# Resumen

Este proyecto de fin de carrera tiene como objetivo la creación de una infraestructura tanto física como de software que facilite y economice la investigación robótica sobre el aprendizaje cognitivo. Además, del desarrollo de herramientas de software que simplifiquen el uso de sonido en investigación robótica.

Para ello, se ha diseñado y construido un robot. La estructura del mismo ha sido elaborada usando software de diseños 3D para poder ser generada a través de impresoras 3D. El fin de esta decisión recae en poder facilitar su reproducción por todos aquellos que así lo deseen. Por otro lado, el hardware usado es el más económico posible a excepción de aquellos componentes que tienen un especial interés para su investigación como posible hardware dentro del mundo de la robótica.

El diseño en general está inspirado en las teorías de robótica social y cognitiva, siguiendo a su vez el principio de economía en el diseño. A nivel de software, se ha desarrollado una serie de herramientas que permiten simular entornos virtuales sonoros y procesar los sonidos generados internamente por una computadora. Además, se ha creado un software para el aprendizaje robótico no supervisado, basándose en modelos matemáticos que se aproximan al funcionamiento cerebral del ser humano. Por último, se ha realizado un experimento a modo de entrenamiento de la estructura de aprendizaje para evaluar si era posible con la infraestructura software creada, y como sistema de evaluación el de la efectividad de las teorías aplicadas, poder reconocer el tono emocional en el que alguien está hablando independientemente del idioma usado.

Como resultado, se ha obtenido un robot completo con los elementos esenciales que permiten la investigación en aprendizaje robótico en robots físicos. Siendo además este robot de un precio de fabricación bajo y modular. Asimismo, en la parte de software se han extraído un conjunto de gráficas que muestran el rendimiento y la eficiencia de los modelos y software creados.

Como conclusión, se puede subrayar que se ha conseguido un sistema con el potencial de detectar los tonos emocionales de los hablantes con una exactitud notable aunque sería necesario un mayor desarrollo y optimización para una mejor precisión .

# Contenido

Resumen ..... 1

1.1. Motivación ..... 9

1.2. Capacidad de aprendizaje ..... 9

1.3. Aprendizaje autónomo ..... 10

2. Introducción ..... 13

2.1. Objetivos del proyecto ..... 13

2.2. Alcance del proyecto ..... 13

4. Núcleo del Proyecto ..... 15

4.1. Hardware: Plataforma Robótica Experimental ..... 15

4.1.1. Filosofía ..... 15

4.1.2. Inspiración y Requerimientos de Diseño ..... 17

4.1.3. Social Robotics e Influencia en el Diseño ..... 22

4.1.4. Diseño por Desglose Sensorial ..... 25

4.1.5. Detalles de Construcción ..... 32

4.1.6. Detalles de control de Hardware ..... 35

4.1.7. Resultados de Hardware ..... 35

4.1.8. Conclusiones del Hardware y futuro trabajo ..... 35

4.2. Software ..... 37

4.2.1. Simulación Sonora ..... 37

4.2.2. Adquisición de sonido interno ..... 39

4.2.3. HTM ..... 46

4.3. Estudio Experimental ..... 64

4.3.1. Datos de Estudio ..... 64

4.3.2. Experimento Procedimiento ..... 66

4.3.3. Resultados Experimentales ..... 67

Conclusiones Experimentales ..... 80

4.3.4. ..... 80

4.3.5. Trabajo futuro Experimental y Software ..... 83

5. Análisis Económico ..... 85

5.1. Gastos en Software ..... 85

5.2. Gastos Hardware ..... 86

5.2.1. Componentes ..... 86

5.2.2. Estructura ..... 87

5.2.3. Sueldo Operario: ..... 89

5.2.4. Coste Global HARDWARE ..... 89

5.3. Coste Global y Análisis ..... 89

6. Estudio Medioambiental ..... 91

6.1. Software ..... 91

6.2. Hardware ..... 91

6.2.1. Estructura impresa en 3D ..... 91

6.2.2. Electrónica ..... 93

6.2.3. Baterías ..... 93

7. Agradecimientos ..... 95

Bibliografía ..... 97

# Glosario

***C***

compliant

Normalmente referido a actuadores, son aquellos que permiten desviaciones de su posicion de equilibrio en base a la fuerza aplicada externa.....9, 24

***E***

*embodied*

*Se refiere a embodied cognition, que consiste en una teoria que dice que muchos de las capacidades cognitivas humanas estan condicionadas por la configuracion y forma del cuerpo físico.* .....24, 27

***F***

FFT

Transformada Rapida de Fourier, un metodo para transformar señales desde el ambito temporal al frecuencial ...37, 39, 50

***H***

HRI

Estudio de las interacciones entre robots y humanos. Se refiere a todos los elementos que permitan una mejor comunicacion. . 16, 18, 19, 20, 22, 24, 26, 28, 62, 88

***P***

par Nominal

Es el par que se produce en un motor para que pueda desarrollar sus condiciones de diseño .....25

PPS

Empresa dedicada a la creacion de sensores tactiles de altas prestaciones 26

***R***

ROS

Robot Operating System, es una infraestructura de software que simplifica el trabajo con robots.... 12, 33, 35, 36, 37, 38, 51, 56, 59, 62, 64, 81

***S***

scikit-learn

herramientas python para analisis datos ..12

SDR

Sparse Distributed Representation, es una representacion de datos basado en lo que se cree que hace el cerebro humano para almacenar informacion.....38, 39, 40, 41, 42, 43, 50, 52

STL

Es un formato de archivo informático de diseño asistido por computadora (CAD) que define geometría de objetos 3D.....16

***V***

velostato

Material de empaquetado de polyolefines impregnado con "carbon black" para hacerlo conductor. Cambia su resistividad con presion o flexión. .....26

# Prefacio

## 1.1. Motivación

En los últimos diez años, el sector de la robótica de servicios ha crecido de manera espectacular, tanto en la cantidad de robots disponibles como en el número de empresas repartidas por todo el mundo dedicadas exclusivamente a ello (1) .

Sin embargo, solo un producto a conseguido implantarse en la sociedad y tiene cierta utilidad real: el robot aspirador. El resto de robots disponibles carecen de esta verdadera utilidad y masiva de la que sí disfrutan los ordenadores o los smartphones hoy en día.

Todas las barreras técnicas que hacían inviable la robótica de servicios hace tan sólo diez años, como la duración de las baterías, la potencia de procesado móvil o hardware asequible; se han superado. Se podría decir que actualmente tan sólo sigue persistiendo una única barrera que impide la explosión del mercado de la robótica de servicios: **la capacidad de aprendizaje**.

## 1.2. Capacidad de aprendizaje

Para que un robot sea comercialmente viable en el mundo real, debe estar listo nada más salir de la caja. Eso lleva implícito que el robot tiene que ser capaz de aprender cualquier cosa que no venga de serie por sí solo.

Por tanto, para que un robot sea capaz de todo ello, debe disponer de unas capacidades cognitivas muy ricas. Como en cualquier mamífero, el conocimiento complejo está asentado en capas de conceptos y habilidades progresivamente más complejos. El funcionamiento se asemejaría al de leer un libro. Para entender la historia que transcurre en el libro y los sentimientos de los personajes hay que primero entender las letras, las palabras, las frases y finalmente las ideas.

La base de toda esta capacidad cognitiva es el auto aprendizaje de patrones del mundo real sin supervisión. Un niño en relativamente poco tiempo es capaz de aprender que un circulo y un cuadrado son formas geométricas diferentes, aunque no sepa como se llaman. Es

capaz de reconocer la diferencia entre el tacto de una manzana y el de una naranja, o de saber si la persona que le habla está siendo agresiva o amable con ellos. Y nadie se lo ha enseñado, lo han aprendido por ellos mismos.

Por tanto, es imperativo desarrollar sistemas que sean **capaces de aprender los conceptos subyacentes en el mundo real por si solos**. Esto a su vez ayudara a asentar las bases para un aprendizaje más complejo.

## 1.3. Aprendizaje autónomo

Uno de los primeros sentidos que permite la introducción de información en el cerebro humano y por tanto aprender, es el oído. Esto permite al feto empezar a aprender sobre el mundo que le rodea desde el interior del vientre materno (2). Por lo tanto, los conocimientos sonoros deben formar parte del conocimiento básico del mundo de todos los seres humanos desde el principio.

Este conocimiento se adquiere de manera no supervisada (3). La diferenciación entre un sonido y otro se hace a través de unos mecanismos naturales del cerebro que permiten desentrañar los patrones que subyacen en un mundo de caos sensorial. La información que entra a través de nuestros oídos está repleta de ruido. Un ejemplo de ello es la dificultad que los robots tienen para mantener un buen rendimiento en condiciones sonoras variables.

La percepción armónica del mundo se da gracias a que el cerebro es capaz de extraer de estos datos de entrada conceptos abstractos (3). Un ejemplo de ello sería cuando nos acercamos a una fuente de sonido. A medida que nos movemos hacia esta fuente, los datos sonoros de entrada van variando. Los patrones sonoros se modifican en función del ángulo o del ruido ambiental, al igual que por cambios momentáneos en el entorno. Sin embargo, la percepción es que la fuente de sonido es la misma sin importar el ángulo en que se oiga o si esta puesta la TV o no.

Para que la robótica de servicios sea una realidad, los robots deberán disponer de todo el conocimiento básico necesario para ser de utilidad, ya que no se puede vender un robot que tenga las capacidades cognitivas de un niño recién nacido. Esto exige la necesidad de la creación de todos esos conocimientos en un periodo de tiempo reducido.

Todo ese conocimiento básico se adquiere a lo largo de años de datos. Es necesario por tanto disponer de una infraestructura que permita **generar este conocimiento lo más rápido y eficientemente posible.**

# 2. Introducción

## 2.1. Objetivos del proyecto

El presente proyecto tiene como objetivo global crear una infraestructura que facilite y acelere el desarrollo de robots con mejores capacidades cognitivas. Éste a su vez puede desglosarse en los siguientes sub-objetivos:

- Crear una **infraestructura hardware** asequible que sirva como punto de partida para el desarrollo de una comunidad destinada al aprendizaje robótico. Así como que disponga de los elementos esenciales para cubrir todo el rango sensorial básico de los seres humanos.
- Crear una **infraestructura software** que facilite el aprendizaje robótico usando ROS, python, NUPIC, GSound y scikit-learn .
- Estudiar la posibilidad de usar la infraestructura creada para **identificar patrones sonoros**.

## 2.2. Alcance del proyecto

Queda dividido según cada sub-objetivo.

- **Hardware:** Constará de todos los archivos digitales necesarios para la edición de los mismos en SolidWorks 2007 o posterior. Los planos estarán destinados a la ayuda en el montaje del dispositivo. Se construirá una versión física con todo el hardware testeado pero sin una integración unificada. No se realizarán ensayos mecánicos sobre el dispositivo ni se realizarán pruebas de seguridad normalizadas. El robot recibirá alimentación de la red.

- **Software:** El software estará centrado en su uso en reconocimiento de patrones sonoros y trabajo con sonido. Sólo testeado en Ubuntu-SO 14. No se creará ningún instalable y por tanto, al instalación del software se realizará manualmente.

- **Estudio:** El estudio se centrará en el reconocimiento de entonaciones sobre personas hablando: inglés, alemán e italiano. La Tasa de reconocimiento se basará en gráficos de aciertos medios. No se testeará en entornos reales.

# 4. Núcleo del Proyecto

Siguiendo la división hecha en el capítulo de Introducción, el proyecto se ha dividido en tres partes : **Hardware**, **Software** y **Estudio**. En la sección de Hardware se trata todo lo referente a las decisiones del diseño y fabricación de la plataforma física. En la sección de Software se desarrolla todos los paquetes informáticos para realizar las pruebas de **Estudio** posteriores y de los programas de control del hardware. Por último, en la sección de **Estudio** se plantea el experimento de aprendizaje con sonido de entonaciones en el habla humana.

## 4.1. Hardware: Plataforma Robótica Experimental

### 4.1.1. Filosofía

El diseño del robot descrito en este proyecto se sostiene sobre cuatro pilares conceptuales básicos: Economía, Aprendizaje, I+D en Robótica y Pragmatismo.

- **Economía**: Los costes de fabricación y consumo energético deben ser lo mínimo posible.
- **Aprendizaje**: Todo el diseño debe favorecer la enseñanza y aprendizaje del robot.
- **I+D en Robótica**: Se ha de usar tecnologías poco usadas en la robótica para explorar sobre alternativas para la robótica actual.
- **Pragmatismo**: Tanto el montaje como el uso del hardware tiene que ser lo más fácil posible para fomentar su uso.

**ECONOMIA:**

El coste de fabricación del robot es un factor determinante. Se han usado los accionamientos más asequibles y pequeños posibles, ya que los accionamientos son un apartado costoso en cualquier proyecto de construcción robótica. Al seleccionar accionamientos pequeños, se reducen los costes y la potencia necesaria, reduciendo así también el coste de la electrónica de control y el consumo energético. A su vez, un consumo

energético reducido repercute en los costes de uso del robot y la viabilidad del uso de baterías.

La economía también condiciona el posicionamiento de los elementos ya que se intenta reducir las inercias en la medida de lo posible, ya que con menores inercias el consumo por el movimiento se ve disminuido a la vez que otros efectos negativos como la inestabilidad. Un factor relacionado es el peso. Tomar decisiones que reduzcan la cantidad de material usado y las dimensiones del robot, reducen los costes de fabricación además de facilitar la portabilidad y mantenimiento.

Un último factor relacionado con la economía es también diseñar estructuras auto soportadas, refiriéndose a que sin ningún tipo de aporte energético, el robot se sostenga por sí solo. Esto reduce el consumo energético en reposo y el desgaste por fatiga de los accionamientos, hecho fundamental ya que los accionamientos son piezas con costes altos.

La economía de diseño es simplicidad, un elemento que reduce la probabilidad de error, el tiempo de fabricación y facilita el cambio de componentes si se requiere.

**APRENDIZAJE:**

Todo el hardware en el robot debe estar orientado a facilitar el aprendizaje por parte del robot y de la enseñanza por parte del tutor humano (4).

Por el lado del aprendizaje robótico, debe disponer de los sentidos cognitivos esenciales para el aprendizaje. Estos son: la visión, el oído, el tacto y la manipulación o capacidad Motriz.

Por el otro lado, el tutor humano debe ser capaz de comunicarse de manera intuitiva con el robot y sentirse cómodo haciéndolo. Esto condiciona aspectos como la necesidad de disponer de algún tipo de comunicación inteligible hacia los seres humanos sin experiencia en robótica o programación.

**I+D en ROBÓTICA:**

Como en cualquier sector en crecimiento, es importante investigar nuevos elementos que simplifiquen, reduzcan costes o aporten nuevas funcionalidades al sector. Por tanto, si es necesario se colocarán elementos que transgredan la filosofía de economía, ya que aportan posibilidades nuevas.

**PRAGMATISMO:**

No se debe olvidar que toda tecnología debe ser lo más simple posible para extender su uso (5). El robot deberá ser por tanto fácil de montar, fácil de usar respecto al hardware y fácil de intercambiar aquellos elementos dañados. Este último punto cobra especial importancia por ser un robot de experimentación que exige mucha robustez frente a choques, fallos mecánicos, situaciones extremas para el hardware o simplemente flexibilidad a la hora de modificar el diseño y añadir nuevos módulos. También la propia fabricación de las piezas debe ser asequible, ya que procesos de fabricación complejos y lentos aumentan los costes y disminuyen la probabilidad de que se lleguen a usar. Para cumplir el objetivo de sentar las bases de una infraestructura robótica comunitaria, es fundamental que su fabricación esté al alcance de todo el mundo sin sacrificar funcionalidad. Por tanto la impresión 3D es la mejor opción. Con ella se consiguen piezas con tolerancias aceptables para el ensamblado de estructuras complejas sin necesidad de conocimientos en fabricación. Otro factor determinante es la facilidad de compartir y modificar los diseños, favoreciendo una comunidad que modifique y mejore el sistema en base a sus necesidades.

### 4.1.2. Inspiración y Requerimientos de Diseño

Una vez planteados los principios del diseño, se pasa a exponer ejemplos similares que se han usado como inspiración para el diseño del robot. Estos se han seleccionado en base al cumplimiento de algunos de los principios filosóficos introducidos en el capitulo anterior. Los robots seleccionados son: iCub, Maki, CB2, Kismet/Leo/Nexi and Domo

- **iCub:** Es un proyecto opensource diseñado por RobotCub Consortium, universidades Europeas y construido por la Universidad IIT (Italian Institute of Technology) bajo la licencia GPL. Esto implica que todos los planos del hardware y el software están accesibles gratuitamente. Su diseño de basa en la hipótesis **embodied cognition**, que la manipulación e interacción de forma humana juega un papel primario en el desarrollo de las habilidades cognitivas de todo ser humano desde su nacimiento. Su coste promedio de adquisición son 250.000 euros.

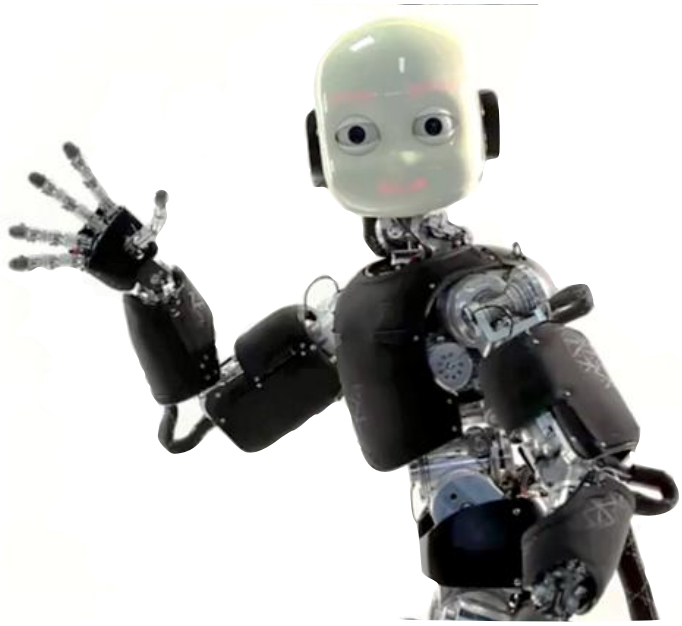


Figura 4-1: iCub

- **Maki:** Es un robot desarrollado por Hellorobo. Diseñado con el objetivo de hacer la robótica HRI (Human Robot Interaction) asequible para el público en general. Los planos de todo el diseño y lista de materiales están disponibles de forma gratuita. Es un robot extremadamente sencillo de fabricar ya que se dispone de los archivos STL necesarios para la impresión directa. Su coste de fabricación es de aproximadamente 300 euros.

Figura 4-3: Maki

- **CB2:** Es una plataforma para la investigación en develomental robotics creada para el Jst Erato Asada Project del JOU (Japan's Osaka University) . Está diseñado para imitar la interacción que hay entre un bebe y sus padres en los primeros meses de vida mientras generan la imagen de sus propios cuerpos. Su característica más destacable es que dispone de piel artificial alrededor de todo el cuerpo, permitiendo una interacción motora única con el mundo.

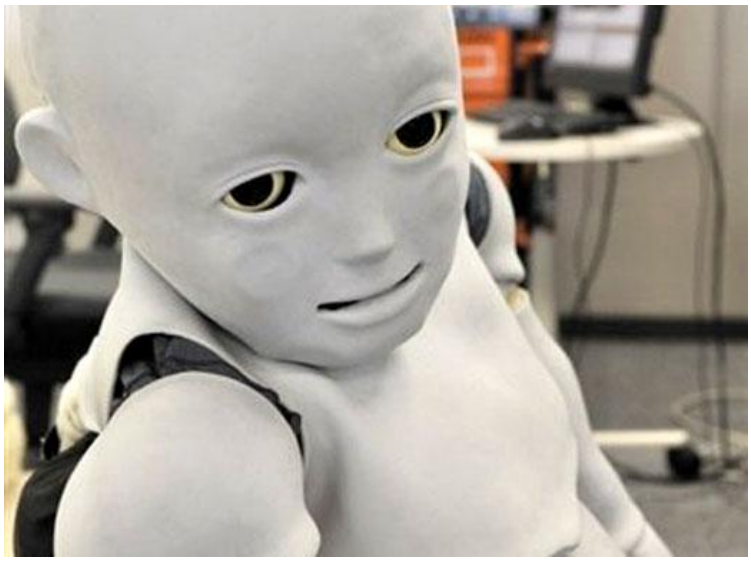


Figura 4-4: Cb2

- **Kismet/Leo:** Estos tres robots fueron creados en el por el Personal Robots Group. Son robots diseñados para la mejora en la HRI y para la robótica social.

  Kismet fue uno de los primeros robots sociales de la historia y capaz de detectar y reaccionar ante entonaciones diferentes como enfado o alegría (4). Leo es una mejora sobre el trabajo ya realizado sobre Kismet, sobretodo en el aspecto de interacción y con un aspecto más de criatura real. Ninguno se puede adquirir de forma directa.

Figura 4-5: Leo

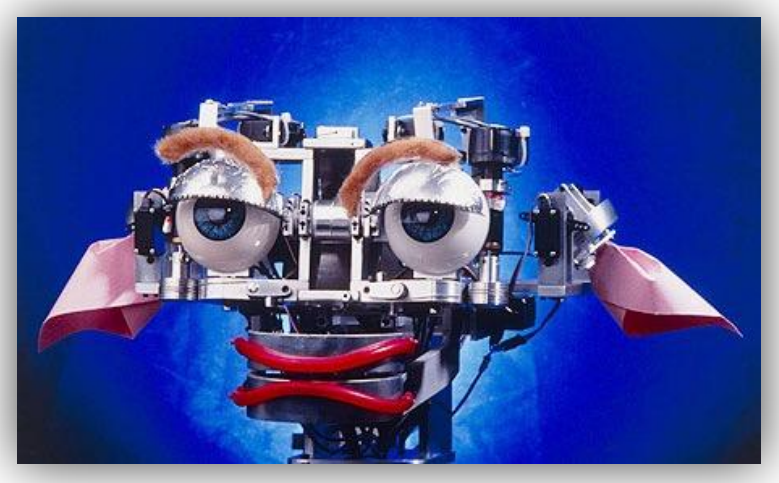


Figura 4-6: Kismet

Tanto Leo como CB2 son robots relativamente antiguos ya que son de alrededor del año 2007-2008, mientras que Kismet es de 1990. Sin embargo, siguen siendo relevantes en sus respectivos campos por haber indagado en la HRI. De CB2 se extrae la necesidad de disponer de algún tipo de sistema táctil sensorial. De Kismet/Leo se ve la necesidad de disponer de métodos de comunicación que faciliten el aprendizaje y el tutelado con seres humanos no técnicos.

