ISSN 1850-2954

IUNA

Instituto Universitario
Nacional del Arte

RIM

revista de investigación multimedia
año 3. número 3. marzo de 2011

3

IUNA

Instituto Universitario Nacional del Arte

editorial

... cabe tener en cuenta las posibilidades que pueden llegar a seguirse de los encuentros entre ciencia y las prácticas artísticas. En muchos casos, los artistas toman inspiración en los hallazgos de la ciencia, o investigan de manera creativa en algunos campos tecno-científicos (por ejemplo, y en estos momentos, la ingeniería genética, la nanotecnología, la matemática del límite, el software avanzado, la astronomía, la robótica, la inteligencia artificial, son campos en los que unos u otros artistas han encontrado inspiración directa y a partir de los que de hecho realizan alguna investigación creativa suficientemente relevante, cuando menos desde el punto de vista artístico).

(José Luis Brea, cultura _RAM, 2007)

Edición especial dedicada al proyecto PICTO "Diseño y desarrollo de aplicaciones e interfases de Realidad Aumentada destinadas a síntesis y procesamiento de audio digital", en el año del Bicentenario (1810-2010).

En el año del Bicentenario (1810-2010) hacemos entrega del tercer número de la Revista de Investigación Multimedia del Área Transdepartamental de Artes Multimediales del Instituto Universitario Nacional del Arte.

La misma se presenta como una edición especial de los documentos de avance de la primera etapa del proyecto PICTO "Diseño y desarrollo de aplicaciones e interfases de Realidad Aumentada destinadas a síntesis y procesamiento de audio digital", aprobado por la Agencia Nacional de Promoción Científica y Tecnológica - Fondo para la Investigación Científica y Tecnológica (Ministerio de Ciencia, Tecnología e Innovación Tecnológica - Presidencia de la Nación).

El proyecto parte del paradigma "continuo realidad-virtualidad" (Milgram, Kishino y Takemura, 1994) el cual ubica a los entornos virtuales en diversas categorías, cuyo rango se extiende desde la realidad física hasta la realidad virtual. Situados en este paradigma, entendemos por entornos de realidad aumentada a aquellos que logran conjugar elementos virtuales con la realidad física que nos rodea. Dentro de esta clasificación se encuadra la mayor parte de las experiencias de multimedia escénica, como las instalaciones, las *performances* y las intervenciones interactivas en las cuales realidad y virtualidad se funden.

En esta entrega volcamos los resultados de siete artículos que avanzan sobre los tópicos planteados. "Nuevas caracterizaciones de la actividad musical en el aula" de Prof. Carmelo Saitta abre un espacio de análisis sobre la vinculación música - espacio áulico. "Composición asistida en entorno PD" de los Dres. Pablo Cetta y Oscar Pablo Di Liscia, y "Medidas de similitud entre sucesiones ordenadas de grados cromáticos" del Dr. Oscar Pablo Di Liscia presentan técnicas de composición y organización del material musical en el entorno de composición en tiempo real *Pure data*. "Desarrollo de un sistema óptico para interfaces tangibles (mesa con pantalla reactiva)" y "Diseño de interface para el desarrollo de una pantalla sensible al tacto con aplicación musical" del Ing. Emiliano Causa exponen, respectivamente, el desarrollo de una mesa con pantalla sensible al tacto y el diseño de una interface de pantalla sensible al tacto (del tipo multitacto) aplicada a un editor gestual de música. "Técnicas de síntesis y procesamiento de sonido y su aplicación en tiempo real" del Lic. Matías Romero Costas repasa algunas de las técnicas de síntesis y procesamiento de sonido más utilizadas y difundidas, desde un punto de vista teórico, a través de ejemplos de aplicación, implementados en el entorno de programación Max-MSP. Por último, el artículo "Técnicas de programación vinculadas a la realidad aumentada y a las interfaces tangibles" del DCV Tarcisio Lucas Pirotta, presenta las principales funciones y bloques de programación disponibles en los paquetes de librería ARToolKit y reacTIVision, para su aplicación a proyectos de realidad aumentada e interfaces tangibles.

Comité Editorial de RIM

staff

RIM

Director
Prof. Carmelo Saitta

Secretario de redacción
Dr. Pablo Cetta

Comité editorial
Ing. Emiliano Causa
Dr. Pablo Cetta
Dr. Pablo Di Liscia
Mg. Raúl Lacabanne
Prof. Gumersindo Jerónimo de Jesús Serrano Gómez
Arq. Daniel Wolkowicz

Colaboran en este número
Carmelo Saitta
Pablo Cetta
Pablo Di Liscia
Emiliano Causa
Matías Romero Costas
Tarcisio Lucas Pirotta

Diseño
Arq. Daniel Wolkowicz

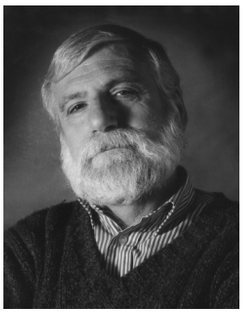
Corrección de textos
Rossana Cabrera

RIM es una publicación del Área
Transdepartamental de Artes Multimediales
del IUNA
Yatay 843, Ciudad Autónoma de Buenos Aires,
República Argentina

ISSN 1850-2954

Impreso en New Press Grupo Impresor S.A.
Paraguay 278, Avellaneda
Provincia de Buenos Aires
Marzo de 2011

índice



CARMELO SAITTA

Compositor y docente.
Estudió composición con Enrique Belloc, José Maranzano, Francisco Kröpfl y Gerardo Gandini. Ha compuesto obras de cámara y de música electroacústica; por "*La Maga o el Ángel de la Noche*" recibió un premio en Bourges en 1990. Y el Primer Premio instituido por la Municipalidad de la ciudad de Buenos Aires en 1991.

También ha compuesto música para cine. Ha estrenado numerosas obras de otros compositores, tanto como percusionista como en calidad de director. Desarrolla una intensa actividad docente en el Instituto Nacional del Arte y la Universidad Nacional de Quilmes.

Son de destacar sus aportes al conocimiento y uso de los instrumentos de percusión en la composición musical, y al tratamiento del sonido y la música en los medios audiovisuales.

Nuevas caracterizaciones de la actividad musical en el aula

Carmelo Saitta

No es aventurado decir que en educación musical, como en otras actividades humanas, los procesos que van configurando el devenir histórico se van articulando a partir del siguiente esquema: tesis, antítesis y síntesis; este hecho se repite con periodicidad variable, pero de manera ineludible.

En nuestro país es fácil ver cómo se han articulado las dos primeras etapas a partir de la oficialización, en el siglo pasado, de la enseñanza de la música. A riesgo de ser simplistas, podemos decir que la primera se constituyó a partir de la llegada de inmigrantes (instrumentistas en su mayoría) procedentes de diferentes países europeos, que pasaron a formar parte de las orquestas de los teatros de ópera que comenzaban a funcionar en Buenos Aires.

Estos músicos no solo se ocuparon de la ejecución de la música europea, sino también de ir formando a los primeros instrumentistas capaces de asumir la responsabilidad de ocupar un atril en las orquestas que se iban creando. Ya fuesen inmigrantes o nativos formados en Europa, no solo tenían como finalidad tocar bien un instrumento, sino también ejecutar el repertorio tradicional que en el desarrollo de su técnica habían aprendido. Estos primeros educadores (algunos de los cuales todavía se encuentra entre nosotros) tenían una finalidad: producir músicos; y para ellos, músicos eran los instrumentistas.

La reacción no se hizo esperar. A este primer movimiento se le opuso otro, surgido a la luz de las corrientes pedagógicas que se desarrollaron en Europa durante la primera mitad del siglo XX, y si a la primera corriente se la podría llamar "de los instrumentistas", a ésta deberíamos llamarla "de los pedagogos".

Esta segunda etapa se fue consolidando sobre una serie de postulados: primero está el niño, su sensibilización, su formación integral, el desarrollo de sus potencialidades creativas, su cultura musical, etc. No fueron pocos los logros de esta etapa: la puesta en crisis de los métodos conservadores, la generalización de la enseñanza de la música en las escuelas primarias y secundarias, la creación de institutos privados de educación musical, etc. Pero, asimismo, no pocos fueron sus errores: se olvidaron de la música.

Ahora ya es tiempo de preguntarnos qué sucede con la tercera etapa: si ésta ha comenzado, cuáles son sus características, sus móviles, qué propósitos persigue, etc. Digamos en principio, y para ser coherentes, que se deberían rescatar aquellos aspectos que han caracterizado a las etapas anteriores y que siguen teniendo valor musical o metodológico. Habrá que proponer una revisión de los mismos y desechar los otros, los que no son pertinentes, los que no han producido lo que de ellos se esperaba.

Para esta reformulación, hoy contamos con que:

• La psicología pone al alcance de nuestra mano las herramientas que permiten detectar en cada persona los indicadores de su capacidad creadora, facultad que debe considerarse entre las más susceptibles de ser cultivadas y desarrolladas. Los métodos de trabajo para lograr sus fines, como los cuidados que deberán tenerse para evitar su retraso o estancamiento. No debe olvidarse que esta conducta es la que presenta una mayor vulnerabilidad.

• La sociología ha demostrado que los movimientos culturales no solo son hechos sucesivos sino también simultáneos, y que por diferentes factores unos prevalecen sobre los otros, emergen a la superficie, se hacen más visibles; en tanto, los otros subyacen en el "interior" de una sociedad, y más allá de los factores que promueven su ascenso, si son consistentes, tarde o temprano terminan por caracterizar un determinado periodo histórico. Cabe entonces pensar que esta corriente (si es que así la podemos llamar) viene gestándose desde hace más de veinte años y, como se verá, indicios más que suficientes avalan esta afirmación.

Esta disciplina ha estudiado en profundidad la indivisible relación entre cultura y sociedad, como también el valor de los productos artísticos dentro de la cultura. Y es aquí que debemos diferenciar el producto de la actividad que le dio origen, y es esta actividad alternativa la que excede el marco de lo individual para instalarse en la periferia del núcleo social, para imponerle a éste su carácter dinámico y transformador.

• En cuanto a la música, los desarrollos tecnológicos ponen al alcance de cualquier persona, en cualquier lugar del planeta, las expresiones musicales más variadas y sorprendentes. Hoy es posible tomar contacto con la música de todos los pueblos del mundo, con las expresiones más actuales y con las más arcaicas; se dispone de lo necesario para conocer sus técnicas, sus instrumentos, y se cuenta con la información para abordar cualquier género o estilo.

• En cuanto a los medios, las posibilidades que nos ofrecen hoy día favorecen en mucho toda realización; piénsese nada más en el hecho que, cien años atrás, para tener contacto con la música solo se contaba con un piano en las familias de mayores recursos, o con algunos instrumentos étnicos si se trataba de sociedades rurales. Hoy es común encontrar institutos u hogares con un teclado y su respectivo secuenciador, o con una computadora cuyos recursos, además de su inmediatez, son casi ilimitados. Además, en los talleres de iniciación musical se tiene a disposición un número grande y variado de instrumentos y medios aptos para registrar cualquier evento.

Sin embargo, los albores del 2000 se nos presentan paradójicos en lo que atañe a la creación infantil: los recursos de que se dispone son prácticamente ilimitados, mientras que la producción es casi nula. Es de esperar que este estado de cosas pueda revertirse.

No obstante, y antes de considerar la caracterización de esta nueva etapa, será necesario tomar dos recaudos: primero, establecer los criterios con que se determinará la validez o no de un postulado; segundo, establecer lo que se propone como alternativa.

Para volver al problema planteado anteriormente –aquello que será desechado o rescatado– la respuesta dependerá del punto de vista, de los móviles que originan tanto las objeciones como la necesidad de incorporar nuevos temas, del tratamiento que se le dará a estos nuevos temas. Es decir, dependerá, en última instancia, de los nuevos objetivos.

Esta tercera etapa, la que responde a esta instancia de síntesis, es para Michel Serres[1] "*la clave de una pedagogía de la innovación y del crecimiento*". Para caracterizar a esta nueva etapa de la enseñanza de la música en nuestro país, será necesario considerar, aunque más no sea a titulo de inventario y para establecer el marco conceptual pertinente, los siguientes criterios:

• Entender que la música es una actividad que puede ser orientada al espectáculo, entretenimiento o diversión. También puede estar orientada a la magia, en cuanto puede realizar una representación de la vida práctica a través de ceremonias religiosas, sociales, actos patrios, canciones escolares que evocan gestas, el folk, etc. Puede ser también una forma de arte (creación de símbolos dirigidos a la imaginación). Con todo ello podremos definir tres orientaciones educativas bien diferenciadas.

1. El área de la animación sociocultural, actividad asistemática, carente de una metodología y finalidad educativa, cuyo fin es básicamente el entretenimiento (donde la música puede integrarse con otras artes).

2. El área educativa general, a partir de la pedagogía musical, orientada tanto a la adaptación sociocultural como al desarrollo de una actitud crítica, teniendo todo como fin general el desarrollo del imaginario.

3. El área educativa especial, ya sea orientada a la reproducción (instrumentistas o intérpretes) o bien a la producción (composición, creación musical), que tienen como fin general el aprendizaje y desarrollo de una forma particular de conocimiento.

Siendo estas orientaciones tan diferentes, forzosamente les caben contenidos y formas de transmisión también diferentes.

• Entender que nada que se diga de la música está fuera de la música. Propiciar, por lo tanto, una actualización de la problemática propia de la transmisión de conocimientos, habilidades y demás, no solo a la luz de las nuevas técnicas pedagógicas, sino también en función de los aportes que ha hecho la música contemporánea en cuanto a la ampliación de sus recursos y la reinterpretación de los principios tradicionales.

• Entender que es necesario reducir la brecha existente entre las estructuras de producción de conocimientos (los creadores) y las de distribución de los mismos (docentes).

• Entender la producción musical local privilegiando sus productos, por ser estos parte de la cultura de la sociedad en que convivimos y fomentar la producción de bienes culturales en la clase de música.

• Entender, en lo que hace al aprendizaje, que los contenidos deben ser abordados desde los aspectos mecánicos, prácticos, teóricos y analíticos.

• Entender que los "cómo" están en función de los "qué" y que estos están al servicio de la función social que asignemos a la música.

Es fácil rastrear indicios de lo que acabamos de decir en una considerable cantidad de trabajos teóricos o prácticos que se ocupan de la educación musical generalizada, desarrollados en las últimas dos décadas y que ponen énfasis en los siguientes aspectos:

• La actividad musical en el aula como parte del lenguaje de la época.

• Un uso mayor del timbre y las texturas, y menor de las alturas y los ritmos exactos.

• La diversidad de estilos que caracterizan a la música del siglo XX permite tener una visión más global del problema que genera su enseñanza.

• Las diferentes formas de representación desarrolladas ponen a la música fuera del código tradicional, permitiendo un fácil acceso al docente no especializado.

• La educación musical actual concibe a la música como un todo, no estudia sus aspectos como variables independientes.

• La incorporación de medios de fácil ejecución y de buena respuesta sonora como pueden ser los instrumentos electrónicos, los instrumentos de percusión y otros medios sonoros.

• La necesidad de un aula especial para la actividad musical y la no determinación de plazos para el desarrollo de los contenidos.

• La posibilidad de que cada alumno pueda realizar sus propias obras y, eventualmente, tocarlas o dirigirlas.

• Desde el punto de vista de los contenidos, se supera la ruptura entre la música del pasado y la música contemporánea; se ve a ésta como una continuación de aquélla pues se han incrementado los recursos (se usan más sonidos y más formas de utilizarlos).

• Un significado nuevo para la música en la escuela, orientado al desarrollo de la actividad simbólica del individuo.

• Se propone independizar la "realidad" del código de representación. (La música no es "signos en un papel" sino sonidos organizados). La teoría de la música no es el repertorio de signos que constituye la así llamada "teoría de la música".

• Que las actividades musicales de los alumnos no solo transcurran fuera de las aulas.

• Se promueve el aprender a escuchar. Dar la misma importancia a lo sensible que requiere tanta atención como las técnicas y las habilidades.

• Si se parte de las cualidades evocativas, se lo hace para pasar al análisis de las cualidades acústicas o constructivas musicales.

• Para ahondar más en las cualidades del sonido se propone tomar un repertorio limitado de estos para estudiar sus posibles combinaciones (dar importancia a la instrumentación y a la orquestación tanto como a las alturas y a los ritmos).

• Incorporar el repertorio de obras o de estudios compuestos en este siglo que se puedan analizar o tocar en clase, en particular el de los compositores recientes.

• Se propone una forma nueva de objetivar la realidad fuera del marco restringido de la Historia o de la Geografía. Es decir, que la multiplicidad del arte actual permite desarrollar conceptos generales que ayudan a entender la música de diferentes períodos históricos como también de diferentes culturas.

• La incorporación de los medios tecnológicos y de las computadoras como valiosos auxiliares de la tarea educativa.

Lo que acabamos de señalar es ya común en las clases de muchos educadores. En ellas se requiere una nueva dinámica y un conocimiento profundo de la psicología del aprendizaje, como también un dominio de los problemas propios de la música. Mucho se ha escrito en los últimos años sobre esta problemática. Las nuevas generaciones de docentes no solo están dando indicios sobre el buen criterio con que seleccionan el material necesario para su formación o para el uso en clase, también están dando indicios sobre la necesidad de que se ahonde en estas transformaciones.

Digamos, asimismo, que somos conscientes de que es necesario hacer un análisis profundo de cada uno de estos temas, pero –como ya dijimos– la caracterización última dependerá de los objetivos que le asignemos a la educación musical, y esto también es tema para una discusión.

Debemos considerar a la creación infantil como una actividad socialmente inexistente. En todo caso, y por la especial naturaleza de esta actividad, no existen en la Argentina –y probablemente en toda América del Sur– programas públicos de incentivos orientados a la producción musical infantil como sucede con las artes plásticas (concursos de dibujo, de expresión libre, de manchas, etc.).

En cambio, sí existen concursos, competencias u otros incentivos para la ejecución precoz de instrumentos (desarrollo de habilidades), lo que no siempre es un indicio de lo creativo.

En primer término, es necesario establecer la diferencia entre la producción musical destinada a la población infantil, cuyo aspecto creativo –salvo honrosas excepciones– es realmente dudoso, y aquello que los niños

pueden producir. Esto último puede ser visto desde dos lugares bien diferenciados: la creación individual espontánea y la creación individual o grupal orientada pedagógicamente.

Cabe entonces pensar que la creación musical infantil puede existir como una actividad individual aislada o formar parte de algún taller especial de iniciación musical donde se privilegie el crear música y no su ejecución (reproducción). Pero, si bien esta actividad puede ser social, no necesariamente es por ello conocida.

Hasta donde sabemos, la creación individual espontánea, tal como sucede con otras conductas, se basa en la imitación. Los infantes tratan de "emular" a sus modelos y su producción dependerá del tipo de música que escuchen, de los medios que estén a su alcance y del incentivo que reciban del núcleo familiar y, si bien las conductas imitativas son opuestas a las creativas, por el complejo mecanismo de transferencia se pueden encontrar indicios de creatividad en sus trabajos.

Del mismo modo, las actividades musicales infantiles orientadas pedagógicamente dependen también de un número considerable de variables. Éstas nos permiten obtener desde productos simplemente recreativos (cuyos rasgos por ser orientados por personas con mayor conocimiento, se ajustan "mejor" a un modelo, presentando un menor grado de desvío y, por lo tanto, una cuota menor de originalidad), hasta productos con un alto grado de originalidad que dan testimonio de la creatividad de sus hacedores.

En este punto cabe preguntarse hasta qué edad un individuo puede ser considerado un infante y hasta que punto la orientación pedagógica considera estas variables y propicia su desarrollo. También interrogarnos sobre qué es la creatividad, si es posible desarrollarla, cómo se la reconoce, de qué depende, etc.

Si bien esta no es la oportunidad para contestar a estos interrogantes, es necesario mencionar, sin embargo, que un objeto creado es un objeto que antes no existía. Es decir, que debe diferenciarse a la creación de la invención, del descubrimiento, de la reproducción y tomar en cuenta que este objeto artístico creado no es un objeto real ni tampoco ideal (no es una idea), sino un objeto virtual. Del mismo modo, el espacio y el tiempo que le son propios, también son virtuales. En esencia, la clave de la actividad artística es **la creación de objetos virtuales de acción** y ésta debe diferenciarse inevitablemente de las otras actividades.

En este sentido, al considerar la creación infantil debemos considerar la relación bidireccional que se establece entre estos objetos simbólicos y su dimensión espacio-temporal, como también su posible correlato con el resto de las actividades, objetos y dimensiones, sean estos reales o pertenecientes al mundo de las ideas. También es necesario tener presente el nivel de desarrollo en que se encuentra cada individuo, su etapa operativa, la que determinará en buena medida de las características del producto resultante.

La creación musical infantil dependerá entonces del grado de creatividad individual, de la conciencia que se tenga de los objetos artísticos y de las posibilidades de concreción, de la capacidad operativa del individuo; es decir, de su nivel de desarrollo.

La consideración de estas pocas variables nos induce a formularnos los siguientes interrogantes: estas condiciones ¿son innatas o adquiridas? Si son innatas ¿pueden ser atrofiadas por la relación del individuo con la sociedad? Si son adquiridas ¿será posible desarrollarlas? Si es así ¿de qué dependerá dicho desarrollo? No hay duda de que parte de esta potencialidad es hereditaria, como tampoco existe duda sobre la posibilidad de su desarrollo o de su inhibición.

En este sentido las orientaciones pedagógicas son un factor determinante. Las mismas son siempre instrumentos de una voluntad política de la cual depende su orientación.

Hechas estas consideraciones, podemos analizar las condiciones que el comienzo del siglo nos presenta; las cuales –dada la realidad– nos obligan a esbozar algunas reflexiones tendientes a comprender la actual situación.

Para ello será necesario contar primero con la voluntad político-social. Es decir, con una estructura social tal que, más allá de su núcleo (sus tradiciones, aquellos rasgos propios que la caracterizan), sea capaz de aceptar y promover las transformaciones deseables en todos sus estamentos, sean del orden que sean.

Será necesario también implementar un sistema educativo eficaz que valorice lo creativo, una pedagogía abierta que prepare, no para resolver problemas ya resueltos, sino para dar respuesta a problemas todavía no planteados.

En el marco restringido de la actividad musical, y en particular en el de la creación infantil, será bueno fomentar esta actividad desde el primer día de clase, asumiendo los riesgos que ello implica, aceptando una producción divergente, experimental e imperfecta.

Será conveniente, entonces, incentivar desde la más temprana edad el desarrollo de la creatividad, a la vez que procurar la transferencia paulatina de aquellos conceptos (transposición didáctica) que hacen al aprendizaje de lo musical.

De este modo, se podrá tener una real dimensión de los beneficios que produce la actividad creativo-musical, tanto al nivel individual como social.

En este sentido, habrá que tener en cuenta por lo menos dos grandes áreas de desarrollo. La primera orientada hacia la música como arte autónomo. Dentro de la misma podemos diferenciar dos formas de abordaje: una referida a la música instrumental que puede comenzar por los instrumentos de percusión, los que (además de su fácil acceso) permiten desarrollar la sensibilidad material fuera de los sistemas convencionales de organización. La otra, orientada a las computadoras, en principio por su gran facilidad para la construcción de estructuras, basadas no solo en los sistemas de altura. De ese modo ambas formas se constituyen en complementarias.

La segunda área es la referida a los multimedios y debe privilegiar los diferentes niveles de semanticidad del mensaje acústico y, por ende, las diferentes formas de comunicación a través de las estructuras audiovisuales.

Éstas u otras orientaciones deberán tener como marco la educación sistemática a través de estructuras educativas promovidas desde el sistema social. A otro nivel, la actividad individual, asistemática, deberá tener igualmente un ámbito social de contención (jornadas, concursos, talleres, etc.). Estos últimos seguramente desde las organizaciones sociales intermedias.

El comienzo del siglo nos pone una vez más frente a la falsa encrucijada: la actividad artística es ¿entretenimiento, lugar de sublimación, terapia? ¿O es una forma de conocimiento?

En ambos casos, para abordar libremente estas orientaciones será necesario implementar el manejo de las estructuras musicales con seriedad, puesto que la actividad musical es siempre una forma de actividad mental particular, imposible de ser sustituida por otra.

En todos los casos habrá que considerar el tema de la creación con seriedad, en particular en la más temprana edad, puesto que, como dice Pierre Francastel, "El mamarracho no es el primer nivel de la plástica".

Si pensáramos por un momento en las transformaciones que ha experimentado el arte del siglo XX (que ha puesto en crisis sus propios principios), no sería fácil referirse a alguno de sus aspectos sin correr el riesgo de parcializar o instalar una visión muy limitada de la realidad, en particular cuando se trata de abordar aspectos tan complejos como lo son aquellos referidos al concepto del lenguaje.

Frente a este problema, cuando de música se trata, se podría decir que hasta el mismo concepto de lenguaje es discutible, puesto que no es pertinente hablar de lenguaje en el sentido en que lo hacemos respecto de los lenguajes verbales, en los que existe, además de un significante, un significado.

Podríamos salvar la situación diciendo que en la música (tal como en otras artes), significado y significante se homologan: el significado es el significante, lo cual suprime toda "semanticidad" (esto, por supuesto, no es tan así). También podríamos decir que el concepto de lenguaje, cuando es aplicado a la música, no se refiere al sistema de organización de una determinada música sino al conjunto de las obras que constituyen la historia de esta particular actividad humana. Podríamos decir, en fin, que el concepto de lenguaje es pertinente en la medida en que esté referido a la idea de los lenguajes expresivos, en oposición a los lenguajes narrativos.

Sea como fuere, deberíamos hacer dos salvedades. La primera es que, en lo que a música se refiere, el concepto de lenguaje cambia de una cultura a otra. La segunda es que en un determinado período histórico pueden coexistir simultáneamente diferentes formas.

Por eso, más allá de la relatividad del concepto mismo, nos referiremos a ciertos aspectos de la música occidental y, en particular, a aquellos que hacen a las innovaciones y aportes en esta disciplina.

A riesgo de ser simplistas, podríamos decir que tradicionalmente la estructura musical se basaba en un doble sistema. Uno referido a las asignaciones de alturas y otro al campo de la organización rítmica. En este doble sistema el sonido era un mero soporte, y toda consideración sobre el mismo lejos estaba de ser organizada. La misma noción de estructura dejaba afuera a la idea del material (entendido aquí como el sonido).

De todas formas podemos decir que en estos casos la altura es considerada como parte del sistema y no como una de las cualidades materiales del sonido. Hoy sabemos que la altura tonal difiere de la altura espectral y, por lo tanto, que estos sistemas solo pueden constituirse a partir de sonidos armónicos. Pero en el siglo XX los compositores se lanzaron a la conquista del sonido y ello trajo como consecuencia otros problemas, otros puntos de vista, otras posibilidades.

Debemos considerar que en una determinada obra, y desde el punto de vista de la altura, se superponen tres sistemas selectivos: la división de la octava en doce partes; la selección de siete de ellas dentro del total cromático; y por último, de estas siete, las que el compositor decide usar. Aquí ya se presentan dos campos de especulación:

1. Es posible vislumbrar un enriquecimiento del lenguaje a partir de obtener sistemas de 7 alturas dentro

del total cromático; es decir, la posibilidad de contar con 49 escalas diferentes (7 x 7), a las cuales se pueden agregar escalas hexatónicas, pentatónicas, etc. En el sistema tonal solo se usaban 4 (una mayor y tres menores).

2. Es posible pensar en la posibilidad de ampliar el repertorio de alturas dentro de la octava, dividiéndola en 18 ó 24 partes, con lo cual se obtendrán tercios y cuartos de tono, siempre considerando a la altura como eje. Por supuesto, otro aspecto sería tener en cuenta las posibilidades politonales o pantonales. Pero, como dos alturas delimitan un intervalo, también es posible pensar en sistemas constituidos a partir de un determinado tipo y número de intervalos, tal como sucede con la música atonal y con otros sistemas como el serial, el serialismo integral y otras derivaciones posteriores, cuyas elaboraciones llegan hasta nuestros días.

Aquí también el siglo XX nos presenta grandes innovaciones. En principio considerando la posibilidad del uso de otras escalas (recordemos que existen diez escalas naturales diferentes), luego reinterpretando viejos principios y dando a las nuevas conclusiones preeminencia en la composición; por último, incorporando nuevas ideas que llevarán a la instauración de la indeterminación, la estadística y el cálculo estocástico.

En cuanto a la organización rítmica, la música occidental se ha basado en los principios de la rítmica prosódica ya establecidos por los griegos. El derrotero que ha seguido la evolución de este sistema nos ha permitido el gran desarrollo alcanzado en este aspecto en las postrimerías del siglo XIX y el principio del siglo XX.

Una mirada general nos permite ver diferentes momentos que van desde la libertad rítmica del canto gregoriano hasta el "encorsetamiento" del barroco; desde el sujetamiento y linealidad de la camerata fiorentina hasta la libertad chopiniana y la multiplicidad de Ives y Stravinsky. No obstante, debemos señalar que todo este desarrollo se ha basado en la conquista de una escala cuyas proporciones responden a la serie 1, 2, 4, 8, 16, 32, 64.

Se podría decir que las innovaciones enumeradas hasta aquí marcan cierta continuidad con la tradición europea, pero en el marco de lo social la importancia de las artes radica en su aspecto innovador y creativo. Las artes deben cuestionar más que garantizar la tradición. Es la mirada crítica del artista lo que favorece las transformaciones sociales, mirada crítica que el arte tiene para sí y que es en esencia el factor que determina su dinámica. Pero si bien en el arte en general y, por ende, en la música, las innovaciones son una constante, nunca como en este siglo el hombre se ha visto frente a un número tal de transformaciones; transformaciones que por su naturaleza han puesto en tela de juicio sus más firmes postulados.

Desde ya que estos cambios no se operan solo en el arte, es fácil constatarlo en todos los órdenes del quehacer humano. Tal vez sea oportuno decir que en muchos casos estos han sido de orden cualitativo, y que sus principios no se explican como el desarrollo de una instancia anterior (el reloj electrónico –semiconductor– no es una evolución del reloj mecánico). En este sentido, la misma idea de música ha sido cuestionada.

Volviendo a nuestro tema, es difícil determinar si la música concreta y la electrónica son estilos evolucionados del lenguaje musical o si constituyen nuevos lenguajes. Sus mismos hacedores se refieren a ellas como esculturas sonoras, plástica sonora, nuevas poéticas sonoras, etc. No hay duda de que estas nuevas expresiones plantean incluso grandes problemas de comunicación (¿qué podemos decodificar cuando se desconoce el código?), pero sabemos que comunicación e información son inversamente proporcionales: a mayor información, menor comunicación.

Será cuestión de rechazar estas expresiones o pensar que tal vez el código es otro; que los elementos de formalización que responden a determinados códigos han sido sustituidos por otros (que dicho sea de paso siempre han estado presentes) que también tienen la virtud de ser nociones de cambio pues son datos perceptibles capaces de cumplir un rol formal tanto o más importante que aquellos, y más ahora cuando los medios tecnológicos han puesto al alcance de nuestros oídos la música de otras culturas y de otros períodos históricos.

Si frente a la música electroacústica tenemos dudas, del mismo modo deberíamos tenerlas frente a una música africana carente de melodía y armonía por estar basada solo en sonidos de tambores. ¿A ésta la podremos llamar música? Tal vez nos quede como único referente el ritmo métrico, ritmo considerado "natural", en oposición a los ritmos irregulares y discontinuos, los cuales más que responder a modelos fisiológicos, como la respiración o la pulsación cardíaca, responden perfectamente a los ritmos psíquicos, los que por supuesto son tan "naturales" como los otros.

Pero cómo hablar de natural en el arte, donde todo es artificio. El tiempo musical es independiente del tiempo psicológico o del cronométrico. Es otra dimensión, es un tiempo virtual, y en la música del siglo XX se ha replanteado no solo el tradicional concepto del tiempo sino también el del espacio y el de la materia. Ahora el sonido es portador de información (por otro lado, siempre lo fue) y es esa información la que los compositores quieren poner de manifiesto.

Es más, en muchos casos la música toda es una proyección macro de esa microestructura. El sonido ya no es un mero soporte, sino el punto de partida de la

composición. El compositor compone primero el sonido, y éste ya no es solamente armónico pues los compositores han avanzado sobre el resto de sus cualidades. Ya no se trata de la relación entre dos alturas, que es una magnitud abstracta, se trata de vincular las cualidades del sonido. Es así como se incorpora a los códigos el uso del timbre, el movimiento, la inarmonicidad, etc.

Si a estas pocas consideraciones respecto del material agregamos las referentes a las nuevas elaboraciones rítmicas, estructurales, texturales, formales, expresivas... en fin, estéticas, vemos la riqueza, la diversidad, la multiplicidad de ideas que caracterizan a la música del siglo XX y que dan cuenta, en última instancia, de la conquista de la libertad por parte de los compositores, libertad expresada en todos los órdenes de la composición.

Todas estas cuestiones (amén de otras) nos llevan a replantearnos el lugar del espectador (del oyente), y ni que decir el de la enseñanza de la música. Está claro que los mecanismos que ponemos en juego en primera instancia para comprender la música de los siglos XVIII y XIX no nos sirven de igual manera para comprender otras expresiones musicales más actuales. Tampoco, paradójicamente, nos sirven para comprender músicas más antiguas o pertenecientes a otras culturas. ¿Será necesario dividir en áreas la enseñanza de la música a fin de que en cada una de ellas enseñemos lenguajes diferentes, o será tal vez más pertinente encontrar normas o criterios generales que permitan incluir diferentes expresiones bajo enunciados más abarcativos? Estas dos diferentes instancias establecen, hoy más que nunca, una diferencia notoria entre una educación conservadora, enciclopedista, acumulativa y una educación dinámica, transformadora, creativa.

Si hacer música es crear una forma virtual de acción en el tiempo (cuando no también en el espacio), entonces podemos decir que cualquiera de los factores que intervienen pueden tener la posibilidad de formalizar. Desde ya no todos los parámetros tienen la misma jerarquía, pero: ¿tal vez el arte no consiste en alterar ese orden, en crear una ilusión, en estimular la imaginación, en ampliar el marco de la experiencia sensible?

Y por último, si la plástica al perder la figuración no hizo más que poner en evidencia los valores plásticos, aquellos aspectos fundamentales que se "ocultaban" detrás del "tema"; en música, de manera análoga, podemos decir que la ampliación de los lenguajes, o si se quiere, la creación de nuevos, ha permitido la profundización y ampliación de los criterios constructivos, mediante la inclusión, en su formalización, de otros parámetros.

Este desarrollo ha llevado a la música occidental a tomar contacto con otras culturas y a conformar un corpus conceptual lo suficientemente abarcativo como para dar cuenta de cuanto planteo temporal o espacial ha sido posible imaginar.

Mencionaremos a título informativo algunas de estas nuevas caracterizaciones:

• Considerando el aspecto estructural, se hace evidente la libertad de que dispone el compositor en este siglo, referida a la organización interna sobre el doble eje ya citado. Abandona la concepción derivada de la tradición para dar lugar a estructuras cuyos modelos, en muchos casos, son inferidos de estructuras matemáticas o de procesos de mutación celular o de expansión orgánica propios de la biología. En otros casos la estructuración es derivada de la escala de los armónicos o se construyen otros sistemas para independizarse de las escalas habituales, dando lugar a escalas microtonales. Hay procesos basados en la materia sonora, sea ésta orientada a la inarmonicidad, con la inclusión de la percusión; a los multifónicos de los instrumentos de viento; o que parten de otras cualidades de sonido como la música concreta o electrónica. En ciertos casos se llega hasta la negación del sistema mismo, ya sea por el permanente uso del total cromático (Ligeti) o por sustentar la obra sobre una sola nota (Scelsi).

• También la sintaxis derivada del modelo verbal ha sido abandonada para adquirir cierta autonomía; en algunos casos conservando las funciones tensionales, y en otros decididamente negándolas, ya sea por el hecho de proponer un cierto modelo estático o por proponer una constante evolución o transformación.

• En cuanto al campo rítmico, ya mencionamos la posibilidad del uso de otras escalas, como por ejemplo la usada por Messiaen en modos de valores e intensidades, cuyas proporciones responden a la serie 1-2-3-4-5-6-7-8-9-10-11-12; podríamos agregar la inclusión de valores irregulares como los usados por Stockhausen en algunas de sus obras 1-2-3-4-5-etc.; la consideración de valores irracionales (fracciones de valores irregulares), como en el serialismo integral; la liberación de toda medida, como sucede en la música aleatoria. También, como hemos dicho, la revisión de ciertas ideas clásicas ha dado nuevas herramientas al compositor; por ejemplo la idea de valores iguales y valores desiguales, la idea de reversibilidad e irreversibilidad rítmica, y operaciones más complejas como son, por caso, los ritmos homeotéticos en Boulez, etc. Otro tanto podemos decir de los pies métricos, de la conquista de la asimetría, del uso de las polimetrías y parametrías, de la conquista de la irregularidad y otras tantas operaciones de organización temporal.

• Con respecto al sonido, éste ha sido tal vez uno de los campos que más ha revolucionado las estructuras musicales. En este siglo hemos pasado del sonido como

un mero soporte, hasta estructuras donde éste impone sus propias condiciones. En este sentido, es tal vez este parámetro el que ha terminado por producir el mayor alejamiento del lenguaje tradicional, primero con la inclusión de los sonidos no temperados de la percusión (Varèse), luego con la de los sonidos "concretos" de fuerte connotación semántica, con la de los sonidos obtenidos mediante las técnicas extendidas de los instrumentos de viento (multifónicos, etc.) y, en última instancia, por el énfasis puesto en otros parámetros del sonido que anulan o enmascaran la tradicional noción de altura. Podemos decir que en las estructuras se han reemplazado, en cierto modo, las nociones de melodía, armonía y contrapunto, por una idea del diseño basado en los recursos instrumentales y sus asignaciones. Es decir, se ha jerarquizado la instrumentación y la orquestación hasta el punto de convertirlas en nuevos factores de formalización, en nuevos códigos.

• En cuanto a la idea de textura, no solo se ha logrado revitalizar tipos tradicionales, también se han agregado criterios que hasta aquí formaban parte de otras culturas; por ejemplo la polifonía oblicua (Webern) o las heterofonías (Ives), que son formas texturales constatables en mucha música oriental. Y esta incidencia de la música de otras culturas ha sido también un factor de transformación de los lenguajes musicales. Es notorio el contacto que se puede establecer entre ciertas corrientes de la música contemporánea y el folclore extra europeo.

• En cuanto al aspecto formal, también son relevantes las transformaciones que se han operado en este terreno. Desde las ideas debussyanas y otras más radicales como las de Varèse (donde la forma es resultante del juego libre de las estructuras), hasta la liberación total de la forma (formas abiertas) que se da en la música aleatoria, en las experiencias de improvisación que caracterizaron las décadas del 50 al 70, y en las nuevas ideas formales de la música electroacústica (live electronic, algoritmos, etc.). Si la forma es la resultante de la concepción de la obra; si, como dice Schönberg, "la forma hace inteligible la idea", entonces podemos decir que la idea es formal y que ésta depende, entre otras cosas, de la posición estética del autor.

• Desde el punto de vista estético, es impresionante la sucesión, cuando no la simultaneidad, de concepciones e ideologías que ha experimentado el arte del siglo XX. Estos movimientos, que en muchos casos han tenido su paralelismo en las otras expresiones artísticas, se suceden con tanta velocidad que, a veces, a pesar de la fluida comunicación que existe en esta era planetaria, no nos enteramos de su existencia hasta que su auge ha pasado. También es cierto que, como contrapartida a estas ideas globales, en los últimos años han surgido muchos movimientos (en particular en países periféricos como los nuestros), con estéticas y rasgos muy particulares, producto de una mirada crítica sobre el pensamiento universal y de una valorización, un resurgimiento de las tradiciones locales. El resultado es una suerte de "mestizaje", de "sincretismo", que ha dado lugar a un regionalismo crítico y que, por supuesto, aporta nuevos puntos de vista sobre la producción musical. Todo ello no hace más que aportar nuevos materiales, nuevos procedimientos, nuevas formas expresivas, nuevas formas de comunicación.

