



UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Linares

Trabajo Fin de Máster

**PROCESADO AUDIO-VISUAL PARA
IMPLEMENTACIÓN DE UN MECANISMO
ATENCIONAL**

Alumno: Antonio Martínez Colón

Tutores: Prof. D. Pedro Jesús Reche López
Prof. D. José Manuel Pérez Lorenzo

Depto.: Ingeniería de Telecomunicación

septiembre, 2017



UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Linares

Trabajo Fin de Máster

PROCESADO AUDIO-VISUAL PARA IMPLEMENTACIÓN DE UN MECANISMO ATENCIONAL

Alumno: Antonio Martínez Colón

Tutores: Prof. D. Pedro Jesús Reche López
Prof. D. José Manuel Pérez Lorenzo

Depto.: Ingeniería de Telecomunicación

Firma del autor

Antonio Martínez Colón

Firmas de los tutores

**PEREZ
LORENZO
JOSE
MANUEL -
238077985**

Firmado digitalmente por
PEREZ LORENZO JOSE
MANUEL - 238077985
Nombre de reconocimiento
(DN): c=ES,
serialNumber=IDCES-238077
985, givenName=JOSE
MANUEL, sn=PEREZ
LORENZO JOSE MANUEL -
238077985
Fecha: 2017.09.13 10:16:41
+02'00'

**RECHE
LOPEZ
PEDRO
JESUS -
26226792F**

Firmado digitalmente por
RECHE LOPEZ PEDRO JESUS
- 26226792F
Nombre de reconocimiento
(DN): c=ES,
serialNumber=IDCES-26226
792F, givenName=PEDRO
JESUS, sn=RECHE LOPEZ,
cn=RECHE LOPEZ PEDRO
JESUS - 26226792F
Fecha: 2017.09.13 10:42:19
+02'00'

TABLA DE CONTENIDO

1	Resumen.....	3
2	Abstract.....	4
3	Introducción	5
4	Objetivos	14
5	Materiales y Métodos.....	15
5.1	Procesado de audio	15
5.1.1	Adquisición y muestreo de las señales de audio	15
5.1.2	Análisis de Fourier	17
5.1.3	Short Time Fourier Transform (STFT)	18
5.1.4	Modelo de señal.....	20
5.1.5	Time Difference of Arrival (TDOA)	21
5.1.6	Correlación cruzada generalizada (GCC).....	24
5.1.7	Steered Response Power (SRP)	25
5.1.8	Steered Response Power PHAT (SRP-PHAT).....	27
5.1.9	Hardware de procesado de audio.....	31
5.2	Procesado de vídeo	35
5.2.1	Detección de rostros	35
5.2.2	Estructura del problema de detección de rostros	37
5.2.3	Enfoques para la detección de rostros	38
5.2.4	Seguimiento de rostros	40
5.2.5	Procesado visual con OpenCV	41
5.2.6	Hardware de captura de vídeo.....	55
5.3	Fusión Audio - Visual	57
5.3.1	Hardware fusionado	60
6	Resultados y Discusión	62
6.1	Procesado de audio	62
6.1.1	Simulación Off-line	62
6.1.2	Entorno acústico para pruebas de SLAV	84

6.1.3	On-line(real time)	85
6.2	Procesado de vídeo	87
6.3	Fusión audio-visual	87
7	Conclusiones	90
8	Referencias bibliográficas.....	91

1 RESUMEN

El proyecto desarrollado que va a ser explicado mediante esta memoria, tiene como fin localizar la posición de hablantes dentro de una habitación o sala.

Una vez se obtiene la posición del hablante mediante procesado de audio, se dirige hacia ésta la atención, es decir, la vídeo cámara apuntará hacia esa dirección. Y será entonces cuando, mediante procesado de vídeo, se ajustará el ángulo de elevación de la cámara para que apunte hacia la cara del hablante. Por otro lado, el sistema desarrollado reproducirá en tiempo real el vídeo capturado por la cámara. Por lo que, mediante la reproducción de este flujo de vídeo será posible ofrecer la sensación de primera persona a una persona que no esté presente en la sala o habitación en la que se encuentra el sistema de localización.

Así, el proyecto está compuesto de los siguientes módulos o fases:

- Localización de fuentes sonoras mediante el procesado audio.
- Localización de caras de personas mediante el procesado de vídeo.
- Fusión de ambos módulos para la detección final.

Por todo esto, algunas aplicaciones para las que este proyecto puede ser útil serían: actuar como un módulo de un sistema de robótica, para conseguir una mejor interacción robot/persona. Cuando la persona se dirija al robot, éste le mirará a la cara, humanizando así el robot.

También puede ser interesante su uso en un sistema de videoconferencia con múltiples hablantes. El sistema de localización detecta la posición de la persona que está hablando en cada instante y dirige la cámara hacia ésta.

Los objetivos inicialmente marcados con este trabajo han sido alcanzados.

2 ABSTRACT

The developed Project which is explained in this document, aims to locate the position of speakers within a room.

Once the position of the speaker is obtained through audio processing, the attention is directed to this point, i.e. the video camera will point in that direction. And then it will adjust, through video processing, the angle of elevation of the camera to point to the face of the speaker. On the other hand, the developed system will reproduce in real time the video captured by the camera. Therefore, by reproducing this video stream will be possible to offer the sensation of first person to a person who is not present in the room where the localization system is.

Thus, the project is composed of the following modules or phases:

- Localization of sound sources through audio processing.
- Localization of human faces through video processing.
- Fusion of both systems for a final detection.

Some applications for which this project can be useful would be: acting as a module of a robotic system, to achieve a better robot / human interaction. When someone speaks to the robot, it will look at the face of this person, humanizing the robot.

It can also be interesting to use it in a multi-speaker videoconference system. The location system detects the position of the person who is speaking at each moment and directs the camera towards it.

The objectives initially marked with this work have been achieved.

3 INTRODUCCIÓN

La capacidad humana para distinguir cuando un objeto sonoro está cerca o lejos de nosotros se desarrolla completamente cuando tenemos unos pocos meses de edad [1]. De hecho, el desarrollo de los mecanismos de localización utilizados por el sistema auditivo humano tiene lugar antes de un año de edad [2]. La localización de fuentes sonoras es posible porque el cerebro humano analiza todas las señales que llegan a través de nuestros oídos, usando sutiles diferencias en intensidad y otras señales espectrales y de tiempo para reconocer la dirección de una o incluso varias fuentes de sonido [3; 4].

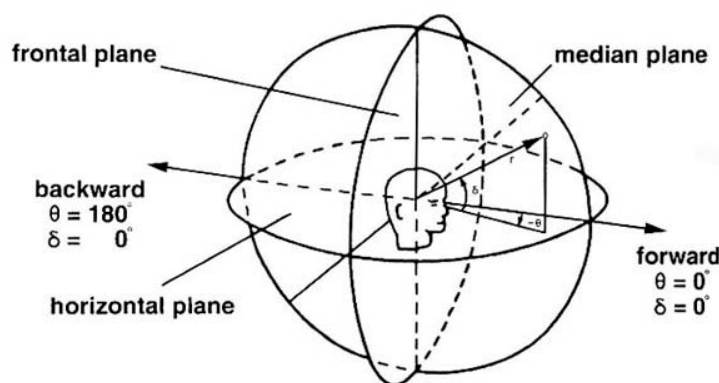


Figura 3.1 Localización sonora

La localización de fuentes sonoras no requiere un esfuerzo especial para un ser humano. No ocurre igual con las máquinas, para las que la localización de fuentes sonoras en una habitación es un proceso complicado, ya que no todos los objetos sonoros tienen las mismas propiedades espectrales. Éstos ocurren en diferentes instantes de tiempo y en diferentes posiciones en el espacio. Además, el proceso está fuertemente afectado por las reflexiones. Las reflexiones acústicas dominan la percepción del sonido en una sala modificando las características espaciales de las fuentes percibidas.[5]

Para la realización de este proyecto ha sido necesario conocer el estado del arte referente a la localización de fuentes sonoras mediante el procesamiento de audio, en inglés conocido como Sound Source Localization (SSL). SSL ha sido objeto de muchos estudios e investigaciones por parte de la comunidad científica en las últimas décadas. De hecho, la localización precisa de fuentes sonoras en condiciones de alto nivel de ruido y reverberación sigue siendo un reto. Los mayores objetivos dentro de la localización de fuentes sonoras es conseguir una alta precisión en la localización, funcionamiento en tiempo real, que el número de micrófonos necesarios para la detección sea razonable y todo ello con unos recursos de computación limitados.[5]

Estos sistemas de localización son útiles en sistemas donde el audio es capturado por un array de micrófonos, donde es necesario localizar varias fuentes de sonido incluso en condiciones adversas (cuando existe una baja relación señal a ruido y una alta reverberación). Algunos ejemplos de su uso serían:

- Sistemas de localización de robots autónomos:

Como por ejemplo en [6], donde se presenta un método de localización de fuentes sonoras robusto en un espacio tridimensional con un array de 8 micrófonos. El método se basa en la estimación del retardo en el tiempo de llegada. Los resultados muestran que un robot móvil puede localizar en tiempo real diferentes tipos de fuentes de sonido en un rango de 3 metros.

- Videoconferencia:

Como es el caso de [7] donde, se presenta un sistema de señalización para una cámara controlado por cálculos en tiempo real de localizaciones de fuentes sonoras, con un array de micrófonos. Mediante un método de calibración cuyo enfoque es el aprendizaje de la asignación de los tiempos entre los pares de micrófonos al pan y la inclinación asociados de una cámara PTZ (Pan Tilt Zoom), para un punto dado del espacio. Por lo que, no es necesario aprender explícitamente las posiciones de los micrófonos y la cámara.

Además los sistemas de localización sonora son usados en: videojuegos [64], sistemas remotos de vídeo vigilancia [65], adquisición de señal en sistemas de manos libres [66], etc. Todas estas aplicaciones requieren la localización de una o más fuentes acústicas. Muchas aplicaciones de procesamiento de audio pueden ser mejoradas si se conoce la localización espacial de la fuente sonora de interés.

Muchos sistemas SSL asumen que las fuentes sonoras se distribuyen en un plano horizontal [9; 10]. Esta suposición simplifica el problema de SSL en casi todos los métodos. Por ejemplo, en las aplicaciones de videoconferencia se supone que todos los hablantes hablan a la misma altura, lo cual puede considerarse cierto, pero el hablante u otros asistentes pueden desviar el sonido entre el hablante principal y el array de micrófonos. Además, en la mayoría de los métodos SSL dominantes, el costo computacional requerido suele ser muy alto, incluso cuando se supone que las fuentes están en el mismo plano [11]. Algunos de estos métodos SSL han sido modificados para cubrir un espacio tridimensional obteniendo un costo computacional muy alto. Por lo tanto, el desarrollo de métodos de localización 3D con baja complejidad sigue siendo una tarea muy difícil.

Los algoritmos de SSL pueden ser divididos según su enfoque en indirectos y directos [12]. Los enfoques indirectos generalmente siguen un procedimiento de dos pasos: primero estiman la diferencia de tiempo de llegada, en inglés Time Difference Of Arrival (TDOA) [13] entre los pares de micrófonos y después, estiman la posición de la fuente basándose en la geometría del array y los retrasos estimados. Por otro lado, los enfoques directos realizan la estimación TDOA y la localización de la fuente en un solo paso explorando un conjunto de ubicaciones de la fuente candidata y seleccionando la posición más probable como una estimación de la ubicación de la fuente.

Uno de los algoritmos más conocidos para la localización de fuentes sonoras en entornos con ruido y reverberantes es Steered Response Power-Phase Transform (SRP-PHAT). Este algoritmo comúnmente interpretado como un enfoque basado en beamforming (conformación de haces), el cual es una forma espacial de filtrado y es usado para distinguir entre las propiedades espaciales de una señal objetivo y el ruido de fondo. Mediante el beamforming se busca la posición de la fuente candidata que maximiza la salida de un beamformer delay-and-sum direccionado. SRP-PHAT es de tipo directo y consigue una muy buena precisión en la localización pero su mayor inconveniente es su elevado coste computacional, lo que hace que su ejecución en tiempo real sea difícil, si el área de búsqueda es muy fina. Por ello, diversas modificaciones y optimizaciones de dicho algoritmo han sido propuestas. En [14] donde se propone la utilización de una restricción estocástica de la región de mallado, en inglés Stochastic Region Contraction (SRC) para hacer práctico el SRP. En [15] se presenta la otra variante del algoritmo: Coarse-to-Fine Region Contraction (CFRC), que consiste básicamente en realizar un mallado grueso, con bajo coste computacional, en un área amplia y posteriormente realizar un mallado más fino, en un área más reducida de búsqueda dentro del espacio de búsqueda inicialmente mallado de forma gruesa. Otra mejora de SRP-PHAT es propuesta en [16] donde se introduce una estrategia efectiva que extiende el SRP-PHAT convencional con el objetivo de considerar el volumen que rodea las ubicaciones discretas de la rejilla espacial. Este enfoque realiza una exploración completa del espacio muestreado en lugar de calcular el SRP en posiciones espaciales discretas, aumentando su robustez y permitiendo una cuadrícula espacial más gruesa. Para ello, la función de correlación cruzada generalizada (GCC) correspondiente a cada par de micrófonos debe acumularse correctamente de acuerdo con la configuración de micrófonos definida.

El algoritmo SRP-PHAT se utiliza también en sistemas de localización de robótica, pero como se ha visto previamente existe el problema de la ejecución en tiempo real por ello en [7] se propone un método de agrupación del espacio de búsqueda diseñado para acelerar el algoritmo de localización de fuentes sonoras basado en SRP-PHAT para

robots domésticos inteligentes, equipados con arrays de micrófonos de pequeña escala.

Para la localización de fuentes sonoras se puede utilizar también la diferencia en intensidad como en [17] donde se comparan cinco métodos para la estimación de la dirección de llegada basada en la diferencia de intensidad sonora, utilizando señales reales de una sala de conciertos. Sin embargo, para la realización de este proyecto se ha implementado solo la diferencia en los retardos de llegada.

Los métodos SSL pueden ser una herramienta muy útil para los sistemas de interacción hombre-máquina, ya que la información espacial proporcionada por los algoritmos de localización es esencial para imitar la habilidad humana natural para discriminar la posición de un hablante. Pero en este proyecto se va a simular además, la habilidad humana para la detección de caras para poder así dirigir mejor la atención hacia el hablante.

La retina humana (la parte del ojo que convierte la luz recibida en señales electroquímicas) tiene alrededor de 100 millones de células sensibles a la luz. Por tanto, las imágenes de la retina contienen una enorme cantidad de información. En las tareas de procesamiento visual de alto nivel, como por ejemplo reconocer objetos, estimar tamaños y distancias, o calcular la trayectoria de un objeto en movimiento, el cerebro probablemente no usa todos los datos disponibles, ya que no parece tener suficientes neuronas dedicadas a ello. Así que los científicos han asumido durante mucho tiempo que el cerebro debe resumir de alguna manera el contenido de las imágenes de la retina, reduciendo así la cantidad de información de éstas antes de transferirlas a procesos mentales de mayor nivel. [18]

La mayoría de los modelos sobre el reconocimiento humano de objetos asumen que lo primero que hace el cerebro con una imagen de la retina es identificar los bordes, es decir los límites entre las regiones con diferentes propiedades de reflexión de la luz, y ordenarlos de acuerdo con su alineación básica: horizontal, vertical y diagonal. A continuación, según esas teorías, el cerebro comienza a ensamblar estos rasgos en formas primitivas, registrando, por ejemplo, que en alguna parte del campo visual aparece un rasgo horizontal por encima de un rasgo vertical, o dos diagonales que se cruzan entre sí. [18]

Después, a partir de estas formas primitivas, se construyen formas más complejas; por ejemplo cuatro elementos con forma de "L" y orientaciones diferentes pueden formar un cuadrado, y así sucesivamente, hasta que las formas construidas ya resultan identificables como rasgos de objetos conocidos. [18]



Figura 3.2 Detección de objetos

La capacidad de los humanos de reconocer rápidamente objetos a pesar de la variación sustancial del aspecto de estos, se produce cuando los ojos están abiertos, es entonces cuando la información visual fluye desde la retina a través del nervio óptico hasta el cerebro, que ensambla esta información en bruto para dar forma a objetos y escenas, a través de una cascada de cálculos reflexivos, con gran retroalimentación, que culminan en una poderosa representación neuronal en la corteza temporal inferior. [19]

En estos últimos años se han realizado muchos estudios respecto a la visión artificial, disciplina científica que incluye métodos para adquirir, procesar, analizar y comprender las imágenes del mundo real con el fin de producir información numérica o simbólica para que puedan ser tratados por un computador. La visión artificial emula el comportamiento de nuestros ojos y cerebro a la hora de comprender el mundo que nos rodea. Dentro de la visión artificial en este proyecto se ha estudiado el campo de la detección de objetos, más concretamente la detección de caras de los posibles hablantes. Diversas publicaciones han sido estudiadas previamente a la realización de este proyecto.

Por ejemplo en [20], se presenta un enfoque de aprendizaje supervisado basado en entrenamiento, para detectar rostros humanos verticales y frontales, en escenas complejas. La técnica modela la distribución de los patrones de la cara humana, por medio de unos pocos grupos de modelos "cara" y "no cara" basados en vistas previas. En cada ubicación de imagen, se calcula un vector de características de diferencia entre el modelo de imagen local y el modelo de distribución. Un clasificador entrenado determina, basándose en las mediciones del vector de características de diferencia, si existe o no un rostro humano en la ubicación actual de la imagen. Demostrando empíricamente que la métrica de distancia que se adopta para calcular los vectores de características de las

diferencias, y los modelos de "no cara" que incluimos en nuestro modelo basado en la distribución, son críticos para el éxito del sistema.

Otra publicación en la investigación de detección de caras es [21], donde se presenta un sistema de detección facial basado en redes neurales. Una red neural conectada retinalmente examina pequeñas ventanas de una imagen, y decide si cada ventana contiene una cara. El sistema arbitra entre múltiples redes para mejorar el rendimiento en una sola red. Utilizan un algoritmo bootstrap para entrenar las redes, lo que añade falsas detecciones al conjunto de entrenamiento a medida que avanza el entrenamiento. Esto elimina la difícil tarea de seleccionar manualmente los ejemplos de entrenamiento sin cara, los cuales deben ser elegidos para abarcar todo el espacio de las imágenes que no muestran una cara.

Viola y Jones en [22] introdujeron un sistema para una rápida detección de objetos basado en una cascada potenciada de características simples. En [23] se extiende este trabajo en dos factores: Se introduce un nuevo conjunto de características Haar rotadas 45°, las cuales enriquecen de manera significativa este conjunto básico de características y que también se pueden calcular de manera muy eficiente. Además, se presenta un nuevo procedimiento de post optimización para un determinado clasificador potenciado que mejora significativamente su rendimiento.

Un buen ejemplo de detección de caras en tiempo real que ha servido de base para la implementación de este trabajo es [24], donde se describe un sistema de detección de rostros que es capaz de procesar imágenes extremadamente rápido al mismo tiempo que alcanza altas tasas de detección. Hay tres aportaciones clave en esta publicación. La primera es la introducción de una nueva representación de imagen denominada "Imagen Integral" que permite calcular las características utilizadas por el detector muy rápidamente. El segundo es un clasificador simple y eficiente que se construye utilizando el algoritmo de aprendizaje AdaBoost [25] para seleccionar un pequeño número de características visuales críticas de un conjunto muy grande de características potenciales. La tercera contribución es un método para combinar clasificadores en una "cascada" que permite que las regiones de fondo de la imagen sean descartadas rápidamente mientras se centra en el cálculo de regiones prometedoras de encontrar una cara. Se presenta un conjunto de experimentos en el campo de la detección de rostros. El sistema proporciona rendimiento de detección de rostros comparable a los mejores sistemas previos ([20], [21], [26] y [27]).

La localización de múltiples hablantes activos en ambientes naturales con sólo dos micrófonos es un problema difícil. La reverberación degrada el rendimiento de la localización de los hablantes basada exclusivamente en señales direccionales. La modalidad de audio por sí sola tiene problemas con la exactitud de la localización,

mientras que la modalidad de vídeo por sí sola tiene problemas con falsas detecciones de actividad de hablantes. En [28] un enfoque basado en la fusión audiovisual en dos etapas es presentado. En la primera etapa, la actividad del hablante se detecta en base a la fusión audiovisual que puede manejar movimientos falsos de los labios. En la segunda etapa, se propone un método de fusión gaussiana para integrar las estimaciones de ambas modalidades. Como consecuencia, la precisión de localización y la robustez en comparación con la modalidad de audio/vídeo por sí sola se incrementa significativamente.