En cuanto a iCub y Maki, el proyecto está fuertemente inspirado en ellos. De Maki se extrae la filosofía opensource fácilmente accesible, la fabricación económica en impresión 3D i portabilidad. De iCub destaca el funcionamiento por módulos y el potencial de investigación a través de software ya preparado.

Por tanto de estos ejemplos se extraen las primeras solicitaciones de diseño, tanto de atributos que tienen los robots como las carencias. Los requerimientos son:

- **Diseños y software Opensource**: Debido a la necesidad de crear una comunidad grande que investigue sobre aspectos diversos del aprendizaje robótico y contribuya, es indispensable que el proyecto en su totalidad esté al alcance de cualquier persona que lo necesite sin ningún tipo de condición.

- **Fabricación en 3D y económica**: Un rasgo común en casi todos los robots de hoy en día es que sus precios son muy elevados e inasequibles para el público en general. Por el contrario, con Maki por 300 euros se consigue un robot. La desventaja de ello es que no se dispone casi ningún sistema sensorial. También si observamos su diseño, éste contiene muchos elementos decorativos o elementos poco robustos para ser fabricados en 3D como engranajes en ángulo recto.

- **Modular**: El robot debe permitir el uso de solo los módulos que interesen al usuario y no obligar a lidiar con sistemas complejos sin necesidad. Debe poderse modificar y ampliar. También reemplazar módulos dañados o deteriorados por un mínimo coste.

- **Simplicidad**: Un diseño simple reduce costes y elimina al usuario la obligación de lidiar con sistemas complejos que no necesita. También reduce costes y dificultad en el diseño.

- **Disponer de Sentidos Básicos**: Debe disponer de todos los sentidos básicos pero sin sobrecargar el sistema de redundancias que encarezcan y compliquen el sistema sin necesidad. Los sentidos deben ser: Visión, oído, tacto/capacidad motora.

- **Sistema de expresión**: Sin un sistema de comunicación compatible con el del ser humano, es muy difícil enseñar a un robot de forma intuitiva. Incluso para usuarios experimentados resulta más cómodo una manera visual y natural de comunicar que los conceptos se entienden o que hay algún error en sistema de aprendizaje.

- **Diseño orientado al Embodied Cognition**: En el caso de este proyecto, implica que en el diseño se debe tener en cuenta que el robot estará interactuando continuamente con su entorno, por tanto se dañara o deteriorará. También deberá reducirse al máximo sus dimensiones y la fuerza de los actuadores para evitar daños en el robot y en los elementos con los que interactúe. Podríamos asemejar los principios de diseño a los de un bebe al nacer, el cual es flexible para no dañarse en las caídas y con una fuerza reducida para no poder hacerse daño con sus interacciones en el mundo y a sus padres cuando les coja de la nariz.

### 4.1.3. Social Robotics e Influencia en el Diseño

De inspiración directa de Kismet/Leo y del libro de “Designing Sociable Robots” (4) se extrae la idea siguiente: un robot que aprende de seres humanos debe disponer de mecanismos que permitan la HRI. Los mecanismos primarios de comunicación en los seres humanos son: el habla, los gestos y las expresiones faciales. Este último es el método con el que se empieza a comunicar con los seres humanos recién nacidos, ya que el habla y los gestos necesitan de un aprendizaje y desarrollo cognitivo mucho más avanzado y aparecen en etapas mucho más tardías del desarrollo.

Basándose en el FACS (The Facial Action Coding System) (6) desarrollado por Carl-Herman Hjortsjö, se extrae que para una correcta expresión facial se necesitan los siguientes elementos: cejas, párpados superiores, párpados inferiores, nariz y boca.

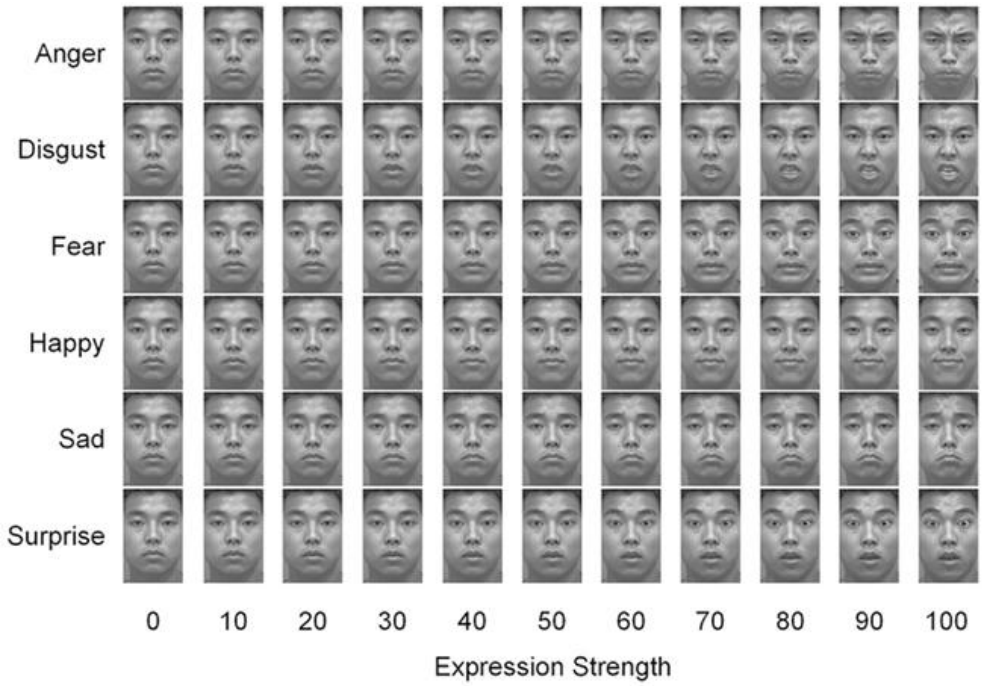



Figura 4-7: FACS Fuerza en la expresion

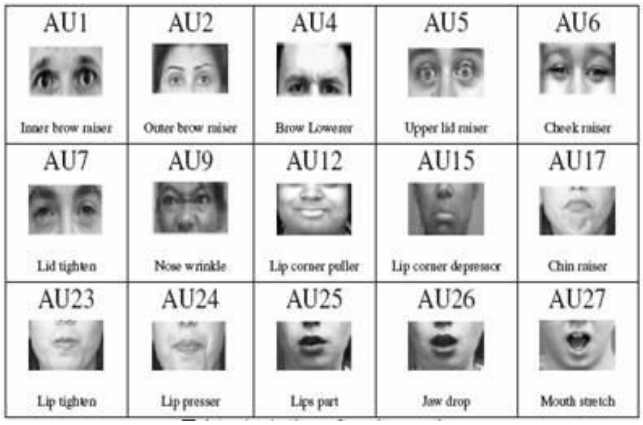



Figura 4-8: FACS muestra de clasificación de microexpresiones

Muchos de estos elementos están incluidos en Kismet y Leo (4) . Sin embargo, para cumplir los requerimientos de diseño, en especial los de Simplicidad y Economía, no es posible incluir todos estos elementos. Por tanto, se debe extraer la esencia de esta interacción y concentrarla para simplificarlo al máximo hasta no poder eliminar ningún elemento sin ya perder el significado (7).

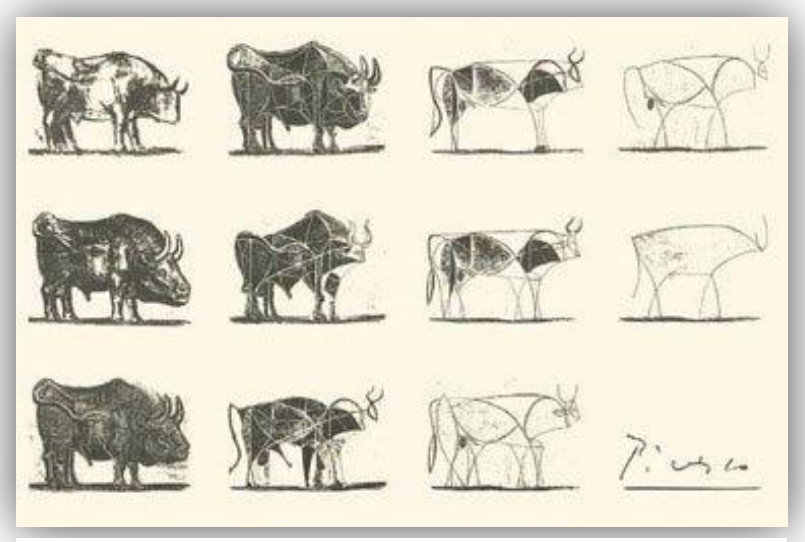


Figura 4-9: Deconstruccion del Toro de Picaso

Un elemento que consigue expresar todo lo necesario para establecer no solo una comunicación por expresión facial sino posteriormente sistemas de gestos o ideas abstractas es: una pantalla. Elimina la necesidad de sistemas mecánicos que necesitan multitud de accionamientos y energía, a la vez que complican el diseño, añaden puntos donde puede fallar y peso adicional . También es un sistema altamente flexible ya que se puede dar una infinidad de usos. Sin embargo, uno de los inconvenientes de las pantallas tradicionales es que consumen energía incluso cuando están mostrando la misma información. Esto en el sistema mecánico no pasa, ya que una vez movido el sistema para mostrar un gesto, este se puede quedar así con un consumo mucho menor de energía. Por tanto, era necesario encontrar una pantalla que tuviera un consumo mínimo una vez mostrada la información y que mantuviese esa información hasta el próximo cambio. Las pantallas de e-ink resuelven ese inconveniente. Con un consumo pasivo nulo y sólo consumo cuando se cambia de estado, encaja en los requerimientos. El defecto de las e-ink que es su relativa lenta velocidad de cambio de estado, juega a favor, ya que para una agradable HRI se sabe que las transiciones de un estado a otro no deben ser muy rápidas.

En conclusión se escoge sublimar todo el sistema de expresión y comunicación a una pantalla de e-ink de bajo coste situada en el centro del robot.

### 4.1.4. Diseño por Desglose Sensorial

A continuación se muestran las distintas decisiones de diseño agrupadas por regiones corporales. Todos los archivos digitales de diseño se encuentran en el soporte digital en **"kodama/diseños".**

#### 4.1.4.1. Oídos

El oído humano tiene tres partes diferenciadas: pinna, oído interno y tambor.


**Figura 4-10: Diagrama de Oído**

Tanto el oído interno como el tambor, en la versión artificial se resume en el micrófono y en realizar la FFTs al sonido. Es la pinna que tiene relevancia en la fase de diseño. La pinna afecta al sonido entrante modificándolo según el ángulo de procedencia y amplificándolo o reduciéndolo según la frecuencia. El diseño se ha centrado en el cambio de un mismo sonido en función de la procedencia. Esta deformación del sonido viene dada por la asimetría en la estructura de la pinna. Sonidos procedentes de delante no se encuentran con nada, mientras que los procedentes de atrás tienen que atravesar la pinna, deformando el sonido. Lo mismo que en el caso de sonidos procedentes de arriba o de abajo, al tener una forma asimétrica se modifican ligeramente, dotando al ser humano de la capacidad de saber de dónde viene una amenaza en segundos de manera rudimentaria. Un estudio sobre esto se puede ver en el trabajo realizado por KAIST de localización de la elevación del sonido en un oído artificial (8).

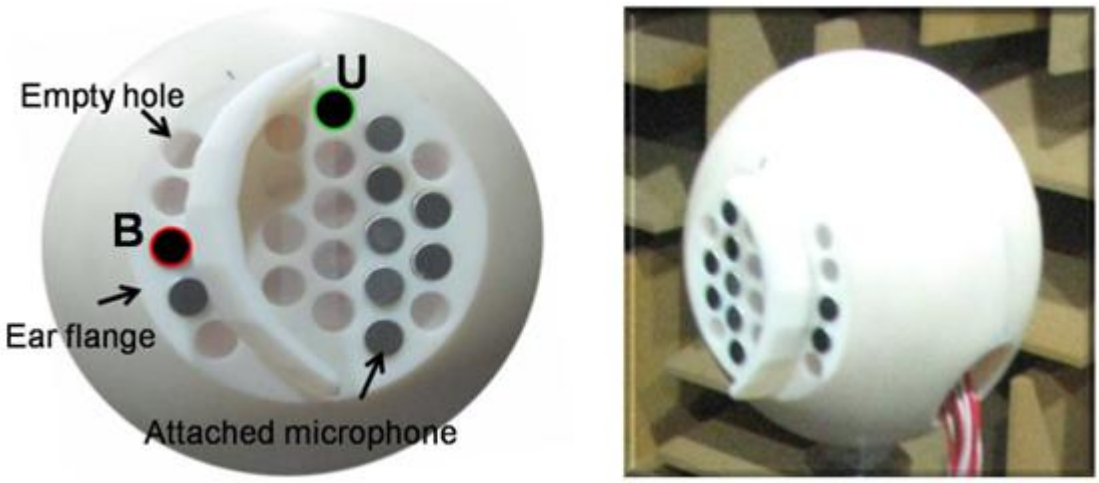



Figura 4-11: KAIST oido artificial

#### 4.1.4.2. Visión

Para el aprendizaje *embodied* es esencial manipular objetos, y para ello, el conocimiento de profundidad es esencial. Por tanto, la visión estereoscópica es necesaria (9). Al necesitar ser lo más económico posible, se han seleccionado cámaras “low-cost USB CMOS board camera module”. Son cámaras con foco graduable manualmente, entrada USB compatible con protocolos estándar y sin carcasa. El foco graduable es adecuado por poder simular las condiciones de visión en bebes en los que su foco visual está delimitado al entorno inmediato. El uso de protocolos estándar facilita su integración y uso. La no existencia de una carcasa reduce al mínimo el peso.

Las dos cámaras deben estar montadas en un sistema de accionamiento preciso y silencioso. Preciso por la solicitaciones que tiene la visión estereoscópica, ya que ha de ser capaz de centrar el foco en puntos concretos. Por otro lado, el aspecto silencioso es un requerimiento marcado por la filosofía de Social Robotics (4), ya que se deben intentar reducir al mínimo, en la medida de lo posible, los ruidos de tipo mecánico para mejorar la HRI. También, un factor determinante en la selección de los accionamientos ha sido el descubrimiento de los PCBMotors (10) y en concreto, el de un modulo de experimentación llamado TwinMotor. Este modulo consiste en una pareja de motores de tipo PCB dispuestos de manera que la distancia entre ellos es similar a la distancia entre los ojos humanos. Los motores PCB son accionamientos basados en vibración de semiconductores, al contrario que los motores tradicionales que basan su funcionamiento en bobinado y campos magnéticos. Esta tecnología tiene gran potencial, ya que producido en masa es mucho más rentable y simple que los motores convencionales, son totalmente escalables y “compliant".

Este último aspecto es de vital importancia en robótica ya que los motores en situaciones de trabado aumentan sustancialmente su consumo o pueden llegar a dañarse. Por el contrario, los PCBmotors simplemente no avanzan. El factor más importante es su reducido consumo, siendo nulo en estado de reposo. Una gran fracción del consumo en los robots de hoy en día es mantener la posición, y siguiendo el requerimiento de economía, unos motores sin consumo en reposo son una tecnología digna de uso.

Sin embargo, las desventaja sobre otros sistemas son que por un lado tienen un precio mucho mayor que los accionamientos estándar y por el otro lado, su par Nominal es reducido en comparación con accionamientos del mismo tamaño. El coste es asumible por la posibilidad de investigar sobre una tecnología aplicable en la robótica que reduciría costes de fabricación una vez fabricado en masa. Sin embargo, la restricción en par condiciona los elementos que pueden mover. Esto implica que se debían realizar pruebas de mover las cámaras. Las cámaras tienen un peso de 4,5 gramos, totalmente asumible. Sin embargo además del peso de las cámaras, el PCB motor debe vencer el par resistente generado por el cable USB, y este no es insignificante. De forma que esta restricción ha obligado a seleccionar unas cámaras ligeras y sin cable. Las únicas cámaras relativamente asequibles, wireless, de uso simple y ligeras para poder cumplir estas restricciones son las Ai-Ball de TREK. Con un peso de 100 gramos consiguen estar dentro del rango de funcionamiento de los PCBmotors. También por las restricciones de peso ha eliminado cualquier soporte y tornillería para la sujeción de las cámaras al PCBmotors y se decidió usar cinta de doble cara para ese propósito. El soporte consiste en una hendidura para permitir cierta regulación a la hora de colocar las cámaras en el soporte .Todo el montaje esta ilustrado en el ANEXO B.

Una vez decidido los componentes, se diseño la estructura y mecanismo de movimiento. En primer lugar para conseguir una visión estereoscópica mínima, se necesita poder inclinar el ángulo de visión para poder llegar a cualquier región de estudio. Por tanto, se necesita poder inclinar la estructura principal. La manera más eficiente de hacerlo es mediante un accionamiento lateral que rote la estructura lo más cerca posible del centro de masas para evitar inercias, pero desviándose lo menos posible del centro de las cámaras para que la rotación no genere una traslación significativa de las mismas. Por tanto, se centra el eje aproximadamente en el centro de las cámaras. Los ejes descansan sobre rodamientos cerrados que reducen la fricción al mínimo y evitan el desgaste que se produciría si se

hubiesen colocado directamente los ejes sobre los soportes. Y todo ello repercute positivamente en el gasto de energía.

En cuanto a la estructura frontal sirve para dos propósitos: el evitar la flexión de las piezas de los ejes dando integridad a la estructura y como elemento personalizador del robot. Darle cara a un robot por mínima que sea, mejora significativamente su HRI. Además la nariz en el centro cumple también una función de refuerzo de la misma estructura al disponer en esa región de menos material debido a que debe introducirse un tornillo de sujeción ( Anexo E ).

#### 4.1.4.3. Tacto

El tacto es un sentido que generalmente supone un gasto elevado, además de complejo a la hora de integrar en sistemas robóticos. Las opciones económicas son los sensores de presión individuales que no permiten ningún tipo de customización y con rangos determinados de funcionamiento. Por otro lado, las superficies multitouch flexibles como los que se pueden encontrar en PPS (11) rondan los 1.200 euros y sin posibilidad de decidir qué forma se desea. Por tanto, se investigó sobre otras tecnologías que pudiesen servir, encontrándose una compañía llamada PlugAndWear, la cual se dedica a la creación de electrónica en ropa. Un sensor en concreto basado en un principio resistivo de un material llamado velostato que permite crear sensores táctiles extremadamente robustos y customizados a un precio de alrededor de 10 euros. Éstos disponen de un módulo ya fabricado de un solo punto de contacto que se ha considerado adecuado para una primera versión.

Figura 4-12: Muestra de Sensor táctil

Una vez seleccionado el sensor, se debe decidir la estructura táctil. Hay varias opciones de complejidad decreciente: un brazo completo, un brazo con pinza, un brazo con terminación táctil y un por último un dedo táctil.

Los factores determinantes para decidir el diseño final son: simplicidad, economía y poder realizar una interacción táctil mínima. Todas las opciones con un brazo exigen un mínimo de cinco actuadores -dos hombros, dos codos y una muñeca-, elevándose el número hasta diez en brazos más completos. Esto exigiría el uso de un gran número de servos, lo que contribuiría a una mayor complejidad en el dispositivo, en su peso y por consiguiente, en el consumo energético. Por tanto, se ha tomado la decisión usar un dedo táctil. Su diseño está basado en las proporciones de los dedos humanos, tanto en su disposición como su curvatura. La curvatura está diseñada específicamente para facilitar el contacto sobre objetos dispuestos delante del robot. Dispone de un solo accionamiento que permite subir y bajar, estando éste apoyado en un rodamiento radial. Por otro lado, el movimiento de rotación se lo proporciona la misma rotación del cuerpo róbotico. Con el dedo táctil se consigue tener un robot con capacidad para iniciar el aprendizaje de objetos con el sentido táctil básico como base.

Otra cuestión a tener en cuenta, es dónde colocar este dedo. El objetivo del robot es facilitar la investigación en robótica cognitiva y embodied cognition. Por tanto, el sistema de manipulación tiene que estar en un lugar cercano a la zona donde se vaya a centrar la atención del robot. La mayor parte de los experimentos de manipulación de objetos se realizan con el objeto situado delante del robot en una mesa o similar. De este modo, se ha valorado que ésta es la zona de interés y que por ello, el dedo táctil debe estar en una posición baja, cerca de la zona de trabajo.

Otro factor determinante en la situación del dedo es intentar compactar el diseño al máximo para minimizar el tamaño del robot, sin sacrificar flexibilidad en el caso de ser necesario cambiar el elemento de manipulación, algo bastante común. Esto provoca que el dedo se ponga en el interior de la estructura.

Y un tercer factor y el más importante es evitar la oclusión del los objetos por el dedo. Si se hubiese centrado el dedo justo debajo de los ojos, la posición de manipulación hubiera sido ideal, pero hubiese tapado el centro de la acción al manipular. Por este motivo, se ha optado por descentrarlo sin que quede fuera del rango de visión de las cámaras; de lo contrario se

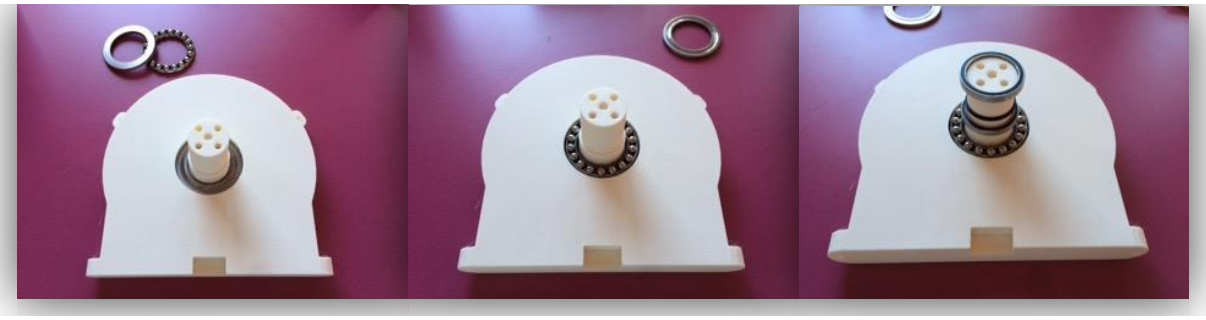
perdería la visión estereoscópica. De esta manera, se deja un espacio libre para que la pantalla quede debajo de los ojos, fundamental para la HRI .