Recordemos que ya en la década del 60, Eco, refiriéndose al lenguaje musical, daba cuenta de la falta de un idioma común. Es más, ni siquiera se refería a la posibilidad de un dialecto, sino a la idea de idiolecto, concepto que usaba para aludir al sistema particular que cada obra tiene; lenguaje propio que le es inherente y que sirve, en muchos casos, para esa sola obra.

Si nos planteáramos una forma posible de abordar el trabajo compositivo en un nivel inicial, en particular para niños en edad escolar, el itinerario aconsejable es aquel que toda persona pondría en juego al tratar de construir algo. Es decir: primero una selección del material de que se dispone, segundo, algún criterio para su organización; en tercer lugar, y como consecuencia de la convergencia de lo anterior (materiales y organización), una determinada configuración, que en el caso de la música nos daría dos fenómenos diferentes: el de la superposición, su textura, y el de la yuxtaposición, es decir, de su forma.

Este esquema (materia-organización-textura y forma) no es la única manera de abordar la composición, pero, en términos educativos y en las primeras etapas, es –desde el punto de vista metodológico– el más indicado; puesto que todo planteo debe "encuadrarse" dentro de las posibilidades operativas que los destinatarios posean en el momento del aprendizaje.

En este sentido, es aconsejable considerar la experiencia acústica y, en particular, si se trata de instrumentos de percusión cuya variedad y espontaneidad de respuesta sonora es directa y fiel en función de la acción que le da origen, el universo sonoro resultante es tan rico que permite toda categorización posible.

En el caso de sonidos digitales a través de una computadora, será conveniente presentar una muestra lo más heterogénea posible a fin de propiciar la posibilidad de agrupamiento por las cualidades de semejanza-diferencia y analogía (esta última en el caso de una segunda etapa, es decir, sobre una acción modulatoria de las cualidades del sonido) lo que implica operaciones más complejas.

De ese modo se pondrán en evidencia gradualmente las cualidades de: tipo, variante formal, material, registro e intensidad.

Una vez determinada la cantidad y calidad de las muestras seleccionadas, el material deberá organizarse siguiendo uno de los posibles criterios temporales. En este caso, primero los relativos a los campos rítmicos (continuidad-discontinuidad, regularidad-irregularidad,

densidad cronométrica; es decir, cantidad de acontecimientos por unidad temporal), para luego, en una segunda etapa, considerar los ritmos elementales y las operaciones más simples (suma, subdivisión, reemplazo por silencio, por valores irregulares, elipsis y las arritmias: contratiempo, sincopa y parametrías).

Por lo que se refiere a las texturas, éstas no son más que la consecuencia de la interacción entre los posibles materiales y la forma en que estos son organizados en el tiempo.

En este sentido debemos decir que, si bien existen en cantidad y variedad, podríamos reducirlas a tres criterios básicos: integración, subordinación e independencia, que son consecuencia de uno o más materiales y/o de uno o más criterios de organización que coexistan simultáneamente. Estas unidades de configuración son consideradas de superposición, dado que lo resultante se da en una misma unidad de tiempo.

En este caso también será necesario planificar su grado de complejidad en diferentes etapas.

Las mismas consideraciones caben para el problema formal. Éste es básicamente un fenómeno de yuxtaposición y, si bien el número de formas posibles es altísimo, los criterios son también tres: permanencia, cambio, retorno; y estas alternativas no solo se dan en la forma total, sino también en todos los demás niveles. Por ser este un fenómeno esencialmente temporal, podemos definir a la forma como una sucesión de texturas, donde al igual que en ellas, caben una cantidad de variables que deberán ser consideradas en etapas sucesivas, que den cuenta de la mayor o menor riqueza musical, de un mayor "espesor" semántico.
En sentido amplio, se pueden prever tres etapas.

La primera debería comprender:

Desde el punto de vista material:	un solo tipo.
Desde el punto de vista temporal:	un único principio de organización global.
Desde el punto de vista textural:	unidades simples.
Desde el punto de vista formal:	sólo el criterio de permanencia.

La segunda debería comprender:

Desde el punto de vista material:	más de un tipo.
Desde el punto de vista temporal:	más de un principio de organización, tanto en sucesión como en simultaneidad
Desde el punto de vista textural:	unidades compuestas.
Desde el punto de vista formal:	los tres criterios.

La tercera debería comprender:

Desde el punto de vista material:	se debería avanzar sobre los grados de tonicidad y las demás cualidades materiales.
Desde el punto de vista temporal:	trabajar ya con ritmos elementales y sus operaciones más simples.
Desde el punto de vista textural:	unidades compuestas por más de un plano, en cuyo interior se elaboren relaciones de complementariedad y de subordinación.
Desde el punto de vista formal:	organizaciones que no solo involucren los criterios ya experimentados, sino también una articulación interna, usando los mismos criterios.

En el caso de un programa informático, éste debería rechazar estadísticamente tanto materiales como procedimientos, cuando estos excedan la media permitida; de modo tal que perceptivamente no queden dudas sobre las características de las unidades en cuestión, en función de un criterio de unidad y continuidad. De ese modo, obtendremos configuraciones más o menos simples, consecuencia de la azarosa distribución en el eje vertical y, en base a ello, el criterio de distribución horizontal.

Esto es la concreción de pequeñas unidades de sentido que respondan, en una primera etapa, a sonidos que tengan un mismo criterio tipológico con una determinada organización global; de modo tal que permita avanzar en una secuenciación de dichas unidades y, así, permitir la construcción de secuencias mayores (formadas por varias de las secuencias ya construidas) siguiendo criterios de unidad y continuidad, a la vez que puedan experimentarse los vectores tensionales propios de la construcción en período o de la pequeña variación.

Esta secuenciación, consecuencia de la yuxtaposición de las unidades ya elaboradas, deberá tener un alto grado de parentesco de modo tal de conservar cierta "fluidez", cierta unidad de sentido del discurso, en el devenir temporal.

Como se puede observar, ya se trate de un programa informático, de una experiencia instrumental, o de un procedimiento mixto, en todos los casos debemos considerar que la música es un lenguaje y que estos tienen estrictas reglas sintácticas que será necesario aprender.

Al igual que los lenguajes verbales, el aprendizaje de la música –muy particular por cierto–, requiere de una experiencia sensible cuyas posibilidades de desarrollo solo pueden sustentarse en un aprendizaje gradual y sistemático.

En todo caso diremos que más allá de su posible afinidad con el lenguaje verbal, se debería considerar a la construcción musical más cercana a un problema matemático o, para ser más precisos, deberíamos considerar al lenguaje verbal, a las matemáticas y a la música como las tres potencias del espíritu humano.

Referencias Bibliográficas

Bourin, F. *Le Tiers Instruit* (1991).

Saitta, Carmelo. *Trampolines musicales (Propuestas didácticas para el área de Música en la EGB)*, (Buenos Aires: Novedades Educativas, 1997).
–Creación e iniciación musical, Hacia un enfoque metodológico (Ricordi: 1978).

PABLO CETTA

Doctor en Música, en la especialidad Composición. Ha desarrollado una extensa labor como compositor, docente e investigador. Ha recibido diversas becas y distinciones, entre ellas la Beca Antorchas del LIPM y de la Fundación Rockefeller para realizar trabajos de composición en el CRCA (Universidad de California, San Diego), el Primer Premio en el Concurso Internacional de Bourges (Francia), el Premio Euphonies d'Or, el Premio Municipal de Música y el Segundo Premio Nacional.

Actualmente es Secretario Académico del Área Transdepartamental de Multimedia. IUNA.

PABLO DI LISCIA

Doctor en Humanidades y Artes, Mención Música en la UNR. Estudió, además, composición en forma particular con los maestros Dante Grela y Francisco Kröpfl. Fué Director de la Carrera de Composición con Medios Electroacústicos en la UNQ y Secretario de Investigación y Posgrado del IUNA. Actualmente es Profesor Titular y Director de la Colección "Música y Ciencia" de la UNQ y Profesor Titular del IUNA.

Ha recibido subsidios del Fondo Nacional de Las Artes, la Fundación Antorchas, la Fundación Rockefeller y la Fundación Música y Tecnología, y premios en competiciones nacionales e internacionales (Fondo Nacional de las Artes, Secretaría de Cultura de la Nación, Concurso Internacional de Bourges, Francia, etc.) Su música se ha difundido tanto en el país como en el exterior, en USA, Francia, Chile, Cuba, España, Holanda, etc. Ha publicado artículos sobre estética y técnica de la música y las nuevas tecnologías, y desarrollado software para proceso de sonido y música, análisis musical y composición.

Más información en: http://musica.unq.edu.ar/personales/odiliscia

Composición asistida en entorno *PD*

Pablo Cetta y Pablo Di Liscia

Palabras claves

Música, composición, matrices combinatorias

Resumen

El presente escrito resume diversas técnicas de composición musical centradas en la utilización de conjuntos de grados cromáticos y matrices combinatorias, y da cuenta de las aplicaciones informáticas especialmente desarrolladas para el tratamiento de estas técnicas. Se presenta aquí el punto de partida para el diseño y la implementación de interfaces que faciliten el uso de los programas mencionados, y conformen una plataforma eficaz destinada a la composición asistida.

1. Introducción

El segundo campo de aplicación elegido para el desarrollo de interfaces destinadas a la composición e interpretación de la música es el de las denominadas *matrices combinatorias*. Este tema ha sido previamente desarrollado por el Dr. Pablo Cetta y el Dr. Pablo Di Liscia en el marco de los Proyectos de Investigación "Desarrollo de aplicaciones informáticas para la organización de la altura temperada en la composición y análisis musical" (Área de Artes Multimediales, IUNA, 2007-2008) y "Formalización de procesos compositivos y desarrollo de aplicaciones informáticas para análisis y composición musical" (Facultad de Artes y Ciencias Musicales, UCA., 2007-2009).

Según expresáramos en el libro "*Elementos de Contrapunto atonal*"*, gran parte de la producción musical de la música occidental descansa en la organización de la altura en base al sistema temperado. Durante los siglos XVII, XVIII y XIX, se desarrollaron simultáneamente tanto la música tonal como su teoría. A partir de la llamada "Música del siglo XX", o "Música Contemporánea", comienza un proceso de disolución gradual del Sistema Tonal en el que se destacan tres tendencias:

1. La persistencia del sistema tonal de manera "extendida", o de sistemas análogos sobre base de otras escalas o modos, o el uso particular de estos –politonalidad, bitonalidad, etc.–, se encuentra en determinadas composiciones de Claude Debussy, Bela Bártok, Sergei Prokofief, Igor Stravinsky, Alexander Scriabin, entre otros.

2. La organización serial dodecafónica de la altura temperada, diseñada por los compositores de la Escuela de Viena (Arnold Schönberg, Anton Webern y Alban Berg), aproximadamente a partir de 1920 y continuada luego por compositores europeos y estadounidenses. La posterior derivación de ésta fue el serialismo integral, tendencia encabezada por compositores como Karlheinz Stockhausen, Pierre Boulez, Luigi Nono, Luciano Berio y Milton Babbitt, entre otros.

3. La organización "no serial" de la altura temperada que, sin embargo, no es tonal (o tonal-extendida, como el caso de la tendencia que mencionamos en el punto 1) ni serial y que se suele denominar "atonalismo libre". Esta se detecta en las primeras composiciones atonales de los compositores de la Escuela de Viena, aunque también en las obras de Edgar Varése, Charles Ives e Igor Stravinsky, entre otros.

Ya que la primera y la segunda de las tendencias mencionadas se basan en conceptos teóricos totalmente –o casi totalmente desarrollados, surgió la necesidad de diseñar un sistema teórico para la música Atonal no serial. Si bien no puede negarse que su fuente de inspiración la constituye la música pre y post serial atonal centroeuropea, fue inicialmente desarrollado por compositores y teóricos americanos (Milton Babbit y Allen Forte, como compositor y téorico y como teórico respectivamente, entre los más destacados). El sistema, se basa en la noción de Pitch class sets (Conjuntos de grados cromáticos) y utiliza recursos del Álgebra Combinatoria y de la Teoría de Conjuntos para organizar los grados cromáticos (Pitch Classes) del sistema temperado en grupos (Sets), y determinar sus propiedades estructurales. Sobre la base de las propiedades de los Pitch Class Sets surgen sus posibilidades de combinación y su potencial sonoro a explotar en la composición musical. Una posterior

*Cetta, P; Di Liscia, P. Elementos de Contrapunto atonal. EDUCA, Buenos Aires, 2009.

proyección es la formalización de las posibilidades de la disposición vertical y horizontal de estos conjuntos en matrices, que explotan sus características estructurales en la generación de diseños compositivos abstractos aplicables a la composición musical, tal como se desarrolla principalmente en los trabajos del compositor estadounidense Robert Morris. La combinatoria aplicada a matrices, sin embargo, no es exclusiva de la música atonal, sino que se desarrolla abundantemente también en la música serial dodecafónica. En este sentido, tanto la teoría de la música serial dodecafónica como la de la música atonal se complementan en una manera orgánica, dado que las formas canónicas de una serie dodecafónica (original, inversión, retrogradación e inversión retrógrada) no alcanzan a dar cuenta de la estructura subyacente de una obra musical y, por otro lado, los conjuntos de grados cromáticos de una obra no serial deben estar distribuídos organizadamente en el total cromático.

2. Los conjuntos de grados cromáticos

Describiremos brevemente algunos aspectos teóricos de la técnica de conjuntos de grados cromáticos. Si bien esta técnica contempla todas las agrupaciones de altura posibles, que van desde el conjunto vacío hasta el total cromático, vamos a concentrarnos solamente en los conjuntos formados por cuatro grados, a modo de ejemplo.

Si deseamos formar todos los subconjuntos posibles de cuatro notas tomadas de las doce pertenecientes al sistema temperado, debemos recurrir a una operación del cálculo combinatorio denominada variaciones simples, que se practica de acuerdo con la siguiente fórmula:

$$V_{n,m} = n\,(n-1)\,(n-2)\,\ldots\,(n-m+1)$$

Las variaciones simples son todos los subconjuntos de *m* elementos tomados de *n* objetos dados, de tal modo que dos subconjuntos se consideran distintos si difieren en algún objeto o en el orden en que van colocados en el conjunto (*ab* es distinto de *ba*).

Las cantidad de variaciones de 12 elementos tomados de a 4 es:

$$V_{12,4} = 12.11.10.9 = 11880$$

Un número tan elevado de posibilidades resulta imposible de controlar. Esta cantidad puede reducirse considerablemente excluyendo los subconjuntos que poseen los mismos elementos pero en orden distinto. A estos ordenamientos los denominamos permutaciones: 0,1,2,3 y 3,0,2,1 son permutaciones de los mismos elementos.

Dado que la cantidad de permutaciones de *n* elementos puede calcularse por:

$$P_n = n\,!$$

La cantidad de permutaciones de un conjunto de cuatro elementos son:

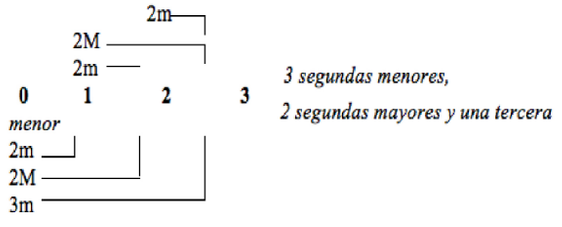
$$P_4 = 4\,! = 4.3.2.1 = 24$$

Eliminando las permutaciones reducimos el primer valor a 495 conjuntos (11.880/24). Resulta obvio que no es lo mismo 0,1,2,3 que 3,0,2,1, no obstante, los elementos son los mismos en ambos casos. Lo que nos interesa particularmente es que los intervalos o distancias entre todos los números, que en nuestro caso representan a los sonidos, son los mismos.

Si consideramos que las representaciones alfanuméricas de las notas son:

0	1	2	3	4	5	6	7	8	9	A	B
do	do#	re	re#	mi	fa	fa#	sol	sol#	la	la#	si

nos queda que 0,1,2,3 es un conjunto que posee 3 segundas menores, 2 segundas mayores y una tercera menor; 3,0,2,1, por su parte, posee la misma interválica, y solo difiere en los intervalos sucesivos que se dan en el orden melódico:



Por esto, siguiendo con nuestra intención de organizar la altura, podemos descartar las permutaciones en favor de reducir el número de combinaciones posibles.

También podemos minimizar el número de casos omitiendo las transposiciones e inversiones; 4,5,6,7 es una transposición de 0,1,2,3, y ambas poseen la misma interválica. Lo mismo sucede con 0,B,A,9, que es la inversión de 0,1,2,3.

Quedan en definitiva, luego de quitar las permutaciones, las transposiciones y las inversiones, 29 conjuntos de altura, cada uno con características interválicas propias.

Semejante reducción puede en principio parecer arbitraria, pero es similar a la que ocurre en la música tonal. El acorde mayor, por ejemplo, forma una estructura reconocible más allá de las transposiciones

o permutaciones que se le apliquen. En este caso, agrupamos ciertas permutaciones según la nota más grave y denominamos a cada grupo inversiones del acorde. A cada inversión se le atribuyen estilísticamente características funcionales específicas, particularmente a la segunda inversión que es la que más se aleja de la fundamental en la escala de los armónicos.

La diferencia principal de los conjuntos en relación con los acordes tonales reside en considerar a la inversión de un conjunto como una simple transformación del conjunto original. En el ámbito tonal, la inversión del acorde mayor es el acorde menor, y ambos son usados como tónicas de dos modos bien diferenciados, como parte de un sistema más complejo. Resulta entonces que, la inversión de un conjunto responde más bien a un criterio contrapuntístico que armónico. Algo similar ocurre en la música tonal, con la inversión de un motivo para ser utilizado como variante en la imitación.

Cada conjunto está caracterizado por un vector interválico, que expresa la cantidad de cada una de las clases interválicas. Las clases interválicas son seis: segunda menor, segunda mayor, tercera menor, tercera mayor, cuarta justa y cuarta aumentada. Una clase interválica comprende a un intervalo y a su inversión. El intervalo de sexta mayor está comprendido en la clase interválica tercera menor. Y esto es porque no consideramos, de momento, la posición que tendrán las notas en el registro.

El vector interválico está representado por seis números entre corchetes cuyos órdenes de aparición hacen referencia a la clase interválica; [321000] indica un conjunto que posee tres segundas menores, dos segundas mayores, una tercera menor, o terceras mayores, o cuartas justas y/o cuartas aumentadas. Y el vector citado es el que corresponde a 0,1,2,3, o a 4,5,6,7, o a 0,B,A,9.

A 0,1,2,3 lo llamamos forma prima, que es la mínima expresión del conjunto, transpuesto sobre do.

En la mayoría de los casos, un conjunto posee 12 transposiciones y 12 inversiones de esas transposiciones. Sin embargo, al realizar estas operaciones de transformación partiendo de la forma prima notamos que los resultados en algunos casos se repiten. Por ejemplo el 4-28 (0,3,6,9), que es el acorde de séptima disminuida: la transposición a la tercera menor da 3,6,9,0 que es simplemente una permutación del anterior y, por lo tanto, es redundante.

Los 29 conjuntos de cuatro notas pueden observarse en la tabla siguiente. En la primera columna aparece el nombre que se da a cada conjunto, en la segunda su forma prima, en la tercera la cantidad de variantes por transposición e inversión posibles y, en la última, el vector interválico correspondiente.

Nombre del conjunto	Forma prima	Transp/Inv	Vector interválico
4-1	0123	12	[321000]
4-2	0124	24	[221100]
4-3	0134	12	[212100]
4-4	0125	24	[211110]
4-5	0126	24	[210111]
4-6	0127	12	[210021]
4-7	0145	12	[201210]
4-8	0156	12	[200121]
4-9	0167	6	[200022]
4-10	0235	12	[122010]
4-11	0135	24	[121110]
4-12	0236	24	[112101]
4-13	0136	24	[112011]
4-14	0237	24	[111120]
4-Z15	0146	24	[111111]
4-16	0157	24	[110121]
4-17	0347	12	[102210]
4-18	0147	24	[102111]
4-19	0148	24	[101310]
4-20	158	12	[101220]
4-21	0246	12	[030201]
4-22	0247	24	[021120]
4-23	0257	12	[021030]
4-24	0248	12	[020301]
4-25	0268	6	[020202]
4-26	0358	12	[012120]
4-27	0258	24	[012111]
4-28	0369	3	[004002]
4-Z29	0137	24	[111111]

Existen dos conjuntos que se distinguen por la Z en sus nombres (4-Z15 y 4-Z29). Ambos poseen el mismo vector interválico pero formas primas distintas. Según vimos, el vector [111111] indica que los dos conjuntos poseen todas las clases interválicas. Esta particularidad fue aprovechada en diversas obras del siglo XX, a modo de ejemplo, en "*Música para cuerdas, percusión y celesta*" de Bartók.

3. Matrices combinatorias no seriales

Las matrices no seriales presentan un mismo conjunto de grados cromáticos en todas sus filas y columnas, o bien, uno para sus filas y otro, distinto al anterior, en todas sus columnas. A partir de estas matrices se mantiene la coherencia interválica, tanto en el aspecto horizontal como en el vertical.

a	b	c	d
b	c	d	a
c	d	a	b
d	a	b	c

Realicemos ahora, a modo de ejemplo, una matriz con el conjunto 4-18 (0, 1, 4, 7) representándola numéricamente y luego en notación musical. Según se observa, contiene las mismas notas en todas sus filas y columnas.

4	1	0	7
7	4	1	0
0	7	4	1
1	0	7	4

El cuadrado romano puede contener más de un elemento por posición dentro de la matriz, o bien una posición vacía (un silencio). En el ejemplo siguiente partimos del conjunto 79A 03 6, que es un 6-27 dividido en tres partes, e intercalamos un silencio entre la segunda y la tercera partición.

3.1 Cuadrado romano

La matriz combinatoria no serial más simple, y a la vez más elemental en términos musicales, es la denominada *cuadrado romano*. Está constituida por un único conjunto que se repite sin transformaciones en sus filas y columnas. Se obtiene realizando permutaciones circulares de los elementos del conjunto dado, ubicando cada permutación en una fila. Si a, b, c y d son los elementos del conjunto, se construye del siguiente modo:

7A9	03		6
03		6	
	6	79A	03
6	79A	03	

En la fila 1, columna 1 del ejemplo anterior, permutamos los elementos de esa posición. Las permutaciones dentro de una misma posición no afectan las propiedades interválicas de la matriz.

Se denomina *norma horizontal* al conjunto comprendido por todos los elementos de cualquier fila, y *norma vertical* a todos los elementos de una columna. En un cuadrado romano, la norma horizontal y la vertical son idénticas.

3.2. Matrices simples de tipo I

Las matrices tipo I poseen también la misma norma, horizontal y verticalmente. La denominación simple indica que la construcción de la matriz no depende de las propiedades de la norma elegida, y cada posición está ocupada por un único elemento. Distinguimos dos casos.

3.2.1. Matriz tipo I A

La norma generadora se establece a partir de un conjunto cualquiera de n elementos. La cantidad de elementos determina las dimensiones de la matriz, que siempre es cuadrada (posee igual número de filas y columnas).

A fin de comprender el principio constructivo, consideremos un conjunto –norma generadora– formado por cuatro notas (a, b, c, d) que sumamos del siguiente modo:

a + a	*b + a*	*c + a*	*d + a*
a + b	*b + b*	*c + b*	*d + b*
a + c	*b + c*	*c + c*	*d + c*
a + d	*b + d*	*c + d*	*d + d*

Como el conjunto considerado posee cuatro elementos, la matriz generada es de 4 x 4.

Crearemos ahora una matriz a partir de las notas 2A740. Se trata de un conjunto 5-34, cuya estructura equivale a la de un acorde de dominante con novena mayor.

2 + 2	A + 2	7 + 2	4 + 2	0 + 2
2 + A	A + A	7 + A	4 + A	0 + A
2 + 7	A + 7	7 + 7	4 + 7	0 + 7
2 + 4	A + 4	7 + 4	4 + 4	0 + 4
2 + 0	A + 0	7 + 0	4 + 0	0 + 0

Efectuando la suma y ajustando los valores entre 0 y 11 (módulo 12) obtenemos:

4	0	9	6	2
0	8	5	2	A
9	5	2	B	7
6	2	B	8	4
2	A	7	4	0

Las filas y columnas de la matriz 5 x 5 están formadas por transposiciones del conjunto empleado como norma.

3.2.2. Matriz tipo I B

Esta matriz se construye de modo similar a la anterior.

El tipo I A posee en sus filas y columnas transposiciones del conjunto usado como norma, pero aquí generamos transposiciones en las filas, e inversiones transpuestas en las columnas.

El procedimiento para construir la matriz recurre también a la suma, pero en este caso, el primer término está dado por cada uno de los elementos del conjunto, y el segundo por la inversión de los mismos.

$a + I(a)$	$b + I(a)$	$c + I(a)$	$d + I(a)$
$a + I(b)$	$b + I(b)$	$c + I(b)$	$d + I(b)$
$a + I(c)$	$b + I(c)$	$c + I(c)$	$d + I(c)$
$a + I(d)$	$b + I(d)$	$c + I(d)$	$d + I(d)$

Para ejemplificar esto, utilicemos ahora el conjunto 4-27 como norma (2085).

2 + I(2)	0 + I(2)	8 + I(2)	5 + I(2)
2 + I(0)	0 + I(0)	8 + I(0)	5 + I(0)
2 + I(8)	0 + I(8)	8 + I(8)	5 + I(8)
2 + I(5)	0 + I(5)	8 + I(5)	5 + I(5)

Que da por resultado:

0	A	6	3
2	0	8	5
6	4	0	9
9	7	3	0

Cada fila presenta un conjunto 4-27, cuya estructura es la de un acorde de séptima de sensible. Por otra parte, cada columna contiene también un 4-27 pero invertido, que corresponde a la estructura del acorde de séptima de dominante.

3.2.3. Matriz tipo II

En una matriz simple tipo II, la norma horizontal es diferente a la norma vertical. Los conjuntos que la generan pueden tener, además, distinta cantidad de elementos, lo que produce una matriz rectangular. Si los elementos de la norma generadora horizontal son a_h, b_h, c_h, d_h , y los de la vertical a_v, b_v, c_v, d_v,e_v, el principio constructivo es el siguiente:

$a_h + a_v$	$b_h + a_v$	$c_h + a_v$	$d_h + a_v$
$a_h + b_v$	$b_h + b_v$	$c_h + b_v$	$d_h + b_v$
$a_h + c_v$	$b_h + c_v$	$c_h + c_v$	$d_h + c_v$
$a_h + d_v$	$b_h + d_v$	$c_h + d_v$	$d_h + d_v$
$a_h + e_v$	$b_h + e_v$	$c_h + e_v$	$d_h + e_v$

Vemos que resulta una matriz rectangular de 4 x 5.

3.3. Matrices generadas a partir de cadenas de conjuntos

Estas matrices se denominan complejas, ya que su construcción depende de las propiedades del conjunto utilizado como norma.

Una cadena se construye a partir de las particiones de un conjunto. Consideremos el PCS 5-15 = {0, 1, 2, 6, 8}, y dos de sus particiones en particular, que contienen un subconjunto en común (3-8):

A) 01|268 PCS : 2-1 | 3-8
B) 16|028 PCS : 2-5 | 3-8

Partimos de A, y luego aplicamos una transformación al conjunto B con un operador T_n ó T_nI que convierta al término 028 en 068 (segundo término de A). Si transformamos la retrogradación de B con T_6 y luego ordenamos, el resultado es el siguiente:

268|07

Obtuvimos así dos eslabones de una cadena, A = 01|268 y $T_6R(B)$ = 268|07. Ahora haremos lo mismo sobre el original B, con la intención de lograr una nueva transformación que mantenga invariante al subconjunto 07. Aplicando T_1I nos queda:

07|15B

Continuando con este procedimiento, en un recorrido de ida y vuelta alrededor del subconjunto común a ambas particiones, arribamos al siguiente resultado:

A B
2-1 ◄► 3-8 ◄► 2-5

01	268	A
268	07	$T_6R(B)$
07	15B	$T_1I(B)$
15B	67	$T_7IR(A)$
67	028	$T_6(A)$
028	16	$T_0R(B)$
16	57B	$T_7I(B)$
57B	01	$T_1I(A)$

A partir de estos datos, estamos en condiciones de armar la cadena, la cual es cerrada, pues el primer subconjunto y el último son idénticos.

01|268|07|15B|67|028|16|57B|01

Uniendo dos subconjuntos consecutivos de la cadena

obtenemos el PCS utilizado como norma (5-15). Podríamos utilizar esta sucesión para la composición de un fragmento melódico con un alto grado de coherencia interválica, ya que cada enlace de conjuntos se realiza mediante dos o tres grados en común.

La generación de la matriz la realizamos ubicando a cada subconjunto sobre la diagonal, de modo tal que la norma se manifieste en cada una de las filas y de las columnas.

01	268		
	07	15B	
		67	028
57B			16

La construcción de matrices utilizando este método requiere de un análisis detallado de los PCS, en particular en relación con sus posibilidades de aplicación en la generación de cadenas de un número considerable de eslabones.

Se denomina *partición redundante* a aquella que transformada resulta invariante de otra partición del mismo PCS. La partición 04A|123B, por ejemplo, transformada con T_2I produce una invariancia total (24A|103B). A una partición redundante se la identifica con un asterisco, y es de poca utilidad en la producción de matrices combinatorias.

Algo similar ocurre con 012|34AB bajo T_2I (210|BA43), con la diferencia que en este caso la partición sigue siendo la misma. Estas observaciones dan la pauta de que no todas las divisiones del conjunto producen la variedad necesaria para la construcción eficaz de una matriz.

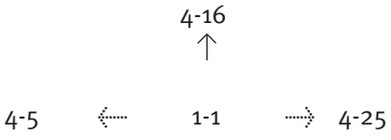
La invariancia parcial es la que resulta efectiva, y se manifiesta a través de las condiciones presentadas en la tabla que sigue (a la derecha, se detalla el modo en que se identifican). Las letras X e Y representan a cada uno de los términos de la partición, y F y G son operaciones de transformación aplicadas a la partición considerada.

Propiedad	Etiqueta
F(X) = X	1
F(X) = X G(Y) = Y	12
G(Y) = Y	2
F(X) = Y F(Y) ≠ X	3

Un análisis de todas las particiones posibles del PCS 5-15 = {0, 1, 2, 6, 8} aporta la siguiente información:

0|1268	1-1	4-16	1
1|0268	1-1	4-25	2
2|0168	1-1	4-16	*
6|0128	1-1	4-5	1
8|0126	1-1	4-5	*
01|268	2-1	3-8	1
02|168	2-2	3-9	
06|128	2-6	3-5	1
08|126	2-4	3-4	1
12|068	2-1	3-8	*
16|028	2-5	3-8	1
18|026	2-5	3-8	*
26|018	2-4	3-4	*
28|016	2-6	3-5	*
68|012	2-2	3-1	

Las cadenas no solo pueden generarse sobre la base de dos particiones, sino a tres o más que posean un subconjunto en común. En consecuencia, las posibilidades de construcción son muy variadas. El siguiente gráfico muestra una posible combinación de tres particiones a partir del subconjunto común 1-1.



3.4. Matrices a partir de ciclos de un mismo operador

Partiendo de un grado cualquiera, podemos establecer una secuencia en la cual cada término es el resultado de una transformación del término anterior utilizando un operador determinado. La secuencia 0, 2, 4, 6, 8, A, por ejemplo, surge de la aplicación cíclica del operador T_2.

Con algunos operadores, la secuencia se limita a dos términos, ya que el tercero es idéntico al primero y la secuencia se repite. Esto ocurre con T_6 y T_nI (donde *n* es un entero entre 0 y 11).

0	1	2	3	4	5	6	7	8	9	A	B	
6	7	8	9	A	B	0	1	2	3	4	5	T_6

0	1	2	3	4	5	6	7	8	9	A	B	
5	4	3	2	1	0	B	A	9	8	7	6	T_5 I

Los operadores T_4 y T_8, por otra parte, generan secuencias de tres términos.

0	1	2	3	4	5	6	7	8	9	A	B	
4	5	6	7	8	9	A	B	0	1	2	3	T_4
8	9	A	B	0	1	2	3	4	5	6	7	T_4 (T_4)

Los operadores T_3 y T_9 generan secuencias de cuatro términos.

0	1	2	3	4	5	6	7	8	9	A	B	
3	4	5	6	7	8	9	A	B	0	1	2	T_3
6	7	8	9	A	B	0	1	2	3	4	5	T_3 (T_3)
9	A	B	0	1	2	3	4	5	6	7	8	T_3 (T_3 (T_3))

Y, por último, los operadores T_2 y T_A producen ciclos de seis términos (0, 2, 4, 6, 8, A; 1, 3, 5, 7, 9, B; etc.).

Una matriz, utilizando ciclos de operaciones, se construye ubicando el PCS utilizado como norma en P1-1, y los términos del ciclo en las posiciones de la diagonal de la matriz (P2-2, P3-3). Luego utilizaremos procedimientos de transformación para enriquecer el resultado.

1064			
	3479		
		067A	
			139A

La modificación de 1, 0, 6, 4 por T_3 da por resultado 4, 3, 9, 7. No obstante, vemos en la posición P2-2 de la matriz el conjunto 3, 4, 7, 9, ordenado de menor a mayor por cuestiones de claridad. Ya no estamos trabajando con conjuntos ordenados, como en el caso de las series dodecafónicas, por lo cual la permutación de los elementos de una misma posición de la matriz no afecta a los resultados.

También es posible ubicar el conjunto de partida en posiciones diferentes y desarrollar el ciclo siguiendo recorridos sobre la diagonal y paralelos a ella. Se debe observar aquí como se rebaten los elementos sobre las posiciones. El operador empleado es T_2.

4	B	0	7		
	6	1	2	9	
		8	3	4	B
1			A	5	6
8	3			0	7
9	A	5			2

Las dimensiones de la matriz dependen de la cantidad de términos del ciclo y, por lo tanto, del operador utilizado. La siguiente tabla lo ilustra.

Operador	Dimensiones de la matriz
T_6 - T_n I	2 x 2
$T_4 - T_8$	3 x 3
$T_3 - T_9$	4 x 4
$T_2 - TA$	6 x 6

3.5. Operaciones sobre matrices combinatorias no seriales
3.5.1. Intercambio de elementos entre posiciones

Esta operación sobre matrices combinatorias seriales la realizábamos exclusivamente entre columnas adyacentes, debido a la necesidad de mantener el orden de los conjuntos distribuidos en cada posición de las filas. En el caso de las matrices no seriales, por tratarse de conjuntos no ordenados, el intercambio se realiza tanto entre filas como entre columnas, sin que importe si son adyacentes o no, lo cual produce una variedad considerable de versiones. Para ejemplificar esto, vamos a partir de una matriz generada por ciclos y a producir todos los intercambios posibles, a fin de lograr una distribución uniforme de elementos.

Matriz por ciclos
Conjunto de partida: A 32 87 (PCS: 5-20)
Operador: TA
Norma horizontal: 5-20
Norma vertical: 5-7

A	32	87			
	8	01	56		
		6	AB	34	
			4	89	12
0B				2	67
45	9A				0

Vamos a intercambiar el elemento A de P_{1-1} por su par de P_{3-4}. El primero baja dos filas sobre su misma columna (a P_{3-1}), y el segundo asciende dos filas, también conservando su propia columna (a P_{1-4}). El resultado no se altera, ni vertical ni horizontalmente.

	32	87	A		
	8	01	56		
A		6	B	34	
			4	89	12
0B				2	67
45	9A				0

Ahora intercambiamos los elementos 3 y 2 de P_{1-2} por sus pares de las posiciones P_{3-5} y $P_{4-}6$. El primer 3 desciende a la tercera fila y su par asciende a la primera. El primer 2 baja a la cuarta, y el 2 de la última columna sube a la primera fila.

		87	A	3	2
	8	01	56		
A	3	6	B	4	
	2		4	89	1
0B				2	67
45	9A				0

Continuando con el procedimiento, logramos una distribución uniforme.

	A	7		38	2
5	8	1			06
A	3	6	B	4	
	2	8	4	9	1
B		0	6	2	7
04	9		5A		

En el próximo ejemplo, creamos una matriz a partir de cadenas, y luego efectuamos la redistribución de sus elementos.

Matriz por cadenas
Conjunto de partida: 14376A (PCS: 6-27)
Particiones utilizadas: 14|376A (2-3|4-17) y 1A|4376 (2-3|4-3)
Norma horizontal: 6-27
Norma vertical: 6-27

376A	14						
	679A	03					
		169A	03				
			679A	14			
				27AB	14		
					679A	03	
						2569	03
14							679A

4		1	6	A	7		3
	A	0	3	7	9		6
A		6	9		1	3	0
	4		7	1	6	9	A
7	1	A		B	4	2	
6	7	3			A	0	9
3	6	9	0	2		5	
1	9		A	4		6	7

3.5.2. Transformación por *Tn* , *TnI* o *M*

La matriz completa puede ser transpuesta, invertida o multiplicada. Veamos la matriz anterior modificada por T_6I.

2		5	0	8	B		3
	8	6	3	B	9		0
8		0	9		5	3	6
	2		B	5	0	9	8
B	5	8		7	2	4	
0	B	3			8	6	9
3	0	9	6	4		1	
5	9		8	2		0	B

3.5.3. Intercambio entre filas y columnas

Una vez más, por tratarse de conjuntos no ordenados, es posible intercambiar columnas entre sí, o filas. En el ejemplo siguiente generamos una matriz Tipo II e intercambiamos las filas 1 y 3, y luego las columnas 2 y 4.

Matriz Tipo II
Conjuntos de partida: 14376A (PCS: 6-27) y 980 (PCS: 3-3)
Norma horizontal: 6-27
Norma vertical: 3-3

A	1	0	4	3	7
9	0	B	3	2	6
1	4	3	7	6	A

1	7	3	4	6	A
9	3	B	0	2	6
A	4	0	1	3	7

3.5.4. Giro de 90°

A través de un giro de 90° la matriz intercambia filas por columnas. La primera fila se convierte en la primera columna, la segunda fila en la segunda columna y así siguiendo. Para el ejemplo creamos una matriz Tipo I B y la rotamos 90°.

Matriz Tipo I B
Conjunto de partida: 26598 (PCS: 5-16)
Norma horizontal: 5-16
Norma vertical: 5-16

0	4	3	7	6
8	0	B	3	2
9	1	0	4	3
5	9	8	0	B
6	A	9	1	0

6	2	3	B	0
7	3	4	0	1
3	B	0	8	9
4	0	1	9	A
0	8	9	5	6

3.5.5. Invariancia en posiciones, filas o columnas

Es posible aplicar una transformación T_n, T_nI ó M que mantenga invariantes a un elemento en particular o, a un conjunto dispuesto en una fila o columna específica. La siguiente matriz se transforma con T_8I, y mantiene invariantes las filas 1 y 4, y las columnas 1 y 2.

Matriz por cadena
Conjunto de partida: 71084 (PCS: 5-22)
Particiones utilizadas: 71|084 (2-6|3-12) con sí misma
Norma horizontal: 5-22
Norma vertical: 5-22

48	17		0
0	8	5B	4
		26A	5B
17	04		8

40	17		8
8	0	31	4
		26A	39
17	48		0

3.5.6. Combinación de matrices no seriales

Al igual que las seriales, estas matrices pueden combinarse con el propósito de generar otras de mayores dimensiones. La condición necesaria para que la combinación

resulte es que las normas de ambas sean idénticas. Vemos, en el próximo ejemplo, el modo en que se disponen las matrices, y a continuación el resultado que se obtiene luego de la aplicación de operaciones de intercambio de elementos.

1	367	42			
03	4	1AB			
456	89	3			
			7	46	125
			58	79A	4
			234	5	0B

6	3		7	12	4
3		1B	4	0	A
4	68	3	5		9
1	7	4	2	5	6
5	9	A	8	4	7
0	4	2	3	B	5

4. Aplicaciones musicales

A fin de ejemplificar la utilización de matrices combinatorias no seriales en la composición musical veremos algunos ejercicios, escritos de acuerdo con las técnicas de generación y transformación anteriormente analizadas.

