Universidad Politécnica de Valencia
Facultad de Bellas Artes
Departamentos de Escultura y Pintura
Máster Artes Visuales y Multimedia

Proyecto Final de Máster en Artes Visuales y Multimedia

Tipología: Proyecto Aplicado

Presentado por:

Carlos Yanes Díaz

Dirigido por:

Moisés Mañas Carbonell

[FIAMETTA 2.0.]

Interfaz Sonora de carácter digital

Introducción

El proyecto final de Master que presentamos se enmarca dentro de la tipología de proyecto aplicado. Así mismo creemos pertinente inscribirlo dentro de la línea que establece el Master de Artes Visuales y Multimedia, denominada Estética Digital, Interacción y Comportamientos, si bien en nuestro trabajo hacemos hincapié en la parte eminentemente práctica de la sublínea Interacción Humano Computadora, nos gustaría apuntar que otra línea comparte nuestro trabajo desde el punto de vista no sólo teórico sino también práctico, en concreto Lenguajes Audiovisuales y Cultura Social y su sublínea Arte sonoro.

La aplicación práctica del proyecto de Master presentado, se resuelve a través de la realización de un prototipo que reúne muchos de los procesos y procedimientos prácticos que se han estudiado durante el curso. Podríamos decir que esta parte del trabajo tiene un peso fundamental en la globalidad del proyecto, ya que la motivación principal de nuestro proyecto y su principal objetivo fue no sólo investigar el cómo sino también desarrollar, en la práctica, un prototipo para mejorar las capacidades performáticas de un usuario con un dispositivo electrónico sonoro, por medio de una interface totalmente adaptada al usuario lo más generativamente libre posible.

El proyecto, no sólo ha hecho hincapié en la realización de un prototipo físico (hardware) sino también se ha desarrollado y se ha explicado en esta memoria la capa de software desarrollada. Software y hardware construyen un interface usable con altos índices de transparencia entre la relación humano máquina y un nivel lúdico musical interesante, creando de ese juego de interacción un interesante catálogo de secuenciación electrónica del sonido.

Si bien somos conscientes de que la aportación fundamental de la investigación del proyecto, responde a la práctica y la memoria de la realización del prototipo, la cual darán a cualquier lector, las herramientas

y los fundamentos necesarios para realizar una interfaz propia, no debemos de olvidar que hemos intentado hilar conceptos, de una manera humilde y sencilla tales como los aspectos de la interacción del punto uno, la creación de un breve panorama sobre el interfaz, concepto y praxis, seguida de una relación que nos parecía importante y que posiblemente se ha presentado sutilmente, pero que aún así, nos ayudaba a comprender mucho mejor la parte práctica como era el aspecto técnico de lo Analógico y digital, así como un apartado más referencial donde apuntamos aspectos sobre el audio digital y una selección de experiencias de interacción con el usuario en el panorama contemporáneo internacional.

Si bien, hemos comentado las líneas del Master en las que inscribimos el proyecto, nos gustaría apuntar que este desarrollo/prototipo que presentamos está fundamentado en principios contemporáneos y culturales digitales denominados Circuit Bending[1], Do it yourself[2], Open Hardware[3] y Hacking Hardware[4].

Nos gustaría apuntar que si bien en todo el trabajo, el lector, detectará que hablamos del sistema, desde un punto de vista técnico-tecnológico no debemos de olvidar la naturaleza digital del mismo. Esta "naturaleza" tiene una gran complejidad para el ser humano en sí, el esfuerzo por hacer una comunicación más fluida entre humano y

[1] **Cicuit Bending** es la adaptación y personalización creativa de circuitos electrónicos entre los que generalmente se encuentran circuitos de bajo coste tales como juguetes, mandos a distancia, pequeños sintetizadores...

[2] **Do it Yourself** (Hazlo tu mismo) es un término que se comenzó a utilizar en los años 50 con la intención de ejecutar uno mismo tareas del hogar y bricolaje sin la necesidad de profesionales o expertos, posteriormente, el Punk Rock y más adelante el Indie Rock se apropiaron del termino , aplicándolo como filosofía o modo de vida.

[3] **Open Hardware** es la colección de artefactos, máquinas, dispositivos y cosas físicas que respetan la libertad de sus creadores de controlar su tecnología y al mismo tiempo pretenden compartir conocimiento y fomentar el comercio a través del intercambio abierto de diseños. (Tales como Arduino). fuente: http://alt1040.com/2011/02/open-hardware

[4] **Hacking Hardware** es, al igual que Circuit Bendin,g la personalización y adaptación creativa de circuitos electónicos, pero a diferencia de este, se centra en circuitos digitales y componentes informáticos.

computadora se basa en uno de sus puntos en la mejora de las interfaces que median entre ellos y en nuestro caso a través de un interfaz tangible como objetivo propuesto en nuestro ejercicio para intentar resolver cuestiones de usabilidad (transparencia, fluidez, intuición, etc.).

Objetivos

Para formular los objetivos de este trabajo nos es necesario responder a las siguientes preguntas:

- **¿Vamos a materializar la tesis de forma práctica?** Si, la intención es construir materialmente la idea, testearla y documentar el proceso.

- **¿Es un prototipo?** Si, buscamos la parte experimental y no definitiva de lo que construimos.

- **¿Qué tipo de prototipo?** Interfaz sonoro de naturaleza electrónico digital.

- **¿Qué tipo de función realiza?** Es una interfaz que reacciona a los gestos del intérprete, convirtiendo estos, en una serie de datos de entrada en datos digitales capaces de procesar por el ordenador.

- **¿Para quién está realizada?** El instrumento está configurado a mis necesidades como artista sonoro.

- **¿Para qué está concebido?** Está concebido para la realización de conciertos o performances, donde la experiencia sonora está

acompañada por la experiencia visual de los asistentes y es parte fundamental de la experiencia artística.

- **¿Cómo será el dialogo de la herramienta?** La interacción será de tres tipos:

    - **Directa.** (usuario-ordenador) cuando el instrumento reacciona a los gestos del artista.

    - **Indirecta.** Pre-establecida, el instrumento es programado para que realice una serie de funciones independientemente del gesto directo del artista.

    - **Intrínseca.** Matemática-randomizada (máquina-máquina) En este tipo de interacción se programa al instrumento para que actúe de forma autónoma, y provoque estímulos autogenerados, aunque previamente contemplados en rangos.

Tras estas preguntas tenemos el esbozo de los objetivos que buscábamos que a priori podríamos decir que eran muchos y variados. La búsqueda de referentes provenientes tanto del desarrollo conceptual, técnico-tecnológico y artístico relacionados con el tema de estudio, el análisis de los mismos y su implementación en el contexto del trabajo. Pero debemos subrayar como objetivo principal del trabajo la realización/desarrollo práctico aplicado de una interfaz sonora de naturaleza digital, que posibilite nuevas experiencias performáticas musicales en dos vertientes: bajo la experiencia y la visualización y cuyos sub-objetivos se podrían enunciar así :

- Búsqueda de una mejora de la experiencia como usuario, del músico, mediante una interface intuitiva, tangible y transparente.

- Búsqueda de una mejora en el entendimiento/relación de la secuenciación de audio, con respecto a una interface gráfica.

- Búsqueda de una mejora de la visualización (de los gestos del intérprete-performance), para los espectadores del concierto / experiencia.

- Comprobar, testear con nuestro prototipo resultante, si la relación de los espectadores de una interfaz grafica tradicional, en este tipo de performance sonoras, que a priori no permite ver con lo que realmente interactúa el músico, es menos atractiva e intensa en la relación sinérgica que se crea en la actuación que con interfaces tangibles.

Metodología

Respecto a la metodología hemos optado por utilizar una estructura sencilla, básica y clásica de ordenación de contenidos. Dividiendo el proyecto en dos partes fundamentales:

- Introducción teórica sobre los elementos y conceptos esenciales que afectan al proyecto, denominada “Marco teórico” y dividida en tres aspectos indispensables para comprender el trabajo práctico, como son fundamentales a nivel conceptual, técnicos y referenciales.

- Desarrollo práctico: Diseño, implementación, medición y evaluación del prototipo.

En el desarrollo práctico de los estudios/prototipos he recurrido a una metodología proveniente de la ciencia, a través de los estudios de usuario y la metodología de carácter práctico basada en el testeo, valoración y modificación de los diversos prototipos realizados.

Respecto a la bibliografía utilizada, teniendo presente la naturaleza práctica y aplicada de este proyecto, hemos optado por centrarnos en un sistema cualitativo de referentes. Ordenando e utilizando estos sin más pretensión que acompañar y ayudar al lector a entender los conceptos planteados en el trabajo.

La selección de referentes prácticos artísticos utilizados ha sido sencilla pero bajo una metodología cualitativa y descriptiva intentando seleccionar aquellas experiencias sonoras que compartían aspectos cercanos a nuestro interfaz prototipo tales como la visualización de resultados, el tipo de interfaz tangible o el modelo de interacción en la performance. Si bien son pocos, si debemos de apuntar que son ejemplos básicos que nos ilustran y acompañan al lector hacia la comprensión de lo que creemos que es el corpus importante de la tesina de Master presentada, su parte práctica.

I.MARCO TEÓRICO

1. Aspectos Relacionales

1.a. Human Computer Interaction

Human-Computer Interaction es la disciplina que estudia el diseño, evaluación e implementación de sistemas informáticos interactivos para el uso humano además del estudio de los fenómenos más importantes que le rodea.[5] La disciplina de Human Computer Interaction formaría parte de las disciplinas conocidas como *cognitive engineering,* que engloban aspectos tales como la psicología cognitiva, la informática, la lingüística, la antropología cognitiva, la sociología, la etnografía, la semiótica y la filosofía de la mente. Los posibles predecesores de la disciplina *Human Computer Interaction* podrían ser disciplinas tales como *Human Factors* y *Ergonomics.* [6]

i. Breve introducción histórica

Hasta finales de 1970, los únicos seres humanos que interactúan con las computadoras eran científicos con un alto nivel de conocimiento tecnológico y los profesionales de la informática. Con la aparición de las Personal Computers (PC) en torno a 1980 se democratiza el uso de las computadoras, sobre todo en estados unidos. Durante este periodo, se sigue dando una rápida evolución tecnológica, dentro del entorno cotidiano. El mundo mecánico, de las máquinas de escribir y las máquinas

[5] ACM SIGCHI Curricula for Human-Computer Interaction
by Hewett, Baecker, Card, Carey, Gasen, Mantei, Perlman, Strong and Verplank
Copyright © 1992,1996 ACM SIGCHI

[6] CARROLL, M. *Human Computer Interaction(HCI).* Documento PDF, [Consulta:02/06/2011] Url: http://www.interaction-design.org/printerfriendly/encyclopedia/human_computer_interaction_hci.html

de calcular, empezó a ser rápidamente relevado por tecnología digital y más concretamente por el nacimiento de las computadoras personales.[7]

Los primeros estudios específicos de Human Computer Interaction aparecieron durante los años sesenta y se centraban sobre todo en el estudio de la relación Persona - Computadora, como por ejemplo el texto de Licklider *On-line Man Computer Communications* de 1962. Este autor afirmó anticipándose a la problemática posterior que el problema de la interacción hombre-computadora no es crear ordenadores productores de respuestas, sino ordenadores que sean capaces de anticipar y participar en la formulación de las preguntas.

En 1962 Licklider y Clark elaboraron una lista de 10 problemas que deberían ser resueltos para facilitar la interacción personas-computadora. Según él, los cinco primeros problemas deberían ser resueltos de manera inmediata, el sexto en un tiempo intermedio y los cuatro últimos, a largo plazo:

1. Compartir el tiempo de uso de los ordenadores entre muchos usuarios.

2. Un sistema de entrada-salida para la comunicación mediante datos simbólicos y gráficos.

3. Un sistema interactivo de proceso de las operaciones en tiempo real.

4. Sistemas para el almacenamiento masivo de información que permitan su rápida recuperación.

[7] RANDELL, B. *Digital Computers: Origins.* Documento PDF, [Consulta: 05/03/2011], Url:http://www.cs.ncl.ac.uk/publications/books/papers/128.pdf

5. Sistemas que faciliten la cooperación entre personas en el diseño y programación de grandes sistemas.

6. Reconocimiento por parte de los ordenadores de la voz, de la escritura manual impresa y de la introducción de datos a partir de escritura manual directa.

7. Comprensión del lenguaje natural, sintáctica y semánticamente.

8. Reconocimiento de la voz de varios usuarios por el ordenador.

9. Descubrimiento, desarrollo y simplificación de una teoría de algoritmos.

10. Programación heurística o a través de principios generales.[8]

Estos 10 principios resultan ser fundamentales, en el camino de una interacción fluida y accesible para cualquier persona, en la comunicación hombre-computador.

Diez años más tarde a estos principios, concretamente en 1973, la primera computadora personal es desarrollada en Xerox Parc. Estamos hablando del llamado Xerox Alto, diseñado por *Chuck Thacker*. Esta computadora es descrita como la primera computadora personal, así mismo, es el primero en utilizar la interfaz metafórica de escritorio y una interfaz grafica de usuario (Graphical User Interface - GUI) que más tarde adoptarían los sistemas operativos Macintosh y Windows. Xerox Alto incorporaba de serie la interfaz física del ratón, patentada por *Douglas Engelbart* en 1963, como elemento novedoso y usable en la interacción.

[8] LICKLIDERY, J. Y CLARK, W.; *On-lineManComputerCommunications*, MIT, 1962.

ii. Fundamentos básicos

Es nuestra intención, en este punto, mostrar sencillamente los ejes/principios fundamentales que tiene un sistema interactivo. Para ello debemos nombrar a *W.J. Hansen* que en su artículo *User Engineering Principles for Interactive Systems* presentado en AFIPS, en noviembre de 1971, hace la primera enumeración de principios para el diseño de sistemas interactivos, la cual nos parece más que relevante para la época. La enumeración dice así:

1. Conocer al usuario

2. Minimizar la memorización, sustituyendo la entrada de datos por la selección de ítems, usando nombres en lugar de números, asegurándose un comportamiento predecible y proveyendo acceso rápido a información práctica del sistema.

3. Optimizar las operaciones mediante la rápida ejecución de operaciones comunes, la consistencia de la interfaz y organizando y reorganizando la estructura de la información basándose en la observación del uso del sistema.

4. Facilitar buenos mensajes de error, crear diseños que eviten los errores más comunes, haciendo posible deshacer acciones realizadas y garantizar la integridad del sistema en caso de un fallo de software o hardware.[9]

No sólo debemos seguir estos principios básicos que apunta Hansen sino que deberíamos desarrollar procesos de desarrollo de sistemas interactivos. Estos actualmente se conciben principalmente en 4 etapas:

[9] LICKLIDERY, J. Y CLARK, W.; *On-lineManComputerCommunications*, MIT, 1962.

1. **Diseño:**

   •Análisis de requerimientos del producto.

   •Análisis de las tareas.

   •Conocimiento del usuario.

   •Generación de posibles metáforas y análisis de tipo de diálogo.

   •Revisión de posibilidades para la implementación.

2. **Implementación:**

   •Generación de prototipos (profundos o amplios, para investigación general o de ajustes).

   •Desarrollo de la aplicación, sitio o sistema.

3. **Medición(test de usabilidad):**

   •Planificación (desarrollo del plan, definición de las medidas, selección de participantes, formación de observadores, preparación de los materiales).

   •Test (prueba piloto, tests con usuarios).

4. **Evaluación:**

   •Conclusión (análisis de los datos, elaboración del informe, resultados y recomendaciones).

   •Comparación contra estándares (internos y/o externos), versiones anteriores del mismo producto y productos competidores.

   •Verificación de las diferencias.

   •Generación de nuevas metas.[10]

[10] MAÑAS, M. *Interfaces. Reglas/Teorías*. Documento: PDF, [Consulta:04/07/2011], Url:http://personales.upv.es/moimacar/master/download/interfaces.pdf

iii. A modo de reflexión:

.

La comunicación que se da entre hombre y computadora, no es una comunicación entre dos entidades del mismo nivel, hay que pensar que las computadoras están creadas y construidas para el uso (realización de tareas) y comodidad (realización de acciones) del hombre, de ahí, que debamos tener en cuenta que la tecnología no siempre entraña un valor positivo para el hombre, sino que puede construir que la tarea y la acción se revelen contra el propio usuario de mil formas, como por ejemplo en forma de control. *Michel Foucault* en su libro *Vigilar y castigar nacimiento de la prisión*[11] o *Guilles Deleuze* en su texto; *Conversaciones, Post-scriptum sobre las sociedades de control*[12], entre otros, han escrito sobre el papel que representa la tecnología en las sociedades de control y las disciplinarias y por consiguiente el papel de la tecnología en el ejercicio del poder. La idea de que la tecnología es algo que solo sirve al hombre de manera positiva es una idea de la modernidad, que en la actualidad ha sido sustituida y parece evidente que la tecnología también sirve al poder y al control y el hombre es, en algunas ocasiones, más de las que querríamos, el que se adapta a la computadora y no al revés en pro de un ejercicio que a veces podríamos definir como supervivencia tecnológica.

No es necesaria la ciencia ficción para concebir un mecanismo de control que señale a cada instante la posición de un elemento en un lugar abierto, animal en una reserva, hombre en una empresa (collar electrónico). Félix Guattari imaginaba una ciudad en la que cada uno podía salir de su departamento, su calle, su barrio, gracias a su tarjeta electrónica (dividual) que abría tal o cual barrera; pero también la tarjeta podía no ser aceptada tal día, o entre determinadas horas: lo que importa

[11] FOUCAULT, M.; *Vigilar y castigar nacimiento de la prisión*, México, Siglo Veintiuno Ediciones, 1976.
[12] DELEUZE, G.; *Postdata sobre las sociedades de control*, Éditions de Minuit, París 1972

no es la barrera, sino el ordenador que señala la posición de cada uno, lícita o ilícita, y opera una modulación universal.[13]

Con esto, me gustaría poner de relieve que disciplinas como Human-Computer Interaction son situadas, en muchas ocasiones, en segundo orden debido a la dependencia que existe entre el mercado y las interfaces que estudia. Un ejemplo de esto es como en muchas ocasiones se hacen interfaces menos funcionales, no aconsejables, solo por tener un aspecto más vendible. Otro ejemplo, que hemos detectado, es la necesidad de adaptarse el/os usuario/s a las características de las interfaces gráficas y físicas que se venden en el mercado actual, ya que si no es vendible, es más compleja de encontrarla en un sistema liberal basado en la ley de la oferta y la demanda como el occidental. El *Circuit Bending, Do it yourself, Open Hardware y Hacking Hardware* nos ofrece, en este sentido, una solución al problema de las interfaces industriales que podemos encontrar en el mercado.

[13] DELEUZE, G.; *Postdata sobre las sociedades de control*, Éditions de Minuit, París 1972.p. 4

1.b. Interacción

Estamos pasando de la estética del mensaje a la estética de la interactividad.[14]

La interacción se puede definir como el intercambio de información entre dos o más participantes activos. El escritor y diseñador de videojuegos *Chris Crawford*[15] describe la interacción como un proceso iterativo de escuchar, pensar, y hablar entre dos o más autores. También podemos encontrar autores como *Joshua Noble*[16] que la definen así: La interacción ocurre a través de señales que son enviadas por los diferentes sistemas participantes en la interacción.[17] Si bien podemos ver infinidad de autores y definiciones estas que hemos utilizado son las que creemos más cercanas a nuestro desarrollo del trabajo ya que se basan sencillamente entre la relación técnica de los actores y la parte sinérgica de los autores.

En líneas generales, entre los modelos de sistemas interactivos y según el grado de interactividad humano-máquina podemos detectar tres tipos de interactividad, entendida como un sistema:

1) Sistema mediador: reacción puntual, simple, normalmente a un programa dado.

2) Sistema reactivo: Injerencia en un programa a través de la estructuración de su desarrollo en el ámbito de posibilidades dadas. Se

[14] LÉVY, P. *¿qué es lo virtual?,* Ed. Paidós, Barcelona, 1999
[15] **Chris Crawford:** Diseñador y escritor conocido por la creación de una serie de importantes juegos en la década de 1980, la mayoría relacionados con la familia ATARI.
[16] **Joshua Noble**: Diseñador e investigador que parte de su trabajo se basa en aplicaciones dínamicas para internet, ha sido profesor de la universidad de Tufts y parte del equipo de desarrollo de Adobe Systems.
[17] NOBLE, J. *Programming interactivity* , Sebastopol CA, O'reilly, 2009, p.5,6

trata de una interactividad de selección, que implica la posibilidad de acceso multidireccional a informaciones audiovisuales para la ejecución de operaciones predeterminadas por el sistema, y por lo tanto limitadas a éstas.

3) Sistema interactivo: Estructuración independiente de un programa que se da cuando un receptor puede actuar también como emisor. Se trata de una interactividad de contenido, en la que el interactuador dispone de un mayor grado de posibilidad de intervenir y manipular las informaciones audiovisuales o de otra naturaleza (como las robóticas) o, en sistemas más complejos, generar nuevas informaciones.

i. Tipos de Interacción:

Si nos ceñimos respecto al tipo de interacción según *Edmond Couchot*[18], existen dos tipos principales de interacción, la externa y la interna.

1. La interacción externa consiste en la interfaz humano-maquina, así como en las formas ofrecidas por el entorno, cuyos datos son procesados por ordenador mediante diferentes interfaces.

2. La interacción interna corresponde, al contrario, al comportamiento comunicativo entre los propios objetos virtuales que pueden generar modelos de comportamiento para la animación de los llamados actores de síntesis.[19]

[18] **Edmond Couchot:** Artista digital y teórico del mismo, fue director del departamento de artes y tecnología de la universidad de Paris desde 1982 hasta 2000.
[19] GIANNETTI, C. *Estética digital: Sintopía del arte, la ciencia y la tecnología*, Ed. Làngelot, Barcelona , 2002, p.119

ii. Niveles de interacción:

Si analizamos la interacción desde el punto de vista de sus niveles *Peter Weibel*[20] distingue entre tres niveles de interacción que tienen como punto de referencia el comportamiento y la conciencia. Estos tres niveles se los define como:

1. La interacción sinestésica, que consiste en la interacción entre materiales y elementos, como por ejemplo imagen y sonido, color y música

2. La interacción sinérgica, que se produce entre estados energéticos, como en obras que reaccionan al cambio en el entorno

3. La interacción comunicativa o interacción cinética entre personas y entre objetos.

En cualquiera de los casos, el entorno o contexto de la obra es determinante para la efectivación de la misma.[21]

En los campos de las ciencias de la información, de la comunicación, y del diseño industrial, hay discusión sobre el significado de la interactividad. Para ellos se utiliza el "punto de vista de la contingencia" de la interactividad, como algo básico y definen tres tipos como elementales y diferenciadores:

[20] **Peter Weibel:** fue Profesor de medios visuales de la Universidad de Artes Aplicadas de Viena (1984); de video y arte digital en el centro de estudios de los Medios de Comunicación en la Universidad de Nueva York (1985); director del instituto para los nuevos medios en la Städelschule, Frankfurt/Main (1989-1993), y finalmente director del museo Neue Gallerie en Graz, Austria.

[21] GIANNETTI, C. *Estética digital: Sintopía del arte, la ciencia y la tecnología*, Ed. Làngelot, Barcelona , 2002, p.120

1. No interactivo, cuando un mensaje no se relaciona con anterior mensajes. (Cine clásico)

2. Reactivo, cuando un mensaje se relaciona solamente con un mensaje inmediatamente anterior. (Puerta)

3. Interactivo, cuando un mensaje se relaciona con un número de mensajes anteriores y con la relación entre ellos (Wikipedia)[22]

iii. Estilos de interacción:

Respecto a la cuestión de estilo, entendiendo "estilo de interacción", como el modo de interactuar, podemos citar cinco estilos como principales:

1. Manipulación directa: Se caracterizan por la representación visual de objetos y acciones, los usuarios pueden de una manera inmediata interactuar con los elementos y observar sus resultados. El prototipo *Fiametta 2.0.* entraría dentro de este apartado. Otros ejemplos serían: metáforas de escritorio, herramientas de diseño asistido, sistemas de control y juegos.

2. Selección de menús: Se caracterizan por que el usuario lee una lista de elementos, seleccionan el más adecuado y observan el efecto. El mayor beneficio es la estructura clara para toma de decisiones, puesto que todas las posibilidades se presentan a un mismo tiempo.

[22] MAÑAS, M, *Reacción Vs Interacción*. *Algunos aspectos sobre interactividad*. Documento: PDF,[Consulta:04/07/2011],Url:http://personales.upv.es/moimacar/master/download/interactivo_reactivo.pdf

3. Formularios: Se caracterizan por la constante entrada de datos, normalmente incómodos utilizando una única selección de menú y la obligatoriedad del rellenado de los mismos.

4. Lenguaje de órdenes: Para usuarios habituales, los lenguajes de órdenes proporcionan una fuerte sensación de poseer el control. Los usuarios aprenden la sintaxis y pueden expresar posibilidades complejas. Los porcentajes de error de estos sistemas son bastante altos.

5. Lenguaje natural: Estilo de interacción donde la computadora y el usuario se interrelacionan de una manera natural, interlocutando entre ellos sin obstáculos de relación, por ejemplo de manera oral.[23]

[23] MAÑAS, M, *Reacción Vs Interacción. Algunos aspectos sobre interactividad*. Documento: PDF,[Consulta:04/07/2011],Url:http://personales.upv.es/moimacar/master/download/interactivo_reactivo.pdf

1.c. Interfaz

Interfaz es aquello que media entre dos individuos.[24]

La palabra interfaz proviene del inglés interface y está formada por el prefijo inter (entre) y el sustantivo face (cara). Básicamente, la palabra, hace referencia a un sistema que permite la comunicación entre dos entidades-individuos, ya sea un sistema tangible (en la comunicación hombre-máquina por ejemplo: teclado, ratón, sistema nervioso, manos, monitor...) o intangible (generalmente asociado a los lenguajes, por ejemplo: lenguaje humano, lenguajes formales de programación...).