El dedo dispone además de unos orificios para el anclaje de los elementos táctiles que faciliten su sustitución o modificación ( Anexo E ).

#### 4.1.4.4. Accionamiento Cuello

El accionamiento cuello es la articulación que soporta más peso de todo el sistema y que se usará más ya que sirve tanto para el direccionamiento del sistema de audio como para el de visión o el de manipulación. Siguiendo la filosofía de economía, el diseño sigue una linea simple en componentes, tal y cómo se tenía en mente. Consiste en un eje que sale de la base inferior de la que hablaremos en la sección siguiente; un servo al que se une mediante tornillería, y un disco para hacer la adaptación de los brazos del servo a un sistema más normalizado que se usa a lo largo de todo el diseño. A su vez, el servo esta atornillado a un soporte para los mismos el cual se ancla al soporte inferior principal. Este soporte inferior sustenta dos paneles laterales donde se colocan los oídos, la pantalla, el dedo táctil y la cabeza con el sistema de visión. Por tanto, debe ser el elemento más robusto de todo el montaje. De ahí sus dimensiones y paredes de más de un centímetro de grosor. En una primera versión maciza, se encontró que era demasiado pesado y por ello, se optimizó vaciando dos zonas no funcionales.

Otro aspecto importante es el uso de dos rodamientos radiales y uno axial. Debido al peso que tendría que soportar, la fricción era un elemento crítico a considerar, especialmente teniendo en cuenta que se usarían los servos más asequibles y pequeños posibles con un bajo par de arranque y nominal. Así que se ha optado por colocar en primer lugar, un rodamiento axial en el que descansa todo el peso del robot, reduciendo significativamente el par necesario de arranque y de funcionamiento nominal. En segundo lugar, debido a la configuración del eje, era necesario el uso de rodamientos radiales para amortiguar posibles sacudidas y el balanceo de toda la estructura que hubiera hecho rozar los laterales de la estructura sobre la plataforma inferior. Estos dos sistemas ayudan a mantener el consumo energético bajo, permitiendo el uso de servos de menor potencia y prolongar la vida de la articulación del cuello ( Anexo E )..



#### 4.1.4.5. Base Inferior

La base inferior tiene varias funciones:

- Servir de módulo base para colocar otros de forma adicional según sea necesario. Además de poderse colocar otros objetos o bancos de pruebas que se encajan en la hendidura central, atornillándose a los orificios laterales.
- Contener en su interior elementos de hardware, en este caso el altavoz.
- Hacer de base pesada para evitar balanceos o movimientos indeseados.

La base inferior está dividida en dos piezas por cuestiones de fabricación en 3D y flexibilidad a la hora de cambiar la base ( Anexo E ).

#### 4.1.4.6. Soporte Pantalla

Para facilitar el montaje de la pantalla, dos de los tres orificios de atornillado son ranuras. De esta forma, los ligeros errores de fabricación no afectan a la instalación de la pantalla, permitiendo que quede alineada con la estructura fácilmente ( Anexo E ).

#### 4.1.4.7. Módulos para Servos

Para economizar el diseño, facilitar el montaje y su sustitución en caso de rotura; los servos van montados en unas estructuras que permiten su anclaje a la estructura. Además es una solución que permite que los servos accionen directamente a los ejes sin necesidad de engranajes que transmitan el movimiento, los cuales simplemente añadirían costes, reducirían la robustez y producirían perdidas energéticas. Hay que notar que tiene cuatro

puntos de anclaje a la estructura pero en el montaje solo se usan dos. La razón es que la estructura está diseñada para que se tenga la posibilidad de colocar accionamientos más pesados, pero que en el este proyecto no son necesarios, ya que los servos son de par y peso muy reducido ( Anexo E ).

### 4.1.5. Detalles de Construcción

En este apartado se describen generalidades del diseño o componentes que comparten todos los elementos de la estructura.

#### 4.1.5.1. Tornillería

De todas las opciones de tornillería evaluadas, se ha optado por tornillería de bajo coste. Todos los tornillos son de cabezas cilíndricas por ser más económicos y se han evitado las avellanadas como precaución, puesto que el material impreso 3D es muy delicado a las tensiones internas que podrían generar. Las cabezas son rasuradas y planas para que queden a ras, debido a que en ciertas zonas el espacio es muy reducido. Las tuercas son de seguridad para evitar posibles desenrosques por vibraciones, a excepción de los servos por ser tuercas extremadamente pequeñas y por no existir de seguridad en esas dimensiones a un precio razonable. Los tornillos empleados en la estructura impresa, se han roscado usando **insertos de rosca de latón** para quedar firmemente sujetos a la estructura impresa, ya que reparten mejor las presiones y en las pruebas realizadas no se ha apreciado ningún deterioro en la estructura.

La lista de Tornilleria necesaria se puede obtener en el Anexo B - Lista de Tornilleria.

#### 4.1.5.2. Cojinetes

Los rodamientos usados son radiales en todas la articulaciones y uno axial en la base del cuello.

Los radiales son de sección estrecha de acero inoxidable, obturados con protecciones de goma en ambos lados. En los montajes que hay dos rodamientos -como en el caso del dedo táctil y del cuello, ya que pueden sufrir mayores esfuerzos e impactos- se separan mediante una anilla impresa que descansa sobre los anillos interiores evitando el rozamiento entre los rodamientos.

En cuanto al axial, éste es totalmente abierto por cuestiones de disponibilidad y presupuesto.

#### 4.1.5.3. Tolerancias 3D

Para la impresión se ha utilizado una impresora 3D por extrusión de ABS fundido. Esta elección condiciona una serie de puntos que se han de tener en cuenta a la hora de diseñar las piezas:

1) No se pueden hacer piezas con espesores inferiores a 0,1 milímetros.

2) Las diferencias entre el diámetro del agujero y el del eje que se aplique deberán ser los siguientes en función de si se desea apriete, con juego ligero o juego:

$$E_{apriete} = 0{,}05\ mm$$

$$E_{juego-ligero} = 0{,}15\ mm$$

$$E_{juego} = 0{,}25\ mm$$

Se ha aplicado ajuste con apriete en las piezas que debían encajar formando parte de la estructura. Este es el caso de los soportes inferiores de la estructura de los oídos y de los ejes con los rodamientos, el cual hace que los ejes sean solidarios al anillo interior del rodamiento, al mismo tiempo que el anillo exterior del rodamiento también es solidario a la estructura en contacto. También se ha aplicado en los encajes de los dos soportes laterales con la estructura del cuello inferior.

Por otro lado, se ha aplicado ajuste con juego ligero en la parte superior de los oídos y los agujeros pasantes de los tornillos para evitar cualquier posible presión al atornillas, aparte de ser necesario por el uso de tornillos pasantes.

Estas diferencias se extraen de la experimentación sobre la máquina de impresión que se ha utilizado.

También basado en la experiencia, se ha elegido el uso de chaflanes de 45º y de entre 0,2-0,5 milímetros en los agujeros en los que se tenga que introducir algún elemento con apriete con el fin de facilitar la inserción.

Las piezas de gran volumen se han fabricado con estructura interna ligera para reducir el peso y el material usado. Sin embargo, las piezas con mayores solicitaciones mecánicas o simplemente demasiado delgadas se han hecho macizas para garantizar su integridad. Las piezas huecas son la base del cuello y la base inferior. Por el contrario, los soportes laterales o el dedo se han hecho macizos para evitar fallos mecánicos **( Anexo B: Tabla 3 )**.

Otro aspecto es el diseño de las piezas. Todas las piezas se han diseñado con la impresión 3D en mente. Al ser una fabricación por capas superpuestas, hay ciertas estructuras desaconsejadas o imposibles. Se ha evitado perfiles que obligaran a usar material de soporte en la medida de lo posible. Además, se han diseñado las piezas de manera que en la medida de lo posible los orificios quedasen paralelos al eje z de fabricación. Un ejemplo de ello son los soportes laterales que inicialmente se diseñaron unidos a la parte inferior del cuello, pero eso hubiera generado orificios para los ejes y tornillos deformados. Por lo tanto, para evitar esto, se ha tomado la decisión de dividirlos en tres entidades, ya que garantiza la exactitud en los orificios circulares y facilita la expansión y mejora de la estructura según las necesidades.

En cuanto al uso del color del material, fue por disponibilidad de un material u otro en la diferentes fases de producción.

### 4.1.6. Detalles de control de Hardware

Los servos funcionan a 5V y están controlados a través de un Arduino UNO conectado vía USB al ordenador central.

La pantalla funciona también a través de Arduino UNO o similares enviando las imágenes en formato xbm (formato X-BitMap) generados previamente en una computadora.

Los PCBmotors se controlan a través de su propio driver y por comandos ejecutados en el Pc central al que está conectado a través del USB.

Los micrófonos y el altavoz se gestionan a través de un módulo compatible con la infraestructura ALSA llamado U-Control UCA202 que permite el envío de datos de audio a través del conector USB.

Las cámaras modelo Ai-Ball son alimentadas a través de una batería CR2. El envío de las imágenes, que son recibidas en el Pc central, es posible gracias al software de control de canales IP.

Respecto a todo el software necesario, éste se encuentra alojado en el paquete de ROS "kodama_hardware_pkg", el cual se describe en la sección de **software**.

### 4.1.7. Resultados de Hardware

Todo el proceso de montaje se encuentra en el Anexo B, además de listas de componentes e imágenes globales del robot acabado. En Anexo E se encuentran las fotografias del robot real.

### 4.1.8. Conclusiones del Hardware y futuro trabajo

La estructura ha sido impresa en una impresora 3D de alto rendimiento y prestaciones. Esto implica que los usuarios que usen impresoras de peor calidad quizás no tengan tan buen comportamiento a la hora de encajar y montar piezas. Esto se deduce por el hecho de que aún imprimiendo con este dispositivo de alto rendimiento, se han encontrado algunas dificultades a la hora del montaje, aunque ligeras y sin efectos funcionales. En el futuros se

deberían ajustar los márgenes entre piezas para garantizar un montaje de mejor calidad. Aun así, el comportamiento de la las piezas impresas en ABS es excepcional, incluso al punto de valorar seriamente la posibilidad de vender un producto construido de esta manera.

Los PCBmotors tienen un rendimiento muy ajustado a las tolerancias de peso, lo que supone que ocasionalmente no tengan un buen comportamiento. Por lo tanto, es necesario un mejor algoritmo de control y unos dispositivos de senado de posición mejorados para evitarlo. Aun así, es un tecnología muy prometedora ya que su funcionamiento no genera prácticamente ruido y sus dimensiones son muy reducidas lo que lo convierte en un candidato excepcional para la robótica del futuro.

Los micrófonos y altavoces cumplen su función aun siendo de calidad inferior. Registran y reproducen adecuadamente el sonido.

En cuanto al sensor táctil, su funcionamiento es muy robusto y las posibilidades de uso en robótica de bajo coste son patentes.

Como trabajo futuro se debería optimizar el peso de las piezas haciendo estudios de resistencia para saber si es posible extraer material reduciendo así costes y peso. Es indispensable también el diseño de una estructura que concentre todo el control de hardware en un único dispositivo. Resaltar también la necesidad de la creación de un circuito impreso que simplifique el método de conexión de la pantalla.

## 4.2. Software

En esta sección se tratan las distintas funcionalidades del software creadas con el propósito de trabajar con sonido y aprender patrones sonoros.

Las funcionalidades de software están divididas en cuatro partes, dependiendo de la su funcionalidad: simulación sonora, adquisición de sonido interno, HTM (Hierarchical Temporal Memory) y software de control de hardware.

### 4.2.1. Simulación Sonora

Este apartado surge del siguiente planteamiento: ¿por qué en el mundo de la robótica no se trabaja tanto con sonido en comparación con otros sentidos? Una de las razones es el hecho de que para ello se necesita un entorno libre de ruido y por tanto, controlado. La otra razón es que trabajar con el sonido real obliga a hacerlo en tiempo real, por lo que dificulta su experimentación. Por tanto, la deducción lógica es usar un simulador para ello, pero en las infraestructuras de simulación gratuitas y de uso más extendido no se dispone de esta funcionalidad. Es por ello que se debe generar un sistema de simulación de entornos sonoros. Sin embargo no se debe olvidar que es necesaria también una infraestructura real para corroborar y ajustar lo aprendido en simulación.

Después de una búsqueda exhaustiva, se llego a la conclusión que el único software útil para la aplicación buscada era GSound. Este proyecto está destinado a la simulación de sonidos en tiempo real en videojuegos y entornos virtuales. Estas características son indispensables a la hora del aprendizaje de sonido de forma simulada. El proyecto GSound (12) dispone del código accesible y una licencia de uso para investigación. Pero no tiene soporte para Ubuntu, lo que ha llevado a la necesidad de desarrollar una versión para Ubuntu Instalable para poder después generar un paquete de ROS que pudiese ejecutarlo. La razón de porque hacer la migración hacia Ubuntu es que el resto de sistemas usados en robótica, en especial ROS, tienen un mayor soporte en UBUNTU que en ningún otro sistema operativo. Al crear una versión para Ubuntu se unifica con gran numero de infraestructuras software gratuitas. El paquete de ROS es capaz de reproducir un archivo WAV dentro de un entorno virtual ya sea generado por código o importando un modelo

tridimensional de un entorno, ya sea una habitación o una jungla. Se definen las propiedades de los materiales del entorno y se define el lugar de la fuente de sonido y del receptor del sonido. Además, se dispone de un entorno, ya creado, usando la infraestructura de turtle-sim para poder mover el receptor en dicho entorno. La fuente de sonido no está representada visualmente pero el receptor sí, el cual es representado a través de un icono de una tortuga, lo que permite observar y controlar su posición y orientación; al mismo tiempo que ver como el sonido estéreo varia en función de su posición.

Por tanto, para tener simulación sonora, se han generado dos elementos: gsound_CMakelists y gsound_pkg.

- Gsound_CMakelists: Es una versión adaptada a los sistemas de sonido de Ubuntu de GSound. Se encarga de generar las librería y los elementos necesarios para usar Gsound en ROS.

- Gsound_pkg: Es un paquete de ROS que permite simular un entorno cúbico en el que hay una fuente del sonido deseado en el centro. La posición del receptor se controla con el teclado, visualizándose la posición y orientación a través de una tortuga. Este usa el entorno tutorial de ROS.


**Figura 4-13: Simulación del sonido en una habitación rectangular ( recuadro rojo) y fuente de sonido ( estrella verde )**

Para realizar esta prueba se deben seguir las instruciones del **Anexo C.**

### 4.2.2. Adquisición de sonido interno

Otra necesidad básica a la hora de trabajar con aprendizaje de patrones sonoros es la posibilidad de adquirir directamente el sonido generado dentro del ordenador sin necesidad de capturarlo de los altavoces o salida de audio física. Esto cobra especial importancia en sistemas a los que no se tiene acceso físico, como por ejemplo computación en la nube. En estas situaciones no se puede conectar un jack a la salida y después a la entrada de audio como recurso básico para la adquisición del sonido generado, y en los casos en los que se puede, se realiza de una manera muy ineficiente.

Otra opción sería generar el sonido en el mundo real y captarlo directamente con micrófonos. Esta opción es aun mas ineficiente al dar problemas de ruido o molestar a la gente de alrededor. De ese modo, con un sistema de adquisición del sonido interno, se puede hacer aprendizaje de forma eficiente y silencio para el entorno. Este sistema sirve para cualquier sonido generado internamente, ya sea de videos online, sistemas de reproducción de audio o de simulaciones sonoras o incluso, de la reproducción del sonido registrado por un micrófono real.

Por este motivo, se crea el paquete de ROS llamado **internal_sound_pkg**. En él se encuentran los módulos de python necesarios para la adquisición de el sonido interno y su conversión a formatos necesarios para el uso en HTMs. Esta es una lista de los principales módulos:

- **talker.py**: este módulo es el elemento principal que se ejecuta cuando se desea adquirir el sonido interno. El proceso consiste en conectarse al canal de sonido interno y convertirlo a FFT (Trasformada Rápida de Fourier) y se publica en un tópico de ro llamado “chatter”, un mensaje de tipo internal_sound_fft. Este mensaje está formado por cuatro arrays de floats : fft_abs_left, freq_left, fft_abs_right, freq_right. Las fft_abs contienen el módulo de las frecuencias correspondientes en freq_*. Teniendo left y right porque se hace de datos de estéreo. El muestreo se hace a 10.000Hz, lo que nos permite registrar por el Teorema del Muestreo (Nyquist–Shannon sampling theorem) frecuencias de hasta 5.000Hz. Teniendo en

cuenta que la voz humana tiene frecuencias fundamentales de entre 85-255Hz y que el ancho de banda usado para telefonía ronda entorno a los 4.000Hz; lo consideramos suficiente para un correcto registro de la voz humana, que es sujeto de estudio en el proyecto actual. No se ha usado más frecuencias de muestreo por que se ralentiza la conversión y se observaba retraso entre el sonido y su conversión. Al ser ideado este sistema para conversión a tiempo real del sonido para el aprendizaje, era imperativo mantener la frecuencia de muestreo lo mas alta posible pero sin que hubiese retraso. Después de la experimentación se ha llegado a la conclusión que 10.000Hz es el valor óptimo.

- **fft_coder_publisher.py**: Este módulo está destinado a codificar las fft's publicadas por talker.py en el formato SDR (Sparse Distributed Representation) (13) que es necesario como datos de entrada en HTMs usados para el aprendizaje y para publicarlas en un tópico de ROS llamado "/sound_sdr". Las SDR consisten en una representación en la que las frecuencias que sean similares tendrán una representación semejante. Un ejemplo podría ser la representación de 1Hz, 2Hz y 4Hz. En primer lugar, se debe escoger el rango de valores a representar, su resolución y el ancho. El rango de valores junto con la resolución indican cuantos bits de información se necesitan para su representación. El ancho, el número de bits activos, afecta al solapamiento de las representaciones. Ejemplo:

  - Dados los parámetros: Rango = [1,4] , resolución = 1, Ancho = 1. La representación de las frecuencias comprendidas sería la siguiente:

$$SDR(\,1Hz\,) = \,1000\,;\, SDR(\,2Hz\,) = \,0100\,;$$

$$SDR(\,3Hz\,) = \,0010\,;\, SDR(\,1Hz\,) = \,0001\,;$$

  - Dados los parámetros: Rango = [1,4] , resolución = 1, Ancho = 2. La representación de las frecuencias comprendidas sería la siguiente:

$$SDR(\,1Hz\,) = \,1100\,;\, SDR(\,2Hz\,) = \,0110\,;$$

$$SDR(\,1Hz\,) = \,0011\,;\, SDR(\,2Hz\,) = \,0011\,;$$

Como se puede observar la selección de la resolución, el rango y el ancho es fundamental para una buena representación de los datos. Si el interés está en una

representación muy exacta, se pondrá una resolución mayor al igual que un ancho menor o incluso unitario. Si por el contrario lo que interesa es que exista más semejanza entre valores parecidos, se escogerá un ancho mayor manteniendo la resolución. También hay que tener en cuenta que cuantos más bits, más procesado necesita la conversión, lo que ralentiza el sistema.

Por lo tanto, después de la experimentación sobre diferentes valores, se ha optado por escoger:

- **Rango de valores = [0, 10]**. El rango de valores viene determinado por el módulo *talker* donde se hacia la conversión de sonido a FFT. El rango de valores absolutos de cualquier sonido tiene un rango de 0-10.

- **Ancho = 3**. Este es escogido basándose en el uso que se hace y recomendaciones de los creadores del software de aprendizaje que se expone más adelante. Este valor permite cierto solapamiento entre valores muy similares, lo que facilita el aprendizaje

- **Resolución = 0,1.** Este valor es escogido por un método recursivo, donde se fue aumentando la resolución hasta conseguir una conversión a tiempo real. Por tanto, es el mínimo valor posible que no ralentiza en exceso la conversión.

❖ **Plot_SDR_animation.py:** Este módulo cumple la necesidad de visualizar la codificación a SDRs de una manera clara e intuitiva. Genera una ventana en el que se representa el SDR en tiempo real, el cual es generado por el fft_coder_publisher.py del sonido interno. Los vectores de salida de la codificación en SDRs se representa como una matriz de manera que los elementos de los vectores se colocan consecutivamente en dicha matriz. Los puntos blancos son bits activos (1) y los negros bits inactivos (0). Al estar dispuestos los bits del canal izquierdo primero y los del canal derecho después, los datos del canal izquierdos se verán representados en la parte superior y los del derecho en la inferior de la matriz.

Para ilustrar su funcionamiento se ha activado el sistema de captación y codificación del sonido interno ( Anexo C ). Una vez activado se lanza el plot_SDR_animation.py. Éste permite la visualización en tiempo real o grabar la animación en un fichero mp4 en función de las necesidades del usuario. Para ello, se ha realizado a una prueba

de sonido estéreo (14) Aquí se muestra las secciones del video y su correspondiente SDR.