Para comenzar, construiremos una matriz a partir del método denominado *cuadrado romano*, partiendo del conjunto {8, 9, 4, 0, 6} que conforma un PCS 5-26. Obtenemos la siguiente matriz que, según sabemos, posee en sus filas permutaciones circulares del conjunto dado.

8	9	4	0	6
9	4	0	6	8
4	0	6	8	9
0	6	8	9	4
6	8	9	4	0

A partir de estos datos, disponemos la primera, segunda y quinta columnas de la matriz en forma de acordes, y la tercera y cuarta en forma de arpegios. Se observa en el ejemplo el desarrollo del contenido de las filas, particularmente en el movimiento de los bajos (quinta fila de la matriz).

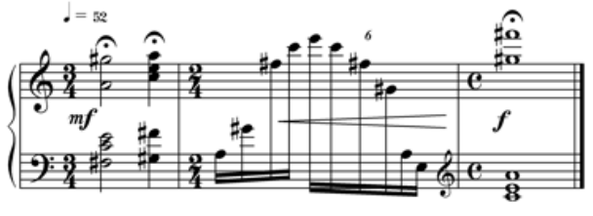


Figura 1: fragmento musical que emplea el cuadrado romano

Continuando con la misma técnica, construiremos una nueva matriz, pero que en este caso contenga distinta cantidad de elementos por posición. Utilizaremos el conjunto {B, 9, 1, 5, 2, 8} (PCS 6-Z28).

Distribuimos el conjunto de partida en tres grupos de dos, tres y un elemento respectivamente: B9, 152 y 8. Obtenemos así una matriz de 3 x 3.

B9	152	8
152	8	B9
8	B9	152

La primera columna ocupa los dos primeros tiempos del ejemplo. Vemos dos notas en el agudo (*si* y *la*), tres en el registro intermedio (*do#* como apoyatura, *re* y *fa*), y una sola nota en el grave (*sol#*). Esta última nota ocupa la diagonal de la matriz, y es el único elemento en la posición. Aparece en los tres registros, y siempre es acompañada por dos y tres sonidos en las otras voces.

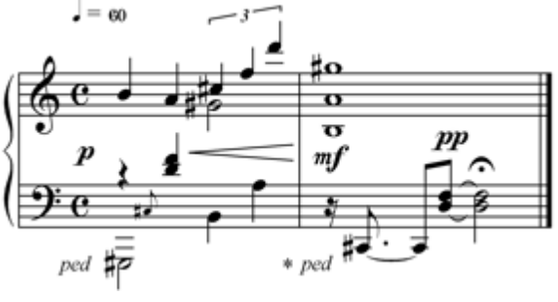


Figura 2: fragmento musical que emplea el cuadrado romano con más de un elemento por posición

Para la realización del ejemplo que sigue nos interesa particularmente comenzar con los sonidos {0, 8, 2, 7, B}, dispuestos del grave al agudo en forma de acorde, y aplicar luego la técnica de generación tipo I A. Analizamos el conjunto y vemos que se trata de un PCS 5-Z18.

Construimos la matriz según el método estudiado, partiendo del conjunto elegido. Nos queda:

0	8	2	7	B
8	4	A	3	7
2	A	4	9	1
7	3	9	2	6
B	7	1	6	A

Pero observamos que la primera columna no contiene el acorde de partida en el orden deseado, sino una retrogradación del mismo, por lo cual vamos a permutar las filas 1 y 5, y luego la 2 y con la 4.

B	7	1	6	A
7	3	9	2	6
2	A	4	9	1
8	4	A	3	7
0	8	2	7	B

Si la realización musical consiste en lograr una sucesión de acordes con matices contrastantes podemos escribir:

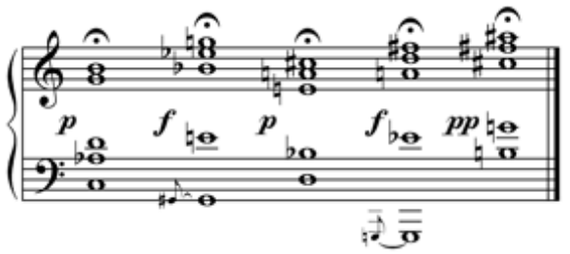


Figura 3: fragmento musical que emplea una matriz tipo I A

En el ejemplo siguiente vamos a alternar las columnas de dos matrices distintas. La idea es alcanzar el total cromático a partir de la alternancia de dos hexacordios 6-35. El primero contiene a los grados 0, 2, 4, 6, 8, A, y el segundo a los grados 1, 3, 5, 7, 9, B.

Para comenzar, generamos una matriz tipo IA a partir del conjunto {0, 2, 4, 6, 8, A}.

0	2	4	6	8	A
2	4	6	8	A	0
4	6	8	A	0	2
6	8	A	0	2	4
8	A	0	2	4	6
A	0	2	4	6	8

Vemos que el resultado es el mismo que si la hubiéramos construido siguiendo la técnica del cuadrado romano.

Ahora construimos otra a partir del conjunto {1, 5, 3, B, 7, 9}

2	6	4	0	8	A
6	A	8	4	0	2
4	8	6	2	A	0
0	4	2	A	6	8
8	0	A	6	2	4
A	2	0	8	4	6

Y observamos que a pesar de haber usado los grados impares (1, 3, 5, etc.) el método constructivo nos lleva a una matriz de grados pares, dado que la suma de dos números impares es un número par. Para revertir esta situación vamos a transportar la matriz en un semitono.

3	7	5	1	9	B
7	B	9	5	1	3
5	9	7	3	B	1
1	5	3	B	7	9
9	1	B	7	3	5
B	3	1	9	5	7

Generadas ambas matrices alternamos sus columnas a fin de obtener el total cromático a partir de esa combinación. Esta alternancia nos lleva a pensar distintas realizaciones compositivas. Podría ocurrir, por ejemplo, que los grados pares se diferenciaran de los impares a partir de la utilización de distintos matices, articulaciones, timbres, ubicación en el registro, o tipo de textura. Esto generaría dos discursos paralelos, ambos con sentido propio, pero que conforman una totalidad.

0	3	2	7	4	5	6	1	8	9	A	B
2	7	4	B	6	9	8	5	A	1	0	3
4	5	6	9	8	7	A	3	0	B	2	1
6	1	8	5	A	3	0	B	2	7	4	9
8	9	A	1	0	B	2	7	4	3	6	5
A	B	0	3	2	1	4	9	6	5	8	7

Esta matriz, producto de la combinación de las anteriores, sirve a la composición del ejemplo siguiente:

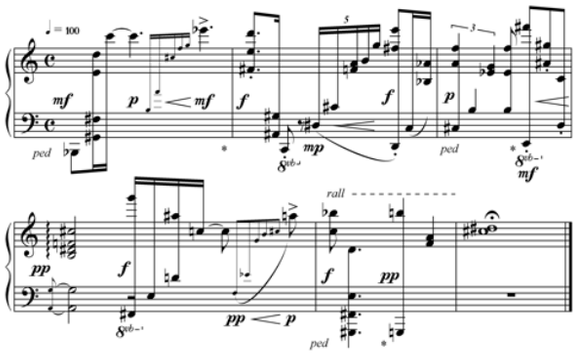


Figura 4: fragmento musical que emplea dos matrices tipo I A alternadas

Según vimos, la matriz tipo I B posee transposiciones de un conjunto en sus filas, y transposiciones de la inversión del mismo conjunto en sus columnas. El ejercicio siguiente persigue la composición de un breve proceso gradual, utilizando este tipo de estructuras.

Fijamos como punto de partida el conjunto {4, 0, 3, 8, 6} (PCS 5-26) y construimos la matriz siguiendo el principio antes visto. Se observa que, debido al método, la primera posición de las matrices tipo I B arroja un 0, debido a que la suma de un grado cualquiera y su inversión módulo 12 es siempre 0.

0	8	B	4	2
4	0	3	8	6
1	9	0	5	3
8	4	7	0	A
A	6	9	2	0

Para evitar esa situación transportaremos la matriz

utilizando un operador arbitrario, por ejemplo, T_4.

4	0	3	8	6
8	4	7	0	A
5	1	4	9	7
0	8	B	4	2
2	A	1	6	4

Las otras cuatro matrices a emplear las obtenemos transportando de a un semitono la matriz anterior.

5	1	4	9	7
9	5	8	1	B
6	2	5	A	8
1	9	0	5	3
3	B	2	7	5

6	2	5	A	8
A	6	9	2	0
7	3	6	B	9
2	A	1	6	4
4	0	3	8	6

7	3	6	B	9
B	7	A	3	1
8	4	7	0	A
3	B	2	7	5
5	1	4	9	7

8	4	7	0	A
0	8	B	4	2
9	5	8	1	B
4	0	3	8	6
6	2	5	A	8

El material así obtenido se aplica en la producción del siguiente fragmento musical.

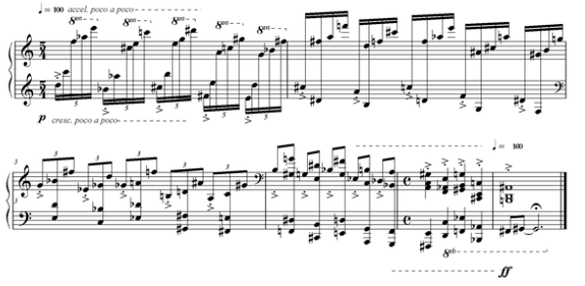


Figura 5: fragmento musical que emplea matrices tipo I B

El próximo ejemplo se basa en una matriz tipo II, que según vimos puede armarse sobre dos conjuntos de distinta cantidad de elementos. En este caso, utilizamos como norma horizontal al conjunto {9, 1, 0, 4, 3, 7, 6} (PCS 7-31), y como norma vertical al conjunto {8, 6, 0, 9} (PCS 4-12).

5	9	8	0	B	3	2
3	7	6	A	9	1	0
9	1	0	4	3	7	6
6	A	9	1	0	4	3

Procedemos luego a rotar 90° la matriz anterior para combinarla con una versión de mayor densidad polifónica.

2	0	6	3
3	1	7	4
B	9	3	0
0	A	4	1
8	6	0	9
9	7	1	A
5	3	9	6

Por último, intercambiamos columnas entre sí, en favor de la sonoridad buscada.

0	3	2	6
1	4	3	7
9	0	B	3
A	1	0	4
6	9	8	0
7	A	9	1
3	6	5	9

Y una aplicación posible es:

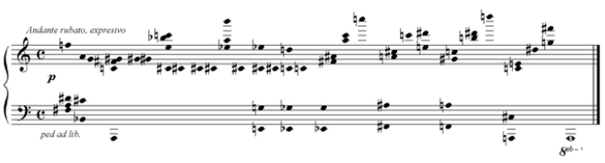


Figura 6: fragmento musical que emplea una matriz tipo II

Para la construcción del siguiente ejemplo utilizamos la técnica de matrices por *ciclos de un operador*. El conjunto elegido es {2, 1, 5, 4} (PCS 4-3), y el operador es T_3.

2154			
	4578		
		78AB	
			12AB

Paso seguido, realizamos intercambios entre elementos de la matriz, de forma gradual.

154			2
	4578		
		78AB	
2			1AB

15	4		2
4	578		
		78AB	
2			1AB

1	45		2
45	78		
		78AB	
2			1AB

1	45		2
45	8	7	
	7	8AB	
2			1AB

1	45		2
45	8	7	
	7	8B	A
2		A	1B

1	45		2
45	8	7	
	7	8	AB
2		AB	1

La aplicación en orden sucesivo de las matrices anteriores da por resultado:

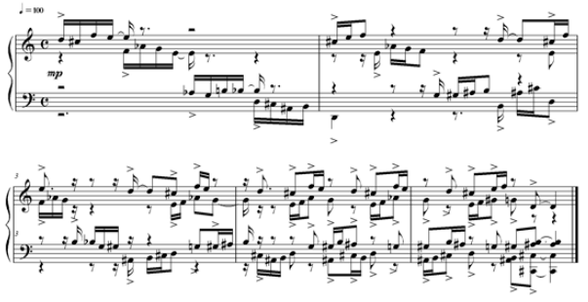


Figura 7: fragmento musical que emplea una matriz generada por ciclos de un operador

Como último ejemplo, partiremos de una matriz generada con cadenas de PCS, empleando {0, 3, 6, A, 2} (PCS 5-26).

36A2	0		
	379B	6	
		0489	6
0			379B

La idea es contraponer la matriz anterior con otra versión cuyos elementos estén distribuidos de manera más uniforme, aplicando operaciones de intercambio de elementos.

2A	3	0	6
3	9	6	7B
6	0	48	9
0	7B	9	3

Y una realización posible es la que sigue:

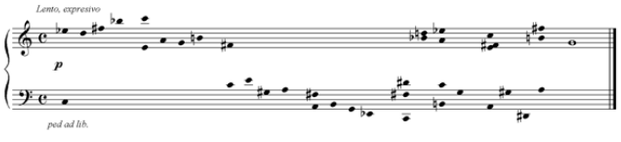


Figura 8: fragmento musical que emplea una matriz generada a partir de cadenas

5. Desarrollo de software

Se creó la Biblioteca de Objetos Externos *Pcslib* (para utilizarse en conjunto con el programa Pure Data, de Miller Puckette), cuyas particularidades se detallan a continuación.

5.1. Objetos para Matrices Combinatorias (CM)

5.2. Generador de CM

Nombre	**cm_roman**
Descripción	El objeto *cm_roman* crea una matriz del tipo cuadrado romano (véase Morris, 1984, 1987).
Entradas	*Inlet*: un puntero a una estructura PCS.
Salidas	*Outlet*: un puntero a una estructura CM. Para acceder a los datos, debe usarse alguno de los objetos lectores de CM.

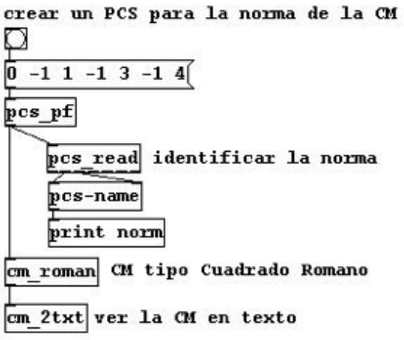


Figura 9: *patch* de pd como ejemplo de uso del objeto *cm_roman*

Nombre	**cm_t1a**
Descripción	El objeto *cm_t1a* crea una matriz del tipo I A (véase Morris, 1984, 1987).
Entradas	*Inlet*: un puntero a una estructura PCS.
Salidas	*Outlet*: un puntero a una estructura CM. Para acceder a los datos, debe usarse alguno de los objetos lectores de CM.

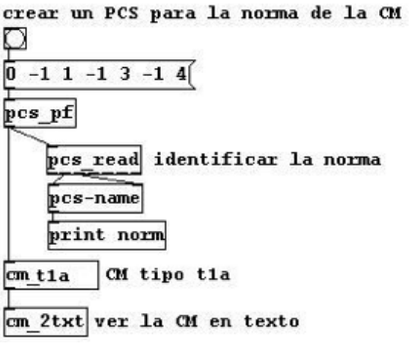


Figura 10: *patch de pd* como ejemplo de uso del objeto *cm_t1a*

Nombre	**cm_t1b**
Descripción	El objeto *cm_t1b* crea una matriz del tipo I B (véase Morris, 1984, 1987).
Entradas	*Inlet*: un puntero a una estructura PCS.
Salidas	*Outlet*: un puntero a una estructura CM. Para acceder a los datos, debe usarse alguno de los objetos lectores de CM.

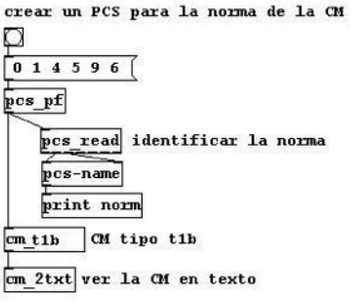


Figura 11: *patch* de *pd* como ejemplo de uso del objeto *cm_t1b*

Nombre	**cm_t2**
Descripción	El objeto *cm_t2* crea una matriz del tipo II (véase Morris, 1984, 1987).
Entradas	*Inlets 1 y 2*: punteros a dos estructuras PCS para las normas vertical y horizontal respectivamente.
Salidas	*Outlet*: un puntero a una estructura CM. Para acceder a los datos, debe usarse alguno de los objetos lectores de CM.

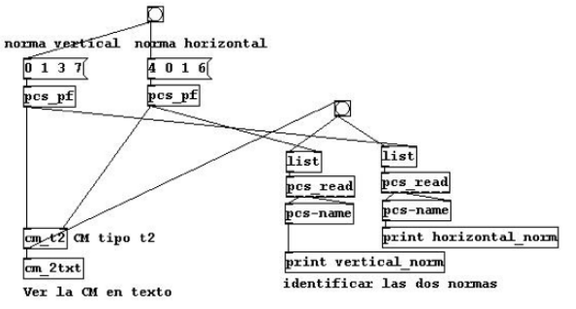


Figura 12: patch de pd como ejemplo de uso del objeto cm_t2

Nombre	**cm_opcy**
Descripción	El objeto *cm_opcy* crea una matriz con el método de ciclos de operadores (véase Morris, 1984, 1987).
Entradas	*Inlet 1*: un símbolo que indica el operador de ciclos a utilizar. Debe ser uno de los siguientes: T2/A, T6TI,T3/9 o T4/8. *Inlet 2*: un puntero a una estructura PCS.
Salidas	*Outlet*: un puntero a una estructura CM. Para acceder a los datos, debe usarse alguno de los objetos lectores de CM.

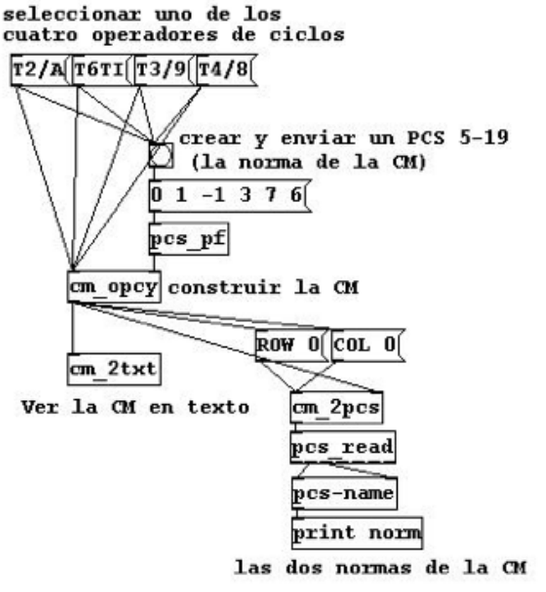


Figura 13: *patch* de *pd* como ejemplo de uso del objeto *cm_opcy*

5.3. Lectores de CM

Nombre	**cm_2txt**
Descripción	El objeto *cm_2txt* imprime en la pantalla de *pd* una versión "limpia" (i.e., no "cruda") de una estructura CM. Los enteros 10 y 11 son reemplazados por *A* y *B*, respectivamente.
Entradas	*Inlet*: un puntero a una estructura CM que puede ser creada con los objetos generadores de CM precedentes. Ver los ejemplos en las imágenes 11, 12, 13, 14 y 15, que usan a los objetos creadores de CM en conjunto con el objeto *cm_2txt*.
Salidas	No posee.
Nombre	**cm_read**
Descripción	El objeto *cm_read* obtiene los datos "crudos" de una estructura CM. Un valor de -3 o de -4 indican espacios o posiciones vacías, respectivamente. Si se desea sólo ver una versión "limpia" de la CM, usar el objeto *cm_2txt*.
Entradas	*Inlet*: un puntero a una estructura CM que puede ser creada con los objetos generadores de CM precedentes.
Salidas	*Outlet 1*: una sucesión de listas de *floats* siendo cada una de ellas una fila de la CM. *Outlet 2*: número de filas de la CM (*float*). *Outlet 3*: número de columnas de la CM (*float*). *Outlet 4*: número máximo de PC para cada posición de la CM (*float*).
Nombre	**cm_2pcs**
Descripción	El objeto *cm_2pcs* extrae un PCS de una CM.
Entradas	*Inlet 1*: una de las siguientes posibilidades: POS (*symbol*) r (*float*) c (*float*): Obtiene el PCS en la posición r (fila) c (columna). ROW (*symbol*) r (*float*): Obtiene el PCS integrado por todos los PC en las posiciones de r (fila). COL (*symbol*) c (*float*): Obtiene el PCS integrado por todos los PC en las posiciones de c (columna). ALL (*symbol*): Obtiene el PCS integrado por todos los PC en todas las posiciones de la CM. *Inlet 2*: un puntero a una estructura CM, que puede ser creada con los objetos generadores de CM precedentes.
Salidas	*Outlet*: un puntero a la estructura PCS resultante. Para acceder a los datos, debe usarse otro objeto *pcs* (usualmente, un objeto *pcs_read*).

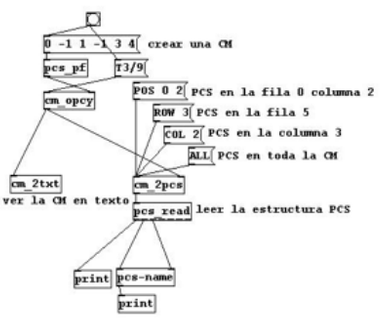


Figura 14: *patch* de *pd* como ejemplo de uso del objeto *cm_2pcs*

5.4. Modificadores de CM

Nombre	**cm_trans**
Descripción	El objeto *cm_trans* realiza varias transformaciones de una CM.
Entradas	*Inlet 1*: una de las siguientes posibilidades: T (*symbol*) n (*float*): transposición por *n*. I (*symbol*) n (*float*): inversión seguida de transposición por *n* (*n* = 0 es obligatorio en este caso si solo se requiere inversión). RD (*symbol*) n (*float*): rotación por la diagonal, *n* = 0 significa izquierda superior, *n* = 1 significa derecha inferior. R90 (*symbol*): rotación de 90 grados. ER (*symbol*) r1 (float) r2 (float): intercambiar el contenido de las filas r1 y r2. EC (*symbol*) c1 (float) c2 (float): intercambiar el contenido de las columnas c1 y c2. SWAP: intentar disminuir la densidad de las posiciones de la CM intercambiando PC en sus filas y/o columnas. Esta transformación puede fallar si la densidad no puede ser reducida por intercambio. En este caso no se generará salida y se enviará un mensaje de advertencia a la pantalla de *pd*. *Inlet 2*: un puntero a una estructura CM que puede ser creada con los objetos generadores de CM precedentes.
Salidas	*Outlet*: un puntero a una estructura CM. Para acceder a los datos, debe usarse alguno de los objetos lectores de CM.

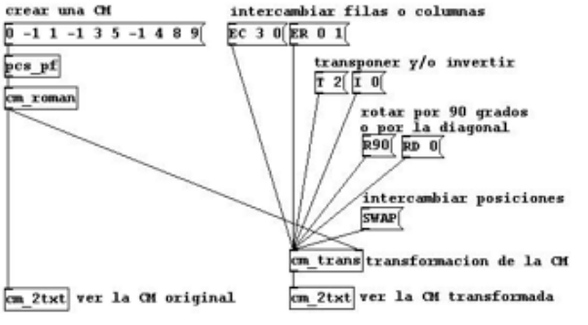
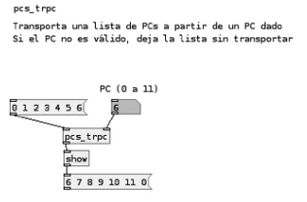
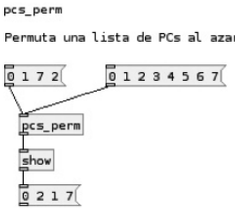
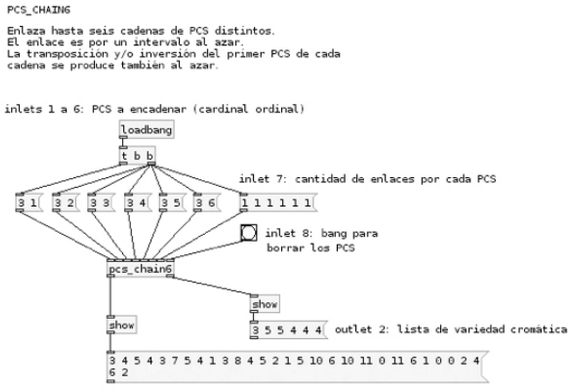


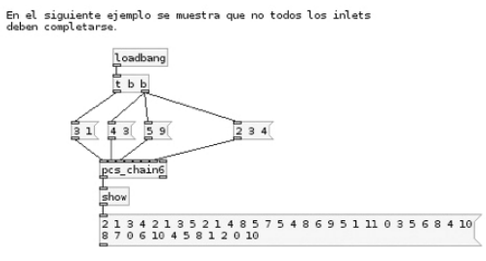
Figura 15: *patch* de *pd* como ejemplo de uso del objeto *cm_trans*

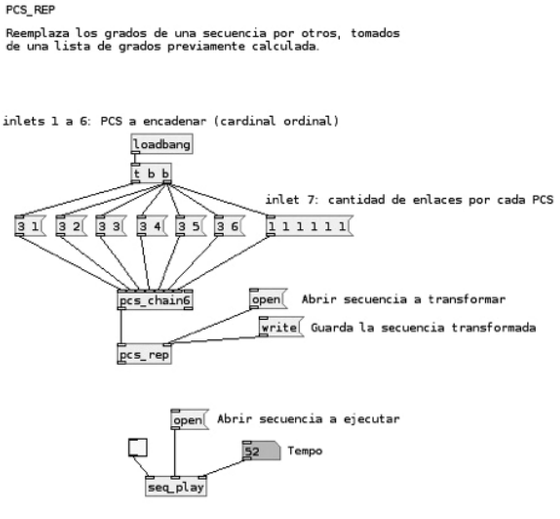
A partir de estos objetos, y otros relacionados con el tratamiento de los conjuntos cromáticos, se desarrollaron diversas abstracciones destinadas a la implementación de una plataforma destinada a la composición musical. Entre ellos los siguientes:



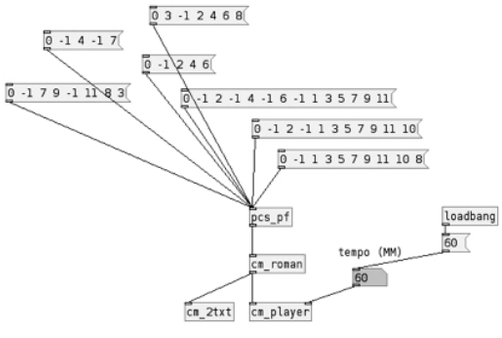








Ejecución de matrices combinatorias:



Referencias bibliográficas

Manovich, Lev. *The Language of New Media* [Bibliografía], (Estados Unidos: MIT-Press, 1998)

Volpe, Gualterio. *Computational models of expressive gesture in multimedia systems* [Bibliografía], (Italia: InfoMus Lab.-Universidad de Génova, 2003).

<http://www.ekac.org, consultada el 25/Oct/2005>, actualmente en línea/fuera de línea.

PABLO DI LISCIA

Doctor en Humanidades y Artes, Mención Música en la UNR. Estudió, además, composición en forma particular con los maestros Dante Grela y Francisco Kröpfl. Fué Director de la Carrera de Composición con Medios Electroacústicos en la UNQ y Secretario de Investigación y Posgrado del IUNA. Actualmente es Profesor Titular y Director de la Colección “Música y Ciencia” de la UNQ y Profesor Titular del IUNA.

Ha recibido subsidios del Fondo Nacional de Las Artes, la Fundación Antorchas, la Fundación Rockefeller y la Fundación Música y Tecnología, y premios en competiciones nacionales e internacionales (Fondo Nacional de las Artes, Secretaría de Cultura de la Nación, Concurso Internacional de Bourges, Francia, etc.) Su música se ha difundido tanto en el país como en el exterior, en USA, Francia, Chile, Cuba, España, Holanda, etc. Ha publicado artículos sobre estética y técnica de la música y las nuevas tecnologías, y desarrollado software para proceso de sonido y música, análisis musical y composición.

Más información en: http://musica.unq.edu.ar/personales/odiliscia

Medidas de similitud entre sucesiones ordenadas de grados cromáticos

Pablo Di Liscia

Introducción

Este trabajo presenta una parte del proyecto de investigación: *Desarrollo de programas informáticos de asistencia a la composición musical*, desarrollado por el Dr. Pablo Di Liscia durante una estadía posdoctoral en el *Music Technology Group* de la Universidad Pompeu Fabra (Barcelona, España) durante el 2009 y financiado a través de una Beca de la Fundación Carolina y la UNQ.

El proyecto realizado constituye un módulo del proyecto de Investigación *Aplicaciones musicales de conjuntos y matrices combinatorias de grados cromáticos*, radicado en la UNQ para el bienio 2009-2010 y dirigido por el Dr. Pablo Di Liscia. A su vez, la librería PCSLIB desarrollada en este proyecto por Pablo Di Liscia y Pablo Cetta es utilizada como uno de los módulos de software en el Proyecto de Investigación *Diseño y desarrollo de aplicaciones e interfases de realidad aumentada destinadas a síntesis y procesamiento de audio digital* (IUNA, Agencia Nacional de Promoción Científica y Tecnológica, Director Carmelo Saitta).

Se asume por parte del lector tanto el conocimiento de la teoría de los *Pitch-Class Sets* (en adelante denominados PCS, véase Forte, 1974) aplicada a composición y análisis musical (véase Morris, 1984, 1987) como de las particularidades y uso del entorno de programación de alto nivel aplicable a música, audio y gráfica Pure Data (Miller Puckette, 2009). Asimismo, en tanto este informe es un módulo que continúa proyectos anteriores, es coherente con lo desarrollado en estos por lo que se recomienda su conocimiento por el lector (particularmente, la Biblioteca de Objetos Externos *Pcslib* para *Pure Data* y su documentación, véase Di Liscia, 2010).

Hipótesis de trabajo y enfoque general

Se parte del supuesto teórico de que la organización de la altura en PCS constituye un rasgo significativo de la música atonal y serial.

Partiendo de la base de los diferentes tipos de relaciones planteadas por la teoría de los PCS, es posible generar grupos de *clases de conjuntos* (en adelante denominadas SC) o de PCS en base al cumplimiento de éstas y establecer diferentes afinidades y/o relaciones entre estos grupos que son significativas en la organización musical de este nivel.

En esta fase del proyecto se eligió la aproximación al problema a través de la noción de similitud entre PCS, partiendo del análisis de las medidas de similitud propuestas por varios autores (Forte, 1974, Morris, 1984, Isaacson, 1990) en función de sus ventajas, limitaciones y aplicabilidad. Luego de ello se diseñó e implementó por software una medida que intenta superar las limitaciones observadas.

Todas las implementaciones de software fueron integradas a la Librería de Objetos Externos *Pcslib* (véase Di Liscia, 2010), desarrollada en los proyectos anteriores por razones de practicidad y consistencia en los tipos de datos y funciones básicas empleados.

1. Relaciones de similitud entre SC y PCS

La teoría "clasica" de los PCS se basa en las llamadas Clases de Sets (SC) producidas por la equivalencia o "reductibilidad" de una combinación particular de PC (Pitch-Classes o Grados Cromáticos) a otra por medio de las operaciones de Transposición o Inversión seguida de Transposición (Véase Forte, 1974, Cap. I).

Pero, como es lógico de suponer, las relaciones de equivalencia por transposición y/o inversión no son suficientes para explicar la combinación de PCS en una composición musical, y confinar los recursos de una obra a estas relaciones que produce, generalmente, resultados limitados. Por ello, los teóricos dedicados a la música atonal han desarrollado diversas maneras de medir y utilizar las variadas similitudes que pueden existir entre SC y PCS.

En principio, debe trazarse claramente la distinción entre similitudes estructurales, que son propias de las SC y existen en abstracto, y aquellas relaciones propias de los PCS que, si bien posibilitadas por su estructura, son puestas de manifiesto en una versión específica (T_n, IT_n) de los PCS y su distribución. En este último sentido, la similitud "percibida" entre dos PCS

que estructuralmente son poco parecidos puede ser tan o más fuerte que la similitud percibida entre otros dos que son muy similares estructuralmente, si se destacan en especial los escasos rasgos de similitud en los primeros a la vez que se destacan los rasgos estructurales distintivos en estos últimos en un contexto musical determinado. Sin embargo es más lógico, por supuesto, trabajar con los rasgos estructurales para luego ponerlos de manifiesto por medio de otras organizaciones.

1.1. Relaciones entre SCI[1]

Forte (Forte, 1974), establece cuatro tipo de relaciones de similitud entre las SC de igual número cardinal. La primera se refiere a los subconjuntos en común, mientras que las otras tres se relacionan con el ICV:

1. La relación R_p: es la que se produce cuando las SC comparadas tienen, al menos, un subconjunto de cardinal n-1 (siendo n el cardinal de las SC comparadas) de la misma SC. La relación puede manifestarse fuertemente (cuando el subconjunto en común mantiene, además, los mismos PC, o sea, es la intersección entre los dos PCS) o débilmente (cuando el subconjunto en común no está formado por PC comunes).

2. La relación R_0: esta relación designa máximo contraste, dado que se cumple cuando no existe una sola entrada coincidente en los ICV de las SC que se comparan. Cuanto más bajo es el número cardinal, más significativa resulta. Por ejemplo:

SC 4-Z15 ICV [111111]
SC 4-9 ICV [200022]

Los ICV de las SC 4-Z15 y 4-9 no tienen ninguna entrada igual.

3. La relación R_2: como ya se mencionó, no existen SC de cardinal mayor que tres que posean cinco entradas de su ICV idénticas. Por lo tanto, la mayor coincidencia que puede encontrarse, es de cuatro entradas. Por ejemplo (las entradas iguales están subrayadas):

SC 4-8 ICV [200121]
SC 4-9 ICV [200022]

4. La relación R_1: esta relación es como la anterior, pero agrega que las dos entradas no coincidentes tienen valores que están intercambiados. Por ejemplo: (las entradas iguales están subrayadas, las intercambiadas, se destacan con negrita)

SC 4-2 ICV [2**21**100]
SC 4-3 ICV [2**12**100]

Debe mencionarse que las cuatro relaciones ya tratadas requieren especiales refinamientos y consideraciones para su aplicación en el análisis y composición. Entre estos refinamientos se cuentan la combinación de ellas (que aumenta la relación de similitud, cuando ninguna es R_0), la posibilidad de realizar "grupos transitivos"(en los que una relación se mantiene para todas las SC que integran el grupo) y la singularidad de la relación para las SC de un determinado número cardinal.

Como ejemplo de la combinación de relaciones se exponen, en la tabla siguiente, las relaciones combinadas que surgen a partir de comparar la SC 4-20 con todas las demás SC de cardinal 4. Las SC que aparecen en la primera columna están en relación Rp con la SC 4-20, pero si se observan las columnas siguientes, solo aquellas marcadas con asterisco están, además, en las relaciones que indica el comienzo de cada columna.

Nombre	r_p	r_0	r_1	r_2
4-4	*			
4-5	*	*		
4-7	*		*	
4-8	*			
4-11	*			
4-14	*			*
4-16	*			
4-17	*		*	
4-18	*			
4-19	*			*
4-22	*			
4-26	*			
4-27	*	*		
4-29	*			

Es posible también extender el concepto de la relación R_p a subconjuntos de una cardinalidad distinta de la de las SC que se comparan menos 1, y aun a todas ellas.

Robert Morris (Morris, 1980) propone una medida de la similitud estructural de SC a partir de sus ICV que llama índice de similitud[2]. El índice de similitud (abreviado como SIM en el trabajo de Morris) tiene la ventaja de que, a diferencia de las relaciones propuestas por Forte (Forte, 1974), se puede aplicar a SC de distinta cardinalidad. La formalización es como sigue:

Sean A y B dos SC y *a* y *b* sus dos ICV.

$$SIM(A,B) = \sum_{n=1}^{6} |a[n] - b[n]| \qquad (6)$$

Si *SIM(A, B)* = Ø, entonces se trata de la misma SC o de SC en relación Z. El mínimo valor para SIM es equivalente a |*#V(A)* - *#V(B)*| (máxima similitud) y el máximo *#V(A)* + *#V(B)* (mínima similitud), en donde *#V(A)* es la suma de todas las entradas del ICV de la SC de A[3]. Morris, desarrolla ocho propiedades de esta relación que imponen refinamientos y restricciones para su consideración.

Eric Isaacson (Isaacson, 1990) realiza un análisis de las diversas funciones de similitud propuestas por Forte y Morris, además de otros autores[4] y afirma que ninguna de ellas satisface completamente las siguientes tres condiciones:

1. Proveer un valor distintivo para cada par de SC a comparar.
2. Ser utilizable para SC de cualquier cardinalidad.
3. Proveer un rango amplio de valores discretos.

En consecuencia, propone la función *IcvSIM*, que satisface las tres condiciones y se define de la manera siguiente:

$$IcvSIM(X, Y) = \sigma(IdV) \qquad (7)$$

Donde el *IdV* (vector de diferencias interválicas) es:

$$IdV = [(y1 - x1)\ (y2 - x2)\ \ldots\ (y6 - x6)]$$

X e *Y* son *ICV* y σ representa la *función de desviación estándar*, que definida en términos del *IdV* es:

$$\sigma = \sqrt{\frac{\sum (IdV_i - \overline{IdV})^2}{6}} \qquad (8)$$

Donde IdV_i es el *iésimo* término del *IdV* e $\overline{IdV}$ es el promedio de los términos en el *IdV*.

Ejemplo, usando las SC 6-35 y 8-28:

6-35 ICV = [0 6 0 6 0 3]
8-28 ICV = [4 4 8 4 4 4]
IdV = [-4 2 -8 2 -4 -1]

$$\overline{IdV} = (-4 + 2 - 8 + 2 - 4 - 1) / 6 \approx -2.16666....$$

$$\sigma \approx (((-4 - 2.167)^2 + (2 - 2.167)^2 + (-8 - 2.167)^2 + (2 - 2.167)^2 + + (-4 - 2.167)^2 + (-1 - 2.167)^2) / 6)^{0.5} \approx 3.5785$$

Al usar esta función, el máximo valor posible para la comparación de cualquier SC es aproximadamente[5] de 3.58. Este valor máximo indica mínima similitud posible y surge, en este caso, de la comparación *IcvSIM* de los ICV de las SC 6-35 y 8-28. Se observa, también, que esta función permite obtener valores cero (máxima similitud) para SC de distinta cardinalidad, como ocurre si se la aplica a la comparación de los ICV de las SC 3-10 y 6-30. A posteriori, Isaacson desarrolló cuatro tipos de similitudes más, que surgen de distintos escalamientos de la misma función propuesta.

Para graficar de manera concisa las similitudes que existen entre un grupo de SC o de PCS a partir de las relaciones de similitud ya explicadas, es usual construir matrices. Por ejemplo, usando las SC 3-12, 4-19, 4-28 y 6-33 y el criterio IcvSIM (Isaacson, 1990).

4-19	0.5			
4-28	2.14	2.00		
6-33	1.83	1.71	2.06	
		3-12	**4-19**	**4-28**

Para encontrar el IcvSIM entre dos de las cuatro SC comparadas, se busca la intersección de la fila de una con la columna de otra o viceversa. Por ejemplo, el IcvSIM entre 4-28 y y 4-19 es de 2.00 y se encuentra en la intersección de la fila (2) y la columna (1).

Otro enfoque de interés que involucra las relaciones de similitud entre SC consiste en la ponderación de determinadas cualidades aurales implícitas en la estructura de un PCS, como su tendencia hacia la consonancia o hacia la disonancia. Sería posible establecer, entonces, determinadas sucesiones de PCS desde o hacia regiones más consonantes o disonantes en una composición musical (Cetta, 2003).