Una interfaz, también es, la zona de contacto que existe entre dos componentes, entre usuario y aplicación, entre aplicación y usuario o entre dos usuarios. La interfaz es por tanto el entorno, mecanismo o herramienta que hace posible dicha comunicación. El nivel de transparencia, que tenga una interfaz, entre los dos sistemas que se comunican, definirá la nitidez de la interacción entre ambas partes.

La interfaz es lo que facilita la comunicación, la interacción, entre dos sistemas de diferente naturaleza, típicamente el ser humano y una máquina como la computadora. Esto implica, además, que se trata de un sistema de traducción, ya que los dos "hablan" lenguajes diferentes: verbo-icónico en el caso del hombre y binario en el caso del procesador electrónico.[25]

[24] HUTCHINS, E.; HOLLAN, J.; NORMAN, D., *Direct Manipulation Interfaces*, en Multi-Media Database System;Visual Database Systems, ed. T. L. Kunii, Elsevier Science Publishers, 1989.
[25] LEWIS, C, RIEMAN, J. Task centered user interface design:A practical introduction. USA 190pp .1993.

El concepto de interfaz es algo muy amplio, Otto E. Rössler y Peter Weibel en su texto *el mundo como interfaz*[26] nos habla de cómo la realidad es transformada por el que la observa, de esta manera se puede concebir el mundo y todo lo que lo compone como una gran interface, formada por subsistemas y pequeñas interfaces.

Dentro de las propias entidades-individuos existen otras sub-entidades-individuos, que se comunican a través de otras interfaces que conforman una red infinita de interfaces y entidades-individuos, que a su vez pueden llegar a ser materialmente no solo la entidad-individuo que median en sí, sino todo lo que le rodea al individuo. Por ejemplo, una persona puede ser una entidad-individuo que puede comunicarse con otras, sin embargo, cuando pensamos en la interfaz que mediará la comunicación, podemos precisar que todo su cuerpo podría ser parte de una o varias interfaces, así como el aire que le rodea o incluso el suelo que pisa.

Es por ello, que considero al concepto de interfaz demasiado amplio para nuestra investigación, y por ello discretearemos el concepto entendiéndolo y utilizándolo en este trabajo en muchas ocasiones como "interfaz de usuario (IU)", un concepto orientado a las interfaces que desarrollan la comunicación entre hombre-máquina y más concretamente, me centraré, en la interacción hombre- computadora. Este “interfaz” tiene una alta correlación entre verosimilitud y presencia multimediática como apunta *Piscitelli*[27] en su texto *Ciberculturas 2.0*.

Existe una alta correlación entre la “verosimilitud” de lo simulado y la presencia multidimensional y multimediática de las experiencias en vivo. Una interfaz convincente necesita de todas las propiedades de la

[26] RÖSSLER, O. E.; SCHMIDT, A. P.;WEIBEL, P. “El mundo como interfaz”, en AA.VV., Dinámicas Fluidas. I Festival internacional de arte, ciencia y tecnología, Ayuntamiento de Madrid, Madrid, 2002, pp. 59-64.

[27] **Alejandro Piscitelli:** Filósofo argentino , especializado en los nuevos medios, profesor titular del Taller de Procesamiento de Datos, Telemática e Informática en Ciencias de la Comunicación.

experiencia directa: color, animación, respuesta instantánea, simulación, "inteligencia".[28]

Por otra parte, *Paul Virilio*[29] apuntaba, con razón desde nuestro punto de vista, que el problema de la técnica es inseparable del lugar de la técnica. El lugar de la técnica debe ser, por consiguiente, el binomio inseparable cultura/sujeto y, por ende, los medios de comunicación empleados en y por la sociedad. Por lo tanto, una de las formas de acercarse al síntoma de la crisis de la imagen técnica sería a partir de la intraestructura y la trasplantación inherentes a la inserción de la interfaz humano-máquina —la acción— en este contexto cultura/sujeto.

La interfaz humano-máquina propicia cambios cualitativos radicales respecto a las formas de comunicación basadas en medios digitales, entre los cuales podemos destacar los siguientes: un replanteamiento del factor temporal (tiempo real, tiempo simulado, tiempo híbrido, simultaneidad); el énfasis en la participación intuitiva mediante la visualización y la percepción sensorial de la información digital; la generación de efectos de translocalidad (como en el caso de Internet) y de inmersión (como en el caso de sistemas de realidad virtual); y el acceso a la información mediante sistemas de conexión ramificada, de nexos o asociaciones pluridimensionales. Por otra parte, la interfaz da testimonio de la transformación de la cultura basada en la escritura, en las estructuras narrativas logocéntricas y los contextos «reales » hacia la cultura digital orientada a lo visual, sensorial, retroactivo, no lineal y virtual.

La interfaz humano-máquina repercute en la propia comprensión de la «arquitectura» de la comunicación, que deja de ser una metáfora de

[28] PISCITELLI, A.; CIBERCULTURAS 2.0, *en la era de las máquinas inteligentes.* Ed. Paidós contextos, Barcelona, 2002 , p.21

[29] **Paul Virilio:** Arquitecto, urbanista, filósofo, la mayoría de sus obras se basan en una visión integral de lo urbano y de la urbanística dentro de un sistema tecnológico avanzado, donde la velocidad, el tiempo, la información y las redes asumen un papel fundamental.

la construcción (articulación) del lenguaje que define un espacio concreto, para asumir una dimensión inmaterial e inestable, que ya no está más sujeta a un espacio físico ni a un tiempo secuencial determinado. (Seguimos hablando, aunque metafóricamente, de descarnación.)

Si abordamos, por ejemplo, el mundo de la creación que utiliza los nuevos medios digitales de interfaz no sólo como herramientas o recursos, sino también como fuente y materia intrínseca, como es el caso del arte interactivo, constataremos el estrecho vínculo existente entre el fenómeno visual y la acción.[30]

i. Interfaz de Usuario

Al añadir la palabra usuario al concepto de interfaz, el concepto toma una nueva dirección y nos explica que está referido a un tipo de interfaz que está construida por y para usarlo y en este casi el usuario a priori es el ser humano. Esta interfaz hace que la comunicación hombre-máquina y en nuestro caso hombre-computadora, sea potencialmente más eficiente.

En la medida que las computadoras se fueron haciendo cada vez más complejas, la comunicación entre el lenguaje humano y lenguaje de la computadora, podía ser cada vez más difícil si no se utilizaban unas interfaces adaptadas a la comunicación entre el lenguaje que manejan los humanos y el lenguaje en el que se mueve el ordenador.

La naturaleza intrínseca de la codificación digital, hace que los lenguajes que utilizan el hombre y el ordenador, empiecen a diferenciarse y alejarse cada vez más el uno del otro, a medida que los ordenadores se

[30] GIANNETTI, C.; *Reflexiones acerca de la crisis de la imagen técnica la interfaz y el juego Analisi,* Quaderns de comunicació i cultura. N.27 UAB. Barcelona. Documento HTML, [Consulta: 06/07/2011] Url: http://www.bib.uab.es/pub/analisi/02112175n27.htm

van haciendo más complejos. La mejora de esta comunicación se puede conseguir adaptando las interfaces de los lenguajes utilizados entre ambos, por medio de interfaces de usuario, para conseguir una comunicación óptima en ambas partes.

Debido al incremento de la complejidad de los circuitos digitales es necesario crear interfaces adaptadas al lenguaje humano. Un programador necesitará, para programar en *lenguaje máquina*[31], una interfaz de usuario programada, *el lenguaje ensamblador*[32], basado en Mnemónicos, es una abstracción que facilita el uso del lenguaje máquina a los seres humanos.

También es lícito apuntar que la interfaz de usuario es el mediador entre capas que ayuda a manipular la relación entre el objetivo, el actor y la tarea. Su herramienta básica es la acción y su nivel de comunicación puede ser simbólico, metafórico y cinético entre otros.

Algunos tipos y modelos de interfaz de usuario que podemos encontrar, por orden cronológico de invención, son:

- CLI - *Command Line Interface*: es una interfaz de usuario en la que una persona interactúa con la información digital a través de un entorno textual y órdenes escritas por el usuario por medio de un interfaz físico del tipo teclado.

- GUI - *Graphical User Interface*: es una interfaz de usuario en la que una persona interactúa con la información digital a través de un entorno gráfico de simulación.

[31] **Lenguaje máquina:** lenguaje de más bajo nivel, que puede estar formado por binarios , es el lenguaje que en última instancia ejecutan todos los ordenadores, cuando llega al microprocesador, para realizar un proceso.
[32] **Lenguaje ensamblador:** Lenguaje de bajo Nivel que implementa una representación de símbolos de codigo máquina binarios, basada en Mnemónicos

- TUI - *Tangible User Interface:* es una interfaz de usuario en la que una persona interactúa con la información digital a través del medio físico.[33]

- NUI - *Natural User Interface:* es una interfaz de usuario con la que una persona interactúa mediante gestos que se consideran naturales y no han de ser aprendidos exclusivamente para el uso de la interfaz.

Por lo tanto podemos apuntar que una interfaz de usuario adecuada permitirá mediar mejor con la programación de un circuito digital. Por esto poco a poco a medida que va avanzando la tecnología digital a finales del siglo XX, la realización y el diseño de una interfaz de usuario adecuada va adquiriendo protagonismo, que finalmente desemboca en la creación de disciplinas dedicadas exclusivamente a dicho propósito, como *Human computer Interaction*.

El prototipo *Fiametta 2.0.* será un híbrido de interfaz tangible y natural, en algunos casos responderá a gestos intuitivos que se realizarán con las manos, además de otros gestos, de posicionamiento de objetos, que serán gestos aprendidos.

[33] MAÑAS, M.; *Reacción Vs Interacción. Algunos aspectos sobre interactividad.* Documento: PDF,[Consulta:04/07/2011],Url:http://personales.upv.es/moimacar/master/download/interactivo_reactivo.pdf

1.d. Input / Output

Un modelo de entrada-salida relaciona la salida del sistema directamente con la entrada, sin importar la estructura interna del sistema representada por la ecuación de estado. El sistema es considerado como una caja negra a la que sólo puede accederse a través de sus terminales de entrada y salida.[34]

Para que se de la comunicación entre hombre-computadora es necesario que varias interfaces trabajen al mismo tiempo, tanto interfaces de entrada de datos (INPUT) como interfaces de salida de datos (OUTPUT).

La forma en que es transmitida una señal de información, ya sea de entrada como de salida, es cambiante, dependiendo desde donde es observada. Por ejemplo, si establecemos que el punto de vista parte del usuario, las señales visuales que genera un monitor son de input, ya que el usuario las recibe, sin embargo, si establecemos el punto de vista en la computadora, las señales visuales que genera el monitor son de output, ya que la computadora, las genera, para que sean recibidas por el usuario.

Como aportación final, podemos apuntar que otra muestra de la relación que existe entre los principios básicos que rigen la tecnología que manejamos y la percepción del mundo es el modelo económico input / output desarrollado por el premio nobel *Wassily Leontief*. El modelo se basa en analizar la interdependencia de industrias y el consumo en una economía.

[34] BRASLAVSKY, J.; *Sistemas No Lineales . Cap. 6 Estabilidad Entrada-Salida.* Documento: PDF, [Consulta: 12/05/2010]
Url:http://www.eng.newcastle.edu.au/~jhb519/teaching/snolin/material/cap06.pdf

1.e. El mundo como Interface

Por primera vez en la historia, la imagen es un sistema dinámico. [35]

Desde el comienzo en que la razón se impone como norma, la evolución cultural, ha cambiado la manera de comprender y percibir el mundo que nos rodea, tanto de una manera racionalista como empírica. La cultura evoluciona fundamentalmente a través de la tecnología, *el materialismo histórico* y *materialismo cultural* hablan de que los cambios en la infraestructura de una cultura están dados por los medios de producción y la tecnología. A su vez la tecnología es la que fundamenta los medios de producción porque al fin y al cabo la tecnología de la industrialización es la que determina dichos medios de producción.

De acuerdo con esto si la evolución cultural cambia nuestra manera de percibir y entender el mundo, entonces podemos decir que la evolución tecnológica cambia sensiblemente nuestra percepción y entendimiento del mundo.

Desde mi punto de vista, los desarrollos tecnológicos han tenido tal fuerza en nuestra manera de percibir el mundo a lo largo de la modernidad, que llegamos a percibir el único concepto que por su significado, intrínsecamente difiere en totalidad del significado de tecnología pero que sin embargo asimilamos de manera muy similar a este, el concepto de naturaleza.

En el renacimiento *Descartes* hablaba del funcionamiento de la naturaleza, como el de una "gran máquina" que estaba sometida a unas

[35] WEIBEL, P.; RÖESSLER, O.; *El Mundo como Interface.* Documento PDF, [Consulta: 24/09/2010] Url: http://www.elementos.buap.mx/num40/pdf/23.pdf

leyes matemáticas y mecánicas. Este concepto de naturaleza estuvo vigente durante todo el siglo XVIII.

Examinemos con alguna atención la economía física del hombre: ¿que halláis? Las quijadas armadas de dientes ¿que son sino unas tenazas? El estomago no es más que una retorta; las venas las arterias, el sistema entero de vasos, son tubos hidráulicos; el corazón es una maquina; las vísceras son filtros, cribas; el pulmón es un fuelle. ¿Que son los músculos sino cuerdas?¿Que es el ángulo ocular, sino una polea? Y así sucesivamente. [36]

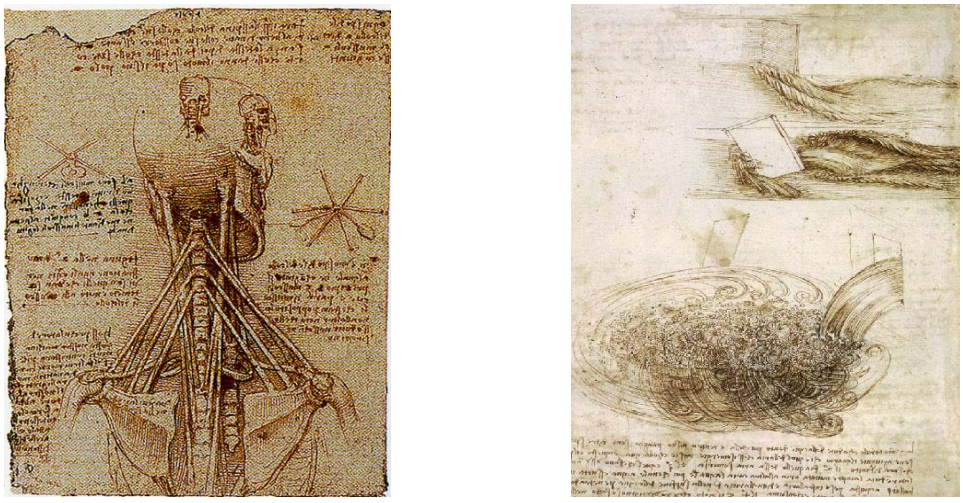

Fig. 1 ilustraciones Leonardo da Vinci

Leonardo Da Vinci trató mediante sus dibujos establecer un vinculo entre naturaleza y maquina, trató de explicar la naturaleza mediante el dibujo, con la concepción de "gran máquina" de la que habla Descartes. En el dibujo de la derecha trata de reproducir el movimiento del agua, con la que Leonardo estaba obsesionado, porque le era muy difícil de explicar y plasmar en un dibujo. Leonardo apunto de esta manera el problema:

[36] ALVAREZ C.J. *Descartes y la ciencia del siglo XVII.* ed. siglo xxi s.a., Mejico, 2000 Pag. 138 citando a Baglivi, Praxis Médica, 1696, citado en Canguilhem, 1979. p121.

"Al abordar un problema científico, dispongo primero diversos experimentos, ya que pretendo determinar el problema de acuerdo con la experiencia, mostrando luego por qué los cuerpos se ven obligados a actuar de ese modo. Ese es el método que hay que seguir en todas las investigaciones sobre los fenómenos de la Naturaleza"[37]

Peter Weibel y Otto E. Rössler en *El mundo como Interfaz*[38] nos habla de cómo percibimos el mundo por medio de diferentes realidades vistas a través de diferentes observadores.

la endofísica (ciencia que investiga el aspecto de un sistema cuando el observador se vuelve parte de él) muestra hasta qué punto la realidad objetiva depende necesariamente del observador. Para la endofísica, la posición de un observador externo sólo es posible en cuanto modelo, fuera de un universo complejo, no en el interior de la realidad misma. En este sentido, la endofísica ofrece una aproximación a un modelo general de teoría de la simulación (así como a las "realidades virtuales" de la era del ordenador).[39]

Según Weibel nosotros no interactuamos directamente con el mundo sino que interactuamos con la interfaz del mundo. En términos estrictos, cada una de las cosas que forman parte de lo que llamamos realidad son una interfaz con información. Nosotros, podemos percibir el mundo solo a través de la interfaz.

En la era de la electrónica, el mundo se está volviendo cada vez más manipulable en cuanto interfaz entre el observador y los objetos. La tecnología electrónica nos ha hecho comprender que sólo somos parte

[37] MORENO, I. *Leonardo da Vinci: el científico.* Documento PDF, [Consulta:05/02/2010] Url:http://centros5.pntic.mec.es/ies.victoria.kent/Rincon-C/Cie-Hist/Leonardo/ciencia.htm
[38] WEIBEL, P.; RÖESSLER, O.; *El Mundo como Interface.* Documento PDF, [Consulta: 24/09/2010] Url: http://www.elementos.buap.mx/num40/pdf/23.pdf
[39] "ibidem"

del sistema que observamos o con el que interactuamos. Por vez primera, también tenemos acceso a una tecnología y una teoría en la que el mundo se nos impone a modo de interfaz que sólo se puede ver desde dentro.[40]

En conclusión, en la actualidad, hay un desarrollo sin precedentes de la virtualidad. Y ese mundo virtual puede ser percibido y manipulado por medio de las interfaces. Peter Weibel en su texto nos hace ver como el elemento interacción entre el observador y la realidad es ahora, más importante que nunca. La información digital es siempre potencialmente variable, para acceder a la información digital es necesario volverla a reproducir y a diferencia de la analógica la información digital no se almacena en sistemas cerrados. Esto es debido a este carácter de la información digital, que permite, con mucha fluidez, el desarrollo de proyectos interactivos. La Interfaz es el elemento y concepto esencial de dichos proyectos.

[40] WEIBEL, P.; RÖESSLER, O.; *El Mundo como Interface.* Documento PDF, [Consulta: 24/09/2010] Url: http://www.elementos.buap.mx/num40/pdf/23.pdf

2. Aspectos Técnicos.

2.a. Analógico vs Digital

Es importante apuntar que para entender la parte práctica necesitamos que el lector comprenda las diferencias técnicas entre analógico y digital. Para ello la intención de este capítulo no es otra que aclarar esos conceptos.

Existen dos tipos de electrónica en la que puede estar basado un circuito, la diferencia entre ambas viene a hablar sobre el tipo de variables que utiliza para los cálculos:

- Electrónica Analógica
- Electrónica Digital

i. Electrónica Analógica

Se denomina electrónica analógica a la que utiliza señales analógicas como variable de cálculo dentro de un circuito.

Las señales analógicas vienen dadas en magnitudes físicas que los circuitos interpretan tal cual les llega, como pueden ser el voltaje, intensidad o potencia a la que se le puede sumar la variable temporal de dicha señal. También se consideran señales analógicas a las señales hidráulicas, de luz, mecánicas, de presión, de peso... Las señales que recibimos a través de nuestros sentidos se consideran también señales analógicas.

Las señales analógicas, en teoría, pueden tomar infinitas posiciones dentro de un rango. En un circuito analógico, la capacidad de emisión y recepción de estas señales serán lo que defina la amplitud de variables de estas señales analógicas.

También podemos decir, en términos prácticos, que la electrónica analógica es toda aquella, que no sea electrónica digital, es decir toda electrónica que se sirva de valores "reales"(analógica) y no de valores en código (digital).

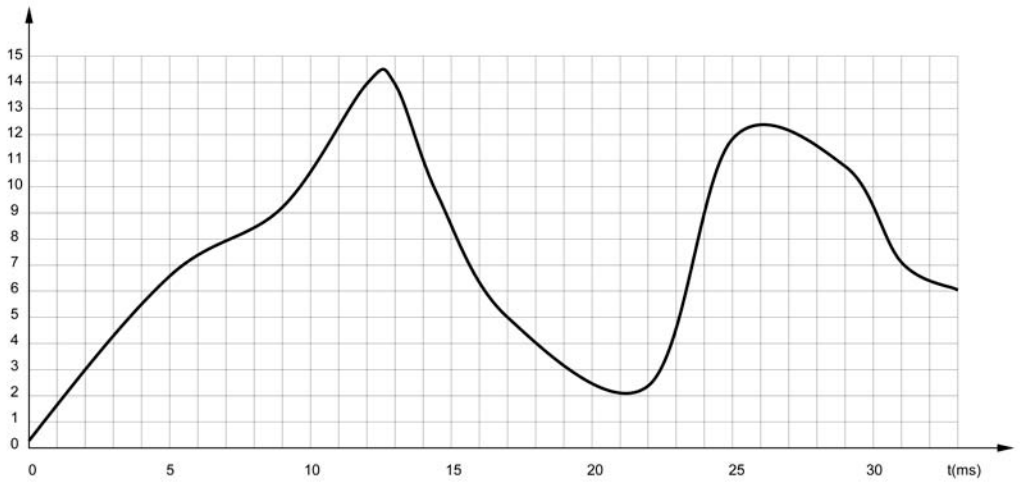

Fig. 2 Función continua (analógico)

Esta sería la representación de una señal analógica que viene dada en funciones continuas.

El hecho de que un circuito electrónico analógico, se sirva de las variables analógicas, es decir, variables reales sin codificación hace que el proceso de comunicación entre las partes y sub-partes de un circuito sea casi inmediato, esto repercute en que en un circuito que este hecho con electrónica analógica, nunca veremos *la latencia*[41] a la que estamos

[41] **Latencia**: Es la suma de retardos temporales dentro de una red. Un retardo es producido por la demora en el procesamiento y transmisión de información dentro de un circuito digital.

acostumbrados en un circuito digital. En este sentido los circuitos analógicos son menos complejos pero generalmente más fiables en cuento a funcionalidad asegurada. Un circuito analógico generalmente será menos preciso que uno digital pero más inmediato.

ii. Electrónica digital

Se denomina electrónica digital a la que utiliza señales digitales como variable de cálculo dentro de un circuito.

Las señales digitales a diferencia de las señales analógicas vienen codificadas en señales discretas. Digital no es sinónimo de binario, ya que una función digital puede estar compuesta por varios números enteros que codifican una señal real. Una representación de una señal digital en funciones discretas tendría este aspecto:

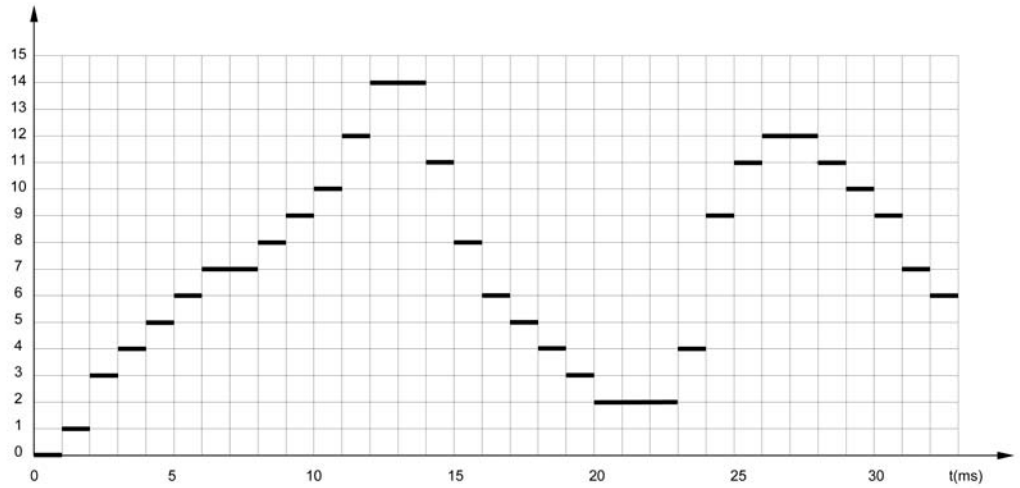

Fig. 3 Función discreta (digital)

Los actuales ordenadores utilizan un código digital binario. Es decir que su código se basa de dos estados: puede ser de baja o alta tensión, nosotros las representamos con 0 y 1, a estas señales individuales se le

llaman bits y componen las unidades mínimas con un significado que pueden leer un circuito digital: el Byte.

El número de Bits que compone un Byte depende del tipo de código que utiliza el circuito digital, siendo el número más normal y que actualmente utilizan los ordenadores el de 8 bits. *Wener Buchholz* propuso el término byte en 1957 durante el periodo en que diseñaba el ordenador IBM 7030 Stretch.

El número de bits que utiliza un Byte proporciona diferentes posibilidades en el código y significado de los bytes, si un byte se compone de 8 bits esto le dará la posibilidad de tener 256 combinaciones diferentes, en un rango de 1 - 256. A partir de los bytes el circuito digital puede ejecuta procesos o funciones, ya sea por la combinación de varios Bytes o por cada uno de ellos.

iii. Conclusión

La gran diferencia entre un circuito analógico y un circuito digital es que lo analógico trabaja con valores reales representado en funciones reales y lo digital con valores en código, representados en funciones discretas.

Las siguientes cuestiones son las más importantes a tener en cuenta en las ventajas e inconvenientes la electrónica analógica y electrónica digital:

- **Ruido:** Debido a que los valores de las señales analógicas se transmiten en valores reales, es mucho más frecuente ver la aparición de ruidos en los circuitos

analógicos que en los circuitos digitales, ya que un pequeño cambio en la señal representa un cambio en la información presente. La perturbación de las señales digitales es más infrecuente ya que la variación de la señal ha de ser mucho más significativa para que se aprecie. Por lo que es más difícil generar ruido en un circuito digital.