**SIN SONIDO**

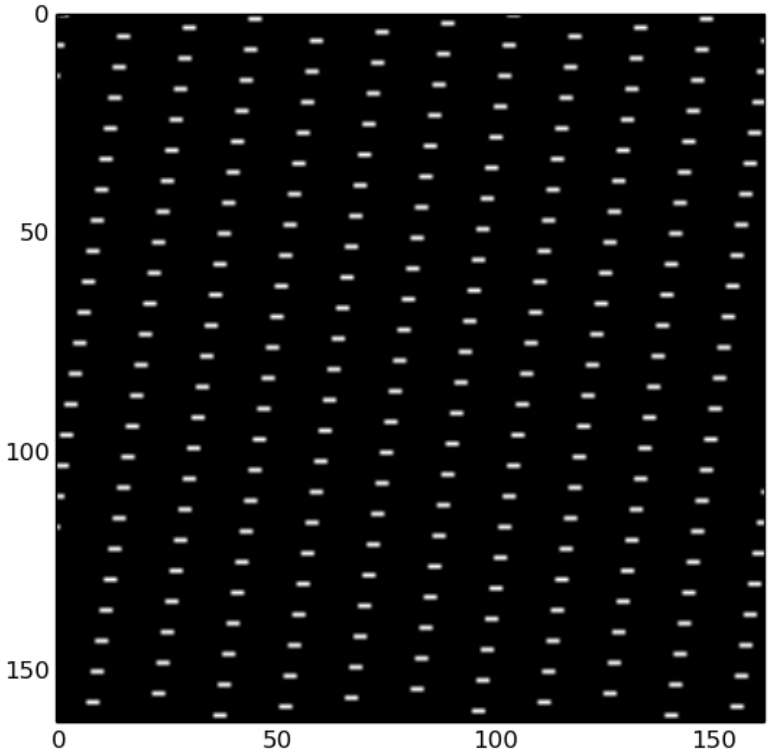



Figura 4-14: Output de Plot_SDR_Animation Sin Sonido


Figura 4-15: Correspondiente momento del video de input de sonido. Sin sonido

Resaltar también la ausencia de audio, la cual está codificada de manera que su disposición corresponde a las secciones sin audio.

**Sonido en canal izquierdo**

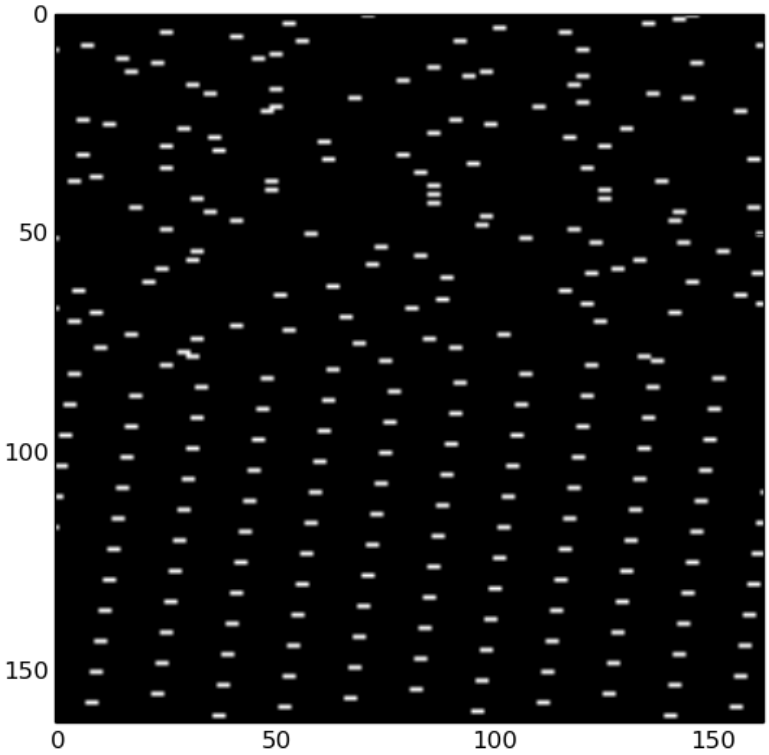



Figura 4-16: Output de Plot_SDR_Animation Con sonido en el canal izquerdo

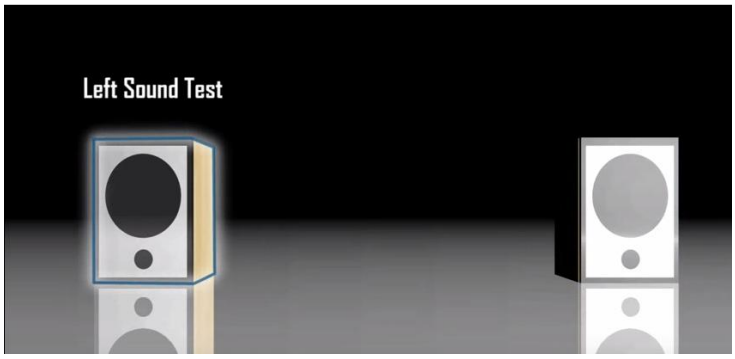



Figura 4-17: Correspondiente momento del video de input de sonido. Sonido por el canal izquierdo

### Sonido en el canal derecho

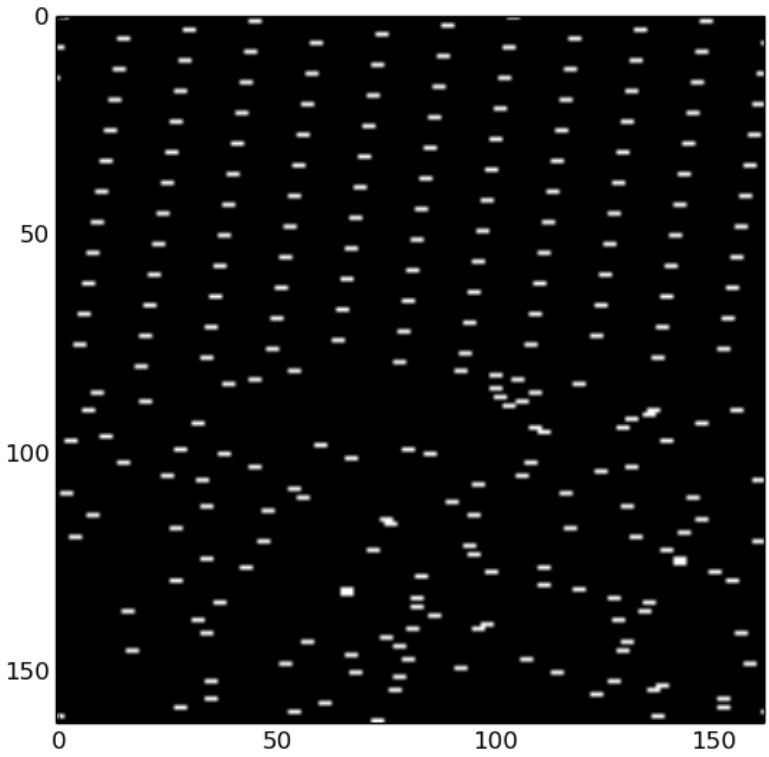



Figura 4-18: Output de Plot_SDR_Animation Con sonido en el canal derecho


Figura 4-19: Correspondiente momento del video de input de sonido. Sonido por el canal derecho

### Sonido en ambos canales

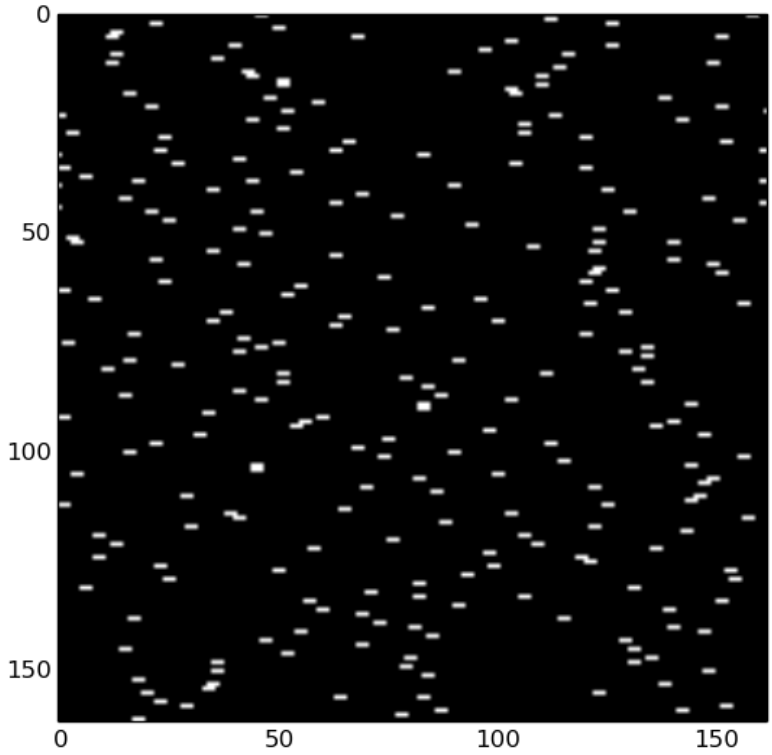



Figura 4-20: Output de Plot_SDR_Animation Con sonido en ambos canales

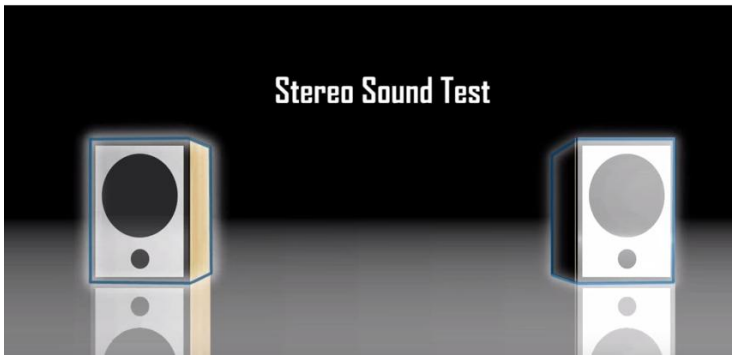



Figura 4-21: Correspondiente momento del video de input de sonido. Sonido por ambos canales

### 4.2.3. HTM

HTM son las siglas que corresponden a “Hierarchical Temporal Memory”. Este concepto fue acuñado por Jeff Hawkins en su libro divulgativo “On Intelligence” (3). El libro explica un modelo matemático que simula la estructura de cómo funciona el Neocortex humano, centro del aprendizaje en los mamíferos. Una de las ideas básicas de esta teoría es que el cerebro humano está continuamente realizando predicciones sobre el mundo que le rodea. Estas predicciones permiten al ser humano saber la mecánica del mundo real gracias a un entendimiento intuitivo de las leyes físicas de nuestra realidad. Hacer predicciones está en la base del aprendizaje.

Este entendimiento intuitivo es a su vez generado por la propia estructura, al ser capaz de abstraer conceptos de los datos de entrada. Esto permite dar al ser humano la ilusión de que el mundo sensorial que le rodea es ordenado y no caótico. Sin embargo, la realidad de los datos de entrada sensorial es que está llena de ruido e indeterminaciones. Pero esta estructura permite eliminar estas indeterminaciones contrastando predicciones de diferentes sentidos de entrada, dotándole de gran robustez hacia ruido.

Por estas propiedades, esta teoría tiene mucho potencial de poderse usar para el aprendizaje robótico autónomo ya que es capaz de aprender por si solo sin supervisión los conceptos cognitivos esenciales y es robusto al ruido. Otro factor interesante es el hecho de que esta estructura la rige un único algoritmo simple, lo que le dota del potencial para usar computación paralela e infraestructuras software como CUDA. Esto aumentaría la velocidad de procesado y aprendizaje de los robots.

#### 4.2.3.1. Teoría

Para comprender toda la estructura que se utiliza en la sección de estudio experimental y muchos de los elementos de los programas realizados alrededor de HTM, se debe primero comprender los conceptos que hay detrás de la implementación (14).

La unidad básica de funcionamiento de HTM es el nodo o célula. Si se comprende el funcionamiento de los nodos, se comprende el 90% de todo el funcionamiento de esta estructura.

**MECANICA DE UN NODO**

El nodo tiene dos fases de funcionamiento: *aprendizaje* y *predicción*. Los nodos pueden aprender y predecir a la vez, y en la realidad es como funciona. Sin embargo por fines didácticos, se explica como si se hiciese de forma secuencial: primero el aprendizaje y después la predicción.

Un nodo realiza tres funciones de forma secuencial en la fase de **aprendizaje**:

- **Memorización de patrones de entrada (P1)**
- **Aprender probabilidades de transición (P2)**
- **Agrupación temporal de patrones (P3)**

**(P1) Memorización de patrones de entrada**

Cada patrón distinto se guarda en un vector de la forma [c1,c2,...cn], donde estos patrones pueden ser cualquier tipo de datos ya sea audio, sonido, tacto o cualquier tipo de información.

**(P2) Aprender probabilidades de transición**

Consiste en crear una gráfica de Marcov (16) y normalizarla. Este procedimiento consiste en memorizar el número de transiciones de un patrón ya memorizado a otro. El número de salidas de un patrón hacia otro es dividido por el número total de salidas de ese patrón. De esta forma, tenemos la probabilidad de que si una entrada es un cierto patrón $c_x$, la del siguiente patrón será $c_y$, por ejemplo.

Esto genera un grafico con “n” vértices (patrones ) y “m” conexiones (transiciones).

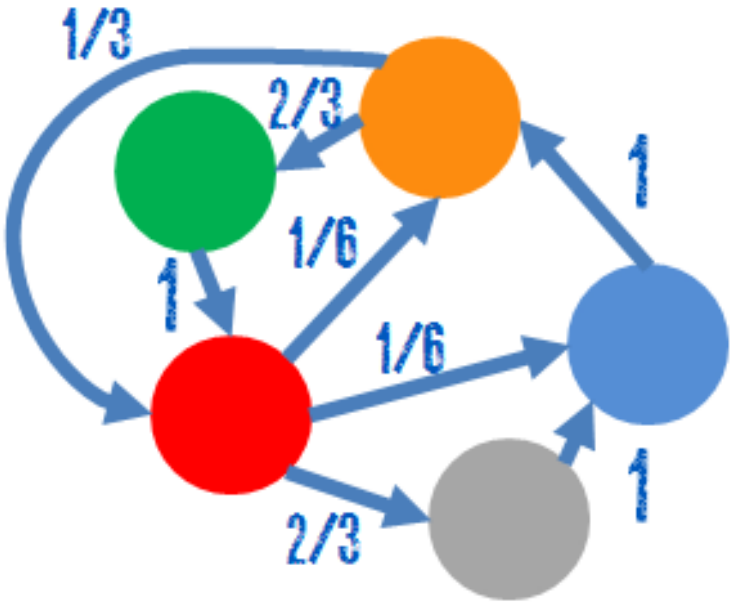



Figura 4-22: Ejemplo de Diagrama de Marcov Normalizado de 5 patrones distintos

**(P3) Agrupación Temporal de patrones**

Consiste en generar subconjuntos de estos vértices del gráfico de Marcov anteriormente mencionado. Estos subconjuntos están formados por aquellos patrones que son muy probable que se sucedan en el tiempo, es decir, secuencias. El procedimiento para esta segmentación se denomina “Clustering”.

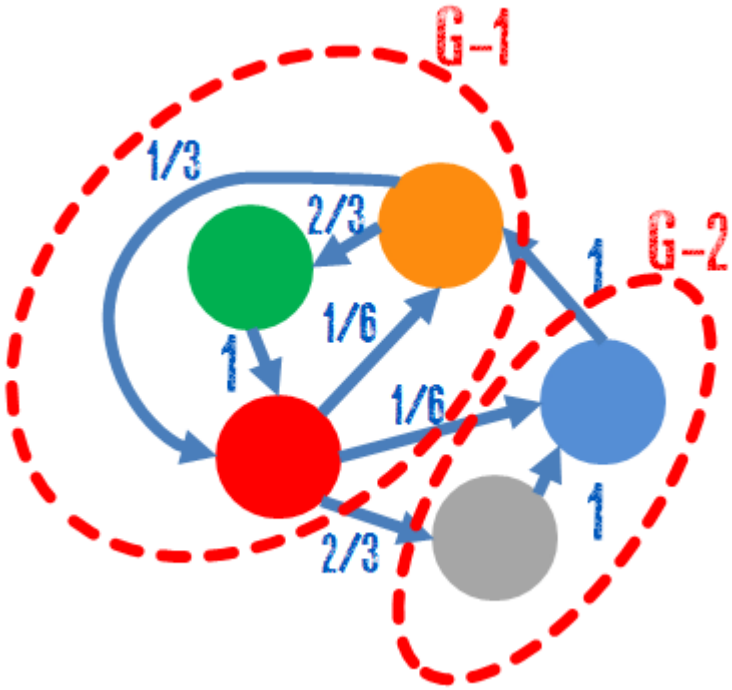



Figura 4-23: Agrupación de los patrones del ejemplo de la figura 4-22

Los resultados del clustering pueden venir en forma de un dendograma. Este representa diferentes niveles de agrupación de patrones. El nivel más bajo consiste en hacer que cada patrón sea un subconjunto diferente. A medida que se sube de nivel se agrupaba cada vez más patrones, generándose menos subconjuntos.

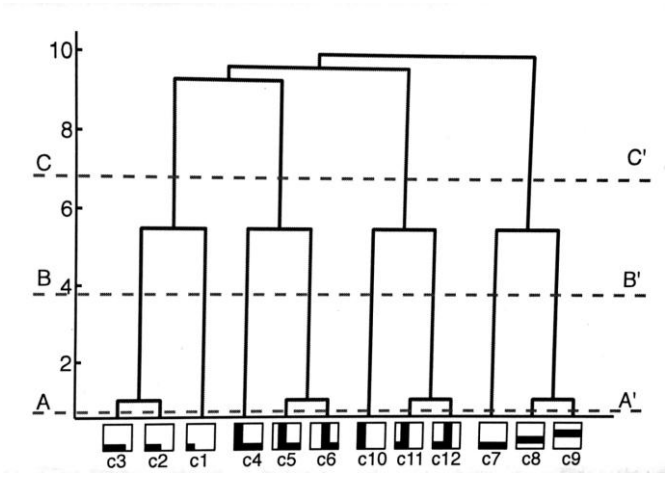



Figura 4-24: Ejemplo de dendograma de patrones visuales (14).

Estos tres pasos se pueden realizar de manera secuencial hasta estabilizar el gráfico y sus subconjuntos o también se pueden realizar primero el paso 1 hasta estabilizarse, después el paso 2 y finalmente el paso 3. En la realidad, se hace de forma simultánea para no tener que someter al nodo a los mismo patrones por triplicado.

El resultado de final en un nodo debe ser un conjunto de patrones memorizados, las probabilidades de transición entre ellos normalizadas y un conjunto de grupos de patrones que representan secuencias de los mismos.

Con los nodos ya construidos se procede a la fase de predicción. En la fase de predicción, el nodo recibe unas entradas que son patrones. Este reconoce que patrón es y a continuación se introduce el siguiente patrón. Durante este proceso va realizando predicciones de que patrón va a seguir y con estos datos saca como salida, no el patrón predicho sino la secuencia a la que pertenece estos patrones de entrada. Este mecanismo es la clave para la abstracción y la reducción de la variabilidad entre los datos de entrada (patrones) y los datos de salida (secuencias). También es la razón de la palabra “temporal” en HTM, ya que estas predicciones tienen en cuenta los patrones pasados y presentes para hacer predicciones y clasificaciones**.**

Este funcionamiento permite el elemento más importante y clave en HTM, la jerarquía (Hierarchical). Las salidas de los nodos se pueden conectar a las entradas de nodos en niveles superiores de forma jerárquica.

**MECÁNICA DE LOS NIVELES**

Una vez los primeros nodos han aprendido, éstos se ponen en modo de predicción y se someten a los mismos patrones de predicción para que los nodos del nivel superior

prendan.

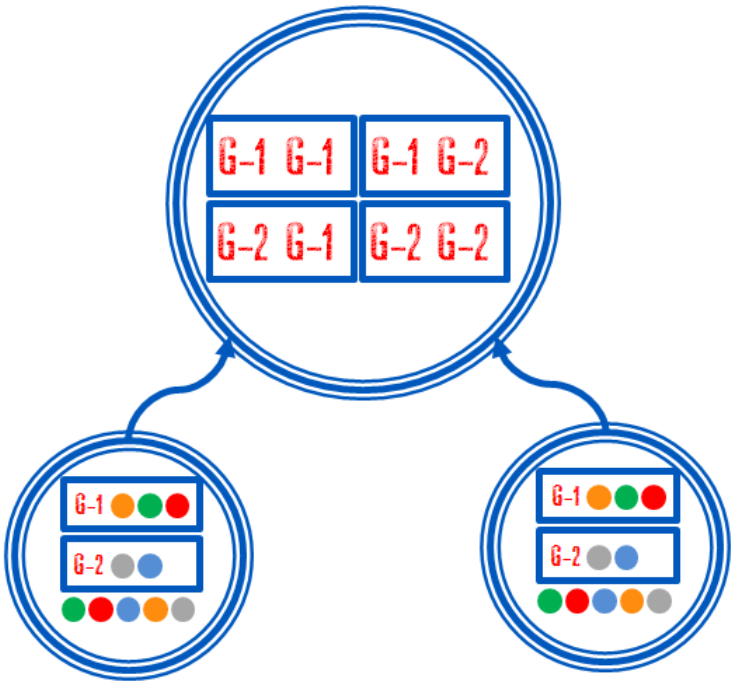



**Figura 4-25: Ejemplo de jerarquía de dos niveles**

El modo en que aprenden los nodos del segundo nivel y posteriores es idéntico a los del primer nivel. La única diferencia es que los niveles superiores memorizan secuencias que las consideran patrones y extraen subconjuntos de estas secuencias.

Esta es la idea básica, pero hay ciertos procesos adicionales. Cuando se realizan predicciones, no solo se propagan los patrones sino también las probabilidades de que ese patrón sea correcto y predicciones (14). Esto permite que se propaguen las probabilidades y que se desambigüen hacia arriba en la jerarquía patrones que debajo no tenían probabilidades altas o incluso cambiar probabilidades cruzando predicciones de otras estructuras de HTM de otro sentidos. Un ejemplo podría ser que la estructura HTM de sonido crea que oye un perro, la de la visión un oso, pero que al cruzar resulte que haya más probabilidad de que sea un puma.

**MECÁNICA DE TRABAJO DE ENTRADAS RUIDOSAS**

Un tema a comentar es que para facilitar el funcionamiento, las entradas deben ser previamente limpiadas. Esto se hace simplemente discretizando los datos. Cuanto más esto se haga, menos patrones diferentes habrá, pero más clara será la diferencia.

Un ejemplo podría ser como clasificar un cubo lleno de confeti. Se sabe que solo hay un número de colores. Sin embargo, por fallos de fabricación y deterioros o incluso la luz incidente, en cada momento pueden haber muchos más. Entonces lo que se hace es trazar una línea que ayuda a discernir cuando un color pasa a ser considerado otro. Este proceso ya se ha mencionado cuando se hablaba del formato SDR, donde se usa para discretizar todo el espectro de frecuencias dadas por el FFT en un número más asequible de datos, en los que las entradas similares acaban siendo representadas por el mismo patrón.