Una posible aplicación del sistema desarrollado en este proyecto es la robótica social (robots autónomos), la cual tiene como objeto dotar a los robots con la capacidad de dirigir la atención a las personas con las que están interactuando. Diferentes enfoques siguen los mecanismos bioinspirados como la fusión de señales auditivas y visuales para localizar a una persona utilizando múltiples sensores. Sin embargo, la mayoría de estos mecanismos de fusión se han utilizado en sistemas fijos, como los que se utilizan en salas de videoconferencia, y por lo tanto, pueden incurrir en dificultades cuando éstos se ven limitados a los sensores con los que puede equiparse un robot. Además, en el ámbito de los robots interactivos autónomos, existe una carencia en términos de evaluación de los beneficios de los mecanismos atencionales audiovisuales, comparados con los enfoques de solo audio o solo visuales, en escenarios reales.. Con el objetivo de demostrar el beneficio de fusionar la información sensorial con una inferencia bayesiana para la robótica interactiva, en [29] se presenta un sistema para localizar a una persona mediante el procesamiento de datos visuales y de audio. Además, el rendimiento de este sistema se evalúa y compara considerando las limitaciones técnicas de los sistemas unimodales (es decir, basado exclusivamente en audio o vídeo).

Algunos sistemas de localización de hablantes han sido implementados directamente en robots. En [30] se presenta un método para detectar y localizar un hablante activo, es decir, un hablante que emite un sonido, a través de la fusión entre la reconstrucción visual con un par de cámaras estereoscópicas y localización de fuente de sonido con varios micrófonos. Las cámaras y los micrófonos están incrustados en la cabeza de un robot humanoide, concretamente el robot NAO. El modelo de fusión estadístico propuesto asocia caras 3D de hablantes potenciales con direcciones de sonido 2D. El artículo tiene dos aportaciones: un método que discretiza el espacio bidimensional de todas las posibles direcciones de sonido y que acumula evidencia para cada dirección estimando la diferencia de tiempo de llegada (TDOA) sobre todos los pares de micrófonos, los micrófonos se usan simultáneamente y simétricamente y un método de alineación audiovisual que mapea características visuales 3D en direcciones de sonido 2D y en TDOA entre pares de micrófonos. Esto permite representar

implícitamente ambas modalidades de detección en una trama de coordenadas audiovisuales común.

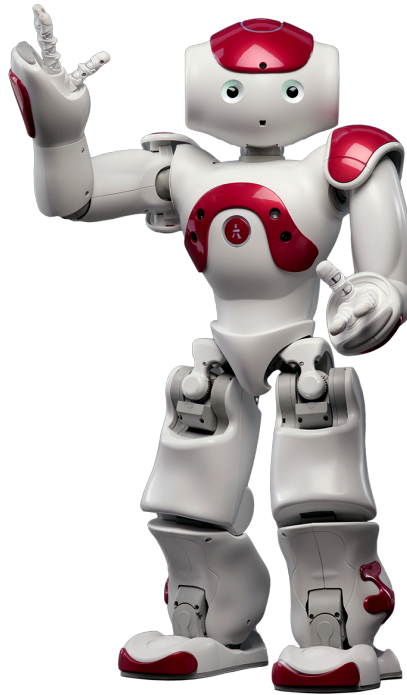


Figura 3.3 Robot NAO empleado en [30]

También se han implementado sistemas de localización de hablantes basado en información audiovisual para salas de videoconferencia como por ejemplo [31], donde se presenta un sensor novedoso y algoritmos de detección correspondientes para abordar la tarea de atender simultáneamente a múltiples hablantes, para videoconferencias. Un sensor visual panorámico se utiliza para capturar una visión de 360° de los hablantes en la sala y de esta visión, los hablantes potenciales se identifican vía un acercamiento del histograma del color. Un sistema de audio direccional basado en beamforming se utiliza entonces para confirmar a los hablantes potenciales y atenderlos. La evaluación experimental del sensor y sus algoritmos han sido presentados incluyendo el rendimiento del sistema en un entorno de videoconferencia.

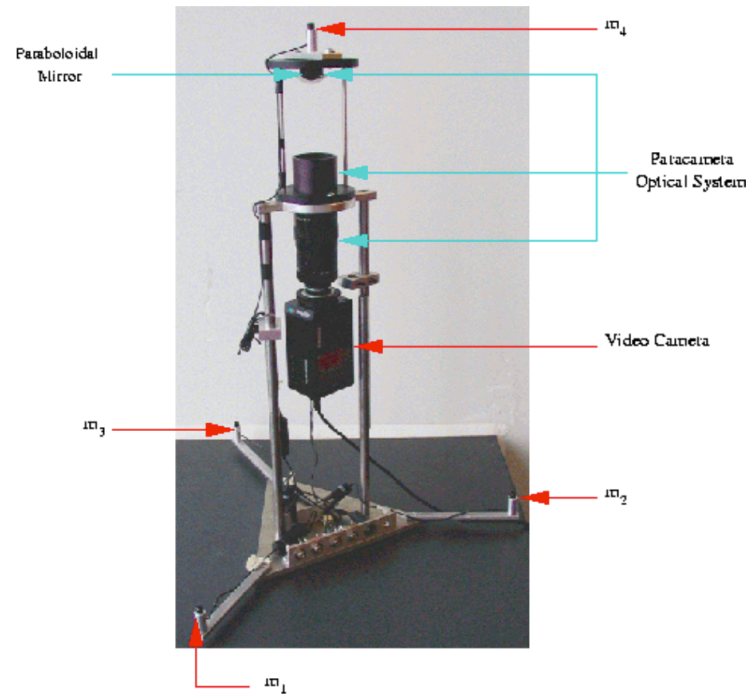


Figura 3.4 Sistema de localización empleado en [31]

4 OBJETIVOS

El objetivo de este proyecto es la construcción de un mecanismo automático que permita dirigir la atención de un sistema con una cámara incorporada hacia regiones de interés (típicamente personas hablantes). Esta tarea es de alto interés en un rango amplio de aplicaciones tales como: videoconferencia, videovigilancia o la mejora de mecanismos atencionales en sistemas inteligentes capaces de interaccionar con seres humanos.

Este proyecto propone para la localización la implementación de un sistema multimodal, dentro de la variedad existente en el procesado multimodal, este proyecto propone principalmente la fusión de información sonora y visual para la localización de múltiples personas hablando. Como paso previo a la fusión de información, se deberán realizar una extracción de características de localización, procedentes del audio mediante un conjunto de micrófonos y del vídeo mediante una cámara.

La fusión y principalmente la extracción de características, suelen involucrar técnicas costosas computacionalmente, que limitan su utilización en tiempo real. Será por tanto, uno de los objetivos de este proyecto, el obtener un sistema que pueda detectar hablantes mediante un procesado “on-line” con corto tiempo de respuesta, que funcione de forma no supervisada y permita la interacción de forma fluida con el usuario.

5 MATERIALES Y MÉTODOS

En este apartado de la memoria se van a explicar los procedimientos y algoritmos seguidos para la realización de este proyecto. El fin de este proyecto es el desarrollo de un sistema que ha sido nombrado SLAV (Sistema de Localización Audio-Visual). Para el desarrollo de éste han sido necesarias tres fases:

- Localización por medio del procesado de audio.
- Localización por medio del procesado visual.
- Fusión de ambos sistemas de localización.

Estas tres fases van a ser explicadas con más detalle en cada uno de los apartados de este capítulo de la memoria.

5.1 Procesado de audio

Como se ha comentado en el apartado de introducción, la localización de las fuentes sonoras por los humanos se basa en el análisis que el cerebro realiza sobre las señales que llegan a los oídos, utilizando sutiles diferencias de intensidad y otras señales espectrales y de tiempo para reconocer la dirección de la fuente que emitió la señal [3; 4]. Los métodos SSL automáticos utilizan arrays de micrófonos y técnicas complejas de procesamiento de señales sonoras para realizar esta tarea. Sin embargo, existen efectos no deseados como las reflexiones acústicas y el ruido o contaminación acústica que hacen este proceso difícil.

En este proyecto para la implementación y desarrollo en el lenguaje C del SLAV propuesto, en la fase de procesado auditivo para localización de fuentes sonoras, se ha optado por implementar el algoritmo SRP-PHAT. Como se ha mostrado en la introducción es un algoritmo que ha sido ampliamente usado y que se comporta bastante bien en entornos reverberantes y ruidosos.

5.1.1 *Adquisición y muestreo de las señales de audio*

El primer paso a la hora de implementar un algoritmo de localización lateral, basado en el sonido, es la captura de señales de audio. Éstas son señales analógicas que no puede ser procesadas directamente por un sistema hardware digital. El proceso de digitalización de la señal registrada por los micrófonos se realiza en este trabajo mediante el uso de una tarjeta de sonido de cuyas características se hablará en el apartado 5.1.9. No obstante, en esta sección hablaremos muy brevemente del proceso de adquisición y muestro de la señal de audio desde una perspectiva más teórica que nos sirva para entender mejor este proceso.

La Figura 5.1 muestra los pasos seguidos a la hora de adquirir y muestrear señales de audio.

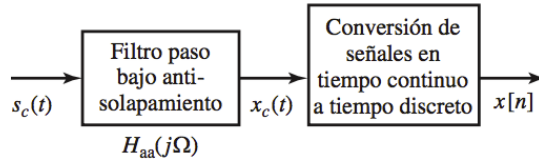


Figura 5.1 Pasos para la captura de señales de audio

La señal $s_c(t)$ es la señal continua de audio emitida por una fuente sonora. Es necesario aplicar a esta señal un filtro anti-solapamiento. Este filtro se incorpora para eliminar o minimizar el efecto del solapamiento cuando la señal en tiempo continuo se convierte en una secuencia discreta. Este solapamiento es conocido como aliasing, y ocurre cuando las muestras iniciales están demasiado espaciadas para representar componentes de alta frecuencia presentes en la señal continua. En ese caso, los valores de la DFT se verán dañados por el aliasing.

El teorema fundamental que permite representar la señal en tiempo continuo mediante una secuencia de muestras es el teorema de Nyquist-Shannon. Dicho teorema establece que para una señal de banda limitada, las muestras periódicas constituyen una representación suficiente siempre que la frecuencia de muestreo sea suficientemente alta con respecto a la máxima frecuencia de la señal en tiempo continuo.

Una vez limitado el ancho de banda de una señal mediante el filtro paso bajo anti-aliasing, tal y como aparece en la Figura 5.1, la representación en tiempo discreto de una señal en tiempo continuo, es obtenida mediante muestreo periódico. Es decir, a partir de una señal continua $x_c(t)$ se obtiene una secuencia de muestras $x[n]$ mediante la relación:

$$x_c(nT) = x[n], \quad n = 0, \pm 1, \pm 2, \dots, \quad (1)$$

Entonces $x_c(t)$ se podrá reconstruir de forma única a partir de sus muestras espaciadas uniformemente un tiempo T (periodo de muestreo), $x[n] = x_c(nT)$ para $n = 0, \pm 1, \pm 2, \dots$, si se cumple la condición de Shannon-Nyquist:

$$\frac{2\pi}{T} \geq 2\Omega_N \quad (2)$$

Siendo Ω_N la frecuencia angular máxima (en radianes por segundo) presente en la señal (ancho de banda de la señal paso bajo). La frecuencia angular $2\Omega_N$, se denomina, generalmente, frecuencia de Nyquist.

La inversa del periodo de muestreo, $f_s = 1/T$, se denomina frecuencia de muestreo (en muestras por segundo) y es un parámetro característico de la adquisición. Como consecuencia del teorema de Nyquist, la frecuencia analógica máxima presente en la señal es $f_s/2$ (en Hz).

Un sistema que realiza la operación definida en la ecuación (1) se denomina conversor ideal de tiempo continuo a discreto (C/D). En la práctica, la operación de muestreo se realiza mediante un conversor analógico–digital (A/D), en el caso de este trabajo la tarjeta de sonido consta de un conversor A/D. Este tipo de sistemas se pueden ver como aproximaciones al conversor C/D ideal.

En este proyecto la frecuencia de muestreo a la que las señales de audio han sido capturadas por la tarjeta de sonido ha sido 44100 Hz, una frecuencia admitida por la tarjeta de sonido empleada.

Para registrar dichas señales sonoras en nuestro sistema se ha empleado la librería C++ STK (The Synthesis ToolKit in C++) [32]. STK es un conjunto de procesamiento de señales de audio de código abierto y clases de síntesis algorítmicas escritas en el lenguaje de programación C++. STK fue diseñada para facilitar el desarrollo rápido de la síntesis de música y software de procesamiento de audio, con énfasis en la funcionalidad multiplataforma, control en tiempo real, facilidad de uso y código de ejemplo educativo. Synthesis ToolKit es extremadamente portátil (es en su mayor parte independiente de la plataforma C y código C++).

Las señales de audio capturadas pertenecen al dominio temporal, para la implementación del algoritmo de localización SRP-PHAT, como se va a ver más adelante cuando éste sea explicado, es necesario convertir las señales al dominio de la frecuencia.

Por tanto, un paso preliminar al cálculo propiamente dicho del algoritmo SRP-PHAT es, la conversión de las señales al dominio frecuencial.

5.1.2 *Análisis de Fourier*

Para la conversión al dominio de la frecuencia, de las señales capturadas por los micrófonos, se va a utilizar la transformada discreta de Fourier (DFT) [33], utilizando la librería de subrutinas C FFTW [34].

La transformada discreta de Fourier (DFT) consiste en muestras de la transformada de Fourier de tiempo discreto en frecuencias equiespaciadas. Por consiguiente, el cálculo de una DFT de N puntos corresponde al cálculo de N muestras

de la transformada de Fourier de tiempo discreto. La transformada discreta de Fourier (DFT) permite una representación de Fourier de señales discretas de longitud finita.[33]

Se define la transformada discreta de Fourier y su inversa (IDFT) mediante el par de ecuaciones siguientes:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-\frac{2\pi j}{N}kn} \quad k = 0, \dots, N-1 \quad (3)$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \cdot e^{\frac{2\pi j}{N}kn} \quad n = 0, \dots, N-1 \quad (4)$$

Siendo $x[n]$ la señal discreta de longitud N que se transforma y $X[k]$ la DFT de dicha señal.

La DFT se puede calcular utilizando algoritmos eficientes como por ejemplo FFT (Fast Fourier Transform) [35]. Este método para calcular la DFT es muy rápido y ha sido el que se ha usado en este proyecto por medio de la librería FFTW [34]. Debido a esta eficiencia en la computación, la DFT juega un papel central en una variedad de aplicaciones de tratamiento de señales, entre las que se encuentra el filtrado y el análisis espectral.

5.1.3 Short Time Fourier Transform (STFT)

Al computar la DFT de una señal compuesta por componentes sinusoidales, para obtener una representación en el dominio de la frecuencia y, si suponemos que las frecuencias de los cosenos no cambian con el tiempo, independientemente, de la longitud de la ventana, las propiedades de la señal (amplitudes, frecuencias y fases) serán las mismas desde el comienzo hasta el final de la ventana. Las ventanas largas producen mayor resolución en frecuencia, pero en aplicaciones prácticas de modelos de señales sinusoidales, las propiedades de la señal (es decir, amplitudes, frecuencia) a menudo cambian con el tiempo. Un ejemplo de señales no estacionarias son las señales de voz humana que se deben analizar en este proyecto. Una sola DFT no es suficiente para describir esas señales y como resultado, llegamos al concepto de transformada de Fourier dependiente del tiempo, denominada también transformada de Fourier de tiempo corto.

La STFT de una señal $x[n]$ se define como:

$$X[n, k] = \sum_{m=-\infty}^{\infty} x[n + m] \cdot w[m] e^{-\frac{2\pi j}{N} kn}, \quad (5)$$

siendo $w[n]$ una secuencia de ventana. En la representación de Fourier dependiente del tiempo (STFT), la secuencia unidimensional $x[n]$, función de una única variable discreta.

5.1.3.1 *Enventanado de la señal*

El principal propósito de la ventana en la STFT es limitar la extensión de la secuencia que se va a transformar, de forma que las características espectrales sean razonablemente estacionarias en el intervalo de duración de la ventana. Cuanto más rápidamente cambien las características de la señal, más corta deberá ser la ventana. A medida que la longitud de la ventana decrece, la resolución en frecuencia también decrece. Por supuesto, se produce el mismo efecto en $X[n, k]$. Por otra parte, a medida que decrece la longitud de la ventana, aumenta la potencialidad de resolver cambios en el tiempo. En consecuencia, aparece un compromiso en la selección de la longitud de la ventana, entre resolución en el tiempo y en la frecuencia.

La voz es claramente una señal no estacionaria. Sin embargo, se puede suponer que las características de la señal permanecen esencialmente constantes en intervalos de tiempo del orden de 30 ó 40 ms. El contenido en frecuencia de la señal de voz puede abarcar hasta 15 kHz o más, pero la voz es altamente inteligible incluso con bandas de frecuencia limitadas a unos 3 kHz. Los sistemas telefónicos comerciales, por ejemplo, limitan típicamente a unos 3 kHz. La frecuencia de muestreo estándar para sistemas digitales de comunicación telefónica es de 8000 muestras/s.

Para el enventanado de señales de voz humana, si el tamaño de la ventana L , no es demasiado grande, las propiedades de la señal no cambiarán apreciablemente desde el comienzo del segmento hasta el final. Por tanto, la DFT de un segmento de voz enventanado debería poner de manifiesto las propiedades en el dominio de la frecuencia de la señal en el instante correspondiente a la posición de la ventana. Si la ventana es demasiado grande, las propiedades de la señal pueden cambiar demasiado en la duración de la ventana. Si la ventana es demasiado corta, la resolución de las componentes de banda estrecha será sacrificada.

Para el proceso de enventanado, se ha optado por la ventana de Hanning. Esta ventana debe su nombre a Julius von Hann y se le suele denominar también ventana de coseno elevado. De entre las posibles ventanas (Blackman, Rectangular, Hamming....)

se ha elegido este tipo de ventana puesto que cumple las condiciones que permiten reconstruir la señal $x[n]$ a partir de su STFT ($X[n, k]$).

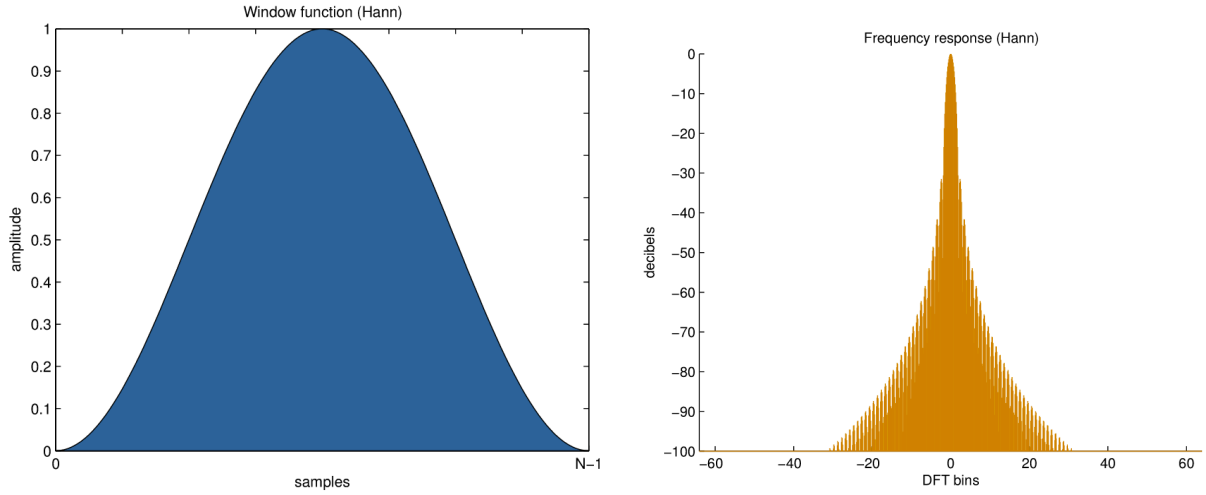


Figura 5.2 Ventana de Hanning y su respuesta en frecuencia.

Esta ventana tiene un comportamiento temporal de medio ciclo de una señal cosenoidal y normalizado en amplitud a la unidad. Esta ventana se caracteriza por el número impar de muestras N . Su ecuación es la siguiente:

$$w(n) = 0.5 - 0.5 \cdot \cos\left(2\pi \frac{n}{N-1}\right), \quad 0 \leq n \leq N-1 \quad (6)$$

La longitud de la ventana empleada ha sido de 4096 muestras. El solape entre ventanas ha sido del 50%. El número de puntos usado para el computo de la FFT ha sido 4096.

