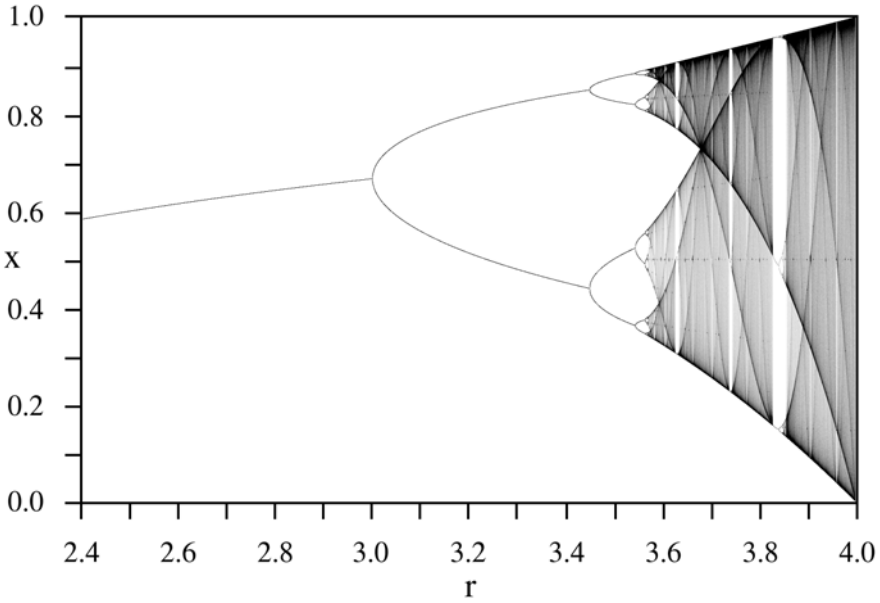


Figura 8.2: Diagrama de bifurcación, ecuación logística

De la figura 3.2 puede concluirse que para $r \leq 1$ la población se extingue, para $1 < r < 3$ la población tiene a un valor estable monotamente creciente proporcional a r (Broer, Takens, 2009). Para $3 < r < 4$ el sistema comienza a presentar comportamiento caótico apareciendo un mayor número de bifurcaciones.

Ahora bien este diagrama es precisamente una de las herramientas más útiles y accesibles para el compositor que desee utilizar sistemas dinámicos como parte de su material creativo. Analizando el diagrama, el compositor puede elegir inicializar el sistema con distintos valores específicos de acuerdo al tipo de comportamiento que desee obtener en la serie de datos. Por ejemplo, si desea obtener una serie con comportamiento estable deberá dar condiciones iniciales tales que $1 < r < 3$. En la figura 3.3 se observa la órbita obtenida con $x = 0.5$, un valor de $r = 2.9$ y 100 iteraciones.

Si por el contrario desea valores con comportamiento caótico puede recurrir a una inicialización del sistema con valores de $x = 0.3$, $r = 3.9$ y 1000 iteraciones. En la figura 3.4 se muestran segmentos de la serie de datos obtenida, para rangos entre 600-700,700-800,800-900 y 900-1000 iteraciones.

La figura 3.5 muestra un diagrama que gráfica los exponentes de Lyapunov respecto al parámetro de control r para la ecuación logística. La gráfica 3.6 muestra la comparación entre el diagrama de bifurcación, los exponentes de Lyapunov y las periodicidades de la ecuación logística (Ramírez Ávila, Gallas 2011):