#### 4.2.3.2. Nupic, OpenHTM y PyHTM

Mientras que la teoría tiene una implementación matemática clara e implementable computacionalmente, después de una investigación posterior a la creación de esta teoría, se han descubierto sistemas de implementación que aun basándose en las mismas ideas, su implementación es diferente. Las razones son diversas: hay aspectos que no se contemplaban en la teoría original como el umbral de activación de los patrones o ineficiencias en la implementación directa.

Hay tres implementaciones hoy en día: NUPIC, OPENTHTM y PyHTM. Sin embargo, de entre las tres, sólo una es adecuada para este proyecto. Por un lado, PyHTM no tiene soporte desde el 2011 y contiene muy poca funcionalidad. Por otro lado, OpenHTM sólo tiene soporte para Windows por el momento.

NUPIC es una infraestructura opensource para propósitos de investigación. Está formada por una comunidad muy activa y recibe actualizaciones cada pocos meses. Además, el propio creador de la teoría de HTM es el fundador de este grupo y está a la cabeza del proyecto.

#### 4.2.3.3. NUPIC Swarming y NupicHTM

NUPIC aporta dos métodos para el trabajo con HTMs. El primer método es el Swarming. Consiste en que a través de objetos ya creados se generan las estructuras jerárquicas necesarias de manera automática en base a los datos de entrada y posteriormente, este modelo generado optimizado se usa para realizar predicciones sobre los datos. Por otro lado, el segundo método obliga a crear los modelos manualmente pero permite un control mayor sobre su configuración y sus salidas.

Por tanto, se ha creado un paquete de ROS que denominado “nupic_swarm_pkg” en el que se incluye la infraestructura que encapsula los métodos de Swarming en ROS. Además, se ha creado el “nupic_pkg” que encapsula todos lo necesario para trabajar con la estructura de HTM.

Después de pruebas preliminares, se ha observado que aun siendo mucho mas cómodo para el usuario el sistema de Swarming, ya que solamente se debe configurar un archivo para iniciar todo el proceso de aprendizaje y predicción, éste no es compatible con la configuración de los datos de entrada necesarios en este proyecto. Descubriéndose que la infraestructura de Swarming no tiene soporte para listas de valores de manera directa, sino solo para secuencias de un solo valor real o escalar. Esto es muy útil para análisis de datos y detectar anomalías. Sin embargo, aun no se dispone de la infraestructura para usar toda la teoría de HTMs. Por tanto, se ha optado por no proseguir con el uso del paquete y se migró al uso del “nupic_pkg”.

#### 4.2.3.4. Nupic_pkg

Nupic_pkg es un paquete de ROS en el que se encapsula la infraestructura necesaria para realizar el estudio experimental de secciones posteriores. De todos los módulos de python disponibles, el principal es **Learn_htm_tests_v2**.

Este programa contiene todo lo necesario para realizar tanto la fase de aprendizaje, la de clustering, como la de predicción. Consiste en un python ejecutable con una selección de comandos según lo que se desee para facilitar su uso y la realización de las posteriores pruebas. Sus tres funciones básicas son:

**1. Aprendizaje.**

La función de aprendizaje genera un nodo de ROS que dado un input de datos leídos desde el tópico de “/sound_sdr”, mencionado anteriormente, reproduce la base de datos de sonidos a memorizar según las fases de aprendizaje y genera un htm_bucket en forma de fichero Pickle. Este htm_bucket es un objeto que contiene todo el conocimiento generado durante el aprendizaje y que se usa para el posterior clustering y predicción. Se podría considerar la unidad de conocimiento. Además, el error durante el aprendizaje de cada nivel de la jerarquía se publica en el tópico “/htm_error_lvlx”.

El aprendizaje tiene tres fases principales: *inicialización, aprendizaje* y *cierre.*

En **la fase de inicialización** se crea el objeto htm_bucket con todos los datos que definen la estructura del HTM, entre ellos el número de niveles, la cantidad de patrones distintos de entrada y la cantidad de secuencias de salida que se quieren aprender. El número de patrones distintos de entrada indica cuanto discretizamos los patrones sonoros de entrada en forma de SDR. Esto se ocupa, lo que se denomina SpatialPooling, un elemento de NUPIC. Una vez pasado el primer nivel no se discretiza sino que simplemente se propagan las secuencias detectadas. El número de secuencias de salida determina como se hace el clustering de patrones. A mayor número más clusters se generaran pero se tendrán secuencias de patrones más cortas. En cuanto al número de niveles esto determinara el nivel de abstracción y complejidad de los conceptos aprendidos. A mayor número de niveles, mayor será la abstracción y de paso, la estabilidad de los patrones predichos.

En la **fase de aprendizaje,** se reproducen los archivos de sonido del que se quiere aprender. La razón es que deben estar sincronizados el final de cada archivo con el reseteo de la estructura de aprendizaje para evitar que aprenda las secuencias de cada archivo y se concentre en considerar cada uno como patrones individuales.

La secuencia de aprendizaje es la siguiente: se empieza a reproducir los archivos de audio durante el tiempo estipulado en la inicialización. A continuación, se pone en modo aprendizaje el primer nivel y se empieza a someter a los datos recogidos en tiempo real desde el tópico “/sound_sdr”. Durante el proceso de aprendizaje este realiza predicciones y se genera un índice de error basado en como de similar al patrón siguiente es la predicción. Este error se publica en el tópico de “/htm_error_lvlx” que a su vez, servirá para monitorizar si se estabiliza el aprendizaje. Una vez agotado el tiempo destinado al aprendizaje, se pone el primer nivel a predecir y el segundo nivel a aprender y se vuelve a someter a los mismos archivos de sonido la estructura. Este proceso se realiza hasta que se agote el tiempo de aprendizaje en la ultima capa de la estructura.

Por último, en la **fase de cierre** se guarda toda la estructura de HTM en forma htm_bucket en un archivo Pickle. Estos archivos no son más que una serialización de un objetos de python (15).

## 2. Clustering

El clustering no es realizado con herramientas NUPIC, ya que éste no tiene aun soporte para ello, sino con “scikt-learn” en el módulo de clustering, más concretamente el de *Aglomerative clustering* (16). Esto surge del descubrimiento de que NUPIC no realiza el *temporal grouping* indicado en la teoría, sino sólo aprendizaje sobre patrones y predicciones, aunque se esta trabajando en ello en estos momentos. Port tanto, se ha decidido crear un software para realizar este clustering.

Lo que se debe clusterizar es una cadena de Marcov, que es un gráfico orientado en el que los caminos tienen un valor que es la probabilidad de transición. Se podría asemejar a un gráfico de distancias entre ciudades, excepto por una diferencia, y es que la distancia por ejemplo entre Barcelona y Madrid que en un gráfico de distancias normal es la misma que de Madrid a Barcelona, en este caso no sería. Esto se debe a que las distancias, en este caso, son probabilidades y no tiene por qué haber la misma probabilidad de transición entre el patrón C1 y el patrón C2 que a la inversa. Por tanto, los sistemas tradicionales de clustering, como Mean-Shift, Affiniti-Propagation o K-means, no sirven.

Para decidir qué tipo de clustering usar, se ha tomado como referencia el sistema usado por George Deleep en su tesis (14) donde usaba una HTM para aprender patrones visuales simples. El Recomienda usar Agglomerative clustering. Este permite usar matrices de conectividad no simétricas, perfecto para Markov-Graphs.

Por tanto, se extrae los datos de las probabilidades de transición entre los patrones aprendidos y se generan matrices de conectividad y matrices de probabilidad de transición. Las matrices de conectividad sólo indican que patrones están conectados. Las matrices de probabilidad de transición ya contienen los datos para saber la probabilidad de transición entre un patrón y todos a los que está conectado. Esta probabilidad se extrae del valor que genera NUPIC de la fuerza en la conexión sináptica entre los patrones. NUPIC representa esta probabilidades bajo el contexto más biológico de cuan fuerte es una conexión sináptica entre dos neuronas, que es la base del conocimiento.

Una vez generados los clusters , estos datos se pasan al visualizador que genera gráficos de Marcov para plasmar la estructura de la HTM por niveles.

**3. Predicción**

La predicción usa el archivo htm_bucket de formato pickle dado y empieza a reproducir el archivo de sonido deseado. En tiempo real realiza predicciones a través de toda la estructura HTM. Publica dos elementos. Por un lado, publica los datos de predicción de los niveles deseados de la HTM en tópicos llamados "/htm_output_lvl"+numero_nivel. De esta forma, sólo se publican los datos de los niveles deseados. Esto cobra especial importancia en estructuras de gran número de niveles en los que quizás sólo se requiera los resultados del último nivel. Aquí se publica ,en forma de vector, el grupo secuencial detectado por cada nivel, el grupo predicho en el instante siguiente de tiempo y la predicción del instante anterior para poder comparar con el grupo detectado en el instante presente.

También se publica en el tópico "/variable_blackboard" la etiqueta del la secuencia detectada en el último nivel de la jerarquía. De esta forma, se puede tener una idea del concepto de más alto nivel que se está activando en cada momento.

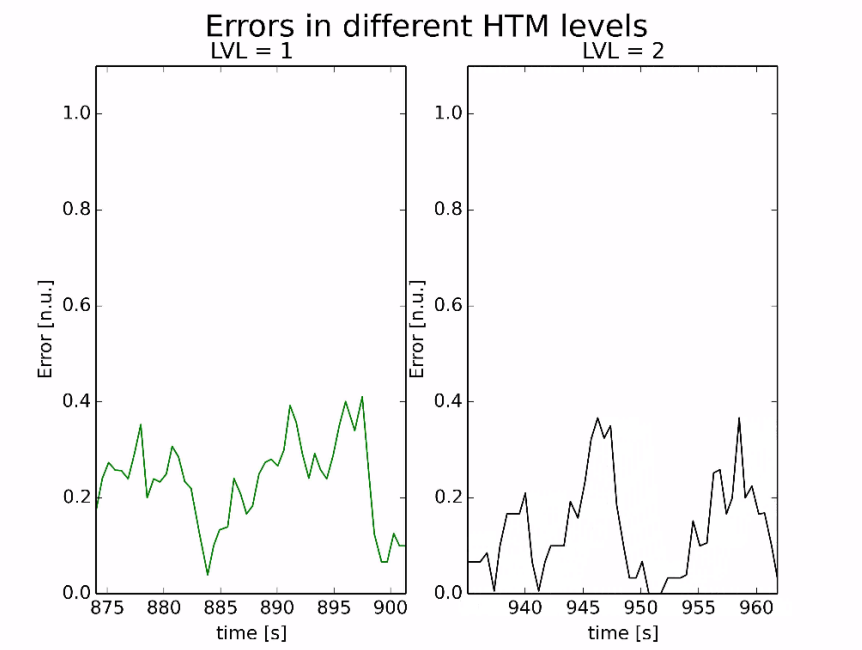
#### 4.2.3.5. Visualización

Es esencial disponer de herramientas de visualización tanto del proceso de aprendizaje como de los resultados. La razón es que se dispone de gran cantidad de información difícil de entender en crudo.

Hay tres módulos para visualizar **el error en el aprendizaje, las predicciones/output del HTM** y **el clustering.**

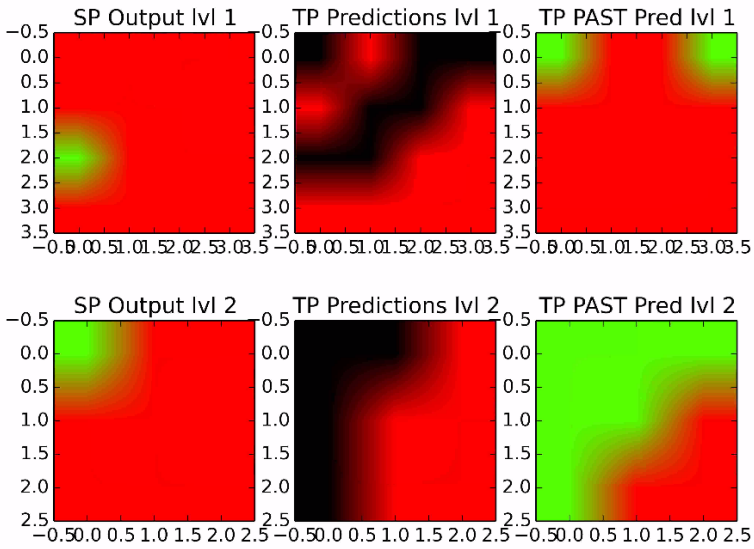
- **Error en el aprendizaje.** Es representado por el modulo **plot_error_lvlx.** Éste lee del tópico "/htm_error_lvlx", mensajes de tipo error_array_lvx.msg. Consiste en un vector con el error actual de aprendizaje de cada nivel, y otro vector indicando si cada nivel esta aprendiendo o no. Con estos datos se genera una ventana en el que se representa el tanto por uno de error en función del tiempo desde que se empezó el aprendizaje. Una vez que el aprendizaje de cierto nivel ha terminado, se queda con los últimos valores registrados mientras se actualizan los datos del nuevo nivel aprendiendo.

También, en vez de observar el error en tiempo real, se puede grabar un archivo mp4 de un fragmento temporal.




**4-26: Ejemplo de graficado del error en dos niveles durante el aprendizaje**

- **Predicciones/output del HTM:** Para su representación se usa el módulo **plot_htm_lvlx**. Este módulo tiene también capacidad de grabar fragmentos temporales o representar los datos en tiempo real sobre una ventana emergente. Consiste en un nodo de ROS que lee los datos de detección y predicción de secuencias del HTM que esté en ese momento prediciendo. Usa los datos de los niveles deseados de cada uno de los tópicos de nombre "htm_output_lvl"+numero_nivel. Los datos de los mensajes son de tipo htm_output.msg que consiste en dos vectores que tiene los datos de predicciones y de secuencias detectadas en formato booleano. Se transforman en matrices rectangulares en las que se representa cada bit activo de forma secuencial hasta llenar la matriz y si faltan elementos en el vector para hacer una matriz cuadrada completa se rellena con valores inactivos.

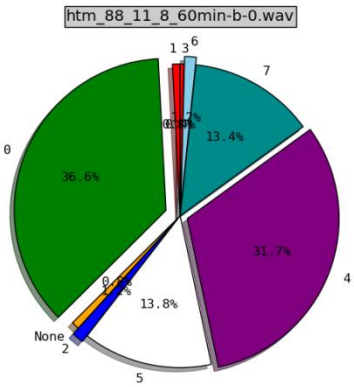



4-27: Ejemplo de predicción de dos niveles de una HTM

Se muestran para cada nivel deseado: *los valores actuales detectados* (SP Output), *la predicción del siguiente instante* (TP Predictions) y *la predicción pasada* (TP PAST).

También para representar las predicciones de la capa superior del HTM, medios durante el periodo de predicción, se generan unos gráficos circulares a través del módulo "**pie_chart.py**". Este representa las detecciones medias de cada secuencia

de salida del último nivel de la jerarquía, que representa a su vez el nivel más abstracto cognitivo. Esto brinda la posibilidad de extraer perfiles de detección útiles para la clasificación de los patrones de entrada.




**4-28: Ejemplo de grafico circular con los porcentajes de predicción de cada entonación emocional**

- **Clustering:** Para realizar los clusters se usa el módulo **plot_graph.** Este extrae los datos de clustering del modelo htm del htm_bucket suministrado y genera un gráfico en base a la matriz de conectividad y de probabilidad de transición; además del etiquetado automático de todos los patrones aprendidos. Este etiquetado se hace en base a la estructura biológicamente inspirada de NUPIC. Cada patrón se representa como una columna compuesta por un número de células. Cuando se activa alguna de las células de esa columna es que ese patrón esta activo, y dependiendo de qué célula se active es el mismo patrón en diferentes contextos

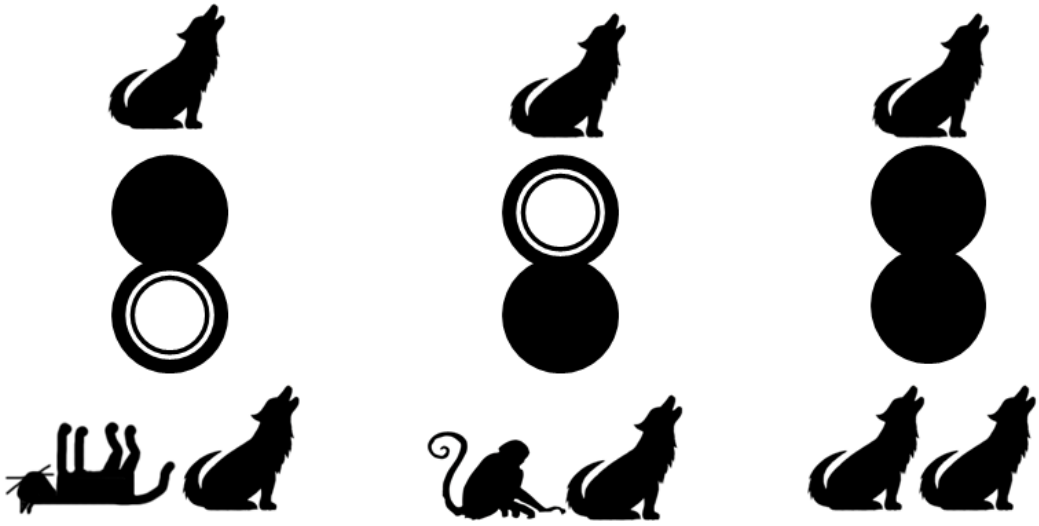
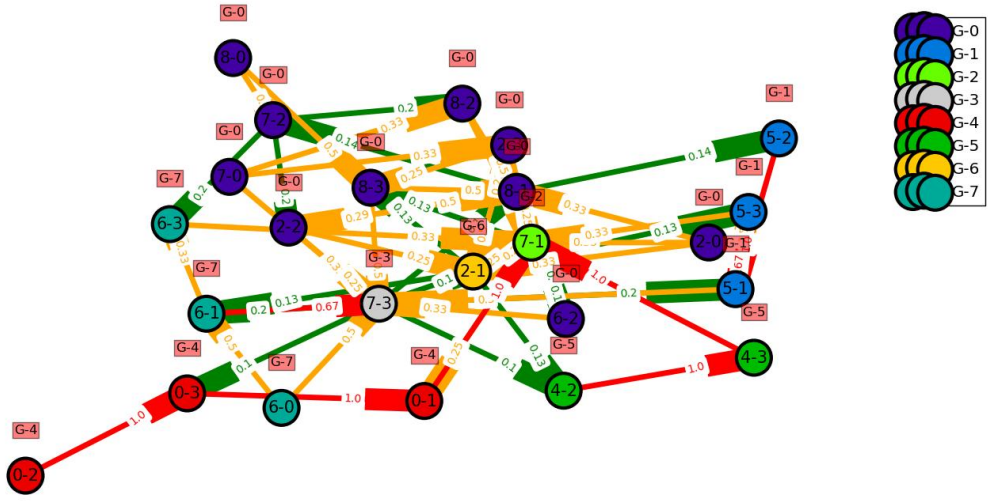


**Figura 4-29: Ejemplo de una columna con dos células y animales como entrada de patrones. Las tres columnas representan perro , pero cada una esta en un estado diferente al haber ocurrido patrones diferentes anteriormente. Esta en diferentes contextos.**

Por tanto, el etiquetado está formado por dos números separados por un guion de la forma X-Y, donde X es el número de columna e Y el número de célula. Se tratan como patrones diferentes, las activaciones de células distintas de la misma columna ( Ej.: 2-1, 2-2 ) ya que son tratados como tales por NUPIC. El gráfico también muestra con codificación de colores y con etiquetado el grupo secuencial al que pertenece, que el resultado del clustering. También indica las conexiones entre patrones de manera direccional, además de la dirección con una sección mas ancha en el patrón de destino. Estas conexiones también tienen una etiqueta, indicando la probabilidad de transición normalizada y un código de colores que indica si la probabilidad es baja (verde), media (amarilla) o alta (roja). De este modo, el propio módulo es capaz de representar únicamente las conexiones entre patrones filtrando por probabilidad.




4-30: Ejemplo de Clustering de nivel con 8 subgrupos de salida

#### 4.2.3.6. Software para Hardware

Este último punto describe el software usado para el control y utilización del software.

Todo el software se ha colocado en un paquete de ROS llamado **kodama_hardware_pkg** por comodidad.

**E-Ink Display**

Para la pantalla de tinta electrónica se usa “epaper_kodama_light.ino”, situado en “epaper_arduino_130_412/ epaper_kodama_light /”, del Arduino UNO que va rotando imágenes de manera secuencial. Estas imágenes están en formato X BitMap (xbm), localizadas en epaper_arduino_130_412/libraries/Images. Las imágenes son representaciones emocionales simples (17) que se han convertido a “xbm” de 264x176 pixeles por el procedimiento indicado en el manual de uso de la pantalla (18). Las imágenes que van rotando son:

**Servos**

En primer lugar, al usar como hardware sobre un Arduino MEGA un shield, se requiere la librería correspondiente para su uso. En este caso, la librería Tinkerkit.

Los ficheros necesarios para mover los servos están situados en “servo_moving”. Los servos se pueden mover de dos maneras: de forma autónoma o a través de USB.

- **Autónoma:** En el caso que se requiera únicamente la realización de movimientos sin la supervisión de un ordenador, ésta es la manera más adecuada. Toda la toma de decisiones es realizada por un arduino. Se dispone por tanto de un script de Arduino llamado “move_3_servos_no_pc”. Este mueve los tres servos conectados en los puertos O1,O2 y O3 entre los siguientes rangos:

  - Servo conectado a O1 entre 30º y 60º
  - Servo conectado a O2 entre 70º y 110º
  - Servo conectado a O3 entre 70º y 110º

  Estos movimientos se realizan infinitamente con paradas de 1 segundo cada tres repeticiones.

- **A través de USB:** Usando la infraestructura Arduino de comunicación serie, se pueden mandar comandos desde el ordenador hasta el Arduino. Para ello, se deben usar por el lado de la computadora el script de python “move_3_servos_tools.py” y por el lado Arduino “servo3_clean.ino”.