- **Latencia:** Los circuitos analógicos no tienen tiempo de retardo, ya que las señales no tienen que pasar por un procesador. Los circuitos analógicos trabajan con las señales directas. Aunque se puede reducir mucho, hasta ser casi imperceptible, los tiempos de retardo de un circuito digital, siempre existirán debido a los procesos que necesita ejecutar en la codificación y descodificación de las señales.

- **Estandarización:** Los circuitos digitales, debido a su naturaleza codificada, son capaces de poseer estándares que son intercambiables, es por ello que en un ordenador podemos tener todo tipo de aplicaciones con diferentes funcionalidades. Sin embargo en los circuitos analógicos necesitaríamos una máquina distinta para cada cometido.

- **Almacenaje de información:** Un sistema digital tiene la capacidad de almacenar grandes cantidades de información, que además cuentan con una exactitud y fidelidad mucho mayor que los sistemas analógicos debido a la capacidad en la reproducción de las señales digitales en contraposición a las señales analógicas.

- **Factores económicos:** Los sistemas digitales encarecieron su precio debido a la producción en cadena de sus componentes, además este abaratamiento se acentuó con la estandarización de la electrónica digital de consumo. Además de esto, los sistemas digitales no presentan grandes problemas de tolerancia, entre sistemas, algo que se acentúa mucho más en los sistemas analógicos.

3. Aspectos Referenciales

3.a. Audio Digital, una introducción a la temática de estudio.

i. Breve Introducción histórica técnica

La modulación de sonido por impulsos codificados, es decir lo que llamamos audio digital, fue inventada por el científico británico *Alec Reeves* en 1937, utilizada en primer lugar como tecnología para telefonía. Si bien ya existían experimentos de codificación digital, Reeves fue el primero en diseñar una codificación digital para audio analógico. La intención que llevo a inventarla fue el hecho de que cada vez que se amplifica una señal de audio analógica, también se amplifica el ruido nuevo que la misma genera, además del ruido que ya la acompañaba, esto conlleva a que al intentar enviar una señal de audio analógica por grandes distancias, la fidelidad de la señal original se pierde irremediablemente. [42]

La modulación por impulsos codificados, se puede basar en códigos binarios de 0 y 1, este tipo de códigos son muy concretos y exactos, por lo que una señal puede regenerarse cuantas veces se necesite sin sufrir ningún cambio o variación de la señal original. Es, por tanto, que el contenido de ruido de la señal no se incrementa, a diferencia de la amplificación normal de una señal analógica.

[42] SHELDON, T.P.; *The Dawn of Commercial Digital Recording* Documento PDF [COnsulta:06/11!2010] Url: http://www.aes.org/aeshc/pdf/fine_dawn-of-digital.pdf

Reeves patentó la modulación de sonido por impulsos codificados en 1938. Sin embargo, la idea de Reeves no fue llevada a la práctica en la época, debido a que requiere un circuito muy complejo y costoso, ya que el circuito estaba integrado por válvulas termoiónicas.

El siguiente paso en la historia del audio digital será la invención del transistor. El transistor sustituirá a las válvulas termoiónicas en los circuitos digitales, y con ello, se conseguirá el abaratamiento de dichos circuitos. Su historia se remonta a 1925, año en el que Lilienfeld comienza a patentar dispositivos que hoy llamaríamos transistores. Ninguno de sus inventos llegó a desarrollarse comercialmente dado el escaso conocimiento de su fundamento físico y el bajo nivel de la tecnología de la época. El siguiente avance de interés, en la historia del transistor, se produce en 1947, cuando Bardeen, Brattain y Shockley, en los laboratorios de la *Bell Telephone*, fabrican el primer transistor, y se inventa su sucesor, el transistor de unión bipolar, en 1949. A partir de 1949 se fueron reduciendo las deficiencias en los procesos de fabricación, culminando con la aparición de transistores en el mercado en 1952.

Paralelamente a esta revolución tecnológica se fueron desarrollando los primeros computadores. En 1946 aparece el primer computador electrónico de propósito general que funcionó satisfactoriamente, el ENIAC I. El computador estaba construido con tecnología anterior al transistor, a base de válvulas de vacío y relés, entre otros. Su instalación requería una superficie de 140 m^2, y pesaba más de 30 toneladas. Más adelante, en 1951, se desarrollaría el primer computador comercial de propósito general fabricado en serie, el UNIVAC I, que también estaba compuesto con dispositivos electrónicos de la primera etapa como válvulas de vacío y relés.

Hasta aproximadamente 1957 se diseñan y construyen otros computadores (como el IBM 650) todos ellos con dispositivos electrónicos

de la primera etapa, que constituyen la llamada primera generación de computadores.

En 1954 los laboratorios de Bell Telephone construyeron el primer computador transistorizado, el TRADIC, marcando el inicio de lo que se conoce como segunda generación de computadores. Con la introducción del transistor, la industria de los computadores creció rápidamente. En 1957 se implanta la utilización de dispositivos semiconductores discretos, como el transistor, continuando así la segunda generación de computadores con componentes semiconductores entre los que destacan el IBM 7090, 7094 y UNIVAC 1004.[43]

El desarrollo del transistor permitió abaratar enormemente los costes de los circuitos para desarrollar circuitos electrónicos digitales y señales mediante impulsos eléctricos codificados.

ii. Breve introducción referencial de Piezas compuestas con/para audio digital.

El ordenador australiano CSIRAC, fue el primero en ejecutar una reproducción de audio digital. *Trevor Pearcey y Maston Beard* diseñaron el computador. El programador *Geoff Hill* programó el CSIRAC para que interpretase versiones de temas musicales populares de la época. En 1951 interpretó públicamente *Colonel Bogey March*, de dicha reproducción no existen grabaciones conocidas.

Otro de los ordenadores, pioneros en reproducciones de audio digital, estaba ubicado en la universidad de Manchester. El ordenador, llamado Ferranti Mark I*,* o también conocido como el *Manchester*

[43] CASTILLO P.A.; *Estructura de Computadores I.* Documento PDF. [Consulta:14/05/2011] Url:http://atc.ugr.es/pedro/docencia/ec1/transparencias/evolucion_tecnologia-computadores.pdf

Electronic Computer, ejecutaba unos comandos que daban la capacidad de variar el sonido en frecuencia y duración. El sonido generado por el ordenador constituye la primera grabación, de audio generado por ordenador, de la que se tenga constancia. La grabación del audio fue realizada por la BBC en otoño de 1951, en ella se interpretaron varias versiones digitales y monofónicas entre las cuales se encontraban *Baa Baa Black Sheep, God save the King y In The Mood*[44].

La primera composición creada y compuesta, por y para un ordenador, fue la *Illiac Suite para cuarteto de cuerda,* creada por *Lejaren Hiller* y *Issacson Leonard* en 1956. Se trata de una obra en la que los autores crearon un sistema de reglas que se programaron para el ordenador llamado *Illiac*. Cada evento sonoro fue calculado mediante procesos aleatorios, tales como tonos, ritmos y dinámicas. Consta de cuatro movimientos:

- *Primer movimiento 4:38*. Se basa en una serie de tonos generados al azar y sometidos, de dos en dos, a las reglas del contrapunto del siglo XVI.
- *Segundo movimiento 4:01.* Se basa en el contrapunto aplicado rígidamente.
- *Tercer movimiento 5:59.* Es un estudio de las variaciones rítmicas y dinámicas.
- *Cuarto Movimiento 3:29.* Se basa en el dodecafonismo, forma de música atonal, con una técnica de composición en la cual las 12 notas de la escala cromática son tratadas como equivalentes, es decir, sujetas a una relación ordenada que (a diferencia del sistema mayor-menor de la tonalidad) no establece jerarquía entre las notas. Este modelo se encuentra por ejemplo en algunos comandos del Max MSP.

[44] FILDES, J.; *Oldest computer music unveiled.*BBC News [Consulta:25/03/2011] Url:http://news.bbc.co.uk/2/hi/technology/7458479.stm . Ejecutar el reproductor, para escuchar la grabaciones de sonido hechas del CSIRAC.

Los tres primeros movimientos del cuarteto de cuerdas se basan en las reglas tradicionales de composición (polifonía simple, contrapunto, técnica tone row), pero el cuarto movimiento funciona con las llamadas «cadenas Markow», un principio técnico y matemático, que es la única base en este movimiento. Una vez calculados los resultados por el ordenador, son transcritos en la notación tradicional como si fueran a ser tocados por instrumentos normales. Cabe decir que *Illiac Suite* no fue la primera composición algorítmica, tiene predecesores como los *Musikalisches Würfelspiel*, S XVIII: Un sistema para crear música aleatoria por medio de dados o el *Panharmonicon* 1805: Construido por *Johann Nepomuk Mälzel*, fue una de los primeras maquinas construidas para hacer música aleatoria.[45]

El audio generado de manera digital, en principio, no se tomo muy en serio, debido a que los sintetizadores y el audio analógico estaban muy avanzados y era capaz de hacer sonidos de mayor complejidad, en comparación con los primeros experimentos de audio digital.

En otra vertiente, *Max Mathews* elaboró en 1957, durante su trabajo en los *Bell Labs*, el primer programa de ordenador utilizado para la generación de sonido, con una interfaz orientada al usuario, es decir, el programa ponía en mano de otras personas, una interfaz de usuario capaz de generar sonido. El programa fue el llamado *MUSIC I* que culminaría en el *MUSIC V*, un programa mucho más robusto. Más adelante se crearía toda una familia de programas *MUSIC-N*. Todos los programas *MUSIC* tenían un entorno y un diseño más o menos común formado por una biblioteca de funciones aplicada en torno a procesos de señales y síntesis de audio. La serie de entornos *MUSIC-N* fue, por tanto, el padre de las herramientas digitales, que años más tarde, a principios de

[45] MEDIEN KUNST NETZ. Web en Linea. [Consulta 20/04/2011]
Url:http://www.medienkunstnetz.de/works/illiac-suite/

la década de los 90 del siglo XX se utilizarían para la generación de audio digital en programas como *Max y PureData*. [46]

En 1967, *John Cage* y *Lerajen Hiller* (compositor de la *Illiac Suite*) colaboran en *HPSCHD*, una composición para clavicordio y sonidos generados por ordenador. Fue escrita entre 1967 y 1969 y se estrenó el 16 de mayo de 1969, en la *University of Illinois*. El trabajo se fundamentó en el procesamiento a través de la computadora de siete solos de clavecín (harpsichord) obteniendo 52 bandas de sonido. El complejo planteamiento de HPSCHD supuso una laboriosa experimentación de las posibilidades de la informática en el tratamiento del sonido que, además de las derivas algorítmicas del I Ching, tuvo como referencia las famosas instrucciones para componer valses al azar mediante el uso de dados atribuidas a Mozart. Instrucciones que extendió a otros compositores, clásicos y contemporáneos, haciendo de este modo de HPSCHD una metafórica "historia de la música". La presentación de HPSCHD fue un gran despliegue multimedia que contó con la proyección de imágenes espaciales de la NASA y de miles de diapositivas desde el centro de un espacio circular marcado por las fuentes sonoras y por el que el público deambulaba libremente.[47]

A finales de los años 60 comienza a darse lo que se conoce como la popularización de la música electrónica. En gran medida fue posible gracias al abaratamiento de los sintetizadores de audio, y en especial al diseñador de sintetizadores *Robert Moog,* que fue pionero en desarrollar sintetizadores modulares que ,por su bajo coste, permitan ser comprados por gran cantidad de músicos.

[46] HASS, J.; *Introduction to computer Music: Volume one.* Chapter Five. Web en linea. [Consulta 20/04/2011] Url:http://www.indiana.edu/~emusic/etext/digital_audio/chapter5_digital2.shtml
[47] LOPEZ A.; *Monografico sobre John Cage.* Documento PDF[Consulta 22/04/2011] Url:http://217.16.255.8/Obert/EP00006/swf/pdf/es/PDFcage.pdf

Artistas como Wendy Carlos popularizan la música electrónica. Con su disco, Switched-On Bach, lanzado en 1968, en el que tocaba conciertos de Johan Sebastian Bach con un sintetizador Moog Modular, consiguió ser disco el disco de música clásica más vendido de la historia con 500.000 copias vendidas.

La banda *Kraftwerk* popularizaría aun más el sonido de los sintetizadores, de manera más internacional. Con su disco *Authoban* consiguieron un éxito sin precedentes además de ser una música que a diferencia de la de Wendy Carlos había sido compuesta expresamente para ser tocada con sintetizadores. Kraftwerk era un grupo experimental que se había formado en el pleno auge del movimiento KrautRock alemán. El KrautRock fue un movimiento experimental con influencias del rock psicodélico, el rock progresivo, la música avant-garde, el rhythm and blues y el jazz. El movimiento se caracterizó por la experimentación con las nuevas tecnologías y sobretodo utilizar nuevas formas de usar esas tecnologías ya fueran de grabación, amplificación y mezcla musical, y además con nuevas estructuras formales. Los músicos de Kraftwerk provenían de otras bandas experimentales de Krautrock como Can, Neu!, Cluster y Harmonia.

A finales de los 70 y en adelante, surge el movimiento conocido como *SinthPop*, género que normalizaría el uso del sintetizador en la música popular, precursores de este movimiento son entre otros: *Kraftwerk*, *Roxy Music, Donna Summer, David Bowie y Devo*. Estos músicos inspirarían a una larga lista de bandas que harían del sintetizador un sonido referencial de la música más popular de toda la década de los ochenta. Cabe mencionar también a *Jean Michel Jarre,* compositor que trabajo en las vertientes del new age y sinth pop y en los años 80 y popularizó la *Laser Arp,* instrumento electrónico, en un principio analógico, que consta de varios láseres, que pueden ser tocados como si las cuerdas de un arpa se trataran. Los sensores miden la luz que las

manos refleja al chocar el haz de luz laser, en este sentido es muy parecido a lo forma en que *Fiametta 2.0.* reacciona a la interacción con los objetos que el usuario de *Fiametta 2.0.* utiliza.

En 1975, la empresa japonesa *Yamaha* adquiere la licencia de los algoritmos *Frequency Modulation Synthesis (FM synthesis)* de *John Chowning,* con los que había experimentado en la *University of Stanford* desde 1971. Los ingenieros de *Yamaha* comenzaron a adaptar el algoritmo de *Chowning* para su uso en un sintetizador digital, añadiendo mejoras, como la "key scaling" método para evitar la distorsión que se producían en los sistemas analógicos de modulación de frecuencia. Sin embargo, el primer sintetizador digital comercial que se lanzará será el australiano *Fairlight CMI* (Computer Musical Instrument) en 1979, con sonido polifónico de síntesis digital y lanzador de samplers, aunque no fue muy popular debido a su elevado coste, rondaba los 25000 $ en 1979.

En 1980, Yamaha finalmente lanzó el primer sintetizador digital de FM synthesis, la Yamaha SG-1, pero su precio era excesivo. Finalmente en 1983, Yamaha presentó un sintetizador digital con *FM synthesis*, el *DX-7*, que se convertiría en uno de los sintetizadores más vendidos de todos los tiempos, debido a bajo coste. El *DX-7* fue conocido por sus tonalidades brillantes que se debió en parte a su frecuencia de muestreo de 57 kHz. algunas de sus características son la monotimbralidad (acceso a sólo un sonido a la vez), polifonía de 16 notas, 32 programas de memoria y teclado de 61 teclas con sensibilidad al velocity (intensidad) y al aftertouch (velocidad de salida del dedo). Incluye los puertos MIDI, pero como fue lanzado un poco antes de que el sistema fuese terminado, no soporta por completo el standard; sin embargo, interconecta con la mayoría de los equipos MIDI.[48]

[48] LIZANA, D. *Música y Media.* Web en linea. [Consulta 23/04/2011] Url: http://musicaymedia.blogspot.com/2007/08/yamaha-dx7-el-hito.html

Con la aparición de los Sintetizadores FM y su abaratado coste, como la serie DX de Yamaha, muchos fueron los músicos que comenzaron a cambiar sus sintetizadores analógicos por sintetizadores digitales, ya que estos empezaron a ofrecer un abanico de posibilidades que los analógicos ya no podían ofrecer, esta demanda se acentuó con la entrada del protocolo MIDI, que permite la comunicación entre diferentes dispositivos musicales digitales.

3.b. Referentes digitales artísticos/creativos seleccionados.

En este apartado hemos seleccionado algunos referentes actuales que creemos que tienen relación bastante directa. No es nuestra intención crear un eje documental exhaustivo sino seleccionar aquellos ejemplos esenciales por su modo de interacción, como en el modelo de interfaz, con nuestro prototipo.

***ReacTable* (2005).**

Autor: *Sergi Jordà, Marcos Alonso, Günter Geiger y Martin Kaltenbrunner (Universidad Pompeu Fabra).*

Descripción: Instrumento digital colaborativo dotado de una interfaz tangible basada en una mesa, e inspirado en los sintetizadores modulares de los años sesenta. Múltiples usuarios simultáneos comparten el control total del instrumento moviendo y rotando objetos físicos sobre la superficie de una mesa circular luminosa. Manipulando dichos objetos, los cuales representan los componentes clásicos de un sintetizador modular, los usuarios pueden crear tipologías sonoras complejas y dinámicas, mediante generadores, filtros y moduladores, en una clase de sintetizador modular tangible. Destaca entre sus intérpretes, Björk, que toco el instrumento durante su gira de apoyo a su disco Volta.[49]

[49] REACTABLE. Web en Linea. [consulta 12/05/2011]Url: http://www.reactable.com/

Fig. 4 ReacTable

***Monome* (2006).**

Autor: *Pennsylvania Company.*

Descripción: Se trata de una interfaz para ordenador, que puede constar desde 64 hasta 256 botones. La interfaz se caracteriza por ser tanto una interfaz de input como output, la luz de de los botones leds, son un output visual que se comunica con el usuario. A pesar de ser producido de forma irregular en pequeñas cantidades desde su introducción en 2006, Monome ha tenido un impacto significativo en la música electrónica. Artistas destacados que la utilizan son Daedelus, Flying Lotus y Nine Inch Nails.[50]

Fig. 5 Monome

[50] MONOME. Web en Linea. [consulta 12/05/2011] Url: http://monome.org/

***Bubblegum* (2007)**.

Autor: *H.Hesse, A. Mc Diarmid y Rosie Han. (UC Berckely-USA)*

Descripción: Interfaz tangible, que es utilizada como secuenciador para crear drum loops mediante la organización de las bolas de colores sobre una superficie. Genera eventos MIDI y puede ser utilizado como dispositivo de entrada para controlar el hardware de audio y software.[51]

Fig. 6 Bubblegum

[51] BUBBLEGUM. Web en Linea. [consulta 12/05/2011] Url: http://www.instructables.com/community/Tangible-Sequence-Interfaces/

***Air Piano* (2007).**

Autor: *Omer Yosha.*

Descripción: interfaz sonora que se toca en cierta manera parecida a un *Theremin* presenta una interfaz intuitiva y simple. La reacción se produce mediante los gestos de la mano del usuario en un espacio tridimensional. Air Piano se basa en una matriz invisible que se toca sobre su superficie, en donde se encuentran "teclas" y "fader", tiene 24 sensores de teclas y 8 sensores para los faders.[52]

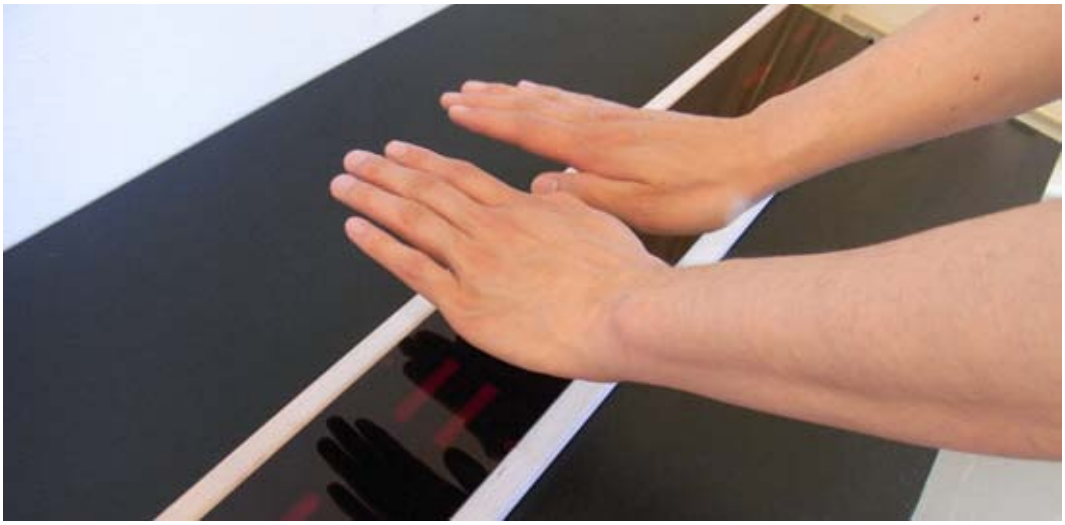

Fig. 7 Air Piano

[52] AIR PIANO. Web en Linea. [consulta 12/05/2011] Url:http://www.airpiano.de/

***Orchestrion* (2010).**

***Autor:** Pat Metheny.*

Interfaz, que mediante un conjunto de instrumentos musicales acústicos y electro-acústicos es capaz de desarrollar mecánicamente una gran variedad de composiciones. Con una guitarra eléctrica, un lápiz y un teclado, la interfaz crea un ambiente detallado de composición o una improvisación espontánea con decenas de instrumentos, que son guiados por una sola persona.[53]

Fig. 8 Orchestrion

[53] ORCHESTRION. Web en Linea. [consulta 12/05/2011] Url:http://www.patmetheny.com/orchestrioninfo/

***Scrapple* (2005).**

Autor: *Golan Levin.*

Descripción: Instalación audiovisual en el que algunos objetos cotidianos se colocan sobre una mesa y una computadora interpreta y utiliza para interactuar con el sonido. Scrapple escanea la superficie de la mesa como si fuera una especie de notación musical, y ejecuta la producción de música en tiempo real mediante los objetos que hay encima de ella. La instalación hace uso de una variedad de formas lúdicas, en particular, largas curvas flexibles que permiten la creación de melodías variables, mientras que un conjunto de formas de tela, objetos pequeños y de cuerda producen los ritmos deseados.[54]

Fig. 9 Scrapple

[54] SCRAPPLE. Web en Linea. [consulta 12/05/2011] Url:http://www.flong.com/projects/scrapple/

***Illusio* (2011).**

Autor: *Peter Kirn*

Descripción: Instrumento musical digital que permite el control en tiempo real de loops grabados a través de actuaciones y colaboración de músicos, la intención de ser lúdico y juguetón. Desarrollado en Processing y OpenFrameworks , mezcla de tecnologías multitouch con la metáfora de la interacción de pedales de guitarra, con una mesa multitouch FTIR DIY - construido con tubos de PVC, cintas... - también integra un teclado modificado como pedal, con un enfoque high-end / low tech.[55]

Fig. 10 Illusio

[55] ILLUSIO. Web en Linea. [consulta 13/05/2011] Url:http://createdigitalmusic.com/2011/06/loops-as-sketches-of-guitar-pedals-in-multitouch-table-music-design/#more-19738

II.MARCO PRÁCTICO

1. Descripción del proyecto "*Fiametta 2.0.*"

Hemos decidido llamar a la interface que vamos a construir y al proyecto "*Fiametta 2.0."*, haciendo alusión al ballet que lleva el mismo nombre, el cual, fue la pieza que *Leon Theremin* interpretó en 1920 en el primer concierto público que ejecutó con un *Etereophone* (más tarde llamado Theremin), el Theremin es un instrumento eléctrico concebido para una interacción física y visual fluida, tanto por parte del músico como del espectador, y es por tanto, un importante referente en mi proyecto.

1.a. Lenguajes e Interacción

En primer lugar, la interface a realizar será ideada para ejecutar la función de secuenciación de audio, aunque más tarde, debido a su naturaleza digital, la interface, una vez acabada, no solo podrá utilizarse como un secuenciador, sino que podrá servir, tanto para la secuenciación de audio, como para generar, distorsionar o variar el sonido, todo dependerá de los códigos y patches que realice y dirijan el trabajo de la interface.

Lo primero que haré, será idear la parte tangible de la interface, la cual, en principio consistirá en un panel, en donde se colocarán objetos en su superficie. Los objetos serán una parte esencial en la comunicación entre el usuario y ordenador, ya que al tratarse de una interface TUI (Tangible User Interface), tiene un punto de contacto, donde el lenguaje entendido por humanos (en este caso un lenguaje visual y táctil), se cruza con el lenguaje binario que está conformado por los impulsos eléctricos que utiliza el ordenador como señales. Por ejemplo, en el caso de un teclado, las teclas, serían este punto de contacto y en el caso de *Fiametta 2.0.*, el punto de contacto, se trata de los objetos que se ponen en su

superficie, el usuario interaccionará con *Fiametta 2.0.* mediante estos objetos, que serán los que envíen la información analógica necesaria que luego es recogida por los sensores. Finalmente arduino transforma estos valores analógicos en señales digitales que se comunican con el ordenador.

La interacción con la interface es una interacción multi-sensorial, una cadena de sucesos que ocurre de la siguiente forma:

1. En primer lugar el usuario visualiza los objetos y donde están colocados, esto, le permite poder ofrecer una respuesta y tomar una decisión de donde colocarlos.

2. En segundo lugar comienza una interacción táctil y visual por parte del usuario, en la que coloca los objetos en la superficie de *Fiametta 2.0.*, los datos son recogidos por los sensores y luego convertidos en señales digitales,

3. En Tercer lugar, estas señales digitales, interaccionarán con el software de *Fiametta 2.0.*, y en función de ellas, se generarán señales analógicas sonoras,

4. En último lugar, las señales analógicas sonoras son vueltas a percibir por el usuario, y este a su vez, responderá en función a estas.

Básicamente se trata de un intercambio de preguntas y respuestas por parte del usuario y el ordenador.