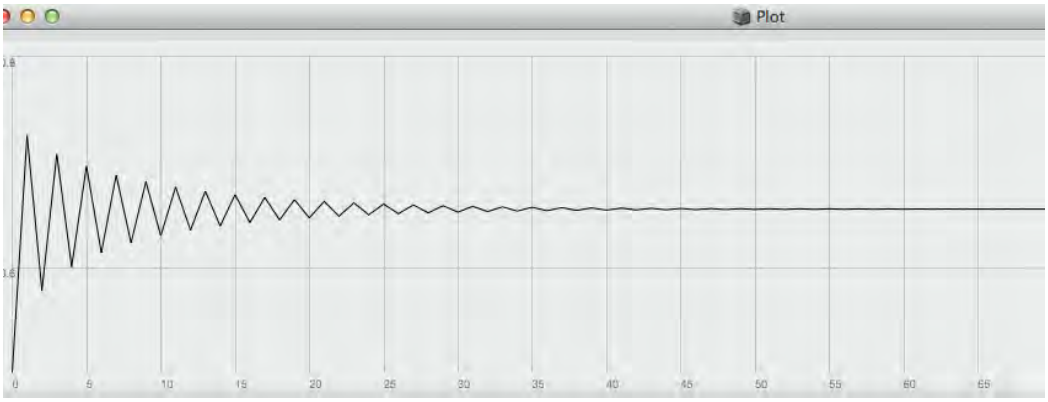


Figura 8.3: Órbita estable para el mapeo logístico

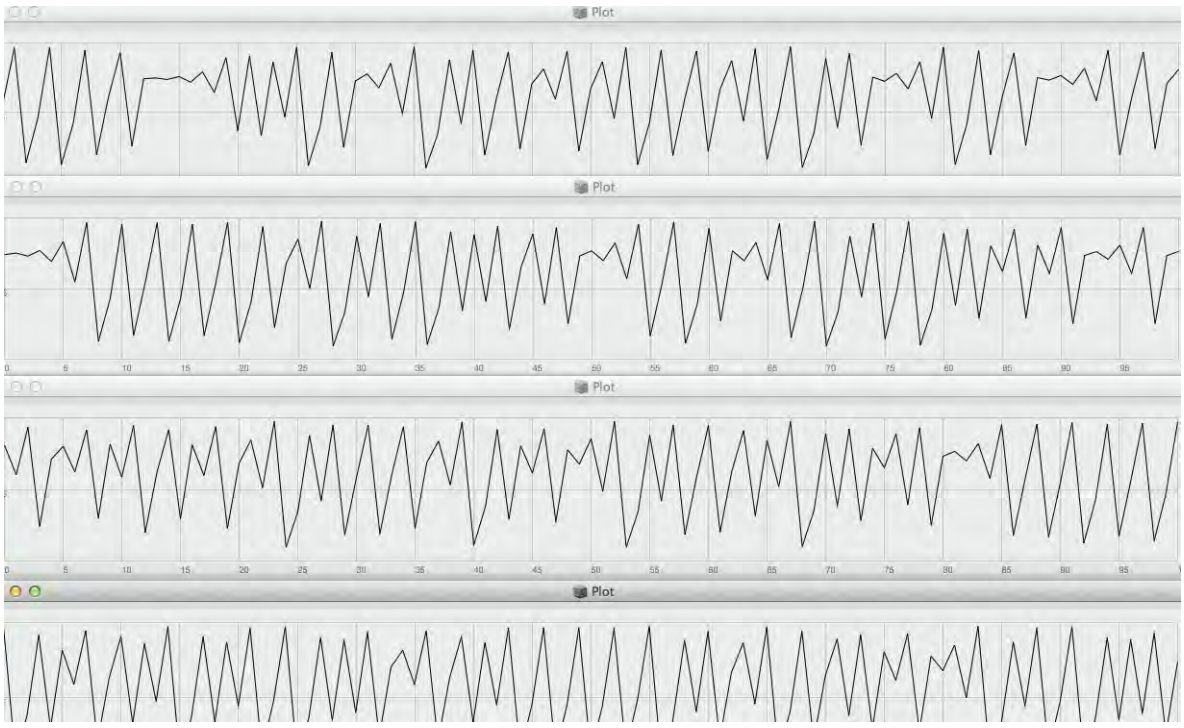


Figura 8.4: Órbita caótica para el mapeo logístico en segmentos de 100 iteraciones, desde la ietración 600 hasta la iteración 1000.

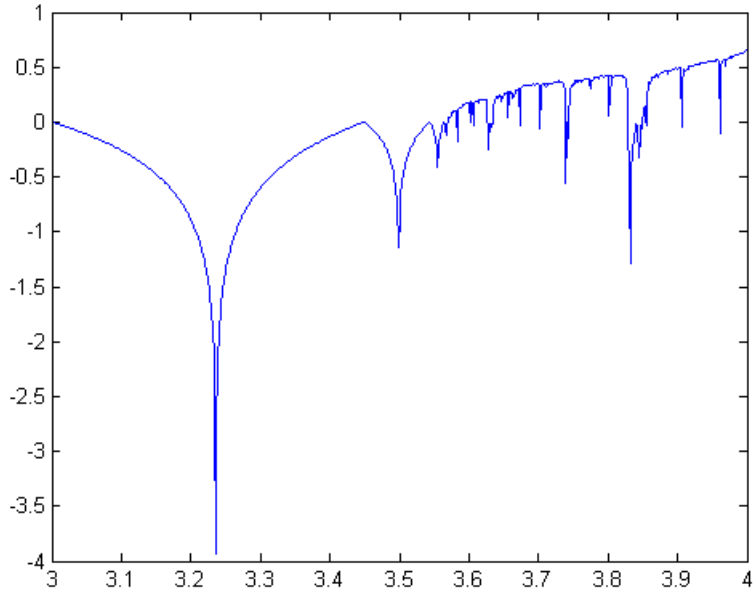


Figura 8.5: Gráfica parámetro de control r vs exponentes Lyapunov

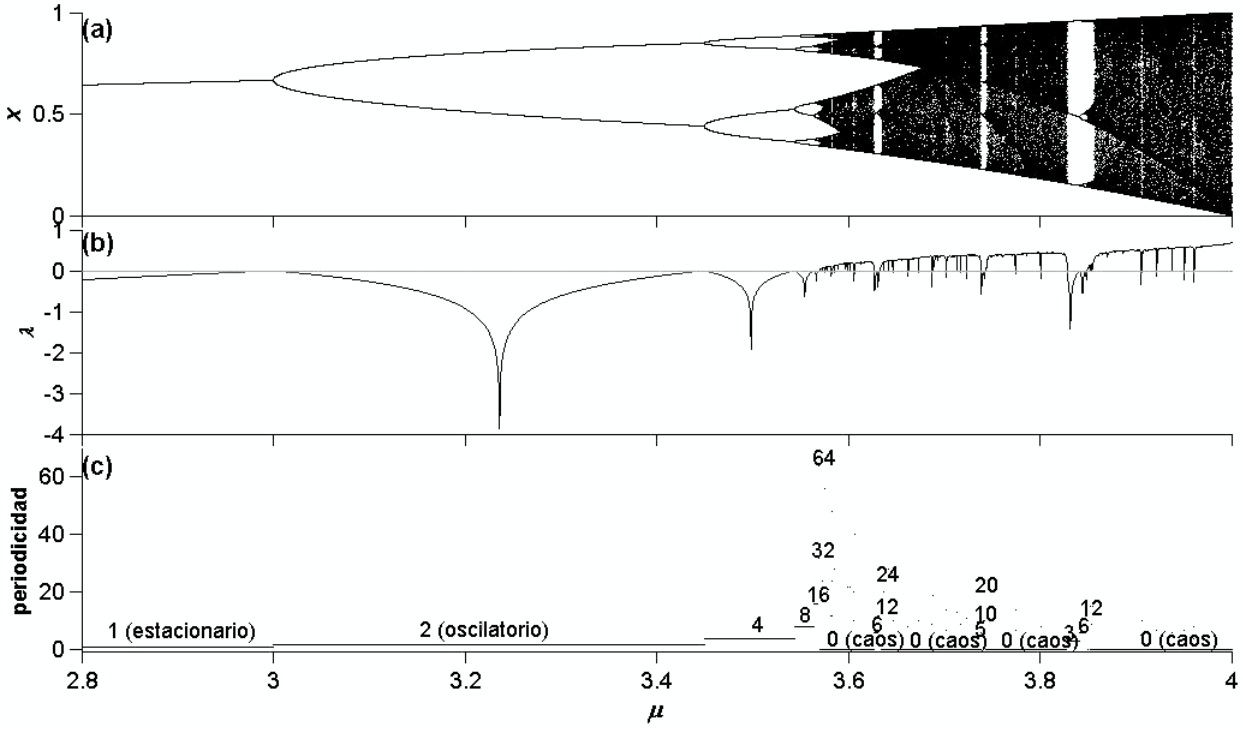


Figura 8.6: Gráfica de bifurcación, Lyapunov y periodicidad.