  El script de Arduino recibe pedidos sobre el movimiento de los servos o sobre su posición. El script de python contiene dos clases:

  1. **ServoArduinoObject:** Esta es la clase de más bajo nivel que permite pedir información sobre la posición de un servo en particular o de mover un servo a una posición deseada.
  2. **Gestures:** Esta clase esta ideada como template para incluir movimientos complejos y enviarlos. Actualmente tiene implementado :

     - Wave: Un movimiento para mover el dedo arriba y abajo.

- Move_test_1: Mueve los tres servos a dos posiciones. Muy útil para ver la simultaneidad de ejecución de los comandos.

- get_info_test1: Ejemplo de cómo adquirir la posición actual de los servos.

- move_by_cmds: Permite acceder a toda la funcionalidad de ServoArduinoObject por comandos.

**PCB_Motors**

Para su control se usa "twin_motor_serial_simple_mod.py", en PCB_SOFTWARE/pcb_drivers/src/. Consiste en un programa de envío de comandos por terminal. Dispone de varios movimientos preestablecidos o permite el envío de comandos manualmente.

**Sensor de presión**

Para la lectura del sensor de presión, únicamente es necesario la lectura de una de las entradas analógicas de cualquier Arduino y enviarlas por USB al ordenador. Por tanto, con el script de Arduino "listen_analog_pin_write_serial.ino" se cumple esa tarea.

Una vez adquiridos los datos, se dispone de dos scripts de python: "Arduino_Monitor.py" y "wx_mpl_dynamic_graph.py". El primero permite únicamente procesar los datos de Arduino y mostrarlos en el terminal. El segundo permite visualizar estos datos sobre una gráfica.

**Cámaras Ai-Ball y Modulo de Sonido UCA202**

Ambas piezas de software usan los software suministrados por el fabricante sin ningún tipo de modificación. En el caso de las cámaras se puede acceder desde una aplicación o vía software de cámaras-IP. En el caso del sistema de sonido el software viene incluido en la mayor parte de las distribuciones de Ubuntu.

## 4.3. Estudio Experimental

Una vez creada toda la infraestructura de software, se procede a la realización de un estudio experimental para evaluar el rendimiento y los resultados de los sistemas utilizados.

El objetivo del estudio es que el software desarrollado adquiera un conocimiento, cuyos elementos sean difícil de aislar en algoritmos (19) y en que la estructura jerárquica pudiese aportar cierta ventaja sobre otros métodos. Por supuesto, éste ha de ser un tema referente a HRI, debido a que está implícito en el objetivo del diseño del hardware y además, porque es un área de estudio de gran relevancia hoy en día. Por último ha de girar en torno al análisis del audio por las mismas razones por las que se decidió crear paquetes de ROS de análisis de sonido, es decir, por el hecho de que el trabajo con sonido es un campo poco investigado en robótica, lo que lleva a la necesidad de crear herramientas para facilitar su estudio de forma general.

Se ha considerado adecuado crear un sistema capaz de detectar el trasfondo sentimental destilado del habla de una persona ya que:

- Es altamente complicado extraer un algoritmo que detecte si alguien está enfadado, triste o alegre por únicamente el sonido. Por tanto, se presta al aprendizaje no supervisado.

- Es de vital utilidad en interacción con personas en lo que al aprendizaje se refiere. Un niño al principio únicamente entiende que ha hecho algo bien o algo mal por la entonación de sus tutores. La comprensión de error es fundamental, ya que permite reforzar lo aprendido o corregir lo erróneamente aprendido. También, tiene una utilidad extraordinaria a la hora de la interacción universal, que está ligada también a la manera inicial en los niños aprenden. Los seres humanos podemos entender el tono de una conversación de un idioma que no entendemos, lo que nos da información básica sobre HRI.

### 4.3.1. Datos de Estudio

En primer lugar, se debe seleccionar la base de datos con la que se realizará el aprendizaje. Debía ser una base de datos que dispusiera de las mismas frases con entonaciones diferentes. Además, estas muestras de voces debían ser pronunciadas en diferentes

idiomas y por distintas personas. De esta forma, el aprendizaje se haría de manera genérica para cualquier individuo con independencia de su sexo, edad o idioma. Otro requisito era que fuesen bases de datos de uso libre.

Se seleccionaron tres bases de datos que cumplían los requisitos:

- **Berlin_Database_Emotional_Speech.** Cuyo idioma es el alemán y contiene siete entonaciones (20).
- **EMOVO.** El idioma hablado es el italiano, incluyendo también siete tonos (21).
- **SAVEE.** En el que el idioma es el inglés y contiene siete tonos (22).

Todas las entonaciones con interés de estudio están en por lo menos dos de las tres bases de datos, y los que son considerados esenciales, en los tres. Los ocho tonos que se aprenderán junto a las bases que los incluyen son:

| Emoción | Sigla | Base de Datos en la que hay representación |
|---|---|---|
| **1. Felicidad** | ( H ) | Berlín, EMOVO, SAVEE |
| **2. Enfado** | ( A ) | Berlín, EMOVO, SAVEE |
| **3. Tristeza** | ( S ) | Berlín, EMOVO, SAVEE |
| **4. Asco** | ( D ) | Berlín, SAVEE |
| **5. Miedo** | ( F ) | Berlín, EMOVO, SAVEE |
| **6. Aburrimiento** | ( B ) | Berlín, EMOVO |
| **7. Sorpresa** | ( X ) | EMOVO, SAVEE |
| **8. Neutro** | ( N ) | Berlín, EMOVO, SAVEE |

**Tabla 1: Listado de 8 emociones representadas en las bases de datos usadas**

Seguidamente, todos los archivos se han clasificado en dos grupos: el 80% (1.279 archivos) se destinan al aprendizaje y el 20% restante (328 archivos) al testeo y predicción. La asignación de un grupo u otro se ha decidido de manera aleatoria, intentando mantener cierta proporcionalidad entre los diferentes hablantes.

Por otro lado, se ha normalizado los formatos de grabación a estéreo y a 44.100Hz. De este modo, en los archivos mono, se ha duplicado en el segundo canal la misma información que contenía el primero. En el caso de archivos muestreados a frecuencias más altas (48.000Hz) se ha rebajado la calidad hasta los 44.100Hz. Por otro lado, los archivos usados

en la predicción se han agrupado por tonos de forma aleatoria en un único archivo de sonido para facilitar su reproducción.

Todas estas conversiones se han realizado a través del programa “sox”, implementándose en el módulo “tools_sound_files.py” en el paquete de ROS “nupic_pkg”.

### 4.3.2. Experimento Procedimiento

El experimento ha constado de las siguientes fases: *aprendizaje, generación de gráficos de clusters y predicción.*

1. **Aprendizaje.** Se usan los archivos de audio destinado a aprendizaje en el mismo orden para cada secuencia de aprendizaje por capa. Se realizan tres experimentos por estructura de HTM durante 15minutos, 30min y 60minutos por nivel. El objetivo es observar el tiempo necesario para la estabilización y reducción del error en diferentes estructuras de HTM. Se generan videos de la evolución del error.

1. **Generación de Gráficos de Clusters.** Se realizan Gráficos de los clusters por cada nivel en la estructura HTM, uno global y otro centrado y aplicado en la región de máximas conexiones a nivel visual.

2. **Predicción:** La predicción se hace con los archivos de predicción restantes agrupados por tonos y unidos en un solo archivo de sonido de manera aleatoria en su orden. Se somete el sistema a detectar que tono oye y se va registrando. Una vez terminado el archivo de sonido se contabilizan el numero de detecciones de cada tono y se crea el porcentaje de tiempo que cada tono ha sido detectado. Esto se grafica en un grafico circular creando una perfil de detección para cada uno a de las estructuras HTM con tiempos de aprendizaje distintos. También se grava el proceso de detección a través del graficado de las detecciones y predicciones por nivel

Se realizaron las tres fases para tres configuraciones de HTM diferentes con n_m_s, siendo “n” el numero de patrones frecuenciales distintos de entrada, “m” el numero de subgrupos/secuencias de salida del primer nivel y “s” el numero de subgrupos/secuencias de salida del segundo nivel. Todas las estructuras son de dos niveles y son las siguientes:

| Nombre Estructura | n | m | s |
|---|---|---|---|
| E1 | 88 | 11 | 8 |
| E2 | 100 | 30 | 8 |
| E3 | 200 | 40 | 8 |

### 4.3.3. Resultados Experimentales

Los resultados se dividen en dos subsecciones: los referentes al aprendizaje que incluyen la evolución del error y el clustering, y por otro lado, los resultados referentes a las predicciones y detecciones.

#### 4.3.3.1. Resultados Aprendizaje y Clustering

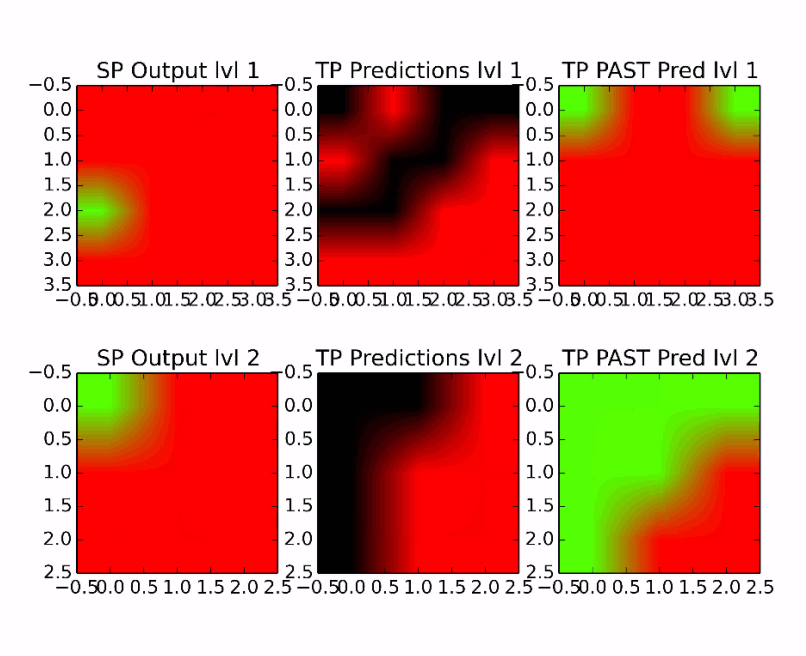
Se disponen de las tres estructuras HTM mencionadas anteriormente: E1: 88_11_8; E2: 100_30_8, y E3: 200_40_8. Estas se han sometido a aprendizajes de una duración con franjas de 15, 30 y 60 minutos por nivel.

**ERROR DE APRENDIZAJE FINAL:** este hace referencia al error cometido durante la experimentación. Este error se calcula usando la media móvil de los errores de la predicción. La ventana es de 30 muestras de error que se aplica a una lista de los errores/anomalías. Se obtiene un vector de errores acumulados, al que a su vez se le calcula la media para tener un único índice de error medio . Esta método se usa para suavizar los posibles picos instantáneos de incremento o disminución del error que se da durante el aprendizaje. Los gráficos de resultados se encuentran en **el Anexo D: Gráficos de Resultados.**

#### 4.3.3.2. Resultados Predicciones

La predicciones nos aportan dos tipos de datos:

1. Videos de la evolución de la detección y predicción de las HTM en forma de gráficos matriciales de la siguiente morfología:




4-31: Ejemplo de morfología de graficado de predicciones

Debido a ello, la única forma de evaluarlo es mediante los archivos de video adjuntos en el soporte digital de esta memoria. **( soporte digital: resultados/predicciones ) .**

2. Gráficos circulares que indican la detección media de cada tono a lo largo de la sesión de predicción. Hay que aclara que los números no tienen porque corresponder directamente a un tono en concreto, sino a un subconjunto de patrones que corresponde a la salida del último nivel de la jerarquía del HTM. Además, quedan representados con la etiqueta “None”, aquellos momentos en los que no se ha podido identificar nada. Por esta razón, se debe considerar que un tono es un perfil de porcentajes diferente a otro. Se disponen de ocho gráficos circulares correspondientes a cada uno de los tonos a predecir. Cada gráfico es el resultado de estar prediciendo durante un periodo de entre 1 y 2 minutos. Durante este tiempo los hablantes pronuncian diferentes frases que son expresadas en el lenguaje propio de cada orador pero bajo un mismo tono emocional. Es decir, durante este tiempo todas frases serán expresadas en de los ocho tonos que se están evaluando.

El orden en que las frases son emitidas es totalmente aleatorio para evitar que la secuencia se repita al pasar a evaluar un entonación diferente.

También hay que indicar que en algunos casos los colores varían ligeramente. Esto es debido a que alguna de las entonaciones no se ha detectado en absoluto, por lo que los colores que se asignan a cada patrón varían.

Estas predicciones están hechas por la versión de HTM con más tiempo de aprendizaje, es decir el 60 minutos por nivel. Cada gráfico se indica en forma de e-x-n.wav , siendo:

- e:la estructura correspondiente con el tiempo de aprendizaje recibido
- x: la entonación de la predicción siendo:

| Emoción | Sigla |
|---|---|
| **Felicidad** | ( h ) |
| **Enfado** | ( a ) |
| **Tristeza** | ( s ) |
| **Asco** | ( d ) |
| **Miedo** | ( f ) |
| **Aburrimiento** | ( b ) |
| **Sorpresa** | ( x ) |
| **Neutro** | ( n ) |