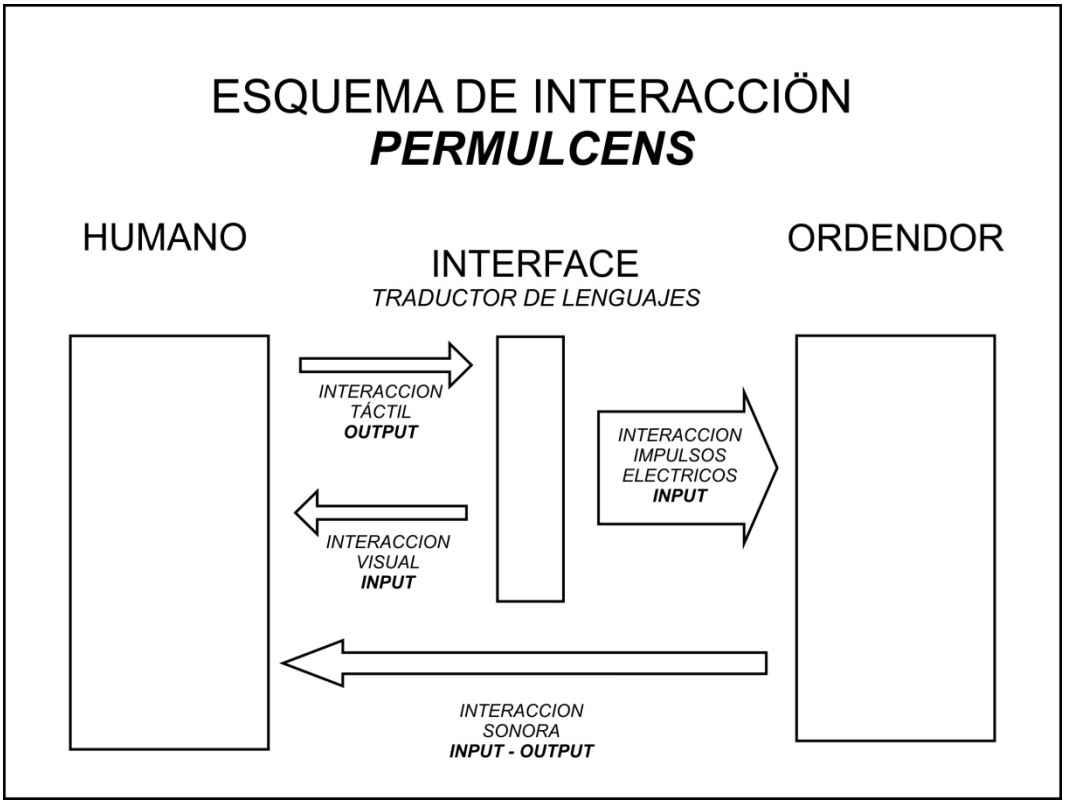

Fig. 11 Esquema de Interacción. *Fiametta 2.0.*

1.b. Técnicas de captación de datos

La segunda pregunta técnica en base a la realización de la interface es: ¿qué tipo de técnicas o sensores voy a utilizar? Contemplo dos posibilidades:

- Sensores de luz: resistencias de luz *LDRs*.
- Cámara de video: técnica de *video-tracking*.

Con motivo del taller de *David Cuartilles* y *Manuel Mazza*, decido preguntarles a ellos que opinan sobre esta cuestión. En dicho taller, entre otras cosas, pongo sobre la mesa que se trata de una interface que ha de ser portable, ya que tiene como objetivo la realización de conciertos o actos performáticos en diferentes lugares. Fue justo esta cuestión la que

terminó por resolver la duda, ya que las instalaciones de videotracking son muy sensibles a los cambios de luz, por lo que no es recomendable realizarla de esta manera, así que finalmente *Fiametta 2.0.* estará realizada con resistencias LDRs.

Una resistencia LDR (light dependent resistor) es un componente electrónico cuya resistencia disminuye con el aumento de intensidad de luz incidente. Su cuerpo está formado por una célula o celda y dos patillas. El valor de resistencia eléctrica de un LDR es bajo cuando hay luz incidiendo en él (puede descender hasta 50 ohms) y muy alto cuando está a oscuras (varios megaohmios). La principal ventaja de este tipo de resistencias es su bajo coste.

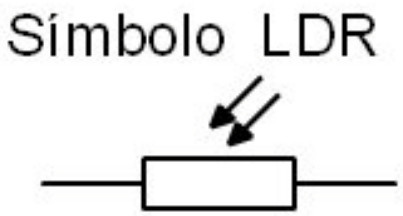

Fig. 12 Simbolo LDR

1.c. Esquematización rítmica

Fiametta 2.0., por su naturaleza digital, será una interface que con un simple cambio en el código base, podrá servir para diferentes tareas, pero en un principio estará orientada hacia la secuenciación de audio en forma de loops, es por esta razón, que he considerado que tomar un modelo rítmico de 4/4, por ser un modelo muy utilizable y típico en la secuenciación electrónica de audio.

Como medida básica de tiempo utilizaré el *beat*, que es una señal transitoria musical y periódica que marca la velocidad del ritmo, es decir, el latido de la música y también es utilizada para comparar la duración de las notas y los silencios. El beat es una unidad temporal básica (aun así sub-divisible) en un *track* de música electrónica. Los *beats* estarán a su vez subdivididos por otras unidades que llamaré *golpes*.

Con un total de 2 *golpes* por 1 *beat* nos quedará una secuencia de 8 *golpes* y 4 *beats* y teniendo en cuenta que tomaré 4 secuenciaciones independientes nos quedará un total de 32 *golpes*, cada golpe necesitará 1 *LDR* independiente, es decir necesitaré un total de 32 ldrs.

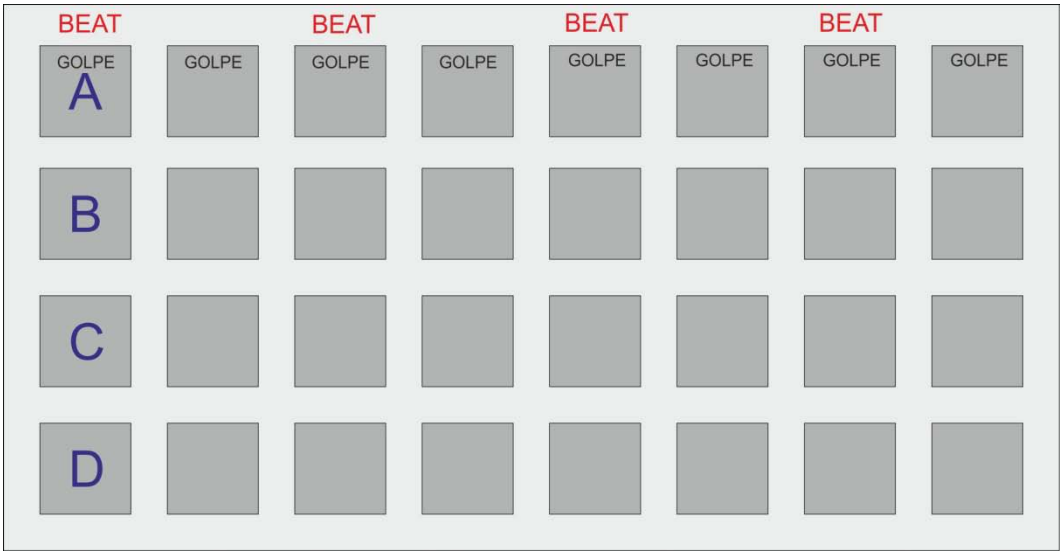

Fig. 13 Esquema Rítmico

2. Desarrollo de *Fiametta 2.0.*

2.a. Primera Fase: Construcción de circuito electrónico básico de 4-8 LDRs.

Lo que ahora debo de hacer, es realizar en una *protoboard*, un circuito más sencillo, que el circuito final de *Fiametta 2.0.*Será un circuito de entrada de los datos, capaz de obtener valores de 8 *LDRs*. He elegido el numero de 8 *LDRs* por ser múltiplo divisible del número final de 36 *LDRs* que conformarán *Fiametta 2.0.*De esta manera puedo ir probando los circuitos y software de manera más básica e independiente y así asegurando paso a paso que todo está montado correctamente.

Para captar los datos de los 32 *LDRs* necesitaríamos 32 entradas analógicas de datos. *Arduino Duemilanove*, dispone de 6 pines analógicos de entrada de datos. La versión más grande de arduino, *arduino Mega* dispone solo de 15 entradas analógicas, así que tendré que utilizar un método alternativo para captar los datos de los *LDRs*. La opción más sencilla es disponer de un chip multiplexor, para así poder disponer, con pocos pines de entradas analógicas, muchos datos entrantes de los *LDRs*.

Un multiplexor es un circuito combinacional con *X* líneas de entrada de datos, *Y* líneas de salidas y *Z* entradas de selección. Que funciona como puertas que dejan, o no, pasar la información entrante. Las entradas de selección indican cuál de estas líneas de entrada de datos es la que proporciona el valor a la línea de salida. Por lo que es posible secuenciarlas y obtener los datos de por ejemplo 8 LDRs, por solo una entrada analógica de arduino, en un margen corto de tiempo, por ejemplo en 1 milisegundo cada una.

El modelo de multiplexor que voy a utilizar será, el conocido como *CD4052BC*, ya que es el multiplexor que más se adecúa al circuito que quiero montar.

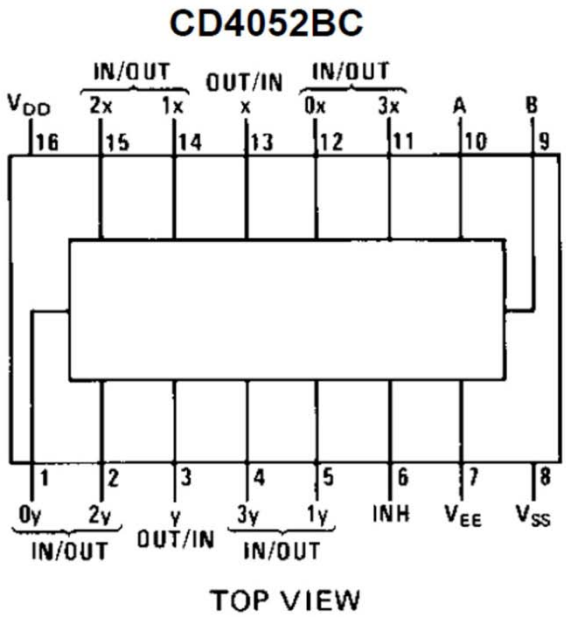

Fig. 14 Esquema CD4052BC Multiplexer

El *CD4052BC* cuenta como se ve en el gráfico, con:

- 8 Pines de entrada de datos
- 2 Pines de salida de datos
- 2 Pines de selección de puertas

Los Pines A y B son pines digitales de selección de puerta, es decir que tienen dos posiciones: (HIGH, LOW), o lo que es lo mismo con electricidad o sin ella. Desde arduino tendríamos el control de estos pines de selección mediante los pines digitales, entonces el código de arduino seleccionaría los pines que deseo leer en ese momento. La tabla de selección sería esta:

INPUT STATES		"ON" CHANNELS
B	A	CD4052B
0	0	0X, 0Y
0	1	1X, 1Y
1	0	2X, 2Y
1	1	3X, 3Y

A continuación montaré el circuito en la protoboard, primero realizo un esquema del mismo. El circuito más básico que voy a hacer, para ir paso por paso, es un circuito formado por:

- 1 Multiplexor CD4052BC
- 8 LDRs
- 8 Resistencias 1600 Ohmios
- Arduino

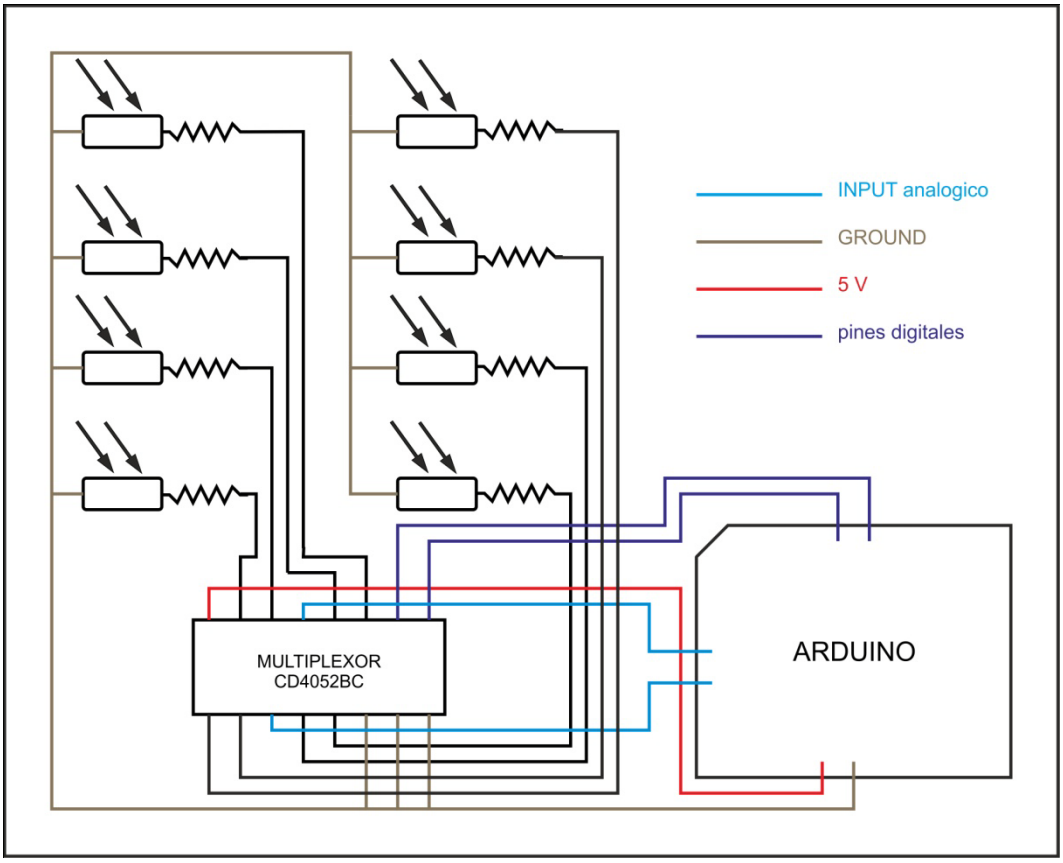

Fig. 15 Esquema CIrcuito 8 LDRs

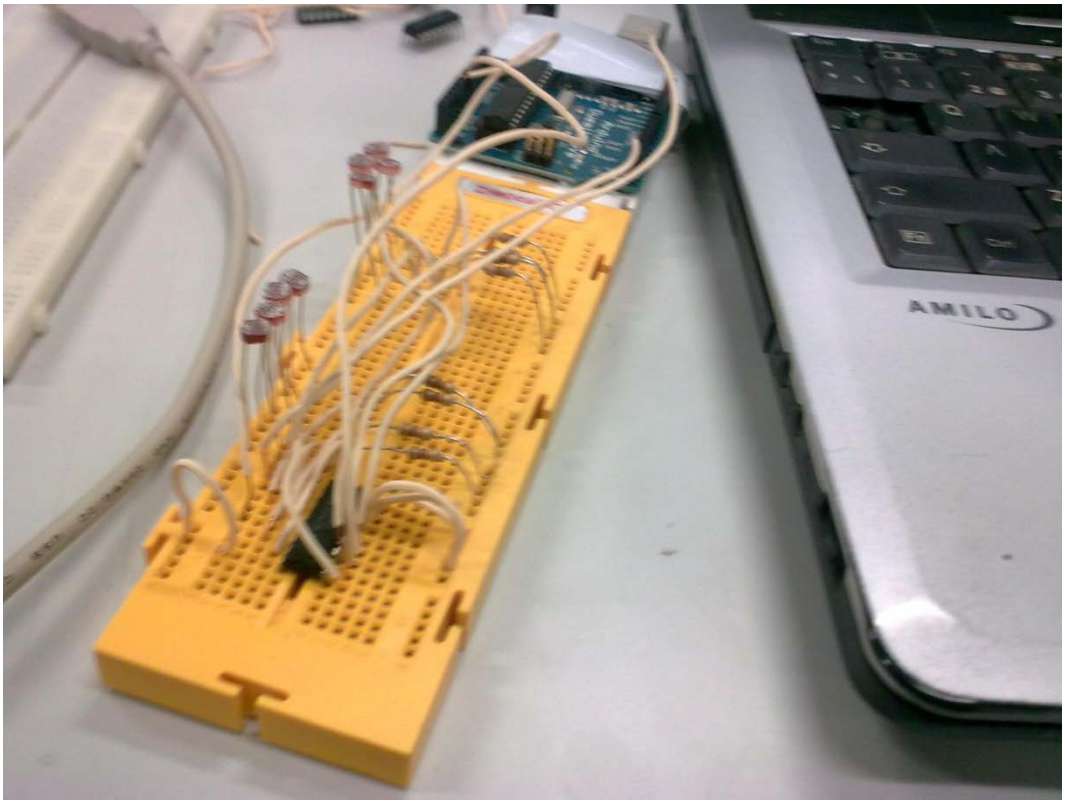

Fig. 16 Foto circuito 8 LDRs

2.b. Segunda Fase: Primer desarrollo del software *Fiametta 2.0.*:

Una vez realizado la parte tangible del circuito, ahora tengo que hacer el código que recoja los datos de los LDRs, los almacene en tiempo real y luego los utilicé para interactuar con el audio de salida. En esta fase probé diferentes plataformas que describo a continuación.

a. Experiencia A. Software Arduino.

i. Descripción de Arduino.

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Se creó para artistas, diseñadores, aficionados y cualquiera interesado en crear entornos u objetos interactivos.

Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar aquello que le rodea controlando luces, motores y otros actuadores. El micro-controlador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectar a un ordenador, si bien tienen la posibilidad de hacerlo y comunicar con diferentes tipos de software (p.ej. Flash, Processing, MaxMSP). [56]

[56] ARDUINO. Web en linea [16/01/2010] Url:http://www.arduino.cc/es/

ii. Descripción del código.

Paso 1. En primer lugar tengo que configurar dos pines digitales de arduino para que hagan de selectores de las puertas de entrada de datos del multiplexor. Una vez configurados, les envío las órdenes de cambio, a cada uno de los pines y hago una lectura de los pines analógicos conectados a arduino. Este proceso se repetirá cuatro veces, porque en cada lectura se leen dos pines analógicos, dando un total de 8 LDRs.

Paso 2. Envío los datos al puerto serie en forma decimal (DEC), para poder leerlos por la consola de arduino, y hacer la comprobación de que todo está bien. En principio utilizaré grandes *delays* entre una y otra lectura para poder asegurar que las lecturas de datos son coherentes y correctas. En el código final, enviare las lecturas en forma de byte (BYTE) en vez de forma decimal (DEC) para que otro programa o entorno pueda leerlas, y además reduciré los tiempos de *delays* al mínimo.

Un byte es una unidad de información. Con esta unidad se puede desde el almacenamiento de datos hasta la capacidad de memoria de una computadora. Representa un carácter, y está constituido por 8 bits consecutivos, de modo tal que un byte, generalmente, equivaldría a 8 bits. Las entradas analógicas de arduino tienen una capacidad de 10 bits por lectura, esto quiere decir que son que tiene 1024 posibles combinaciones, por lo que si se quiere pasar la información al ordenador, ha de ser enviada segmentada, en varios bytes.

Paso 3. El primer problema surge a la hora de enviar la información al puerto serie, ya que la lectura de entrada de datos analógico tiene 10 bits, y sin embargo los bytes que se pueden a través del puerto serie son, lógicamente, de 8 bits. Contemplo la posibilidad de dividir la lectura de datos de 10 bits en dos y enviarlo en dos bytes de 8 bits, para después volverlo a montar con un algoritmo, pero, en principio, decido no tomar

este camino porque no voy a necesitar un margen tan exacto y pequeño de datos. Así que lo que hago es dividir por cuatro el byte de 10 bits y queda como si fuera uno de 8 bits, por ejemplo el byte 1024 quedaría en 256 o el byte 848 quedaría en 212... Nunca dará decimal porque utilizo números enteros INT.

iii. Código.

```
int x0,x1,x2,x3,y0,y1,y2,y3,xx3,xx2,xx1,xx0,yy0,yy1,yy2,yy3;
int a,b;
void setup(){
        a=8;
        b=9;
        pinMode (a, OUTPUT);
        pinMode (b, OUTPUT);

  Serial.begin(57600);
}
void loop(){
         digitalWrite (a,HIGH);
         digitalWrite (b,HIGH);
                delay(2);
         x3 = analogRead(1);
         y3 = analogRead(2);
         xx3 = x3/4;
         yy3 = y3/4;
         Serial.println(xx3, BYTE);
         Serial.println(yy3, BYTE);
                delay(2);

         digitalWrite (a,LOW);
```

```
        digitalWrite (b,HIGH);
                delay(2);
        x2 =analogRead(1);
        y2 = analogRead(2);
        xx2 = x2/4;
        yy2 = y2/4;
        Serial.println(xx2,BYTE);
        Serial.println(yy2, BYTE);
                delay(2);

        digitalWrite (a,HIGH);
        digitalWrite (b,LOW);
                delay(2);
        x1 =analogRead(1);
y1 = analogRead(2);
xx1 = x1/4;
yy1 = y1/4;
Serial.println(xx1, BYTE);
Serial.println(yy1, BYTE);
       delay(2);


digitalWrite (a,LOW);
digitalWrite (b,LOW);
       delay(2);
x0 =analogRead(1);
y0 = analogRead(2);
xx0 = x0/4;
yy0 = y0/4;
Serial.println(xx0, BYTE);
Serial.println(yy0, BYTE);
       delay(2);
```

}

iv. Medición y Resolución de problemas

Ahora voy a hacer la prueba de que los datos entrantes son correctos y están en orden, abriré la consola que incorpora el entorno de arduino y comprobaré que están saliendo los datos de acuerdo con su orden (para ello los estoy enviando en forma decimal).

Utilizaré un método muy sencillo para esta comprobación: tapare uno de los *LDRs* de manera que cada 8 valores de datos, uno de los valores tiene que ser muy bajo, y lo haré con cada uno de ellos. Finalmente compruebo que los datos entrantes son correctos. Estos son los datos en una de las pruebas:

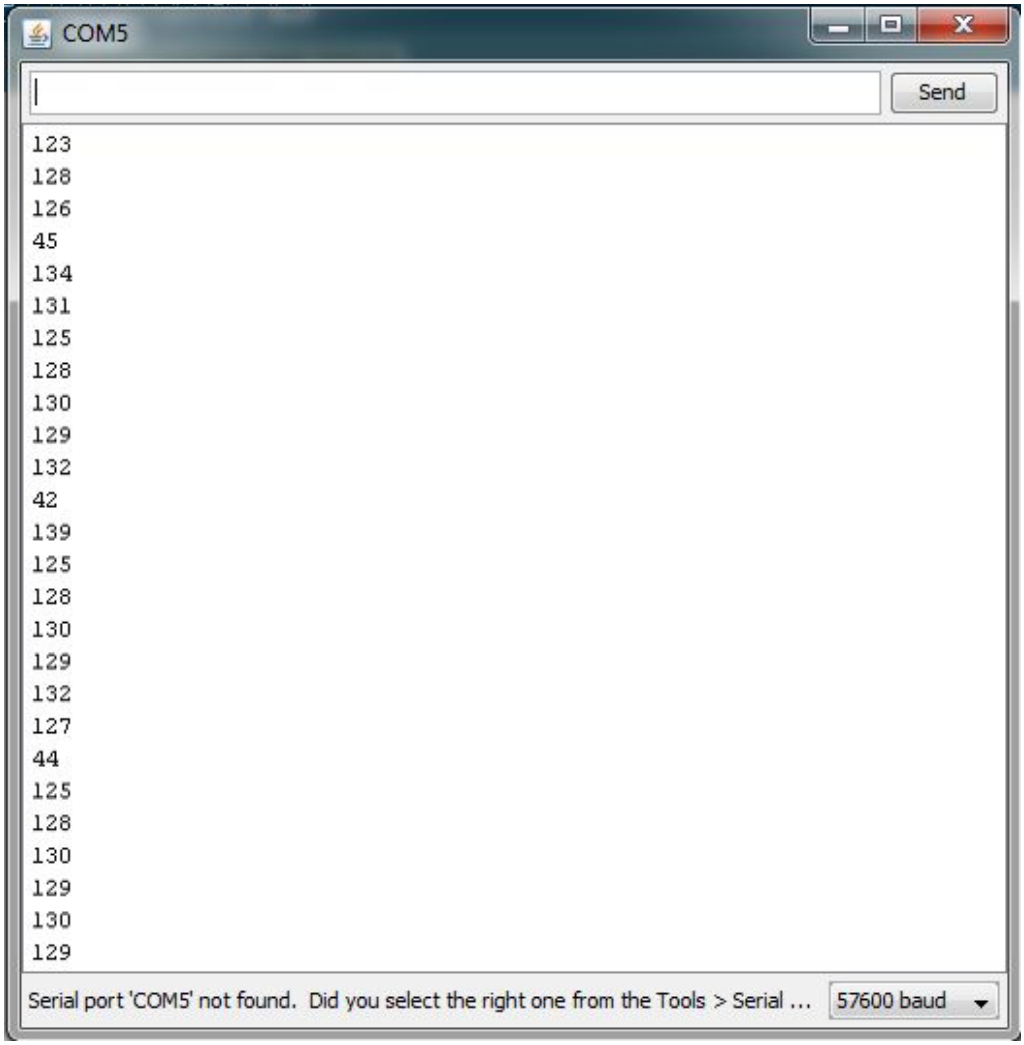

Fig. 17 Resultados Prueba 1

v. Conclusión

Como se puede comprobar, los datos están correctamente leídos y en orden, ya que cada 8 valores uno de ellos es bajo, debido a que un *LDR* está tapado.