La figura 3.7 muestra el algoritmo implementado en Supercollider para modelar el mapeo logístico. En la figura 3.8 se presentan algunas trayectorias obtenidas con dicho algoritmo para distintas condiciones iniciales.

```
96
97 //logistic map
98     logistic {
99
00         arg iteration, initCondition, controlParameter;
01         dataList = List[];
02
03
04         dataList.add(initCondition);
05        iteration.do({arg i;
06        x = controlParameter*dataList[i]*(1-dataList[i]);
07        dataList.add(x);
08         });
09    ^dataList.asArray;
10
11     }
12
```

Figura 8.7: Algoritmo del mapeo logístico en Supercollider.

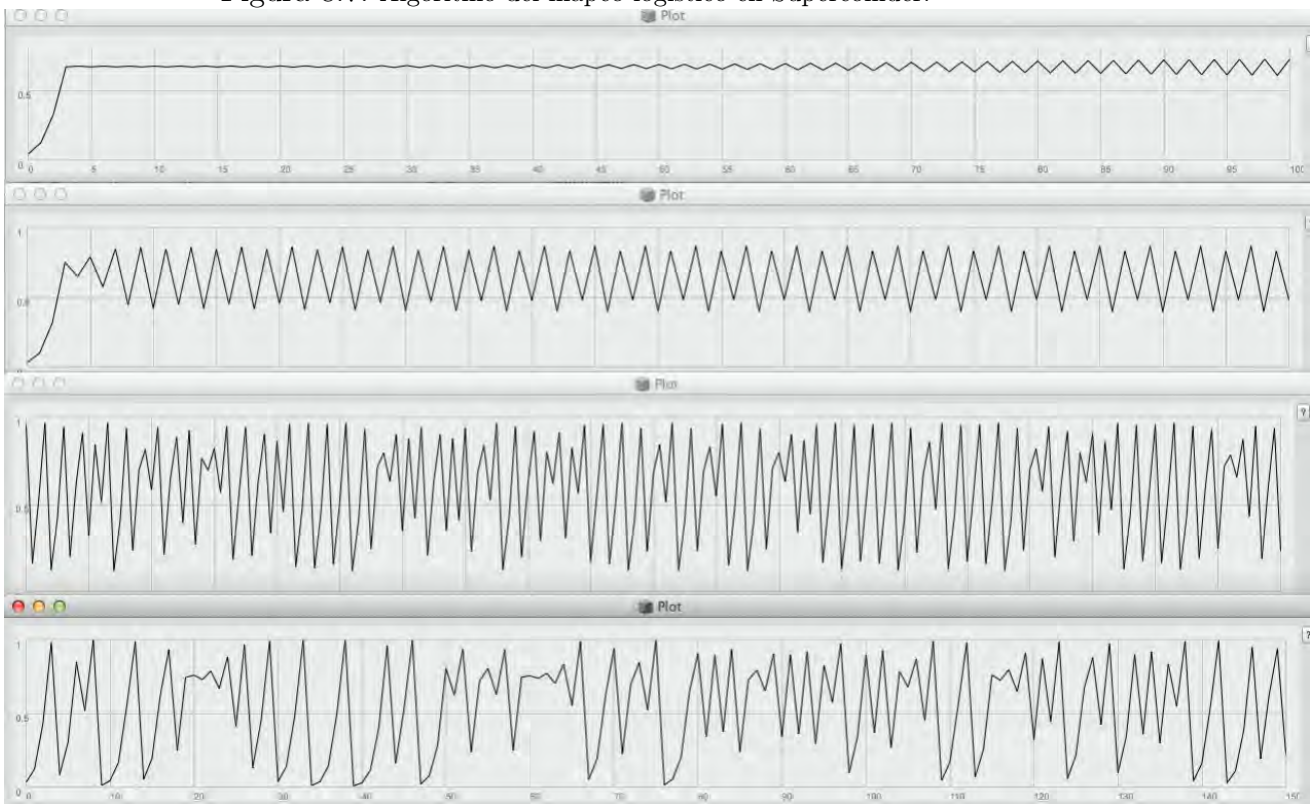


Figura 8.8: Mapeo Logístico con condiciones iniciales: a)$iteraciones = 100$,$x_0 = 0.03$, $r = 3.475$, b) $iteraciones = 100$, $x_0 = 0.0053$, $r = 3.0176$, c) $iteraciones = 1000$ (800-950), $x_0 = 0.4$,$r = 3.891$ y d) $iteraciones = 1000$ (800-950),$x_0 = 0.83$,$r = 3.9965$.

8.8. El mapeo de Henon.

El astrónomo francés Michel Henon publicó en 1976 un trabajo en el que presenta una familia de mapeos bidimensionales, a modo de propuesta de un modelo reduccionista capaz de reproducir de modo computacionalmente mas simple los resultados provenientes del paradigmatico modelo de Lorenz. (Ramírez Ávila, Gallas 2011). El mapeo de Henon en su versión discreta está dado por:

$$H_{\alpha\beta}\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} 1 - \alpha x_n^2 + \beta \\ \beta x_n \end{bmatrix}$$

Haciendo una sustitución directa del segundo término en función del primero se observa que:

$$y_n = x_{n-1}$$

Por lo que la expresión discreta unidimensional del mapeo Henon resulta así:

$$x_{n+1} = 1 - \alpha x_n^2 + \beta x_{n-1}$$

la cual es una ecuación en diferencias con un retraso de segundo orden. α se denomina *coeficiente de no linealidad* y β *coficiente de disipación*. La expresión anterior resulta sencilla de implementar en un algoritmo de programación para poder calcular distintas órbitas. El mapeo de Henon arroja órbitas interesantes para $0.91 < \alpha < 1.41$, $\beta = 0.3035$, $(x_0, y_0) = (-0.1, 0)$. El usuario puede inicializar el sistema con estos parámetros y obtener material útil para posibles mapeos a parámetros sonoros. En la figura 3.9 se muestra el diagrama de bifurcación de este mapeo: [4]