En los siguientes apartados, se explican algunos conceptos y los algoritmos implementados para la localización de fuentes sonoras desarrollada.

5.1.4 Modelo de señal

La localización de una fuente sonora se puede determinar a partir de señales recibidas en varios sensores. Uno de los métodos más eficaces es el uso de estimaciones del TDOA y/o la diferencia de frecuencia de llegada (FDOA) entre pares de señales recibidas en los sensores [36].

El modelo de señal supuesto (anecoico) para el retardo de tiempo entre las señales recibidas en los sensores, $m_1(t), m_2(t), \dots, m_M(t)$, está dado por:

$$m_1(t) = a_1(t - t_1) + w_1(t)$$

$$m_2(t) = a_2(t - t_2) + w_2(t)$$

.

.

.

$$m_M(t) = a_M(t - t_1) + w_M(t) \quad (7)$$

donde a_m son los factores de atenuación de amplitud, t_m son los retardos del tiempo de llegada de la señal entre la fuente y el sensor y $w_m(t)$ son señales de ruido aditivo. Se supone que el ruido es gaussiano, blanco, estacionario e incorrelado con la señal de interés $s(t)$. Dada la ecuación (7), $\tau = t_1 - t_2$, sería el TDOA entre los micrófonos 1 y 2.

En el dominio de la frecuencia el modelo de señal es el siguiente:

$$\begin{aligned} M_1(w) &= A_1 S(w) e^{-jw t_1} + W_1(w) \\ M_2(w) &= A_2 S(w) e^{-jw t_2} + W_2(w) \\ &\vdots \\ M_M(w) &= A_M S(w) e^{-jw t_M} + W_M(w) \end{aligned} \quad (8)$$

donde la señal, el ruido y la señal recibida tienen densidad espectral:

$$\begin{aligned} G_{s,s}(w) &= E[S(w)S^*(w)], & G_{w_1,w_1}(w) &= E[W_1(w)W_1^*(w)] & y \\ G_{m_1,m_1}(w) &= E[M_1(w)M_1^*(w)] \end{aligned}$$

respectivamente. Las amplitudes A_m dependen de la distancia entre la fuente y los micrófonos, la directividad de la fuente con el micrófono, las propiedades de las superficies reflectantes y la absorción de aire. Aquí se supone que las amplitudes son iguales a la unidad, es decir, $A_m = 1, \forall m$. Este modelo se asume por simplicidad en los casos estudiados en este proyecto.

5.1.5 Time Difference of Arrival (TDOA)

La mayoría de los esquemas prácticos de localización de fuentes acústicas se basan en la estimación TDOA (diferencia en el tiempo de llegada) porque estos sistemas son conceptualmente simples. Son razonablemente eficaces en ambientes moderadamente reverberantes y, además, su baja complejidad computacional los hace de fácil adaptación a la implementación en tiempo real con varios sensores. [37]

Un array de micrófonos está compuesto por M micrófonos, y cada micrófono se coloca en una ubicación espacial única. Por lo tanto, las ondas de sonido de trayectoria directa se propagan a lo largo de M líneas, desde la fuente hasta cada micrófono, simultáneamente. Las orientaciones de estas líneas en el sistema de coordenadas globales definen las direcciones de propagación de los frentes de onda en cada micrófono. Los vectores de propagación para un array lineal de cuatro elementos ($m = 1, \dots, 4$) se ilustran en la Figura 5.3, denominados $\vec{d}_m$.

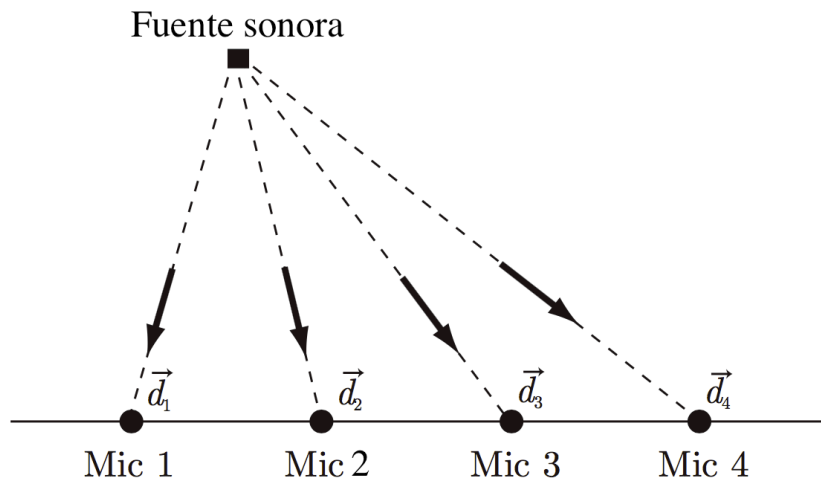


Figura 5.3 Vectores de propagación

Time Delay Estimation (TDE) se refiere al cálculo del TDOA relativo entre los diferentes micrófonos. Es una técnica fundamental en el procesamiento de señal de micrófono y el primer paso en sistemas de localización de fuentes acústicas pasivas basadas en TDOA. Con este tipo de localización, se adopta una estrategia de dos pasos como se muestra en la Figura 5.4.

El primer paso es la estimación del TDOA entre micrófonos mediante el uso de técnicas TDE [13]. Los TDOAs estimados se transforman a continuación en mediciones de diferencia de alcance entre micrófonos, dando como resultado un conjunto de ecuaciones de diferencia de distancia hiperbólica no lineal. La segunda etapa utiliza algoritmos eficientes para producir una solución inequívoca a estas ecuaciones hiperbólicas no lineales. La solución producida por estos algoritmos obtiene la posición estimada de la fuente sonora [38]. Estos datos, junto con el conocimiento de las posiciones del micrófono, se utilizan para generar curvas hiperbólicas, que luego se intersecan en sentido óptimo para llegar a una estimación de localización de fuente como se muestra en la Figura 5.5.

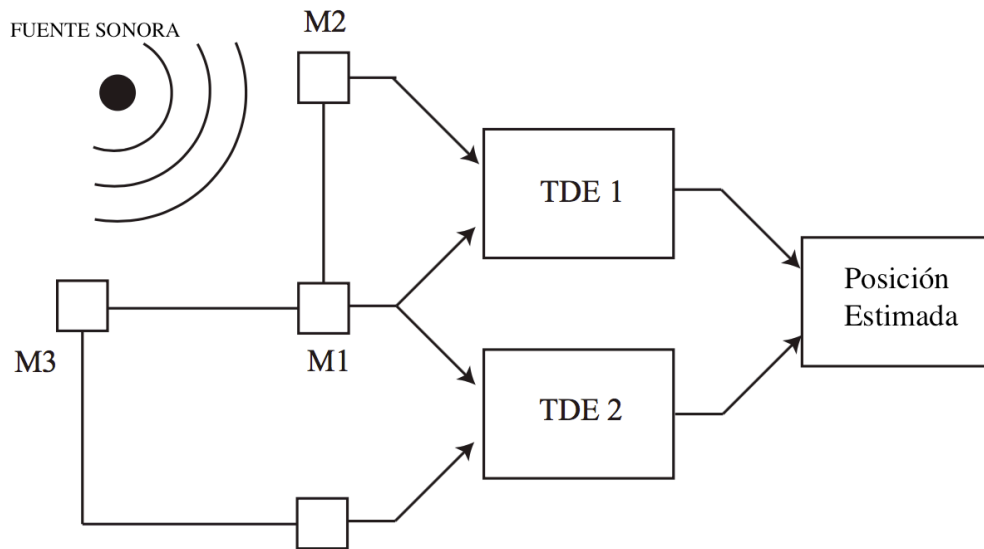


Figura 5.4 Pasos del algoritmo de localización

Algunas variaciones de este principio se han desarrollado en [39]. Estas difieren considerablemente en el método de derivación, el alcance de su aplicabilidad (2D frente a 3D, fuente en campo cercano frente a fuente en campo lejano) y sus métodos de solución.

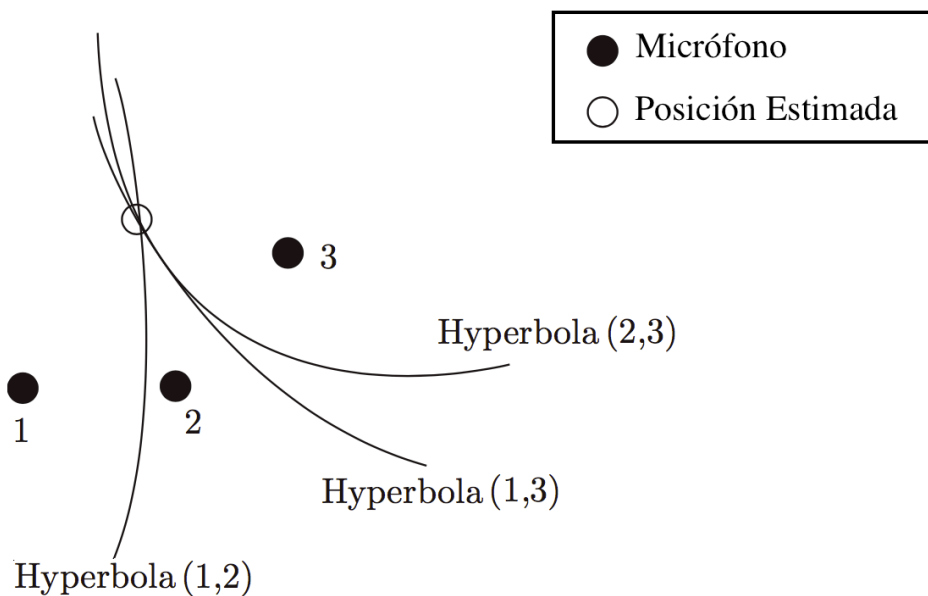


Figura 5.5 Posición estimada con tres micrófonos

5.1.6 Correlación cruzada generalizada (GCC)

Las estrategias existentes de SSL pueden dividirse en dos grandes clases: indirectas y directas [12]. Los enfoques indirectos de la localización de la fuente son usualmente métodos de dos pasos: en primer lugar, se evalúan los retardos relativos de los distintos pares de micrófonos y luego se encuentra la localización de la fuente como la intersección de un par de un conjunto de semi-hipérbolas centradas alrededor de los diferentes pares de micrófonos. Cada semi-hipérbola determina la localización posible de una fuente sonora basada en la medida de la diferencia de tiempo de llegada entre los dos micrófonos. Por otro lado, los enfoques directos exploran generalmente un conjunto de posiciones de fuente candidatas y eligen el candidato más probable como una estimación de la localización de la fuente de sonido, realizando así la localización en un solo paso.

En ambos enfoques, técnicas como la correlación cruzada generalizada, en inglés, Generalized Cross Correlation (GCC), propuesta por Knapp y Carter en [40], son ampliamente utilizadas.

Considerando la salida del micrófono l , $m_l(t)$, en un sistema de M micrófonos, la GCC para un par de micrófonos (k, l) se calcula como:

$$R_{m_k m_l}(\tau) = \int_{-\infty}^{\infty} \Phi_{kl}(w) M_k(w) M_l^*(w) e^{-j\omega\tau} dw, \quad (9)$$

donde τ es el retardo en el tiempo, $*$ indica el complejo conjugado, $M_l(w)$ es la transformada de Fourier de la señal del micrófono $m_l(t)$, y $\Phi_{kl}(w) = W_k(w) W_l^*(w)$ es una función de ponderación combinada, en el dominio de la frecuencia.

El TDE entre señales de cualquier par de micrófonos se puede realizar calculando la función de correlación cruzada de las dos señales de cada micrófono, después de aplicar la etapa de ponderación adecuada. El retardo para el que la función de correlación cruzada tiene su máximo se toma como el retardo de tiempo entre ellos.

El tipo de ponderación utilizado en el cálculo de la GCC es crucial para el rendimiento de la localización. Entre varios tipos de ponderación, la transformación de la fase (PHAT) es el pre-filtro más utilizado para la GCC porque es más robusto contra la reverberación, incluso en entornos con valores moderados de relación señal a ruido, si éste se considera espacialmente incorrelado [67]. La GCC con la transformada de la fase (GCC-PHAT) ha demostrado un buen desempeño en un ambiente reverberante suave:

$$\Phi_{kl}(w) = \frac{1}{|M_k(w)M_l^*(w)|} \quad (10)$$

Desafortunadamente, en presencia de niveles de reverberación incluso moderados, el desempeño del algoritmo empeora, debido a la presencia de picos espúreos. También las reflexiones de la señal en las paredes producen diferentes picos en la respuesta al impulso de la sala que pueden generar picos en la función GCC y pueden ser más fuertes que el pico correspondiente a la trayectoria directa. En la Figura 5.6 se muestra un ejemplo de respuesta al impulso de una sala.

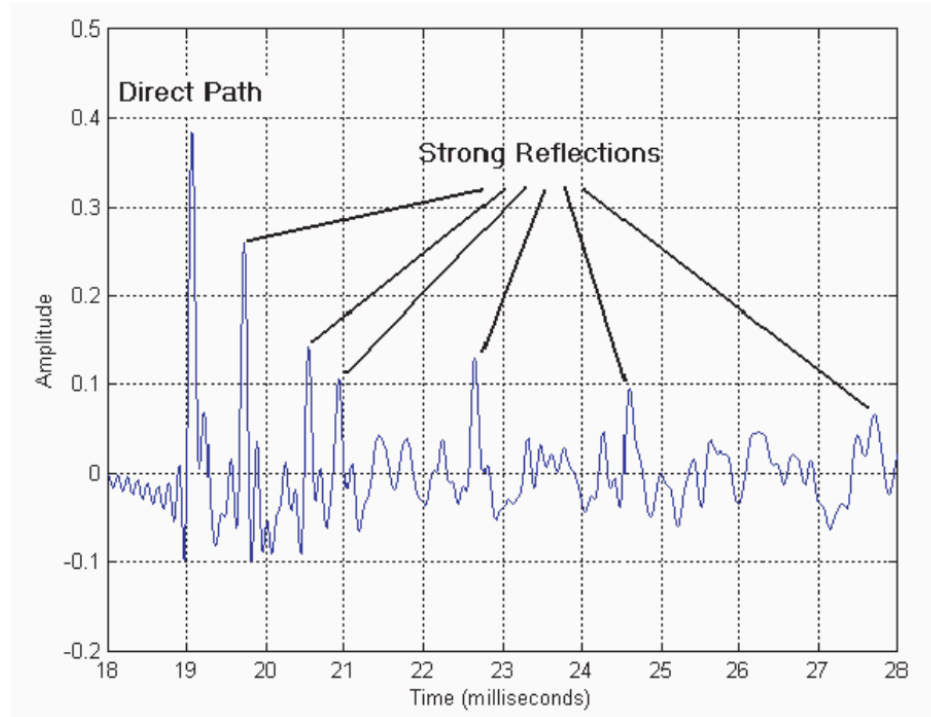


Figura 5.6 Respuesta al impulso de una fuente sonora a un micrófono

Se aprecia claramente como la onda procedente del trayecto directo “Direct Path” es la primera que es recibida por el micrófono, siendo además en ésta en la que se captura un mayor nivel de señal. Posteriormente se reciben las “Strong Reflections” que son reflexiones de la onda procedentes de rebotes en las paredes y obstáculos que puedan existir en la sala. Las reflexiones además, se reciben con una menor amplitud, ya que al rebotar en un obstáculo la señal se ve atenuada.

Para la implementación del algoritmo SRP-PHAT es necesario calcular la correlación cruzada generalizada como se verá más adelante.

5.1.7 Steered Response Power (SRP)

Otro algoritmo de SSL importante es el basado en un conformador de haz dirigido (steered beamformer). Cuando no se conoce la ubicación de la fuente sonora, se puede

utilizar un conformador de haz para escanear sobre una región espacial predefinida ajustando sus parámetros de dirección. La salida de un conformador de haz se conoce como respuesta dirigida. Cuando el punto o la dirección de la exploración coincide con la ubicación de la fuente sonora, el SRP es máximo. Sin embargo, las técnicas de formador de haces direccionales convencionales, que aplican filtros a las señales de agrupamiento han sido derivadas para mejorar su rendimiento en la localización. Cuando se utiliza el filtro de la transformada de la fase en el método de conformador de haz direccional, el algoritmo resultante es SRP-PHAT, este algoritmo es mejor para combatir los efectos adversos del ruido de fondo y la reverberación en comparación con el método convencional de conformador de haz direccionado y que el método GCC-PHAT [40].

Hoy en día, el algoritmo SRP-PHAT se ha convertido en un método de localización muy utilizado por su robusto rendimiento en entornos reales. Sin embargo, los requisitos computacionales del método son elevados y esto dificulta su implementación en tiempo real. Desde que el método SRP-PHAT fue propuesto, ha habido varios intentos de reducir los requisitos computacionales de su proceso de búsqueda [41; 42].

Las técnicas de procesamiento de señales de arrays de micrófonos se basan en la capacidad de enfocarse en señales que se originan desde una ubicación o dirección particular en el espacio. La mayoría de estas técnicas emplean algún tipo de beamforming, que generalmente incluye cualquier algoritmo que explota la capacidad de captura de sonido de un array de micrófonos [43]. El beamforming, se puede definir mediante un proceso de filtro y suma, que aplica algunos filtros temporales a las señales de cada micrófono antes de sumarlas para producir una única señal enfocada. Estos filtros se adaptan a menudo durante el proceso de formación de haz, para mejorar la señal del trayecto directo a la fuente sonora deseada, mientras que atenúan otras. Los filtros más simples ejecutan cambios de tiempo que han sido adaptados a los retardos de propagación de las señales de origen. Este método se denomina formación de haz de retardo y suma. Retrasa las señales de cada micrófono de modo que todas las versiones de la señal fuente se alinean en el tiempo antes de que se sumen. Los filtros de técnicas de filtro y suma más sofisticadas suelen aplicar esta alineación de tiempo, así como otros procesos de mejora de señal.

Las técnicas de beamforming se han aplicado tanto a la captación de la señal de la fuente sonora como a la localización de la fuente. Si se conoce la ubicación de la fuente (y tal vez también se conoce algo acerca de la naturaleza de la señal fuente), entonces un conformador de haz puede enfocarse en la fuente y su salida se convierte en una versión mejorada (en cierto sentido) de las entradas desde los micrófonos. Si no se conoce la ubicación de la fuente sonora, se puede utilizar un conformador de haz para

explorar, o dirigir, sobre una región espacial predefinida ajustando sus retardos de dirección (y posiblemente sus filtros). Como se ha comentado anteriormente, la salida de un conformador de haz, cuando se usa de esta manera, se conoce como respuesta dirigida (steered response). El SRP puede alcanzar un máximo bajo una variedad de circunstancias, pero en condiciones favorables, se maximiza cuando los retardos de dirección coinciden con los retardos de propagación. Mediante la predicción de las propiedades de las ondas de propagación, estos retardos de dirección pueden ser asignados a una ubicación concreta, que debe coincidir con la ubicación de la fuente sonora.

Para la aplicación de este algoritmo en la captura de voz, aplicación demandada en este proyecto, los filtros aplicados por la técnica de filtro y suma no sólo deben suprimir el ruido de fondo y las contribuciones de fuentes no deseadas, también deben hacerlo de manera que no distorsione significativamente la señal deseada, la señal proveniente del trayecto directo. El más común de estos filtros es la transformada de la fase (PHAT), que aplica una función de ponderación normalizada en magnitud al espectro cruzado de dos señales de micrófono.

A continuación se va a describir el algoritmo SRP-PHAT, el cual está estrechamente relacionado con el GCC-PHAT y es el algoritmo que se ha implementado en este proyecto para detección de hablantes mediante el procesado de audio. Posteriormente se explicará la implementación de dicho algoritmo.

5.1.8 *Steered Response Power PHAT (SRP-PHAT)*

Considerando la salida del micrófono $l, m_l(t)$, en un array con M micrófonos. Entonces, el punto espacial SRP ($\mathbf{x} = [x, y, z]$) para una trama temporal de longitud T viene dado por:

$$P_n(\mathbf{x}) = \int_{nT}^{(n+1)T} \left| \sum_{l=1}^M w_l m_l(t - \tau(\mathbf{x}, l)) \right|^2 dt, \quad (11)$$

donde w_l es un peso y $\tau(\mathbf{x}, l)$ es el tiempo del camino directo desde el punto $\mathbf{x}$ hasta el micrófono l .