A continuación he de elegir la plataforma o entorno que recibirá los datos para convertirlos en interacciones sonoras.

b. Experiencia B. Software openFrameworks.

El siguiente paso es elegir que programa o entorno que va a recibir los datos por el puerto serie, para luego tomar los datos y ejecutar procedimientos. En un principio pienso en probar con *processing*, ya que tiene una buena comunicación predefinida con arduino. Aunque prefiero contemplar otra posibilidad, la de utilizar un entorno de programación que justo en estos momentos estoy aprendiendo en el taller de *David Cuartilles y Manuel Mazza*, openFrameworks. Al final decido utilizar este último por ser más completo en cuanto a comandos y posibilidades con el sonido.

i. Descripción de openFrameworks

OpenFrameworks es una colección de herramientas basado en el entorno de C++, y está diseñado para ayudar al proceso creativo, proporcionando un marco sencillo e intuitivo para la experimentación.

La biblioteca está diseñada para funcionar como una herramienta multiusos de uso general, y envuelve a varios entornos utilizados comúnmente en las bibliotecas de interface abierto, entre ellos: OpenGL para gráficos, RTAudio para la entrada y salida de audio, freetype para las fuentes, FreeImage de imagen de entrada y salida, quicktime para la reproducción de vídeo[57].

OpenFrameworks además cuenta con la posibilidad de trabajar con Arduino directamente desde su código, utilizando el software de *Firmata*. Este software, también libre, es un código de arduino que se instala en el propio arduino para que este lo ejecute, y entonces, desde otro código

[57] OPENFRAMEWORKS. Web en linea [18/04/2010] Url:http://www.openframeworks.cc/about

creado en OF, se puede controlar arduino a través de sencillos comandos predefinidos.

ii. Descripción del código

Me dispuse a crear un código en *OF* que controlara a arduino y leyera los diferentes datos de los *LDRs*. Básicamente el código de *OF* debía de hacer lo mismo que el código creado en arduino, es decir, tenía que enviar HIGHs y LOWs a los pines digitales que seleccionan las puertas de entrada de datos, para luego hacer lecturas de los pines analógicos a los que el multiplexor está conectado.

Lo primero entonces fue introducir el software Firmata en Arduino y a partir de esto realizar el código en OF que controle arduino.

iii. Código.

```
void testApp::setup(){
        ofSetVerticalSync(true);
        ofSetFrameRate(60);
        ofBackground(255,0,130);
        bgImage.loadImage("firmata.png");
        font.loadFont("franklinGothic.otf", 20);
                // ard.connect("/dev/ttyUSB0", 115200);
                ard.connect("COM5", 9600);
                bSetupArduino       = false;
         z = 0;
         sensorValue     = new int[3];
}
void testApp::update(){
```

```
	if ( ard.isArduinoReady()){
			if (bSetupArduino == false){
			setupArduino();
			bSetupArduino = true;	// only do this once
			}
			updateArduino();
	}
}
void testApp::setupArduino(){


	ard.sendDigitalPinMode(8, ARD_OUTPUT);
	ard.sendDigitalPinMode(9, ARD_OUTPUT);
	ard.sendAnalogPinReporting(0, ARD_ANALOG);
	ard.sendAnalogPinReporting(1, ARD_ANALOG);
}
void testApp::updateArduino(){

	ard.update();
}
void testApp::draw(){

	if (!ard.isArduinoReady()){
		font.drawString("arduino cargando", 545, 40);
	} else {
	z=0;
	  ard.sendDigital(8, 0);
	  ard.sendDigital(9, 0);
		ofSleepMillis(5);
	  x1 = ard.getAnalog(1);
	  font.drawString("sensor 0: " + ofToString(x1), 545, 40);
```

```
                ofSleepMillis(5);
        z++;
            ard.sendDigital(8, 1);
            ard.sendDigital(9, 0);
                ofSleepMillis(5);
            x2 = ard.getAnalog(1);
            font.drawString("sensor 1: " + ofToString(x2), 545, 60);
                ofSleepMillis(5);
        z++;

            ard.sendDigital(8, 0);
          ard.sendDigital(9, 1);
                ofSleepMillis(5);
          x3 = ard.getAnalog(1);
          font.drawString("sensor 2: " + ofToString(x3), 545, 80);
                ofSleepMillis(5);
    z++;
           ard.sendDigital(8, 1);
           ard.sendDigital(9, 1);
                ofSleepMillis(5);
           x4 = ard.getAnalog(1);
           font.drawString("sensor 3: " + ofToString(x4), 545, 100);
                ofSleepMillis(5);
     z=0;
}
```

iv. Medición y Resolución de problemas

En este momento pude detectar el primer problema. Sabía que el circuito electrónico era correcto, gracias a la prueba que realicé con el software de arduino, sin embargo, el código de openFrameworks no

funcionaba, aprovechando una vez más el taller de David Cuartilles y Manuel Mazza, propuse revisar el código, con la conclusión final de que el código era correcto.

Entonces probé a adoptar soluciones, y realizar un código más básico. Como primera solución decidí aumentar el tiempo de delay por operación, para comprobar si funcionaba con mas retardo, es decir, aumente (ofSleepMillis()) de 5 milisegundos a 500 milisegundos por operación de escritura y lectura, con ello aumente la separación temporal entre uno y otro proceso de arduino. Con la conclusión de que sí funcionaba, con los tiempos de delays mas largos. EL siguiente paso fue reducir el tiempo de los delays, y lo máximo que conseguí reducir funcionando, fue de 200 milisegundos por operación.

Como conclusión determiné que era un tiempo imposible de contemplar en una interface que pretende utilizarse para el directo en escena.

EL siguiente paso fue buscar en foros algo sobre el tema de *Firmata* y sus posibles fallos, y entonces, pude comprobar cómo mucha gente, había tenido problemas parecidos con Firmata.

v. Conclusión

Así que concluyo el aprendizaje de la experiencia B, en que firmata, utilizado desde openFrameworks, solo sirve para operaciones básicas que no conlleven cierta complejidad de escritura y lectura simultaneas en arduino, esto es evidente, por lo menos hasta la fecha de marzo del 2010 con la versión 2.1 de Firmata y 0.62 de openFrameworks.

c. Experiencia C. Software Pure Data

Decidí pasarme a Pure Data por varias razones, entre ellas porque comencé a aprenderlo, en el Master de Artes Visuales y Multimedia, justo en el momento que tuve problemas con openFrameworks y Firmata, pero la más importante para decidirme por Pure Data fue que es un entorno de programación especialmente ideado para audio digital.

i. Descripción de Pure Data.

Pure Data (o Pd) es un lenguaje de programación gráfico desarrollado por *Miller Puckette* durante los años 90 para la creación de música computarizada interactiva y obras multimedia. Aunque Puckette es el principal autor del software, Pd es un proyecto de código abierto y tiene una gran base de desarrolladores trabajando en nuevas extensiones al programa.

Pure Data se basa como muchos otros lenguajes en la programación por diagrama de flujo de datos, un diagrama de flujo de datos (DFD por sus siglas en español e inglés) es una representación gráfica, en donde los datos circulan en torno a un diagrama de flujo de datos que a su vez está basado en un sistema de información. Un diagrama de flujo de datos también se puede utilizar para la visualización de procesamiento de datos (diseño estructurado). Estos diagramas utilizan símbolos con significados bien definidos que representan los pasos del algoritmo, y representan el flujo de ejecución mediante flechas que conectan los puntos de inicio y de término. Este modelo de programación es utilizado entre otros por Max Msp, Pure Data o LabView.

. Pure Data también puede utilizar el software de Firmata para controlar Arduino desde un patch realizado en Pure Data, con una aplicación predefinida denominada *Pduino*.

ii. Descripción del patch 1.

Yo nunca había utilizado un lenguaje con diagrama de flujo de datos, así que sabía que el proceso de desarrollo sería lento.

Paso 1. El patch que realicé programaba dos pines de arduino como outputs que serían los selectores de puertas del multiplexor, así mismo, con dos metrónomos, uno con la mitad del tiempo que el otro, conseguía que los valores de 0 y 1 pasaran por todas sus combinaciones posibles es decir 00 10 01 11 para así dar las lecturas de todos los LDRs. Así mismo en un principio puse el metrónomo con un amplio margen a 100 y 200 para evitar posibles problemas con la comunicación de Firmata y arduino.

Paso 2. He decidido que para simplificar al máximo posible el patch solo haré las lecturas de 4 LDRs, en vez de los 8 que tengo en el circuito. En principio lo que haré después de ir obteniendo los valores de los LDRs será almacenarlos en una lista para poder luego llamarlos cuando los vaya necesitando. La lista tendrá que ir suplantando los antiguos valores por los nuevos que se obtienen que una vez cambiados serán innecesarios, esto no es solo una opción, sino que es necesario por la cantidad de valores que estaremos manejando.

Paso 3. Después de experimentar y probar algunos objetos para almacenar los valores de los LDRs como *ARRAY* y *ROUTE*, sin buenos resultados, me centro en el objeto *LIST*, más concretamente en el objeto *LIST PREPEND* con el que puedo almacenar los datos de los LDRs y

además me permite la posibilidad de resetear la lista, borrar los datos antiguos, y comenzar a escribir otros nuevos. Entonces coloco un metrónomo de 400 ms para que reseteé la lista, lo coloco de 400 ms porque es justo una vuelta completa de los valores que leen un LDR cada 100 ms, además le añado, a este metrónomo, un delay de 30 ms para que no quede muy ajustada la lectura del LDRs, antes de pasar al siguiente.

iii. Patch 1.

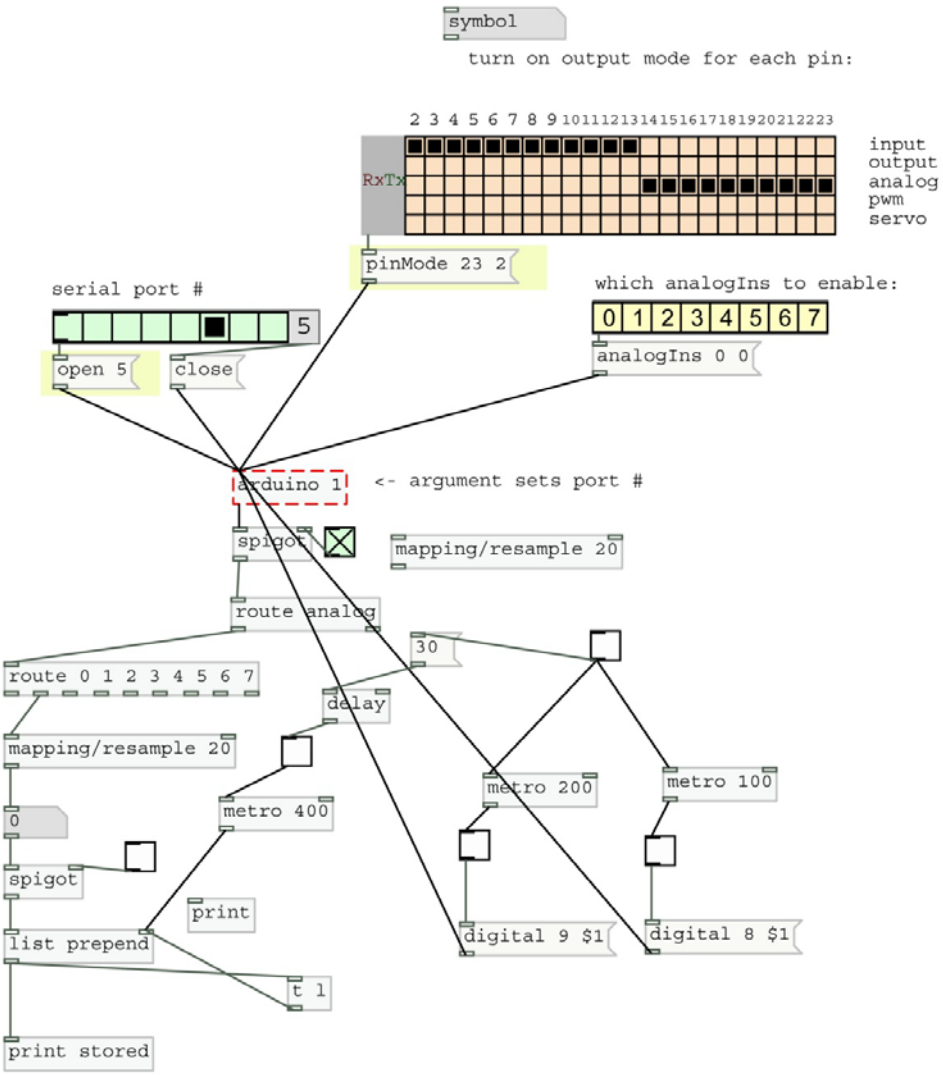

Fig. 18 Patch 1

iv. Medición y Resolución de problemas: Patch 1.

Prueba 1. Comienzo a ejecutar la prueba que muestra los datos almacenados en la lista y que se imprime en la consola de pure data. Los datos de la prueba son datos correctos, ya que se repiten en cuatro intervalos y luego se resetean.

stored: 0.414467
stored: 0.414467 0.385142
stored: 0.414467 0.385142 0.058651
stored: 0.414467 0.385142 0.058651 0.44086
stored: 0.41349
stored: 0.41349 0.385142
stored: 0.41349 0.385142 0.058651
stored: 0.41349 0.385142 0.058651 0.44086
stored: 0.41349
stored: 0.41349 0.386119
stored: 0.41349 0.386119 0.0596285
stored: 0.41349 0.386119 0.0596285 0.44086
stored: 0.41349
stored: 0.41349 0.385142
stored: 0.41349 0.385142 0.058651
stored: 0.41349 0.385142 0.058651 0.44086
stored: 0.41349
stored: 0.41349 0.386119
stored: 0.41349 0.386119 0.058651
stored: 0.41349 0.386119 0.058651 0.44086

Solución 1. Mi objetivo, una vez más, es bajar al máximo el tiempo de los Delays y los metrónomos, para dar la máxima agilidad a una interface que está pensada para el directo. Concluyo, mediante pruebas, que el mínimo tiempo de lectura posible por LDR es de 40 ms, por lo que

determino que es imposible utilizar este patch, ya que si contamos con los 32 LDRs que utilizaré cuando la interface esté acabada, serían un total de 1,28 segundos, algo que en un directo es impensable proponer.

La cuestión sería entender porque no puedo bajar de este tiempo mínimo, encontrar el problema y atajarlo.

Prueba 2. Lo primero que se me ha ocurrido hacer es doblar el número de lecturas y poner a la mitad de tiempo los delays y los metrónomos, al pasar esto, podré analizar con más exactitud cuál de las lecturas está fallando. La prueba dio los siguientes resultados:

stored: 0.00782014
stored: 0.00782014 0.00782014
stored: 0.00782014 0.00782014 0.146628
stored: 0.00782014 0.00782014 0.146628 0.146628
stored: 0.00782014 0.00782014 0.146628 0.146628 0.125122
stored: 0.00782014 0.00782014 0.146628 0.146628 0.125122 0.173998
stored: 0.00782014 0.00782014 0.146628 0.146628 0.125122 0.173998 0.173998
stored: 0.00782014 0.00782014 0.146628 0.146628 0.125122 0.173998 0.173998 0.173998
stored: 0.00879765
stored: 0.00879765 0.148583
stored: 0.00879765 0.148583 0.148583
stored: 0.00879765 0.148583 0.148583 0.148583
stored: 0.00879765 0.148583 0.148583 0.148583 0.124145
stored: 0.00879765 0.148583 0.148583 0.148583 0.124145 0.173998
stored: 0.00879765 0.148583 0.148583 0.148583 0.124145 0.173998 0.173998

stored: 0.00879765 0.148583 0.148583 0.148583 0.124145 0.173998 0.173998 0.173998

Se puede comprobar cómo hay lecturas que fallan siempre, lo que tendría que haber ocurrido es que los números se repitieran de dos a dos hasta el final, y sin embargo a veces se leen tres iguales.

v. Descripción Patch 2.

He decidido cambiar la forma del patch a ver si puedo resolver el problema con otro método de programación. Voy a utilizar el objeto *SPIGOT* como puerta de datos y así sincronizarlo con metrónomos y delays para que los datos pasen de manera selectiva. También voy a cambiar el método de la Lista, es decir voy a dejar de utilizar LIST PREPEND y voy a pasar los datos directamente a objetos de números, utilizando el comando de FLOAT, tal como se ve en el patch.

vi. Patch 2.

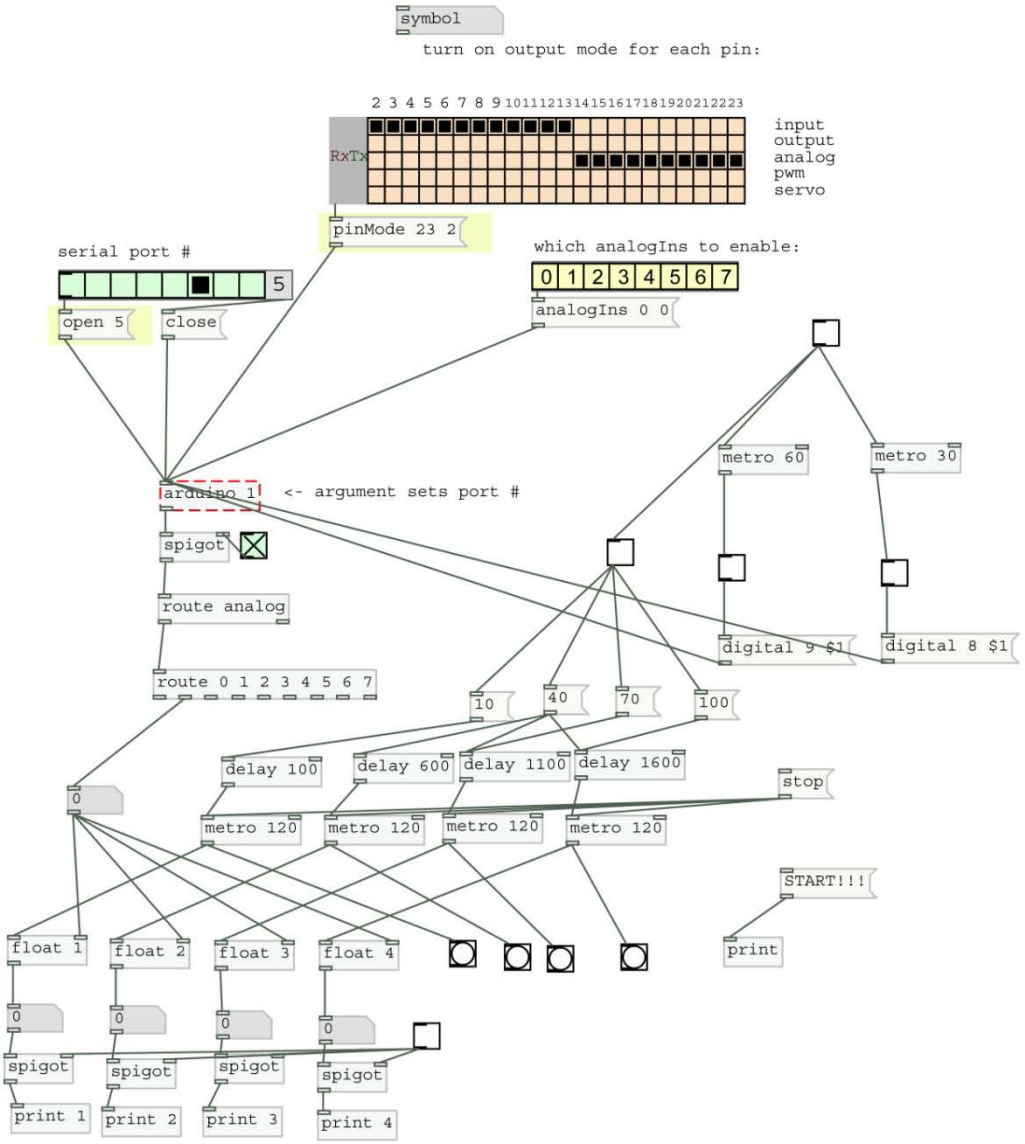

Fig. 19 Patch 2

vii. Medición y Resolución de problemas: Patch 2

Prueba 3. Ejecuto la prueba de las lecturas de datos, una vez más, con resultados insatisfactorios, ya que las lecturas son aun más dispares que con el anterior método.

viii. Descripción Patch 3.

Decidido cambiar por completo el patch y el método de programación. Voy a intentar programar el software con el componente de pure data COMPORT, ya que creo que la raíz del problema no está en el patch sino en el software de Firmata, con el que ya tuve problemas en openFrameworks.

El nuevo método es sencillo, utilizare el patch de arduino que había realizado en la Experiencia A de *Fiametta 2.0.* y que elaboré antes de empezar con openFrameworks y pure data, este patch enviará, al puerto serie, todos los datos de las lecturas, uno detrás de otro y luego con el objeto COMPORT, de pure data, recogeremos esos datos del puerto serie.

El nuevo patch funciona de la siguiente forma, el objeto COMPORT recibe los datos del puerto serie, una puerta SPIGOT permite el paso de los datos si esta se encuentra abierta, la puerta SPIGOT lanza por su outlet, a la misma vez, un BANG y el numero recogido por COMPORT del puerto serie (es decir el de los valores de los LDRs). El BANG va a parar a un contador que está configurado para contar 8 veces y recomenzar la cuenta al llegar a 8, que es entonces cuando el valor recogido por COMPORT y el valor de cuenta se empaquetan con el elemento PACK, para luego desempaquetarse, por orden, con el elemento ROUTE, que

permite que los paquetes se ordenen por el numero o carácter por el que empiezan.

ix. Patch 3.

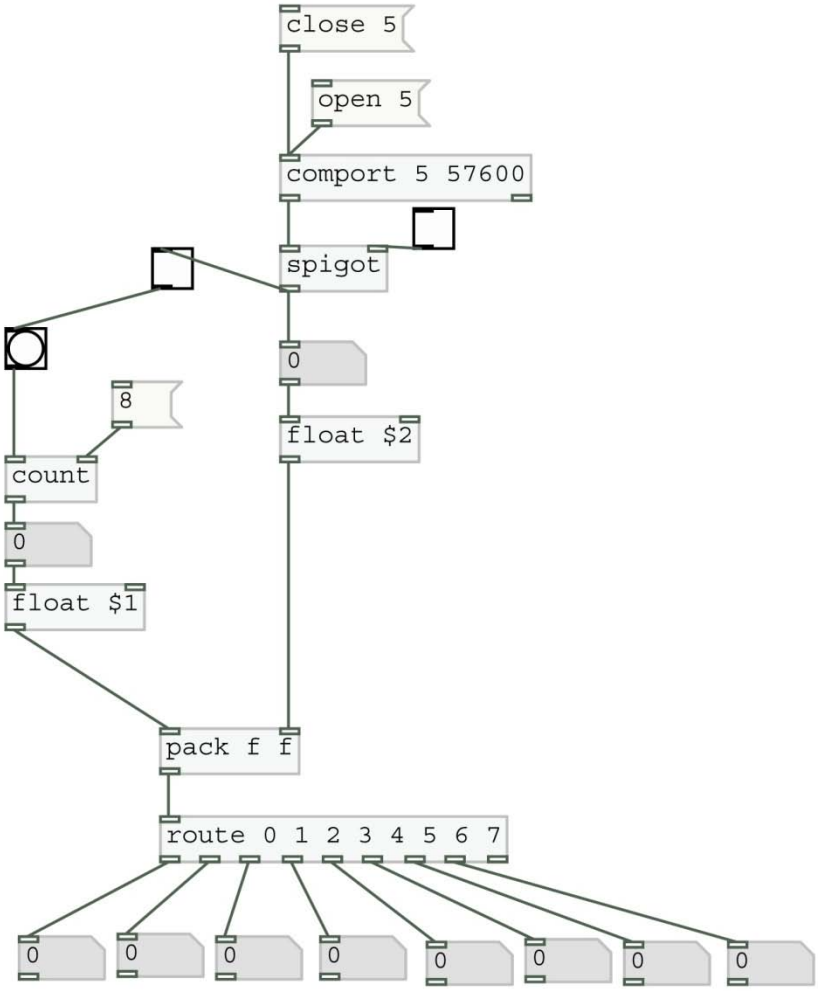

Fig. 20 Patch 3

x. Conclusión.

Como se puede comprobar es un código mucho más sencillo que el utilizado con Pduino, y las pruebas resultan ser satisfactorias, todos LDRs enviaron el valor que les corresponde, mediante la prueba de tapar

uno a uno cada LDR. Después de afinar el patch bajando el tiempo de delays al mínimo, que finalmente consigo bajarlo a 2 milisegundos cada LDR, por lo que completo el ciclo de 8 LDRs en 16 milisegundos, y cuando lo tenga listo con sus 32 LDRs serán 64 milisegundos una vuelta completa, una latencia totalmente asumible para la interfaz de directo que quiero realizar.

2.c. Tercera Fase: Segunda construcción del hardware *Fiametta 2.0.*

a. LDRs. Ley de Ohm

Una vez terminado con éxito la primera fase del desarrollo de *Fiametta 2.0.*, me dispongo a hacer pruebas de luz y lectura de datos con los LDRs.

En primer lugar, coloco luces intensas de 200 W cerca de los LDRs, entonces puedo comprobar cómo a pesar de poner estas fuertes luces, las lecturas de los LDRs solo llegan a la mitad de sus valores posibles, esto es de 120 a 140. Existen dos formas de solventar el problema, y las dos formas requieren de la utilización de la Ley de Ohm.

La Ley de Ohm afirma que la corriente que circula por un conductor eléctrico es directamente proporcional a la tensión e inversamente proporcional a la resistencia siempre y cuando su temperatura se mantenga constante. La formula matemática de la ley de Ohm es:

$$I = \frac{V}{R}$$

Intensidad es igual al voltaje partido por su resistencia.