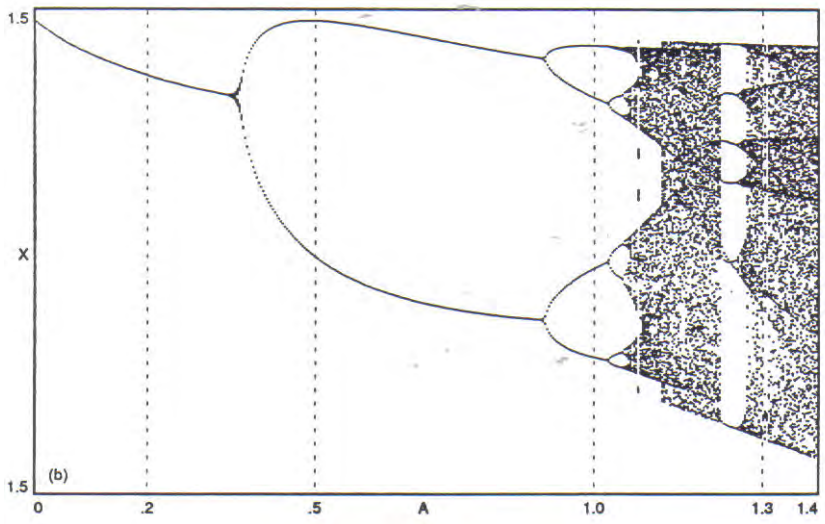


Figura 8.9: Diagrama de bifurcación, mapeo de Henon.

[4]Para un análisis exhaustivo sobre el mapeo de Henon consúltese (Rudakov 2000).

Se recuerda al lector la precaución que debe tener al momento de definir las condiciones iniciales por precisamente el hecho de la sensibilidad ante dichos parámetros que los sistemas caóticos presentan. Condiciones iniciales distintas a las definidas dentro de algún atractor pueden generar órbitas con valores totalmente no acotados.

Las figuras 3.10 y 3.11 muestran respectivamente, el algoritmo implementado en Supercollider para el mapeo de Henon y algunas trayectorias obtenidas para distintas condiciones iniciales.

Chaos Map.sc (~/Library/A

Chaos Map.sc | Untitled

```
212
213
214 //henon map
215   henon {arg a,b,init_x,init_y,it;
216     var x,list;
217         list = List[];
218     list.add(init_x);
219     list.add(init_y);
220
221     it.do({arg i;
222
223         x = 1-(a*(list[i]**2))+(b*list[i+1]);
224         list.add(x);
225     });
226         ^list.asArray;
227 }
```

Figura 8.10: Diagrama de bifurcación, mapeo de Henon.

Figura 8.11: Órbitas del mapeo de Henon para condiciones iniciales: a)$\alpha = 1.1$, $\beta = 0.3035$, $x_0 = -0.1$, $y_0 = 0$, $iteraciones = 150$, b) $\alpha = 0.93741$, $\beta = 0.3035$, $x_0 = -0.1$, $y_0 = 0$, $iteraciones = 150$, c) $\alpha = 1.193741$,$\beta = 0.435$, $x_0 = -0.31$, $y_0 = 0.01$, $iteraciones = 150$.

8.9. El mapeo cúbico.

La familia de mapeos cúbicos está dada por:

$$\begin{array}{ccc} f:\mathbb{R} & \longrightarrow & \mathbb{R} \\ x & \rightarrow & ax^3+(1-a)x \end{array}$$

R.M. May trabajó con este mapeo con el fin de desarrollar soluciones a ciertos problemas en el área de genética, en particular el problema de un locus y dos alelos, estableciendo la existencia de una sucesión de bifurcaciones cíclicas de periodo 2 (R.M. May 1979). Posteriormente David C. Whitley profundiza en el tema y presenta resultados de valores críticos para los que el mapeo presenta comportamiento caótico (Whitley 1981).

De acuerdo a los resultados obtenidos por May, para valores de $1 \leq a \leq 4$, el mapeo queda restringido al intervalo $[-1.0, 1.0]$, además de que el mapeo alcanza valores de 0 para $x = 0, \pm\sqrt{(a-1)/a}$ y puntos fijos en $x = 0, \pm 1$. Para $a > 1$ existen dos puntos críticos, $x = \mp\sqrt{(a-1)/3a}$. A continuación se muestra el diagrama de bifurcación del mapeo cúbico para $3 \leq a \leq 4$.

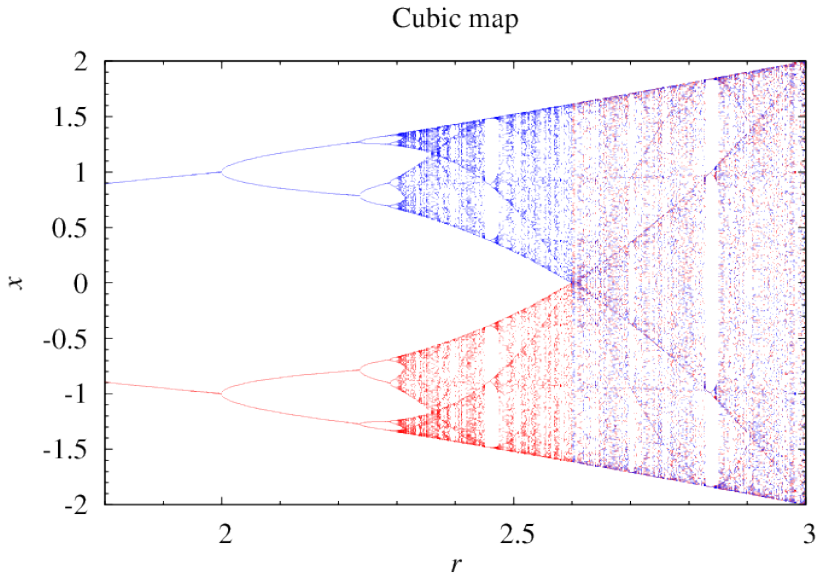


Figura 8.12: Diagrama de bifurcación, mapeo cúbico.

Este mapeo genera órbitas interesantes para valores $3 \leq a \leq 4$ y $-0.3 < x < 0.3$. Valores de $a \leq 2$ generan órbitas periódicas estables, por lo que el material en series de tiempo resulta monótono, sin embargo puede resultar útil para ciertos propósitos composicionales.