Teniendo en cuenta las simetrías involucradas en el cálculo de la ecuación (11) y eliminando algunos términos de energía fijos [44], la parte de $P_n(\mathbf{x})$ que cambia con $\mathbf{x}$ se expresa como:

$$P'_n(\mathbf{x}) = \sum_{k=1}^M \sum_{l=k+1}^M R_{m_k m_l}(\tau_{kl}(\mathbf{x})), \quad (12)$$

donde $R_{m_k m_l}$ es la correlación cruzada generalizada para un determinado par de micrófonos, $\tau_{kl}(\mathbf{x})$ es la Inter-Microphone Time-Delay Function (IMTDF), función de retardo de tiempo entre micrófonos. Esta función es muy importante, ya que representa el retardo del trayecto directo para el par de micrófonos (k, l) resultante de una fuente puntual localizada en $\mathbf{x}$. El IMTDF se expresa matemáticamente como:

$$\tau_{kl}(\mathbf{x}) = \frac{\|\mathbf{x} - x_k\| - \|\mathbf{x} - x_l\|}{c}, \quad (13)$$

donde c es la velocidad del sonido que viene dada por la siguiente ecuación:

$$c = cc \cdot \sqrt{1 + \frac{\text{temperatura_sala}}{273}} \quad (14)$$

donde cc toma como valor 340,6032. La temperatura de la sala se expresa en grados Celsius, se ha supuesto una temperatura de sala de 20 °C.

En (13) x_k, x_l son los puntos en el espacio donde se localizan los micrófonos k y l .

El algoritmo SRP-PHAT evalúa la función $P'_n(\mathbf{x})$ sobre una rejilla fina G con el fin de encontrar una posición de fuente sonora puntual x_s , que proporciona el valor máximo:

$$x_s = \arg \max P'_n(\mathbf{x}), \quad (15)$$

donde $\mathbf{x} \in G$.

Para el cálculo del algoritmo SRP-PHAT se han seguido los siguientes pasos:

1. Definir un grid o mallado espacial G con una resolución espacial r (distancia entre los puntos del mallado). Los retardos teóricos desde cada punto del grid a cada par de micrófonos son precalculados. Para el cálculo de los retardos es necesario definir previamente los puntos del espacio en los que están localizados cada uno de los micrófonos que componen el array.

A la hora de definir el grid o mallado, si éste define un área menor o la distancia entre cada uno de los puntos que lo definen (r) es mayor, el tiempo de computación para estimar la posición del hablante se reducirá. Por esto, y como una de las exigencias del proyecto es el funcionamiento en tiempo real, se ha optado por definir un mallado no excesivamente amplio. El mallado implementado define el área en el que las personas habitualmente se hablan unas a otras.

2. Para cada trama de análisis, se calcula la GCC (Correlación Cruzada Generalizada) de cada par de micrófonos en un conjunto predeterminado de la variable retardo (τ).

En este cálculo es clave el número de retardos (taus), que estarán equiespaciados. Estos taus equiespaciados se definen desde:

$$-\text{diámetro array} / c \leq \tau \leq \text{diámetro array} / c$$

Estos retardos se definen desde y hasta los dos extremos desde los que pueden recibirse señales sonoras emitidas por un hablante, con un array de micrófonos de un determinado *diámetro*.

La cantidad de retardos equiespaciados que se empleen para calcular la GCC varía considerablemente el tiempo de computación para obtener la posición de la fuente sonora en este algoritmo. Los diferentes valores empleados para este parámetro y cómo éste afecta en el desempeño del algoritmo, serán expuestos en el apartado de resultados y conclusiones.

3. Para cada posición del grid $x \in G$, la contribución de las diferentes correlaciones cruzadas son acumuladas (usando los retardos precalculados en el primer punto) para cada par de micrófonos.

Como se ha especificado en los pasos previos, existen dos tipos de retardos calculados hasta el momento. Los retardos del mallado que son los retardos existentes entre cada par de micrófonos y los diferentes puntos del mallado o grid (calculados en el primer paso). Por otro lado, existen los retardos equiespaciados para los que se ha calculado la GCC (definidos en el segundo paso). Ambos retardos no tienen forzosamente que coincidir. Por esto, ha sido necesario realizar una interpolación lineal [45] la cual viene dada por:

(1)



4. Finalmente, la posición x con el máximo valor es elegida, una vez sumadas las contribuciones de todos los pares de micrófonos en la Ecuación (12).

30

Llegados a este punto, la posición del hablante obtenida es la posición final del sistema de localización de hablantes mediante procesamiento de audio, dicha posición se expresa en coordenadas cartesianas.

El diagrama de bloques de la Figura 5.8 muestra los pasos seguidos para la localización de hablantes mediante el procesamiento de audio:

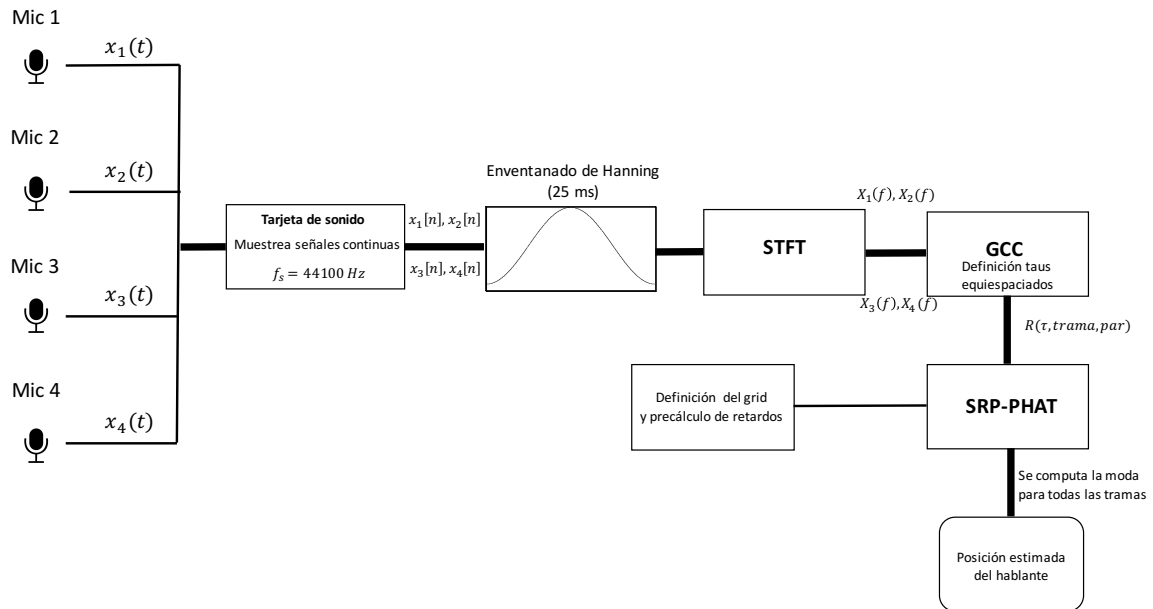


Figura 5.8 Diagrama de bloques del procesamiento de audio

Este proyecto ha sido desarrollado enteramente en el lenguaje de programación C. Por tanto, el algoritmo SRP-PHAT ha sido implementado en dicho lenguaje, siguiendo los pasos previamente descritos.

5.1.9 Hardware de procesamiento de audio

En la fase de procesamiento de audio de este proyecto, se ha empleado un array de micrófonos, para la captura de audio.

El número de micrófonos y la geometría (como se disponen los micrófonos) del array son dos aspectos clave y deben tenerse en cuenta, para poder obtener una buena precisión en la localización de las fuentes sonoras [46]. Así, ha sido necesario realizar un estudio previo de las diferentes geometrías y tipos de arrays utilizados en los sistemas de SSL hasta el momento, para ver cual es el que más se amoldaba a las necesidades de este proyecto. Las pruebas realizadas para este estudio se han realizado simulando un entorno acústico con el software para Matlab "Roomsim" [47;48]. Los resultados y la descripción de los arrays probados se muestran en el apartado de resultados y conclusiones.

Después de probar con diferentes disposiciones de arrays de 6 y 4 micrófonos y atendiendo a los resultados obtenidos se ha optado por un array formado por 4 micrófonos con una geometría cuadrada. Como se muestra en la Figura 5.9 la separación entre micrófonos es de 30 cm. Esta geometría, debido a sus dimensiones, puede ser implementada fácilmente en un robot autónomo.

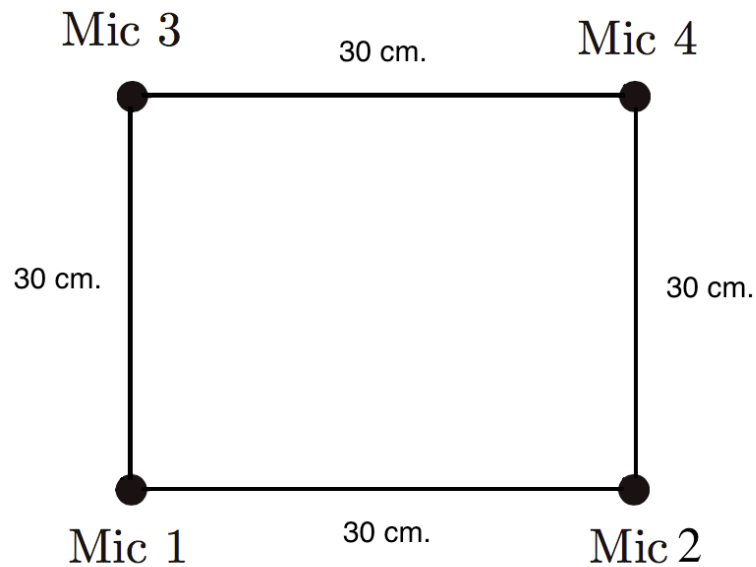


Figura 5.9 Esquema del array de micrófonos implementado

Los micrófonos que se han utilizado para este array son los micrófonos omnidireccionales modelo AKG C 417 PP.



Figura 5.10 AKG C 417 PP

Las especificaciones técnicas de este modelo de micrófono son:

- Respuesta en frecuencia: 20 – 20.000 Hz
- Diagrama polar: omnidireccional
- Sensibilidad: 10 mV/Pa
- Impedancia: 200 Ohms
- Máximo SPL para 1% / 3% THD: 118 dB/126 dB

- Dimensiones: diám. 7.5 x 15 mm.
- Peso neto: 8 g.

Estos micrófonos con conector XLR, han sido conectados a una tarjeta de sonido en sus conexiones traseras mediante los conectores XLR hembra de los que dispone ésta.

La tarjeta de sonido dispone de 8 canales de entrada. El modelo elegido para la realización del proyecto ha sido: MOTU 8 Pre Firewire.



Figura 5.11 MOTU 8 PRE Firewire

Las especificaciones técnicas de esta tarjeta de sonido son:

- Convertidor AD/DA de 24Bit/96kHz
- Frecuencias de muestreo permitidas: 44100, 48000, 88200, 96000 Hz.
- Número de entradas analógicas: 8
- 8 preamplificadores de micrófono con alimentación phantom (combo XLR/TRS)
- Entrada/Salida ADAT (hasta 96kHz)
- Entrada/Salida MIDI
- 2 salidas analógicas (TRS de 1/4")
- Salida de auriculares
- Funciona como convertidor ADAT de 8 canales autónomo
- Requiere Mac OSX 10.6 o superior o Windows Vista o superior
- Tamaño: 19"/1U

Debido a que el interfaz utilizado por dicha tarjeta de sonido es firewire, ha sido necesario utilizar un adaptador firewire a PCI Express, ya que el ordenador asignado para la realización del proyecto, no disponía de dicho interfaz.

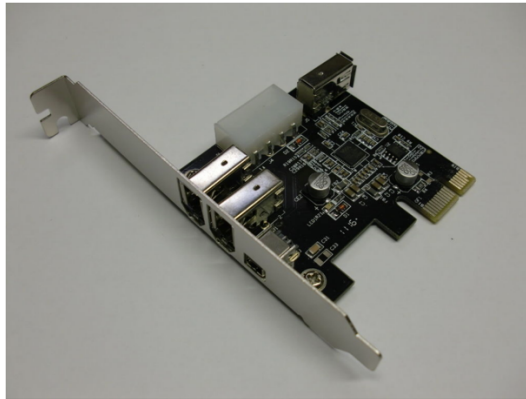


Figura 5.12 Adaptador Firewire PCI Express

Los drivers necesarios para que dicha tarjeta fuera reconocida por el sistema operativo Windows 7, han sido instalados y configurados.

Posteriormente se han utilizado dos herramientas software que Motu facilita para la monitorización de los canales (CueMix FX) y la asignación de los parámetros adecuados para la captura de audio con dicha tarjeta (MOTU Audio Console), comprobando así que el equipo reconoce la tarjeta de sonido conectada.

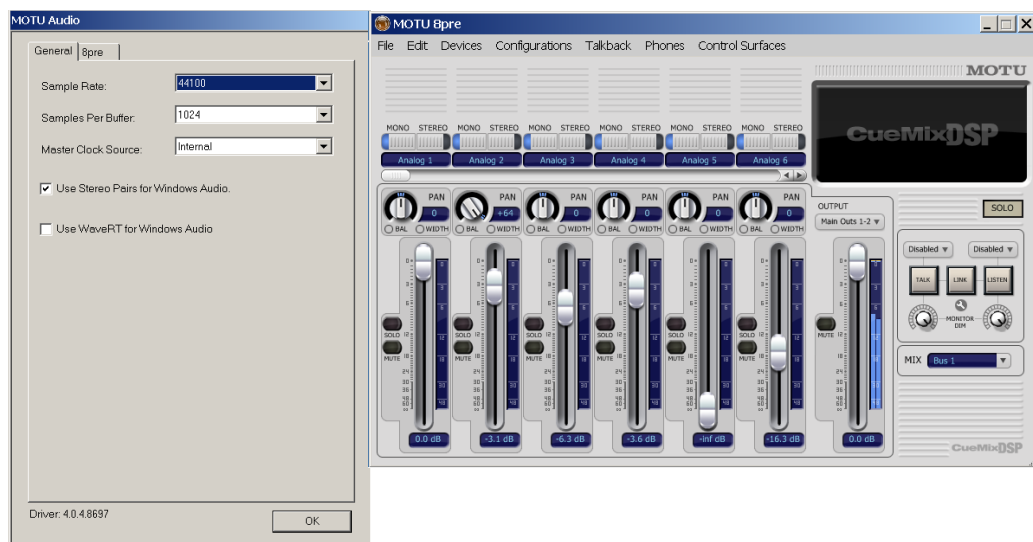


Figura 5.13 Software MOTU

Realizados todos estos pasos, nuestro hardware está listo para capturar audio a través del programa C desarrollado en este proyecto.

5.2 Procesado de vídeo

La detección de objetos es la parte de la visión artificial que estudia cómo detectar la presencia de objetos en una imagen sobre la base de su apariencia visual, bien sea atendiendo al tipo de objeto (una persona, un coche) o a la instancia del objeto (mi coche, el coche del vecino). Generalmente se pueden distinguir dos partes en el proceso de detección: la extracción de características del contenido de una imagen y la búsqueda de objetos basada en dichas características.

La extracción de características consiste en la obtención de modelos matemáticos compactos que "resuman" el contenido de la imagen con el fin de simplificar el proceso de aprendizaje de los objetos a reconocer. Dichas características son comúnmente llamadas descriptores.

Existen diversos tipos de descriptores, que tendrán mejor o peor rendimiento en función al tipo de objeto a reconocer y a las condiciones del proceso de reconocimiento (la luz controlada o no, distancia al objeto a reconocer conocida o no). Se pueden usar desde básicos histogramas de color o intensidad de luz, descriptores LBP (Local Binary Pattern, usado sobre todo para texturas) o más avanzados como el HOG (Histogram of Oriented Gradients), SIFT (Scale Invariant Feature Transform) o SURF (Speeded Up Robust Features).

Para el proceso de clasificación se pueden usar diferentes técnicas de aprendizaje máquina. Existen diferentes métodos, como la regresión logística, o más avanzados basados en técnicas de aprendizaje automático como el SVM o AdaBoost (Adaptative Boost), utilizado por OpenCV.

Los mayores retos tanto de la extracción de características como la clasificación es encontrar descriptores y clasificadores que sean invariantes a los cambios que pueda tener un objeto, como su posición o iluminación.

Para este trabajo, el objeto a detectar por medio de la visión artificial es una cara o rostro de una persona humana.

5.2.1 Detección de rostros

Hoy en día algunas aplicaciones de reconocimiento facial no requieren una etapa previa de detección de rostros. En algunos casos, imágenes faciales normalizadas son almacenadas en bases de datos. En estas aplicaciones hay un formato de entrada de imagen estándar, por lo que no es necesario un paso previo de detección. Un ejemplo de esto podría ser una base de datos criminal. Allí, la policía almacena las caras de las personas con un informe criminal. Si hay un nuevo sujeto y la policía tiene su fotografía de pasaporte, la detección de rostros no es necesaria. Sin embargo, una imagen de entrada convencional puede no ser adecuada, y contener muchos elementos además

de caras. Los sistemas de videovigilancia tratan de incluir la detección de rostros, el seguimiento y el reconocimiento. En estos casos, la detección previa de los posibles rostros en la imagen es obligatoria. Por lo tanto, es razonable asumir la detección de rostros como parte del problema de reconocimiento facial más amplio. También es inevitable la detección facial si queremos desarrollar un sistema automatizado de rastreo de rostros, que es el caso concreto de este proyecto. [49]

La detección de rostros debe hacer frente a varios desafíos [50; 51]. Estos están por lo general presentes en imágenes capturadas en entornos incontrolados, tales como sistemas de videovigilancia. Estos desafíos pueden atribuirse a algunos factores:

- Variación en la pose: El escenario ideal para la detección de rostros sería uno en el que sólo las imágenes frontales estuvieran involucradas. Pero, como se ha dicho, esto es muy improbable en condiciones generales no controladas. Por otra parte, el rendimiento de los algoritmos de detección de cara decae gravemente cuando hay grandes variaciones de pose. Este es un tema en el que se está investigando. La variación en la pose del rostro puede suceder debido a los movimientos del sujeto o ángulo de la cámara.
- Característica de oclusión: La presencia de elementos como barba, gafas o sombreros introduce una alta variabilidad. Las caras también pueden estar parcialmente cubiertas por otras caras u objetos.
- Expresión facial: Las características faciales también varían mucho debido a los diferentes gestos faciales.
- Condiciones de la imagen. Diferentes cámaras y condiciones ambientales pueden afectar a la calidad de una imagen, afectando a la apariencia de la cara a detectar.

Hay algunos problemas estrechamente relacionados con la detección de rostros, además de la extracción de características y la clasificación facial. Por ejemplo, la localización de la cara es un acercamiento simplificado de detección de rostros. Su objetivo es determinar la ubicación de una cara en una imagen donde sólo hay una cara. Podemos diferenciar entre la detección de la cara y la localización de la cara, ya que ésta es una versión simplificada de la primera. Métodos como la localización de los límites de una cabeza en una imagen [52] se utilizaron por primera vez en este escenario y luego fueron exportados a problemas más complicados. La detección de características faciales se refiere a la detección y localización de algunas características relevantes, como la nariz, las cejas, los labios, las orejas, etc. Algunos algoritmos de extracción de

características se basan en la detección de rasgos faciales. El seguimiento de rostros es otro problema que a veces es una consecuencia de la detección de rostros. La meta de muchos sistemas no es sólo detectar una cara, sino poder localizar esta cara en tiempo real. Las exigencias de este proyecto es un buen ejemplo.

5.2.2 Estructura del problema de detección de rostros

La detección de rostros es un concepto que incluye muchos sub-problemas. Algunos sistemas detectan y localizan las caras al mismo tiempo, otros primero realizan una rutina de detección y luego, si es positivo, intentan localizar la cara. A continuación, se pueden aplicar además algoritmos de seguimiento para un *tracking* facial (ver Figura 5.14).

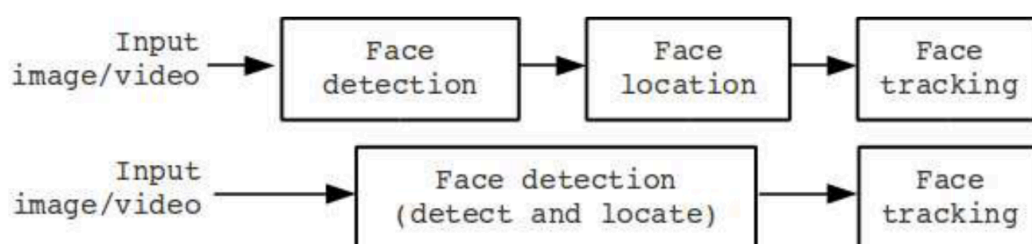


Figura 5.14 Sistemas de detección, localización y tracking de rostros

Los algoritmos de detección de rostros usualmente comparten pasos comunes. En primer lugar, se realiza alguna reducción de la dimensión de datos, con el fin de conseguir un tiempo de respuesta admisible. Se realiza algún pre-procesamiento para adaptar la imagen de entrada a los requisitos previos del algoritmo. Entonces, algunos algoritmos analizan la imagen tal cual es, y otros tratan de extraer ciertas regiones faciales relevantes. La siguiente fase generalmente implica la extracción de rasgos faciales o medidas. Estos serán entonces ponderados, evaluados o comparados para decidir si hay una cara y dónde está. Finalmente, algunos algoritmos tienen una rutina de aprendizaje e incluyen nuevos datos a sus modelos.