La Resolución del problema puede hacerse de dos formas, para que la intensidad sea diferente, una de dos, o bien vario las resistencias o bien vario el voltaje. Existe una posibilidad de que variando las resistencias, el circuito sea ineficaz, esto es debido a que las resistencias por tratarse de un elemento pasivo, tienen un tope en la capacidad de

variación de la formula, por lo que entonces tendría que variar el voltaje utilizando elementos activos, como por ejemplo transistores.

En segundo lugar, tenemos que las resistencias que estoy utilizando en el circuito es de 1600 Ohm y los fotoresistores de los LDRs varían de 2-20 Kohms, cuando la luz incide sobre ellos y hasta 2 Mohms, cuando están a oscuras. Si el voltaje es constante de 5 V, me da un rango de ~0,5 mA hasta unos pocos 0,0025 mA, este ultimo numero no es significativo, el importante es su máximo, es decir el de 0,5 mA. Si lo que necesito es que aporte unos 1 mA entonces simplemente tengo que encontrar unos LDRs, que tengan un grado de variación en su resistencia algo mayor. Es decir que necesitaré LDRs que su valor de resistencia en ohms cuando incida la luz sobre ellos se encuentre en torno 0,5-10 Kohms.

i. Conclusión

Después de inspeccionar el catálogo de LDRs, me decido por comprar los de modelo GL05AP02. por tener un valor de resistencia máximo de 0,5-10 Kohms.

El problema del balance inefectivo, de los valores tomados por los LDRs, ha quedado solucionado. Los valores que ahora marcan en el patch de Pure-Data son mayores cuando les incide la luz (entre 180 y 210) y prácticamente iguales cuando no les incide la luz (20-40). Así que he conseguido aumentar el espectro del rango.

b. Diseño y realización del circuito de 32 LDRs

i. Descripción del circuito.

Ahora comenzaré con la esquematización del circuito final de 32 LDRs. En primer lugar he de decidir si seguiré utilizando el multiplexor 4052 de 8 canales con 2 salidas, o por el contrario decido utilizar el multiplexor 4051 de 8 canales con 1 salida o quizás considere la opción de montar el circuito con multiplexores de 16 canales. He decidido descartar el multiplexor de 16 canales porque cuesta casi 20 veces más que uno de 8 canales, y por último decido seguir utilizando el multiplexor 4052 por ya tener ya experiencia con el funcionamiento de dicho multiplexor.

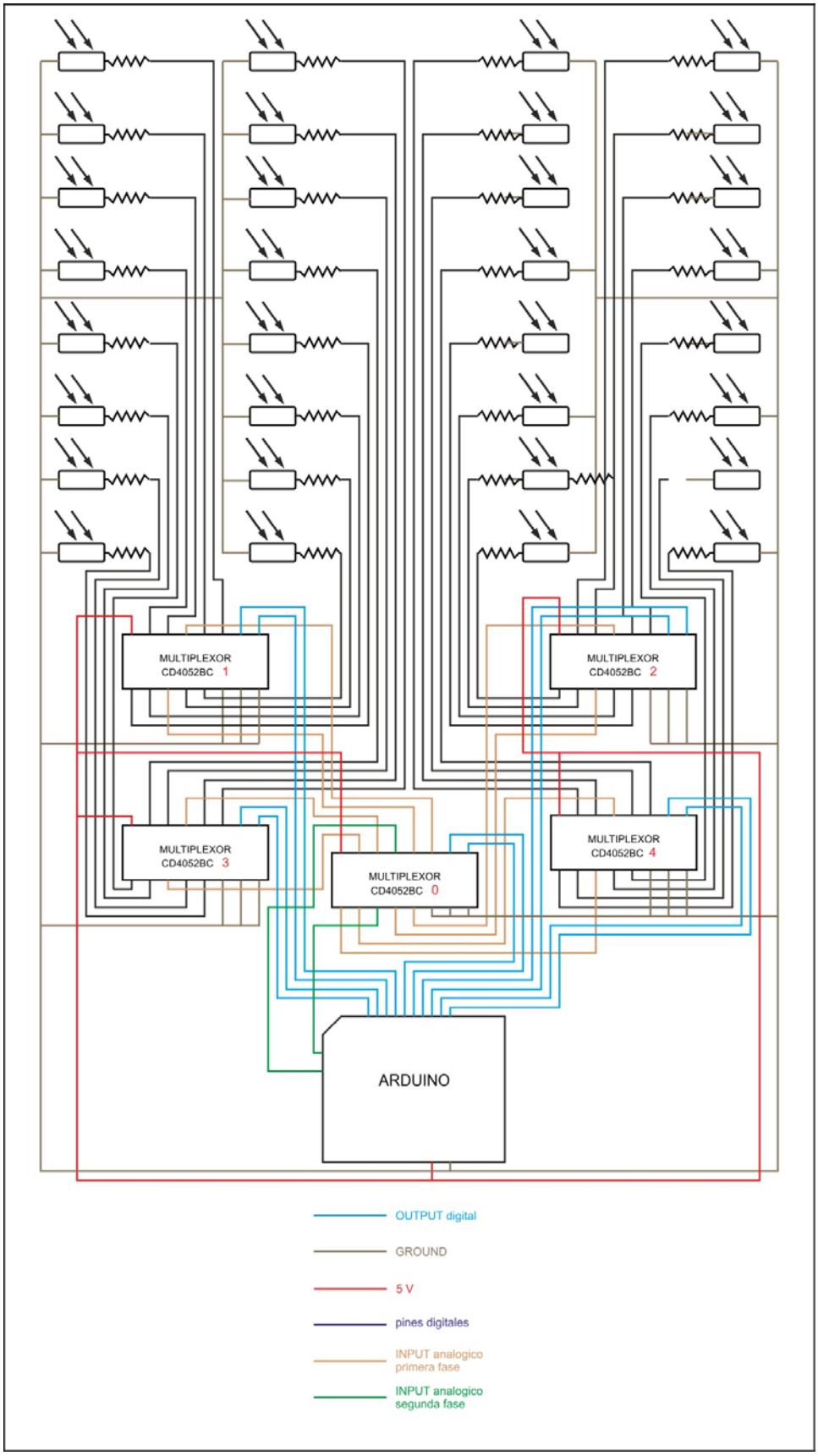

Fig. 21 Esquema Circuito 32 LDRs

c. Implementación tangible del hardware:

i. Descripción del hardware *(box y beats)*.

Fiametta 2.0. tendrá forma de cuboide rectangular, es decir forma de caja y a esta parte la llamaré *box*, por tratarse de una forma muy funcional, que simplificará el entendimiento en el uso de *Fiametta 2.0.*. La box incorporara los 32 LDRs a lo largo de su cara superior, que será la parte más importante porque es en donde se desarrollará todas las interacciones del usuario con *Fiametta 2.0.*

La cara superior de la box estará dividida en 32 partes, una por cada LDR, a las que llamaré beats para ser tratado como unidad básica del desarrollo de las frases musicales electrónicas que se ejecuten. Encima de estas partes irán los objetos y materiales que harán posible la interactuación con *Fiametta 2.0.*El material que utilizaré para hacer la box tendrá que ser transparente, ya que si no fuera así, tendría que agujerear uno por uno, la superficie de la cara superior de la box para poder poner los 32 LDRs. La box será de metacrilato por varios motivos: en primer lugar es un material que se presta mucho a ser manipulado y trabajado, cosa muy a tener en cuenta sobre todo a la hora de solapar las diferentes caras de la box, además de esto, no es un material totalmente rígido como el cristal, por lo que es mucho más difícil de romper si se llevara un golpe.

El tamaño de la box será todavía algo a decidir, empezaré trabajando solo con la cara superior de la box, para poder implementar los LDRs y entonces luego poder montar el resto de la box a una altura óptima, según los resultados de las pruebas que vaya ejecutando con la luz ya montada.

Cada beat de la cara superior de la box tendrá 6 X 6 cm, decido este número porque los objetos tendrán un tamaño optimo que pueda recoger una sola mano, teniendo en cuenta estas medidas, el resultado final de la cara superior será de 24 X 48. Finalmente para no apurar demasiado las medidas y el espacio colocaré los LDRs en una placa de metacrilato de 50 X 30. El aspecto final que tendría la caja con los LDRs incorporados sería como muestra el grafico.

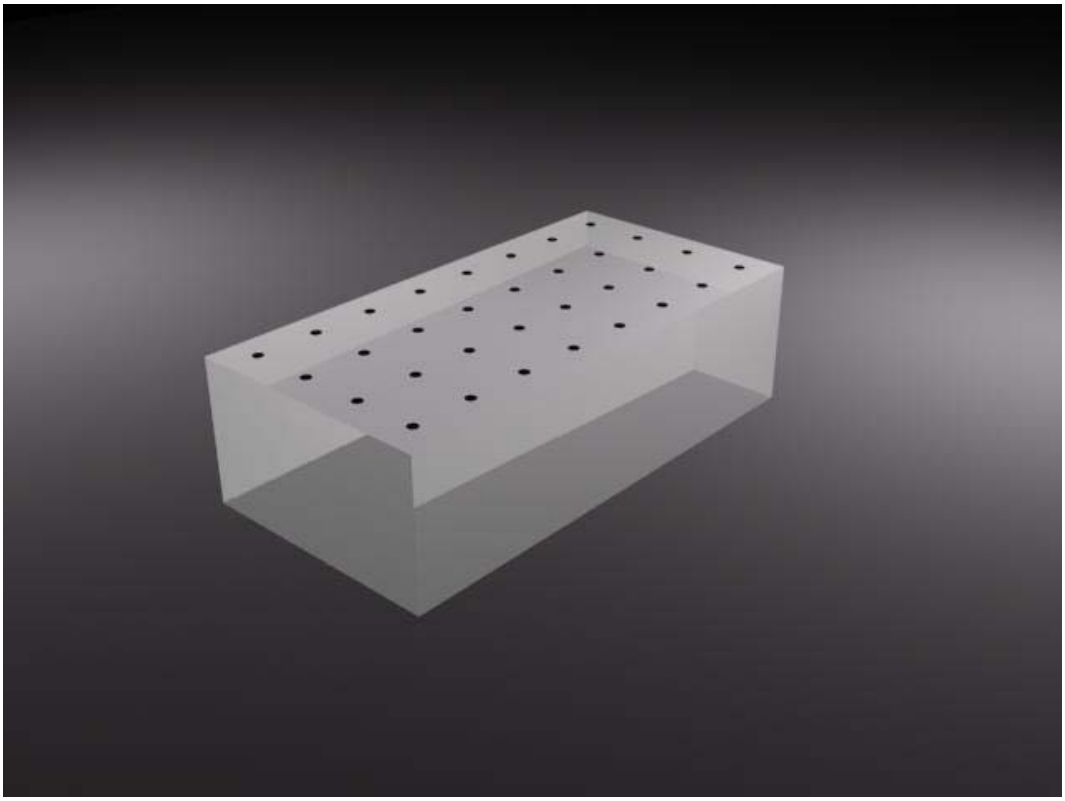

Fig. 22 Representación 3D *Fiametta 2.0.*

ii. Implementación del hardware

Para comenzar con la implementación de los LDRs en la cara superior de la "box" primero he de idear como irán pegados a ella. He pensado en encapsular los LDRs dentro de algo que le impida tomar lectura de la luz que viene de los lados y por debajo de los LDRs, y así

solo tomen lectura de la luz que incide justo encima de ellos, con la cantidad mínima de interferencias lumínicas.

Estoy pensando en hacer las cápsulas con algunos materiales como el corcho, pero después de hacer algunas pruebas con linternas, descubro que el corcho no es un material del todo opaco, así que necesito buscar otro material. Decido probar con una goma de borrar, ya que al realizar las pruebas compruebo que no es traslucido, y además es un material de muy fácil manejo para poder encapsular los LDRs. El Grafico muestra la primera implementación de los beats.

Fig. 23 Proceso de Implementación del Hardware

Fig. 24 Proceso de Implementación del Hardware

Con la punta de un cable RCA roto, hice la pequeña bóveda donde va incrustado el LDR. La goma de borrar irá pegada a la cara superior de la box.

Luego de unas pruebas determino que la goma de borrar no va a ser el material definitivo que le ponga a los LDRs, por dos motivos:

- Al hacer agujeros a la goma con las propias partas del LDR, quedan partes por donde entra la luz, algo no compatible con el resultado final que voy a darle.

- La goma de borrar resulta ser un material muy poco fiable a la hora de ser pegado al metacrilato, y se despega con una media de 8 pequeños golpes en la box.

Considerando otros diferentes materiales entre chatarra, descubro que las partes de un cubo de rubik, tienen en el centro, un agujero, que es del tamaño exacto de un LDR, así que me determino a usar estas partes del cubo de Rubik que luego relleno con una pasta no conductora y opaca. El acabado final del encapsulado de los LDR sería el que muestra el gráfico.

Fig. 25 Proceso de Implementación del Hardware

Fig. 26 Proceso de Implementación del Hardware

A continuación de haber encapsulado los LDRs, me dispongo a colocarlos en la cara superior de la box, para ello trazo, encima del metacrilato, en una pegatina, los puntos en donde irían las capsulas con los LDRs.

Fig. 27 Proceso de Implementación del Hardware

Crearé un patrón de colores en las capsulas, para saber con una simple y rápida mirada en que beat me encuentro. Así que pinto algunas cápsulas en total ocho con spray, al final irá una capsula pintada cada 4 beats.

Fig. 28 Proceso de Implementación del Hardware

Antes de comenzar a pegar las capsulas de LDRs al metacrilato, ejecuto una sencilla prueba con un LDR conectado directamente a arduino y sobrepuesto en el metacrilato, para así ir determinando, si el futuro montaje de la box será adecuado para *Fiametta 2.0.*, y también ir definiendo dudas que todavía mantengo de como irá puesta la luz, si esta irá por dentro o por encima de la box.

Fig. 29 Proceso de Implementación del Hardware

Los valores del LDR que arroja la prueba son normales, como sin el metacrilato, y la prueba sentencia que la luz podrá ir colocada tanto por encima de de la box como por debajo. Finalmente dejo por decidir donde se colocará la luz para cuando pueda hacer pruebas más consistentes. Aunque en un principio prefiero inclinarme porque la luz vaya colocada por debajo del metacrilato, debido a la mejora en la manejabilidad de *Fiametta 2.0.*

Una vez terminada la prueba comienzo a pegar las cápsulas de los LDRs al metacrilato.

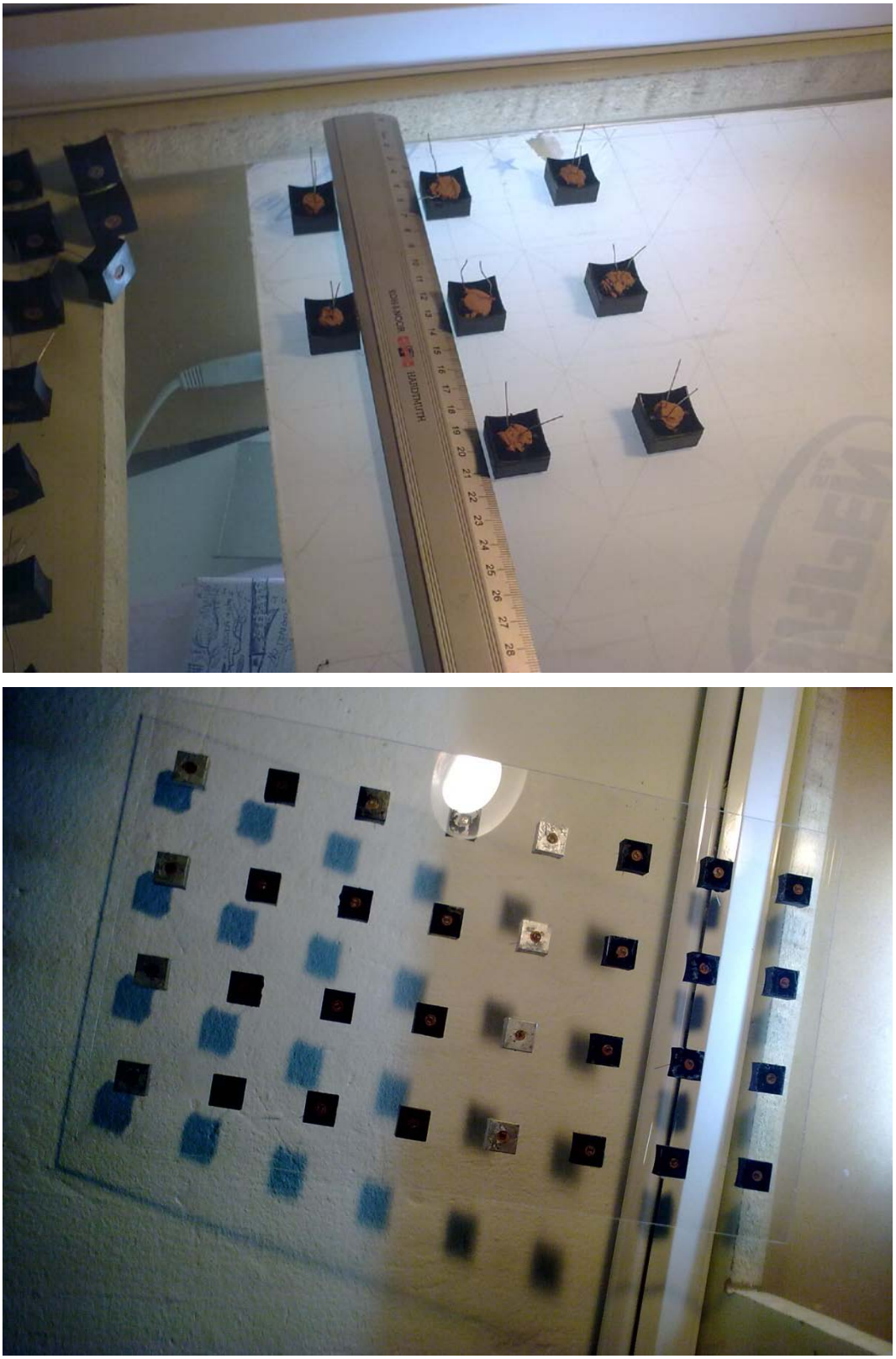

Fig. 30 Proceso de Implementación del Hardware

Una vez pegadas las cápsulas, ahora soldaré las resistencias que acompañan a los LDRs directamente a una de las patas de estos. Los LDRs son resistencias variables no tienen polaridad, por lo que da lo mismo a que pata del LDR debo soldarlo. Para empezar las soldaduras, recubro el metraquilato con cinta de pintor para evitar que el estaño caliente caiga en ella y manche o queme el metacrilato.

Fig. 31 Proceso de Implementación del Hardware

Luego comienzo a soldar, utilizando lo que se llama *tercer brazo*, que vienen a ser unas pinzas que juntan ambas piezas para que puedan ser soldadas. Tal como se ve en el gráfico.

Fig. 32 Proceso de Implementación del Hardware

Fig. 33 Proceso de Implementación del Hardware

Una vez soldadas todas las resistencias a una de las patas de los LDRs, sueldo dos cables, uno que recorra las patas de los 5 Voltios y otro que recorra las patas de las resistencias que llevarán el Ground. Como muestra el gráfico.

Fig. 34 Proceso de Implementación del Hardware

Finalmente comienzo a soldar los cables que van desde los LDRs hasta la protoboard y conectados a los multiplexores, es decir, los que se encargan de la lectura de datos de los LDRs. Los montaré con cables de colores para facilitar su identificación en el entramado de cables.

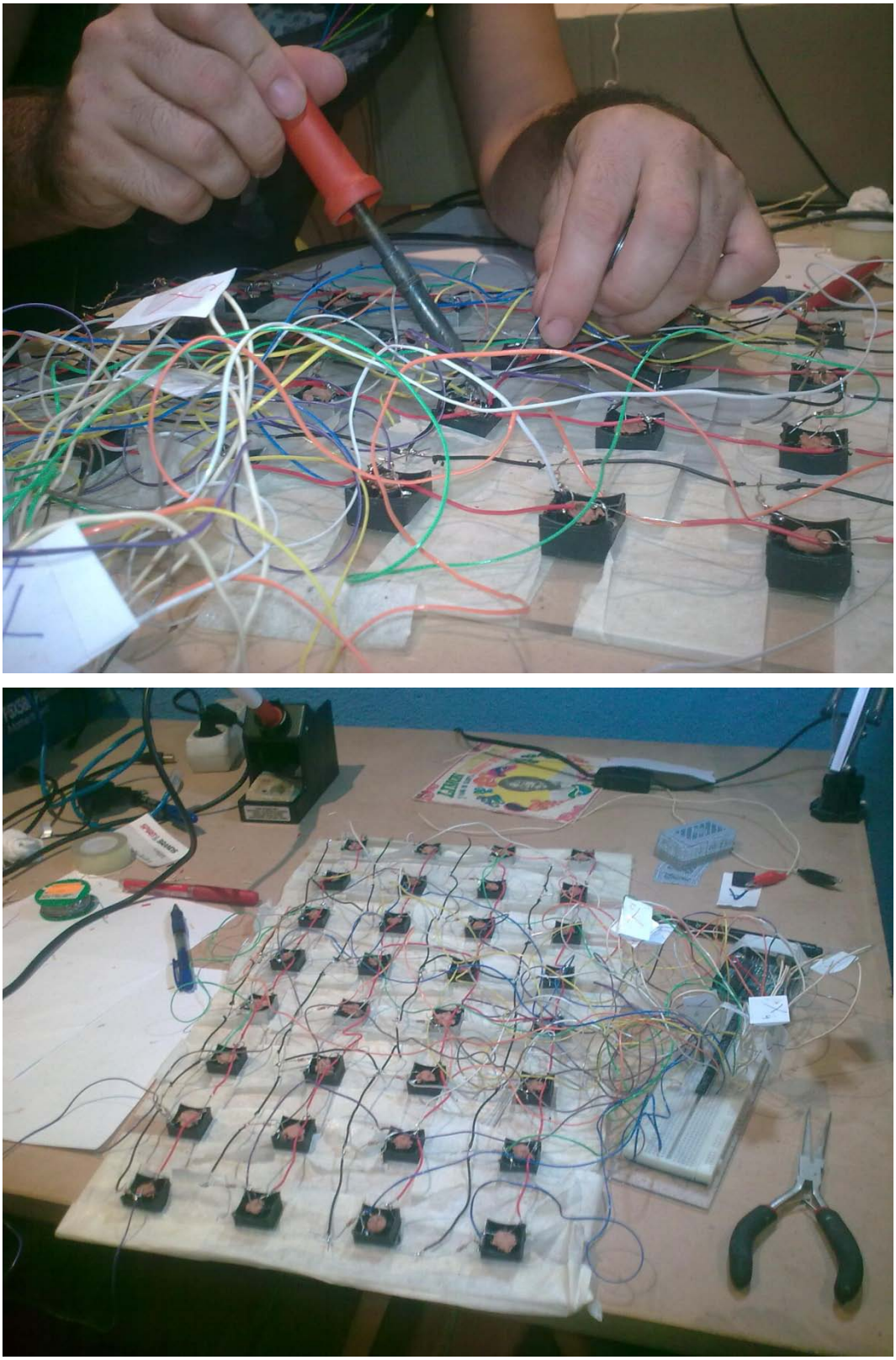

Fig. 35 Proceso de Implementación del Hardware

iii. Medición del hardware

Finalmente, después de haber montado todo el circuito provisional, me dispongo a hacer pruebas con cada fila de 8 LDRs para facilitar la búsqueda de cualquier problema en la lectura de algún LDR en particular.

I. Descripción de la prueba 1.

Dispongo a realizar una prueba, cada 8 LDRs, con un solo multiplexor, es decir, que en total haré cuatro pruebas. Cada vez que comience una prueba, conectaré las LDRs pertinentes a un solo multiplexor. Este, a su vez, estará conectado a Arduino. Para las cuatro pruebas utilizaré un código ejecutado en la consola de Arduino, tendrá un delay grande en cada lectura, para poder identificar algún problema o un cortocircuito en alguno de los LDRs.

La prueba la realizo colocando una linterna justo delante de cada uno de los LDRs, si el ciclo se completa con 7 LDRs de nivel bajo de luz y uno con un nivel alto. Es que el LDR está leyendo correctamente los datos y el circuito está bien.

II. Código de la prueba 1.

```
  int x0,x1,x2,x3,y0,y1,y2,y3,xx3,xx2,xx1,xx0,yy0,yy1,yy2,yy3;
  int a,b;
void setup(){
 a=8;
 b=9;
  pinMode (a, OUTPUT);
  pinMode (b, OUTPUT);
```

```
  Serial.begin(57600);
}
void loop(){

  digitalWrite (a,HIGH);
  digitalWrite (b,HIGH);
  delay(500);
  x3 = analogRead(1);
  y3 = analogRead(2);
  xx3 = x3/4;
  yy3 = y3/4;
  Serial.println(xx3,DEC);
  Serial.println(yy3,DEC);
  delay(2);


  digitalWrite (a,LOW);
  digitalWrite (b,HIGH);
  delay(500);
  x2 =analogRead(1);
  y2 = analogRead(2);
  xx2 = x2/4;
  yy2 = y2/4;
  Serial.println(xx2,DEC);
  Serial.println(yy2,DEC);
  delay(2);


  digitalWrite (a,HIGH);
  digitalWrite (b,LOW);
  delay(500);
```

```
  x1 =analogRead(1);
  y1 = analogRead(2);
  xx1 = x1/4;
  yy1 = y1/4;
  Serial.println(xx1,DEC);
  Serial.println(yy1,DEC);
  delay(2);


  digitalWrite (a,LOW);
  digitalWrite (b,LOW);
  delay(500);
  x0 =analogRead(1);
  y0 = analogRead(2);
  xx0 = x0/4;
  yy0 = y0/4;
  Serial.println(xx0,DEC);
  Serial.println(yy0,DEC);
  delay(2);

}
```

III. **Medición y Resolución de problemas.**

Según los resultados de la prueba, he podido comprobar cómo hay dos LDRs que no están dando los datos correctamente, al comprobarlos determino que era debido a un pequeño cortocircuito provocado por el contacto de las dos patas del LDR, luego de solucionarlo, ejecuto las pruebas de nuevo, esta vez con buenos resultados.