La siguiente gráfica muestra el algoritmo implementado en Supercollider junto con algunas trayectorias obtenidas para distintas condiciones iniciales.

```
242
243 //cubic map
244    cubic {
245     arg lambda = 2.59,init = 0.101562,it;
246     var x,data;
247     data = List[];
248     data.add(init);
249
250     it.do({ arg i;
251             x = (lambda*data[i]**(3))+((1-lambda)*data[i]);
252         data.add(x);
253     });
254     ^data.asArray;
255 }
256
```

Figura 8.13: Algoritmo del mapeo cúbico.

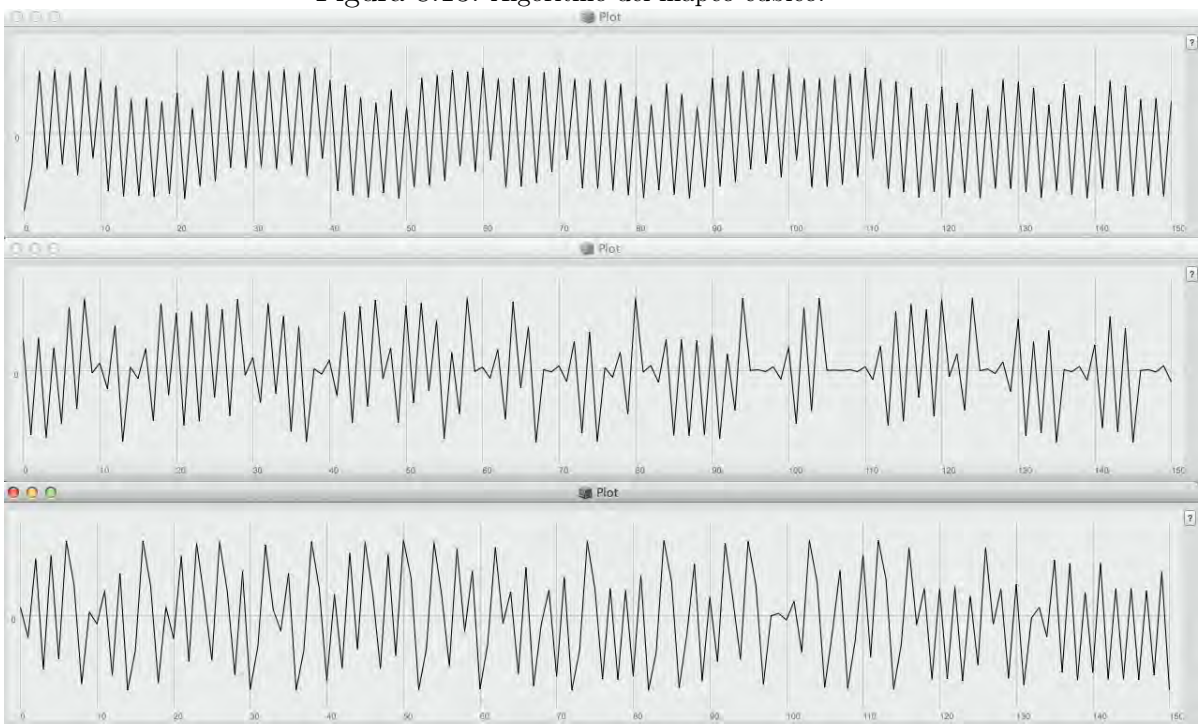


Figura 8.14: Órbitas del mapeo cúbico para condiciones iniciales a) $a = 3.38$, $x_0 = -0.2699$, $iteraciones = 150$ b) $a = 3.634$, $x_0 = 0.0029$, $iteraciones = 150$ c) $a = 3.818$, $x_0 = -0.02599$, $iteraciones = 150$.

8.10. Los mapeos Duffing y Tent

El mapeo Duffing está definido por el siguiente sistema de ecuaciones:

$$\begin{aligned} x_{n+1} &= y_n \\ y_{n+1} &= -bx_n + ay_n - y_n^3 \end{aligned}$$

Este modelo es la versión discreta de la ecuación diferencial original la cual describe la dinámica periódica de una masa puntual dentro de un potencial doble. Yoshisuke Ueda, un ingeniero japonés, trabajó con este modelo y una versión ligeramente modificada del mismo le permitió encontrar características caóticas en él (Kanamaru 2005). Este sistema puede resultar útil como material composicional con las condiciones iniciales: $a = 2.75$, $b = 0.2$, $x_0 = 1.5$, $y_0 = 0.5$. La figura 3.15 muestra el diagrama de bifurcación del mapeo Duffing para los valores de los parámetros a y b mencionados anteriormente:

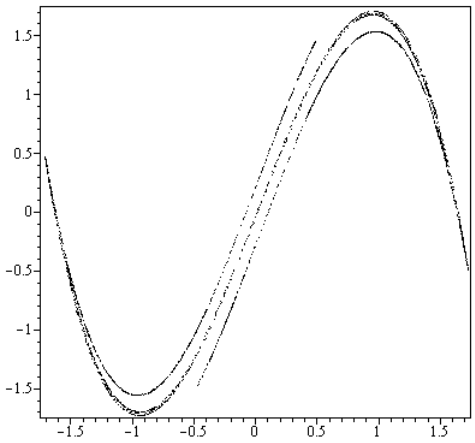


Figura 8.15: Diagrama de bifurcación del mapeo Duffing.

El mapeo tent (tienda) puede ser visto como el estiramiento y replegado de una banda elástica unidimensional de longitud unitaria. Cada vez que el sistema realiza una iteración la banda se alarga el doble de su longitud y se repliega sobre sí misma. Todo punto dentro del intervalo $[0, 1/2]$ duplicarán su magnitud y mapearán en puntos dentro del intervalo $[0, 1]$ (Starret 1997). El mapeo tienda está definido por:

$$x_{n+1} = \begin{cases} 2x_n & x_n \in [0, 1/2] \\ 2(1 - x_n) & x_n \in (1/2, 1] \end{cases}$$

Cuyos puntos fijos están dados por $x = 0, 2/3$. La figura 3.16 muestra el diagrama de bifurcación de dicho sistema:

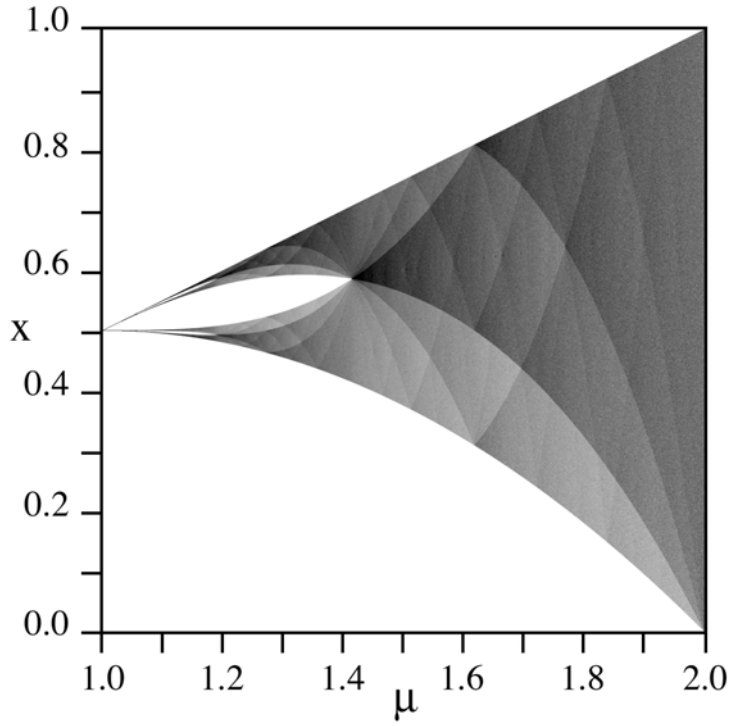


Figura 8.16: Diagrama de bifurcación del mapeo Tent.

En la gráfica 3.15 se muestra el algoritmo implementado en Supercollider para calcular los mapeos Duffing y tent respectivamente. Las gráficas..muestran algunas trayectorias obtenidas para distintas condiciones iniciales de dichos sistemas.

Chaos Map.sc | Untitled

```
}

//tent map
//lambda = 1.97, init = 0.101562, ok
 tent {
    arg lambda = 1.97,init = 0.101562,it;
    var x,data;
    data = List[];
    data.add(init);

    it.do({ arg i;
        if(((data[i]<=0.5)&&(data[i]>=0)),{
            x = lambda*data[i];
                data.add(x)},
            {
                x = lambda*(1-data[i]);
              data.add(x)});
    });
   ^data.asArray;
}

//Duffing Map
  duff {arg a=2.75,b=0.2,x_init=1.5,y_init=0.5,it;
    var x_list, y_list,y;

    x_list = List[];
    y_list = List[];

    x_list.add(x_init);
    y_list.add(y_init);

    it.do({arg i;
        x_list.add(y_list[i]);

        y = ((a*y_list[i])-(b*x_list[i])-(y_list[i]**3));

        y_list.add(y);
    });
   ^y_list.asArray;
}

}
```

Figura 8.17: Algoritmo en Supercollider del mapeo Duffing y del mapeo Tent.

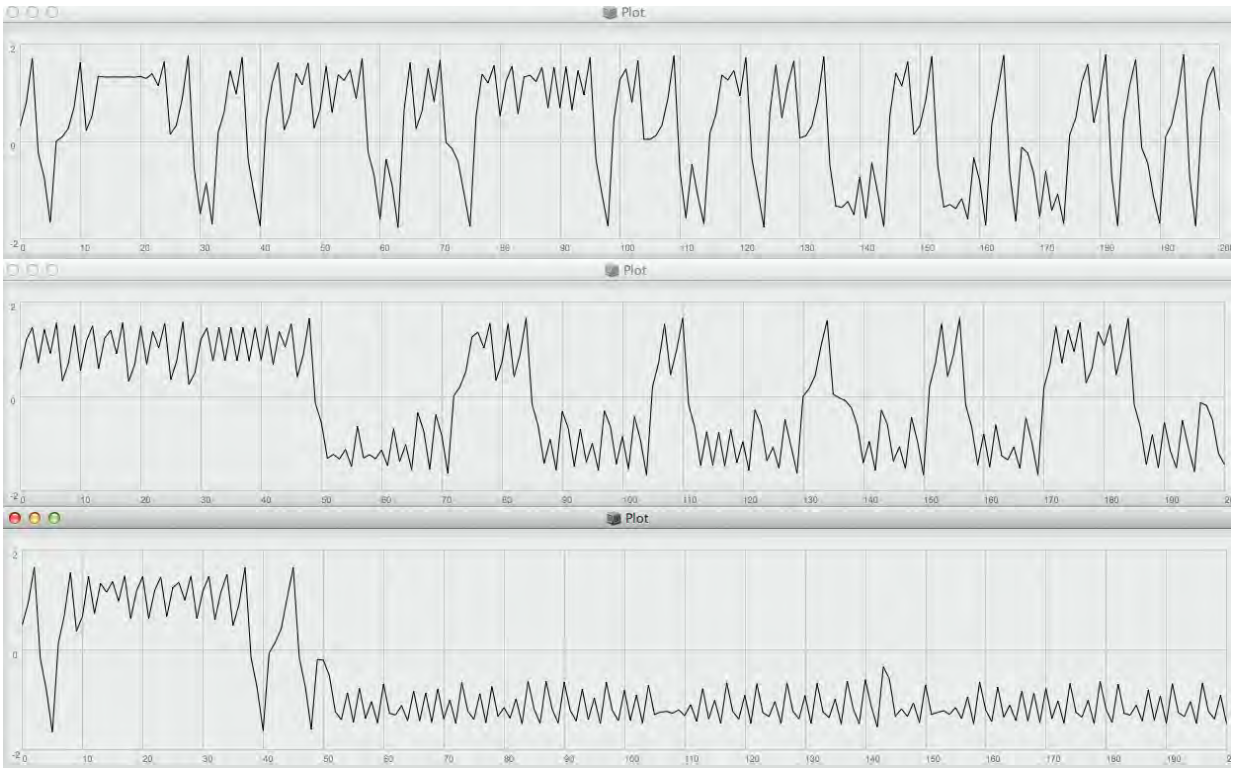


Figura 8.18: Órbitas del mapeo Duffing para condiciones iniciales: a) $a = 2.8$, $b = 0.051$, $x_0 = 1.316$, $y_0 = 0.32$, $iteraciones = 200$. b) $a = 2.7$, $b = 0.1$, $x_0 = 1.56$, $y_0 = 0.57$, $iteraciones = 200$ c) $a = 2.75$, $b = 0.2$, $x_0 = 1.5$, $y_0 = 0.5$, $iteraciones = 200$