❖ 88_11_8

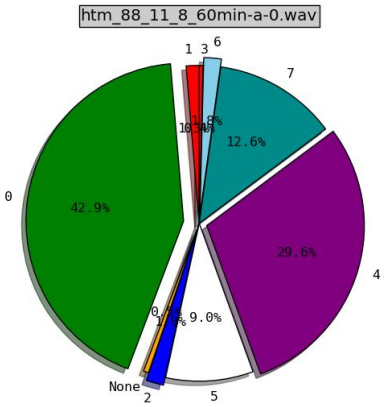



Figura 4-32: Grafico Circular de 88_11_8 60 minutos Enfado

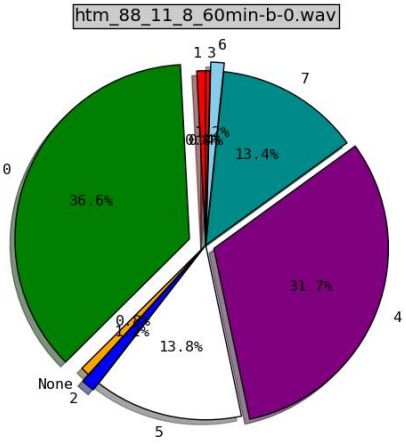



Figura 4-33: Grafico Circular de 88_11_8 60 minutos Aburrimiento

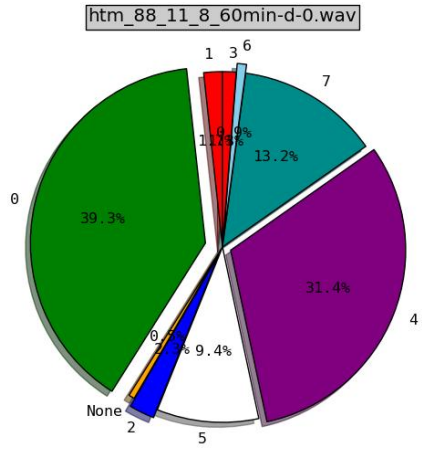



Figura 4-34: Grafico Circular de 88_11_8 60 minutos Asco

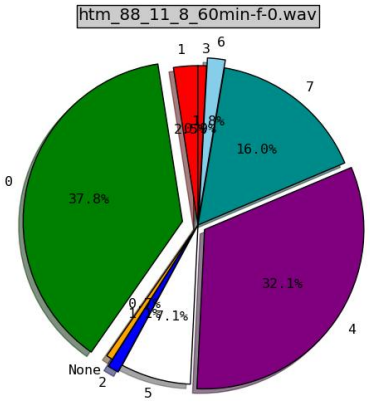



Figura 4-35: Grafico Circular de 88_11_8 60 minutos Miedo

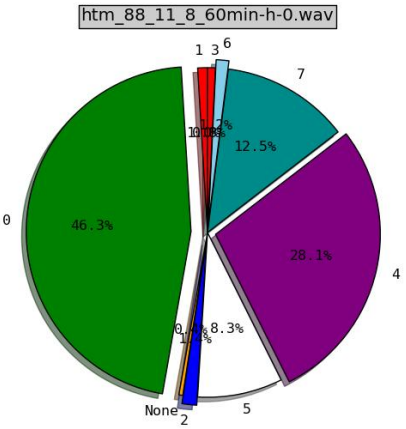



Figura 4-36: Grafico Circular de 88_11_8 60 minutos Felicidad

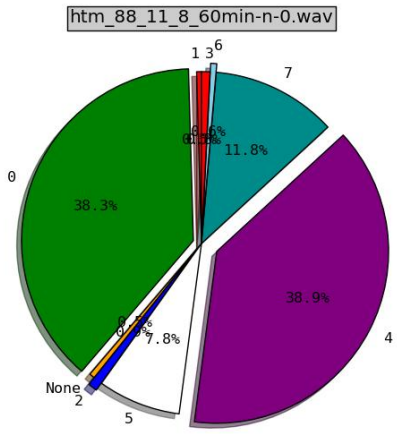



Figura 4-37: Grafico Circular de 88_11_8 60 minutos Neutro

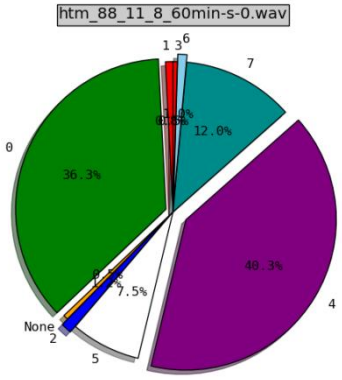



Figura 4-38: Grafico Circular de 88_11_8 60 minutos Triste

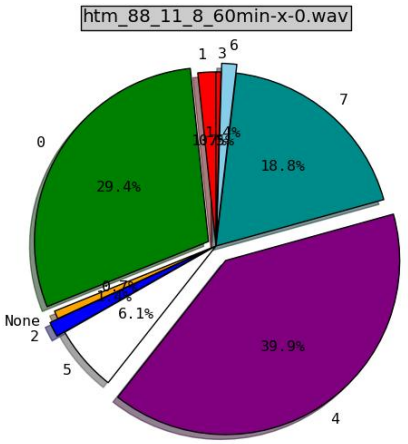



Figura 4-39: Grafico Circular de 88_11_8 60 minutos Sorpesa

- 100_30_8

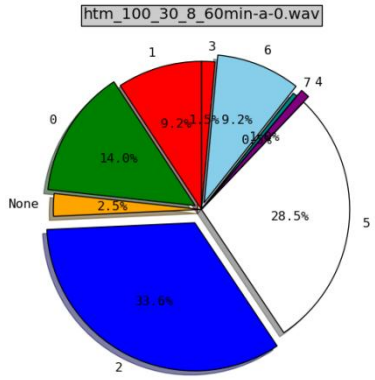



**Figura 4-40: Grafico Circular de 100_30_8 60 minutos Enfado**

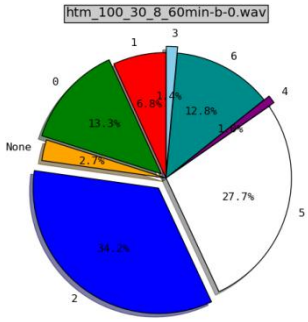



**Figura 4-41: Grafico Circular de 100_30_8 60 minutos Aburrimiento**

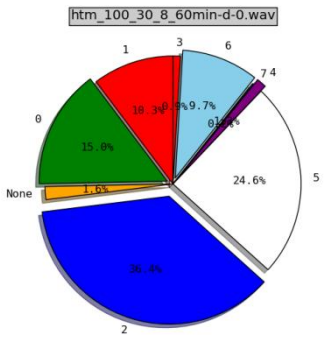



**Figura 4-42: Grafico Circular de 100_30_8 60 minutos Asco**

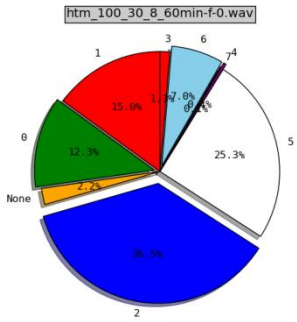



Figura 4-43: Grafico Circular de 100_30_8 60 minutos Enfado

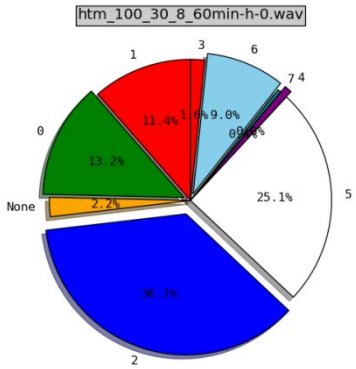



Figura 4-44: Grafico Circular de 100_30_8 60 minutos Felicidad

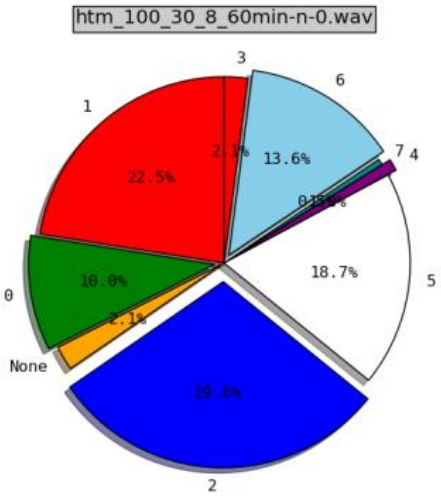



Figura 4-45: Grafico Circular de 100_30_8 60 minutos Neutro

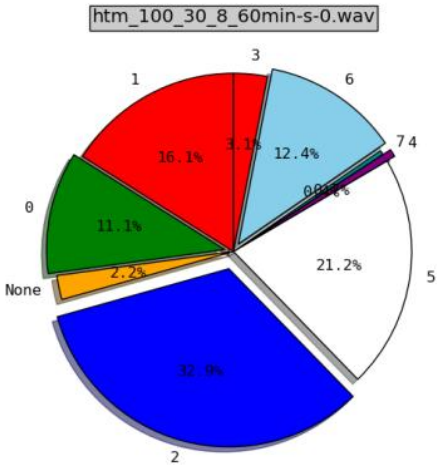



**Figura 4-46: Grafico Circular de 100_30_8 60 minutos Tristeza**

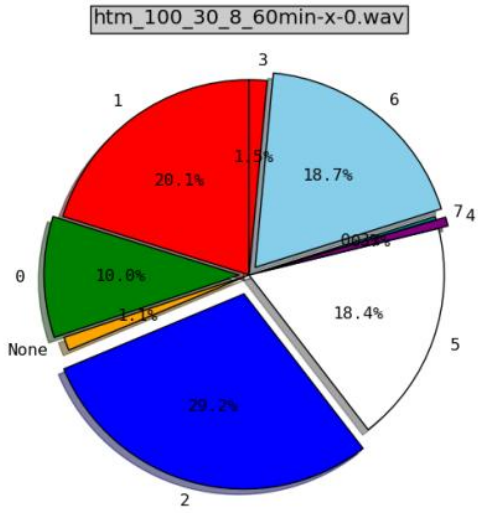



**Figura 4-47: Grafico Circular de 100_30_8 60 minutos Sorpresa**

❖ 200_40_8

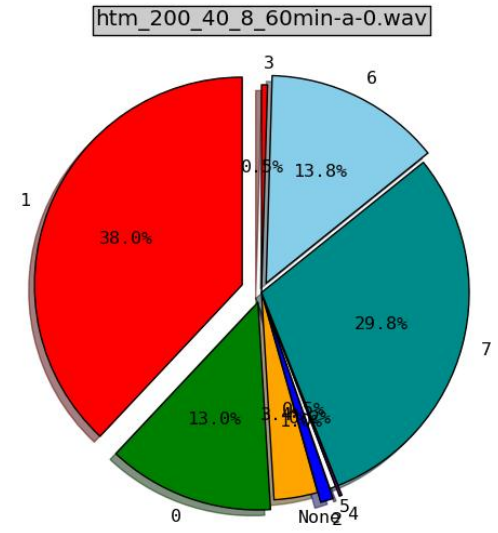



Figura 4-48: Grafico Circular de 200_40_8 60 minutos Enfado

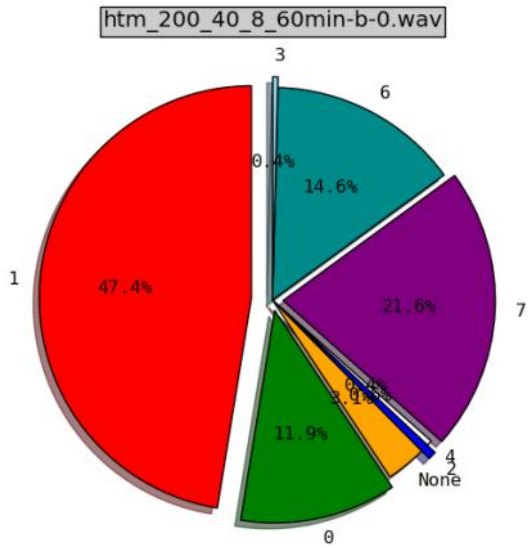



Figura 4-49: Grafico Circular de 200_40_8 60 minutos Aburrimiento

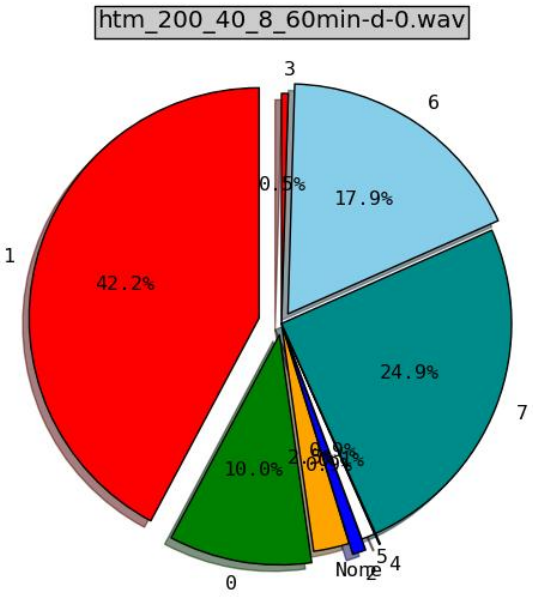



Figura 4-50: Grafico Circular de 200_40_8 60 minutos Asco

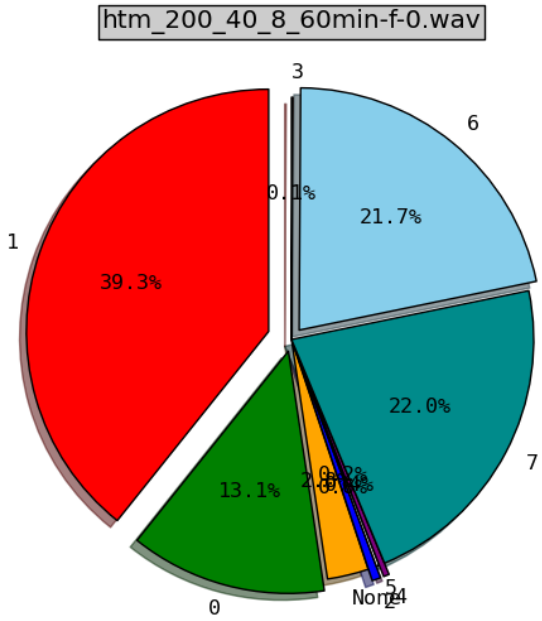



Figura 4-51: Grafico Circular de 200_40_8 60 minutos Miedo

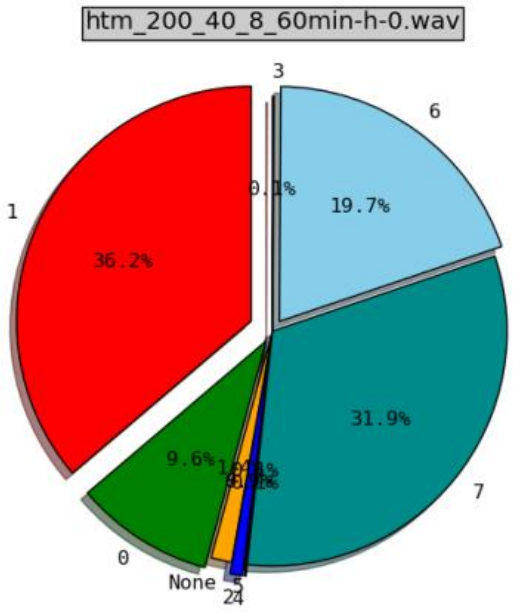



Figura 4-52: Grafico Circular de 200_40_8 60 minutos Felicidad

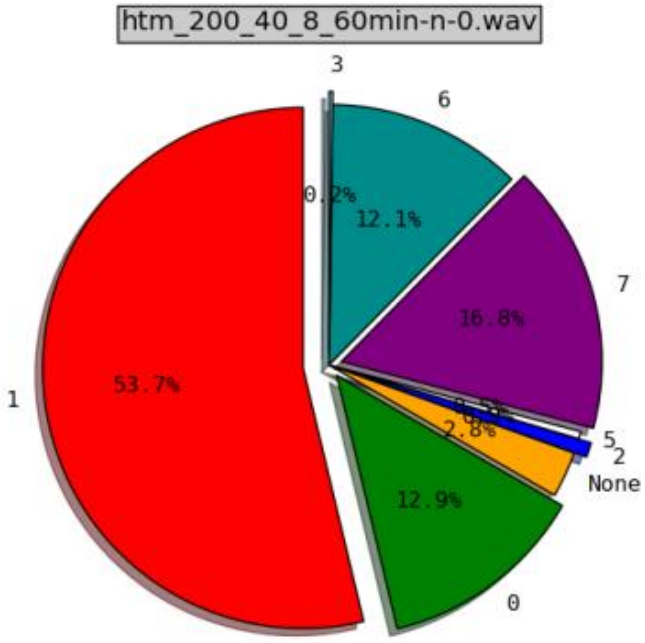



Figura 4-53: Grafico Circular de 200_40_8 60 minutos Neutro

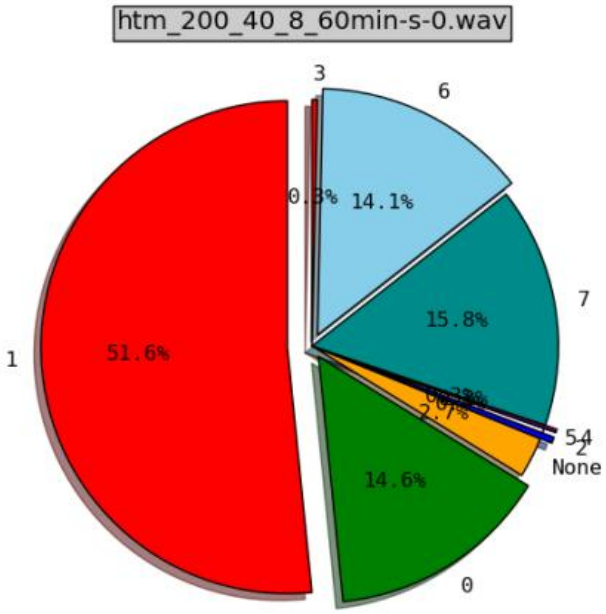



Figura 4-54: Grafico Circular de 200_40_8 60 minutos Tristeza

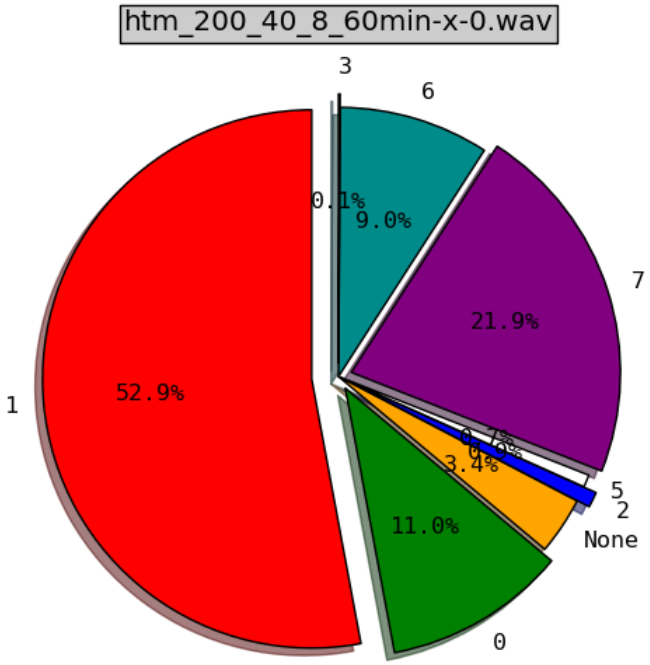



Figura 4-55: Grafico Circular de 200_40_8 60 minutos Sorpresa

## 4.3.4. Conclusiones Experimentales

### 4.3.4.1. Conclusiones referentes al aprendizaje y clustering

Referente al aprendizaje, basándose en los resultados del error al final se observa una disminución sustancial del error a medida que el tiempo de aprendizaje por nivel se aumenta. Notar también que con estructuras más complejas como las 200_40_8 , el error tarda más en disminuir durante el proceso de aprendizaje **( soporte digital: resultados/aprendizaje )** como también en las diferentes pruebas. Mientras que el 88_11_8 en el training de 15 minutos ( Anexo D Fig. 1) ya consigue un error que oscila alrededor del 20% , el 200_40_8 obtiene un error final mayor de oscilación alrededor del 40% ( Anexo D Fig. 7). Este efecto es más patente en training de 30 minutos, en el que el 88_11_8 obtiene un error alrededor del 10% ( Anexo D Fig. 2), el de 200_40_8 aun oscila en torno al 30% ( Anexo D Fig. 8). Este efecto también se observa en el segundo nivel en todas las estructuras. El segundo nivel converge más rápidamente a un error prácticamente nulo. Todo este comportamiento es coherente con el hecho de que a mas patrones a memorizar , mas difícil es la estabilización de patrones. También se concluye que este modelo se puede considerar relativamente fiable al tener errores finales del alrededor de un 20% en el primer nivel y nulos en el segundo nivel (23). Este hecho de mayor estabilidad a medida que se sube en la jerarquía es un factor característico de las HTMs.

Por otro lado, en los datos de *clustering* se observa que el tiempo tiene un efecto de limpieza de las conexiones entre patrones además de reducir el número de patrones memorizados. Se puede observar que comparando la ( Anexo D Fig. 10) correspondiente al 88_11_8_15min y la ( Anexo D Fig. 18) correspondiente al 88_11_8_60min, la cantidad de patrones y conexiones disminuye a la vez que las conexiones tiene más numero de porcentajes altos ( color rojo o amarillo) y menos conexiones débiles ( verdes). También se observa una mejor clusterización respecto a tener *clusters* más equilibrados. De esto se deprende que independientemente de la estructura el tiempo juega un papel primordial a la hora del aprendizaje. Esto concuerda con el cómo los seres humanos aprenden: cuanto más patrones se les somete, mejor lo aprende y más eficientes cerebralmente se vuelven en los conceptos aprendidos. Como también en el inicio del aprendizaje se crean muchas conexiones que equivaldrían a patrones también memorizados, pero a lo largo del tiempo se desechan los conocimientos poco relevantes.

Por tanto concluimos que el comportamiento del sistema de aprendizaje en general se ajusta bastante al comportamiento que cabria esperar de un sistema inspirado biológicamente en el funcionamiento cerebral y HTMs. Se considera adecuado su rendimiento aunque hay que destacar que las agrupaciones en clusters necesitan un mayor refinamiento y ajustes.

#### 4.3.4.2. Conclusiones referentes a la predicción

Se observa en los videos de reconocimiento de patrones y predicciones **( soporte digital: resultados/predicciones )** que normalmente el patrón reconocido está dentro de los predichos en el paso anterior. Sin embargo, siguen habiendo indeterminaciones y se ve la necesidad de mejorar el sistema para añadir más robustez. Estas indeterminaciones se dan en todos los niveles y aproximadamente igual en todas las estructuras, así que es deducible que es cuestión del propio software. Se aprecia una ligera mejora a la hora del funcionamiento de las estructuras más complejas.

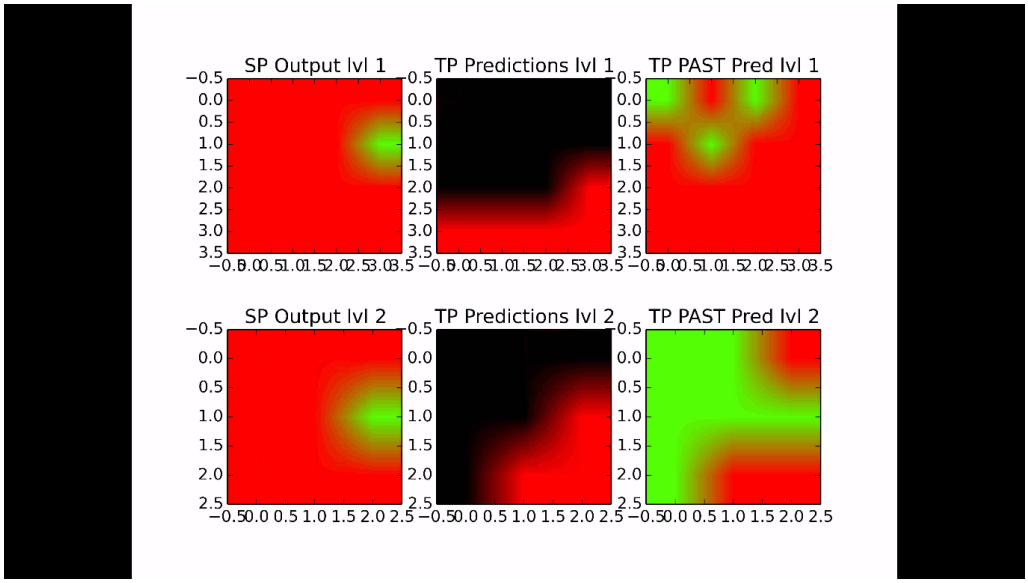



Figura 4-56: Salida de predicciones en estructura 88_11_8 de 60 minutos de aprendizaje

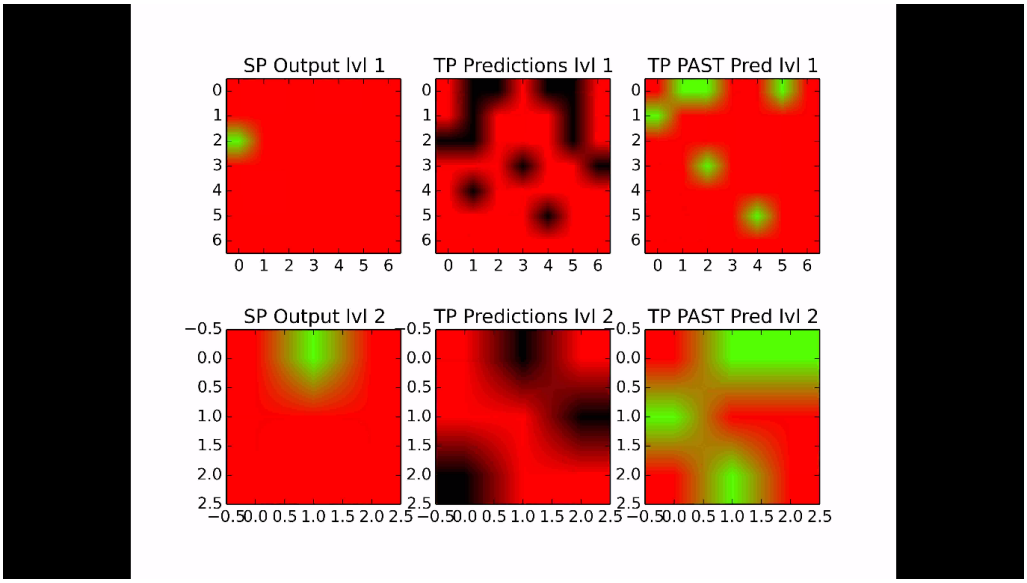



**Figura 4-57: Salida de predicciones en estructura 200_40_8 de 60 minutos de aprendizaje**

Otro aspecto observado es el comportamiento más estable del nivel superior. La variación en el valor del nivel 1 es muy superior a la frecuencia de variación del nivel 2. Esto concuerda perfectamente con la teoría de HTM, ya que indica que a medida que se sube en la jerarquía, la variabilidad en las salidas disminuye. Lo que está ocurriendo es que el nivel inferior quizás coge frecuencias muy cortas mientras que el nivel superior posiblemente coge palabras. Una palabra como casa tiene dos sílabas, por lo tanto, el patrón frecuencia varia dos veces para las sílabas pero el patrón casa sigue activo.

Referente a los gráficos circulares, se observa que no hay un claro vencedor diferente en cada entonación, sino que es un perfil de porcentajes diferentes. Esto se observa en todas las estructuras tanto las sencillas como las más complejas. Esto podría indicar varias cosas. En primer lugar, puede ser que la estructura no esté clasificando bien. En segundo lugar, puede ser que para aprender conceptos complejos como la entonación se necesiten mas niveles que recojan las secuencias de los patrones de salida del nivel 2.

Un ejemplo es 200_40_8, ( Figuras de la Fig. 4-88 hasta la Fig. 4-95 ) donde las diferencias en porcentajes son especialmente destacables, aunque se da también de la misma forma en las otras dos estructuras . Existe una diferencia apreciable entre el “s ( tristeza)”, “n (neutro)” y “x (sorpresa)” y el resto de tonos. Mientras que estos tres el porcentaje del patrón “1” ronda el 52%; en “a (enfado)”, “b (aburrimiento)”, “h (felicidad)” , “f ( miedo)” y “d( asco)” está en torno al 40%.

También existen diferencias entre cada uno de los perfiles de valores entorno al 5-10%. Por ejemplo entre el “f ( miedo)” y el “a (enfado)” existe una proporción inversa entre el patrón **6** y el **7**. Esto cobra importancia, ya que implica que se podría discernir entre alguien enfadado y alguien con miedo, dos sentimientos complicados de diferenciar a veces entre los seres humanos.

Por tanto, como conclusión de la predicción, se observa que el sistema, aun siendo ineficaz y de tener indeterminaciones, introduce unos primeros pasos hacia el aprendizaje de conceptos complejos de forma no supervisada. También es interesante recalcar que esta detección se ha hecho sobre conceptos aprendidos de idiomas distintos, algo de gran interés.

### 4.3.5. Trabajo futuro Experimental y Software

Para un trabajo futuro se debe optimizar el código de todos los paquetes de ROS. Esto permitirá trabajar con estructuras y frecuencias de muestreo de audio mayores. También se debe intentar paralelizar procesas de código, ya que muchas de las operaciones se prestan a ello, lo que reduciría significativamente el tiempo de procesado

Además, es recomendable mejorar el método de clusterización, ya que se han observado incoherencias en la clasificación de patrones.

Por otro lado, se ha de mejorar la adquisición de videos de errores en el aprendizaje o del output en predicción, debido a que por el momento se graban a un cuadro por segundo superior al real, deformando la imagen de la evolución temporal.

Se debe también crear un programa que utilice estos perfiles de gráficos circulares para hacer una detección directa de los tonos.

Además, se ha de incluir soporte para *feedback* sensorial de HTM. Esto consiste en que una vez aprendido un concepto, éste se propaga hacia abajo en la jerarquía y se representa el concepto visual o auditivo de lo aprendido. Un ejemplo es si ha aprendido cual es la entonación alegre, podrá reproducir a través de un altavoz su concepto del mismo.

Finalmente, sería aconsejable crear sistemas de fusión de estructuras HTM para poder volcar conocimientos específicos en un HTM global que contenga todos los conocimientos.

# 5. Análisis Económico

El análisis económico se divide en las siguientes partidas de gastos, cuyo sumatorio contabiliza el precio real de la producción del robot impreso en 3D.

## 5.1. Gastos en Software

Los gastos en la creación del software son el sueldo de un no licenciado en régimen de becario a tiempo completo y del gasto en energía consumida por los equipos informáticos y el movimiento del robot en las pruebas.