1.2. Discusión de las limitaciones y evaluación de la aplicabilidad de las relaciones de similitud entre SC

Don Gibson (Gibson y Hippel, 2009) presenta el resultado de cuatro experimentos (Bruner, 1984, Gibson, 1986, 1988, 1993) destinados –según sus autores– a la investigación sobre la percepción de las relaciones de similitud planteadas por Forte y las medidas de similitud de Morris –discutidas antes– por parte de sujetos. Se presenta a continuación un breve resumen:

1. Bruner (1984): investigación con sujetos entrenados musicalmente, para determinar si se percibe la medida de similitud planteada en Morris (1980). Los sujetos fueron expuestos a pares de tricordios (PCS de cardinal 3) y debían proporcionar un número que represente la medida de similitud entre ambos. El número de sujetos de la experimentación no se provee. Los resultados indicaron escasa correlación con la medida de Morris, pero una cierta correlación con los siguientes aspectos:

a. Consonancia o asociaciones tonales.
b. Numero de sonidos en común entre los dos PCS.
c. Construcción armónica: en terceras versus en cuartas/quintas.

2. Gibson (1986): investigación realizada con 198 sujetos entrenados musicalmente, para determinar si se percibe la medida de similitud planteada en Morris (1980) y las relaciones de Forte (incluyendo la relación de "pares Z"). Los sujetos fueron expuestos alternativamente a dos pares de tetracordios (PCS de cardinal 4) ejecutados por un sintetizador en acordes, y debían determinar en cual de los dos pares escuchados se percibía mayor similitud entre los dos PCS que lo constituían. Los tetracordios con fuerte connotación tonal fueron excluídos. Sobre un total de 39 pares, solo las respuestas de tres sujetos exibieron una significativa correlación con la medida de Morris y las similitudes de Forte.

3. Gibson (1988): investigación realizada con 133 sujetos entrenados musicalmente, para determinar si la relación de "equivalencia de octava" (noción sobre la que se basa el concepto de grado cromático, o Pitch-Class que usa la teoría atonal) influye en la apreciación de similitud entre PCS. Los sujetos fueron expuestos alternativamente a dos pares de PCS ejecutados por un sintetizador en acordes. Uno de los pares estaba realizado con PCS que repetían los mismos Pitch-Classes (mismos grados cromáticos), mientras que el otro no. Los sujetos debían determinar en cuál de los dos pares escuchados se percibía mayor similitud entre los dos PCS que lo constituían. El resultado mostró que entre el 55% y el 57% de las respuestas concordaban con la teoría.

4. Gibson (1993): investigación realizada con 107 sujetos entrenados musicalmente, para determinar si la relación de "equivalencia de octava" (noción sobre la que se basa el concepto de grado cromático, o Pitch-Class que usa la teoría atonal) influye en la apreciación de similitud entre PCS. Cada uno de los 24 items a usar como estímulo consistía en dos pares de hexacordios complementarios. En un par había Pitch-Classes en común (no se especifica cuantos), en el otro no. En 12 de los 24 pares los Pitch-Classes en común eran, además, sonidos comunes (tenían igual registro). Los resultados mostraron una cantidad de 56%-57% de respuestas coincidentes, pero cuando se excluyeron los pares con Pitch-Classes en registro común, la coincidencia bajó al 49%-52%.

Ya se mencionó en el punto **1** que la similitud "percibida" entre dos PCS que estructuralmente son poco parecidos puede ser tan o más fuerte que la similitud percibida entre otros dos que son muy similares estructuralmente, si se destacan en especial los escasos rasgos de similitud en los primeros a la vez que se destacan los rasgos estructurales distintivos en estos últimos en un contexto musical determinado. Por lo tanto, no sorprende el resultado de las investigaciones precedentes, en tanto no se tiene en cuenta la particular distribución de PC. Al respecto de esto, pueden hacerse, además, las siguientes consideraciones:

1-En tres de las investigaciones se demuestra la pertinencia de PC en común (afianzada, como parece ser lógico de suponer, por registro en común). El grado de correlación es bajo, pero no lo suficiente como para descartarla. La coincidencia de la relación Rp (sub-conjunto de la misma SC en común) con el Conjunto Invariante (Intersección de PC) es considerada por Forte como un caso especial muy significativo[6].

2. En solo una de las investigaciones se usan sucesiones melódicas y en las otras tres, acordes. La música atonal utiliza frecuente, por supuesto, acordes[7]. Rara vez, sin embargo, dichos acordes se presentan en la forma de "coral" en donde la percepción de las relaciones de altura se confina, taxativamente, a la dimensión vertical o a la horizontal. Más bien, es característico de la música atonal el énfasis en un tipo de percepción que podría denominarse como "oblicua", en el sentido en que se mezclan constantemente (a través de la orquestación[8] y los cruzamientos de registro) la dimensión vertical y horizontal.

3. En todas las investigaciones se usan como estímulo notas tocadas por un sintetizador con un espectro y envolvente dinámica que no se detalla, pero se presume similar al de un piano. La percepción de varias notas simultáneas tocadas por un sintetizador es, en general, un estímulo empobrecido, dado que no tiene las propiedades de directividad en la difusión de la señal acústica y que las notas no poseen la riqueza en evolución espectral y la decorrelación que ocurre naturalmente en un instrumento acústico.

En todo caso, las investigaciones prueban que el translado al espacio musical de un PCS (por ejemplo, sus asignaciones de registro y de orden) puede producir una incongruencia con ciertas medidas de similitud entre estos, si no se ponen de manifiesto en dicho translado los rasgos comunes.

1.3 Relaciones de similitud en PCS ordenados

La discusión anterior genera la necesidad de analizar el concepto de similitud entre PCS ordenados a los efectos de evaluar y construir segmentos basados en estos. La teoría clásica de los PCS los considera no ordenados, en el sentido en que dado un PCS, cualquiera de sus permutaciones es considerada equivalente. Sin embargo, a los efectos de evaluar su similitud en una distribución lineal, es necesario considerar el orden en que aparecen los diferentes PC, dado que un orden determinado puede poner de manifiesto o disimular similitudes o diferencias estructurales subyacentes.

Los cuatro casos que se analizan a continuación presentan medidas para la evaluación de similitudes en diferentes ordenamientos de PCS, (dichas medidas se exponen en Morris, 1987 pp. 116-122).

Estas son:

1. Inversiones de órden
2. Desplazamiento
3. Dispersión
4. Correlación

Previo a su tratamiento específico, se debe mencionar que todas ellas comparan relaciones de orden y, por consiguiente, son solo apropiadas para evaluar la similitud entre las diferentes permutaciones de un mismo PCS . Sea, por ejemplo, el PCS[9] {0 1 4 6}, sus 24 permutaciones –expuestas en orden lexicográfico– son:

{0 1 4 6}{0 1 6 4}{0 4 1 6}{0 4 6 1}{0 6 1 4}{0 6 4 1}
{1 0 4 6}{1 0 6 4}{1 4 0 6}{1 4 6 0}{1 6 0 4}{1 6 4 0}
{4 0 1 6}{4 0 6 1}{4 1 0 6}{4 1 6 0}{4 6 0 1}{4 6 1 0}
{6 0 1 4}{6 0 4 1}{6 1 0 4}{6 1 4 0}{6 4 0 1}{6 4 1 0}

Asimismo, y por lo expuesto anteriormente, dichas medidas trabajan con *segmentos de referencia*. Se denominará *segmento de referencia* a la secuencia de números enteros que representan el orden de una determinada sucesión de Pitch-Classes. El segmento de referencia del primer PCS será siembre una sucesión ascendente que comienza con "0" y termina con *n-1* siendo *n* el número Pitch-Classes del PCS. Los segmentos de referencia de los PCS que se deseen comparar con este serán diferentes permutaciones del primer segmento, de acuerdo con los cambios de posición. Por ejemplo, si tomamos el ya citado PCS {0 1 4 6}, su segmento de referencia será: 0 1 2 3.
Si tomamos dos permutaciones cualesquiera de este PCS, sus respectivos segmentos de referencia son:

{1 4 0 6}1 2 0 3
{6 1 4 0}3 1 2 0

1.3.1. Inversiones de órden

La medida OI, basada en inversiones de órden (Babbitt, 1960), pondera el número de pares en orden reverso que tiene un segmento de referencia con respecto a su versión original. Una forma sencilla de implementarla consiste en sumar todos los casos en que las combinaciones binarias de elementos en el segmento de referencia NO sean ascendentes.

Siendo A=0123, B=1203 y C=3120, tres segmentos de referencia:

Las combinaciones binarias de B son:

12 **10** 13 **20** 23 03

De éstas, aquellas dos que han sido destacadas son las que constituyen inversiones de órden, dado que son descendentes.

Por lo tanto, en este caso:

OI(A,B)=2

Para el caso del segmento C, las combinaciones binarias son:

31 **32** **30** 12 **10** **20**

Por lo tanto:

OI(A,C)=5

OI varía entre un mínimo de *OI(A,A)=o*, hasta un máximo de *OI(A,RA)=np* en donde *RA* indica la retrogradación del segmento *A* y *np* es el número de combinaciones binarias posibles entre los elementos de *A (np=(n*n-n)/2* siendo n el número de elementos del segmento).

Desde el punto de vista musical, de acuerdo con esta medida, sería posible ordenar las diferentes permutaciones de un segmento gradualmente desde la que posee el mínimo de inversiones de órden hasta su retrógrado, o viceversa. Asimismo, también sería posible agrupar diferentes permutaciones por su similitud de valor en OI.

1.3.2. Desplazamiento

Según (Morris, 1987, p.119), la medida DIS(A,B) expresa cuánto han sido desviados de su posición original los Pitch-Classes de un segmento B, respecto de otro, A, tomado como referencia.

$$DIS(P,Q) = \sum_{n=0,\#P-1} ABS(n - S_n)$$

En donde: *P* y *Q* son dos permutaciones de un PCS y *S* es el segmento de referencia de uno de ellos.

Así, entonces, siendo *P*={0 1 4 6} y *Q*={1 0 6 4}, y *S(Q)*, el segmento de referencia de *Q*, es 1032.

DIS(P,Q)= ABS(0-1) + ABS(1-0) + ABS(2-3) + ABS(3-2) = 1+1+1+1= 4

DIS varía entre un mínimo de OI(A,A)=0 hasta un máximo de X, siendo X igual a:

$$X(S) = 2 + \sum_{n=1}^{\#S} n$$

Es decir, para S=0 1 2 3 , #S=4, entonces:

X= 2 + (1 + 2 + 3) = 8

1.3.3. Dispersión

Según (Morris, 1987, pp.119-120), la medida SCAT(A,B) expresa cuánto han sido dispersados respecto de sus Pitch-Classes vecinos los Pitch-Classes de un segmento B, respecto de otro, A, tomado como referencia.

$$SCAT(P,Q) = X - (\#P - 1)$$

En donde:

$$X = \sum_{k=0,\#P-1} DIST(k)$$

y:

DIST(k)=ABS(a-b) cuando *Va=k* y *Vb=k+1* siendo *V* el segmento de referencia de *Q*.

Si se calcula SCAT(P,Q) para V=10423, tenemos:

DIST(0) = ABS(0-1) V1=0, V0=1
DIST(1) = ABS(0-3) V0=1, V3=2
DIST(2) = ABS(3-4) V3=2, V4=3
DIST(3) = ABS(4-2) V4=3, V2=4
SCAT(P,Q) = (1 + 3 + 1 + 2) – 4 = 3

1.3.4. Correlación

Según (Morris, 1987, pp.120-122), es posible usar el llamado *coeficiente de correlación en Estadísticas*[10] como medida de la similitud entre PCS con el mismo contenido y diferente ordenamiento. El coeficiente de correlación entre dos PCS A y B se calcula como:

$$CC(A,B) = FSUM(S,V) / FSUM(S,S)$$

En donde S es el segmento de referencia del original y V es el segmento de referencia del PCS a comparar, y la función FSUM(X,Y) se define como sigue:

$$FSUM(X,Y) = \sum_{n=0,\#x-1} (x_n y_n) - \frac{(\sum_{n=0,\#x-1} n)^2}{\#x}$$

CC varía desde 1 para PCS comparado consigo mismo hasta -1 para la comparación con su retrogradación. Por ejemplo, en la tabla siguiente se exponen los valores de CC(A,B), en donde A={0,1,2,3} y B corresponde sucesivamente a cada una de las 24 permutaciones de A ordenadas lexicográficamente. Los valores 0, o aquellos que están cercanos son los que indican un grado más bajo de decorrelación (señalados con grisado en la tabla), mientras que los valores 1 y cercanos a 1 indican mayor correlación con el orden original y los valores -1 y cercanos a -1 mayor correlación con el retrógrado.

B	CC(A,B)
0 1 2 3	**1**
0 1 3 2	0.8
0 2 1 3	0.8
0 2 3 1	0.4
0 3 1 2	0.4
0 3 2 1	0.2
1 0 2 3	0.8
1 0 3 2	0.6
1 2 0 3	0.4
1 2 3 0	-0.2
1 3 0 2	0
1 3 2 0	-0.4
2 0 1 3	0.4
2 0 3 1	0
2 1 0 3	0.2
2 1 3 0	-0.4
2 3 0 1	-0.6
2 3 1 0	-0.8
3 0 1 2	-0.2
3 0 2 1	-0.4
3 1 0 2	-0.4
3 1 2 0	-0.8
3 2 0 1	-0.8
3 2 1 0	**-1**

1.3.5. Limitaciones y significación de las medidas para PCS ordenados de la misma SC

Como ya se mencionó, las cuatro medidas antes tratadas (OI, DIS, SCAT y CC) son solo útiles para comparar diferentes ordenamientos del mismo PCS. Por definición, estas medidas se basan en nuestra habilidad para "recordar y comparar" el orden de diferentes Pitch-Classes (y no las clases interválicas que forman entre sí). Es más, si consideráramos dichas clases interválicas, se revelarían claramente las limitaciones de estas medidas, por tanto, ciertos PCS poseen simetrías internas que producen similitudes en la distribución sucesiva de clases interválicas y que, sin embargo, no serían detectadas por las medidas que se trataron.

Un ejemplo trivial servirá para ilustrar este caso. Tómense las seis permutaciones posibles del PCS {0,4,8}, perteneciente a la SC 3-12. Sea A=0,4,8 y B equivalente a cada una de las permutaciones de A. En la tabla siguiente es fácil advertir que, mientras la sucesión de IC (clases interválicas) permanece invariante en las seis permutaciones, el coeficiente de correlación cambia.

B	Sucesión de IC	CC(A,B)
0,4,8	4,4	1
0,8,4	4,4	0.5
4,0,8	4,4	0.5
4,8,0	4,4	-0.5
8,0,4	4,4	-0.5
8,4,0	4,4	-1

Si se intentara extender la acción de tales medidas a las diferentes transposiciones de un PCS ordenado, se deberían calcular segmentos de referencia a partir de la reducción del PCS a comparar al PCS original por medio de transposición.

Sean, por ejemplo, los PCS A={0,1,4,6} y B={0,10,6,7}, pertenecientes ambos a la SC 4-Z15, el segmento de referencia de A sería, por supuesto {0,1,2,3} y para calcular el de B, habría que reducirlo por transposición a A. En este caso, el operador de transposición necesario es 6, dado que T6(B)=A:

B={0,10,6,7}
T6(B)={6,4,0,1}

El segmento de referencia para T6(B) ={6,4,0,1} respecto de A={0,1,4,6}, sería 3,2,0,1.

Es fácil deducir de los datos precedentes que extensión de las medidas de evaluación de orden a diferentes transposiciones ordenadas del mismo PCS están sujetas a las mismas limitaciones que se mencionaron, con el agravante que, en este caso, el oyente no tendrá la posibilidad de comparar sucesiones de Pitch-Classes, sino de intervalos y, más aún, Pitch-Classes invariantes que pueden producirse en las diferentes transposiciones entregarán información incongruente con el resultado de la medida de orden que se use, cualquiera sea esta.

1.4. Similitud entre PCS de diferente SC ordenados: la medida OPSC

1.4.1. Discusión preliminar

En esta investigación se ha diseñado una medida de similitud entre PCS ordenados que intenta superar las limitaciones observadas en las analizadas precedentemente. Esta medida, que ha sido denominada OPSC (Ordered PCS Similarity Coefficient), se basa en los siguientes supuestos teóricos:

1. La sucesión de PCS (subconjuntos adyacentes) puesta de manifiesto por los determinados ordenamientos de dos o más PCS, es relevante para evaluar la similitud entre estos.
2. La Intersección entre dos o más PCS (por ejemplo, los PC que tienen en común) es relevante para evaluar la similitud entre estos.

Ambos supuestos teóricos se basan en las dos habilidades principales de los oyentes en cuanto a la apreciación de la altura tonal: la similitud de intervalos y la similitud de altura.

Se ofrece a continuación un ejemplo que da cuenta de la complejidad de la situación:

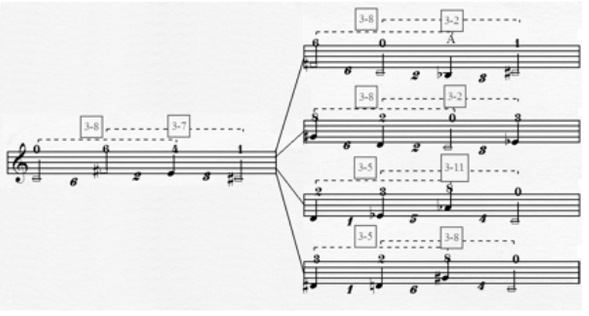


Figura 1: comparación de PCS ordenados en notación musical.

La Figura 1 muestra, en el compás de la izquierda, un determinado orden y transposición del PCS {0,6,4,1}, perteneciente a la SC 4-Z15, con el que son comparadas las diferentes transposiciones ordenadas: {6,0,A,1}, {8,2,0,3}, {2,3,8,0} y {3,2,8,0}, pertenecientes todas a la SC 4-Z29. Las comparaciones son entonces:

{0,6,4,1} versus {6,0,A,1}
{0,6,4,1} versus {8,2,0,3}
{0,6,4,1} versus {2,3,8,0}
{0,6,4,1} versus {3,2,8,0}

En la figura, los números en itálicas entre las notas denotan las clases interválicas que se ponen de manifiesto en la sucesión. Las llaves superiores (en línea punteada) delimitan los PCS de cardinal 3 adyacentes imbricados que se ponen de manifiesto en la sucesión. En el cuadro siguiente se puede observar el resultado de la evaluación de acuerdo con los criterios mencionados:

PCS1	PCS2	IC en comun	PCS en comun	PC en comun
{0,6,4,1}	{6,0,A,1}	3 (100%)	1(50%)	3(75%)
{0,6,4,1}	{8,2,0,3}	3 (100%)	1(50%)	1(25%)
{0,6,4,1}	{2,3,8,0}	0(0%)	0(0%)	1(25%)
{0,6,4,1}	{3,2,8,0}	1(33%)	1(50%)	1(25%)

Sobre la base de lo expuesto, se diseñó la medida OPSC, que tiene las siguientes ventajas:

1. Se basa en las dos características de una sucesión de PC para las que está probada nuestra habilidad perceptiva en la altura tonal: los PC intervinientes y los conjuntos sucesivos que forman.

2. Se puede aplicar tanto a grupos de PCS de distintas SC como a diferentes permutaciones de un PCS de la misma SC.

3. Se puede aplicar a grupos de PCS de diferente cardinalidad.
4. Se puede aplicar a grupos de PCS que tengan PC repetidos.

1.4.2. Definición de la medida OPSC

La medida OPSC se define como sigue:

$$OPSC(A,B,w) = CSP(A,B)(1-w) + CSI(A,B)(w)$$

En donde:

A y **B** son PCS ordenados

CSP(A,B) es el coeficiente de similitud de Pitch-Classes entre **A** y **B**

CSI(A,B) es el coeficiente de similitud Intervá lica entre **A** y **B**

w es un factor de escalamiento entre **CSI** y **CSP** (w=1 toma en cuenta solo la similitud interválica, mientras que w=0 toma en cuenta solo la similitud de Pitch-Classes, w=0.5 toma en cuenta tanto a **CSI** como a **CSP** por igual, etc.).

A su vez, **CSP**(A,B) se define como:

$$CSP(A,B) = npcc(A,B)/s$$

En donde:

npcc(A,B) es el número de Pitch-Classes en común entre A y B y s es el número de Pitch Classes que contenga el conjunto de menor tamaño[11] .

Y **CSI**(A,B) se define como:

$$CSI(A,B) = \left(\sum_{n=1, n=NSA(A)} \#SA_n(A) * SAC(SA_n, B) \right) / scal(A,B)$$

En donde:

NSA(A) es el número de PCS adyacentes de número de PC mayor que 1 de un PCS orde nado. Así, por ejemplo, el PCS ordenado o **A**={0,6,4,1}, presenta 6 de tales PCS en suce sión. Tres de dos PC ({0,6}{6,4} y {4,1}), dos de tres PC ({0,6,4} y {6,4,1}) y uno de cuatro PC (el mismo PCS **A**).

$\#\mathbf{SA}_n$ es el número de PCs del n-ésimo PCS adyacente perteneciente al PCS ordenado **A**.

La función **SAC**(SAn, B) retorna **1** si el PCS adya-cente **n** de A pertenece a la misma clase que cualquier PCS adyacente de igual número de PC que se encuentre en el PCS ordenado B y coincide con su estatus de inversión (si está invertido o no), retorna **0.5** si hay coincidencia de SC, pero no de estatus de inversión, y retorna **0** si no hay coincidencia de SC.

Finalmente, **scal**(A,B) es un factor de escalamiento para mantener a **CSI** en el rango de 0 a 1, y se define como sigue:

$$scal(A,B) = \sum_{n=0, n=\#A-1} (\#\mathrm{A} - \mathrm{n})(\mathrm{n}+1)$$

En donde:

#A es la cantidad de PC en el PCS de menor tamaño, **A**.

Por ejemplo, dados los PCS ordenados:

A={0,6,4,1}
B={6,0,A,1,2}
Y el coeficiente w=0.5

OPSC(A,B,w) se calcularía de la forma siguiente:

CSP(A,B)= 3 / 4 = 0.75

Dado que npcc(A,B)=3 (los PC en comun entre A y B son 3: 0,1 y 6) y el número de PC del PCS de menor tamaño (s) es igual al número de PC de A.

Explicamos ahora el cálculo de CSI(A,B). Para ello, exponemos a continuación los PCS adyacentes presentes en ambos PCS ordenados, A y B y sus correspondientes SC:

A	
{0,6,4,1}	SC= 4-Z15
{0,6,4}	SC= 3-8
{6,4,1}	SC= 3-7
{0,6}	SC= 2-6
{6,4}	SC= 2-2
{4,1}	SC= 2-3

B	
{6,0,A,1,2}	SC= 5-13
{6,0,A,1}	SC= 4-Z29
{0,A,1,2}	SC= 4-2
{6,0,A}	SC= 3-8
{0,A,1}	SC= 3-2
{A,1,2}	SC= 3-3
{6,0}	SC= 2-6
{0,A}	SC= 2-2
{A,1}	SC= 2-3
{1,2}	SC= 2-1

Repetimos, por conveniencia, la ecuación para el cálculo de scal(A.B):

$$scal(A,B) = \sum_{n=0, n=\#A-1} (\#\mathrm{A} - \mathrm{n})(\mathrm{n}+1)$$

Entonces:
scal(A,B) = (4*1) + (3*2) + (2*3) = 4 + 6 + 6 = 16

Y también la ecuación para el cálculo de CSI(A,B):

$$CSI(A,B) = \left(\sum_{n=1, n=NSA(A)} \#SA_n(A) * SAC(SA_n, B) \right) / scal(A,B)$$

Ni la SC 4-Z15, ni la SC 3-7, presentadas de manera adyacente en A se encuentran entre los PCS adyacentes de B. Pero las SC 2-6, 2-2 y 2-3, también presentadas de manera adyacente en A, se encuentran entre los PCS adyacentes de B y coinciden en su estatus de inversión. Por lo tanto:

CSI(A,B)= (4*0 + 3*1 + 3*0 + 2*1 + 2*1 + 2*1) / 16
(3 + 2 + 2 + 2) / 16
9 / 16
0.5625

Entonces:

$$OPSC(A,B,w) = CSP(A,B)(1-w) + CSI(A,B)(w)$$

OPCS(A,B,w) = (0.75 * 0.5) + (0.5625*0.5)
0.375 + 0.28125
0.65625

1.4.3. Discusión de las limitaciones y aplicabilidad de la medida OPSC

La medida OPSC presentada antes posee evidentes ventajas por sobre las precedentes. Sin embargo, algunas de sus características que se comentarán en lo que sigue, son experimentales y permanecen todavía en evaluación.

1. La ponderación de las coincidencias de SC adyacentes posee, de momento, una cierta arbitrariedad. Todo lo que se hace aquí es asignar un puntaje equivalente a la cantidad de PC del PCS adyacente coincidente con el PCS a comparar. Pero se podría experimentar con otras asignaciones de puntaje.

Una posibilidad sería establecer porcentajes iguales para cada grupo de PCS de igual cantidad de PC, de acuerdo con su cantidad en el PCS a comparar. Por ejemplo, ya se mencionó antes que un PCS de 4 PCs presenta 6 PCS imbricados de manera adyacente (1 PCS de 4 PC, 2 PCS de 3 PC y 3 PCS de 2 PC). Se podrían establecer porcentajes iguales para los PCS de 4, los de 3 y los de 2 PC (1/3 para cada uno de ellos). En este caso, las coincidencias sumarían al puntaje (1/3)/ns, siendo ns el número de PCS de cada número de elementos (en este caso 1 para los de 4, 2 para los de 3, etc.). Esta posibilidad favorecería a los PCS de mayor número de elementos, una característica que parece lógica, ya que la similitud se haría más significativa en la medida en la que segmentos adyacentes de mayor extensión se presentan. Sin embargo, hay que considerar que, en la medida en que los segmentos se alargan, resulta más difícil retener todos los detalles de la sucesión interválica que presentan. Otra posibilidad sería establecer escalamiento de la comparación en base a una función c=f(n), siendo n el número de elementos de cada PCS adyacente. Pero, en cualquiera de los casos, resulta claro que la importancia de la contribución de cada coincidencia de PCS adyacente debería medirse en función de su número de PC.

2. Si bien la presencia de PC repetidos en los PCS ordenados a comparar resulta correctamente reflejada en esta medida, determinados casos "extremos" que podrían producirse todavía están en estudio. Por ejemplo, la comparación del PCS A={0,0,0,0} con el PCS B={0,1,2,3} con w=0.5 daría como resultado 0.5.

3. La medida OPSC no toma en cuenta las distintas "distancias" a las que se encuentran los PCS adyacentes a comparar. Dicho de otra manera, un PCS adyacente del PCS A recibe el mismo puntaje si se encuentra como adyacente al comienzo del PCS B como al final. No se trata aquí de cuestiones de orden serial, sino de la posibilidad de ejercitar la memoria en la comparación en la medida en que más PC son presentados en sucesión. Si este último fuera el caso, una ponderación extra de acuerdo con la "distancia" podría incluirse.

4. La medida OPCS no incluye las relaciones de complementariedad, los complejos Kh ni las de los "pares Z" (Forte, 1974) por considerar que son rasgos estructurales importantes a los fines constructivos, pero no necesariamente proveen criterios mediciones de similitud.

Por último, debe mencionarse el hecho de que no está garantizado que la medida OPSC produzca un número único para cada comparación posible. Dicho de otra manera, distintas parejas de PCS de SC y cardinalidad diferentes en diferentes ordenamientos podrían producir coeficientes de similitud iguales. Lejos de ser una desventaja, ésta es una característica que podría ser considerada el *desideratum* de la organización musical de los compositores de música atonal: lograr una sensación de coherencia a través de un juego sutil de diferencias.

2. Implementaciones de software para aplicación de la teoría

Para la aplicación de los recursos teóricos expuestos, se desarrollaron, en el marco de la presente investigación, dos objetos externos para el programa PD (*Pure Data*, Miller Puckette *et all*) que se integraron a la Librería de Objetos Externos *Pcslib* (Pablo Di Liscia y Pablo Cetta, 2007-2008) ya desarrollada antes en este proyecto. La siguiente discusión asume por parte del lector el conocimiento del programa *Pure Data* (véase Puckette, 2009) y de la Biblioteca de Objetos Externos Pcslib (véase Di Liscia, 2010).

En primer lugar, a los efectos de disponer eficientemente de todas las permutaciones de un PCS, se desarrolló el objeto *pcs_perm*.

El objeto *pcs_perm* calcula todas las diferentes permutaciones posibles de un PCS. Estas permutaciones son almacenadas internamente en orden lexicográfico. Las repeticiones de PC y posiciones diferentes no se toman en cuenta para el cálculo de las permutaciones.

Por ejemplo, un PCS entregado como: 0 1 -1 0 11 -1 5 será considerado como: 0 1 11 5. El número cardinal del PCS debe ser menor o igual a 7.

Entrada:
Inlet2: un puntero a una estructura PCS.
Inlet1: cualquier mensaje produce la salida de todas las permutaciones diferentes. Estas se entregarán en orden lexicográfico a través de una serie de listas. El mensaje "get n" siendo "n" el número de permutación, causará la salida de la permutación n solamente (recuérdese que el objeto almacena las permutaciones en orden lexicográfico).

Salida:
Outlet1: véase la explicación de los mensajes del Inlet1 y su efecto en la salida.
Outlet2: el número de permutaciones diferentes del PCS (equivale a n siendo n el número cardinal del PCS).

En segundo lugar, se diseñó el objeto externo *pcs_sim2*[12] a los efectos de implementar eficientemente los tipos de medidas estudiados.

El objeto *pcs_sim2* evalua el grado de similitud de dos o más SC o PCS de acuerdo con distintos criterios.

Entrada:
Inlet2: una serie de PCS a ser mutuamente comparados (deben ser al menos dos). Los PCS son acumulados internamente hasta que un mensaje de "reset" es recibido.
Inlet1: hay varias posibilidades de mensajes, que se pueden clasificar en tres categorías:

1. El mensaje "reset" elimina la lista de PCS guardados, si la hubiera.
2. Cualquiera de estos símbolos producirá la salida del resultado de los siguientes tipos de similitudes:
-"ro" "r1" "r2" or "rp": similitudes de Forte. (solo para diferentes SC de la misma cardinalidad que debe ser >3 y <7).
"ICVSIM" symbol: coeficiente de similitud de Isaacson (sin limitaciones).
"SIM" or "ASIM" symbols: coeficientes de similitud de Morris (sin limitaciones).
-"OI" symbol: "inversiones de órden" de Babbitt (solo para diferentes permutaciones del mismo PCS).
-"DIS" symbol: medida de "desplazamiento" de Morris (solo para diferentes permutaciones del mismo PCS o sus trasposiciones).
-"SCAT" symbol: medida de "dispersion" de Morris (solo para diferentes permutaciones del mismo PCS o sus trasposiciones).
-"CC" symbol: coeficiente de correlación(solo para diferentes permutaciones del mismo PCS o sus trasposiciones).
-"OPSC" w(float): "Ordered PCS Similarity Coefficient". Diseñado experimentalmente por Pablo Di Liscia. (sin limitaciones, pero nótese que esta medida toma en cuenta tanto los PC en común como la similitud en PCS adyacentes y fue especialmente diseñada para PCS ordenados). Esta medida varía desde 0 (mínima similitud) hasta 11(máxima similitud). Al símbolo OPSC se debe agregar un float (w) indicando el valor de ponderación de similitud de Pitch-Class versus la similitud interválica. W varía desde 0 a 1, en donde w=0 indica solo evaluación de similitud de Pitch-Classes (PC en comun entre los PCS a comparar), mientras que w=1 indica solo evaluación de similitud interválica (a través de la ponderación de PCS adyacentes, como ya se ha explicado). Un valor de w=0.5 pondera ambas características por igual.
3. El mensaje "write" dirige la salida de la comparación a un archivo de texto plano. Este mensaje debe ser seguido de un nombre de archivo válido y uno de los símbolos explicados en el apartado 2. Por ejemplo: "write similarities.txt ro" causará que la matriz de comparación con el criterio ro se escriba en el archivo "similarities.txt".

Salida:

Outlet1: La salida es una serie de listas que imitan el orden de una matriz triangular de comparación. Sigue un ejemplo de tal matriz. Siendo (A), (B), (C), (D) y (E) PCS entregados en ese orden, la salida será:

	(B)	(C)	(D)	(E)	
(A)	ab	ac	ad	ae	<--(lista 1 de N-1 floats)
(B)		bc	bd	be	<--(lista 2 de N-2 floats)
(C)			cd	ce	<--(lista 3 de N-3 floats)
(D)				de	<--(lista 4 de N-4 floats)

Así, por ejemplo, para saber el resultado de la comparación entre (B) y (C) se debe leer el contenido de la posición que sea la intersección entre la fila con la etiqueta (B) con la columna etiquetada (C), que está completa en este caso con "bc", pero en realidad deberá tener un número o un símbolo que indique el resultado.

El ejemplo que se muestra antes presenta una serie de cinco PCS, por consiguiente, la salida será una lista de cuatro listas de N-n floats, siendo N el número de PCS comparados y n el número de lista de la salida. El contenido de cada posición de la matriz se interpreta como sigue:

Similitudes de Forte: un float (0 o 1). "1" significa que la relación se cumple, mientras que un "0" significa

que no. Un -100 indicará que los PCS a comparar no cumplen las condiciones que se establecen para la relación y, por consiguiente, no se pueden comparar. Otras relaciones: un float (valor calculado de la similitud). En el caso de las relaciones OI, DIS, SCAT y CC, un -100 indicará que los PCS a comparar no cumplen las condiciones que se establecen para la relación y, por consiguiente, no se pueden comparar.
Outlet2: en el caso de las relaciones de Forte, no hay salida. En los otros casos, una lista con dos floats indicando respectivamente el máximo y mínimo valor encontrado en la comparación.

En el caso del mensaje "write", no habrá salida de los outlets y los datos escritos en el archivo tendrán el mismo formato explicado, excepto que el -100 se reemplaza por "NC" (no comparable) y en el caso de las relaciones de Forte, se usan "+" o "-" para el cumplimiento o no de la relación.

En la Figura 2 se muestra un *patch* de PD realizado para demostrar la acción del objeto pcs_sim2, en este caso, tomando como base los cinco PCS ordenados que se presentaron en la Figura 1.

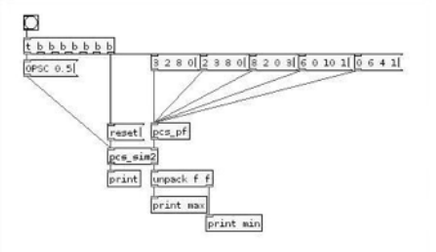


Figura 2: un patch de PD que evalúa la medida OPSC con w=0.5 en los PCS ordenados de la figura 1.

La salida que produce este *patch* en el prompt de PD es la siguiente:

0.65625	0.40625	0.12500	0.28125
	0.62500	0.25000	0.40625
		0.62500	0.78125
			0.84375

min: 0.125
max: 0.84375

Si tomamos únicamente la primera línea:
0.65625 0.40625 0.12500 0.28125

Allí aparecen los valores de OPSC para la comparación binaria del primer PCS entregado versus los siguientes, o sea:

OPSC({0,6,4,1},{6,0,A,1})	=	0.65625
OPSC({0,6,4,1},{8,2,0,3})	=	0.40625
OPSC({0,6,4,1},{2,3,8,0})	=	0.12500
OPSC({0,6,4,1},{3,2,8,0})	=	0.28125

En el *patch* de PD que se ve en la Figura 3, se presenta un ejemplo de utilización combinada de los objetos pcs_perm y pcs_sim2. En primer lugar, se crea un PCS a través del objeto pcs_pf, luego se obtienen todas las permutaciones del mismo y, finalmente se envían en orden lexicográfico al objeto pcs_sim2. El mensaje "CC" causa que el objeto pcs_sim2 imprima en el prompt de PD la siguiente línea[13]:

0.8 0.8 0.4 0.4 0.2 0.8 0.6 0.4 -0.2 0 -0.4 0.4 0 0.2 -0.4 -0.6 -0.8 -0.2 -0.4 -0.4 -0.8 -0.8 -1

que muestra el valor del coeficiente correlación (CC) que surge de la comparación del PCS inicial (0, 1, 3, 7) con cada una de sus permutaciones, en orden lexicográfico.

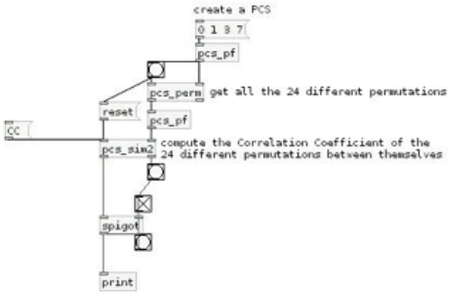


Figura 3: un patch de PD que evalúa la medida CC en un PCS de cardinalidad 4 y sus 24 permutaciones.

Agradecimientos

El autor agradece a la Fundación Carolina de España y a la Universidad Nacional de Quilmes por el financiamiento de la Beca Posdoctoral que posibilitó esta investigación.

Asimismo, agradece al Music Technology Group de la Universidad Pompeu Fabra de Barcelona, España, y a su Director, el Dr. Xavier Serra por albergar la presente investigación.

Referencias Bibliográficas

Babbit, Milton. "Set Structure as Composicional Determinant", en *Journal of Music Theory 5* (1961: Nº 1).

Bruner, Cheryl L. "The Perception of Contemporary Pitch Structures", en *Music Perception* (1984: Vol. 2, Nº 1 pp. 25-39).

Di Liscia, Pablo. "Pitch-class composition in the pd environment", en Registros Históricos del XII Simposio Brasileño de Computación y Música (Recife, Brasil: UFP, 2009).
— <https://puredata.info/author/pdiliscia>, sitio de la biblioteca Pcslib (2010).
— y Cetta, Pablo. *Elementos de contrapunto atonal* (Buenos Aires: EDUCA-Universidad Católica Argentina, en prensa).

Cetta, Pablo. *Principios de estructuración de la altura empleando conjuntos de grados cromáticos* (Buenos Aires: Facultad de Ciencias y Artes Musicales de la UCA, 2003, pp. 9-35). Capítulo "Altura-Timbre-Espacio".

Forte, Allen. *The Structure of Atonal Music* (Londres: Yale University Press, 1974).

Gibson, Don. "The Aural Perception of Nontraditional Chords in Selected Theoretical Relationships: A Computer-Generated Experiment", en *Journal of Research in Music Education* (1986: Vol. 34, Nº 1 pp. 5-23).
— "The Aural Perception of Similarity in Non-traditional Chords Related by Octave Equivalence", en Journal of Research in Music Education (1988: Vol. 36, Nº 1 pp. 5-17).
— *The Effects of Pitch and Pitch-class Content on their Aural Perception of Dissimilarity in Complementary Hexachords. Psychomusicology* (1993: Vol. 12, Nº 1, pp. 58-72).
— <http://www.musiccog.ohio-state.edu/Gibson/research.summary.html>, sitio consultado el 01-10-2009.

Isaacson, Eric. "Similarity of Interval-Class Content between Pitch-Class Sets: The IcVSIM Relation", en *Journal of Music Theory* (1990: Vol. 34 Nº 1, pp. 1-28).

Morris, Robert. "A similarity Index for Pitch-Class Sets", en *Perspectives of New Music* (1980: Vol. 18, Nº 1-2, pp. 445-460).
— "Combinatoriality without the Aggregate", en *Perspectives of New Music* (1984: Vol. 21, Nº 1-2, pp. 432- 486).
— *Composition with Pitch-Classes: A Theory of Compositional Design* (Londres: Yale University Press, 1987).

Puckette, Miller. "PD Documentation", <http://crca.ucsd.edu/~msp/Pd_documentation/>,
(2009).

Notas

1. Lo expuesto en este punto pertenece a (Di Liscia, Cetta, 2008).

2. Similarity Index, véase Morris, 1980.

3. Existe, también, una manera propuesta por Morris para "normalizar" los valores máximos y mínimos de manera tal que sean independientes de la cardinalidad de las SC que se comparan. Tal función se llama ASIM.

4. Teitelbaum, Lewin, Rahn y Lord.

5. El número exacto dependerá, por supuesto, de la cantidad de decimales usados en el cálculo, pero, el autor no usa más que tres de ellos.