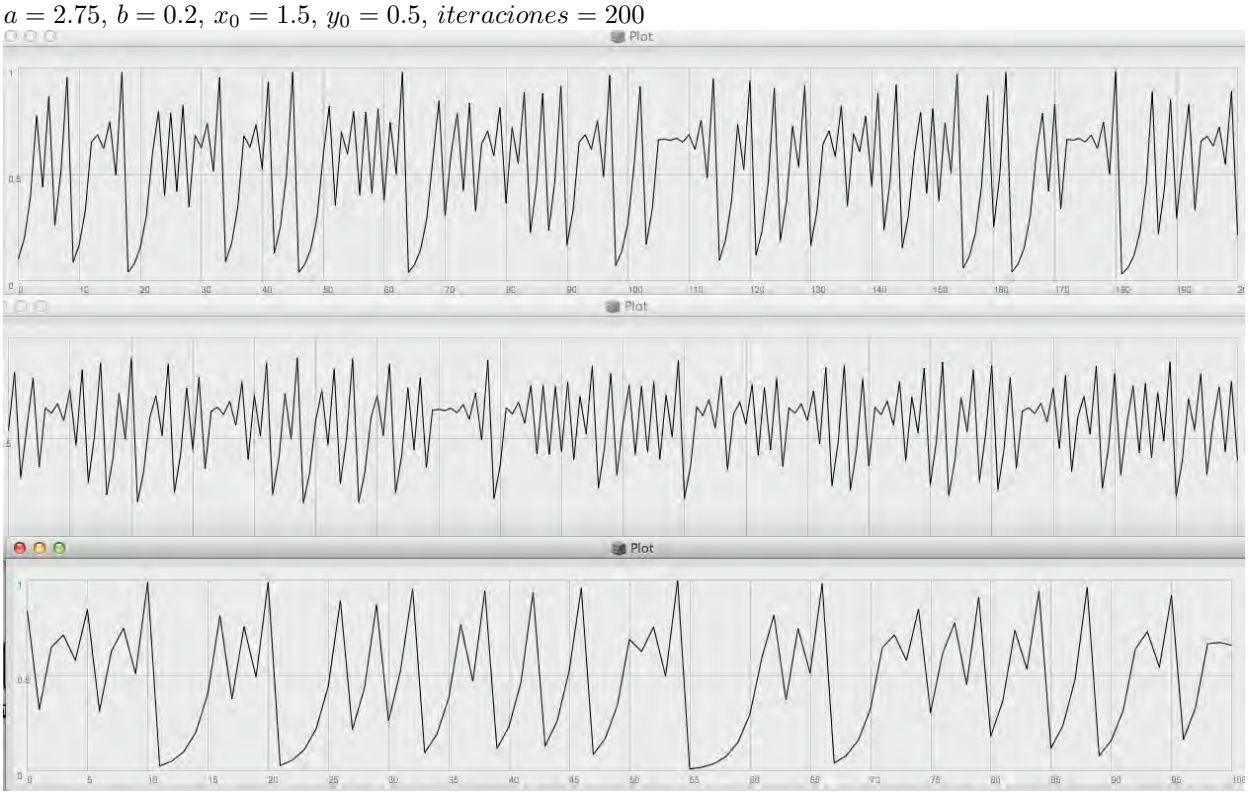


Figura 8.19: Órbitas del mapeo Duffing para condiciones iniciales: a) $a = 2.8$, $b = 0.051$, $x_0 = 1.316$, $y_0 = 0.32$, $iteraciones = 200$. b) $a = 2.7$, $b = 0.1$, $x_0 = 1.56$, $y_0 = 0.57$, $iteraciones = 200$ c) $a = 2.75$, $b = 0.2$, $x_0 = 1.5$, $y_0 = 0.5$, $iteraciones = 200$

8.11. El mapeo seno

Este es uno de los mapeos más sencillos dentro de la literatura de Teoría del Caos, la versión discreta para ecuaciones en diferencias está dada por:

$$x_{n+1} = sen(rx_n)$$

El compositor italiano Agostino Di Scipio hace una exploración profunda de dicho mapeo en el área de síntesis de texturas sonoras llegando a ciertas conclusiones como el hecho de que a su parecer, la sensibilidad a las condiciones iniciales del sistema revela la no integrabilidad de los sistemas caóticos al oído o el hecho de que de acuerdo a su valoración, es imposible fijar valores para obtener un sonido específico, especialmente si el modelo no es inicializado dentro de los parámetros de comportamiento periódico (Di Scipio 1999).

A continuación se muestra el diagrama de bifurcación del sistema, para la quinta iteración y valores del parámetro de control $3 \leq r \leq 4$.

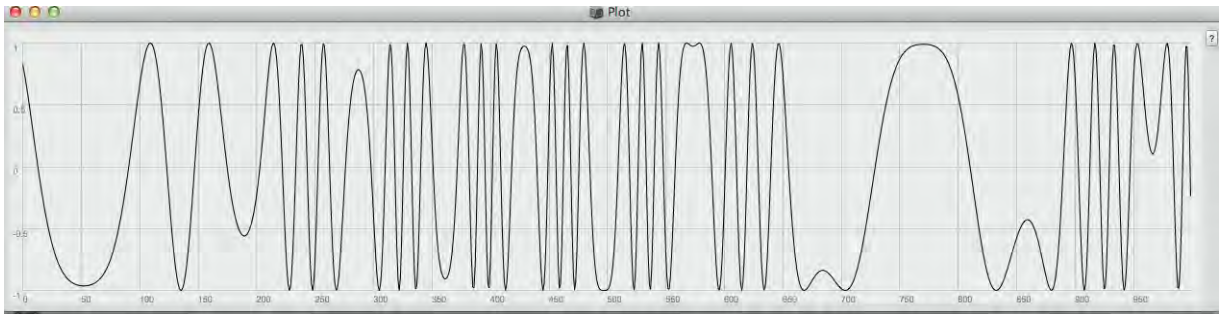


Figura 8.20: 5a iteración del mapeo seno con $3 \leq r \leq 4$

La siguiente gráfica muestra el algoritmo implementado en Supercollider junto con algunas trayectorias obtenidas para disintas condiciones iniciales.

```
229 //sine map
230   sine {
231     arg lambda = 0.99,init = 0.101562,it;
232     var x,data;
233     data = List[];
234     data.add(init);
235
236     it.do({ arg i;
237         x = (sin(lambda*pi*data[i]));
238         data.add(x);
239     });
240     ^data.asArray;
241 }
```