Finalmente realizo unas últimas soldaduras que une cada serie de 8 LDRs con las otras, tanto en el cable del Ground como en el cable que lleva los 5 Voltios.

iv. Conclusión

Las pruebas realizadas dan, por primera vez, resultados totalmente satisfactorios en la lectura de los datos que aportan los LDRs, este ha sido un gran paso en el desarrollo de *Fiametta 2.0.*A continuación he de disponer que hacer con esos datos mediante la programación de nuevos códigos o patches que ejecuten los cambios y la secuenciación de audio.

He decidido finalmente utilizar otro software, Max MSP, un entorno de programación muy parecido a Pure data, el motivo de este cambio es que este software Ableton Live, que es un secuenciador de audio y MIDI, conocido como DAW (Digital Audio Workstation). Y que me ahorraría mucho trabajo y problemas en la programación para conectar Pure Data con otro software DAW.

2.d. Cuarta Fase: Segundo desarrollo del software *Fiametta 2.0.*

a. Experiencia D: Software Max MSP

i. Descripción de Max MSP.

Max MSP es un lenguaje de programación de relativa similitud con Pure Data, también utiliza un modelo de programación que se denomina como "diagrama de flujo de datos" (DFD por sus siglas en español e inglés), recordemos que viene a ser una representación gráfica para la maceta del "flujo" de datos a través de un sistema de información. Es decir que un diagrama de flujo de datos también se puede utilizar para la visualización de procesamiento de datos (diseño estructurado). Estos diagramas utilizan símbolos con significados bien definidos que representan los pasos del algoritmo, y representan el flujo de ejecución mediante flechas que conectan los puntos de inicio y de término.

Más en concreto Max MSP es un entorno de desarrollo gráfico para música y multimedia desarrollado y mantenido por Cycling '74, una empresa de programas situada en San Francisco. El programa ha sido usado durante más de quince años por compositores, artistas y diseñadores de programas interesados en la creación de programas interactivos.

Max MSP es bastante modular, y la mayoría de las rutinas forman parte de una biblioteca compartida. La IPA (Interfaz de Programación de Aplicaciones) permite el desarrollo de nuevas rutinas (llamadas «objetos externos») por terceras personas. Por consecuencia, muchos de los usuarios de Max son programadores no afiliados a Cycling '74 que

mejoran el programa, creándole extensiones comerciales y no comerciales. Debido a su diseño extensible e interfaz gráfica (que de manera novedosa representa la estructura del programa y el IGU presentadas simultáneamente al usuario), Max es considerado por muchos como la lengua franca para el desarrollo de programas de música interactiva.

MI decisión ha sido finalmente realizar el software final de *Fiametta 2.0.* en Max MSP por una razón principal, y es que quiero utilizar el software Ableton Live como secuenciador ya construido, sin necesidad de empezar desde cero a construir un secuenciador virtual con Pure Data. Max MSP ofrece la posibilidad de integrar un patch realizado en Max MSP, en el software Ableton Live. Por lo que la tarea de pasar los datos leídos por los LDRs mediante un proceso de Serial, se verá mucho más facilitada que utilizando Pure Data. Además de esto, Max MSP tiene una interface muy parecida Pure Data, por lo que en mi opinión no tendré muchos problemas en aprender este nuevo lenguaje de programación.

ii. Descripción del Código de Arduino para 16 LDRs.

Comenzaré haciendo el patch de 16 LDRs, debido a su mayor sencillez y luego modificare el patch para convertirlo un patch de 32 LDRs.

En primer lugar necesito crear un código para Arduino que soporte la lectura de 16 LDRs. Básicamente tendría el mismo proceso que el patch que realicé para 8 LDRs, simplemente multiplicando por 2 los procesos y los cambios. El sketch final sería capaz de leer los 16 LDRs, utilizando 4 entradas analógicas y 2 multiplexores 4052.

iii. Código de Arduino para 16 LDRs.

```
int x0,x1,x2,x3,y0,y1,y2,y3,xx3,xx2,xx1,xx0,yy0,yy1,yy2,yy3;
int x0b,x1b,x2b,x3b,y0b,y1b,y2b,y3b,xx3b,xx2b,xx1b;
int xx0b,yy0b,yy1b,yy2b,yy3b;
int a,b,c,d;

void setup(){
        a=6;
        b=7;
        c=8;
        d=9;
         pinMode (a, OUTPUT);
         pinMode (b, OUTPUT);
         pinMode (c, OUTPUT);
         pinMode (d, OUTPUT);

         Serial.begin(57600);
}
void loop(){

         digitalWrite (a,HIGH);
         digitalWrite (b,HIGH);
                 delay(2);
         x3 = analogRead(0);
         y3 = analogRead(1);
         xx3 = x3/4;
         yy3 = y3/4;
         Serial.print(xx3,BYTE);
         Serial.print(yy3,BYTE);
                 delay(2);
```

```
digitalWrite (c,HIGH);
digitalWrite (d,HIGH);
        delay(2);
x3b = analogRead(2);
y3b = analogRead(3);
xx3b = x3b/4;
yy3b = y3b/4;
Serial.print(xx3b,BYTE);
Serial.print(yy3b,BYTE);
        delay(2);


digitalWrite (a,LOW);
digitalWrite (b,HIGH);
        delay(2);
x2 =analogRead(0);
y2 = analogRead(1);
xx2 = x2/4;
yy2 = y2/4;
Serial.print(xx2,BYTE);
Serial.print(yy2,BYTE);
        delay(2);

digitalWrite (c,LOW);
digitalWrite (d,HIGH);
        delay(2);
x2b =analogRead(2);
y2b = analogRead(3);
xx2b = x2b/4;
yy2b = y2b/4;
Serial.print(xx2b,BYTE);
```

```
 Serial.print(yy2b,BYTE);
          delay(2);

 digitalWrite (a,HIGH);
 digitalWrite (b,LOW);
          delay(2);
 x1 =analogRead(0);
 y1 = analogRead(1);
xx1 = x1/4;
 yy1 = y1/4;
 Serial.print(xx1,BYTE);
 Serial.print(yy1,BYTE);
          delay(2);

 digitalWrite (c,HIGH);
 digitalWrite (d,LOW);
          delay(2);
 x1b =analogRead(2);
 y1b = analogRead(3);
 xx1b = x1b/4;
 yy1b = y1b/4;
 Serial.print(xx1b,BYTE);
 Serial.print(yy1b,BYTE);
          delay(2);


 digitalWrite (a,LOW);
 digitalWrite (b,LOW);
          delay(2);
 x0 =analogRead(0);
 y0 = analogRead(1);
 xx0 = x0/4;
```

```
        yy0 = y0/4;
        Serial.print(xx0,BYTE);
        Serial.print(yy0,BYTE);
                delay(2);

        digitalWrite (c,LOW);
        digitalWrite (d,LOW);
                delay(2);
        x0b =analogRead(2);
        y0b = analogRead(3);
        xx0b = x0b/4;
        yy0b = y0b/4;
        Serial.print(xx0b,BYTE);
        Serial.print(yy0b,BYTE);
                delay(2);
}
```

iv. Medición y Resolución de problemas: Código para 16 LDRs.

Para realizar la prueba y comprobar que el código es correcto, cambiaremos la forma de envío de los valores a decimal, es decir ,cambiamos todos los comandos de "Serial.print(*x*,BYTE)" a "Serial.print (*x*,DEC)", para poder comprobar de una forma sencilla que los valores de las lecturas de los LDRs enviados al puerto serie son correctos, además, se amplían todos los delays a 500 milisegundos, así dará tiempo a comprobar en la consola de Arduino que los datos se están recibiendo correctamente.

A continuación de hacer la prueba en la consola de Arduino, con resultados satisfactorios, nos disponemos a comenzar con el patch que había que desarrollar en Max MSP.

v. Descripción del patch de Max MSP para 16 LDRs.

En primer lugar necesito utilizar el objeto *serial*: la sintaxis de este objeto en Max sería *serial (letra del numero de puerto) (Velocidad de transferencia de datos),* en nuestro caso, al tratarse del puerto 5 el comando sería: "Serial e 57600". El objeto Serial tiene 1 inlet y 2 outlet, en mi patch utilizare el mensaje *open* y *close* a través del inlet para abrir y cerrar la lectura del puerto. Además introduciré el objeto *metro*: la sintaxis de este objeto es: *metro (nº de milisegundos),* el objeto *metro* envía un bang cada vez que realiza un ciclo de x milisegundos. En nuestro ejemplo, utilizaremos el objeto *metro* de 100 milisegundos, por lo que cada 100 milisegundos el objeto *metro* enviara un bang. El objeto *metro* será introducido por el inlet del objeto *serial* y esto provocará que los datos recibidos por el puerto serie se impriman en la consola de Max MSP, cada 100 milisegundos.

Después de estar estudiando y leyendo más acerca de los comandos de Max MSP, encuentro el objeto match: La sintaxis del objeto sería *match (x x x...).* El objeto match tiene 1 inlet y 1 outlet. La función del objeto *match* es colocar y ordenar una lista de valores que entran a través de su inlet, la forma de ordenarlos es la siguiente: Se envía un número infiltrado en la lista de los valores que leen los LDRs que enviará arduino, para ello tendré que variar la programación del sketch de Arduino, añadiéndole el Byte elegido para enviar por el puerto serie. Este byte enviado será el 1, por ejemplo, y luego se envían la lectura de datos realizada por los LDRs, el objeto "match" recogerá esos datos y los

ordenara poniendo en primer lugar al Byte 1, por lo que el orden de llegada de datos siempre será el mismo.

Por último utilizaré el objeto *unpack* para desempaquetar la lista de datos que nos envía el objeto *match*, para que después vayan a objetos números, y queden impresos en pantalla.

vi. Patch de Max MASP para 16 LDRs.

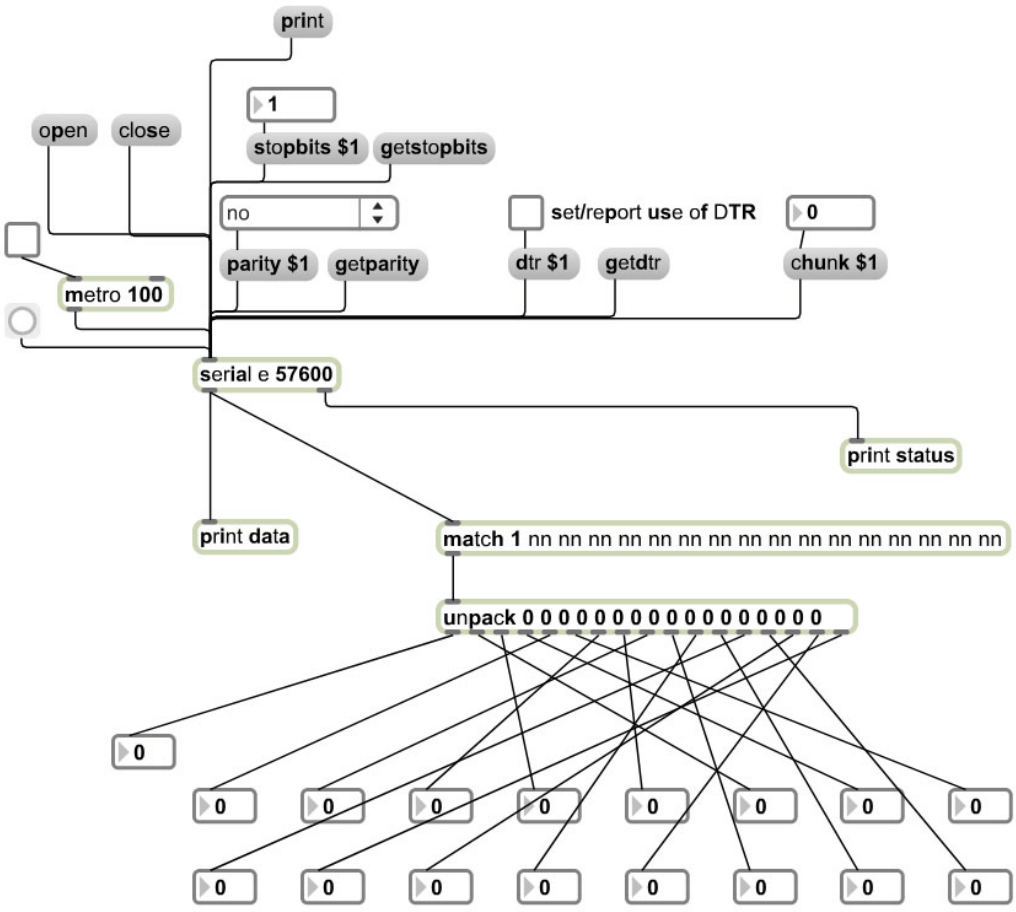

Fig. 36 Patch 16 LDRs Max MSP

vii. Medición y Resolución de Problemas: Patch para 16 LDRs.

Realizo una sencilla prueba con una linterna, verificando que los datos del patch se están reflejando adecuadamente en los números, y con el orden predispuesto.

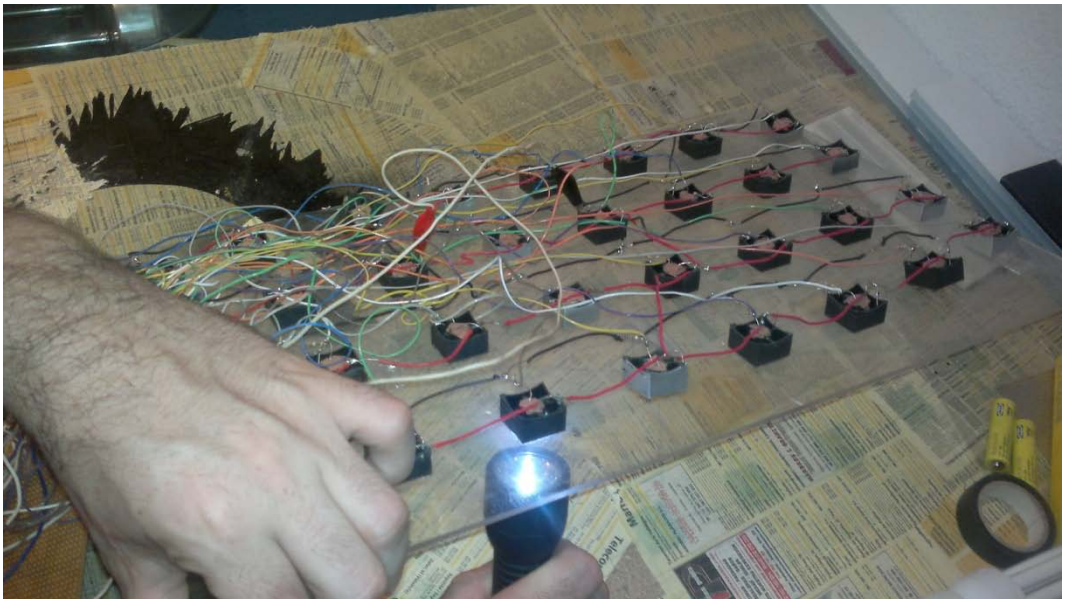

Fig. 37 Prueba 1

Encuentro un problema con el patch, y es que los números del patch se refrescan muy lentamente, y sin embargo los números de la consola se refrescan adecuadamente. Finalmente al no encontrar una solución visible, empiezo por revisar el sketch de arduino. Descubro cual es la problemática y es que sin querer en el sketch de arduino había puesto una A en vez de un 1 en el byte que envía arduino y que garantiza que la lista llega en orden a Max MSP mediante el objeto match.

Las pruebas, finalmente, resultan ser satisfactorias, cada LDR envía un valor correcto y constante a los números del patch de Max MSP. Ahora me dispondré a convertir el patch que he realizado de 16 LDRs en un patch de 32 LDRs.

viii. Descripción del código de Arduino para 32 LDRs

Lo primero que tengo que pensar es que para realizar el cambio de programación que necesita el código de Arduino, hay tener en cuenta que ahora voy a cambiar una programación hecha para que 16 lecturas de datos provenientes de 2 multiplexores entren en 4 entradas analógicas de Arduino. Lo que quiero conseguir con la nueva programación es que 32 lecturas de datos provenientes de 5 multiplexores, entren en tan solo 2 entradas analógicas de Arduino.

La programación de Arduino cambiaría bastante ya que ahora debemos manejar dos multiplexores en cada lectura de datos, en vez de uno solo, ya que hay que coordinar al multiplexor que va conectado directamente al LDR, con el multiplexor que recibe la señal de los otros multiplexores.

He cambiado el primer byte de la lista que envío al puerto serie de 1 a 254, ya que en ocasiones los Leds alcanzaban el valor 1 y esto hacia que toda la lista se desordenase al llegar al patch de Max MSP, sin embargo nunca alcanzaban el 254, y es por esto que decidí cambiarlo.

ix. Código de Arduino para 32 LDRs

```
int
x0x0,x0x1,x0x2,x0x3,y0y0,y0y1,y0y2,y0y3,x1x0,x1x1,x1x2,x1x3,y1
y0,y1y1,y1y2,y1y3,x2x0,x2x1,x2x2,x2x3,y2y0,y2y1,y2y2,y2y3,x3x0,
x3x1,x3x2,x3x3,y3y0,y3y1,y3y2,y3y3;
 int
x0x0cuarto,x0x1cuarto,x0x2cuarto,x0x3cuarto,y0y0cuarto,y0y1cuar
to,y0y2cuarto,y0y3cuarto,x1x0cuarto,x1x1cuarto,x1x2cuarto,x1x3c
```

```
uarto,y1y0cuarto,y1y1cuarto,y1y2cuarto,y1y3cuarto,x2x0cuarto,x2x
1cuarto,x2x2cuarto,x2x3cuarto,y2y0cuarto,y2y1cuarto,y2y2cuarto,y
2y3cuarto,x3x0cuarto,x3x1cuarto,x3x2cuarto,x3x3cuarto,y3y0cuart
o,y3y1cuarto,y3y2cuarto,y3y3cuarto;
  int a0,b0,a1,b1,a2,b2,a3,b3,aaa,bbb;

void setup(){
        a0=4;b0=5;a1=6;b1=7;aaa=8;bbb=9;
        a2=10;b2=11;a3=12;b3=13;

        pinMode (a0, OUTPUT);
        pinMode (b0, OUTPUT);
        pinMode (a1, OUTPUT);
        pinMode (b1, OUTPUT);
        pinMode (a2, OUTPUT);
        pinMode (b2, OUTPUT);
        pinMode (a3, OUTPUT);
        pinMode (b3, OUTPUT);
        pinMode (aaa, OUTPUT);
        pinMode (bbb, OUTPUT);

        Serial.begin(57600);
}
void loop(){

        Serial.print(254,BYTE);

        digitalWrite (aaa,LOW);
        digitalWrite (bbb,LOW);

        digitalWrite (a0,LOW);
        digitalWrite (b0,LOW);
```

```
        delay(1);

x0x0 = analogRead(0);
y0y0 = analogRead(1);
x0x0cuarto = x0x0/4;
y0y0cuarto = y0y0/4;
Serial.print(x0x0cuarto,BYTE);
Serial.print(y0y0cuarto,BYTE);
        delay(1);

digitalWrite (a0,HIGH);
digitalWrite (b0,LOW);
        delay(1);

x0x1 = analogRead(0);
y0y1 = analogRead(1);
x0x1cuarto = x0x1/4;
y0y1cuarto = y0y1/4;
Serial.print(x0x1cuarto,BYTE);
Serial.print(y0y1cuarto,BYTE);
        delay(1);

digitalWrite (a0,LOW);
digitalWrite (b0,HIGH);
          delay(1);

x0x2 = analogRead(0);
y0y2 = analogRead(1);
x0x2cuarto = x0x2/4;
y0y2cuarto = y0y2/4;
Serial.print(x0x2cuarto,BYTE);
Serial.print(y0y2cuarto,BYTE);
```

```
        delay(1);

digitalWrite (a0,HIGH);
digitalWrite (b0,HIGH);
         delay(1);

x0x3 = analogRead(0);
y0y3 = analogRead(1);
x0x3cuarto = x0x3/4;
y0y3cuarto = y0y3/4;
Serial.print(x0x3cuarto,BYTE);
Serial.print(y0y3cuarto,BYTE);
         delay(1);


digitalWrite (aaa,HIGH);
digitalWrite (bbb,LOW);

digitalWrite (a1,LOW);
digitalWrite (b1,LOW);
         delay(1);

x1x0 = analogRead(0);
y1y0 = analogRead(1);
x1x0cuarto = x1x0/4;
y1y0cuarto = y1y0/4;
Serial.print(x1x0cuarto,BYTE);
Serial.print(y1y0cuarto,BYTE);
         delay(1);

digitalWrite (a1,HIGH);
digitalWrite (b1,LOW);
```

```
        delay(1);

x1x1 = analogRead(0);
y1y1 = analogRead(1);
x1x1cuarto = x1x1/4;
y1y1cuarto = y1y1/4;
Serial.print(x1x1cuarto,BYTE);
Serial.print(y1y1cuarto,BYTE);
        delay(1);

digitalWrite (a1,LOW);
digitalWrite (b1,HIGH);
        delay(1);

x1x2 = analogRead(0);
y1y2 = analogRead(1);
x1x2cuarto = x1x2/4;
y1y2cuarto = y1y2/4;
Serial.print(x1x2cuarto,BYTE);
Serial.print(y1y2cuarto,BYTE);
       delay(1);

digitalWrite (a1,HIGH);
digitalWrite (b1,HIGH);
        delay(1);

x1x3 = analogRead(0);
y1y3 = analogRead(1);
x1x3cuarto = x1x3/4;
y1y3cuarto = y1y3/4;
Serial.print(x1x3cuarto,BYTE);
Serial.print(y1y3cuarto,BYTE);
```

```
        delay(1);

digitalWrite (aaa,LOW);
digitalWrite (bbb,HIGH);

digitalWrite (a2,LOW);
digitalWrite (b2,LOW);
      delay(1);

x2x0 = analogRead(0);
y2y0 = analogRead(1);
x2x0cuarto = x2x0/4;
y2y0cuarto = y2y0/4;
Serial.print(x2x0cuarto,BYTE);
Serial.print(y2y0cuarto,BYTE);
      delay(1);

digitalWrite (a2,HIGH);
digitalWrite (b2,LOW);
        delay(1);

x2x1 = analogRead(0);
y2y1 = analogRead(1);
x2x1cuarto = x2x1/4;
y2y1cuarto = y2y1/4;
Serial.print(x2x1cuarto,BYTE);
Serial.print(y2y1cuarto,BYTE);
        delay(1);

digitalWrite (a2,LOW);
digitalWrite (b2,HIGH);
        delay(1);
```

```
x2x2 = analogRead(0);
y2y2 = analogRead(1);
x2x2cuarto = x2x2/4;
y2y2cuarto = y2y2/4;
Serial.print(x2x2cuarto,BYTE);
Serial.print(y2y2cuarto,BYTE);
        delay(1);

digitalWrite (a2,HIGH);
digitalWrite (b2,HIGH);
        delay(1);

x2x3 = analogRead(0);
y2y3 = analogRead(1);
x2x3cuarto = x2x3/4;
y2y3cuarto = y2y3/4;
Serial.print(x2x3cuarto,BYTE);
Serial.print(y2y3cuarto,BYTE);
        delay(1);


digitalWrite (aaa,HIGH);
digitalWrite (bbb,HIGH);

digitalWrite (a3,LOW);
digitalWrite (b3,LOW);
        delay(1);

x3x0 = analogRead(0);
y3y0 = analogRead(1);
x3x0cuarto = x3x0/4;
```

```
y3y0cuarto = y3y0/4;
Serial.print(x3x0cuarto,BYTE);
Serial.print(y3y0cuarto,BYTE);
        delay(1);

digitalWrite (a3,HIGH);
digitalWrite (b3,LOW);
        delay(1);

x3x1 = analogRead(0);
y3y1 = analogRead(1);
x3x1cuarto = x3x1/4;
y3y1cuarto = y3y1/4;
Serial.print(x3x1cuarto,BYTE);
Serial.print(y3y1cuarto,BYTE);
        delay(1);

digitalWrite (a3,LOW);
digitalWrite (b3,HIGH);
        delay(1);

x3x2 = analogRead(0);
y3y2 = analogRead(1);
x3x2cuarto = x3x2/4;
y3y2cuarto = y3y2/4;
Serial.print(x3x2cuarto,BYTE);
Serial.print(y3y2cuarto,BYTE);
        delay(1);

digitalWrite (a3,HIGH);
digitalWrite (b3,HIGH);
        delay(1);
```

```
x3x3 = analogRead(0);
y3y3 = analogRead(1);
x3x3cuarto = x3x3/4;
y3y3cuarto = y3y3/4;
Serial.print(x3x3cuarto,BYTE);
Serial.print(y3y3cuarto,BYTE);
        delay(1);

}
```

x. Medición y Resolución de problemas: Código de Arduino para 32 LDRs.

Al igual que antes, para realizar la prueba y comprobar que el sketch es correcto, cambio la forma de envío de los valores a decimal, es decir cambié todos los comandos de "Serial.print(*x*,BYTE)" a "Serial.print (*x*,DEC)", y para poder comprobar, fácilmente, que los valores de las lecturas de los LDRs enviados al puerto serie son correctos, amplío los valores de los comandos *delays* a 500 milisegundos, así tengo el tiempo suficiente para comprobar en la consola de Arduino, si los datos se están recibiendo correctamente.