6. Forte considera que, en este caso, la relación Rp puede cumplirse "fuertemente" o "débilmente" ("Strongly", "Weakly"; véase Forte, 1974, pp. 50).

7. Un caso célebre lo constituye el *continuum* principal de la pieza Op. 16 Nº 3 (Farben) para orquesta de Arnold Schönberg –cuyo PCS básico es el 5-Z17– y en la que, sin embargo, la percepción "acórdica" o "vertical" está absolutamente velada por la orquestación y la espacialidad.

8. La llamada *klangfarbenmelodie* ("melodía de colores y timbres"), típica de la Escuela de Viena.

9. Luego se tratará un intento de extender estas medidas a diferentes transposiciones de un mismo PCS.

10. El autor cita a Allen, Edwards C. *An introduction to linear regression and correlation* (San Francisco: W.H. Freeman, 1976), pero es posible encontrar abundantes referencias al respecto en textos especializados en Estadísticas.

11. Es decir, si #A ≤ #B s=#A y viceversa.

12. El objeto pcs_sim, ya existente, se mantuvo sin cambios por razones de compatibilidad.

13. Por medio del uso del objeto spigot, se evita que las demás líneas resulten impresas, para mayor claridad.

EMILIANO CAUSA

Artista multimedia e ingeniero en Sistemas de Información (recibido en la Universidad Tecnológica Nacional).

Es integrante fundador del Grupo Proyecto Biopus, Coordinador del MediaLab del Centro Cultural de España en Buenos Aires. Se desempeña como docente e investigador en la Facultad de Bellas Artes (UNLP) y en el Área Transdepartamental de Artes Multimediales (IUNA).

Se dedica al arte multimedia, arte biogenerativo, net-art, a la música y video experimental, a la construcción de instalaciones con sensores y a la aplicación de la informática al arte en general.

Diseño de interface para el desarrollo de una pantalla sensible al tacto con aplicación musical

Emiliano Causa

Palabras claves

Pantalla sensible al tacto, Pitch-Class Set, interpretación gestual, interfaces tangibles

1. Introducción

El presente texto expone el proceso de diseño de una interface de pantalla sensible al tacto (del tipo multitacto) aplicada a un editor gestual de música. La aplicación consiste en un editor de partitura analógica que permita escribir gestos musicales de una forma sensilla, a la vez que se determina su dinámica, tempo y grado de consonancia. El presente trabajo forma parte de la investigación *Diseño y desarrollo de aplicaciones e interfaces de realidad aumentada destinadas a síntesis y procesamiento de audio digital* (dirigida por Carmelo Saitta y Pablo Cetta). El Dr. Pablo Cetta viene desarrollando (junto al Dr. Pablo Di Liscia) investigaciones sobre los Pitch-Class Set, para el diseño armónico de la música. El presente desarrollo intenta crear una interface intuitiva para la creación musical aprovechando el conjunto de librerías de programación creadas en la investigación sobre Pitch- Class Set. Una aplicación de este tipo permitiría crear gestos musicales y definir su nivel de consonancia en forma dinámica, mediante la aplicación de los Pitch-Class Set.

2. Elementos musicales

En los primeros pasos de este desarrollo, durante reuniones con el Dr. Pablo Cetta y el Lic. Matías Romero Costas, se definió un conjunto de gestos musicales (notación musical) que podrían trazarse sobre una pantalla sensible al tacto. Este conjunto está definido por la siguiente notación:

Alturas puntuales

Diseño melódico

Acordes

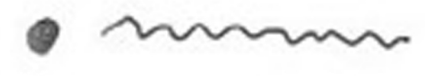
Arpegio

Duración

Glissando

Trémolo

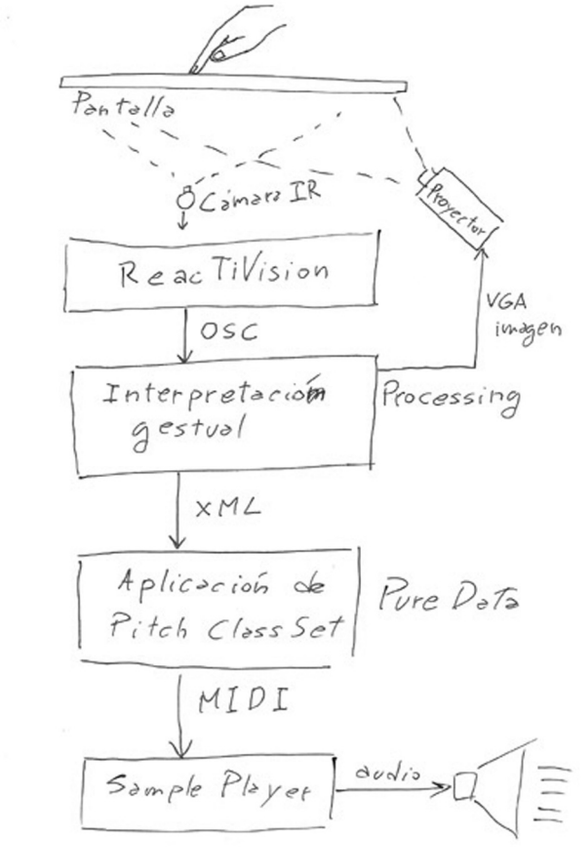
Trino



3. Elección de herramientas

Dado que las librerías ya citadas están desarrolladas en Pure Data (una herramienta de software para desarrollos de composición algorítmica) se diseñó una plataforma que vincula diferentes softwares para implementar una pantalla sensible al tacto con una interface que interprete la notación recién descripta. Para el desarrollo de la pantalla se eligió ReacTiVision. Una aplicación que realiza un sistema de sensado óptico de interfaces tangibles y también funciona con una pantalla sensible al tacto. Este software es open source, lo que nos permite utilizarlo para este desarrollo.

Con el fin de interpretar los gestos desarrollados sobre la pantalla, para traducirlos en los diferentes elementos musicales, se utiliza Processing. ReacTiVision se comunica con Processing vía OSC, un protocolo de comunicación pensado para la transmisión de sonido, pero que en este caso transmite el tipo de elemento de interface que se encuentra sobre la pantalla, así como los gestos desarrollados con los dedos.



Una vez que Processing interpreta los gestos y los traduce en elementos (por ejemplo, un dedo apoyado sobre la pantalla en forma solitaria es interpretado como una altura puntual), se comunica con Pure Data mediante un protocolo de red, transmitiendo en formato XML cada elemento. Por último, Pure Data se encarga de gestionar los mensajes MIDI necesarios para reproducir los elementos musicales. Estos mensajes llegan a un Sample Player que se encarga de producir finalmente el sonido.

3.1. ReacTiVision y las interfaces tangibles (*)

ReacTiVision (mtg.upf.es/reactable/) es una herramienta de software desarrollada por Sergi Jordà, Martin Kaltenbrunner, Günter Geiger y Marcos Alonso, quienes conforman el Grupo de Tecnología Musical dentro del Instituto Audiovisual en la Universidad Pompeu Fabra (Barcelona, España). Esta tecnología permite reconocer patrones bitonales (llamados “fiducials”) impresos a piezas de interfaces tangibles que funcionan sobre una pantalla sensible al tacto. Esta interface tangible consiste en piezas de acrílico que se apoyan sobre una pantalla sensible, ésta es capaz de reconocer las piezas, gracias a los patrones bitonales y generar respuestas audiovisuales en consecuencia. Los creadores, construyeron este software para desarrollar una pantalla sensible para interpretación musical en tiempo real (un instrumento de improvisación de música electrónica) llamada ReacTable.

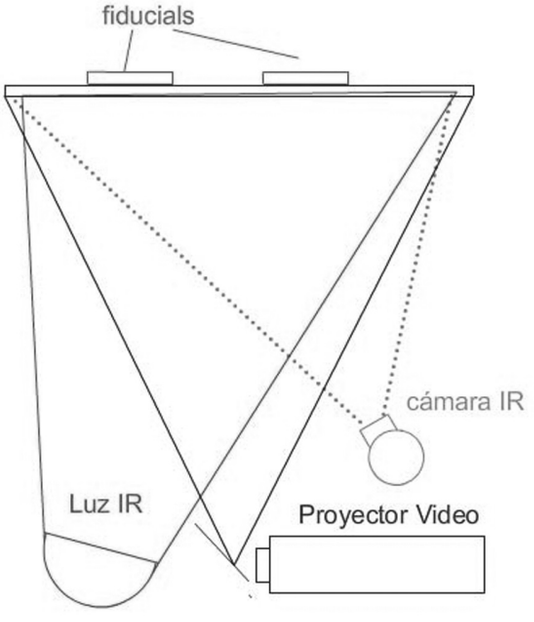


Figura 3: esquema ReacTable

Como muestra la Figura 3, ReacTiVision permite hacer el reconocimiento de patrones bitonales, a través de un sistema óptico, que en el caso de la ReacTable se implementa con luces y cámara infrarrojas. La pantalla es un acrílico con superficie esmerilada, las imágenes

se retro proyectan desde abajo usando un cañón de video, a su vez una luz infrarroja permite iluminar los patrones que serán captados por una cámara, también infrarroja. Dicha luz y cámara son infrarrojas para no interferir con la luz del proyector de video (que pertenece al rango visible de la luz), y para que la cámara no vea a su vez las proyecciones.

Uno de los aspectos más interesantes de ReacTiVision es que está construido como un software independiente, que envía datos respecto de los parámetros de los "fiducials": la ubicación, identificación, rotación y otros; vía el protocolo OSC (Open Sound Control). Esto permite que cualquier otro software que reciba mensajes en OSC, pueda comunicarse con ReacTiVision e interpretar información respecto del estado de cada uno de los patrones bitonales ubicados sobre la pantalla. Debido a esto, existe en el sitio de ReacTiVision, ejemplos de conexión de este software con lenguajes como: C++, Java, C#, Processing, Pure Data, Max/MSP, Flash y otros.

3.2. Processing y la librería ReacTiVision (*)

Processing (www.processing.org) es un lenguaje de programación diseño para artistas por Ben Fry y Casey Reas. Es un lenguaje "open source" desarrollado en Java de gran potencia y facilidad de aprendizaje. Una de las ventajas de Processing es el extenso desarrollo de librerías que extienden las posibilidades de conexión de este lenguaje con otros formatos, protocolos o lenguajes. Como hemos dicho en el apartado anterior, existe una librería que permite conectar, vía OSC, a Processing con ReacTiVision.

ReacTiVision tiene una aplicación ejecutable que se conecta a la cámara y reconoce los patrones (fiducials) que estén en la imagen, enviando por OSC los parámetros de cada patrón (en un protocolo que los autores llamaron TUIO). Esta librería implementa un conjunto de instrucciones que permiten leer dichos parámetros. Por ejemplo, la siguiente línea de código crea un objeto "cliente de protocolo TUIO" :

```
TuioClient client = new TuioClient(this);
```

Y las instrucciones que siguen, son funciones que se ejecutan frente a eventos provocados por los patrones:

```
void addTuioObject(int s_id, int f_id, float
xpos, float ypos, float angle) {}

void removeTuioObject(int s_id,int f_id ) {}

void updateTuioObject (int s_id, int f_id,
float xpos, float ypos, float angle, float xs-
peed, float yspeed, float rspeed, float maccel,
float raccel) {}
void addTuioCursor(int s_id, float xpos, float
ypos) {}
void removeTuioCursor(int s_id) {}

void updateTuioCursor (int s_id, float xpos,
float ypos, float xspeed, float yspeed, float
maccel) {}
```

Por ejemplo, TUIO distingue dos tipos de elementos: objetos y cursores. Los objetos son los patrones bitonales, mientras que los cursores son los dedos que se apoyan sobre la pantalla (dado que el sistema también es capaz de reconocer el tacto). Cada una de estas funciones, informan un evento de un patrón o de tacto:

addTuioObject: informa la aparición de un nuevo patrón sobre la pantalla.
removeTuioObject: informa que un patrón salió de la pantalla.
updateTuioObject: informa los cambio que sufre un patrón, ya sea de posición como de rotación.
addTuioCursor: informa la aparición de un dedo sobre la pantalla.
removeTuioCursor: informa que un dedo salió de la pantalla.
updateTuioCursor: informa los cambio que sufre un dedo, un cambio de posición.

4. Diseño gestual

Los últimos tres eventos, que son disparados por acciones con los dedos, son los que se utilizan para interpretar los gestos que son traducidos en los elementos musicales. Basados en esto se desarrolló un diseño gestual que intenta lograr una interface muy intuitiva para el usuario. A continuación explicaremos, para cada elemento, cuál es el tipo de gesto que los dedos deben realizar sobre la pantalla, así como la forma en que este elemento es descripto en un mensaje de XML. La aplicación generada con Processing debe encargarse de traducir estos gestos en cada uno de sus correspondientes mensajes XML.

4.1. Alturas puntuales

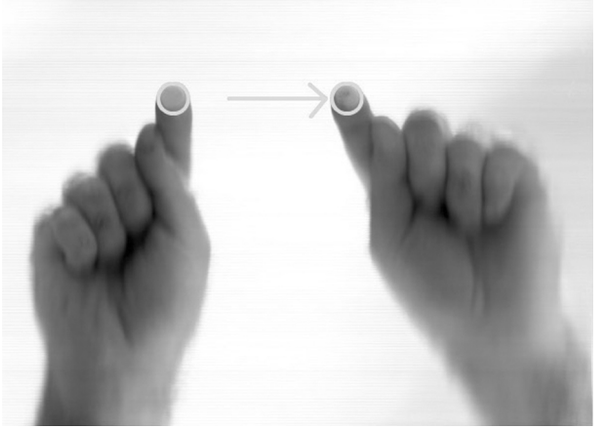
Para la realización de este gesto el usuario deberá hacer presiones sobre la pantalla, usando la punta del dedo y en forma breve:

Este gesto es traducido en el siguiente mensaje XML:

```
<NOTA ID='#'>
<A> altura </A>
<T> tiempo_inicio</T>
</NOTA>
```

4.2. Duración

Para establecer la duración de una nota sostenida, luego de apoyar el dedo de la nota, se apoya otro por el lado derecho y se realiza un trazo en forma paralela:

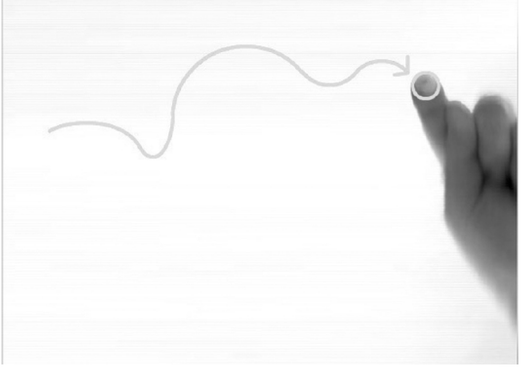


Luego, el gesto es traducido en el siguiente mensaje XML:

```
<NOTA ID='#'>
<A> altura </A>
<T> tiempo_inicio </T>
<D> duración </D>
</NOTA>
```

4.3. Diseño melódico

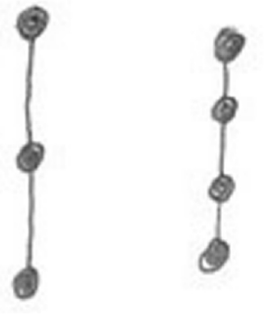
Este elemento se construye mediante la realización de un trazo continuo en la pantalla:



El mensaje XML que le correspone a este gesto (y su respectivo elemento), es de longitud variable, ya que la melodía podría estar formando por diferente cantidad de notas. El trazo continuo es traducido mediante un algoritmo de cuantización en una secuencia de notas:

```
<MELODIA ID='#'>
<CANTIDAD> cantidad_de_notas </CANTIDAD>
<NOTA>
<A> altura </A>
<T> tiempo_inicio </T>
<D> duración </D>
</NOTA>
<NOTA>
<A> altura </A>
<T> tiempo_inicio </T>
<D> duración </D>
</NOTA>
...
</MELODIA>
```

4.4. Acordes



Los acordes se construyen mediante el apoyo simultáneo de varios dedos, respetando una forma vertical. En casos de necesitar hacer acordes de varias notas, 4,5 ó más, es posible apoyar los dos primeros dedos y manteniendo el primero, ir agregando nuevas notas. ReacTiVision no distingue respecto de la mano a la que pertenece cada dedo, por lo que para la realización de este gesto es posible (y seguramente necesario) utilizar ambas manos:

Al igual que lo que sucede con los mensajes XML para las melodías, los mensajes para acordes pueden tener diferentes longitudes en función de la cantidad de notas que contengan:

```
<ACORDE ID='#'>
<CANTIDAD> cantidad_de_notas </CANTIDAD>
<NOTA>
<A> altura </A>
<T> tiempo_inicio </T>
<D> duración </D>
</NOTA>
<NOTA>
<A> altura </A>
<T> tiempo_inicio </T>
<D> duración </D>
</NOTA>
...
</ACORDE>
```

4.5. Arpegios

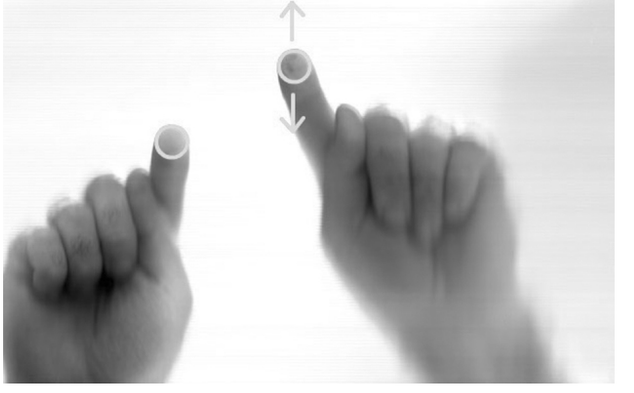
Este elemento se construye de forma similar al acorde, se apoyan varios dedos en forma vertical, pero luego, dejando algunos, se realiza un trazo zigzagueante en forma paralela:

De igual manera, la construcción del mensaje XML correspondiente al arpegio es similar al del acorde:

```
<ARPEGIO ID='#'>
<CANTIDAD> cantidad_de_notas </CANTIDAD>
<NOTA>
<A> altura </A>
<T> tiempo_inicio </T>
<D> duración </D>
</NOTA>
<NOTA>
<A> altura </A>
<T> tiempo_inicio </T>
<D> duración </D>
</NOTA>
...
</ARPEGIO>
```

4.6. Glissando

El glissando se construye mediante dos dedos que se colocan en forma horizontal, desplazando luego el del extremo derecho (hacia arriba o abajo) para establecer el grado del glissando:

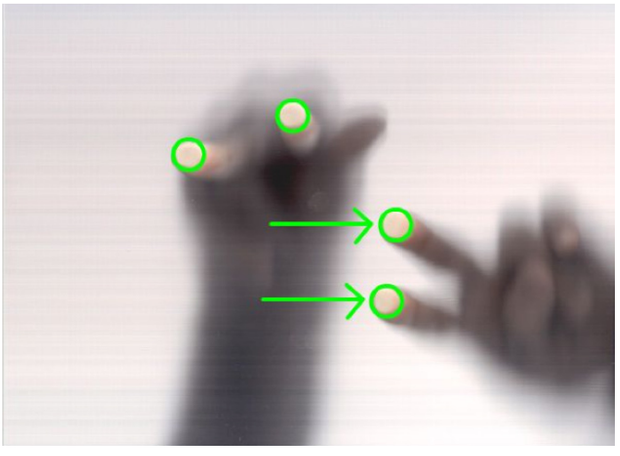


El mensaje XML correspondiente, expresa los puntos incial y final del glissando:

```
<GLISSANDO ID='#'>
<INICIO>
<A> altura_inicial </A>
<T> tiempo_inicial </T>
</INICIO>
<FINAL>
<A> altura_final </A>
<T> tiempo_final </T>
</FINAL>
</GLISSANDO>
```

4.7. Trémolo

El trémolo se construye apoyando dos dedos para establecer las alturas y moviendo otros dos dedos debajo, en forma horizontal para indicar que es un trémolo y no un glissando:

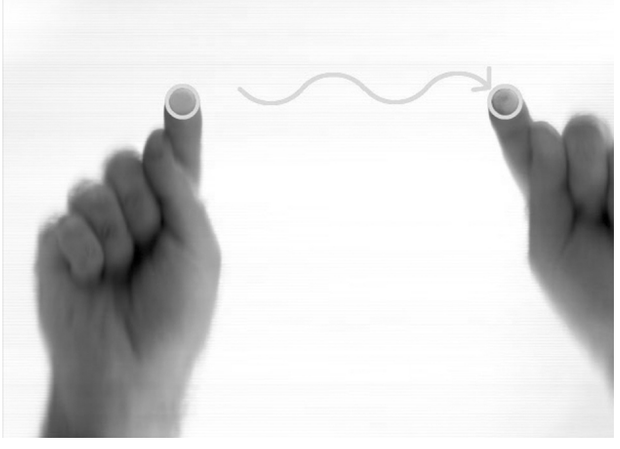


El mensaje XML del trémolo es:

```
<TREMOLO ID='#'>
<A> altura_inicial </A>
<T> tiempo_inicial </T>
<AF> altura_final </AF>
<D> duración </D>
</TREMOLO>
```

4.8. Trino

El trino se construye de forma similar a la nota tenida, solo que cuando el dedo a la derecha se desplaza, lo hace siguiendo un movimiento zigzagueante:



El mensaje XML del trino es similar al de la duración:

```
<TRINO ID='#'>
<A> altura</A>
<T> tiempo_inicio </T>
<D> duración </D>
</TRINO>
```

5. Algoritmo

Con el fin de interpretar los gestos descriptos anteriormente se diseñó el siguiente algoritmo, que es aquí presentado en la forma de un pseudo código que mezcla instrucciones estándares de cualquier lenguaje de programación junto a enunciados expresados en lenguaje coloquial. El algoritmo se presenta de esta manera para dejar en claro la noción de que el mismo podría ser implementado en cualquier lenguaje de programación (independientemente de nuestra preferencia por Processing). Durante el algoritmo se evaluan diferentes condiciones que permiten distinguir a cada uno de los gestos. Por ejemplo, una de las primeras condiciones a evaluar es la cantidad de dedos implicados en el gesto, ya que solo dos de los gestos utilizan un solo dedo. Otra de las condiciones se relaciona con la existencia y discriminación entre dedos quietos y en movimiento, así como la cantidad que hay de cada uno de estos tipos. Cuando hay movimiento es importante reconocer aquellos gestos que involucran un movimiento zigzagueante, tales como el trino o el arpegio. Otro tema importante es analizar el tipo de distribución que se expone cuando hay varios dedos, es decir, ver si es horizontal o vertical. Se presenta, entonces el siguiente algoritmo:

```
if( cantidad de dedos es mayor a 1 ){

  if( existen dedos móviles ){

    if( cantidad de dedos móviles es igual a 1 AND
    cantidad de dedos quietos es igual a 1 AND
    dedo movil se ubica a la derecha respecto del quieto )
{

      if( el dedo movil se desplaza en forma horizontal ){

        if( el dedo movil se desplaza en forma zigzaguean-
te ){

          TRINO( tomar nota inicial de la posicion x,y del
dedo quieto y
          y la distancia en x entre ambos dedos para la
duración )
        }else{

          DURACIÓN( tomar nota inicial de la posicion x,y
del dedo quieto y
          y la distancia en x entre ambos dedos para la
duración )
        }
      }else{

        GLISSANDO( tomar nota inicial de la posicion x,y
del dedo quieto y
```

```
        definir el glissando con la posición del dedo dere-
cho )
      }

    }else if( cantidad de dedos móviles es mayor a 2 AND
    cantidad de dedos quietos es igual a 2 AND
    dedo movil se ubican debajo respecto de los quietos ){

      if( los dedos móviles se desplazan en forma horizon-
tal ){

          TREMOLO( tomar nota inicial de la posicion x,y
del dedo quieto
          de la izquierda y la posicion y del otro dedo
quieto )
      }

    }else{

      if( la cantidad de dedos móviles es igual a 1 AND
      la distribución de los dedos quietos es vertical ){

        if( el dedo móvil se ubica a derecha AND
        el dedo movil tiene movimiento zigzagueante){

          ARPEGIO( tomar la posción horizontal del dedo
ubicado arriba y
          las posiciones verticales de cada dedo )
        }else{

          ERROR
        }
      }else{

        ERROR
      }
    }
  }else{

   if( los dedos tiene una distribución vertical ){

     ACORDE( tomar la posción horizontal del dedo ubicado
arriba y
     las posiciones verticales de cada dedo )
   }else{

     ERROR
   }
  }
}else{

  if( el dedo está quieto ){

    NOTA_PUNTUAL( tomar posición x,y para posición tempo-
ral y altura )
  }else{

    if( no hay superposición vertical ){

      DISEÑO_MELÓDICO( tomar recorrido )
    }else{

      ERROR
    }
  }
}
```

5.1. El análisis del movimiento

A la hora de ejecutar el algoritmo expuesto arriba, es necesario que el análisis se realice una vez que el gesto haya finalizado. Para especificar esto mejor, definiremos un gesto como lo que sucede en pantalla, desde que aparece algún dedo hasta que desaparecen todos, esto quiere decir que la aplicación debe iniciar el registro del gesto en el momento que la pantalla pase del vacio a la existencia de uno o varios dedos, luego debe registrar todo acontecimiento hasta el momento que terminó toda acción, una vez sucedido esto, el gesto se presentará al algoritmo de interpretación de gestos.

5.2. Las condiciones del algoritmos

Existe en el algoritmo ciertas condiciones que requieren su propio algoritmo de análisis. Estas condiciones son:

1. Dedos quietos versus dedos en movimiento
2. Desplazamientos horizontales versus desplazamientos verticales
3. Desplazamientos zigzagueantes versus desplazamientos directos
4. Superposición vertical

5.3. Quietud versus movimiento

De las condiciones presentadas, esta es quizás la más sencilla de evaluar. Pero una aclaración válida para todas es que el registro de los gestos se realiza guardando la posición inicial de estos (es decir, el punto en el que aparecen), y luego los desplazamientos relativos que se producen de un tiempo al siguiente (entendiendo que los tiempos de ejecución se producen en pasos discretos). Los registros de desplazamientos deben hacerse usando coordenadas polares, es decir, en términos de ángulos y distancias, en cambio las posiciones iniciales deben describirse con coordenadas cartesianas.

En función del tipo de registro descripto, la quietud puede analizarse como “distancias de desplazamientos” que son menores a un umbral. No se puede pretender que sean de valor cero, ya que el ruido de la

captura de movimiento hace que haya pequeños desplazamientos. Un valor de umbral permite discriminar el movimiento real respecto del ruido.

5.4. Desplazamientos horizontales versus desplazamientos verticales

Antes de intentar dilucidar la forma de establecer este análisis, cabe aclarar que existen trazos horizontales, otros verticales y, por último, un tipo de trazo que no pertenece a ninguna de estas dos categorías. Una de las variables principales que permite interpretar la horizontalidad de un trazo sería el "ángulo promedio" de los desplazamientos. Si este promedio se acerca a 0º, 180º ó 360º, es probable que el trazo sea horizontal, por supuesto que esto debe medirse nuevamente en función de un umbral. Cuando el ángulo promedio se diferencia por un valor menor al umbral, entonces se puede presuponer que es horizontal.

Sin embargo, surge aquí el problema de que ciertos trazos podrían tener este tipo de ángulo en promedio, pero en sus variaciones terminar no siendo horizontal. Para distinguir estos casos, además del "ángulo promedio", hay que analizar la "amplitud de los ángulos", es decir, cuál fue el ángulo mínimo y cuál el máximo. Los trazos claramente horizontales deben tener una amplitud menor a 90º (u otro valor que se crea conveniente estipular como umbral), entre el mínimo y el máximo, pero de seguro un trazo con amplitudes mayores a los 180º, no son trazos horizontales.

Una vez que se determina que un trazo no es horizontal, entonces queda determinar si es vertical o de otro tipo. El procedimiento es el mismo, pero esta vez los promedios debe acercarse a 90º o 270º.

5.5. Desplazamientos zigzagueantes versus desplazamientos directos

Para determinar si un trazo tiene un desplazamiento zigzagueante o directo, no sirve el ángulo promedio, ni la amplitud, sino que debe utilizarse la "varianza", es decir, las variaciones respecto del promedio. Si la varianza es muy baja, el trazo es directo, en cambio si esta supera cierto umbral, entonces el trazo es zigzagueante.

5.6. Algoritmo de análisis de trazos de dedos

El siguiente algoritmo muestra el análisis sobre un "arreglo" que contiene las posiciones en X e Y que ha recorrido el dedo. En función de esto se realiza un promedio ponderado, es decir, se suman los ángulos de cada punto multiplicados por el peso de su recorrido en el peso actual:

```
float factor = distancia[i] / totalDistancia;
promedio += angulo[i]*factor;
```

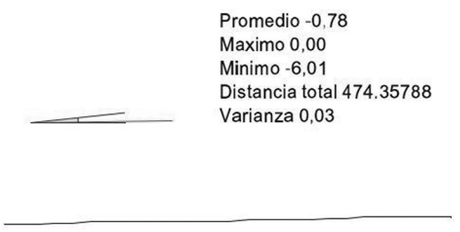
Para esto, es necesario obtener previamente la distancia total recorrida en el trazo.
El algoritmo completo:

```
        promedio = 0;
varianza = 0;
totalDistancia = 0;

boolean min_y_max_cargados = false;

for( int i=0 ; i<cantidad ; i++ ){
  totalDistancia += distancia[i];
}

if( totalDistancia > 0 ){

  for( int i=0 ; i<cantidad ; i++ ){

    float factor = distancia[i] / totalDistancia;
    promedio += angulo[i]*factor;

    if( distancia[i] > umbral_distancia ){
      if( min_y_max_cargados ){

        if( angulo[i]<minimo ){
          minimo = angulo[i];
        }
        if( angulo[i]>maximo ){
          maximo = angulo[i];
        }

      }
      else{
        minimo = angulo[i];
        maximo = angulo[i];
        min_y_max_cargados = true;
      }
    }

  }

  for( int i=0 ; i<cantidad ; i++ ){
    float factor = distancia[i] / totalDistancia;
float diferencia = abs( angulo[i] - promedio );
    varianza += diferencia*factor;

  }
 }
```

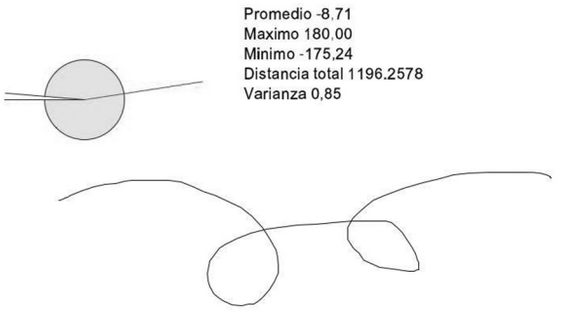
Puede ver los resultados de analizar diferentes curvas con este algoritmo.

A continuación se puede ver un trazo el línea casi recta. Su varianza es muy baja, así como sus extremos (mínimos y máximos) se encuentran cerca entre sí.

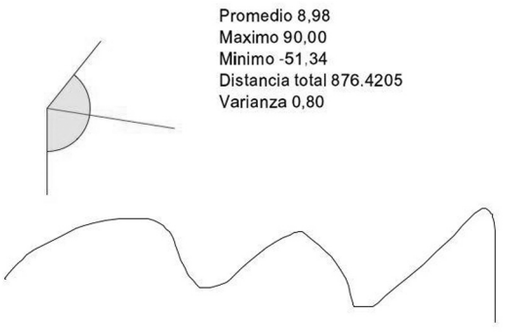
La siguiente es una curva que posee un promedio cercano a la curva anterior, pero sus extremos se encuentra muy lejanos, justamente producto de los bucles que posee:



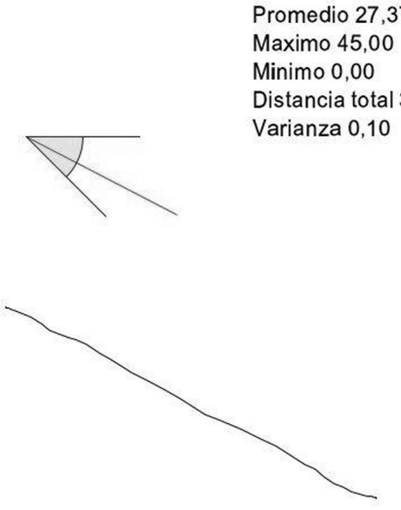
En la curva que sigue, los extremos están en un rango medio, pero su varianza sigue siendo alta en comparación con la primer curva:

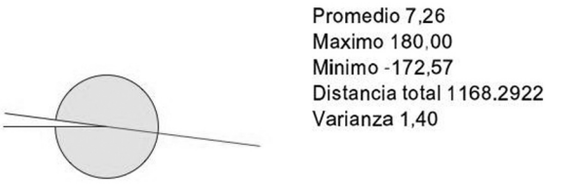


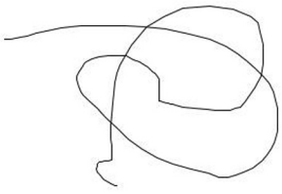
La siguiente curva muestra un trazo casi recto pero en otra dirección:



Por último, se puede ver una curva sin dirección definida, que produce una varianza y extremos, muy altos:







Las gráficas y sus respectivos análisis, nos muestran que se pueden hacer las siguientes asociaciones:

1. Desplazamientos horizontales versus desplazamientos verticales: pueden medirse en función del ángulo promedio, atendiendo que los extremos estén dentro de ciertos límites.

2. Desplazamientos zigzagueantes versus desplazamientos directos: puede medirse con la varianza, nuevamente atendiendo a los extremos.

6. Conclusión

El texto presentado, expone los criterios de diseño adoptados para la realización de esta aplicación, demostrando los algoritmos involucrados en la resolución de la interpretación gestual. Queda pendiente para posteriores trabajos el diseño de una interface para la creación de estructuras musicales a partir de los resultados obtenidos con la notación presentada.

7. Referencias bibliográficas

Manovich, Lev, *The Language of New Media* [Bibliografía], (Estados Unidos: MIT-Press, 1998)

Volpe, Gualterio. *Computational models of expressive gesture in multimedia systems* [Bibliografía], (Italia: InfoMus Lab.-Universidad de Génova, 2003).

<http://www.processing.org>.

<http://www.mtg.upf.es/reactable/>.

(*) Estos capítulos están extraídos del artículo "Desarrollo de una aplicación con interfaces tangibles" del mismo autor, y fueron agregados (a pesar de la obvia duplicación) debido a la pertinencia de sus contenidos para el presente texto.

Desarrollo de un sistema óptico para interfaces tangibles (mesa con pantalla reactiva)

Emiliano Causa

Palabras claves

Interfaces tangibles, realidad aumentada, sistema de proyección con espejos

1. Introducción

El objetivo de este texto es mostrar el desarrollo de una mesa para pantalla sensible al tacto e interfaces tangibles. La idea consiste en diseñar una mesa que logre maximizar el tamaño de la pantalla, conteniendo la altura de la mesa dentro de un parámetro ergonómico. El problema surge a partir de que las pantallas para interfaces tangibles utilizan un proyector de video para retroproyectar la imagen que emite la pantalla. El proyector se coloca dentro del mueble, debido a que el común de los proyectores de video posee una relacion de 1:1,4 a 1:1,8 (es decir, que para que el proyector, genere una imagen de 1 metro, requieren entre 1,5 y 1,8 metros de distancia a la superficie de proyector. Si pensamos que la pantalla debe estar en una posición horizontal, si queremos que la misma sea de 1 metro de ancho, se necesitaría que la mesa tenga entre 1,5 y 1,8 metros de alto. Obviamente una mesa con estas dimensiones no resulta práctica para el uso de una persona de altura promedio.

El siguiente capítulo expone el funcionamiento del sistema ReacTiVision, el mismo está extraído de otro artículo del autor y se duplica aquí a fines de ilustrar el sistema:

1.1. ReacTiVision y las interfaces tangibles (*)

ReacTiVision (mtg.upf.es/reactable/) es una herramienta de software desarrollada por Sergi Jordà, Martin Kaltenbrunner, Günter Geiger y Marcos Alonso, quienes conforman el Grupo de Tecnología Musical dentro del Instituto Audiovisual en la Universidad Pompeu Fabra (Barcelona España). Esta tecnología permite reconocer patrones bitonales (llamados “fiducials”) impresos a piezas de interfaces tangibles que funcionan sobre una pantalla sensible al tacto. Esta interface tangible consiste en piezas de acrílico que se apoyan sobre una pantalla sensible, ésta es capaz de reconocer las piezas, gracias a los patrones bitonales y generar respuestas audiovisuales en consecuencia. Los creadores, construyeron este software para desarrollar una pantalla sensible para interpretación musical en tiempo real (un instrumento de improvisación de música electrónica) llamada ReacTable.

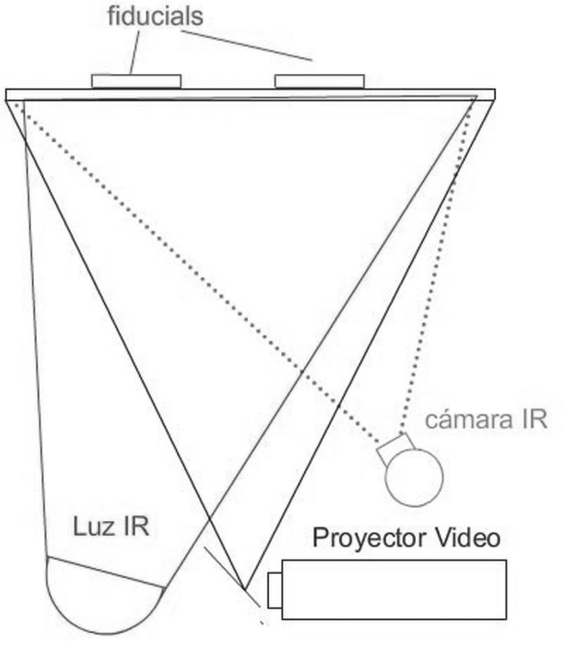


Figura: Esquema ReacTable

Como muestra la figura 3, ReacTiVision permite hacer el reconocimiento de patrones bitonales, a través de un sistema óptico, que en el caso de la ReacTable se implementa con luces y cámara infrarrojas. La pantalla es un acrílico con superficie esmerilada, las imágenes se retroproyectan desde abajo usando un cañón de video, a su vez una luz infrarroja permite iluminar los patrones que serán captados por una cámara, también infrarroja. Dicha luz y cámara son infrarrojas para no interferir la luz del proyector de video (que pertenece al rango visible de la luz), y para que la cámara no vea a su vez las proyecciones.

2. Modelo inicial

Como modelo inicial se partió de un sistema de tres

espejos que utilizamos en un proyecto personal (anterior a esta investigación) y que había mostrado eficacia para reducir la altura de una mesa con una pantalla de retroproyección, el sistema en cuestión utiliza otra tecnología de captura y la superficie de proyección es una tela elástica, pero el principio de reducción de altura y ampliación del tamaño de la pantalla es el mismo.

A partir de este modelo se tomó la decisión de ponerlo a prueba con un sistema que permitiera medir su eficiencia a la hora de maximizar el tamaño de la pantalla.

3. Requisitos para el diseño de la mesa

A la hora de diseñar la mesa existen un conjunto de restricciones que la misma debería respetar:

1. La altura de la mesa no debería sobrepasar los 115 cm.
2. La utilización de espejos no debería producir deformaciones en la imagen.
3. La utilización de espejos no debería producir reflejos que dupliquen porciones de la imagen (superposiciones).
4. Los espejos deberían estar ubicados de forma tal que no proyecten sombra sobre la imagen.
5. Los espejos deberían permitir ubicar al proyector en una posición horizontal (dado que algunos proyectores no pueden ser usados en posiciones oblícuas).

Para probar el diseño se desarrolló un algoritmo que permite visualizar (modelizar) el haz de luz del proyector y sus reflejos en el sistema de espejo.

4. Sistema de tres espejos

Como se pudo apreciar en el modelo original, un sistema de tres espejos permiten resolver algunos de estos requisitos.