La detección de rostros es, por tanto, un problema de dos clases donde tenemos que decidir si hay una cara o no en una imagen. Este enfoque puede ser visto como un problema simplificado de reconocimiento facial. El reconocimiento facial tiene que clasificar una cara determinada, de entre una serie de candidatos. En consecuencia, muchos métodos de detección de rostros son muy similares a los algoritmos de reconocimiento de rostros. O de otra manera, las técnicas utilizadas en la detección de rostros se utilizan a menudo en el reconocimiento facial.

5.2.3 Enfoques para la detección de rostros

No es fácil clasificar los métodos de detección de rostros. No hay un criterio de agrupamiento aceptado a nivel mundial. Por lo general se mezclan y se superponen. En esta sección se presentarán dos criterios de clasificación. Uno de ellos diferencia entre distintos escenarios, dependiendo de las características de éstos, pueden ser necesarios diferentes enfoques. El otro criterio divide los algoritmos de detección en cuatro categorías.

5.2.3.1 Detección dependiendo del escenario

- Entorno controlado: Es el caso más sencillo. Fotografías tomadas bajo luz controlada, fondo controlado, etc. Las técnicas simples de detección de bordes pueden usarse para detectar caras [53].
- Imágenes en color: Los colores típicos de la piel se pueden utilizar para encontrar caras. Pero este tipo de detección se ve afectado por los cambios en las condiciones de luz. Además, el color de la piel humana cambia mucho, desde casi blanco hasta casi negro. Sin embargo, varios estudios muestran que la principal diferencia radica en su intensidad, por lo que la crominancia es una buena característica [50]. No es fácil establecer una sólida representación del color de la piel humana. Sin embargo, hay intentos de construir sólidos algoritmos de detección de la cara basada en el color de la piel [54].
- Imágenes en movimiento. El vídeo en tiempo real da la oportunidad de usar la detección de movimiento para localizar rostros. Hoy en día, la mayoría de los sistemas comerciales deben localizar las caras en vídeos. Existe un desafío continuo para lograr los mejores resultados de detección con el mejor rendimiento posible [55]. Otro enfoque basado en el movimiento es la detección de parpadeo de ojos, que tiene muchos usos aparte de la detección de rostros [56; 57].

5.2.3.2 Métodos de detección divididos en categorías

Yan, Kriegman y Ahuja presentaron una clasificación [50]. En ella los métodos se dividen en cuatro categorías. Estas categorías pueden superponerse, por lo que un algoritmo podría pertenecer a dos o más categorías. Esta clasificación consiste en:

- Métodos basados en el conocimiento: Basados en métodos que codifican nuestro conocimiento de las caras humanas.
Tratan de captar nuestro conocimiento de los rostros y de traducirlos en un conjunto de reglas. Es fácil adivinar algunas reglas simples. Por ejemplo, una cara normalmente tiene dos ojos simétricos, y el área de los ojos es

más oscura que las mejillas. Características faciales podrían ser la distancia entre los ojos o la diferencia de intensidad de color entre el área de los ojos y la zona inferior. El gran problema con estos métodos es la dificultad en la construcción de un conjunto apropiado de reglas. Podría haber muchos falsos positivos si las reglas fueran demasiado generales. Por otro lado, podría haber muchos falsos negativos si las reglas fueran demasiado detalladas. Una solución es construir métodos jerárquicos basados en el conocimiento para superar estos problemas. Sin embargo, este enfoque por sí solo es muy limitado. Es incapaz de encontrar muchas caras en una imagen compleja.

- Métodos de características: Algoritmos que intentan encontrar rasgos invariantes de una cara a pesar de su ángulo o posición. La idea es superar los límites de nuestro conocimiento instintivo de los rostros. Uno de los primeros algoritmo fue desarrollado por Han, Liao, Yu y Chen en [58].
- Métodos de coincidencia de patrones: Estos algoritmos comparan imágenes de entrada con patrones almacenados de rostros o características faciales.

Estos métodos intentan definir una cara como una función. Tratan de encontrar un patrón estándar de todas las caras. Las diferentes características se pueden definir independientemente. Por ejemplo, una cara se puede dividir en ojos, contorno del rostro, nariz y boca. También un modelo de la cara se puede construir por los bordes. Pero estos métodos se limitan a caras que son frontales y no ocultas. Una cara también puede ser representada como una silueta. Otros patrones usan la relación entre las regiones de la cara en términos de brillo y oscuridad. Estos patrones estándar se comparan con las imágenes de entrada para detectar rostros. Este enfoque es simple de implementar, pero es inadecuado para la detección de rostros. No puede lograr buenos resultados con variaciones en la pose, la escala y la forma. Sin embargo, se han propuesto patrones deformables para tratar estos problemas.

- Métodos basados en la apariencia: Un método de adaptación de patrones cuya base de datos patrón se aprende de un conjunto de imágenes de entrenamiento.

En general, los métodos basados en la apariencia se basan en técnicas de análisis estadístico y de aprendizaje automático para encontrar las características relevantes de las imágenes faciales. Algunos métodos

basados en la apariencia trabajan con una red probabilística. Un vector de imagen o característica es una variable aleatoria con alguna probabilidad de pertenecer a una cara o no. Otro enfoque consiste en definir una función de discriminación entre las clases de cara y no de cara. Estos métodos también se usan en la extracción de rasgos para el reconocimiento facial y se discutirán más adelante. Sin embargo, estos son los métodos o herramientas más relevantes, entre ellos se encuentran: los basados en Eigenface, los basados en distribución, los referidos a redes neuronales, los SVM (Support Vector Machines), SNoWs (Sparse Network of Windows), clasificadores Naive Bayes, Hidden Markow model, los de enfoque de información teórica y los de aprendizaje inductivo.

En este trabajo el algoritmo de detección empleado, se emplea en un escenario de imágenes en movimiento, capturadas con una vídeo cámara descrita más adelante. Además, el algoritmo se podría clasificar dentro de los métodos basados en el conocimiento de las características faciales, mediante la coincidencia de patrones. Este método ha sido empleado usando la librería de visión artificial OpenCV, explicada más adelante.

5.2.4 Seguimiento de rostros

Muchos sistemas de reconocimiento facial tienen una secuencia de vídeo como entrada. En esos sistemas se puede exigir que sean capaces no sólo de detectar sino de seguir la posición de las caras. El seguimiento de rostro es esencialmente un problema de estimación de movimiento, éste puede realizarse usando muchos métodos diferentes, por ejemplo, seguimiento de la cabeza, seguimiento de rasgos faciales, seguimiento basado en imágenes o basado en modelos. Estas son diferentes maneras de clasificar estos algoritmos [51]:

- Seguimiento de la cabeza / seguimiento de una característica individual. La cabeza puede ser seguida como una entidad completa, o ciertas características seguidas individualmente.
- 2D / 3D. Los sistemas bidimensionales siguen un rostro y emiten una imagen del espacio donde se encuentra el rostro. Los sistemas tridimensionales, por otro lado, realizan un modelado 3D del rostro. Este enfoque permite estimar las variaciones de pose u orientación.

El proceso básico de seguimiento de rostros busca localizar una imagen de una cara dada en una imagen. Luego, tiene que calcular las diferencias entre los fotogramas para actualizar la ubicación de la cara. Hay muchos problemas que deben ser

enfrentados como por ejemplo: oclusiones parciales, cambios de iluminación, velocidad computacional y deformaciones faciales.

Un ejemplo de un algoritmo de seguimiento de rostros puede ser el propuesto por Baek et al. en [59]. El vector de estado de una cara incluye la posición central, el tamaño del rectángulo que contiene la cara, el color medio del área de la cara y sus primeras derivadas. Las nuevas caras candidatas son evaluadas por un estimador de Kalman. En el modo de seguimiento, si la cara no es nueva, la cara del marco anterior se utiliza como plantilla. La posición de la cara es evaluada por el estimador de Kalman y la región de la cara es buscada alrededor por un algoritmo SSD usando el patrón mencionado. Cuando el algoritmo SSD encuentra la región, la información de color se incrusta en el estimador de Kalman para confinar exactamente la región de la cara. A continuación, se actualiza el vector de estado de esa cara.

El seguimiento de rostros implementado en este trabajo, realiza un seguimiento de las características de un rostros en una imagen 2D (coordenadas 'x','y' del frame de visión).

5.2.5 Procesado visual con OpenCV

Para la detección y localización de hablantes en la fase de procesado visual, se ha desarrollado un software utilizando la librería OpenCV para C.

OpenCV [60] es una biblioteca de visión artificial de código abierto, desarrollada por Intel. Está escrita en C y C ++ y se ejecuta bajo Linux, Windows y Mac OS X. Esta librería da soporte a varias plataformas de desarrollo activo como: Python, Ruby, Matlab y otros idiomas.

OpenCV fue diseñado para aprovechar la eficiencia computacional y con un fuerte enfoque en aplicaciones en tiempo real. OpenCV está escrito en C optimizado y puede aprovechar los procesadores multicore.

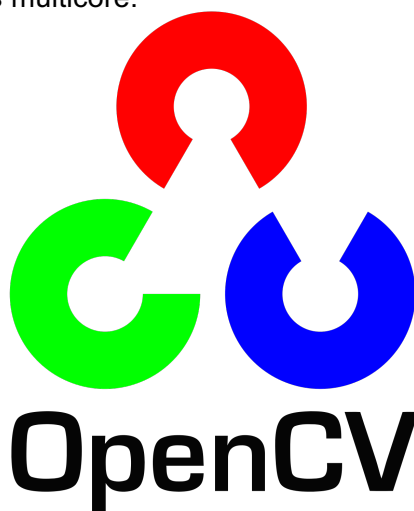


Figura 5.15 OpenCV

Uno de los objetivos de OpenCV es proporcionar una infraestructura de visión artificial fácil de usar que ayude a las personas a construir aplicaciones de visión bastante sofisticadas rápidamente. La biblioteca OpenCV contiene más de 500 funciones que abarcan muchas áreas de visión, incluyendo inspección de productos de fábrica, imágenes médicas, seguridad, interfaz de usuario, calibración de cámara, visión estéreo y robótica. Debido a que la visión computada y el aprendizaje automático de la máquina van de la mano, OpenCV también contiene una Machine Learning Library (MLL) biblioteca de aprendizaje automático de uso general. Esta sub-biblioteca se centra en el reconocimiento de patrones estadísticos y agrupación.

5.2.5.1 Instalación de OpenCV

El primer paso para poder trabajar con la biblioteca OpenCV es la instalación de dicha biblioteca en el equipo utilizado para el desarrollo de este proyecto. El equipo es un pc con sistema operativo Windows 7 de 64 bits, por lo tanto, se ha descargado e instalado la versión más reciente disponible para dicho sistema (2.4.13). OpenCV es una librería de código abierto, así, la librería está disponible de forma gratuita, para todo aquel que desee descargarla. Para su descarga se debe acceder a la web www.opencv.org. Una vez descargada la última versión de la biblioteca, se pide un directorio en el que alojar los archivos de la biblioteca, se ha decidido usar el directorio C:/.

El siguiente paso es registrar en las variables del sistema los directorios de OpenCV, para ello se debe:

- Seleccionar *Equipo* en el menú *Inicio*.
- Seleccionar *Propiedades del sistema* en el menú contextual.
- Hacer clic en *Configuración avanzada del sistema* > pestaña *Opciones avanzadas*.

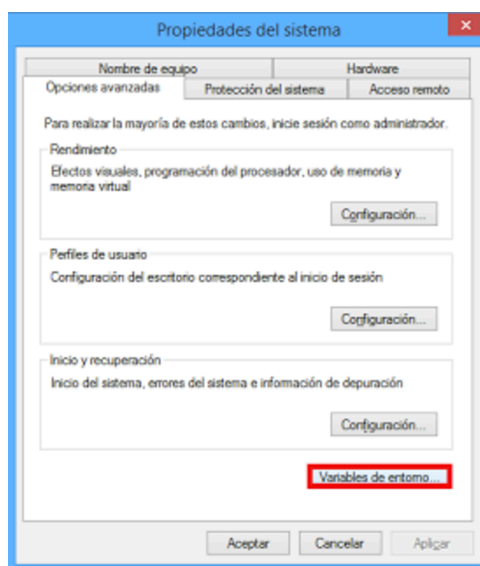


Figura 5.16 Configuración OpenCV

- Hacer clic en *Variables de entorno*, en *Variables del sistema*, buscar *PATH* y haga clic en él.
- Pulsar *Editar*, modificar *PATH* agregando a la ubicación de la clase el build de OpenCV (;C:\opencv\C:\opencv\build;).

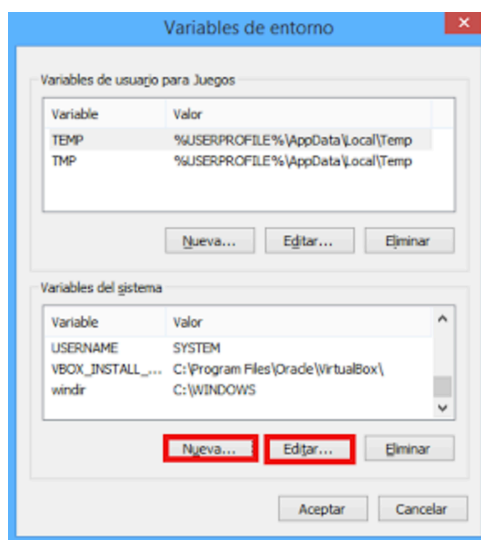


Figura 5.17 Configuración OpenCV

- Si no existe el elemento *PATH*, se puede optar por agregar una nueva variable, agregando *PATH* como el nombre y el build de OpenCV como la ubicación de ésta.

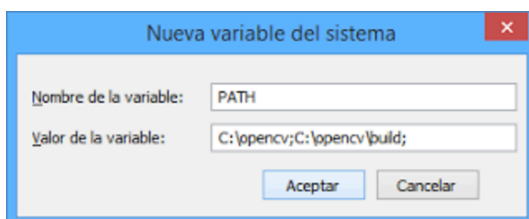


Figura 5.18 Configuración OpenCV

Una vez realizados estos pasos ya tenemos la librería debidamente instalada en el equipo, para utilizarla en el sistema de detección de rostros desarrollado en este proyecto.

5.2.5.2 Conexión vídeo cámara

El primer objetivo a realizar utilizando OpenCV es establecer conexión con la vídeo cámara ip y reproducir el flujo de vídeo emitido por ésta.

Usando la biblioteca OpenCV se puede acceder a cualquier dispositivo de captura que tengamos instalado en nuestro sistema. En el caso concreto de este proyecto se accede a una vídeo cámara ip (AXIS V5914), cuyas características serán descritas más

adelante. Dicha cámara consta de un servidor streaming, al que es posible acceder a través de la dirección ip 192.168.1.90. Cada una de las imágenes capturadas podrán ser almacenadas para su análisis o procesamiento en tiempo real, como se ha realizado en nuestro sistema. Existe una clase que nos servirá para guardar los videos previamente capturados y procesados, el formato de almacenamiento depende de las características habilitadas, pero puede ser MP4, AVI, WMV, etc., y otros si tenemos los códec.

Para acceder al flujo de vídeo de la cámara se utiliza la función de OpenCV *VideoCapture::open*. Es necesario definir un objeto de tipo *VideoCapture* en el que se almacene el flujo de vídeo. El proceso es análogo al seguido cuando se muestra un vídeo alojado en un directorio. En este caso, en lugar de especificar un archivo de vídeo a la clase *VideoCapture*, le indicaremos el identificador de dispositivo o índice de cámara que se desea usar. Ejemplo de uso: *VideoCapture cap.open("http://192.168.1.90/axis-cgi/mjpg/video.cgi?.mjpg");*

Se usa la función *cap.isOpened()* para verificar que la cámara se ha iniciado correctamente, y *cap.read(frame)*; envía la imagen capturada por la vídeo cámara al objeto Mat indicado. Mediante la función *imshow(window_name, frame)*, se reproduce en una ventana con nombre, asignado en el String *window_name*, el flujo de vídeo capturado del objeto MAT indicado.

5.2.5.3 Clasificadores de cascada Haar con OpenCV

La detección de objetos utilizando los clasificadores de cascada Haar fue el primer framework de detección de objetos efectivo propuesto por Paul Viola y Michael Jones en [22]. Este enfoque hace uso de la función matemática Wavelet Haar propuesta por Alfred Haar en 1909 y se basa en un método de aprendizaje donde la función de la cascada se entrena con muchas imágenes positivas (imágenes en las que aparece el objeto a detectar) y negativas (imágenes en las que no aparece el objeto a detectar). Asimismo, este enfoque se puede utilizar para detectar otra clase de objetos visuales. [61]

OpenCV facilita la tarea de detectar rostros o caras pues éste cuenta con clasificadores en cascada entrenados para esta tarea, estos clasificadores son los conocidos como Haar cascade classifiers y son almacenados en archivos XML. Además, si se desea se puede crear clasificadores propios para detectar el objeto para el que haya sido entrenado, por ejemplo: coches, animales, frutas, etc.[62] En nuestro caso concreto se han utilizado los clasificadores Haar, disponibles por defecto al descargar la librería OpenCV, para detectar rostros humanos.

Inicialmente, el algoritmo necesita muchas imágenes positivas (imágenes de caras) e imágenes negativas (imágenes en las que no aparecen caras) para entrenar al clasificador. En ese momento se extraen características de ella. Para esto, se usan las

características Haar que se muestran en la Figura 5.19, las cuales son el núcleo convolucional.

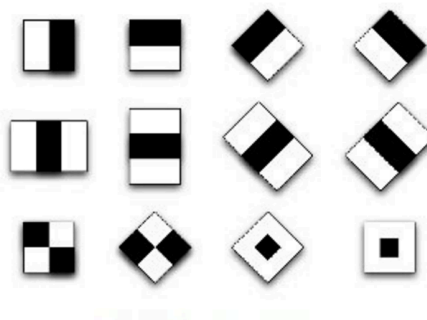


Figura 5.19 Clasificadores Haar

Todos los tamaños y ubicaciones posibles de cada clasificador se utilizan para calcular muchas características. Para el cálculo de cada característica, se necesita encontrar la suma de píxeles bajo rectángulos blancos y bajo rectángulos negros. Para resolver esto, se introdujeron las imágenes integrales, las cuales simplifican el cálculo de la suma de píxeles.

No obstante, entre todas estas características que se calculan, la mayoría de ellas son irrelevantes. Como ejemplo ilustrativo, se puede considerar la imagen de la Figura 5.20.

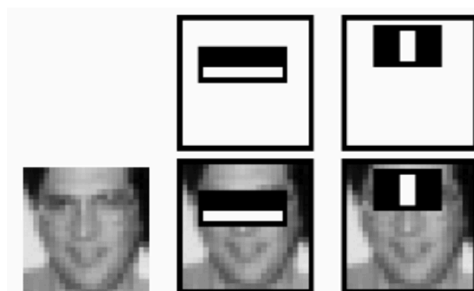


Figura 5.20 Ejemplo aplicación Clasificadores Haar

La fila superior muestra dos buenas características. La primera característica seleccionada parece centrarse en la propiedad de que la región de los ojos es a menudo más oscura que la región de la nariz y las mejillas. La segunda característica seleccionada se basa en la propiedad de que los ojos son más oscuros que el puente de la nariz. Pero las mismas ventanas aplicadas en las mejillas o en cualquier otro lugar de la cara, serían irrelevantes.

Para seleccionar las mejores características se utiliza Adaboost. Para ello, se aplican todas y cada una de las características en todas las imágenes de entrenamiento. Para cada característica, se encuentra el mejor umbral que clasificará las caras a positivo

y negativo. Pero, obviamente, habrá errores de clasificación. Se seleccionan las características con una tasa de error mínima, lo que significa que son las características que mejor clasifican las imágenes de cara y no cara. El proceso no es tan simple como esto, cada imagen recibe un peso igual al principio, después de cada clasificación se incrementa el peso de las imágenes mal clasificadas. A continuación se realiza el mismo proceso de nuevo, se calculan nuevas tasas de error. El proceso continúa hasta que se alcanza la precisión o tasa de error requerida o hasta que se encuentra el número necesario de características.