La Medición de la prueba es correcta en todos los LDRs, el siguiente paso es crear un código para Max MSP que recoja los 32 datos de las mediciones de los LDRs.

xi. Descripción del patch de 32 LDRs

El patch para 32 LDRs, en nuestro caso sería idéntico al patch original de 16 LDRs añadiendo el doble numero de cajas de objetos números y elementos en los comandos match y unpack.

xii. Patch de Max MSP de 32 LDRs

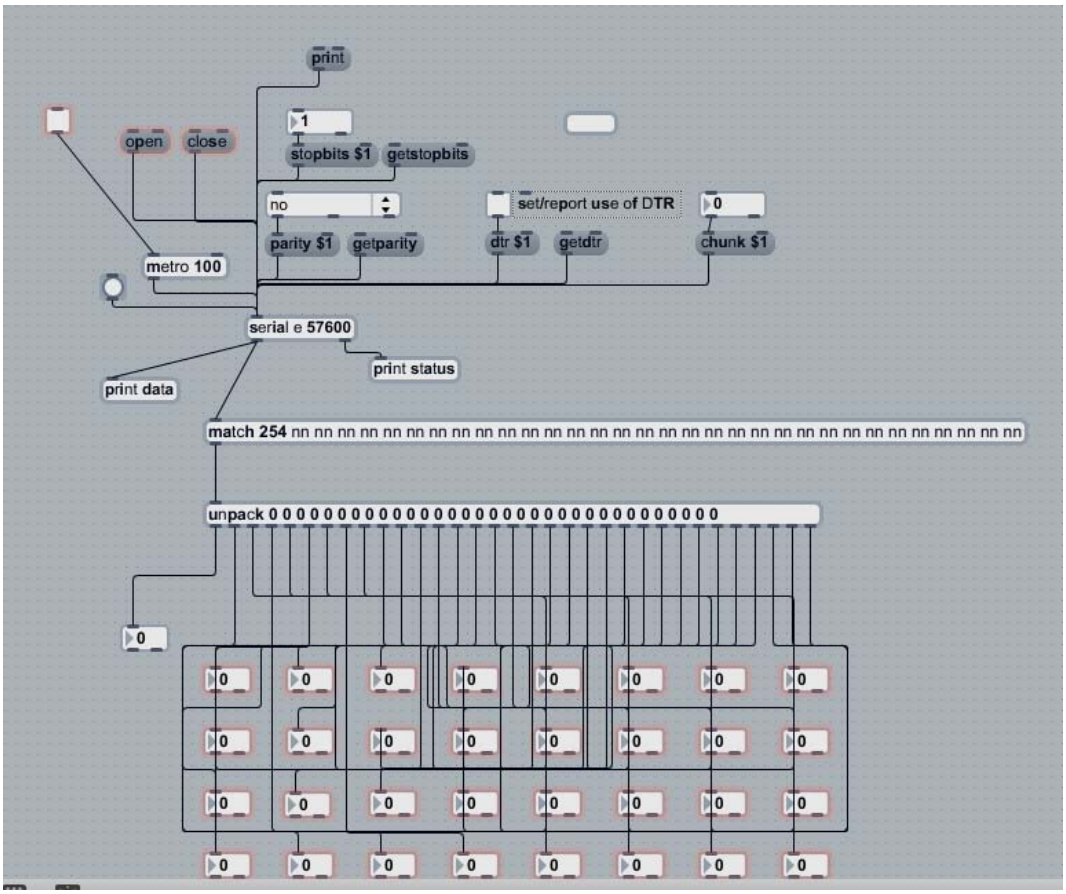

Fig. 38 Patch 32 LDRs Max MSP

xiii. Descripción del Patch "generación y distorsión 1"

Una vez, completado el patch que recibe los datos de las lecturas de los 32 LDRs de arduino, comienzo a hacer un patch en el que utilice los datos entrantes de los LDRs, para generar y distorsionar el sonido con diferentes métodos.

Paso 1. En primer lugar descargo un patch para Max for Live llamado Total Random, este patch lleva integradas gran cantidad de variaciones y generaciones de sonido, utilizaré cada uno de los valores de los LDRs, para asignarlo a diferentes campos.

Paso 2. *Harmonicity Ratio, (Proporción de armonicos, H)* Utilizaré los 3 primeros LDRs, para interactuar con H, la relación de armonicos es uno de los más importantes componentes que se utiliza en la *FM syntesys* para obtener diferentes resultados en el sonido. Básicamente controla la proporción de armonicos en un sonido. En nuestro caso tenemos tres controles*: Harm, LFOfrq, LFOamp,* los rangos interactuables en las diferentes variables son distintos para cada uno, por ejemplo, el rango interactuable de LFOfrq, va de 0 a 100, y no solo esto, sino que es un rango matemático, no aritmético, por lo que la diferencia en el sonido es más notable, cuanto más bajo son los valores, y cuanto más altos son menos se nota. Por ello tengo que ajustar los valores que llegan de los LDRs utilizando formulas matemáticas y algoritmos con sumas, divisiones, multiplicaciones, cosenos, logaritmos... todo ello, para que los rangos se adapten a la naturaleza de los gestos del usuario.

Paso 3. *Modulation Index (Indice de Modulación)* Otro de las variaciones que encontramos, en la FM syntesys, es el Indice de Modulación. Básicamente con él, cambiamos, tanto la frecuencia como la amplitud, en la modulación de las ondas de sonido. También tendremos

que conectarlo con algoritmos que den un buen resultado en la experiencia de la interacción del usuario. También son tres controles los que permiten cambiar el Índice de Modulación*: Index, LFOfrq, LFOamp.* Los cuales conecto a los siguientes 3 LDRs.

Paso 4. *Forma de la onda* También adaptaré uno de los números que dan los valores de un LDR, para que cambia la forma de la onda, hay cuatro posibles cambios, onda cuadrada, triangular, dientes de sierra y sinusoidal. Para ello, aplico rangos en los valores del LDR, teniendo en cuenta el factor usuario y como la luz que reciban los LDRs, será la responable de interactuar con el patch, mido la luz recibida para los diferentes objetos, y coloco los rangos en 40-65, 65-90. 90-120, 135-160. Cada rango representa diferentes materiales que reflejan más la luz que otros, en el último caso se trata de un espejo.

Paso 5. Repito los mismos pasos para otras variaciones en el sonido, tales como *Tremolo, Ganancia, Envelope o Brigthnes.* Mido y aplico algoritmos en cada una de las variaciones según los resultados de las pruebas*.* Así mismo también cambio la notación y tonalidad en los demás LDRs restantes.

xiv. Patch "generación y distorsión 1"

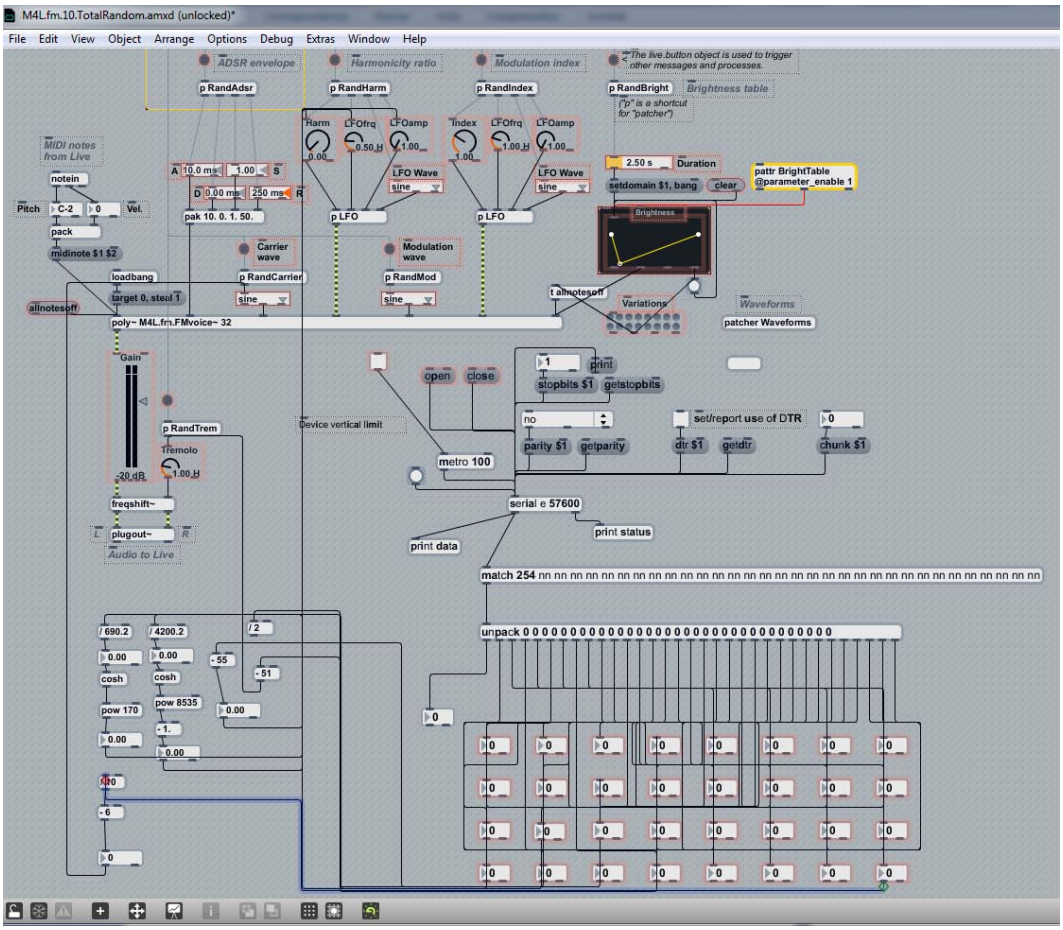

Fig. 39 Patch "generación y distorción 1"

xv. Medición y Resolución de Problemas

Las pruebas que ejecuté, para la medición de los algoritmos consistieron, en colocar los diferentes objetos que utilizaría en el desarrollo de la interacción Usuario-Computador, hacer mediciones, y aplicar los algoritmos adecuados a cada uno de los valores.

Fig. 40 Prueba 1 Patch Generación y Distorsión 1

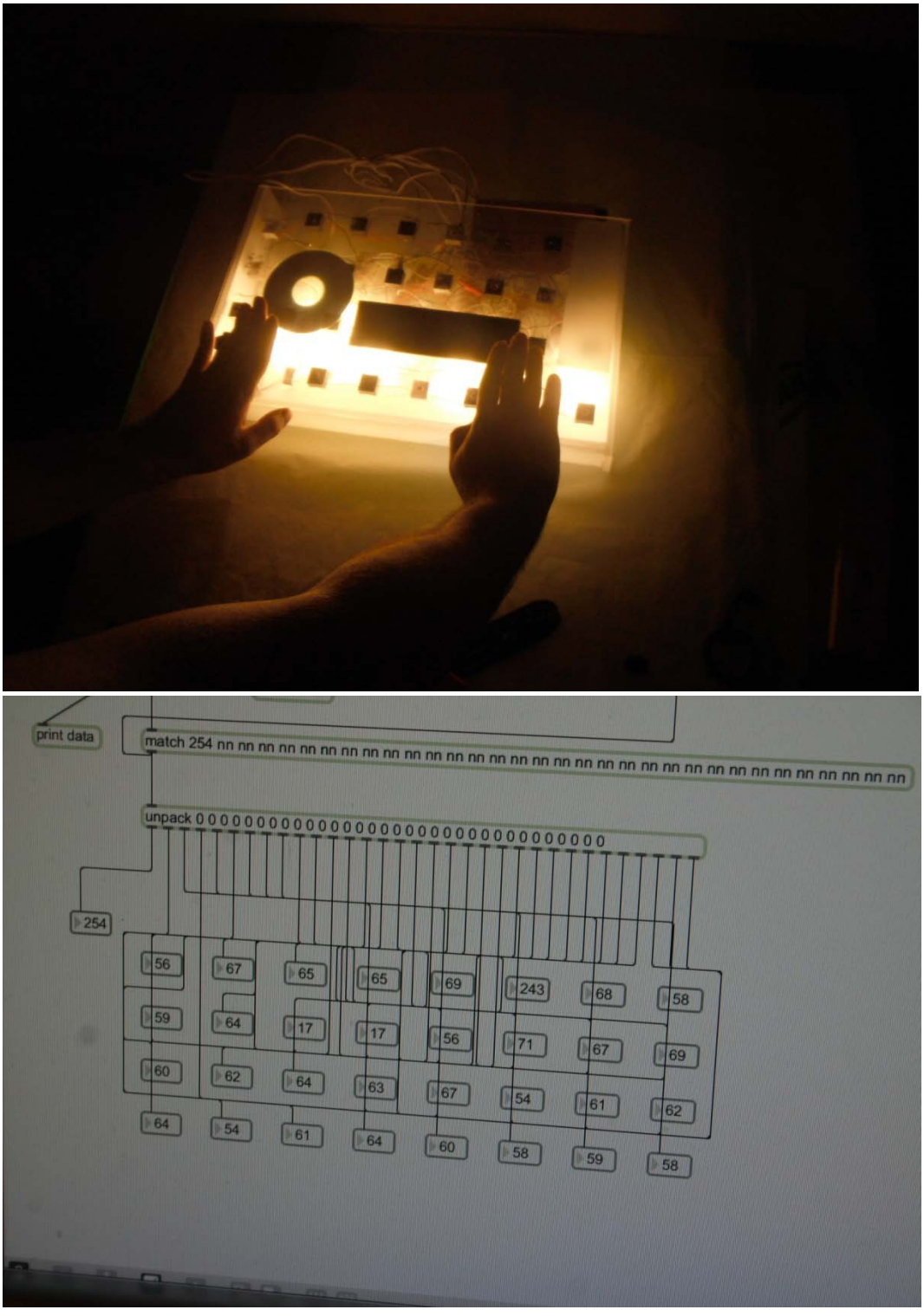

Fig. 41 Prueba 1. Generación y Distorsión 1

En el siguiente enlace se puede ver un video de la prueba 1 realizada en el desarrollo del patch "generación y distorsión 1", última fase cuyo proceso es descrita en el proyecto escrito de Fiametta 2.0. Actalmente la interface continua en proceso de ampliación y desarrollo.
URL: http://www.youtube.com/watch?v=wWx-k4Wt5kU

Conclusiones

A modo de conclusión nos gustaría apuntar que en esta Tesis de Máster hemos intentado formalizar de manera sobre todo práctica nuestra primera idea que no era otra que construir un estudio/prototipo de carácter artístico experimental y de naturaleza digital interactiva sonora.

Trabajando en paralelo entre cómo construir físicamente y cómo construir conceptualmente nuestra propuesta y marco de estudio, hemos sido conscientes que debemos enfocar nuestro trabajo hacia la resolución del ejercicio denominado tesis de máster (con todas sus características), no sin antes apuntar que la experiencia teórico/práctica que hemos asumido en este proceso nos ha despertado un aspecto metodológico, analítico y crítico para el estudio que antes no teníamos y que podría perfectamente ser una primera conclusión de este trabajo.

Si esa podría ser una primera conclusión, creemos haber detectado otras que nos permiten apuntar que:

La interfaz que hemos realizado está basada en lo que podemos denominar en las conclusiones como “naturalización”, una búsqueda de una mayor naturalidad/comodidad en la ejecución de experimentaciones digitales sonoras por medio de interfaces tangibles. Esta “naturalización” creemos haberla obtenido a través dos experiencias que hemos intentado reflejar en el trabajo:

- La experiencia del usuario/interprete en la ejecución de la performance.

- La experiencia de los espectadores del concierto/performance sonora.

Parte de la intención de este estudio/memoria final, al tratarse de una interfaz tangible, tenía como objetivo que la usabilidad, comodidad e

intuitividad sea mejorada con respecto a una interfaz grafica, ya que creemos, como conclusión, que resulta más cómodo y más natural manejar objetos tangibles para comunicarse con el ordenador, que mediante una interfaz grafica, y creemos que en un tanto por ciento hemos podido demostrar que estábamos en lo cierto en este aspecto al formular el objetivo ya que la conclusión de nuestro experimento/artefacto respecto a este punto ha sido que:

- Las interfaces tangibles permiten una mejor visualización a los espectadores, no-usuarios y un acompañamiento más natural de la interacción de un intérprete con su "instrumento".

En cuanto al diseño de la interfaz, de este proyecto, podemos también apuntar que algunos de los objetivos técnicos para la creación de una interfaz deben de ser la precisión, consistencia y fluidez en la traducción e interrelación entre lenguajes informáticos, algo que se contempló desde un principio, y que la sencillez de esta interrelación y en el aspecto del fácil y rápido aprendizaje en la ejecución/performance por parte del usuario , es crucial para que se den estos factores y es por ello que podemos apuntar como una conclusión más de este trabajo que:

- La interacción humano-ordenador debe de tener fluidez, consistencia, precisión en la traducción/interrelación de lenguajes, algo que se puede conseguir mediante la sencillez de las acciones que el usuario necesita ejecutar.

También es lícito apuntar en este apartado de conclusiones que la experimentación en el desarrollo práctico, en un marco más elevado, a nivel de programación y desarrollo de hardware, en vez de ceñirnos a utilizar software y hardware ya preconcebido por y para el mercado, nos ha dado un control más alto en la creación del proyecto, por lo que hemos

podido construir un prototipo "a medida" del propio artista. Este desarrollo”a medida” nos ha permitido dilucidar la siguiente conclusión:

- Los artefactos/aplicaciones técnicos-tecnológicos actuales en el mercado ya de por sí, nos imponen unas pautas de uso bastante férreas y limitadas las cuales sufrimos todos los usuarios de esas aplicaciones. Es posible flexibilizar, de una manera que podríamos denominar “subjetiva”, estas pautas rígidas, en pro de obtener un control más alto, libre y flexible del artefacto en pro de la performance resultante, manipulando y experimentando lo intrínseco de los propios artefactos-aplicaciones bajo los principios abiertos de Circuit Bending , Do it yourself , Open Hardware y Hacking Hardware .

El desarrollo del corpus práctico como memoria de trabajo, materializado en el prototipo presentado pone de manifiesto la importancia que se le ha de dar a documentar el proceso de trabajo en las prácticas neo mediales, para conseguir el resultado final deseado y la evaluación de los resultados obtenidos, para enlazarlos con futuros proyectos que sigan experimentando con los interfaces y las experiencias sonoras como medio de expresión, interacción y participación. Cabe añadir que el prototipo "Fiametta 2.0." sigue en proceso de ampliación y mejora, debido la extremada complejidad en la elaboración de una interface, robusta, transparente y que se ajuste totalmente a los resultados esperados.

Todos estos conceptos, hemos intentado que se reflejaran humildemente en el desarrollo tanto práctico como teórico de este proyecto de fin de Master.

Dado que mi proyecto, principalmente, se basa en el desarrollo práctico de una interfaz experimental sonora, a título individual, me

gustaría apuntar que los talleres y cursos propuestos en el Master AVM me han sido de gran ayuda y me han abierto infinitas puertas pero en especial, Diseño de Interfaces, Programación de comportamientos y modelos, Proyectos Interactivos y el Taller de Arduino impartido por el profesor David Cuartielles, me han sido vitales para el desarrollo y materialización del proyecto, tanto en cuanto a conceptos y referentes en el arte electrónico, que contemplo en mi proyecto, como de programación y electrónica básica, para poder elaborar la interfaz. Estos conocimientos y rudimentos básicos han sido explorados y creemos que han sido ampliados en la realización del proyecto práctico, pero sin esas bases no hubiera sido posible.

Agradecemos a todos los autores citados su trabajo ya que sin estos no hubiera sido posible el desarrollo del proyecto.

Y si bien ahora somos conscientes del volumen del tema que elegimos como estudio, no debemos olvidar y apuntar que en el proceso hemos encontrado líneas futuras de investigación interesantes en relación a la performance sonora en tiempo real (U2U- User to User) y multitask online (multitareas en red/multiusuarios) que antes no teníamos presentes.

Sin más agradecer al lector su paciencia e invitar a la lectura y uso de este trabajo a futuros investigadores esperando que les sea de utilidad.

BIBLIOGRAFÍA

ALVAREZ C.J. *Descartes y la ciencia del siglo XVII.* ed. siglo xxi s.a., Méjico, 2000 Pag. 138 citando a Baglivi, Praxis Médica, 1696, citado en Canguilhem, 1979. p121.

BANZI, M, *Getting started with Arduino,* O'Reilly, 2009

BRASLAVSKY, J.; *Sistemas No Lineales .* Cap. 6 Estabilidad Entrada-Salida. Documento: PDF, [Consulta: 12/05/2010] Url:http://www.eng.newcastle.edu.au/~jhb519/teaching/snolin/material/cap06.pdf

CAGE, J., *Silencio: conferencias y escritos*, Ardora, Madrid, 2002.

CARROLL, M. *Human Computer Interaction(HCI).* Documento PDF, [Consulta:02/06/2011]Url:http://www.interactiondesign.org/printerfriendly/encyclopedia/human_computer_interaction_hci.html

CASTILLO P.A.; *Estructura de Computadores I.* Documento PDF. [Consulta:14/05/2011]Url:http://atc.ugr.es/pedro/docencia/ec1/transparencias/evolucion_tecnologia-computadores.pdf

CHION, M. *La audiovisión: introducción a un análisis conjunto de la imagen y el sonido.* Paidós, Barcelona. 1998.

COLLINS, N. *Handmade Electronic Music: The art of hardware hacking.* Routledge Taylor & Francis Group, New York,1999.

DELEUZE, G.; *Postdata sobre las sociedades de control*, Éditions de Minuit, París, 1972.

FILDES, J.; *Oldest computer music unveiled.* BBC News [Consulta:25/03/2011]
Url:http://news.bbc.co.uk/2/hi/technology/7458479.stm

FOUCAULT, M.; *Vigilar y castigar nacimiento de la prisión,* México, Siglo Veintiuno Ediciones, 1976.

GIANNETTI, C. *Estética digital: Sintopía del arte, la ciencia y la tecnología,* Ed. Làngelot, Barcelona , 2002, p.119

GIANNETTI, C.; *Reflexiones acerca de la crisis de la imagen técnica la interfaz y el juego,* Analisi Quaderns de comunicació i cultura. N.27 UAB. Barcelona. Documento HTML, [Consulta: 06/07/2011] Url: http://www.bib.uab.es/pub/analisi/02112175n27.htm

GLINSKY A. *Theremin: Ether Music and Espionage.* University of Illinois Press, Illinois, 2000.

GUILLETT, C. *Historia del Rock: El sonido de la ciudad,* Ed. Robinbook, Barcelona, 2003.

HASS, J*.; Introduction to computer Music: Volume one.* Chapter Five.Web en linea.[Consulta:20/04/2011]
Url:http://www.indiana.edu/~emusic/etext/digital_audio/chapter5_digital2.shtml

HUTCHINS, E.; HOLLAN, J.; NORMAN, D., *Direct Manipulation Interfaces, en Multi-Media Database System;Visual Database Systems*, ed. T. L. Kunii, Elsevier Science Publishers, 1989.

JORDA. S, *Audio digital y MIDI,* Anaya, 1997

KETTLEWELL, B. *Electronic Music Pioneers.* ProMusic Press, California, 2002.

LÉVY, P. *¿qué es lo virtual?,* Ed. Paidós, Barcelona, 1999.

LICKLIDERY, J. Y CLARK, W.; *On-lineManComputerCommunications,* MIT, 1962.

LEWIS, C, RIEMAN, J. *Task centered user interface design: A practical introduction*. USA 190pp .1993.

LIZANA, D. *Música y Media.* Web en linea. [Consulta 23/04/2011] Url: http://musicaymedia.blogspot.com/2007/08/yamaha-dx7-el-hito.html

LLORENTE, A. y FRÍAS, R. *La electrónica en tus manos.* Ed. Penthalón, Madrid, 1988.

LOPEZ A.; *Monografico sobre John Cage.* Documento PDF[Consulta 22/04/2011]
Url:http://217.16.255.8/Obert/EP00006/swf/pdf/es/PDFcage.pdf

MAÑAS, M. Interfaces. Reglas/Teorías. Documento: PDF, [Consulta:04/07/2011]Url:http://personales.upv.es/moimacar/master/download/interfaces.pdf

MAÑAS, M, Reacción Vs Interacción. Algunos aspectos sobre interactividad. Documento: PDF, [Consulta:04/07/2011], Url:http://personales.upv.es/moimacar/master/download/interactivo_reactivo.pdf

MEDIEN KUNST NETZ. Web en Linea. [Consulta 20/04/2011] Url:http://www.medienkunstnetz.de/works/illiac-suite/

MORENO, I. *Leonardo da Vinci: el científico.* Documento PDF, [Consulta:05/02/2010]Url:http://centros5.pntic.mec.es/ies.victoria.kent/Rincon-C/Cie-Hist/Leonardo/ciencia.htm

SHELDON, T.P.; *The Dawn of Commercial Digital Recording.* Documento PDF [COnsulta:06/11!2010] Url: http://www.aes.org/aeshc/pdf/fine_dawn-of-digital.pdf

MORGAN, R.P. *La Música del siglo XX.* Akal, Madrid, 1999

NOBLE, J. *Programming interactivity* , Sebastopol CA, O'reilly, 2009, p.5,6

PISCITELLI, A.; *CIBERCULTURAS 2.0, en la era de las máquinas inteligentes.* Ed. Paidós contextos, Barcelona, 2002 , p.21

RANDELL, B. *Digital Computers: Origins*. Documento PDF, [Consulta:05/03/2011],Url:http://www.cs.ncl.ac.uk/publications/books/papers/128.pdf

RECK, E.Y OTROS. *Música y nuevas tecnologías:Perspectivas para el siglo XXI.* Associació de Cultura Contemporanea L´Angelot, Barcelona, 1999.

RÖSSLER, O. E.; SCHMIDT, A. P.;WEIBEL, P. *El mundo como interfaz,* en AA.VV., Dinámicas Fluidas. I Festival internacional de arte, ciencia y tecnología, Ayuntamiento de Madrid, Madrid, 2002, pp. 59-64.

RUSSOLO, L. *El arte de los ruidos.* Centro de Creación Experimental, Taller de Ediciones Universidad de Castilla-La Mancha, Cuenca, 1998.

SHARP, H ROGERS, Y , PREECE, J, *Interaction Design:* Beyond Human Computer Interaction , John Wiley& Sons, 2007.

VVAA., *Bang/ Pure data* (1. International PD-Convention Graz), Wolke-verlag, 2006