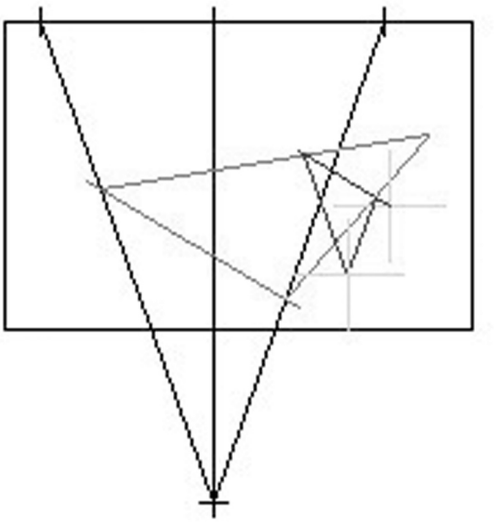
El espejo más pequeño permite reflejar el haz del proyector de forma tal que el dispositivo se encuentre horizontal y la imagen se emita en forma vertical.

Los espejos, medio y grande permiten desviar el haz para extender la longitud del mismo (es decir, la distancia entre el proyector y la pantalla) sin extender la altura de la mesa.

A continuación se puede ver una foto del modelo original y la forma en que los espejos funcionan:

La proyección entre el espejo medio y el grande puede producir un efecto de trapecio sobre la imagen deformándola. La forma de solucionar esto es logrando que ambos espejos estén ubicados en forma paralela, es decir, con el mismo ángulo, ya que la deformación que produce el espejo medio es corregida por el grande.

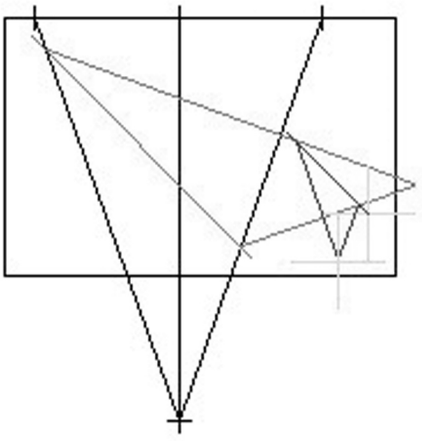
El objetivo del proyecto es hacer la mesa más pequeña posible que maximice el tamaño de la pantalla, por eso en el modelo computacional se partió de una caja de 112 cm (para dejar un margen para el apoyo en el piso) y se proyectó sobre su superficie una pantalla de 125cm de ancho. La relación entre ancho y distancia del haz se estableción en 1:1,4. Este haz se puede apreciar en el gráfico hecho por el modelo, es el triángulo de color negro. El vértice inferior corresponde a la posición en la que tendría que estar el proyector si no hubiese espejos.



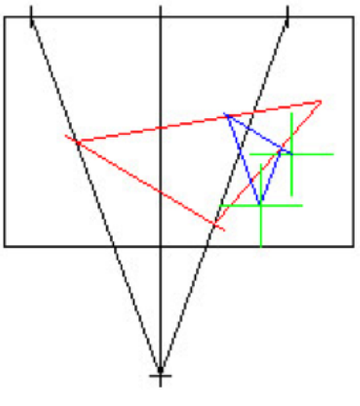
En rojo se grafica el espejo grande y la forma en que este refleja el haz. Aquí cabe aclarar que el sentido de la luz es inverso, es decir que la luz sale del proyector y viaja hacia la pantalla, pero en el modelo se parte desde la pantalla porque éste es un elemento constante (inamovible del sistema), mientras que las posiciones de los espejos y el proyector son los elementos variables del mismo.

Por último, en azul se puede ver al espejo medio y su reflejo. El vértice del haz azul muestra la posición del proyector. El espejo pequeño sirve a los fines de verticalizar el haz con el proyector puesto en forma horizontal. El tipo de modificación que produce en la longitud del haz es despreciable y no es necesario incluirlo en el sistema.

En el gráfico mostrado arriba, el ángulo de los espejos es 34,75°, esto permite que los mismos no generen sombras sobre el haz ni reflejos parásitos.



En esta nueva gráfica los espejos se ubican a 45,15° de ángulo. Con este nuevo ángulo, los tamaños de los espejos se agrandan de a 39 a 50 cm (espejo medio) y 97 a 134 cm (espejo grande), lo que encarece el mueble y reduce el espacio para el resto del equipo, a la vez que deja menos margen de error. Esto es importante dado que no siempre se cuenta con un proyector que respete la relación 1:1,4 y el margen de error permite absorver estos cambios.También se reduce el espacio para ubicar el proyector.



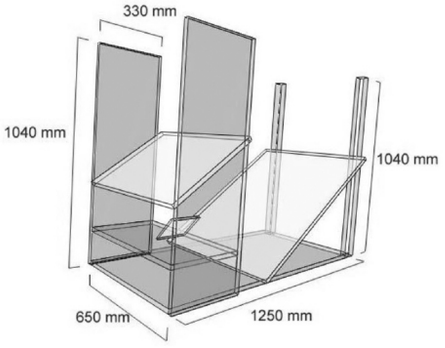
Aquí mostramos otro ejemplo en donde el ángulo se modificó a 30,57°. Como se puede ver en el gráfico, el espejo medio hace sombra en el haz que parte del espejo grande, y dada la posición es probable que el proyector también lo haga. Desde esta perspectiva, los ángulos cercanos a las 35° son buenos para un haz que sale de un proyector de relación 1:1,4.

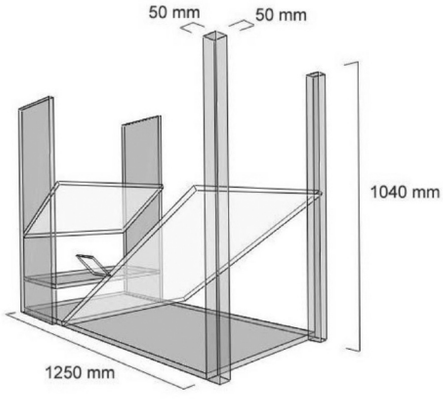
4.1. Del método

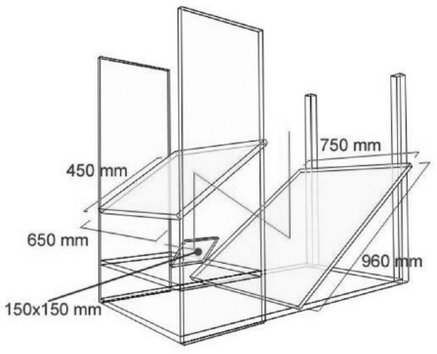
Esta forma de simulación permite probar diferentes ángulos y posiciones hasta entontrar los adecuados para cumplir con los requisitos del diseño. Es cierto que sería posible realizar un programa que busque por sí mismo dichas posiciones y ángulos haciendo el recorrido de un espacio solución, el equivalente a plantear un sistema de funciones a ser maximizadas, cuyas variables son las posiciones y los ángulos. Pero, dado que la mesa está pensada para tener cierta tolerancia respecto del tipo de proyector (específicamente, respecto de su relación ancho-profundidad), estas funciones son más complejas de ser definidas. En este punto, la complejidad y tiempo necesario para programar este algoritmo, puede ser reemplazado por un trabajo más sencillo y la utilización de ciertos criterios heurísticos, como se ha hecho en este caso.

4.2. Diseño de la estructura interna

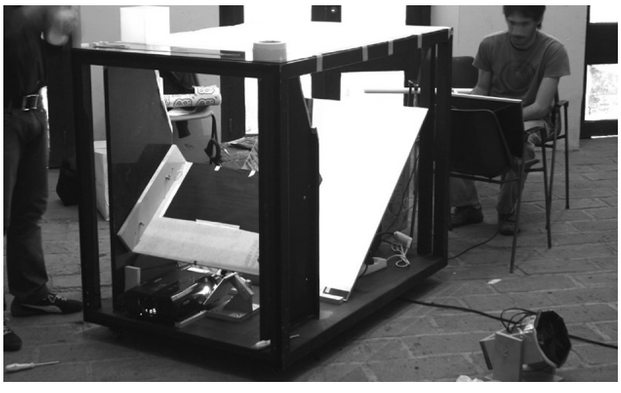
En función de los resultados obtenidos con el programa se diseño la estructura interna del mueble. A continuación puede verse una serie de gráficos que muestra sus medidas:



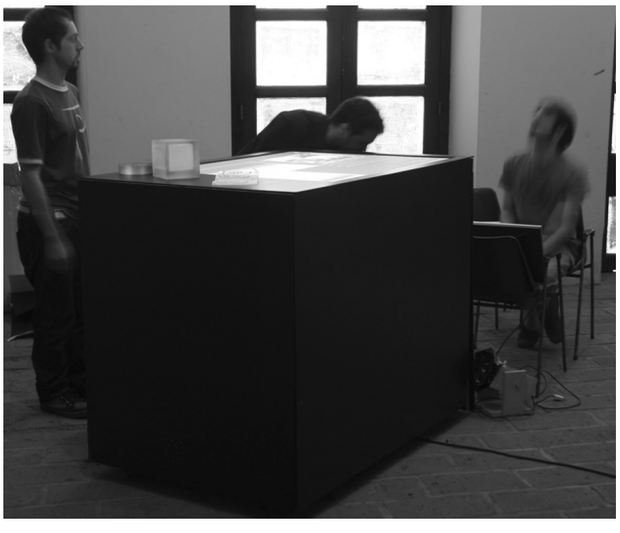


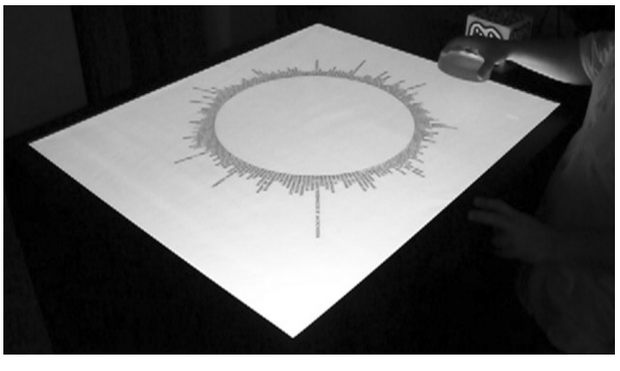


La construcción de la estructura, así como el diseño de la estructura externa del mueble estuvo a cargo de Diego Pimentel y Mariano Cataldi. Debajo se puede ver una fotografía del mueble construido antes de ser cerrado con sus tapas.



Las siguientes fotos muestran el mueble ya cerrado:





5. Deficiencias del diseño

El diseño realizado a partir del uso del programa resultó eficiente a la hora de realizar la proyección de la imagen, coincidiendo en la realidad con los resultados mostrados en el modelo computacional. Sin embargo aparecieron problemas en la implementación con aquellos elementos que no estuvieron en el modelo, particularmente con el sistema de captación.

5.1. Ubicación de la cámara

El primer problema surge a la hora de querer ubicar la cámara adecuada para poder captar la totalidad de la pantalla. El tamaño de la pantalla es de 125 cm de ancho, siendo que la cámara tiene una relación entre el ancho del cuadro y la distancia cercana a 1:1,10, la distancia a la que tendría que estar una cámara es aproximadamente de 137 cm. Obviamente la altura del mueble no alcanza y, por otra parte, la ubicación del espejo grande impide poner la cámara en esa posición, sin embargo, es posible ubicar a la cámara mirando hacia el espejo grande para extender la distancia a la pantalla.

En el complejo sistema de espejos, pensados para el proyector, las posiciones que la cámara puede usar no son las más adecuadas, obligando a obtener cuadros torcidos y deformados de la pantalla. Este defecto, por suerte es corregible con el sistema de calibración que utiliza ReacTiVision.

5.2. Resolución del sistema

Otra paradoja que surge del agrandamiento de la pantalla es que el tamaño que deben alcanzar los patrones (fiducials) es mayor, ya que la resolución de la cámara no alcanza para captar patrones pequeños. Esto no es en sí una deficiencia del diseño, ya que es corregible mediante el uso de cámaras de mayor calidad.

5.3. La superficie de la pantalla y los reflejos

Uno de los mayores focos de problema está en el material que se elige para usar de superficie de retroproyección (este tema es ampliamente tratado en el texto de Diego Pimentel). Esta superficie debe cumplir con una serie de requisitos complejos de lograr:

1. Resistencia: debe ser lo suficientemente resistente para permitir apoyar objetos y posar los dedos para interactuar (a esto se suma el hecho de que la altura de la mesa invita al público a apoyarse sobre esta).
2. Opacidad: debe ser lo suficientemente opaca para permitir ver nítidamente (desde abajo) solo los objetos apoyados, pero no aquellos por encima. Por ejemplos, debe poder verse los dedos apoyados, pero no la mano por encima de ellos, ni las caras de las personas.

3. Transparencia y nitidez: tambien debe poder ser lo suficientemente transparente para dejar ver la imagen proyectada por detrás, pero lo suficientemente opaca como para atrapar la luz y no dejarla pasar. Por ejemplo: si es muy transparente, la proyección seguirá de largo (proyectándose en el techo, en vez de en la pantalla) , si es muy opaca no podrá verse bien la imagen (hasta puede perder nitidez).

4. Antireflejo: la superficie debajo de la pantalla debe poseer un tratamiento antireflejo ya que el sistema de captura es muy sensible a los altos contrastes, principalmente si los producen los reflejos en esta superficie. Cuando la superficie no es antireflejos, funciona como un espejo que refleja el sistema de iluminación.

En nuestra implementación no pudimos encontrar una superficie que cumpla con todos estos requisitos. El material que mejor cumplía el segundo y tercer requisito es el papel vegetal (de tipo calco), pero claramente no cumplía el primer requisito, lo que obligó a apoyarlo sobre un acrílico que no cumplía con el cuarto requisito.

5.4. La iluminación

La iluminación que este sistema utiliza es infrarroja. Nosotros optamos por utilizar lámparas incandecentes filtradas con acetatos de color rojo y azul, que permiten generar luz infrarroja con focos comunes. Este tipo de iluminación tiene como ventaja su bajo costo y la facilidad de encontrar esos elementos en el mercado de nuestro país. Una de las más importantes desventajas es el calor que genera, lo que obliga a tomar medidas de ventilación. El sistema alternativo que más se utiliza es el de reflectores de *leds*, que no genera calor y tienen una mayor durabilidad.

El mayor problema encontrado con la iluminación fue con la superficie de la pantalla, que al no ser antireflejo funcionaba como un espejo más. Este pseudo espejo termina haciendo que la cámara vea las lámaras, lo que produce ruido en la captación de patrones. Debido a esto nos vimos obligados a poner las lámparas lo más apartadas hacia los lados posible, para que no entre en el área del reflejo, pero esta solución genera un tipo de iluminación desigual que complica nuevamente la captura.

5.5. Conclusión respecto de la deficiancias

Las deficiencias que hemos enunciado obligan a repensar en el futuro el diseño de este mueble, integrando a su evaluación estas nuevas variables que se han encontrado durante la implementación. Particularmente, la posición de la cámara, la posición y el tipo de iluminación, y el material de la superficie de proyección. Consideramos que los logros obtenidos en cuanto a la extensión del tamaño de la pantalla no terminan de justificar algunos de los problemas encontrados, sin embargo, la mayor parte del problema se relaciona con el tipo de material, y si este factor se resolviese el balance del diseño sería positivo y simplemente habría que corregir la posición de la cámara y la iluminación, que sería sencillo de resolver.

6. Algoritmo de cáculo de reflexiones

A continuación se muestra el algoritmo utilizado para el cálculo de la reflexiones. Este se organiza a través de una serie de objetos (clases) que son: Cono_Luz (que es usado para modelar el haz de luz), Segmento (utilizados para calcular las reflexiones) y Punto (para establecer ciertas posiciones en el modelo).

La principal parte del algoritmo es la función “reflejar” que permite establecer la reflexión de un haz de luz en un espejo:

```
void reflejar( Segmento espejo , Segmento original
, Segmento antes , Segmento despues ){

  float anguloEspejo = angulo( espejo );
  float anguloOriginal = angulo( original );

  float largoTotal = largo( original );

  Punto contacto = obtieneCruceDosLineas( espejo.a
, espejo.b , original.a , original.b );

  antes.iniciar( original.a , contacto );
  float largoAntes = largo( antes );

  float diferencia = anguloOriginal-anguloEspejo;

  float nuevoAngulo = anguloEspejo - diferencia;
  float largoDespues = largoTotal-largoAntes;

  despues.iniciar( contacto , largoDespues , nue-
voAngulo );
}
```

7. Referencias bibliográficas

<http://www.biopus.com.ar>.

<http://www mtg.upf.es/reactable/>.

<http://www.processing.org>.

(*) Este capítulo está extraído del artículo “Desarrollo de una Aplicación con Interfaces Tangibles” del mismo autor, y fue agregado

(a pesar de la obvia duplicación) debido a la pertinencia de sus contenidos para el presente texto

8. Anexo código de algoritmo

```
Cono_Luz cono;

float mx = 100;
float my = 300;

float botton = 0;
float top = -112;
float left = 0;
float right = 170;

void setup(){

  size( 800 , 600 );
  background( 255 );
  rectMode(CORNERS);
  noFill();

  translate( mx , my );

  float anchoPantalla = 125;
  float relacion = 1.4;
  cono = new Cono_Luz( anchoPantalla , relacion );

  float profundidad = cono.profundidad;
  float margen = 13;
  float xproyector = margen + anchoPantalla/2;
  float yproyector = profundidad + top;
  cono.ubicar( xproyector , yproyector , radians(270) );

  cono.dibujar();

  float correr = 0;
  float x1esp = 107+correr;
  float y1esp = -8.5;
  float x2esp = 27;
  float y2esp = -64+correr;

  Segmento espejo = new Segmento( x1esp , y1esp , x2esp , y2esp );

  float anguloEspejo = angulo( espejo );
  float tamanioEspejo = largo( espejo );

  println( "anguloEspejo = " + (180+degrees( anguloEspejo )) );
  println( "tamanioEspejo = " + tamanioEspejo );
```

```
  rect( left , top , right , botton );

  Segmento antes_1 , despues_1 , original_1 ;
  original_1 = new Segmento( cono.extremoR , cono.foco );
  antes_1 = new Segmento();
  despues_1 = new Segmento();
  reflejar( espejo , original_1 , antes_1 , despues_1 );

  Segmento antes_2 , despues_2 , original_2 ;
  original_2 = new Segmento( cono.extremoL , cono.foco );
  antes_2 = new Segmento();
  despues_2 = new Segmento();
  reflejar( espejo , original_2 , antes_2 , despues_2 );

  stroke(255,0,0);
  espejo.dibujar();
  despues_1.dibujar();
  despues_2.dibujar();

  Punto baseEspejo2 = new Punto( 158, -43 );
  float tamEspejo2 = 39;
  Segmento espejo2 = new Segmento(   baseEspejo2 , tamEspejo2 , anguloEspejo );

   Segmento antes_3 , despues_3 , original_3 ;
  antes_3 = new Segmento();
  despues_3 = new Segmento();
  reflejar( espejo2 , despues_1 , antes_3 , despues_3 );

  Segmento antes_4 , despues_4 , original_4 ;
  antes_4 = new Segmento();
  despues_4 = new Segmento();
  reflejar( espejo2 , despues_2 , antes_4 , despues_4 );

  baseEspejo2.dibujarCruz();
  stroke(0,0,255);
  espejo2.dibujar();
  despues_3.dibujar();
  despues_4.dibujar();

  despues_4.b.dibujarCruz();
  despues_4.b.string();
}
void draw(){

}

//CLASE CONO DE LUZ

class Cono_Luz{
```

```
float relacion;
float anchoPantalla;
float profundidad;

float x,y,angulo;

Punto foco, centro, extremoR , extremoL;

//-----------------------------------------

Cono_Luz( float anchoPantalla_ , float relacion_ ){

  anchoPantalla = anchoPantalla_;
  relacion = relacion_;
  profundidad = anchoPantalla * relacion;

  ubicar(0,0,0);
  calcularCono();
}
//-----------------------------------------

void ubicar( float x_ , float y_ , float angulo_ ){
  x = x_;
  y = y_;
  angulo = angulo_;
  calcularCono();
}
//-----------------------------------------

void calcularCono(){

  foco = new Punto( x , y );
  centro = puntoUbicadoA( foco , profundidad , angulo );
  extremoR = puntoUbicadoA( centro , anchoPantalla/2 , angulo + HALF_PI );
  extremoL = puntoUbicadoA( centro , anchoPantalla/2 , angulo - HALF_PI );
}
//-----------------------------------------

void dibujar(){
  foco.dibujarCruz( color(0,0,0) , 5 );
  centro.dibujarCruz( color(0,0,0) , 5 );
  extremoR.dibujarCruz( color(0,0,0) , 5 );
  extremoL.dibujarCruz( color(0,0,0) , 5 );

  dibujaLinea( foco , extremoR );
  dibujaLinea( foco , extremoL );
  dibujaLinea( extremoL , extremoR );
  dibujaLinea( foco , centro );
```

```
  }
}

//--------------------------------------------------------------
// FUNCIONES MATEMÁTICAS
//--------------------------------------------------------------

float linea( float x , float x1 , float x2 , float y1 , float y2 ){
  return (x-x1)/(x2-x1)*(y2-y1)+y1;
}
//--------------------------------------------------------------

float lineaLim( float x , float x1 , float x2 , float y1 , float y2 ){
  float valor = (x-x1)/(x2-x1)*(y2-y1)+y1;
  valor = min(valor,y2);
  valor = max(valor,y1);
  return valor;
}
//--------------------------------------------------------------

float menorDistAngulos( float origen , float destino ){
  float distancia = destino - origen;
  return anguloRangoPI( distancia );
}
//--------------------------------------------------------------

float anguloRango2PI( float angulo ){
  float este = angulo;

  for( int i=0 ; i<100 ; i++ ){
    if( este >= TWO_PI ){
      este -= TWO_PI;
    }
    else if( este < 0 ){
      este += TWO_PI;
    }
    if( este >= 0 && este <= TWO_PI ){
      break;
    }
  }
  return este;
}
//--------------------------------------------------------------

float anguloRangoMenos2PI( float angulo ){
  float este = angulo;

  for( int i=0 ; i<100 ; i++ ){
    if( este > 0 ){
```

```
      este -= TWO_PI;
    }
    else if( este <= -TWO_PI ){
      este += TWO_PI;
    }
    if( este > - TWO_PI && este <= 0 ){
      break;
    }
  }
  return este;
}
//--------------------------------------------------------------------

float anguloRangoPI( float angulo ){
  float este = angulo;
  for( int i=0 ; i<100 ; i++ ){
    if( este > PI ){
      este -= TWO_PI;
    }
    else if( este <= -PI ){
      este += TWO_PI;
    }
    if( este >= -PI && este <= PI ){
      break;
    }
  }
  return este;
}
//--------------------------------------------------------------------
// CLASE: SEGMENTO
//--------------------------------------------------------------------
class Segmento{
  Punto a,b;
  //-------------------------

  Segmento(){
    a = new Punto();
    b = new Punto();
  }
  //-------------------------

  Segmento( Punto a_ , Punto b_ ){
    iniciar ( a_ , b_ );
  }
  //-------------------------

  Segmento( float x1 , float y1 , float x2 , float y2 ){
    iniciar( x1 , y1 , x2 , y2 );
  }
```

```
  //--------------------------

  Segmento( Punto a_ , float distancia , float angulo ){
    iniciar( a_ , distancia , angulo );
  }
  //--------------------------

  void iniciar( float x1 , float y1 , float x2 , float y2 ){
    a = new Punto( x1 , y1 );
    b = new Punto( x2 , y2 );
  }
  //--------------------------

  void iniciar( Punto a_ , float distancia , float angulo ){
    a = new Punto();
    a.copiarDe( a_ );
    b = puntoUbicadoA( a , distancia , angulo );
  }
  //--------------------------

  void iniciar ( Punto a_ , Punto b_ ){
    a = new Punto();
    b = new Punto();
    a.copiarDe( a_ );
    b.copiarDe( b_ );
  }
  //--------------------------

  void dibujar(){
    line( a.x , a.y , b.x , b.y );
  }
}
//--------------------------
float largo( Segmento este ){
  return dist( este.a.x , este.a.y , este.b.x , este.b.y );
}
//--------------------------
float angulo( Segmento este ){
  return atan2( este.b.y - este.a.y , este.b.x - este.a.x );
}
//--------------------------
void reflejar( Segmento espejo , Segmento original , Segmento antes , Segmento
despues ){

  float anguloEspejo = angulo( espejo );
  float anguloOriginal = angulo( original );

  float largoTotal = largo( original );
```

```
  Punto contacto = obtieneCruceDosLineas( espejo.a , espejo.b , original.a ,
original.b );

  antes.iniciar( original.a , contacto );
  float largoAntes = largo( antes );

  float diferencia = anguloOriginal-anguloEspejo;

  float nuevoAngulo = anguloEspejo - diferencia;
  float largoDespues = largoTotal-largoAntes;

  despues.iniciar( contacto , largoDespues , nuevoAngulo );
}
//-----------------------------------------------------------------
// CLASE: PUNTO
//-----------------------------------------------------------------
class Punto{
  float x,y;
  //--------------------------

  Punto(){
    iniciar( 0 , 0 );
  }
  //--------------------------

  Punto( float x_ , float y_ ){
    iniciar( x_ , y_ );
  }
  //--------------------------

  void iniciar( float x_ , float y_ ){
    x = x_;
    y = y_;
  }
  //--------------------------

  void copiarDe( Punto otro ){
    x = otro.x;
    y = otro.y;
  }
  //--------------------------
  void string(){
    println( "(  "+x+"  ;   "+y+"  )");
  }
  //--------------------------
  void dibujarCruz( ){
    dibujarCruz( color(0,255,0) , 20 );
  }
  //--------------------------
```

```
  void dibujarCirculo( color esteCol , float ancho ){
    stroke( esteCol );
    ellipse( x , y , ancho*2 , ancho*2 );
  }
  //-------------------------

  void dibujarCruz( color esteCol , float ancho ){
    stroke( esteCol );
    line( x-ancho , y , x+ancho , y );
    line( x , y-ancho , x , y+ancho );
  }

}
//--------------------------------
Punto puntoUbicadoA( Punto a , float distancia , float angulo ){
  Punto aux = new Punto();

  float xx = a.x + distancia * cos( angulo );
  float yy = a.y + distancia * sin( angulo );

  aux.iniciar( xx , yy );

  return aux;
}
//--------------------------------
float distancia( Punto a , Punto b ){
  return dist( a.x , a.y , b.x , b.y );
}
//--------------------------------
void dibujaLinea( Punto a , Punto b ){
  line( a.x , a.y , b.x , b.y );
}
//--------------------------------
Punto obtieneCruceDosLineas( Punto p1 , Punto p2 , Punto p3 , Punto p4 ){
  Punto aux = new Punto();

  float x = 0;
  float y = 0;

  float a = p2.y - p1.y;
  float b = p2.x - p1.x;
  float c = p4.y - p3.y;
  float d = p4.x - p3.x;

  if( b != 0 && d != 0 ){

    float ab = a/b;
    float cd = c/d;
```

```
    float e = p3.y - p1.y + p1.x*ab - p3.x*cd;

    float p = ab-cd;

    if( p != 0){
      x = e / p;
      y = linea( x , p1.x , p2.x , p1.y , p2.y );
    }

  }
  else{
    if( b == 0 && d != 0 ){
      x = p1.x;
      y = linea( x , p3.x , p4.x , p3.y , p4.y );
    }
    else if( b != 0 && d == 0 ){
      x = p3.x;
      y = linea( x , p1.x , p2.x , p1.y , p2.y );
    }
  }

  aux.iniciar(x,y);

  return aux;
}
```

MATÍAS ROMERO COSTAS

Compositor, artista multimedia, docente e investigador. Licenciado en Composición; profesor de Armonía, Contrapunto y Morfología Musical; y profesor en Producción Multimedial, diplomado en la Universidad Nacional de La Plata.

Trabaja como docente de grado, posgrado e investigador en la UNLP, el IUNA y ENERC (INCAA).

Forma parte del grupo “Proyecto Biopus” con el que ha presentado obras de net-art, video arte, *performances* e instalaciones interactivas. Sus obras han sido presentadas en Argentina y el exterior. Ha recibido premios nacionales e internacionales.

Técnicas de síntesis y procesamiento de sonido y su aplicación en tiempo real

Matías Romero Costas

Palabras claves

Audio digital, síntesis de sonido, procesamiento, tiempo real

Resumen

En el presente artículo estudiaremos algunas de las técnicas de síntesis y procesamiento de sonido más utilizadas y difundidas, desde un punto de vista teórico y a través de ejemplos de aplicación, implementados en el entorno de programación Max-MSP. Este software fue creado originalmente en el IRCAM en 1996, y desarrollado actualmente por Cycling74, es un entorno de programación pensado como una herramienta para la composición algorítmica. La elección de este entorno se debe a que es una herramienta muy poderosa para síntesis, generación, procesamiento y ejecución de sonido, música e imagen en tiempo real. Por otro lado, su lenguaje de programación con objetos gráficos lo convierte en una herramienta muy didáctica para abordar estos temas.

Finalmente analizaremos la programación de una instalación sonora interactiva, donde se combinan varios de estos módulos de generación y procesamiento, para crear un instrumento complejo.

1. Síntesis aditiva

La síntesis aditiva es una de las primeras técnicas de síntesis, y tiene sus orígenes en la música electrónica.

Esta técnica encuentra sus fundamentos en los descubrimientos del físico francés Jean Baptiste Joseph Fourier (1768-1830), quien reveló que cualquier onda compleja periódica puede ser descompuesta en una serie de ondas simples o sinusoidales. Basada en estos principios, la síntesis aditiva consiste en construir sonidos complejos a partir de la sumatoria de sonidos sinusoidales de diferentes amplitudes, frecuencias y fases.

La síntesis aditiva, expresada en forma de ecuación, queda definida como:

$$y(n) = \sum_{n=0}^{N} A_n \cos(\omega_n + \theta_n)$$

Donde n es el número de muestra, An es la amplitud de la muestra n, ω es su frecuencia angular y θ su fase inicial.

Otra manera de entenderlo es a través de un diagrama de flujo[1]:

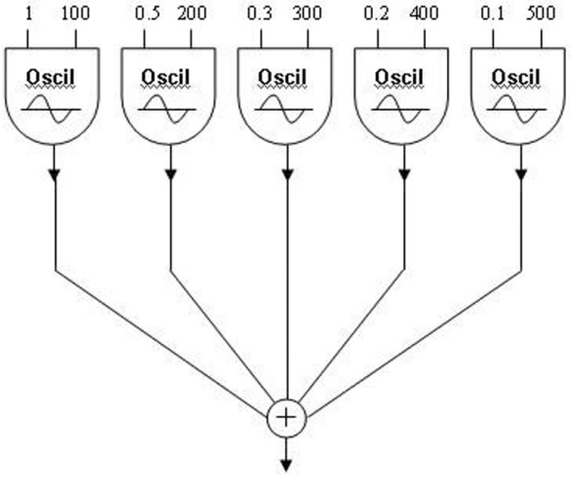


Figura 1: esquema o gráfico de flujo de un instrumento de síntesis aditiva

1.1. Osciladores

El elemento fundamental utilizado en la síntesis aditiva, al igual que en muchas otras técnicas de síntesis, es el oscilador.

Un oscilador digital es una unidad generadora que lee cíclicamente una función para generar una onda periódica. Existen dos formas de implementar un oscilador, una es especificar la forma de onda como una función matemática dependiente del tiempo. Para ello, el algoritmo tiene que calcular el valor de la función matemática para cada una de las muestras. Por ejemplo, si queremos generar una onda sinusoidal, la función que debe calcularse para cada una de las muestras es la siguiente:

$$y(n) = Aseno\ (2\pi f\ \frac{n}{sr} + \theta)$$

En donde: n es el número de muestra; A es la amplitud; f la frecuencia (Hz); sr es la frecuencia de muestreo (*sampling rate*) y θ es la fase.

Otro método utilizado, más veloz y con menor consumo de procesamiento, es el llamado oscilador de acceso a tablas (*wavetable lookup oscilator*), que consiste en almacenar en memoria un solo ciclo de forma de onda cualquiera. En ese bloque, o tabla, se almacenan una secuencia de números, cada una de las cuales corresponde al valor de amplitud de cada muestra de la forma de onda. El oscilador, por tanto, solo debe leer de esta tabla los valores sucesivos para generar la forma de onda correspondiente. Una vez que se lee la última muestra, el oscilador vuelve a comenzar desde el principio.

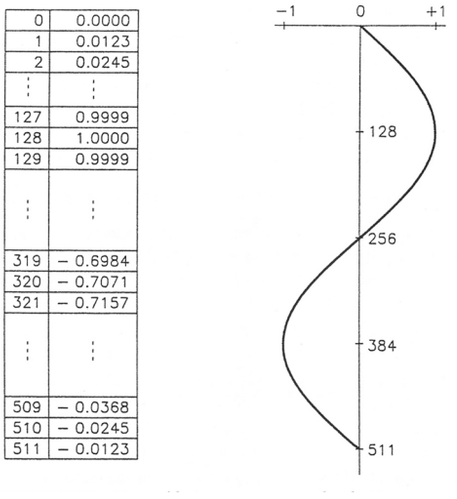


Figura 2: tabla de onda (izquierda) de una sinusoide (derecha)

En la Figura 2, observamos una tabla de onda (wavetable) cuyo tamaño es de 512 muestras. La primera columna indica los índices de cada muestra y la segunda los valores de amplitud asociados a ellas. Para obtener los valores de amplitud que representan a la onda sinusoidal, el oscilador vinculado a la tabla accede, mediante un puntero, a cada uno de los índices de la tabla. En este caso los valores de amplitud están "normalizados", es decir que están ajustados en el rango que va entre 0 y 1. Pero también, al ser esta una onda sinusoidal bipolar, su hemiciclo negativo se encuentra entre 0 y -1. Es decir, que el oscilador que lea esta tabla devolverá a una velocidad igual a la frecuencia de muestreo valores de amplitud entre 1 y -1 que representan un ciclo de una función senoidal.

En el siguiente esquema vemos la representación de un oscilador en el que se muestra la unidad generadora con un símbolo en su interior que designa la forma de onda. El valor que ingresa por la izquierda representa la amplitud pico de la señal; el valor de la derecha, la frecuencia a la que se repite el ciclo de la onda. Otro parámetro posible a determinar es la fase, que establece en qué punto de la onda el oscilador comienza a leer las muestras.

Figura 3: esquema de un oscilador. En este caso una sinusoide

1.2. Implementación de la síntesis aditiva en tiempo real

En el gráfico siguiente observamos la programación de una síntesis aditiva con cinco osciladores, y a la derecha la representación gráfica de su espectro, su forma de onda y su sonograma.

En la parte superior, los objetos de mensajes especifican la frecuencia fundamental del sonido resultante (100 Hz, 350 Hz y 1500 Hz). Esta frecuencia es luego multiplicada por 1, 2, 3, 4 y 5 respectivamente para establecer la frecuencia de los parciales de cada uno de los cinco osciladores, en este caso en relación armónica (múltiplos enteros con respecto a la fundamental).

Si quisiéramos obtener un espectro inarmónico bastaría modificar los valores de la multiplicación por números reales.

Antes de sumarse en la salida, los armónicos son multiplicados por un factor de 0.2 para evitar la saturación.

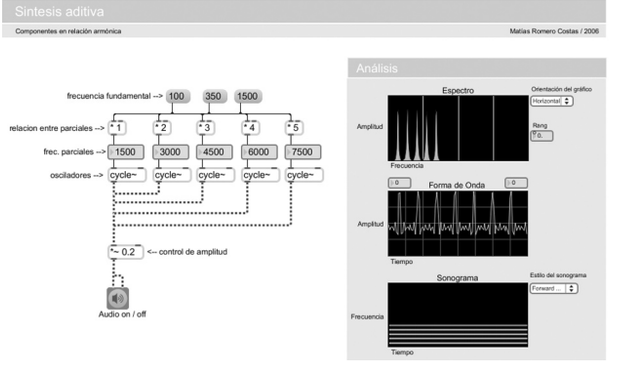


Figura 4

El ejemplo que sigue incluye el control de la amplitud de cada uno de los componentes del espectro a partir de una envolvente dinámica, cuyo perfil puede editarse de manera muy simple con el mouse.

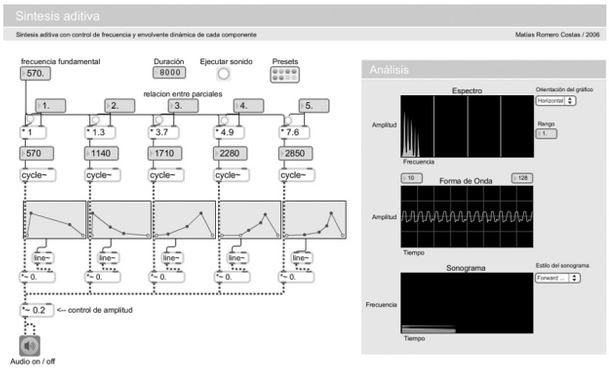


Figura 5

Si analizamos cualquier sonido de la naturaleza, o los que provienen de los instrumentos musicales, vamos a encontrar un espectro sumamente complejo, con cientos o miles de componentes, donde cada uno de ellos evoluciona de manera diferente a lo largo del tiempo. El ejemplo anterior solo intenta demostrar el funcionamiento básico de la síntesis aditiva, y para conseguir un resultado sonoro realmente interesante, se debería utilizar una cantidad mucho mayor de osciladores.

2. Modulación de amplitud

La variación periódica o cuasi periódica de la amplitud de una señal, denominada portadora, en función de la variación de otra, la moduladora, es conocida como modulación de amplitud.

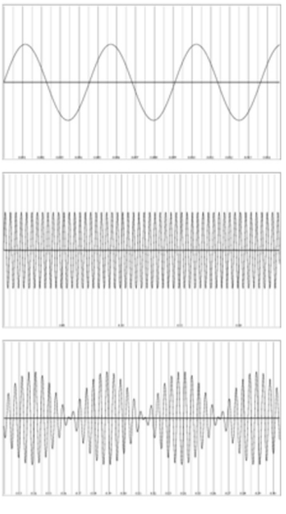


Figura 6: de arriba hacia abajo, señal moduladora, señal portadora y resultado de la modulación.

En audio digital, la manera de implementar la modulación es multiplicando las dos formas de onda, la portadora y la moduladora. Como vimos en el caso de la síntesis aditiva, cuando dos o más formas de onda se suman, lo mismo sucede con sus espectros. El espectro resultante se conforma con la suma de cada una de las frecuencias de los componentes simples. Sin embargo, cuando multiplicamos dos formas de onda el resultado en el espectro no se corresponde con la multiplicación de ambos, sino con su convolución.

Para poder entenderlo tomemos por caso la multiplicación de dos sinusoides de frecuencias $f1$ y $f2$. El espectro resultante estará conformado por dos componentes iguales a:

$$f1 + f2$$
$$f1 - f2$$

En la Figura 7 aparecen, en la parte superior, los espectros de las señales a ser convolucionadas, cuyas frecuencias son, respectivamente $f1 = 1000$ Hz y $f2 = 100$ Hz. El resultado de la convolución, en la parte inferior, nos muestra un espectro complejo, con dos parciales de 900 y 1100 Hz, ubicados a los lados de $f1$, y a una distancia igual a $f2$, a la derecha e izquierda de esta.

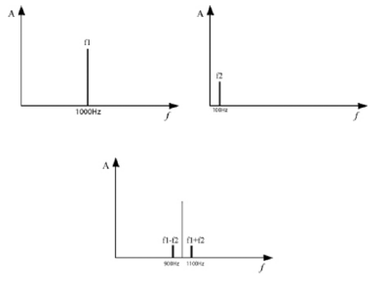


Figura 7: convolución de dos espectros simples.

En el caso de que la moduladora sea de baja frecuencia, por ejemplo 1 Hz, lo que escuchamos no es un espectro complejo sino una sola componente, cuya frecuencia es el promedio de las dos anteriores, con un batido de 1Hz. Este fenómeno solo sucede en el nivel perceptivo y está relacionado con los umbrales de percepción de la frecuencia.