El clasificador final es una suma ponderada de estos clasificadores débiles. Se llaman débiles porque solos no pueden clasificar la imagen, pero junto con otros forman un clasificador fuerte.

Una ventana de 24x24 en una imagen tiene sobre 160000 características. Después de seleccionar las mejores características se puede reducir hasta obtener 6000 características. No obstante, comprobar las 6000 características es ineficiente computacionalmente. En una imagen, la mayor región de la imagen es la región en la que no aparece cara. Por lo tanto, previamente mediante un proceso sencillo se comprueba si la ventana a analizar es una región en la que no aparece cara. Si es así, se desecha y no se vuelve a procesar. Centrándose en las regiones donde puede haber una cara. De esta manera, se dedica más tiempo de computación para comprobar las regiones en las que es posible encontrar cara.

Por ello se introduce el concepto de Cascade of Classifiers (cascada de clasificadores). En lugar de aplicar todas las características en una ventana, se agrupan las características en diferentes etapas de los clasificadores y se aplican una por una (normalmente las primeras etapas contendrán un número muy pequeño de características). Si una ventana falla en la primera etapa, se desecha. No se comprobarán las características restantes en ella. Si por el contrario, la primera etapa es positiva, se aplica la segunda etapa de características y se continúa el proceso. La ventana que pasa por todas las etapas es una región de la cara.

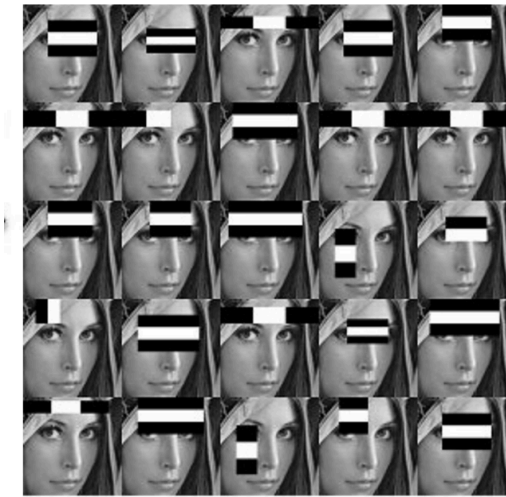


Figura 5.21 Cálculo de Haar sobre una imagen

El detector descrito en [22] tenía 6000 características con 38 etapas de: 1, 10, 25, 25 y 50 características en las primeras cinco etapas. Las dos características de la Figura 5.20 se obtienen realmente como las dos mejores características de Adaboost.

5.2.5.4 Detección de rostros en OpenCV

Para detectar un rostro primero se debe procesar la imagen en la cual se desea detectar un rostro, una vez cargada la imagen se debe aplicar los siguientes pasos:

Convertir la imagen a escala de grises, este paso es necesario para el correcto funcionamiento de los algoritmos de detección de caras usados por la biblioteca OpenCV. Para convertir una imagen a escala de grises o a otro formato contamos con la función `cvtColor`, la cual se utiliza del siguiente modo:

```
cvtColor(InputArray img_origen, OutputArray img_destino,
         int code);
```

Los parámetros de esta función son:

- **img_origen**: imagen de entrada: 8-bit sin signo, 16-bit sin signo (`CV_16UC...`), o precision simple con coma flotante.
- **img_destino**: imagen de salida del mismo tamaño que la **img_origen**.
- **code** : Código de conversión de espacio de color. Para la transformación a escala de grises se ha usado la constante `CV_BGR2GRAY`.

El siguiente paso que se debe aplicar es una ecualización de histograma a la imagen en escala de grises para estandarizar el contraste y brillo de la imagen. Este paso es necesario para que distintas condiciones de iluminación no afecten a la detección del rostro en la imagen, de este modo el algoritmo es más eficaz al detectar las caras presentes en una imagen.

```
equalizeHist(InputArray img_origen, OutputArray img_destino);
```

Esta función ecualiza la imagen de tipo Mat introducida como primer parámetro y la imagen resultante es guardada en el segundo parámetro especificado, el cual es también un objeto Mat del mismo tamaño.

La evolución del proceso seguido se muestran en la Figura 5.22.

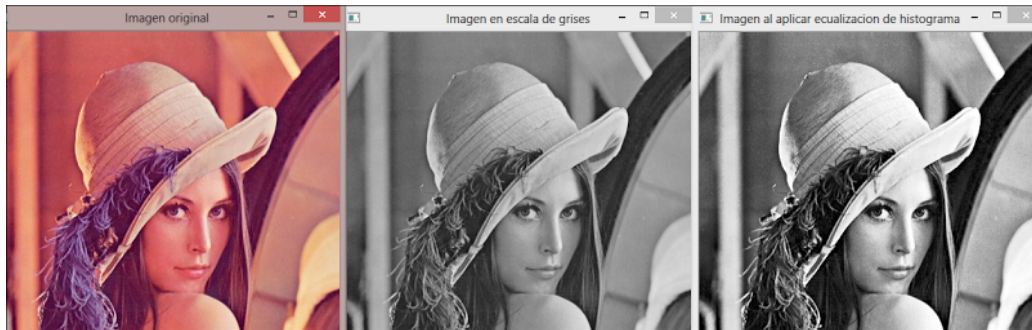


Figura 5.22 Proceso de ecualización de imágenes con OpenCV. De izquierda a derecha: imagen original, imagen en escala de grises e imagen ecualizada.

Con la imagen procesada y preparada ahora debemos cargar el detector que deseamos utilizar. Para ello, pasaremos el nombre del clasificador al método *load* de la clase *CascadeClassifier*. Los archivos .xml utilizados para la detección son los disponibles por defecto al descargar la librería, éstos se encuentran alojados en *C:\opencv\data*, donde se encuentran varias carpetas que contienen distintos tipos de clasificadores. En la carpeta *C:\opencv\data\haarcascades* se encuentran varios clasificadores no solo para detectar rostros sino también para la detección de ojos, boca, nariz, entre otros. En el caso concreto de este proyecto, para la detección de posibles hablantes, se han cargado tanto los clasificadores de rostros como los de ojos.

Para detectar rostros de frente se ha usado *haarcascade_frontalface_alt.xml* y para detectar ojos *haarcascade_eye.xml*. La inicialización para el detector de rostros con el archivo indicado se realiza del siguiente modo:

```
CascadeClassifier detector;  
if(!detector.load("haarcascade_frontalface_alt.xml"))  
cout << "No se puede abrir clasificador."<< endl;
```

Llegados a este punto, mediante la función *detectMultiScale* es posible detectar rostros de diferentes tamaños, presentes en la imagen. Las coordenadas de los rostros detectados se guardarán en la variable llamada **objects**:

```
vector<Rect>& objects;
```

```
detectMultiScale(const Mat& image,vector<Rect>& objects,
double scaleFactor=1.1,int minNeighbors=3,int flags=0,Size min
Size=Size(),Size maxSize=Size());
```

Los diferentes parámetros introducidos en esta función se explican a continuación:

- **image**: Matriz del tipo CV_8V que contiene una imagen en la que los objetos rostro y ojos quieren ser detectados.
- **objects**: Vector de rectángulos donde cada rectángulo contiene el objeto detectado.
- **scaleFactor**: Parámetro que especifica cuanto es reducida la imagen en cada escalado.
- **minNeighbors**: Parámetro que especifica cuántos vecinos tiene que tener cada rectángulo candidato para mantenerlo como rectángulo.
- **flags**: Parámetro con el mismo significado para una cascada antigua que en la función *cvHaarDetectObjects*. No se utiliza para una nueva cascada. Se ha configurado con la constante *CV_HAAR_FIND_BIGGEST_OBJECT*, consiguiendo así que si en la imagen capturada por la cámara existe más de un rostro el seguimiento se haga de acuerdo al rostro que se encuentre más cercano a la cámara.
- **minSize**: Tamaño mínimo del objeto a detectar. Objetos más pequeños serán ignorados.
- **maxSize**: Tamaño máximo del objeto a detectar. Objetos mayores serán ignorados.

La función *detectMultiScale* se ejecuta dos veces. En primer lugar se examina la imagen a procesar y se buscan los rostros que puedan existir, si el número de rostros es mayor de 1 entonces se ejecuta de nuevo la función *detectMultiScale* para detectar ojos en cada uno de los rostros detectados. Si en un rostro se detecta al menos 1 ojo, entonces el rostro detectado puede ser considerado definitivamente un rostro y se mostrará una elipse para indicar su localización en la imagen. Con este método de detección de rostros y ojos se reduce la detección de falsos positivos.

Para mostrar la elipse que indica la localización en la imagen de cada rostros se usa la función:

```
void ellipse(Mat& img,Point center,Size axes,double angle,
double startAngle,double endAngle,const Scalar& color, int thickness=1,
int lineType=8, int shift=0);
```

Los parámetros de los que depende esta función son:

- **img:** imagen en la que se desea detectar el rostro.
- **center:** centro de la elipse (centro de la cara detectada).
- **Axes:** mitad del tamaño del eje principal de la elipse.
- **Angle:** ángulo de rotación de la elipse en grados.
- **startAngle:** ángulo inicial del arco de la elipse en grados.
- **endAngle:** ángulo final del arco de la elipse en grados.
- **box:** representación alternativa de la elipse usando *RotatedRect* o *CvBox2D*. Esto significa que la función dibuja una elipse inscrita en el rectángulo rotado.
- **color:** color de la línea que define la elipse.
- **thickness:** grosor de la línea que define la elipse si es positivo, si es negativo significa que se dibuja un sector interior de la elipse.
- **lineType:** tipo de línea que define la elipse.
- **shift:** número de bits fraccionales en las coordenadas del centro y valor de los ejes.

El resultado final cuando una cara es detectada se muestra en la Figura 5.23.

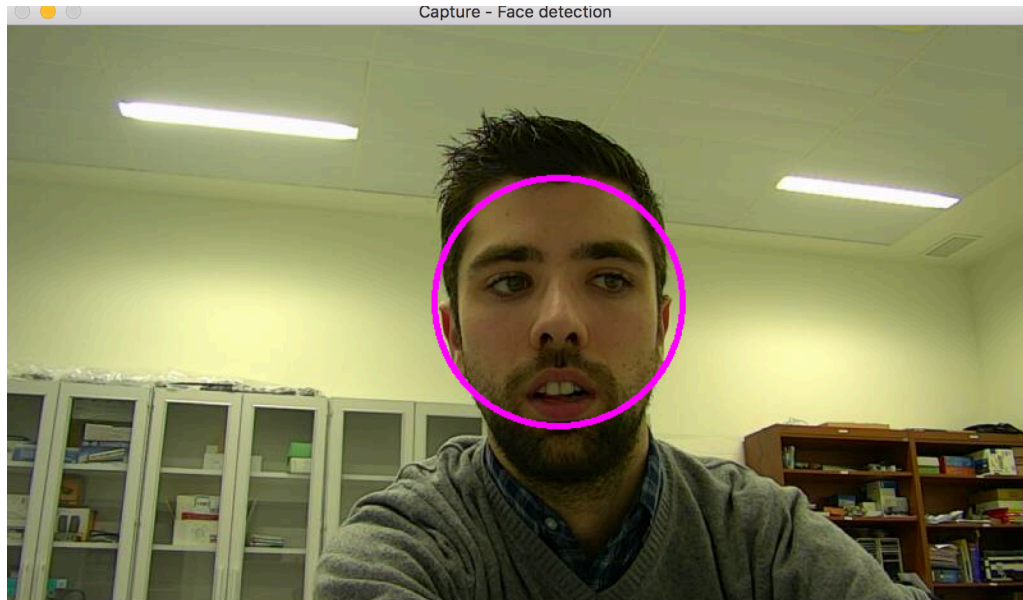


Figura 5.23 Detección de cara en el sistema de OpenCV desarrollado

5.2.5.5 *Control remoto de la cámara para seguimiento de rostros con OpenCV*

Una vez que se ha conseguido detectar un rostro o cara con la librería OpenCV, el siguiente paso es conseguir que la cámara pueda hacer un seguimiento de dicho rostro para tener localizado en todo momento, la posición del espacio en la que se encuentra la cara del hablante y poder dirigir la cámara hacia dicha dirección.

Por lo tanto para esta fase es necesario enviar comandos a la cámara para cambiar la posición hacia la que ésta enfoca. La vídeo cámara utilizada (AXIS V5914) permite varias formas de comunicación entre la cámara y el pc, como por ejemplo:

- Comunicación a través del protocolo Hypertext Transfer Protocol (HTTP).
- VAPIX, la propia Application Programming Interface de Axis.
- AXIS Media Control ActiveX, para uso desde ordenadores con sistema operativo Windows.

Se ha decidido emplear la comunicación mediante el protocolo HTTP, por su facilidad y sencillez.

Los comandos que dicha cámara admite mediante el protocolo http son explicados en el documento Pan Tilt Zoom API que ofrece el fabricante de la cámara. Dentro de este documento existe un apartado dedicado a la HTTP API. De entre los comandos posibles, los usados en este proyecto para controlar remotamente la cámara han sido:

- Comandos de control PTZ:
Comandos que permiten el control del pan (ángulo azimut), tilt (ángulo de elevación) y zoom de la cámara. El método HTTP empleado para mandar la petición del comando es tanto GET como POST.

La sintaxis general que siguen estos comandos viene dada por:

[http://<servername>/axiscgi/com/ptz.cgi?<argumento>=<valor>\[<argumento>=<valor>...\]](http://<servername>/axiscgi/com/ptz.cgi?<argumento>=<valor>[<argumento>=<valor>...])

De entre los posibles argumentos, los usados con estos comandos han sido:

Argumento	Valor posible	Descripción
center=<int>,<int>	<x>,<y>	Centra la cámara en la posición x,y donde x,y son las coordenadas de los píxeles en el stream de vídeo del cliente.
pan=<float>	-180.0 ... 180.0	Eleva el ángulo de la cámara a una determinada coordenada absoluta.
tilt=<float>	-180.0 ... 180.0	Mueve el ángulo azimut de la cámara a una determinada coordenada absoluta.
continuouspantiltmove=<int>,<int>	-100 ... 100, -100 ... 100	Movimiento continuo del pan y el tilt de la cámara. Valores positivos indican movimiento a la derecha (pan) y arriba (tilt), valores negativos indican movimiento a la izquierda (pan) y abajo (tilt). 0, 0 significa parar. Los valores se expresan como: <velocidad de pan>,<velocidad de tilt>

Tabla 5.1. Argumentos empleados de los comandos PTZ

Los comandos HTTP enviados a la cámara mediante el código C desarrollado en este proyecto, han sido encapsulados mediante la librería libcurl. La cual es una librería libre y fácil de usar de transferencia de URL del lado del cliente, apoyando DICT, FILE, FTP, FTPS, Gopher, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAP, POP3, POP3S, RTMP, RTSP, SFTP, SMTP, SMTPS, Telnet y TFTP. libcurl soporta certificados SSL, HTTP POST, HTTP PUT, carga de FTP, HTTP basado en formulario de carga, proxies, cookies, autenticación de usuario + contraseña (Basic, Digest, NTLM, Negotiate, Kerberos). El uso de esta librería se ha utilizado por medio de su API C correspondiente. Se ha hecho uso de la interfaz de fácil uso de dicha API, la cual consiste en:

Primero se inicializa la sesión y se obtiene un manejador, que se utiliza como entrada para las siguientes funciones de interfaz utilizadas.

Se continúa configurando todas las opciones, la más importante entre ellas es la propia URL.

Cuando todo está configurado, se le indica a libcurl que mande la petición HTTP. A continuación, realizará toda la operación y no volverá hasta que se haya completado o haya fallado.

Después de que esto se produzca, se puede obtener información de la petición y, a continuación, liberar el manejador de sesión empleado.

Estos pasos previamente descritos han sido realizados en cada comunicación con la cámara para modificar su posición. A continuación se muestra un código de ejemplo:

```
CURL *curl = curl_easy_init();

if(curl) {

    CURLcode res;
    curl_easy_setopt(curl,CURLOPT_URL,"http://example.com");
    res = curl_easy_perform(curl);
    curl_easy_cleanup(curl);

}
```

Las funciones implicadas son:

- ***CURL *curl_easy_init();***

Esta función debe ser la primera función a llamar y devuelve un identificador CURL que se debe utilizar como entrada para otras funciones en la interfaz. Esta llamada DEBE tener una llamada correspondiente a *curl_easy_cleanup* cuando la operación se completa.

- ***CURLcode curl_easy_setopt(CURL *handle, CURLOPToption option, parameter);***

Se utiliza para indicar a libcurl cómo comportarse. Al configurar las opciones apropiadas, la aplicación puede cambiar el comportamiento de libcurl. Todas las opciones se establecen con una opción seguida por un parámetro. Ese parámetro puede ser un *long*, un *puntero de función*, un *puntero de objeto* o un *curl_off_t*, dependiendo de lo que espere la opción específica.

- ***CURLcode curl_easy_perform(CURL *easy_handle);***

Invocando esta función después de *curl_easy_init* se realizan todas las llamadas *curl_easy_setopt*, y se realizará la transferencia como se describe en las opciones. Se debe llamar con el mismo manejador de entrada a *curl_easy_init*.

- ***void curl_easy_cleanup(CURL * handle);***

Esta función debe ser la última función a la que se llame en una sesión fácil. Es la opuesta de la función *curl_easy_init* y se debe llamar con el mismo manejador de entrada a *curl_easy_init*.

En este proyecto se ha utilizado el mismo código mostrado previamente cambiando `http://example.com` por el comando HTTP apropiado para cambiar la dirección hacia la que la cámara enfoca. De esta forma, cuando el software desarrollado detecta que la cara del hablante detectada no está centrada en la imagen, se gira la cámara los grados pan y tilt necesarios para centrar el centro de la cara del hablante en la imagen capturada, teniendo así, al hablante localizado y realizando un seguimiento de la posición de éste.

5.2.6 Hardware de captura de vídeo

Para la realización del proyecto en la fase de captura de vídeo, se ha utilizado una cámara ip modelo: AXIS V5914. Dicha cámara ha sido la encargada de capturar el flujo de vídeo con el que se va a localizar al hablante visualmente.



Figura 5.24 Vídeo cámara AXIS V5914

Primero se han analizado las características y funciones de dicha cámara, así como las diferentes posibilidades que existen para manejarla remotamente.

Las especificaciones técnica de este modelo de cámara son:

- Transmisión de vídeo HDTV (máxima resolución 1280x720)
- Zoom óptico de 30x con enfoque automático
- Tasa de fotogramas: 60 fotogramas por segundo
- Rango panorámico (grados): De -170 a +170
- Rango panorámico (grados): De -20 a +90
- Distancia focal: 4.3 – 129 mm
- Audio estéreo con calidad CD
- Conexiones disponibles: SDI, HDMI, Ethernet 10Base-T /100Base-TX
- Peso: 1.49 kg
- Dimensiones: 13.6 x 13.6 x 18 cm.

La cámara ha sido conectada punto a punto, mediante conexión Ethernet, con el pc encargado de ejecutar el software desarrollado, disminuyendo así los posibles retardos existentes en la transmisión del vídeo a éste. La cámara dispone de un servidor

web al cual es posible acceder a través de la ip 192.168.1.90 (dirección ip asignada por defecto). Desde este servidor web es posible manejar la cámara en cualquiera de las direcciones. Esta función sirve de testeo para la funcionalidad que va a tener en este proyecto. Por lo que, en un principio y como toma de contacto con el software a desarrollar, se ha manejado la cámara mediante el servidor web, descubriendo las diferentes funciones PTZ (Pan Tilt Zoom) que ésta ofrece.