- Gasto en personal: 800 euros mensuales, para tanto diseño de software como creación de los modelos digitales en SolidWorks.

$$800 \cdot 6 = 4.800\ euros$$

- Gasto en equipos informáticos: Considerando un periodo máximo de amortización del equipo de 4 años debidos a que se realizan trabajos de tecnología punta (24), y un precio medio de 1.000 euros por unidad.

$$\frac{\left(\frac{1.000}{4}\right)}{2} = 125\ euros$$

Que se considera como el desgaste del ordenador en ese periodo.

Además, si se tiene en cuenta que un ordenador estándar tiene un consumo anual medio de 215 kWh, el consumo tenido en estos 6 meses es de 107,5kWh. Todo ello se traduce en un coste económico, basándose en el 0,1815 euros/kWh, (25) de:

$$107'5 \cdot 0'1815 = 19'5 euros$$

**El total del gasto en software y personal de software es:**

$$4.800 + \mathbf{125} + \mathbf{19'5} = \ \mathbf{4.944'5\ euros}.$$

## 5.2. Gastos Hardware

En gastos de hardware se deben incluir tanto los gastos en componentes y material de impresión, como en energía consumida para su fabricación y sueldo del personal destinado a la fabricación.

### 5.2.1. Componentes

Se puede obtener un listado detallado de las fuentes de compra del hardware en el Anexo A: Lista de Materiales.

| Articulo | Cantidad | Precio (€) | Total (€) |
|---|---|---|---|
| Cámaras Ai-Ball | 2 | 36.6 | 73.2 |
| PCBmotors | 1 | 299 | 299 |
| e-ink Display | 1 | 29 | 29 |
| Servos Tinkerkit | 3 | 7.9 | 23.7 |
| Shield Tinkerkit UNO | 1 | 16.95 | 16.95 |
| Shield Tinkerkit MEGA | 1 | 16.95 | 16.95 |
| Micrófonos | 2 | 1.55 | 3.1 |
| Altavoz | 1 | 8.16 | 8.16 |
| Modulo de sonido | 1 | 33 | 33 |
| ARDUINO UNO | 1 | 24.95 | 24.95 |
| ARDUINO MEGA | 1 | 48.95 | 48.95 |
| USB CODO | 1 | 6.79 | 6.79 |
| USB A/B | 2 | 4.6 | 9.2 |
| Tornillería | 1 | 10.42 | 10.42 |
| Rodamientos radiales | 8 | 8.29 | 66.32 |
| Rodamientos axiales | 1 | 1.66 | 1.66 |
| Sensor Presión | 1 | 8 | 8 |
| Hilo conductor | 1 | 5.45 | 5.45 |

Tabla 2: Lista de Componentes con sus precios

| TOTAL COMPONENTES | IVA | SIN IVA |
|---|---|---|
| 684,8 € | 0,21 € | 540,99 € |

Tabla 3: Coste total de componentes

### 5.2.2. Estructura

La estructura consiste en los costes de producción en 3D de la estructura, lo que corresponde al material usado, consumo de energía en producción y sueldo de operario destinado a ello.

- **Estructura**: Se debe tener presente que no sólo es el material final de la pieza, sino también el destinado a soporte para la producción de la misma y el desechado en modelos fallidos. De media de han producido entre 2-3 unidades de la misma pieza hasta conseguir el modelo final. Como cálculo conservador, se contabilizara como 3 copias del mismo robot para obtener el robot final. Se debe tener en cuenta que en la producción del usuario, sólo se necesita realizar una vez a no ser que se encuentre con problemas en la producción.

  El precio de una bobina de 1kg ABS = **17 euros**

  Material para un robot = 0,7346kg

  Copias Realizadas = 3

  Material Total = 3x0,7346 = 2,2038kg —> 3kg de material

  **Gasto en bobinas = 51 euros en material**

  **Para un robot seria sólo: 0,7346kg. Por lo que el gasto es de 17euros**

- **Consumo:** Extrayendo las emisiones de kg$CO_2$ emitidas (28) a partir de los datos de la Figura 5-1, se seleccionan los datos correspondientes a usar una Makerbot r2 para fabricar una funda de Iphone5s de plástico de 0,02 kg. El resultado es 22,3kg$CO_2$/kgplastico, que es el resultado de 0,446kg$CO_2$/0,02kg de plastico.

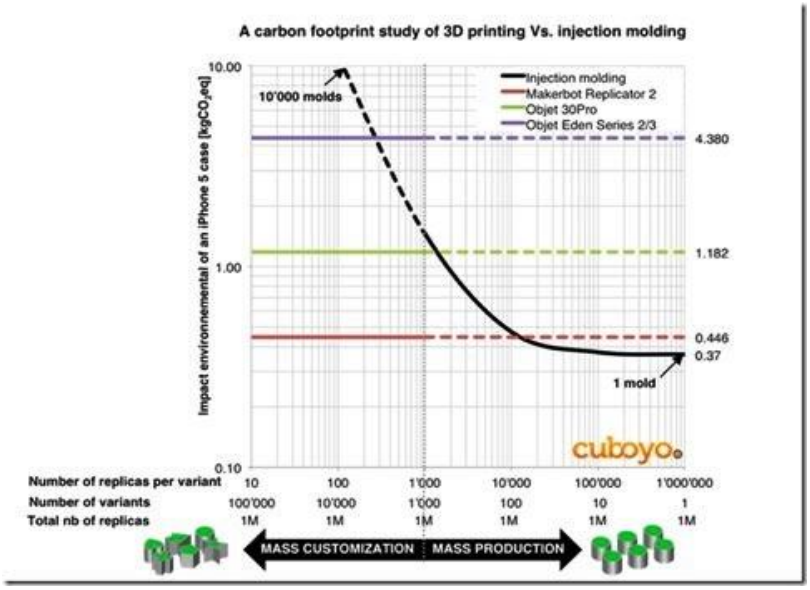



**Figura 5-1:** KgCO2 por unidades

Teniendo en cuenta que un factor de emisor medio de 0,33 $kgCO_2$/KWh (27) y que el precio del KWh es de aproximadamente 0,1815euros/KWh se obtienen los siguientes resultados:

**Total del proyecto ( Equivalente a tres robots ):**

$$\text{EmisioneskgCO2} = \left(\frac{22{,}3\text{kgCO2}}{1\text{kgplastico}}\right) \cdot (2{,}2038\text{KgPlastico}) = \mathbf{49,14kgCO_2}$$

$$\mathbf{49,14kgCO2} \cdot \left(\frac{1\text{kWh}}{0{,}33\text{kgCO}_2}\right) \cdot (0{,}1815\text{euros}/1\text{kWh}) = \mathbf{27,02\ euros}$$

$$\mathbf{Total\ =\ 51\ euros\ en\ material\ +\ 27\ en\ energía\ =\ 78,03\ euros}$$

**Total para un robot:**

$$\boldsymbol{EmisionesKgCO_2 = \left(\frac{22,3kgCO_2}{1kgplastico}\right) \cdot (0,7346kgplastico) = 16,38kgCO_2}$$

$$\mathbf{16,38kgCO2} \cdot \left(\frac{1\text{kWh}}{0,33\text{kgCO2}}\right) \cdot (0,1815\text{euros}/1\text{kWh}) = \mathbf{9euros}$$

$$\boldsymbol{Total\ Estructura = 17\ euros\ en\ material + 9\ en\ energía = 26\ euros}$$

### 5.2.3. Sueldo Operario:

Se debe destinar un sueldo al operario dedicado a la creación de los modelos y montado de la estructura y montado de los componentes. En la fabricación 3D, los conocimientos requeridos son mínimos y no se necesita de ninguna formación específica ya que su labor es preparar la bandeja de impresión, controlar que haya filamento en la máquina suficiente. Una vez terminado, tiene la labor de limpiar la pieza de el material de soporte y endurecer con acetona los orificios y estructuras pequeñas que sobresalen o que necesitan de cierto refuerzo mecánico. En el montaje no se requieren tampoco de ninguna habilidad especifica ni uso de herramientas no convencionales.

Se considera un único pago de **400 euros** por medio mes de trabajo continuo.

### 5.2.4. Coste Global HARDWARE

EL Coste global de hardware ascendería a :

$$684,8\ en\ componentes + 78,03\ en\ estructura + 400\ en\ operario = \mathbf{1.162,8\ euros}$$

## 5.3. Coste Global y Análisis

El coste global del proyecto es de:

$$1.162,8\ euros\ de\ hardware + \mathbf{4.944'5}\ euros\ de\ software = \mathbf{6.107,3\ euros}$$

De este total, sólo una fracción que se denominan **gastos de creación** son necesarios para los próximos usuarios que deseen usar la estructura hardware y software, ya que el software es opensource en la medida de lo posible y los modelos digitales de hardware en su

mayoría también lo son. Los **gastos de creación** incluyen los gastos en componentes, los gastos en material de impresión y los gastos en energía para la impresión.

**Gastos_de_creación = 684,8 en componentes + 26 euros en materialABS/Energia un robot**

**Gastos_de_creación = 710,8 euros**

Se puede observar que comparado con otros robots de investigación de relativamente bajo coste ampliamente usados como Nao (9.500 euros Anexo A: Lista de Materiales, Precios Robots) o Darwin (1.1000 euros Anexo A: Lista de Materiales, Precios Robots ); Kodama a nivel de investigación cognitiva dispone de elementos como tacto, estereovisión, stereoaudición y sistema de HRI en forma de pantalla del cual los anteriormente mencionados no disponen, y a un precio más bajo, además de ser modular, completamente opensource y disponer del software cognitivo básico necesario.

Un especial mención a Qbo al ser mucho as asequible (3.795 euros Anexo A: Lista de Materiales, Precios Robots ) y al tener un diseño algo más simple. Sin embargo, no dispone del elenco sensorial adecuado para investigación cognitiva. Si es cierto que los robots anteriormente mencionados son capaces de auto locomoción y de sistemas totalmente integrados en producción. Resulta difícil la comparación al no existir un robot de este rango de precio destinado al aprendizaje cognitivo integro, con una filosofía opensource y que se pueda imprimir en 3DPrinters.

Y robots mejor dotados para el aprendizaje cognitivo, como el iCub, tienen unos precios oscilando alrededor de los 250.000 euros (Anexo A: Lista de Materiales, Precios Robots ), lo que los coloca en un rango de precios distinto a Kodama.

De esto se deduce, que con las apropiadas mejoras y esfuerzos en él, la integración del hardware a la vez que una interfaz software gráfica, Kodama podría llegar a convertirse en un producto con cierta cuota de mercado en el mundo de la investigación cognitiva de bajo presupuesto.

# 6. Estudio Medioambiental

Se deben considerar el impacto medioambiental del hardware y del software.

## 6.1. Software

Referente al impacto medioambiental en el desarrollo de software, se puede considerar el presente proyecto como un proyecto de una empresa TIC, por tanto está sujeto a las posibles emisiones como tal. En este caso es un proyecto realizado mayoritariamente por un sólo empleado, trabajando a jornada completa durante aproximadamente 6 meses .

Los posibles residuos generados son las emisiones indirectas debidas al consumo de energía. Si se tiene en cuenta que un ordenador estándar tiene un consumo anual medio de 215 kWh (28) y una producción de alrededor de 0,33KgCO2/KWh (27) en España, en un uso de medio año esto se traduce en:

$$\left(\frac{215}{2}\right) \cdot 0{,}33 \; = \; \mathbf{35{,}475 \; KgCO_2 \; generados}$$

## 6.2. Hardware

Para la evaluación de este apartado, se ha calculado por un lado la estructura impresa en 3D y por el otro, los componentes de hardware como electrónica y baterías.

### 6.2.1. Estructura impresa en 3D

Podemos dividir sus efectos en tres secciones:

**1. Escala de producción**

La impresión 3D en comparación con otros métodos de producción en plástico, como la inyección en moldes , es menos contaminante en tiradas cortas, pero mucho más en tiradas largas. En el caso del robot del proyecto, ha sido más eficiente desde el punto energético el producirlo por impresión 3D, ya que sólo es necesaria una tirada, más aquellas pruebas o intentos has conseguir la pieza deseada (28).

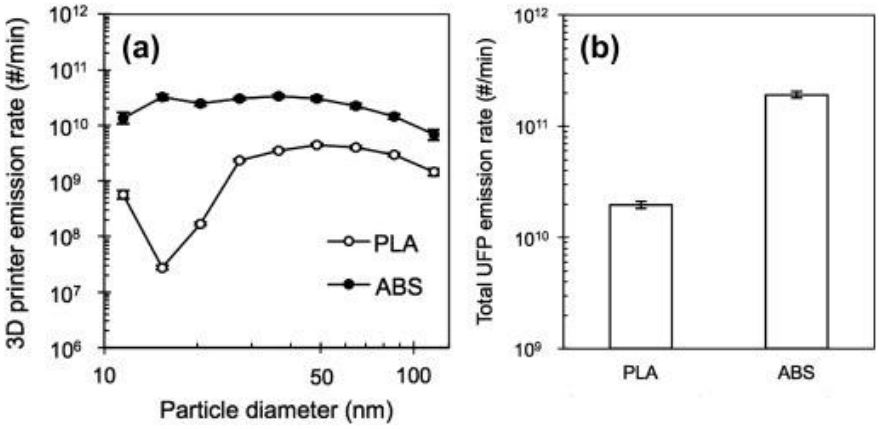
## 2. Material y Consumo energético

Hay un gasto de material en los elementos de fabricación fallidos. En este caso, se han realizado de media tres unidades de cada pieza debido a errores en los diseños preliminares o en fallos de impresión. En el apartado del análisis económico, se han calculado las emisiones totales de KgCo2 generadas por la impresión de todas las piezas necesarias para el proyecto. En total se generan alrededor de **49,14kg$CO_2$.**

Se ha consumido una significativa cantidad de energía menos, ya que no hubo necesidad de crear moldes para su producción, tal y como lo necesita la inyección. Al ser impresas en 3D, las piezas que se generan pueden llegar a ser hasta un 50% más ligeras que sus homólogos de inyección, lo que reduce significativamente el uso de material.

Al producir en 3D ha sido posible generar las piezas en el entorno local con una disminución en los costes de transporte.

Hay que contabilizar las emisiones en la producción de monóxido de carbono, cianuro de hidrogeno y partículas ultra finas (UFP) de diámetros inferiores a 100nm. Esto es equivalente a las emisiones generadas por cocinar en un horno a media potencia (28).




Como se muestra en las gráficas las emisiones, hubiesen sido menores con el uso de PLA. También el consumo energético se hubiese reducido, ya que el PLA no necesita cama caliente de impresión, a diferencia del ABS.

**3. Longevidad**

Al ser un diseño modular, este robot ha reducido el consumo de material, debido a que en los casos de error en producción o diseño, no se tuvo que repetir grandes secciones de pieza, sino solo el modulo defectuoso. Esto se puede aplicar a la vida de todos los robots que se construyan, ya que se verá alargada su vida mucho más que un diseño cerrado. La razón principal es que se podrá seguir actualizando los elementos necesarios sin tener que desechar el robot entero. Esto no es únicamente aplicable a la estructura sino a todos los elementos de hardware.

### 6.2.2. Electrónica

En este apartado entran las cámaras, el PCB-Motor, los micrófonos, altavoces, pantalla, servos y electrónica de control como ARDUINO o computadora. Según el Real Decreto Real Decreto 208/2005, entran en la categoría de RAEES en a**paratos electrónicos de Consumo** y en **equipos de informática y telecomunicaciones.**

El impacto medioambiental de la generación de estos productos dependerá directamente de los métodos de fabricación de los mismos. En este caso, no existían alternativas más sostenibles a los productos escogidos y por eso incrementa la necesidad de que los componentes sufran el menor deterioro posible para aumentar su vida y reducir la necesidad de su reemplazo.

En cuanto a su desechado, aunque no entra en el análisis de impacto medioambiental de la realización del proyecto, apuntar que se deberán tratar según la normativa vigente para su desechado según DIRECTIVA 2012/19/UE DEL PARLAMENTO EUROPEO (29) .

### 6.2.3. Baterías

Para el funcionamiento de las cámaras Ai-Ball se usaron 2 pilas CR2.

Las baterías usadas en las cámaras Ai-Ball son CR2, las cuales deben ser recicladas en puntos adecuados. En este caso, las baterías usadas son no recargables, lo que añade mayor contaminación a corto plazo.

# 7. Agradecimientos

Debo agradecer a toda mi familia, tanto a los viejos como a los nuevos miembros, por el apoyo incondicional ante esta locura que es la robótica.

Debo agradecer a mis tutores Guillem Alenyà y Ricardo Téllez su guía y consejos sobre la realización técnica y teórica de este proyecto.

También a todo el equipo de IRI en especial a Patrick Grosch por su guía y participación en la fabricación de la estructura 3D y a Sergi Foix por ayudar en la generación del paquete gsound_pkg.

Por último al equipo NUPIC por haber sido tan diligente en la resolución de dudas técnicas.

# Bibliografía

1. **IFR Statistical Department.** worldrobotics.org. [En línea] 30 de Septiembre de 2015. http://www.worldrobotics.org/uploads/tx_zeifr/Charts_PC_09_30_2015.pdf.

2. **Paul, Annie Murphy.** ted.com. [En línea] Noviembre de 2011. https://www.ted.com/talks/annie_murphy_paul_what_we_learn_before_we_re_born/transcript.

3. **Hawkins, Jeff.** *On Intelligence.* New York : Holt Paperbacks, 2005. págs. 85-176.

4. **Breazeal, Cynthia.** *Designing Sociable Robots.* 2002. págs. 51-60, 81-100, 173-180.

5. **37 Signals.** *Getting Real.* s.l. : 37 Signals, 2006. págs. 25-36.

6. **Paul Ekman Group.** paulekman.com. [En línea] http://www.paulekman.com/facs/.

7. **Wainwright, Oliver.** theguardian.com. [En línea] 12 de Agosto de 2014. http://www.theguardian.com/technology/shortcuts/2014/aug/12/why-apple-uses-picasso-bull-teach-minimalist-design.

8. **Kaist.** kaist. [En línea] http://sdac.kaist.ac.kr/research/index.php?mode=area&act=ear.

9. **Group of Robotics and Cognitive Systems.** robotics.pme.duth. [En línea] http://robotics.pme.duth.gr/research/topics/stereo-vision/.

10. **PCBMotor.** *How to reduce motor size by integrating accurate, low cost piezo .* Denmark : Pcbmotors, 2012. white paper.

11. **Pressure Profile Systems, Inc. .** pressureprofile.com. [En línea] http://static1.squarespace.com/static/53836bf1e4b011aa8ac0ffb9/t/53c85b3ae4b052122aacf107/1405639482642/PPS+_RoboTouch_SpecSheet.pdf.

12. **C. SCHISSLER, D. MANOCHA.** *GSOUND: INTERACTIVE SOUND PROPAGATION FOR GAMES.* The University of North Carolina at Chapel Hill.

13. **cortical.io.** cortical.io. [En línea] http://www.cortical.io/technology_representations.html.

14. **Prime, Kheecha.** youtube Stereo Sound Test. [En línea] https://www.youtube.com/watch?v=RNrF1I5mf3c.

15. **Dileep, George.** *How the brain might work: A Hierarchical and temporal model for learning and recognition.* 2008. págs. 1-130.

16. *Markov Graphs.* **Frank, Ove y Strauss, David.** Journal of the American Statistical Association, Vol. 81, No. 395. (Sep., 1986), pp. 832-842..

17. **python.org.** python.org. [En línea] https://docs.python.org/2/library/pickle.html.

18. **scikit-learn developers.** scikit-learn.org. [En línea] 2014. http://scikit-learn.org/stable/modules/clustering.html#.

19. **mnrART.** deviantart.com. [En línea] 2011. http://mnrart.deviantart.com/art/facial-expressions-270039582.

20. **embeddedartists.** embeddedartists. [En línea] http://www.embeddedartists.com/sites/default/files/support/displays/epaper/Epaper_arduino.pdf.

21. **Schlesinger, Angelo Cngelosi & Matthew.** *Develomental Robotics.* Cambridge, Massachusetts : MIT Press, 2015. Foreword.

22. **Felix Burkhardt, Miriam Kienast, Astrid Paeschke and Benjamin Weiss.** expressive-speech.net. [En línea] http://www.expressive-speech.net/.

23. **Giovanni Costantini, Iacopo Iadarola, Andrea Paoloni, Massimiliano Todisco.** *EMOVO Corpus: an Italian Emotional Speech Database.* Electronic Engineering, University of Rome. Rome, Italy : s.n.

24. **Haq, Philip Jackson and Sanaul.** surrey.ac. [En línea] 2 de Abril de 2015. http://personal.ee.surrey.ac.uk/Personal/P.Jackson/SAVEE/.

25. **Thomas D'Arcy, Christopher Stanton, and Anton Bogdanovych.** *Teaching a robot to hear: a real-time on-board sound classi.* MARCS Institute, University of Western Sydney. Sydney : s.n., 2013.

26. **asesor-contable.** asesor-contable. [En línea] http://asesor-contable.es/tablas-amortizacion-2015/.

27. **tarifasgasluz.** tarifasgasluz. [En línea] http://tarifasgasluz.com/faq/precio-kwh-2015.

28. **impresiontresde.** impresiontresde.com. [En línea] http://impresiontresde.com/blog/una-nueva-revision-al-impacto-ambiental-de-la-impresion-3d/.

29. **Gobierno de España, Ministerio de Industria, Energia y Turismo.** *FACTORES DE EMISIÓN DE CO2 Y COEFICIENTES DE PASO A ENERGÍA PRIMARIA Y DE DIFERENTES FUENTES DE ENERGÍA FINAL CONSUMIDAS EN EL SECTOR EDIFICIOS EN ESPAÑA.* 2014.

30. **leantricity.** leantricity. [En línea] 11 de Julio de 2012. http://leantricity.es/cuanta-energia-gasta-un-ordenador-aproximaciones/.

31. **Diario Oficial de la Unión Europea.** *DIRECTIVA 2012/19/UE DEL PARLAMENTO EUROPEO Y DEL CONSEJO, sobre residuos de aparatos eléctricos y electrónicos (RAEE).* 2012.

32. **Carl Schissler, Dinesh Manocha.** gamma.cs. [En línea] http://gamma.cs.unc.edu/GSOUND/.

33. Impacto ambiental en las Pymes TIC, Manual de Buenas Practicas. [En línea] Ministerio de Agricultura, Alimentacion y Medioambiente, Gobierno de España.