Figura 8.21: Algoritmo en Supercollider del mapeo seno

Figura 8.22: Órbitas del mapeo seno para condiciones iniciales: a) $r = 0.8597$, $x_0 = 0.21$, $iteraciones = 150$ b) $r = 0.8897$, $x_0 = 0.13$, $iteraciones = 150$, c)$r = 0.99$, $x_0 = 0.101562$, $iteraciones = 150$

Bibliografía

[1] Juan Antonio Pérez Ortiz (2000). *Música fractal: EL SONIDO DEL CAOS*, Departamento de Lenguajes y Sistemas Informáticos Universidad de Alicante.

[2] Peter Worth, Susan Stepney. *Growing Music: musical interpretations of L-Systems*, Department of Computer Science, University of York.

[3] Bidlack Rick (2008). Chaotic Systems a simple (but complex) Compositional Algorithms. Department of Music, State University of NewYork.

[4] Di Scipio Agostino (2009). *Composition by exploration of Non-Dynamical Systems*. CSC Uiversity of Padova.

[5] Timothy Tristram Opie (2003). *Creation of a Real-Time Granular Synthesis Instrument for Live Performance.* Creative Industries: Music Queensland University of Technology, Master Thesis.

[6] Eduardo Reck Miranda (2007). *Evolutionary Computer Music*. Springer

[7] Gerhard Nierhaus (2009). *Algorithmic Composition*. Springer Verlag.

[8] Marcus Lauks (2010). *Computer-Aided Algorithmic Composition*.

[9] Dan Hosken (2011). *An Introduction to Music Technology.* Ed. Routledge.

[10] Eduard Rec Miranda (2002). *Computer Sound Design*. Focal Press.

[11] Alpern Adam (1995). *Techniques for Algorithmic Composition*. Hampshire College.

[12] Boon Jean Pierre, Decroly Olivier (1994). *Dynamical systems theory for music dynamics*. Universite libre de Bruxelles.

[13] Hiller Lejaren, Isaaccson Leonard (1959). *Experimental Music: Composition with an Electronic Computer*, McGraw-Hill, New York.

[14] Di Nunzio Alex (2011). *Illiac Suite*, MusicaInformatica, Computer Music Resources.

[15] Clark Robinson (1998). *Stability, Symbolic Dynamics and Chaos*, CRC Press.

[16] Hale-Resnick (1992). *Física Vol. 1.* Compañía Editorial Continental.

[17] Elaine A. Rich (2007). *Automata, Computability and Complexity: Theory and Applications.* Prentice Hall.

[18] Arto Salomaa (2011). *Computation and Automata.* Cambridge University Press.

[19] Nguyen Dinh Cong (1997). *Topological Dynamics of Random Dynamical Systems.*

[20] Clark Robinson (1998). *Stability, Symbolic Dynamics and Chaos*, CRC Press.

[21] J. De Vries (2000). *Elements Of Topological Dynamics.*

[22] Gonze Diddier (2012). *Linear Difference Equations.*

[23] Lang Serge (1995). *Linear Algebra, Third Edition.* Springer.

[24] Rogers D. Thomas (1981). *Chaos in the Cubic Map.* Department of Mathematics, University of Alberta.

[25] Rudakov N. Vadim (2000). *Bifurcation Analysis of the Henon Map.* Kursk State Technical University, Department of Computer Science.

[26] Scheinerman R. Edward (1996). *Dynamical Systems* Department of Mathematical Sciences the Johns Hopkins University.

[27] Artuso Roberto (2004). *Chaos Classical and Quantum* ChaosBook.org.

[28] Sternberg Shlomo (2000). *Invitation to Dynamical Systems* Math 118,Spring 2,000.

[29] Leonel D. Edson (2013). *Relaxation to Fixed Points in the Logistic and Cubic Maps:Analytical and Numerical Investigation* The Abdus Salam.ICTP, Strada Costiera,11, Trieste 34151, Italy.

[30] Hassan Ahmad Kawa. *The Dynamics of Henon Map* Department of Mathematics,College of Education,Babylon University.

[31] Avila Ramirez (2011). *Caracterizacion de Sistemas Dinámicos Mediante Periodicidades* Instituto de Investigaciones Fisicas,Carrera de Fisica Universidad Mayor de San Andres.

[32] Mocanu Marcelina (2012). *A Problem solving approach to some applications of the roots of unity to regular polygons*. International Journey of Geometry Vol. 1.

[33] Wilson, S., Cottle, D. and Collins (2011). *The SuperCollider Book*. Cambridge, MA: MIT Press

[34] Finn Jacobsen (2011). *THE SOUND FIELD IN A REVERBERATION ROOM*. Acoustic Technology, Department of Electrical Engineering, Technical University of Denmark,

[35] Spickler Don, Bergner Jennifer (2011). *The mathematics behind anamorphosis*. Salisbury University.

[36] Wang DeLiang, Brown Guy J (2005). *Computational Auditory Scene Analysis*. (eds.) John Wiley Sons, Inc.

[37] Marlon Schumacher, Jean Bressons (2010). *Spatial Sound Synthesis in Computer Aided Composition*. Schulich School of Music of McGill University, Montreal, QC, Canada.

[38] Nils Petersa, Trond Lossiusb, Jan Schacherc, Pascal Baltazard, Charles Bascoue, Timothy Place (2009). *A STRATIFIED APPROACH FOR SOUND SPATIALIZATION.*

[39] Juha Merimaa (2006). *ANALYSIS, SYNTHESIS, AND PERCEPTION OF SPATIAL SOUND BINAURAL LOCALIZATION MODELING AND MULTICHANNEL LOUDSPEAKER REPRODUCTION.* Helsinki University of Technology Laboratory of Acoustics and Audio Signal Processing.

[40] Lambert M. Surhone (Editor), Mariam T. Tennoe (Editor), Susan F. Henssonow (Editor). *Ambisonics.* Beta Script Publishing.