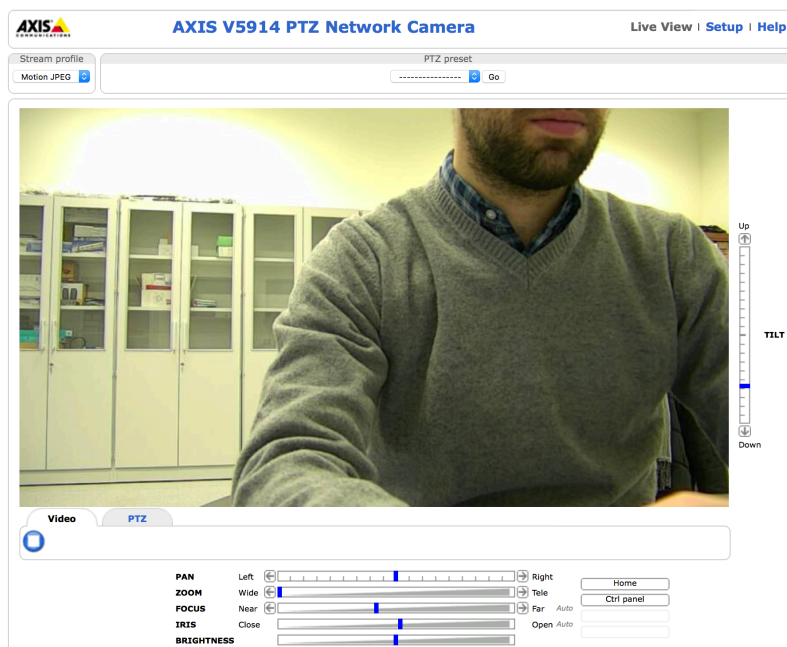


Figura 5.25 Servidor streaming y control de la vídeo cámara

En los ajustes de la cámara, se ha configurado ésta para que sea accesible por cualquier usuario sin contraseña. Siendo así más fácil, la comunicación de la vídeo cámara con el pc. En el *servername* de la sintaxis de comandos HTTP enviados a la cámara, sólo será necesario especificar la dirección ip de ésta. No siendo necesario especificar ninguna contraseña. El número de fotogramas por segundo que la cámara captura, se ha limitado a: 15 fps. La resolución a la que la cámara captura vídeo es de: 800 x 450 (16:9).

Una vez realizados estos pasos el hardware de procesamiento de vídeo está preparado para ejecutar el software desarrollado en esta fase para detección facial de hablantes. El objetivo es que la cámara enfoque hacia la dirección en la que se encuentra el hablante, controlada por el software OpenCV desarrollado.

5.3 Fusión Audio - Visual

Una vez que se han implementado tanto la detección de hablantes por medio del procesado de audio como la detección de rostros por medio del procesado de vídeo, es el momento de implementar la fusión de ambos sistemas, es decir, la fusión audio-visual.

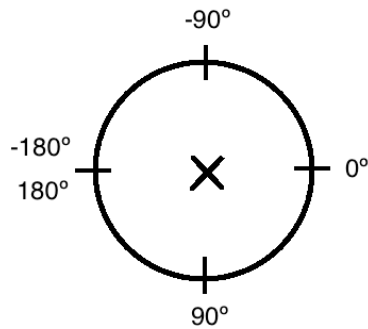
El primer paso en esta fase del proyecto ha sido la adaptación de ambos sistemas de localización para poder ser fusionados, componiendo así el SLAV definitivo.

En el sistema de localización de hablantes mediante procesado de audio ha sido necesario cambiar el sistema de referencia. Previamente cuando éste actuaba de manera independiente la posición de referencia ($x=0$, $y=0$, $z=0$) de la sala se consideraba una de las esquinas de la sala en la que se realizaban las pruebas y era desde esta referencia desde la que se indicaba la posición del hablante en la sala. Debido a la necesidad de que este proyecto pueda ser implementado en robótica ha sido necesario cambiar la posición de referencia ($x=0$, $y=0$, $z=0$) a la posición en el espacio en la que se encuentra localizado el centro del sistema SLAV desarrollado, es decir, el centro de la base de la vídeo cámara y el centro del array de micrófonos.

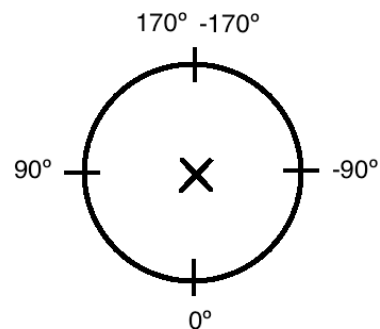
El área mallada por el sistema en su fase de fusión se ha reducido, fijando unas dimensiones de: $-1.5 : x : 1.5$, $-2 : y : 2$, $0 : z : 1$ (en metros). Al reducir las dimensiones de este mallado se consigue reducir el tiempo de computación del sistema de localización auditivo.

Además, cuando el sistema de localización de hablantes por medio de procesado de audio actuaba de manera independiente, acabado su proceso emitía la posición estimada del hablante en coordenadas cartesianas, para poder así comprobar el desempeño del sistema de localización. En este momento la posición estimada del hablante se ha convertido a coordenadas esféricas (azimut, elevación y distancia radial) para poder enviarlas a la cámara, la cual trabaja con este tipo de coordenadas. Dirigiendo el flujo de vídeo hacía una determinada dirección en la que se estima que está el hablante. Aún así, el sistema de referencia de los grados azimut tanto de la cámara como del sistema de localización por procesado eran diferentes, como se muestra en la Figura 5.26.

Sistema de referencia localización mediante audio



Sistema de referencia vídeo cámara

*Figura 5.26 Sistemas de referencia*

Estas imágenes muestran el sistema de referencia desde una vista de planta. La X del centro de la circunferencia representa el centro del SLAV. Como se puede ver entre ellos existe un desfase de 90° . Para solucionar esto se ha modificado el sistema de referencia de audio para hacerlo coincidir con el de la cámara del siguiente modo:

- Si el ángulo detectado $\geq -90^\circ \rightarrow$ se suman 90° al ángulo detectado.
- En caso contrario $\rightarrow$ se suman 270° .

Además, en el sistema de referencia de la vídeo cámara los ángulos máximo y mínimo son 170° y -170° , esto ya ha sido mencionado en las especificaciones técnicas de dicha cámara. Por lo que si la posición de un hablante es localizada en un ángulo inferior a -170° o mayor de 170° entonces la cámara apuntará al azimut -170° ó 170° debido a que la cámara no es capaz de rotar 360° .

El sistema de detección visual, en un principio ajustaba tanto el ángulo de elevación como el ángulo azimut cuando se realizaba un seguimiento de rostro de manera independiente. Para ello empleaba el argumento “*continuouspantiltmove*” de la tabla 5.1. Este comando al enviarlo a la cámara modificaba el ángulo de elevación y/o el ángulo azimut hasta conseguir fijar en el centro del frame de visión, la cara del hablante. Pero al ser un movimiento de la cámara continuo, no permitía que otro comando fuera enviado a la cámara, manteniendo la cámara bloqueada mientras se realizaba el ajuste visual. En el aparatado de fusión esta implementación se ha modificado. Ahora mediante procesamiento de vídeo con OpenCV se obtiene el centro de la cara del hablante, y mediante el comando “*center*” se sitúa el centro de la cara en el centro del frame de visión con un solo comando, posibilitando que el sistema de audio pueda cambiar el azimut de la cámara en cualquier momento. Como se ha mencionado previamente la precisión del sistema de audio en la detección del ángulo azimut es suficientemente precisa, pero una

vez obtenido dicho ángulo azimut, es necesario ajustar el ángulo de elevación de la cámara con procesado visual.

Realizadas estas modificaciones se han fusionado ambas partes del SLAV, para ello se ha desarrollado un sistema que se ha configurado del siguiente modo:

- En primer lugar, se ejecuta la detección de hablantes por medio del procesado de vídeo. Con este sistema de localización se obtiene una muy buena precisión en el ángulo azimut (pan) pero la localización en el ángulo de elevación (tilt) no es precisa, como se mostrará en el apartado de resultados y conclusiones.
- Por ello, una vez que se ha dirigido la cámara hacia un determinado azimut, se ejecuta la detección de rostros del procesado visual obteniendo el centro de la cara del hablante si lo hubiese. En el caso de que el sistema de detección visual detecte una cara, se comprueba que la diferencia en las coordenadas del centro de la cara con respecto a la posición actual es mayor de 20 píxeles en cualquiera de los dos ejes (horizontal y vertical). Si es así, se modifica la dirección a la que apunta la cámara, de lo contrario la cara detectada se considera en la misma posición y no es necesario ajustar visualmente de nuevo el frame de visión.
- Por otro lado y de forma continuada en un hilo de ejecución independiente, se reproduce el vídeo capturado por la vídeo cámara.

Al iniciar el sistema de detección, el ángulo de elevación de la vídeo cámara se ha fijado en un punto intermedio (0° de elevación) desde el que poder detectar posibles caras de hablantes.

A modo de aclaración se muestra el diagrama del funcionamiento del SLAV desarrollado, Figura 5.27:

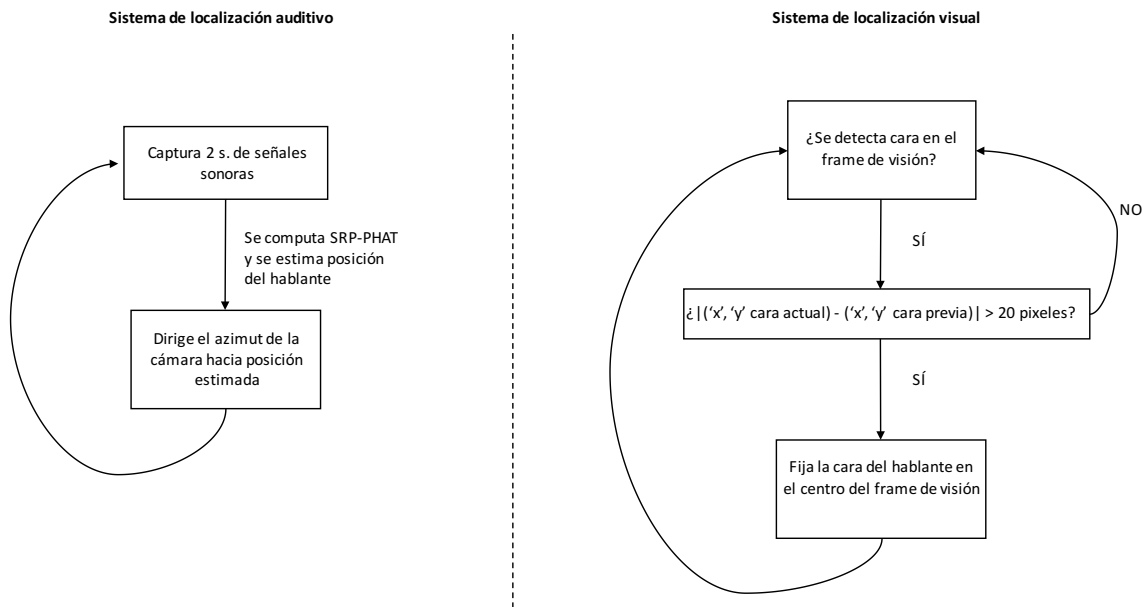


Figura 5.27 Diagrama funcionamiento de SLAV

5.3.1 Hardware fusionado

Del mismo modo que los software de ambos sistema de localización han sido fusionados, el hardware de ambos sistemas obviamente también se ha fusionado. La Figura 5.28 muestra una imagen del hardware del SLAV desarrollado:

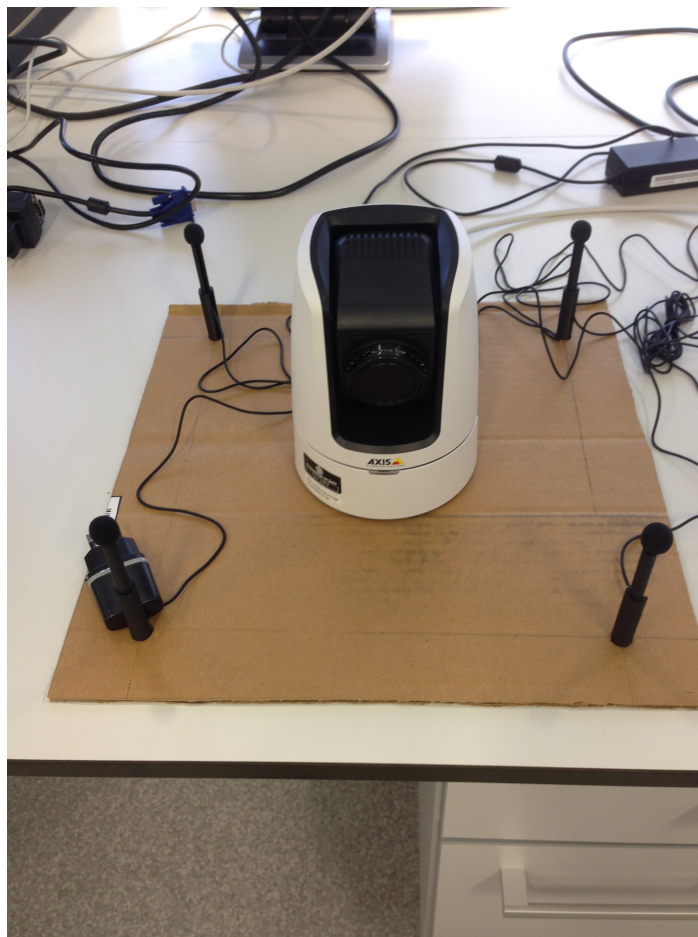


Figura 5.28 Hardware del SLAV desarrollado

Se ha utilizado una base de cartón sobre la que se han ubicado los 4 micrófonos que componen el array de micrófonos, según la geometría del array finalmente elegida que ha sido especificada en el apartado de procesado de audio. En el centro de dicho array de micrófonos se ha ubicado el centro de la base de la vídeo cámara, así ambos sistemas tienen una referencia común. Con esta configuración los 4 micrófonos pueden recibir la señal sin producirse oclusiones o reflexiones significativas y a su vez la vídeo cámara puede capturar vídeo, sin que la imagen se vea ocultada por los micrófonos, ya que la lente de dicha cámara se encuentra más elevada que los micrófonos.

Un factor importante a la hora de fusionar ambos sistemas ha sido lo especialmente silenciosa que es la vídeo cámara cuando sus motores rotan tanto su ángulo azimut como su ángulo de elevación. Al producir un ruido muy leve, éste no afecta al sistema de localización de audio para la detección.

Esta disposición del hardware no es excesivamente voluminosa, sus medidas no superan los 30 cm. de lado y su altura es de 18 cm. (altura de la vídeo cámara), por lo que sería posible utilizarla tanto en sistemas de robots autónomos, como en un posible sistema de videoconferencia.

6 RESULTADOS Y DISCUSIÓN

6.1 Procesado de audio

6.1.1 Simulación Off-line

El paso previo a la ejecución en tiempo real, del sistema de localización de hablantes mediante procesado de audio, ha sido su simulación off-line, pudiendo determinar así la mejor geometría para el array de micrófonos, así como la influencia de la distancia entre micrófonos en la precisión de la localización. Para la ejecución de dichas pruebas previas se ha utilizado un software llamado Roomsim [47; 48].

6.1.1.1 Simulaciones acústicas con Roomsim

Roomsim es un software de Matlab que es capaz de simular un entorno acústico como una habitación con forma de caja de zapatos (shoebox). El objetivo de los creadores del programa Roomsim era proporcionar una herramienta de generación de señales a la comunidad investigadora relacionada con temas de habla y audición, así como una herramienta para ilustrar el método de simulación de imagen acústica y algunos efectos acústicos. Se trata de un programa de libre disposición a través de Internet bajo un Licencia Pública General GNU. Está dirigido mediante un menú, tiene una guía de usuario y ejemplos de archivos de datos.

El primer paso para usar dicho software ha sido su descarga y su ejecución desde el software Matlab.

Una vez descargado el software se debe elegir una ubicación para alojar la carpeta que contiene dicho software. Se ha optado por ubicar la carpeta en el escritorio del pc asignado al proyecto.

Cuando el software de simulación quiera ser ejecutado, se debe abrir el software Matlab y elegir el directorio en el que está alojado Roomsim.

En el command window de Matlab se introduce el comando *roomsim* y en este momento se abre el menú de la Figura 6.1.

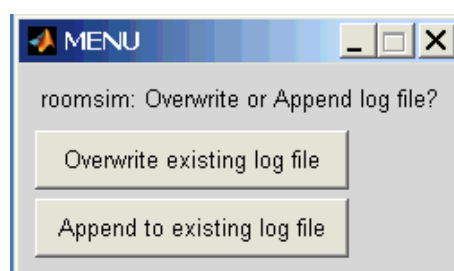


Figura 6.1 Primer menú Roomsim

Seleccionamos la opción *Overwrite existing log file* y aparece el menú de la Figura 6.2.



Figura 6.2 Segundo menú Roomsime

Seleccionamos la opción *Set-up and Run the Room Simulation* con la que comenzaremos la configuración del entorno acústico en el que se a realizar la simulación.

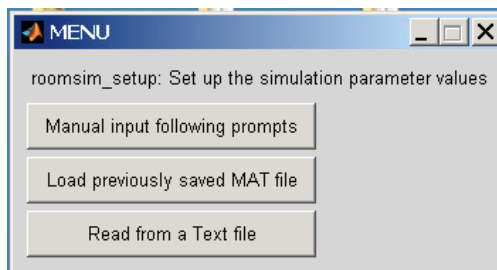


Figura 6.3 Tercer menú Roomsime

Si se dispone de un archivo .mat con la configuración del entorno acústico a simular, se debe seleccionar *Load previously saved MAT file* (Figura 6.3). En el caso de la primera vez que se simule el entorno se selecciona la opción *Manual input following prompts*.

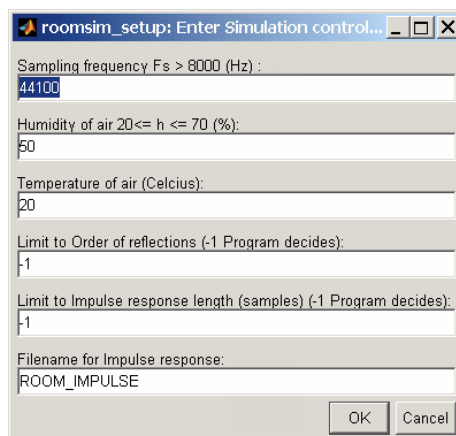


Figura 6.4 Cuarto menú Roomsime

En el primer campo del menú se selecciona la frecuencia de muestreo de nuestro sistema (Figura 6.4). Se selecciona además la humedad, la temperatura, el límite del orden de las reflexiones de la sala (por defecto -1, el programa lo decide), así como el límite de la longitud de la respuesta al impulso (por defecto -1, el programa lo decide) y por último el nombre del fichero en el que se guardará la respuesta al impulso del entorno acústico simulado.

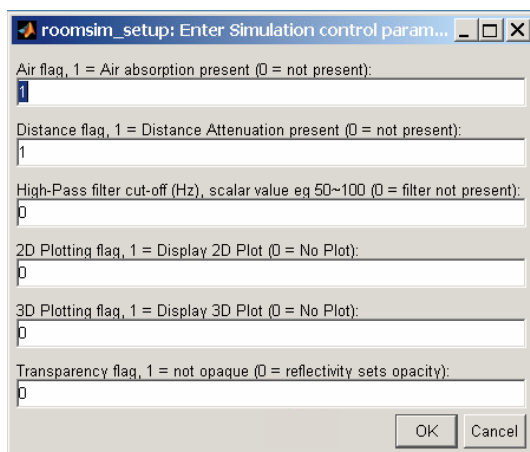


Figura 6.5 Quinto menú Roomsime

En el menú mostrado en la Figura 6.5 se configuran más parámetros de control de la simulación. En las simulaciones realizadas estos parámetros se han configurado con los valores por defecto.

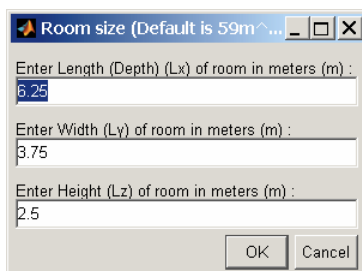


Figura 6.6 Quinto menú Roomsime

En el menú de la Figura 6.6 se especifican las dimensiones de la sala (shoebox) en la que se va a realizar la simulación acústica. Pulsamos OK y nos aparecerá la pregunta de la Figura 6.7.

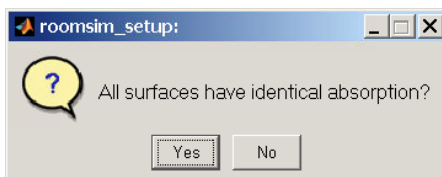


Figura 6.7 Sexto menú Roomsime

Se nos pregunta si todas las paredes, suelo y techo de la sala simulada tienen el mismo coeficiente de absorción, es decir, si las paredes, suelo y techo de la sala son del mismo material. En las simulaciones realizadas en este proyecto se ha supuesto que todas las superficies de la sala eran iguales por lo que se pulsa YES. Entonces nos parecerá el menú para seleccionar el tipo de material del que están compuestas las superficies de la sala (Figura 6.8):

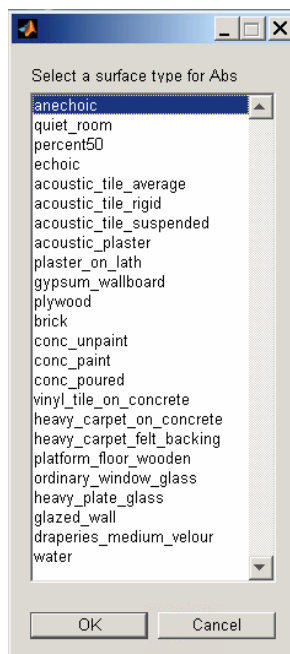


Figura 6.8 Séptimo menú Roomsim

Podemos ver que el software Roomsim nos permite elegir entre una gran variedad de materiales tales como: azulejos, yeso, madera contrachapada, ladrillo, vinilo, moqueta, etc. Sin embargo, en nuestras simulaciones se ha considerado un entorno anecoico, evitando así reflexiones indeseadas en el sistema. Este sería un entorno ideal, pero para el estudio preliminar de la geometría del array de micrófonos a implementar, era suficiente.