2.1. Trémolo

Se utiliza este término en música para designar el modo de ejecución de algunos instrumentos acústicos, y consiste en alterar la amplitud del sonido repetidamente a gran velocidad. La velocidad de la modulación, al ser controlada por un LFO[2] (velocidad de control), está en el rango de entre 0.5 seg. y 1 seg. Esto quiere decir que lo que se ve afectado al nivel perceptivo es la cualidad de superficie del sonido.

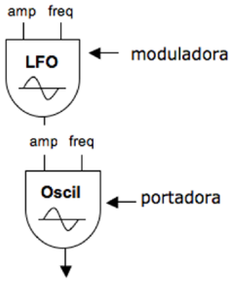


Figura 8: diagrama de flujo del trémolo

En un instrumento, los parámetros que suelen manejarse en la modulación de amplitud son:

- La velocidad de la modulación, controlada por la frecuencia de la moduladora.
- La profundidad de la modulación, que se corresponde con la amplitud de la moduladora.
- La silueta, determinada por la forma de onda de la moduladora.

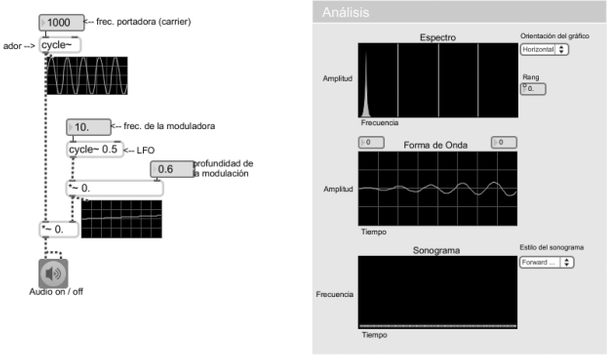


Figura 9: implementación en Max MSP de un trémolo

Si observamos con detenimiento el ejemplo anterior, podemos notar que la profundidad de modulación no está del todo bien resuelta, veamos porqué. Si la amplitud máxima de la portadora es igual a 1, y la profundidad de la modulación es igual a 0.5, los valores de amplitud máximos que resulten de la modulación estarán en el rango de 0 a 0.5. Pero, si queremos que la profundidad sea mayor, sin afectar la amplitud máxima de la portadora, es necesario invertir esta relación. De esta manera nos aseguramos que la modulación "cale" más o menos profundo en la silueta de la señal portadora. Esto queda resuelto en el ejemplo que sigue.

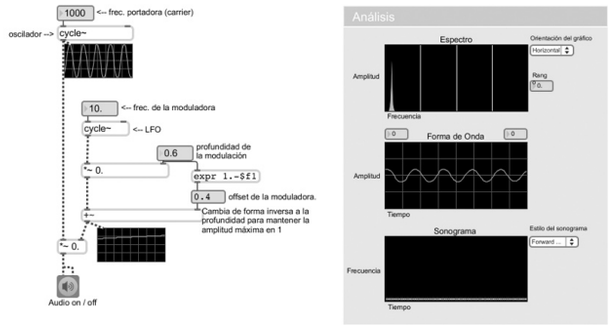


Figura 10: profundidad de modulación normalizada

2.2. Modulación en anillo

Cuando la modulación de amplitud se genera utilizando una moduladora cuya frecuencia se encuentra en rango audible (mayor a 20 Hz) se lo denomina modulación en anillo. Como puede deducirse, la envolvente afecta una porción temporal de la señal portadora que está en el orden de entre los 0.5 segundos y los 0.0005 segundos.

La aplicación que veremos a continuación es una modulación en anillo aplicada sobre un archivo de audio levantado del disco rígido y ejecutado por el objeto sfplay~.

Sabemos que la multiplicación de dos formas de onda es equivalente a la convolución de sus espectros, y hemos visto que en el caso de multiplicar dos sinusoides con frecuencias $f1$ y $f2$, obtenemos un espectro con dos parciales iguales a $f1$ - $f2$ y $f1$ + $f2$. En el caso de multiplicar un sonido complejo con una sinusoide, la operación de suma y resta se produce por cada uno de los componentes de la portadora. De esta forma se duplica la cantidad de parciales, y dependiendo de la frecuencia de la moduladora, esta técnica permite convertir muy fácilmente un sonido armónico en inarmónico.

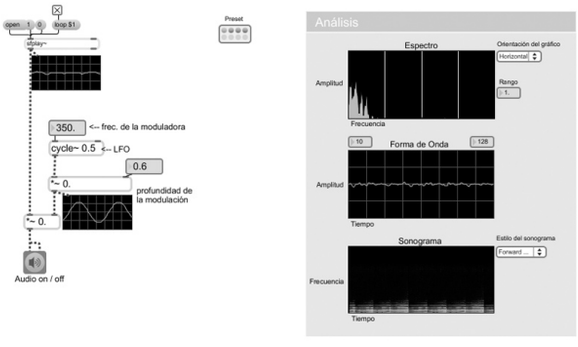


Figura 11: modulación en anillo sobre un archivo de audio

Si trabajamos con señales periódicas, existen ciertas relaciones entre la frecuencia fundamental de la portadora y la frecuencia de la moduladora que tienen resultados muy interesantes. Por ejemplo, si la frecuencia de la moduladora es la mitad de la frecuencia fundamental de la portadora, el espectro resultante solo va a contener armónicos impares.

Otro caso especial se produce cuando la frecuencia de la moduladora es múltiplo de la fundamental de la portadora. Allí el resultado es una transposición de la envolvente espectral.

3. Síntesis por FM

Es posible utilizar una señal moduladora para para crear desviaciones periódicas de frecuencia sobre otro oscilador. Si el modulador se comporta como un LFO, el efecto es el de una variación periódica de la altura, lo que se conoce como vibrato, por su relación con los instrumentos acústicos. Pero, el ingeniero y músico John Chowning descubrió que si el modulador oscilaba

a frecuencia de audio ya no se producía una modificación de la frecuencia sino del espectro.

En el caso de modular dos sinusoides entre sí, se producen una serie de bandas laterales alrededor de la portadora. Estas bandas aparecen equidistantes entre sí, a distancia de múltiplos enteros de la frecuencia de la moduladora.

$$f_c + - kf_m$$

Donde f_c es la frecuencia de la portadora (carrier), f_m la frecuencia de la moduladora, y k es un índice entero que varía de o a infinito.

El ancho de banda del espectro resultante se puede determinar a partir del índice de modulación definida por la relación entre la cantidad de desviación en frecuencia *D* (lo que equivale a la profundidad de modulación) y la frecuencia de la moduladora f_m.

Si bien sabemos que el espectro resultante de la FM es infinito, existen solo algunas frecuencias con amplitud significativa. El número de estas bandas es aproximadamente *I* + 2.

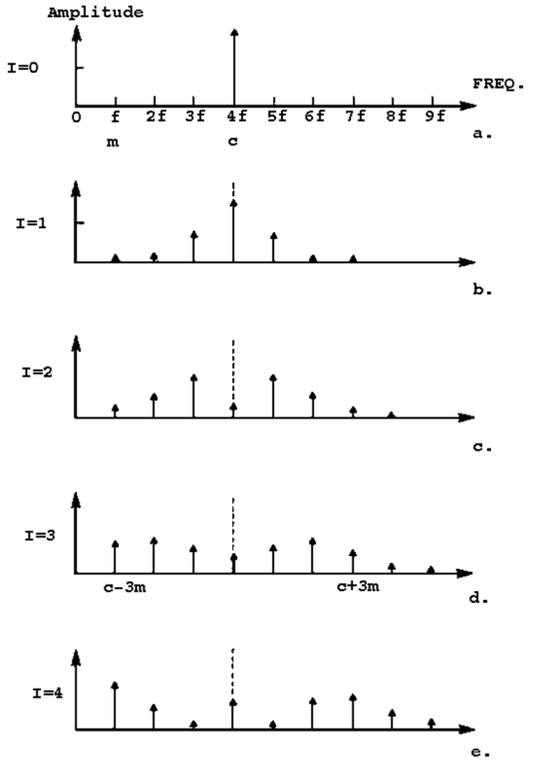


Figura 12: espectro de la FM a medida que crece el índice de modulación I

Las amplitudes de las bandas laterales J_n (I) pueden calcularse , una función de Bessel de primera clase de orden n, evaluada en función del índice de modulación I.

Al ser infinitas, hay ciertas bandas que se reflejan a los extremos del espectro o Hz y r/2 (frecuencia de Nyquist). La amplitud de dichas bandas cambian de signo al reflejarse.

Un parámetro muy útil al manejar una FM es lo que suele llamarse grado de armonicidad, determinado por la razón entre la frecuencia de la moduladora y de la portadora.

$$H = f_m / f_c$$

Relaciones enteras devuelven espectros armónicos, en cambio razones más complejas devienen en espectros inarmónicos.

3.1. Instrumento de FM simple

En la Figura 14 vemos una FM simple, donde los parámetros que puede definir el usuario son: la duración total del sonido, la frecuencia de la portadora, el grado de armonicidad y el índice de modulación, y la envolvente dinámica de la onda resultante.

Sabemos que el índice de modulación determina la complejidad del espectro. Si variamos el espectro en función del tiempo logramos una evolución espectral más cercana a lo que sucede en los sonidos instrumentales o de la naturaleza. El objeto envelope permite editar muy fácilmente dicha envolvente a partir de nodos creados con el mouse.

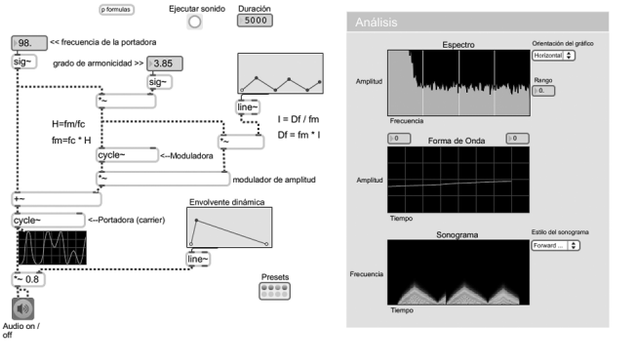


Figura 13

4. Síntesis por modelado físico

La síntesis por modelado físico se vale de modelos matemáticos para simular fuentes reales, en especial instrumentos musicales. Las ecuaciones utilizadas representan leyes físicas que describen, por ejemplo, la forma de producción del sonido, las dimensiones y los materiales del instrumentos, entre otros.

Los modelos de síntesis fueron mejorando, desde los primeros de Hiller y Ruiz, pasando por el algoritmo de Karplus-Strong, y los posteriores desarrollos de Julius O. Smith con la digital waveguide synthesis.

4.1. Algoritmo de Karplus-Strong en Max MSP

Kevin Karplus y Alex Strong desarrollaron, a finales de los años 70, un modelo de síntesis para emular el sonido de las cuerdas pulsadas. El algoritmo está conformado con una línea de retardo retroalimentada, y un filtro pasa bajos dentro del loop de realimentación.

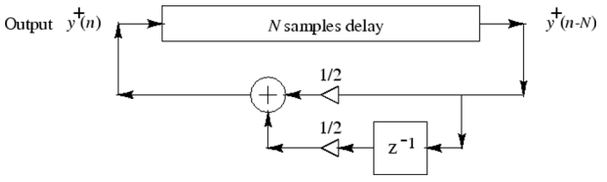


Figura 14: diagrama de flujo del algoritmo de Karplus-Strong. (Imagen obtenida en: https://ccrma.stanford.edu/~jos/SimpleStrings/Karplus_Strong_Algorithm.html)

El filtro pasa bajos cumple una doble función. Por un lado, atenúa las frecuencias más altas del sonido, cada vez que la señal vuelve a dar una vuelta. Esto es lo que en la realidad sucede en los instrumentos de cuerda pulsados, que durante el ataque contienen un sonido con una banda de ruido en el ataque, que luego va perdiendo parciales a medida que transcurre el tiempo. Por otro lado, el filtro afecta levemente la línea de retardo y, como consecuencia la percepción de la altura.

La ventaja de este modelo deviene tanto de su simpleza como de su efectividad, ya que es un algoritmo sumamente eficiente en ahorro de recursos computacionales.

Se genera una onda muy corta, con una duración de N muestras. Si bien, el algoritmo original indica utilizar un generador de ruido blando, suele utilizarse cualquier señal con un espectro rico en parciales. En el caso implementado, la fuente de sonido está conformada por un generador de ruido blanco rand~ multiplicado por una sinusoide, lo que genera una banda de ruido limitada, centrada sobre la frecuencia de la sinusoide.

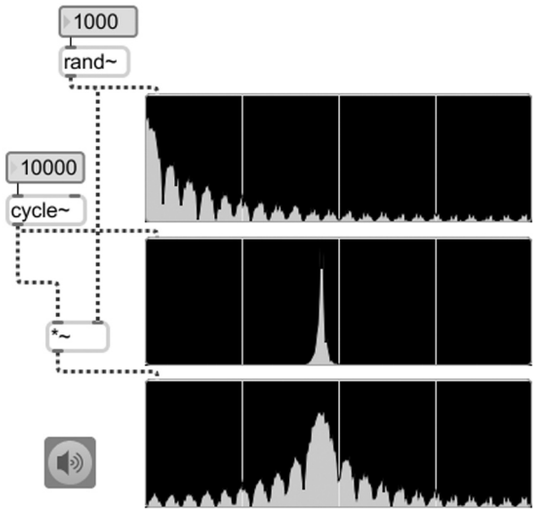


Figura 15: banda de ruido limitada y centrada sobre una frecuencia

La onda de excitación es enviada a una línea de retardo, controlada por el par de objetos tapin~ y tapout~, con un tiempo de delay igual a N.

La salida de la línea de retardo alimenta luego a un filtro pasa bajos de primer orden, con una ganancia menor a 1. El filtro está construido de una forma muy simple, con la suma de la señal que sale de la línea de retardo más una copia de la misma señal, pero retrasada una muestra.

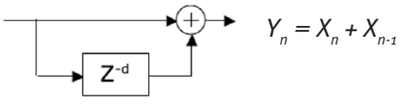


Figura 16: diagrama y ecuación en diferencias del filtro FIR de primer orden

Finalmente la salida del filtro realimenta la línea de retardo, además de dirigirse a la salida.

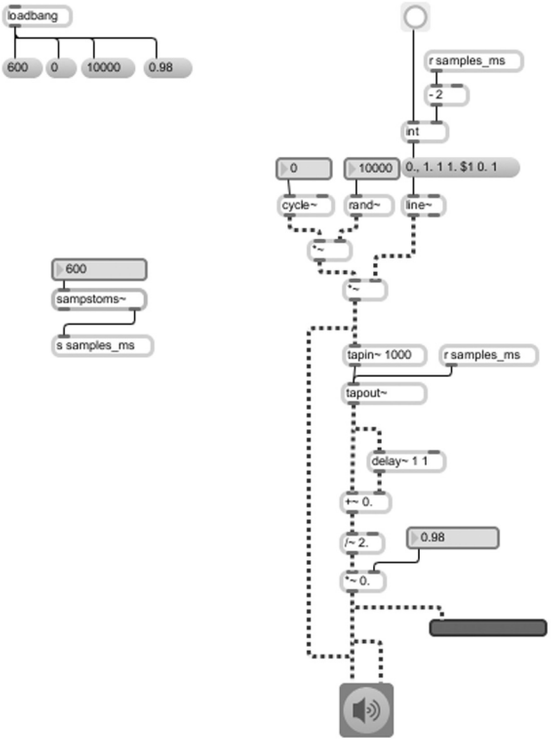


Figura 17: algoritmo de Karplus-Stron. Programación de Pablo Cetta

5. Síntesis granular

La construcción de un evento sonoro complejo a partir de pequeños fragmentos de sonido (granos), de entre 1 y 100 milisegundos, distribuidos en el tiempo se conoce como síntesis granular. La fuente original puede ser un sonido sintético (una sinusoide, un sonido de FM, etc.), o leído de un *sampler* (un archivo de sonido leído de un *buffer*). Básicamente esta técnica consiste en aplicar pequeñas envolventes dinámicas, para crear los granos, y disponerlos en el tiempo de maneras diferentes, de tal

manera que se pueda controlar de manera independiente la altura y la duración del evento sonoro, o bien crear nubes de sonidos, con una distribución aleatoria de los granos.

Esta técnica tiene su origen en la concepción del sonido como un conjunto de partículas dispuestas en el tiempo, conceptos que provienen del paradigma de la física cuántica. La idea de "grano" o "quantum" relacionada con el sonido fue propuesta por primera vez por el físico inglés Dennis Gabbor en 1947, pero la primera implementación digital de la síntesis granular fue hecha por Curtis Roads en 1974, en la Universidad de California, y luego desarrollada en 1981 en el MIT.

El compositor griego Iannis Xenakis fue uno de los primeros en introducir el concepto de los granos de sonido en la música, y la utilizó por primera vez en su obra *Analogique A-B*, para orquesta de cuerdas y cinta.

Entre los parámetros que se pueden controlar en un instrumento de síntesis granular aparecen:

- El tamaño del grano
- La forma de la envolvente (gaussiana, quasi gaussiana, trapezoidal, etc.)
- La forma de onda del sonido dentro del grano
- La frecuencia de onda dentro del grano
- Dispersión en el tiempo de los granos

Existen varios modelos de implementación de esta técnica, diferenciados esencialmente por la organización que hacen de los granos.

5.1. Síntesis granular sincrónica por altura

Esta técnica se constituye desde el análisis y resíntesis de un sonido. Del análisis de altura se deduce la frecuencia fundamenta del sonido, y de allí su período, que será tomado como unidad o grano. A estas unidades se les realiza un análisis espectral, del que se extrae la respuesta a impulso de dicho espectro. Estas respuestas a impulso controlan los parámetros de un banco de filtros en la resíntesis. La señal de salida resulta de un tren de pulsos con un período igual al del análisis, que excitan el banco de filtros de respuesta a impulso finita (FIR), que actúan como resonadores.

5.2. Síntesis granular cuasisincrónica

Se trata de una o más cadenas de granos puestos en sucesión con diferencias de tiempo variables entre cada uno de los granos. Una envolvente dinámica controla la forma de los granos, que se suceden uno a otro con intervalos de tiempo levemente diferentes.

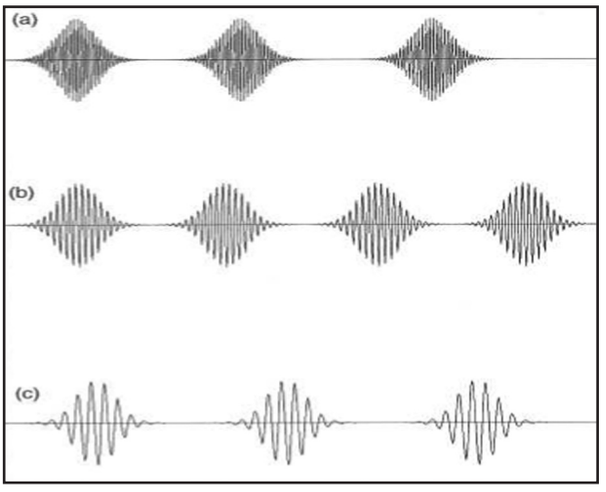


Figura 18: esquema de tres hileras de grains en síntesis granular cuasi asincrónica. La posición en el eje vertical indica la frecuencia del grano. El tiempo entre el inicio de los granos es aleatorio (imagen tomada de Computer music tutorial. Curtis Roads).

5.3. Síntesis granular asincrónica

En la síntesis granular asincrónica los granos son distribuidos de manera estadística en los ejes de la frecuencia y el tiempo. De esta forma pueden crearse "nubes" de granos, en las cuales pueden definirse sus rasgos generales: el comienzo de la nube, la duración de los granos, la densidad (cantidad de granos por segundo), la distribución general de los granos en el registro, la envolvente de amplitud de la nube, la forma de onda dentro de los granos y la distribución espacial de los granos.

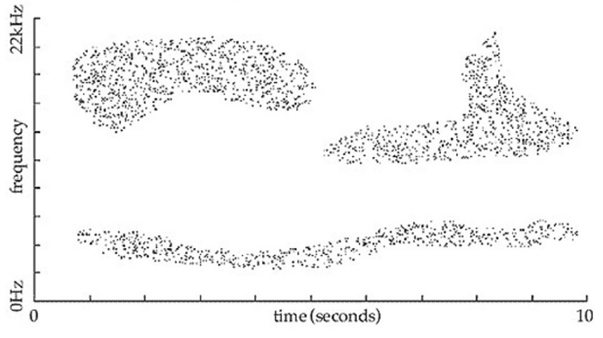


Figura 19: partitura donde se definen "nubes" de granos en función del tiempo y la frecuencia

5.4. Granulación de archivos de sonido grabados

En este caso la envolvente dinámica se aplica a pequeños fragmentos de un archivo de sonido grabado, leídos de un buffer. El control de la lectura de las muestras dentro del *buffer* hace posible distribuciones aleatorias de los fragmentos, reordenamientos en el tiempo, o la mezcla de fragmentos de diferentes archivos, si se utiliza más de un *buffer*.

La flexibilidad de este método hace posible manipular de forma independiente la frecuencia de lectura (*pitch shift*) y la duración del archivo de sonido (*time-stretch*). El ejemplo que sigue es un ejemplo de esta técnica de síntesis granular.

5.5. Un instrumento simple para generación de granos de sonido

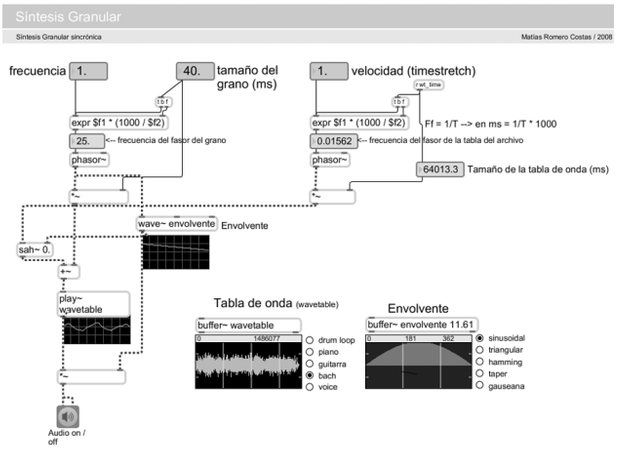


Figura 20

El oscilador objeto play~ *wavetable* es el objeto vinculado a la tabla de onda, del objeto buffer~ con el mismo nombre, que contiene el archivo de audio a ser procesado. El objeto wave~, por su parte, es el oscilador que lee el buffer con la envolvente dinámica que define cada grano.

Este instrumento tiene la capacidad de procesar de manera independiente la frecuencia y la velocidad del archivo de sonido, lo que se conoce respectivamente como *pitch shift* y *time stretch*.

Esto es posible porque existen dos lectores de posición que trabajan en paralelo, mientras que el fasor de la izquierda es el encargado de leer internamente cada uno de los granos, el de la derecha es el que indica la posición general de lectura de la tabla de onda completa. Una vez que el primer fasor finaliza de recorrer un grano, salta a una nueva posición dentro de la tabla. El objeto sah~ (*sample and hold*) almacena los valores de posición que recibe por su primera entrada, pero solo devuelve un nuevo valor cuando la amplitud de la envolvente que ingresa por su segunda entrada pasa por, o, repite su estado anterior.

6. Síntesis cruzada

La síntesis cruzada consiste en multiplicar los valores de amplitud de los componentes del espectro de una señal por los de otra diferentes. Para decirlo de otra manera, se aplica la envolvente espectral de un sonido al espectro de otro. A diferencia de las técnicas anteriores, donde las operaciones se realizan en el dominio del tiempo, es decir su forma de onda, en la síntesis cruzada, se realiza en el dominio del tiempo, su espectro.

Por convención en las ecuaciones, se utilizan las letras minúsculas para identificar la forma de onda y la mayúscula para su espectro. Por ejemplo:

$$DFTx(n) = X(k)$$

Quiere decir que la transformada discreta de Fourier de una onda x(n) nos devuelve su espectro X(k).

Sabemos que si multiplicamos dos formas de onda, el espectro resultante no es equivalente a la multiplicación de ambos espectros, sino a la convolución de los mismos. De la misma manera, la multiplicación de dos espectros es equivalente a la convolución de sus formas de onda (la operación de convolución es conmutativa, al igual que la multiplicación).

$$h(n).x(n) = H(k)*X(k)$$
y
$$H(k).X(k) = DFT(h(n)*x(n))$$

La convolución lineal o aperiódica se define por la siguiente relación:

$$y(n) = \sum_{m=0}^{N_h-1} h(m)x(n-m)$$

La convolución se utiliza, muy comúnmente, en la creación de reverberación natural, donde se mezcla un sonido "seco", de un instrumento por ejemplo, con la respuesta a impulso grabada en un recinto real en particular. Luego de convolucionar ambas señales, el sonido original adquiere las características acústicas de reverberación de dicho lugar.

La forma más común de implementar la convolución en la síntesis cruzada es a través de la FFT, lo que se conoce como convolución rápida. Para ello se siguen una serie de pasos desde los sonidos originales a ser "cruzados" y el resultado final de la síntesis.

1. Se realiza la FFT de cada uno de los dos sonidos para pasar de la forma de onda a espectro de la señal.
2. Se multiplican los valores de amplitud de ambos espectros.
3. Del nuevo espectro obtenido de la operación anterior, se realiza la IFFT (transformada inversa) para obtener la forma de onda del sonido resultante.

6.1. Aplicación en Max MSP

A la izquierda observamos al implementación de la síntesis cruzada de dos archivos de audio leídos y ejecutados por los objetos sfplay~. Ambas señales son enviadas al objeto pfft~, encargado de la convolución. Los argumentos del objeto se corresponden, respectivamente, con

el tamaño de la ventana de análisis (FFT size), especificado en muestras, y el factor de solapamiento de la ventana de análisis y resíntesis.

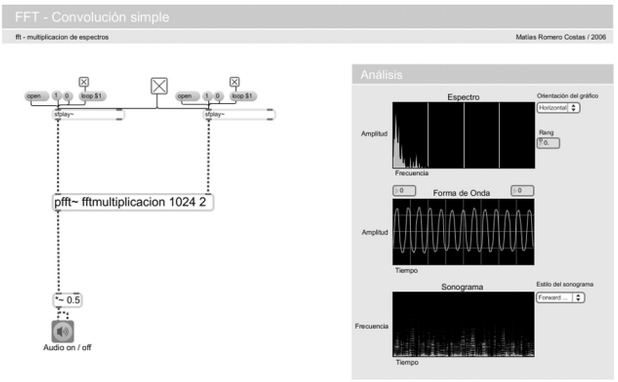


Figura 21

La siguiente figura corresponde a la programación que se encuentra dentro del subpatch pfft~ fftmultiplicacion. Los objetos fftin~ realizan la FFT y devuelven los valores reales (a) e imaginarios (b). Para obtener los valores de amplitud se utiliza el objeto cartopol~ (que convierte valores expresados en coordenadas cartesianas a polares), a partir de la siguiente ecuación:

$$A = \sqrt{a^2 + b^2}$$

Los valores de amplitud obtenidos son multiplicados por los valores reales e imaginarios de la primera señal, y, finalmente transformados nuevamente al dominio del tiempo a través de la IFFT (transformada inversa).

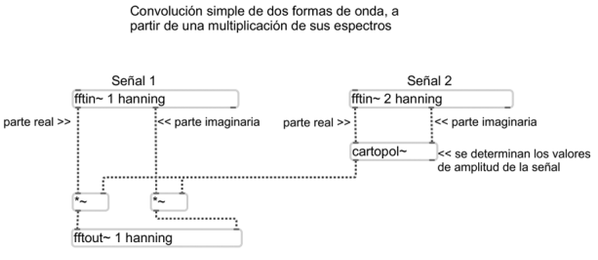


Figura 22: subpatch pfft~ fftmultiplicacion

7. Operaciones con líneas de retardo

El valor de salida y(n) es igual al valor de la entrada x(n) retrasado por d muestras. Como el muestreo comienza en el tiempo t=0, es decir con la muestra n=0, todas las muestras anteriores a t=0 no están definidas. Entonces se toma, por lo general, a toda muestra anterior con un valor igual a 0 (x-1=0).

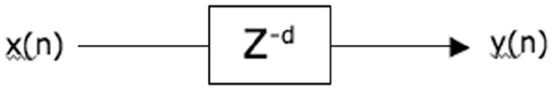


Figura 23. línea de retardo (*delay network*)

7.1. Delay simple

Un delay simplemente toma una señal de audio de entrada y la retrasa un tiempo determinado (en muestras o milisegundos). Una línea de delay digital se implementa asignando un buffer de valores en memoria, que van a ser leídos un tiempo más tarde. El delay se utiliza para crear ecos, reflexiones en los reverberadores, propagación de ondas, efectos de chorus, flanging, etc.

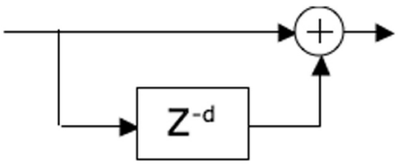


Figura 24: delay simple

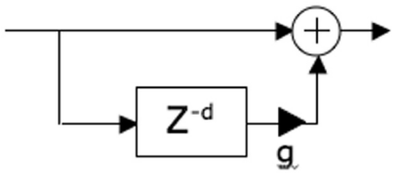


Figura 25: delay con control de ganancia

En Max, los objetos encargados de la línea de retardo son tapin~ y tapout~. Se utilizan siempre juntos, y el primero es el que define el tamaño máximo del buffer y el segundo el tiempo de delay, en milisegundos.

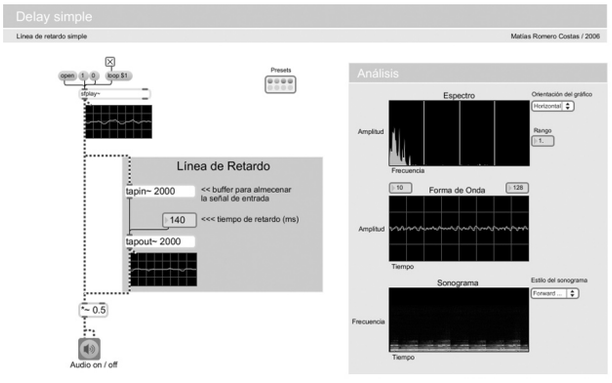


Figura 26

7.2. Delay con realimentación

La realimentación se establece cuando la salida de una línea de retardo vuelve a alimentar su propia entrada. El control de ganancia posterior a la salida del retardo controla el nivel de la realimentación. Con un valor menor a 1 cada nueva entrada en la línea de delay tendrá un valor de amplitud menor (con g>1 el sistema se vuelve inestable). En la figura 27 la señal original es sumada a la salida de la línea de retardo; ésta a su vez es nuevamente ingresada en su entrada; la entrada estará formada ahora por la suma de la entrada original y la

salida del retardo, multiplicado por un valor g antes de reingresar a la cadena. Una señal con un valor original de amplitud igual a 1, luego de ser retrasada y multiplicada por g tendrá una amplitud igual a g1; después de ser multiplicada la segunda vez su amplitud, antes de realimentar el delay, será igual a g2; la tercera vez igual a g3; y así sucesivamente. Quiere decir que la señal será multiplicada por gn, donde n es igual a la cantidad de veces que fue retrasada y multiplicada.

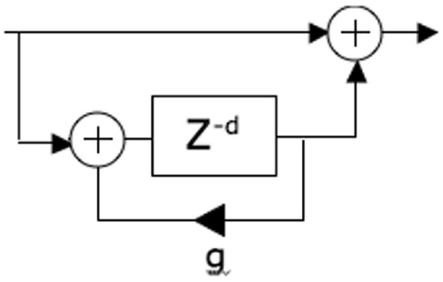


Figura 27. Delay con realimentación

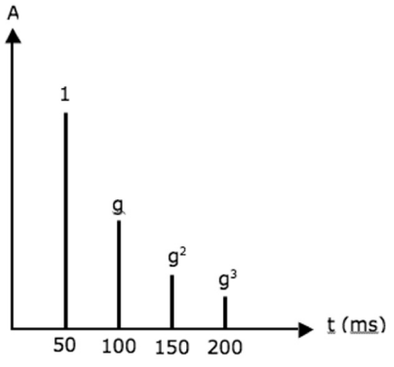


Figura 28. decaimiento de la amplitud de cada una de las repeticiones. El tiempo de retardo es igual a 50 ms.

7.3. Tap-Delay

Se trata de una línea de retardo que permite el acceso en cualquier lugar intermedio de la línea de delay. En el delay anterior la salida es tomada luego de finalizado el retardo total, sin embargo, es posible también acceder a porciones menores del tiempo de delay. A la acción de tomar la salida desde puntos dentro de la línea de delay se las denomina "*tapping*" (*tap* significa llave). En su implementación en software el nombre del delay está relacionado con la cantidad de puntos intermedios (taps) que posea. Así por ejemplo un denominado 3-tap-delay tiene el siguiente esquema:

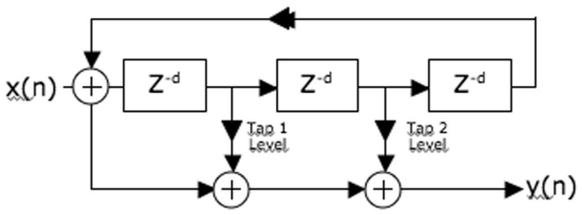


Figura 29: 3 tap-delay

El objeto tapout~ permite implementar múltiples retardos, tantos como argumentos se definan. En este caso un 10 multitap-delay con realimentación de todas las salidas. Cada una de las salidas puede tener un tiempo de retarde diferente, así como su nivel de realimentación.

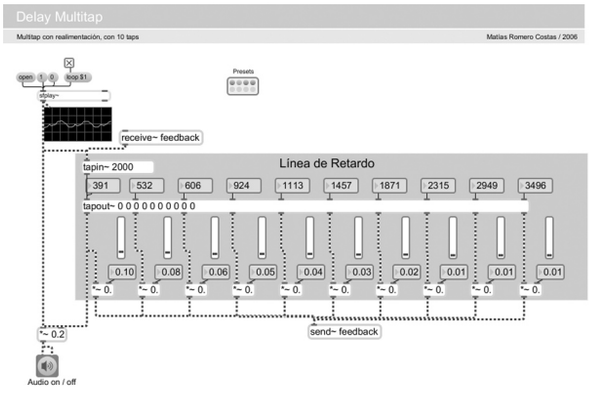


Figura 30

7.4. Filtro comb

Un caso simple de línea de retardo con realimentación son los filtros comb, que toman su nombre por la forma de su respuesta en frecuencia que asemeja a los dientes de un peine (*comb* significa peine).

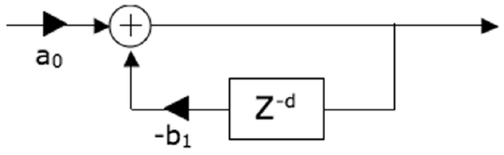


Figura 31: filtro comb con realimentación

Su ecuación en diferencias queda definida por:

$$y(n) = a_0 x(n) - b_1 y(n-d)$$

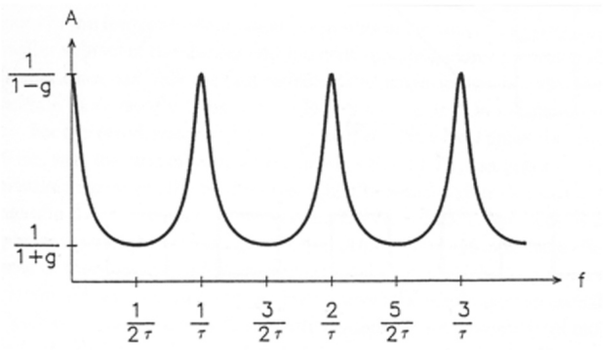


Figura 32: filtro comb con realimentación

Los picos de amplitud de la respuesta en frecuencia están ubicados a una distancia igual a la "frecuencia natural del filtro" que es a la inversa del tiempo de delay:

$$f_0 = 1/\tau$$

La profundidad del mínimo y la altura del máximo dependen de la elección de g, en donde valores más cercanos a 1 significan mayor diferencia entre los extremos. Esto genera que el sonido que entra haga resonar al filtro a su frecuencia natural f_0 , adicionando otro sonido con esa fundamental al original.

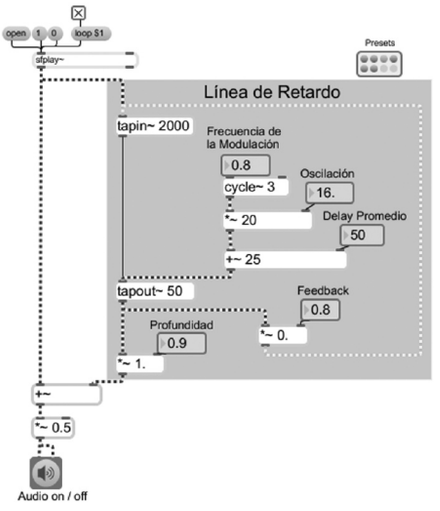


Figura 33: programación de un filtro comb

Si observamos el diagrama, y el algoritmo de Max, podemos notar que el filtro comb es en realidad un delay con realimentación, pero con un valor de retardo muy pequeño, menor a 10 ms. La superposición de la señal junto a copias de sí misma a diferencias temporales tan pequeñas provoca una deformación de la forma de la onda que tiene consecuencias perceptibles al nivel espectral.

7.5. Flanger

El flanger se implementa sumando una señal a una versión retrasada de sí misma con una duración de delay corta y variable en el tiempo. La respuesta en frecuencia, en un instante, es similar a la de un filtro comb, pero donde los picos y ranuras se comprimen y expanden con el tiempo. El espectro de un sonido que pasa por un flanger es enfatizado y atenuado por regiones de frecuencia de forma variable en el tiempo. La duración del retardo es generalmente modulada por un oscilador de baja frecuencia (LFO) y las formas de ondas típicas son la sinusoidal, triangular o exponencial. La profundidad controla entre que valores extenderá el tiempo de retardo.

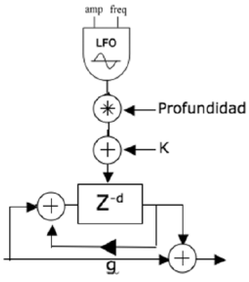


Figura 34

Al escuchar el resultado tan característico de este efecto, y más aun, observando el espectro, podemos entender al flanger como un filtro comb en el cual la frecuencia natural del filtro varía en el tiempo de forma sinusoidal.

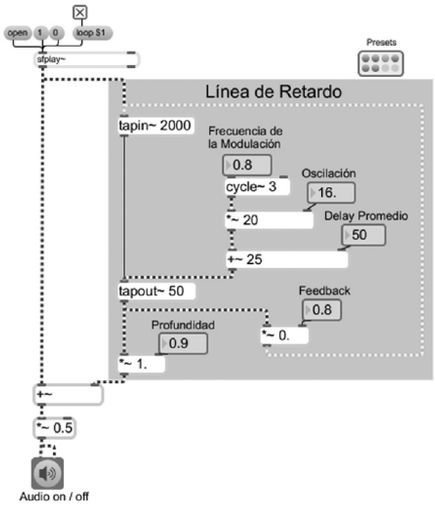


Figura 35: flanguer

8. Campanas moduladas

Campanas moduladas es un trabajo realizado en 2009 por los docentes (Pablo Cetta y Matías Romero Costas) y alumnos (Ariel Bottigheimer, Cecilia Chantril, Agustín Genoud, Romina Pow y Leonardo Saltamerenda) de la cátedra de Laboratorio de Sonido III, de la carrera de Artes Multimediales, dentro del Área Transdepartamental de Artes Multimediales del IUNA.

Se trata de una instalación sonora interactiva donde el público se encuentra con unos instrumentos musicales colgantes. Un grupo de pequeñas campanas tubulares suspendidas en el espacio, que pueden ser tocados por los visitantes. Estos sonidos, que en un principio son los propios del objeto, evolucionan lenta y sutilmente en un complejo sonoro musical.

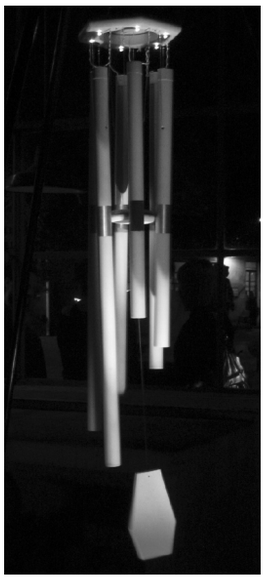


Figura 36: campanas moduladas. Exposición “La noche de los museos”. 2009

Cada uno de los seis tubos de la campana tiene colocado un sensor de contacto, conectado a un circuito electrónico, y a una placa arduino. Este sistema de censado está conectado a una computadora, y administrado y controlado desde Max-MSP.

La parte sonora del instrumento, también programada en Max-MSP, está conformada por una serie de módulos de generación y procesamiento de sonido, con 36 configuraciones diferentes, que se suceden automáticamente y de forma aleatoria. Las unidades de procesamiento y generación de sonido utilizados son:

- Players de sonido.
- Multitap delay.
- Flanger.
- FM.
- Modulador en anillo.
- Armonizador.
- Banco de filtros comb.

En la figura siguiente se muestra la pantalla principal del programa. En la parte superior se puede monitorear el estado de los valores de los parámetros de cada módulo de sonido. Abajo y a la izquierda se encuentra el control de nivel general, y los niveles relativos del sonido directo del instrumento y de los procesos. A su derecha la matriz de interconexionado, la matriz de presets de configuraciones, el acceso a los procesadores, y la monitorización de los seis sensores de los tubos de la campana.

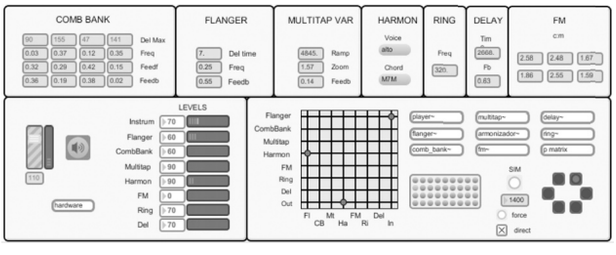


Figura 37: pantalla principal del programa

8.1. Players de sonido

Cada uno de los sensores dispara una copia grabada del mismo sonido del tubo de la campana. Como vemos en el *patch* hay seis objetos sfplay~ con los sonidos de cada uno de los seis tubos. Cada objeto recibe la orden de ejecutar su sonido cuando el sensor correspondiente es activado.

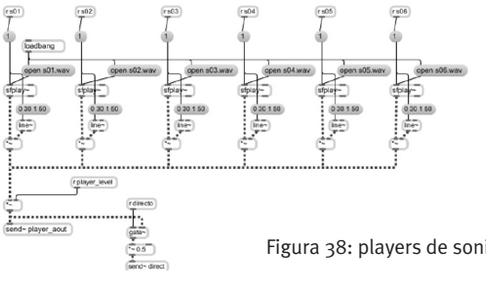


Figura 38: players de sonido

8.2. Multitap con delay variable

Si modificamos el tiempo de delay durante la ejecución, lo que se produce es alteración de la frecuencia. De hecho, existen dos métodos para implementar los transpositores de altura (*pitch shift*): a través de los delays variables, o con un phase vocoder.

El factor de transposición $t[n]$ de la salida de una línea de retardo puede calcularse a partir de:

$$t[n] = y[n] - y[n\text{-}1] = 1 - (d[n] - d[n\text{-}1])$$

Donde y[n] d[n] es el tiempo de delay (variable). Si d[n] no varía con el tiempo, el factor de transposición t[n] es igual a 1, es decir, no hay cambio de la altura.

Este procesador contiene 10 líneas de retardo variable, con realimentación. Los tiempos de retardo de cada línea son generados aleatoriamente, y pasando de forma gradual del valor actual al próximo con una rampa lineal. Un factor de multiplicación mt_zoom actúa a la manera de un "zoom" que permite expandir considerablemente los tiempos de retardo.

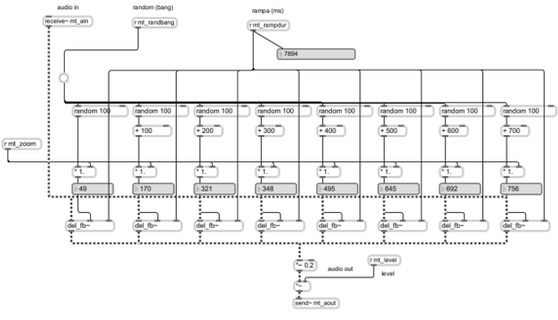


Figura 39

8.3. Flanger

Los valores de delay, frecuencia del oscilador, y nivel de *feedback* del flanger son elegidos aleatóriamente

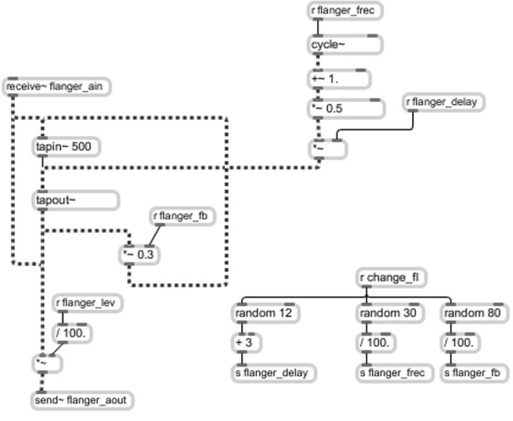


Figura 40

8.4. FM

Aquí la síntesis por FM se utiliza para transformar de manera gradual el sonido original de la campana. Los parámetros iniciales otorgan las características tímbricas que asemejan el sonido sintético al propio del instrumento, pero luego van variando lentamente a espectros disonantes, más cercanos al ruido

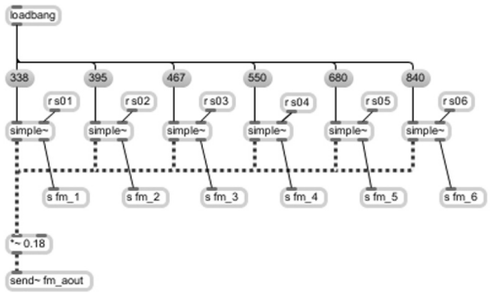


Figura 41

8.5. Modulador en anillo

El parámetro que regula las características tímbricas del sonido a ser procesado es la velocidad de la modulación, es decir la frecuencia a la que oscila la moduladora (objeto cycle~). Los valores posibles van desde o, sin modulación, hasta 1999 milisegundos. Esto quiere decir que el mismo procesador puede convertirse en una envolvente, un trémolo, o una modulación en anillo, dependiendo de la velocidad a la que se module a la portadora.

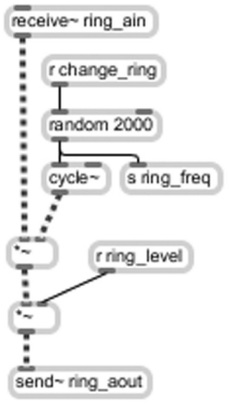


Figura 42

8.6. Armonizador

El armonizador utilizado en este *patch* es un procesador en el cual cada sonido producido por la campana es armonizado en un acorde de cuatro voces, por lo tanto, posee tres transpositores (figura 43), cada una de estas voces es asignada a una posición en particular del registro y de forma aleatoria, en este caso en particular (bajo, tenor, alto y soprano, (Figura 45). Las posibilidades de armonización de cada una de las voces son de cinco acordes también aleatoriamente (Figura 46).

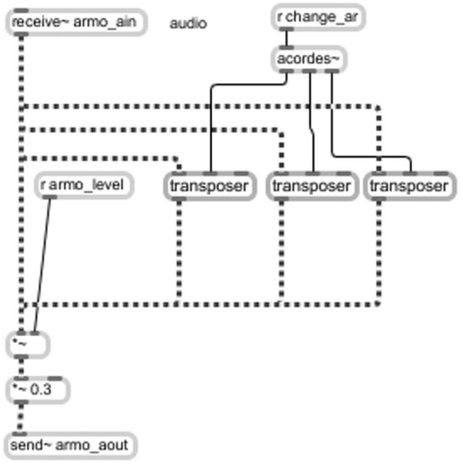


Figura 43: subpatch armonizador

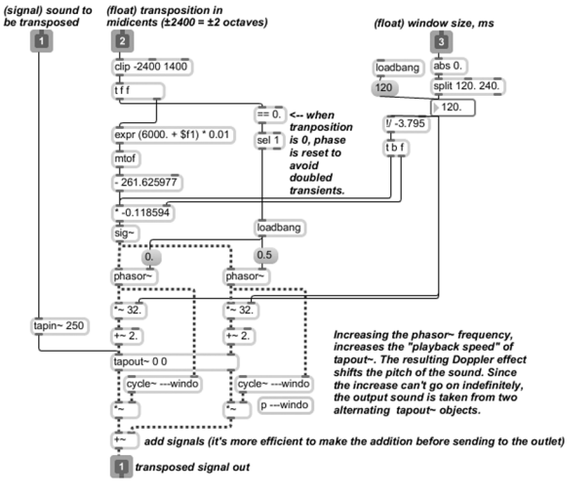


Figura 44: objeto transposer

El objeto transposer es una abstracción que viene con el programa Max-MSP, que permite un rango de transposición de dos octavas hacia arriba y hacia abajo.

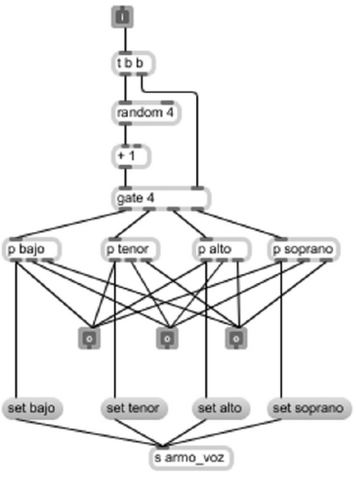


Figura 45: objeto acordes~

El objeto acordes asigna de forma aleatoria el sonido entrante hacia alguna de las cuatro voces (bajo, tenor, alto o soprano).

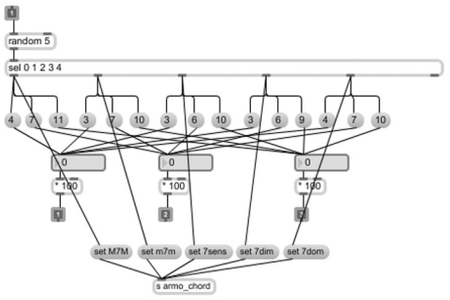


Figura 46: subpatch p bajo

Los subpatch p bajo, p tenor, etc. son los encargados de armonizar los sonidos en cinco acordes diferentes, en este caso en particular pueden ser un acorde mayor con séptima mayor, acorde menor con séptima menor, de séptima de sensible, de séptima dim., y de séptima de dominante. A saber las notas de los acordes están dadas en semitonos y la selección de los acordes es de manera aleatoria.

8.7. Banco de filtros comb

Este procesador está conformado por cuatro filtros comb en paralelo, una característica en particular es que el tiempo de retardo varía en función de un oscilador. Además, podemos controlar el delay máximo, la ganancia del filtro, así como también el nivel del *feedfoward* y del *feedback*, y el nivel general de amplitud.

La ecuación en diferencias implementada en este filtro es:

$$y[n] = ax[n] + bx[n - delay] + cy[n - delay]$$

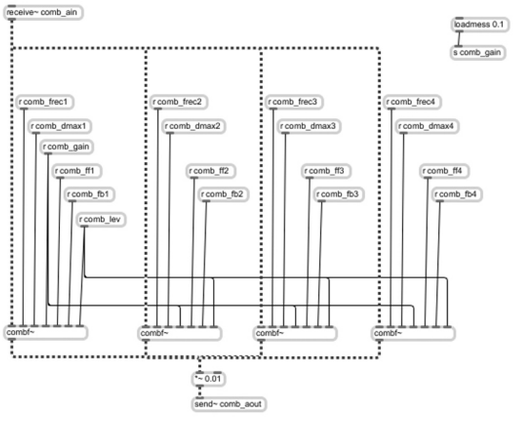


Figura 47: subpatch comb_bank–

8.8. Encadenamiento de las unidades de procesamiento

El programa permite encadenar los módulos de procesamiento entre sí, tanto en serie como en paralelo, mediante una matriz de 8 x 8. Las filas corresponden a las entradas y las columnas a las salidas. El objeto encargado de esto es matrix~, que cumple la función de una mezcladora de sonido.

En el gráfico podemos observar en la parte superior las entradas, y en la inferior las salidas. El parámetro ramp define un tiempo de crossfade entre las señales (en este caso 2.000 ms), para evitar los clics en el pasaje.

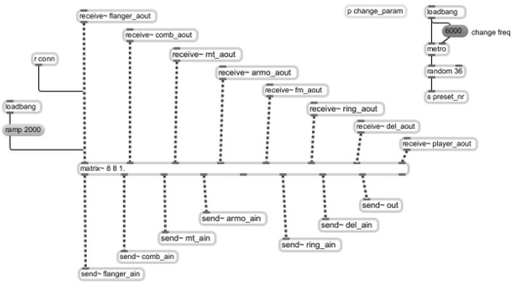


Figura 48

El objeto matrix~ puede controlarse con un objeto de interfaz gráfica (matrixcontrol) que hace más sencillo el manejo de los cambios, al mismo tiempo que facilita la visualización de los estados.

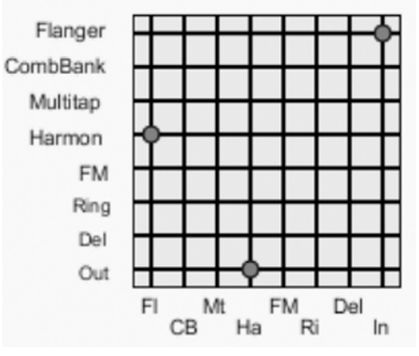


Figura 49

8.9. Automatización de los programas

Es importante remarcar, que al tratarse de una instalación, el sistema funciona de manera totalmente autónoma, sin control alguno por parte de operadores. Una vez puesta en funcionamiento la obra puede permanecer activa un tiempo indefinido, con solo la intervención del público.

Un metrónomo, que se activa al encender la instalación, cambia de programa cada 6 segundos. Cada cambio incluye de que manera se interconectan los procesadores, y los parámetros de configuración de cada uno de ellos.

En la Figura 50 que sigue observamos, como ejemplo, el programa N° 10. Allí el sonido directo (in) pasa a través del flanger (FL) hacia el armonizador (Ha) y, de allí a la salida de audio.

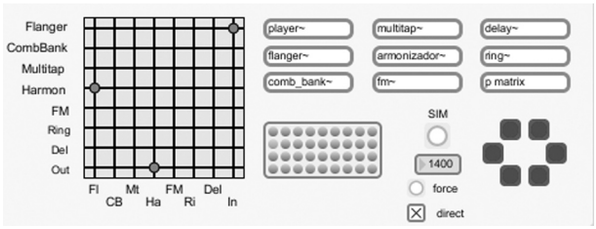


Figura 50: preset de configuración Nº 10

Referencias bibliográficas

Chowning, John M. "The Synthesis of complex audio spectra by means of frecuency modulation", en *Journal of the Audio Engineering Society* (California: Stanford, 1973).

Curtis Roads. *The computer music tutorial* (Cambridge, Massachusetts: The MIT Press, 1996).

Di Liscia, Pablo. *Generación y procesamiento de sonido y música a través del programa Csound* (Bernal, Buenos Aires: Editorial de la Universidad Nacional de Quilmes, 2004).

Dodge, Charles y Jerse, Thomas A. *Computer Music* (Estados Unidos: Library of Congress, 1997).

Moore, F. Richard. *Elements of computer music* (Ney Jersey: PTR Prentice Hall Inc., 1990).

Puckette, Miller. *Theory and Techniques of Electronic Music* (San Diego: University of California, 2005).

Smith, Julius O. III. "Physical Modelling Synthesis Update", en *The Computer Music Journal* (1996: Vol. 20, Nº 2, pp. 44-56).

Notas

1. Diagrama de flujo: representación gráfica o esquema simplificado para representar un sistema, sus componentes, la interconexión y la relación entre las unidades generadoras.

2. LFO: Low Frecuency Oscilator (Oscilador de baja frecuencia). Es un oscilador que trabaja a frecuencias por debajo del rango audible (20Hz), y que se utiliza como una unidad de control de parámetros en la síntesis de sonido.

3. En las ecuaciones, la operación de convolución se representa con el signo *

(*) www.arduino.cc

TARCISIO LUCAS PIROTTA

Es diseñador en Comunicación Visual y profesor en Producción Multimedial, diplomado en la Facultad de Bellas Artes, Universidad Nacional de La Plata.

Se desempeña como docente e investigador en la Facultad de Bellas Artes (UNLP) y en el Área Transdepartamental de Artes Multimediales (IUNA). Ha dictado cursos de posgrado y talleres relacionados con arte multimedia y nuevas tecnologías.

Como artista multimedia integra el Grupo Proyecto Biopus, desarrollando desde el año 2003 instalaciones interactivas que conjugan arte, ciencia y tecnología.

Técnicas de programación vinculadas a la realidad aumentada y a las interfaces tangibles

Tarcisio Lucas Pirotta

Palabras claves

Programación, realidad aumentada, interfaces tangibles, ArtoolKit, reactivision

1. Resumen

El presente trabajo tiene por objetivo analizar las principales herramientas de programación que permiten específicamente el reconocimiento de patrones para la creación de aplicaciones de realidad aumentada y la implementación de interfaces tangibles.

2. Generalidades

Para la elaboración de este artículo se ha tomado como objeto de análisis dos herramientas de programación, ArtoolKit y reacTiVision, vinculadas al desarrollo de aplicaciones de realidad aumentada y de interfaces tangibles respectivamente.

La primera consideración a tener en cuenta en la elección de estas herramientas y sus correspondientes técnicas de programación para el desarrollo de este escrito, es que han demostrado óptimos resultados en aplicaciones desarrolladas por integrantes de este proyecto de investigación, como por ejemplo la obra interactiva "*Mundo Circular*", y son hasta el momento unas de las mejores herramientas que se han implementado en diferentes proyectos a escala internacional. Por esto mismo es numerosa e interesante la cantidad de publicaciones y documentación que existe acerca de estas herramientas, lo que contribuye a realizar un correcto análisis y relevamiento de las técnicas de programación que emplean.

Otro aspecto importante es la especificidad de cada una de las herramientas, reactivision fue diseñada especialmente para el desarrollo de interfaces tangibles, más específicamente de mesas reactivas, mientras que Artoolkit permite desarrollar todo tipo de aplicaciones de realidad aumentada, excediendo aun aquellas donde se implementan interfaces tangibles.

Relacionado con esto último es interesante establecer brevemente cuál es la relación entre la realidad aumentada y las interfaces tangibles, y cuáles son las diferencias principales en las herramientas que permiten la creación de cada una de ellas.

Podemos considerar que el desarrollo de interfaces tangibles se encuentra siempre dentro del ámbito de las aplicaciones de realidad aumentada, son una variante que podríamos definir como un subconjunto del conjunto de realidad aumentada, lo cual nos permite deducir que todas las aplicaciones que involucran interfaces tangibles son consideradas realidad aumentada, aunque no sucede lo mismo de manera inversa, no todas las aplicaciones de realidad aumentada implican necesariamente la utilización de interfaces tangibles, sino que es un campo de acción mayor y que tiene mas posibilidades de alcance.

Por esto mismo si bien con la herramienta ARToolKit podemos desarrollar aplicaciones de realidad aumentada, lo que incluye el desarrollo de interfaces tangibles, con la herramienta reacTiVision solo podemos desarrollar interfaces tangibles.

La principal diferencia es el modo que cada una tiene para captar los patrones bitonales y detectar la posición del punto de vista del usuario, y de ahí las limitaciones y diferencias de cada una.

En las aplicaciones que emplean interfaces tangibles, como en el caso de las mesas reactivas, los patrones se encuentran siempre a la misma distancia de la cámara y dispuestas de manera perpendicular, por lo tanto, reacTiVision solo esta diseñada para captar patrones bitonales en 2D, o sea, patrones que solo se trasladan en relación con la cámara en los ejes X Y. Mientras que en las aplicaciones de realidad aumentada los patrones pueden estar dispuestos en cualquier lugar y posición, y pueden ser trasladados y rotados en cualquiera de sus ejes, por lo que las aplicaciones, como es el caso de ArtoolKit, deben ser capaces de captar los patrones con perspectivas en 3D.

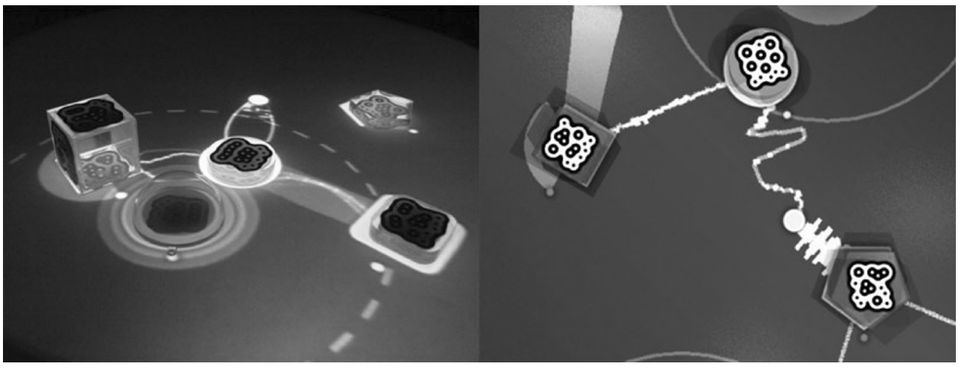


Figura 1: imágenes de la instalación reacTables desarrollada con reacTIVision (http://www.iua.upf.es/mtg/reactable)

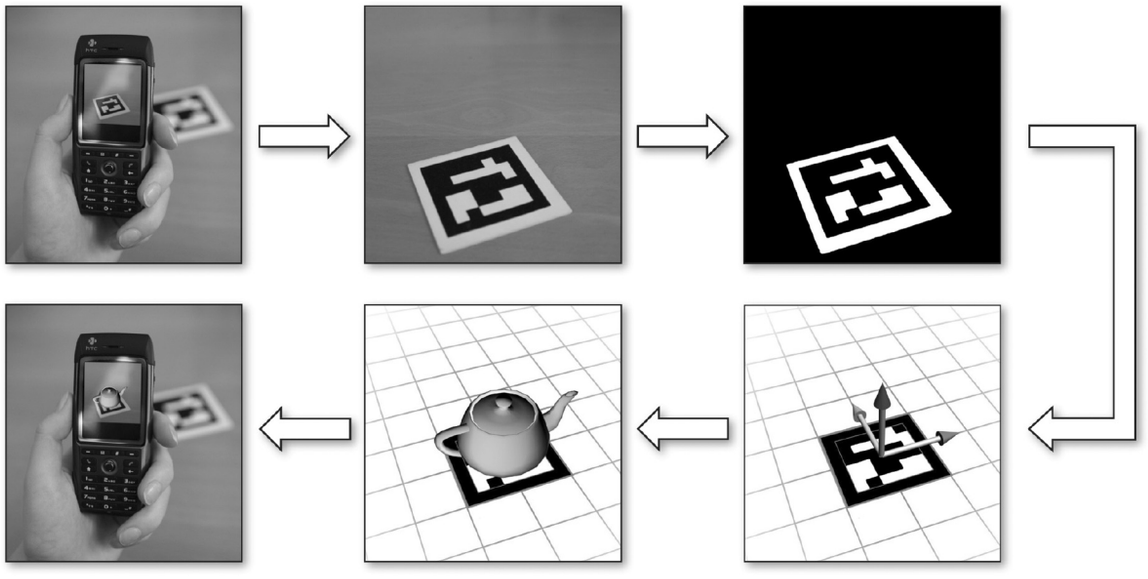


Figura 2: ARToolKit. imagen obtenida de http://www.icg.tu-raz.ac.at/Members/daniel/ARToolKitPlusMobilePoseTracking/download

3. ARToolKit

http://www.hitl.washington.edu/artoolkit/ ARToolKit es un conjunto de librerías para C/C++ para la construcción de aplicaciones de realidad aumentada que implican la superposición de imágenes virtuales con las del mundo real.

Como mencionamos anteriormente una de las dificultades en el desarrollo de este tipo de aplicaciones es el seguimiento del punto de vista del usuario. Para saber desde que perspectiva se construirá la imagen virtual, la aplicación necesita saber cómo el usuario está mirando en el mundo real.

ARToolKit utiliza algoritmos de visión artificial (o CV, Computer Vision) para resolver este problema, sus librerías de seguimiento de video, calculan en tiempo real la posición y orientación de la cámara en relación con las marcas ubicadas en diferentes objetos. Esto habilita el fácil desarrollo de un amplio rango de aplicaciones AR

3.1. Principales funciones

A continuación, analizaremos un programa básico implementado en ARToolkit que permite captar un patrón a través de una cámara y sobreimprimirle digitalmente un modelo 3D.

El esquema de funcionamiento lo vemos en el siguiente cuadro:

Etapa	Función
1. Inicio y config. de la aplicación	init
2. Captura de un cuadro de video	arVideoGetImage (llamada desde mainLoop)
3. Detección de Patrones	arDetectMarker (llamada desde mainLoop)
4. Cálculo de def. de cámara	arGetTransMat (llamada desde mainLoop)
5. Dibujo de los objetos virtuales	draw (llamada desde mainLoop)
6. Finaliza la captura de video	cleanup

Figura 3: esquema de funcionamiento de programa implementado en ArtoolKit

Las funciones más importantes son main, init, mainLoop, draw y cleanup, y serán explicadas en detalle a continuación.

3.1.1. Main

La función main es, como dice su nombre en inglés, la función principal del programa y la que llama y ejecuta a todas las demás funciones mencionadas en el párrafo anterior.

```
main(int argc, char *argv[])
{
init();
arVideoCapStart();
argMainLoop( NULL, keyEvent, mainLoop );
}
```

3.1.2. Init

La función init, declarada en la primera línea de la función main, contiene el código para comenzar la captura de video, leyendo los parámetros de la cámara, y definiendo la ventana de gráficos. Esto corresponde al paso 1 en el cuadro de la Figura 3. Luego, entramos en el estado de ejecución en tiempo real, con una llamada a la función arVideoCapStart, que es la que inicia específicamente la captura de video. A continuación, es llamada la funcion argMainLoop, la cual comienza el bucle con el programa principal, teniendo como argumentos a otra función llamada keyEvent, que detecta cualquier evento del teclado, y la función mainLoop que se asocia con el bucle principal de procesamiento de gráficos.

Como hemos mencionado, init es llamada desde la función main, y se utiliza para inicializar la captura de video y leer los parámetros iniciales de ArtoolKit.

En primer lugar, el dispositivo de video es abierto y se detecta el tamaño de la imagen, como vemos a continuación:

```
/* abre dispositivo de video */
if( arVideoOpen( vconf ) < 0 ) exit(0);

/* encuentra el tamaño de la imagen */
if( arVideoInqSize(&xsize, &ysize) < 0 ) exit(0);
printf("Image size (x,y) = (%d,%d)\n", xsize, ysize);
```

Luego es necesario inicializar los parámetros de ARToolKit. Los más importantes son:

- Los patrones a detectar que serán usados en la aplicación los objetos virtuales que corresponden a cada uno de esos patrones.

- Las características de la cámara utilizada.

Estos parámetros son leídos de archivos externos.

```
/* define parámetros iniciales */
   if( arParamLoad(cparaname, 1, &wparam) < 0 ) {
       printf("Error al detectar cámara !!\n");
       exit(0);
   }
```

Posteriormente se lee la definición del patrón en comparación con la información del patrón incluido en un archivo externo y se asocia el patrón con un identificador:

```
if( (patt_id=arLoadPatt(archivo)) < 0 ) {
    printf("error al leer patron !!\n");
    exit(0);
}
```

Por último, una ventana gráfica es abierta:

```
/* abre ventana gràfica */
argInit( &cparam, 1.0, 0, 0, 0, 0 );
```

3.1.3. MainLoop

La función mainLoop realiza la mayoría de las llamadas a funciones, abarcando desde el paso 2 al paso 5 del esquema del programa de la Figura 3.

Primero se captura un cuadro de video utilizando la función arVideoGetImage:

```
/* captura un cuadro de video */
if( (dataPtr = (ARUint8 *)arVideoGetImage()) == NULL ) {
    arUtilSleep(2);
    return;
}
```

La imagen de video es luego visualizada en la pantalla con la función argDispImage, que tiene como argumentos la imagen capturada previamente con arVideoGetImage y la posición donde será mostrada:

```
argDispImage( dataPtr, 0,0 );
```

Luego la función arDetectMarker es utilizada para buscar dentro de la imagen cuadrados que puedan incluir los correspondientes patrones cargados anteriormente en la inicialización:

```
if( arDetectMarker(dataPtr, thresh, &marker_info, &marker_num) < 0 ) {
    cleanup();
    exit(0);
});
```

El número de patrones encontrados es asignado a la variable marker_num, mientras que marker_info es un puntero a una lista de estructuras de patrones que

contienen la información de coordinación y reconocimiento, como así también los valores de id correspondientes a cada uno de los patrones.

En este punto del programa, la imagen ha sido mostrada y analizada, y a continuación todos los valores de los patrones detectados son comparados para asociar el patrón correcto y su correspondiente id:

```
/* chequea la visibilidad del patrón */
k = -1;

for( j = 0; j < marker_num; j++ ) {
    if( patt_id == marker_info[j].id ) {
        if( k == -1 ) k = j;
        else if( marker_info[k].cf < marker_
info[j].cf ) k = j;
    }
}

if( k == -1 ) {
    argSwapBuffers();
    return;
}
```

La deformación entre el patrón real impreso y la imagen que muestra la cámara puede ser detectada utilizando la función arGetTransMat:

```
/* determina la deformación entre el patrón real
y la imagen de la càmara */
arGetTransMat(&marker_info[k], patt_center,
patt_width, patt_trans);
```

La posición y orientación real de la cámara relativa al patron impreso es introducida en una matriz 3 x 4, patt_trans.

Finalmente, los objetos virtuales pueden ahora ser dibujados sobre la imagen del patrón empleando la función draw:

```
draw();
```

La función draw se divide en inicialización del render, configuración de la matriz y, por último, el render final del objeto.

Para inicializar el render es necesario definirlo en modo 3D y configurar unos mínimos estados de OpenGL:

```
argDrawMode3D();
argDraw3dCamera( 0, 0 );
glClearDepth( 1.0 );
glClear(GL_DEPTH_BUFFER_BIT);
glEnable(GL_DEPTH_TEST);
glDepthFunc(GL_LEQUAL);
```

Luego de esto se requiere convertir los cálculos de deformación (matriz de 3 x 4) a un formato OpenGL a modo de arreglo de 16 valores, usando la función llamada argConvGlpara. Estos 16 valores corresponden a la posición y orientación real de la cámara, por lo cual al utilizarlos para definir los valores de la cámara virtual produce que cualquier objeto gráfico a dibujarse aparezca exactamente alineado con el patrón físico.

```
/* carga los datos de la matriz */
argConvGlpara(patt_trans, gl_para);
glMatrixMode(GL_MODELVIEW);
glLoadMatrixd( gl_para );
```

La posición de la cámara virtual es definida utilizando la función OpenGL glLoadMatrixd (gl_para). La última parte del código es la renderización del objeto 3D propiamente dicha. En el código a continuación se dibuja un cubo azul con una iluminación de color blanco:

```
glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
glLightfv(GL_LIGHT0, GL_AMBIENT, ambi);
glLightfv(GL_LIGHT0, GL_DIFFUSE, lightZeroColor);
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_flash);
glMaterialfv(GL_FRONT, GL_SHININESS, mat_flash_shiny);
glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
glMatrixMode(GL_MODELVIEW);
glTranslatef( 0.0, 0.0, 25.0 );
glutSolidCube(50.0);
```

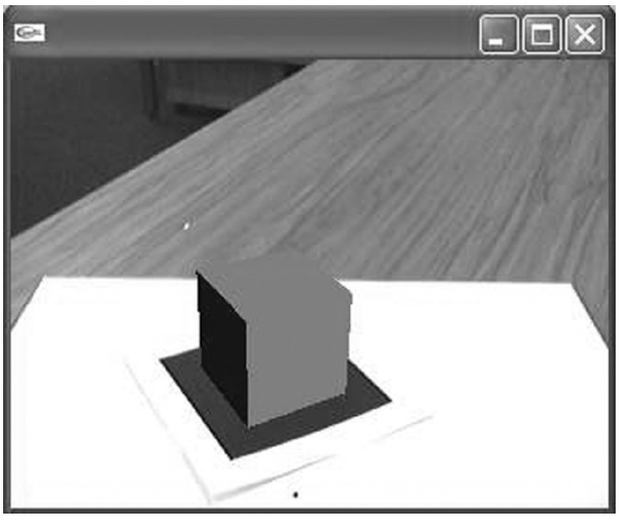


Figura 4: ejemplo de programa básico implementado en ArtoolKit
http://www.hitl.washington.edu/artoolkit/documentation/devstartup.htm

Estos pasos mencionados anteriormente (del 2 al 5 correspondientes al esquema de la Figura 3) se repiten indefinidamente mientras se ejecute la función mainLoop.

3.1.4. Cleanup

Al ser llamada la función cleanup, se detiene el procesamiento de video y se libera el disposivido de video, quedando disponible para otras aplicaciones:

```
arVideoCapStop();
arVideoClose();
argCleanup();
```

4. reacTIVision

http://reactivision.sourceforge.net/

Es un software de código abierto, un entorno para el reconocimiento rápido y efectivo de patrones en tiempo real.

Se diseñó principalmente como una herramienta para la creación de mesas tangibles a modo de interfaces. Basada en otras librerías, esta aplicación fue desarrollada por Martin Kaltenbrunner, en el Music Technology Group, en Barcelona, como parte del proyecto de reacTable, un instrumento musical que emplea las mesas tangibles anteriormente mencionadas.

Estas mesas tangibles están constituidas por una superficie translucida, sobre la cual se colocan diversos objetos que tienen la impronta de las diferentes marcas a reconocer (patrones).

Al ser ReacTiVision una aplicación ejecutable independiente, envía mensajes OSC (OpenSound Control) a través del puerto UDP, para lo cual se implementó el protocolo TUIO, especialmente diseñado para transmitir el estado de cada uno de los objetos. Es posible leer dichos parámetros desde otras plataformas utilizando una librería, como en el caso de TUIO client para la plataforma Processing

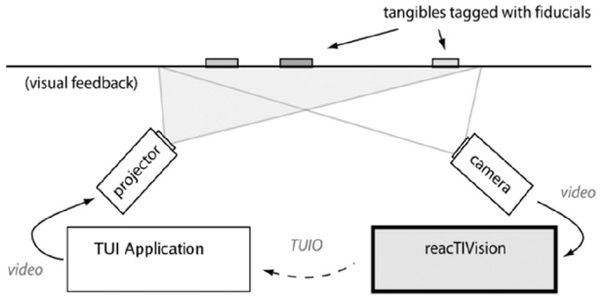


Figura 5: gráfico que describe el dispositivo empleado en reacTables (http://www.iua.upf.es/mtg/reactable)

4.1. Principales funciones

Tomando como referencia la librería anteriormente mencionada para Processing, a continuación mencionaremos los principales métodos y funciones necesarios para realizar una correcta lectura de los datos provenientes de reacTiVision y poder así construir una aplicación de interfaces tangibles.

En primer lugar es necesario crear una instancia cliente del tipo TuioProcessing client, de inmediato el cliente comienza a leer los mensajes provenientes de TUIO y genera eventos de nivel superior basados en los movimientos de los patrones o cursores.

```
TuioProcessing tuioClient = new TuioProcessing(this);
```

En este punto es importante destacar que reacTiVision no solo está diseñado para capturar patrones bitonales, sino que también puede detectar cursores, es decir, es capaz de captar las impresiones digitales a modo de puntos (dedos que se apoyan sobre la pantalla).

Cualquier aplicación que utilice reacTiVision para la captación de patrones o cursores necesitará implementar las siguientes funciones para ser capaz de recibir los eventos TUIO correctamente:

- addTuioObject (TuioObject tobj) se ejecuta cuando un nuevo patrón es detectado.
- removeTuioObject(TuioObject tobj) se ejecuta cuando un patrón ha sido eliminado.
- updateTuioObject(TuioObject tobj) se ejecuta para establecer el movimiento de un patrón, ya sea ubicación o rotación.
- addTuioCursor(TuioCursor tcur) se ejecuta cuando un nuevo cursor es detectado.
- removeTuioCursor(TuioCursor tcur) se ejecuta cuando un cursor ha sido eliminado.
- updateTuioCursor(TuioCursor tcur se ejecuta para establecer el movimiento de un cursor, ya sea ubicación o rotación.
- refresh(TuioTime bundleTime) actualiza continuamente la pantalla.

Cada TuioObject or TuioCursor está identificado con un único identificador, llamado SessionID, el cual se mantiene durante la existencia en pantalla del patrón.

A su vez, cada objeto de tipo patrón posee también un SymbolID que corresponde al número de patrón al cual está asociado (el diseño, la referencia gráfica impresa). En el caso de los objetos de tipo cursor, poseen un CursorID, el cual es siempre un número dentro del rango de todos los cursores detectados en ese momento.

Es posible obtener el valor de esos ID a través de los métodos getSessionID(), getSymbolID() o getCursorID().

Todas las referencias a los objetos son actualizadas automáticamente por el cliente TuioProcessing y siempre refieren a la misma instancia de los objetos correspondientes durante su tiempo de vida.

Por otro lado, los atributos de los objetos, tanto patrones como cursores, están encapsulados, y puede accederse a ellos utilizando métodos como los siguientes:

- getX() devuelve la posición en el eje x
- getY() devuelve la posición en el eje y
- getAngle() devuelve el ángulo de rotación

Existen otros métodos que también permiten obtener la información de los valores de velocidad o aceleración, como así también métodos adicionales muy convenientes para el cálculo de distancias o ángulos entre objetos.

El método getPath() devuelve un vector de TuioPoint que representa el trayecto de movimiento del objeto.

Finalmente, la librería contiene algunos métodos para la determinación de los estados de los patrones o cursores, ya sea de manera conjunta o individual.

- **getTuioObjects()** devuelve un vector con todos los patrones actualmente en pantalla.
- **getTuioCursors()**devuelve un vector con todos los cursores actualmente en pantalla.
- **getTuioObject(long s_id)** devuelve un patron específico o NULL según la presencia del mismo.
- **getTuioCursor(long s_id)** devuelve un cursor especifico o NULL según la presencia del mismo.

5. Conclusión

Luego del recorrido realizado por las técnicas de programación descriptas, es posible concluir en primer lugar que el avance tecnológico de los últimos años ha favorecido el desarrollo de estas herramientas y contribuido a una redefinición de las interfaces, como así también de la simplificación en la implementación de estas herramientas que permiten el desarrollo de aplicaciones de realidad aumentada que hace algunos años hubieran parecido imposible de poner en práctica.

La utilización de estas herramientas en el proceso de creación artística y educativo es un aporte positivo, ya que pone a disposición de artistas y educadores un recurso más de expresión y comunicación, donde la tecnología contribuye a generar sentido y, en cierta medida, forma parte del discurso.

Referencias bibliográficas

Causa, Emiliano. *Desarrollo de una Aplicación con Interfaces Tangibles* (2008), <http://www.biopus.com.ar>.

<http://www.arduino.cc>.

<http://artoolkit.sourceforge.net/apidoc/files.html>.

<http://www.hitl.washington.edu/artoolkit/documentation>.
<http://mtg.upf.es/reactable/>.

<http://www.panda3d.org/apiref.php?page=ARToolKit>.

<http://www.processing.org>.

<http://studierstube.icg.tu-graz.ac.at/handheld_ar/artoolkitplus.php>.

<http://www.tuio.org/?processing>.
.