A continuación, es necesario introducir las coordenadas cartesianas del receptor, es decir, las coordenadas cartesianas de la posición del hablante en la sala simulada (Figura 6.9).

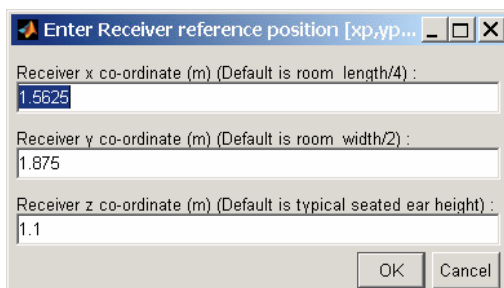


Figura 6.9 Octavo menú Roomsim

El criterio que sigue Roomsim con las coordenadas cartesianas es:

- x : Es el ancho de sala.
- y : Es el fondo o largo de la sala.
- z : Es la altura del techo de la sala.

Introducimos dichas coordenadas y pulsamos OK. El siguiente paso es introducir las coordenadas esféricas de la posición de cada uno de los micrófonos que compongan el array de micrófonos (Figura 6.10)

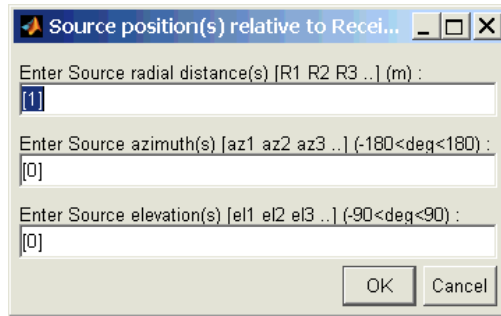


Figura 6.10 Noveno menú Roomsim

A partir de las coordenadas cartesianas de las posiciones de los micrófonos se han obtenido las coordenadas esféricas mediante:

$$distancia_radial = \sqrt{(x_s - x_{mic})^2 + (y_s - y_{mic})^2 + (z_s - z_{mic})^2} \quad (16)$$

$$Azimut = \frac{180}{\pi} \cdot \left(\tan^{-1} \left(\frac{(y_s - y_{mic})}{(x_s - x_{mic})} \right) \right)$$

$$Elevación = \frac{180}{\pi} \cdot \left(\tan^{-1} \left(\frac{(z_s - z_{mic})}{\sqrt{(x_s - x_{mic})^2 + (y_s - y_{mic})^2}} \right) \right)$$

donde x_s, y_s, z_s son las coordenadas cartesianas de la posición de la fuente sonora y $x_{mic}, y_{mic}, z_{mic}$ son las coordenadas cartesianas de la posición de un determinado micrófono.

A continuación se puede elegir el número de sensores del sistema receptor y de la fuente (Figura 6.11).

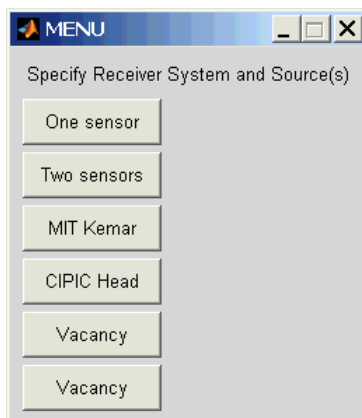


Figura 6.11 Décimo menú Roomsim

En las simulaciones realizadas se ha configurado como un solo sensor. El siguiente menú nos permite configurar la bandera de suavizado a la respuesta de nuestro sistema (Figura 6.12). Si se ha seleccionado un solo sensor, este será el único flag que se pueda activar.

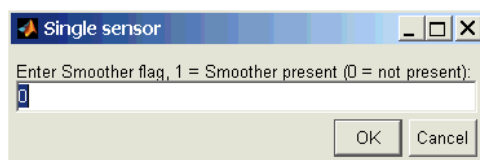


Figura 6.12 Décimo primer menú Roomsim

En las simulaciones realizadas este valor se ha mantenido a 0, estando por tanto inactivo. En el siguiente menú se selecciona el tipo de sensor a utilizar (Figura 6.13).

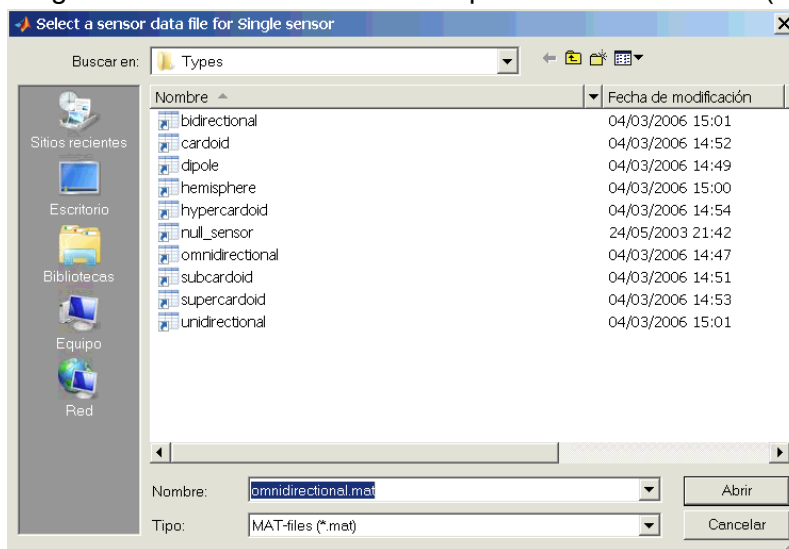


Figura 6.13 Décimo segundo menú Roomsim

Se ha simulado considerando sensores omnidireccionales, ya que los micrófonos que se han usado son de este tipo. Posteriormente, se permite la posibilidad de guardar la configuración del entorno simulado para poder ser cargado más adelante en otras simulaciones.

Al finalizar la simulación se obtiene la siguiente respuesta del sistema:

Por un lado se muestra un modelo 3D de la sala simulada con la posición del hablante y el micrófono (Figura 6.14).

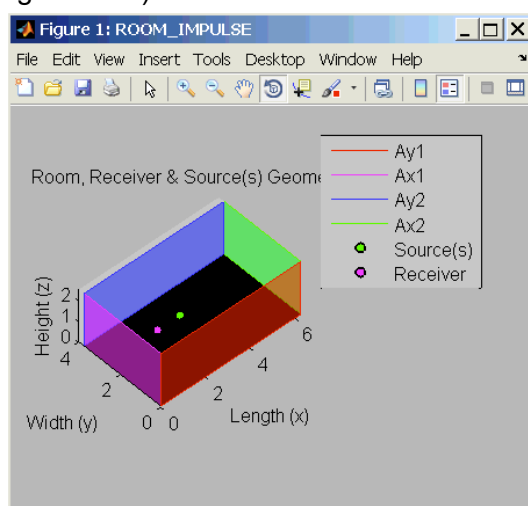


Figura 6.14 Modelo 3D del entorno acústico simulado con Roomsim

Además se muestra un gráfico con los coeficientes de absorción de cada superficie a diferentes frecuencias (Figura 6.15).

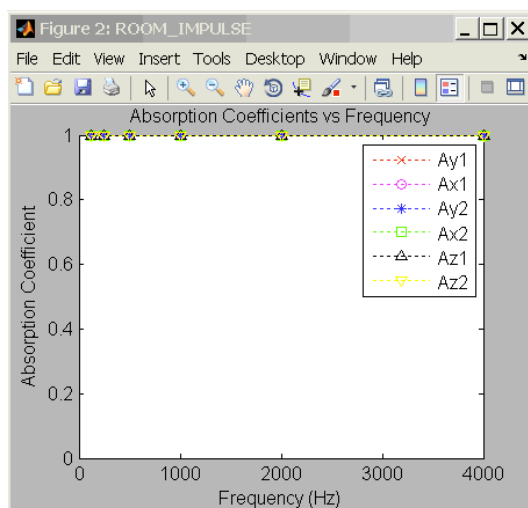


Figura 6.15 Coeficientes de absorción del entorno acústico simulado con Roomsim

Se puede apreciar que el coeficiente de absorción es 1 para cualquier frecuencia, esto es debido a que se ha simulado un entorno anecoico.

Por último se muestra un menú con la posibilidad de representar la respuesta al impulso del sistema y la energía relativa del sistema.

Estos pasos previos se han realizado para simular cada micrófono que compone el array de micrófonos a simular. Al final se obtiene la respuesta al impulso del hablante, para cada uno de los micrófonos. Estas respuestas al impulso han sido convolucionadas con una señal de voz anecoica y se ha obtenido la señal final capturada por los micrófonos simulados. Esta señal es la que se ha introducido en nuestro sistema de localización de hablantes off-line, para comprobar la precisión de la localización del sistema.

6.1.1.2 Estudio de diferentes geometrías de array

Utilizando el software Roomsim de Matlab se han realizado diferentes simulaciones con diferentes configuraciones de arrays de micrófonos, configurando el entorno acústico como anecoico y de dimensiones: 6.25 x 3.75 x 2.5 m. Se han realizado simulaciones con diferentes número retardos precalculados (taus equiespaciados utilizados para el cálculo de la correlación cruzada) y con una $F_s=44100$ Hz y 16000Hz. Se han comparado las dos configuraciones implementadas para determinar el tau del que se acumulan sus correlaciones. Una en la que se estima la posición mediante el retardo más próximo al retardo equiespaciado y otra en la que la posición es estimada realizando una interpolación lineal entre los retardos equiespaciados. Los arrays simulados han sido:

- **Array lineal de 6 micrófonos:** Micrófonos alineados en el eje x con una separación entre micrófonos de 0.1 m.

La configuración para esta prueba es:

- Coordenadas cartesianas de la posición del hablante: $x=4.2$ $y=2.0$ $z=2.0$
- Coordenadas cartesianas del centro del array de micrófonos: $x=3.125$ $y=1.0$ $z=1.5$

En las siguiente tablas se muestran los resultados obtenidos para esta configuración de array de micrófonos.

Considerando una velocidad del sonido de 341,7 m/s son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	0.0 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4,1	1,4	0,6	4,1	1,4	2,4
362	4	1,5	0,8	4,2	1,4	0,5
543	4,1	1,4	0,6	4	1,4	1,4
724	4,2	1,6	0,6	4	1,5	2,2

Con el mismo mallado pero con una velocidad del sonido de 343,0 m/s, se obtiene:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.2 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4	1,2	0,6	4,1	1,5	2,4
362	4,1	1,5	0,6	4,1	1,5	2,4
543	4,1	1,5	0,6	4,1	1,5	2,4
724	4,1	1,5	0,6	4,1	1,5	2,4

De esta prueba podemos determinar que el valor de la velocidad del sonido, dependiente de la temperatura de la sala según ecuación (14), no varía los resultados de la posición obtenida significativamente. Por lo que, a partir de esta prueba la velocidad del sonido se configurará a 343 m/s. Por otro lado, se aprecia como en la coordenada 'x' el valor obtenido es muy próximo al valor real de la posición del hablante, en ambas configuraciones tanto para tau interpolado como tau más próximo pero en la coordenada 'y', 'z' la precisión en la localización no es tan buena. Aún así en ambas configuraciones se aprecia como el valor de las coordenadas 'y', 'z' es más preciso cuando se calcula el tau interpolado. Obteniéndose para velocidad de sonido = 343 m/s un error relativo de 2.38% en 'x', 25% en 'y' y 20% en 'z'.

- **Array cúbico de 6 micrófonos:** Un micrófono en el centro de cada una de las caras del cubo. Separación de los micrófonos de 0.2 m en el mismo eje.

La configuración para esta prueba es:

- Coordenadas cartesianas de la posición del hablante: $x=4.2$ $y=2.0$ $z=2.0$
- Coordenadas cartesianas del centro del array de micrófonos: $x=3.125$ $y=1.0$ $z=1.5$

En las siguiente tablas se muestran los resultados obtenidos para esta configuración de array de micrófonos.

Los resultados obtenidos son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	0.0 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	5,1	2,9	2,4	4,7	2,5	2,2
362	4,9	2,7	2,3	4,7	2,5	2,2
543	4,9	2,7	2,3	4,7	2,5	2,2
724	4,7	2,5	2,2	4,7	2,5	2,2
905	4,7	2,5	2,2	4,7	2,5	2,2

Para tau interpolado se obtiene un error relativo de 11.9% en 'x', 25% en 'y' y 10% en 'z'. Se han mejorado los resultados en la coordenada 'z', a costa de una peor precisión en la coordenada 'x', manteniéndose constante el error relativo en 'y'.

Se ha realizado una simulación con este mismo array pero, acotando el mallado al área en la que se encuentra el hablante. Los resultados son:

		Mallado x:	Mallado y:	Mallado z:		
		3.0 : 0.1 : 5.0	1.5 : 0.1 : 2.5	1.5 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4,5	2,3	2,1	4,5	2,3	2,1
362	4,5	2,3	2,1	4,5	2,3	2,1

Se puede apreciar como la precisión en la localización ha mejorado. Además, en este caso los resultados para el método del tau más próximo y el interpolado son los mismos. Se obtiene un error relativo de 7,14% en 'x', 15% en 'y' y 5% en 'z'. Por lo tanto en la coordenada 'y' se obtienen los peores resultados. En las siguientes pruebas se ha acotado el mallado solo para esta coordenada, variando el salto en la definición del mallado de cada coordenada:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.0 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4,2	2	2	4,2	2	2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 6.25	1.0 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4	2	2	4,5	2,5	2
362	4,5	2,5	2	4,5	2,5	2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.25	1.0 : 0.2 : 3.75	0.0 : 0.2 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4,8	2,6	2,2	4,2	2	2

En la primera y la tercera simulación para el tau interpolado, se obtiene exactamente el valor de la posición del hablante. En la segunda, al configurar un mallado con mayores saltos, se comprueba que se pierde precisión.

- **Array en cruz con 6 micrófonos:** Geometría de array simulada para tratar de conseguir mayor precisión en la coordenada de profundidad 'y'. Consta de 2 micrófonos en el eje 'x' (separación entre micros de 0.1 m), 4 micrófonos en el eje 'y' (separación entre micros de 0.2 m). Con esta geometría se ha simulado al hablante en diferentes posiciones de la sala.

- Coordenadas cartesianas del centro del array de micrófonos: x=3.125 y=1.0 z=1.5

- La primera simulación se ha realizado con coordenadas cartesianas de la posición del hablante: x=2.6 y=2.8 z=1.6 obteniéndose como resultados:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.25	1.2 : 0.2 : 3.75	0.0 : 0.2 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	2,6	2,8	1,2	2,6	2,8	1,2
362	2,6	2,8	1,2	2,6	2,8	1,2
543	2,6	2,8	1,2	2,6	2,8	1,2
724	2,6	2,8	1,2	2,6	2,8	1,2
905	2,6	2,8	1,2	2,6	2,8	1,2

Con esta geometría de array se ha mejorado la precisión en la localización de la coordenada 'y' a costa de una reducción en la precisión de la coordenada 'z' (error relativo del 25%).

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.2 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	2,6	2,8	1,2	2,8	2,1	1,4
362	2,8	2,1	1,4	2,8	2,1	1,4
543	2,8	2,1	1,4	2,8	2,1	1,4
724	2,8	2,1	1,4	2,8	2,1	1,4
905	2,8	2,1	1,4	2,8	2,1	1,4

Se aprecia como al configurar un mallado más fino, los resultados de localización obtenidos son peores, respecto a la configuración previa en la que para el tau interpolado se obtienen las coordenadas 'x', 'y' exactas.

A continuación se han configurado mallados con dos saltos diferentes, estos son los resultados:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 6.25	1.2 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	2,5	3,2	2	2,5	3,2	2
362	2,5	3,2	2	2,5	3,2	2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.3 : 6.25	1.2 : 0.3 : 3.75	0.0 : 0.3 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	2,7	2,4	1,8	2,7	2,4	1,8

Al modificar el salto los resultados son peores en las tres coordenadas, en cuanto a precisión, respecto a la prueba con mallado de 0.2 m.

- La segunda simulación se ha realizado con coordenadas cartesianas de la posición del hablante: x=6.2 y=2.8 z=1.6, una posición en la que el hablante se encuentra más próximo a la pared derecha de la habitación:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.2 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4,8	2	1,2	4,8	2	1,4
362	4,8	2	1,5	4,8	2	1,4
543	4,8	2	1,4	4,8	2	1,4

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.25	1.2 : 0.2 : 3.75	0.0 : 0.2 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4,8	2	1,6	4,8	2	1,4
362	4,8	2	1,6	4,8	2	1,4
543	4,8	2	1,4	4,8	2	1,4
724	4,8	2	1,4	4,8	2	1,4
905	4,8	2	1,4	4,8	2	1,4
1086	4,8	2	1,4	4,8	2	1,4

Se aprecia claramente como los resultados son peores respecto a las pruebas previas, con el mismo tipo de mallado pero con el hablantes situado en (x=2.6, y=2.8, z=1.6) un posición más centrada en la habitación. El motivo de que la precisión haya

disminuido es que cuando el hablante no está tan centrado en la sala, la localización es más complicada.

A continuación se realizan más pruebas con el mismo mallado de la prueba anterior:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.3 : 6.25	1.2 : 0.3 : 3.75	0.0 : 0.3 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4,5	1,8	1,5	4,5	1,8	1,5
362	4,5	1,8	1,8	4,5	1,8	1,5
543	4,5	1,8	1,5	4,5	1,8	1,5
724	4,5	1,8	1,5	4,5	1,8	1,5
905	4,5	1,8	1,5	4,5	1,8	1,5

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 6.25	1.2 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	5	2,2	1	5	2,2	1
362	5	2,2	1	5	2,2	1
543	5	2,2	1	5	2,2	1
724	5	2,2	2	5	2,2	1
905	5	2,2	1	5	2,2	1

Los resultados siguen siendo peores al encontrarse el hablante más alejado y pegado a la pared de la habitación.

- La tercera simulación de esta geometría se ha realizado con coordenadas cartesianas de la posición del hablante: $x=3.6$ $y=2.8$ $z=1.6$, una posición en la que el hablante se encuentra más próximo al centro de la habitación:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.2 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3,8	3,3	1,5	3,7	3	1,4
362	3,8	3,4	1,2	3,8	3,4	1,3
543	3,7	3	1,4	3,8	3,4	1,3

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.25	1.2 : 0.2 : 3.75	0.0 : 0.2 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3,8	3,4	1,2	3,8	3,4	1,4
362	3,8	3,4	1,2	3,8	3,4	1,4
543	3,8	3,4	1,4	3,8	3,4	1,4
724	3,8	3,4	1,4	3,8	3,4	1,4

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.3 : 6.25	1.2 : 0.3 : 3.75	0.0 : 0.3 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3,6	2,7	1,8	3,6	2,7	1,8
362	3,6	2,7	1,8	3,6	2,7	1,8
543	3,6	2,7	1,8	3,6	2,7	1,8
724	3,6	2,7	1,8	3,6	2,7	1,8

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 6.25	1.2 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3,5	2,2	1,5	3,5	2,2	1,5
362	3,5	2,2	1,5	3,5	2,2	1,5
543	3,5	2,2	1,5	3,5	2,2	1,5
724	3,5	2,2	1,5	3,5	2,2	1,5

Podemos apreciar como al localizar al hablante más centrado en la sala y más alienado con el centro del array de micrófonos, los resultados obtenidos mejoran, obteniéndose un error relativo para el mejor caso (mallado de 0.3 m) de: 0% en x, 3.57% en y, 12.5% en z. Hasta el momento los resultados en la precisión de localización con los arrays propuestos, no proporcionan un error relativo pequeño, por esto se propone otra nueva geometría de array para ser evaluada.

- **Array lineal de 6 micrófonos:** Micrófonos alineados en el eje y (eje de la profundidad de la sala) con una separación entre micrófonos de 0.1 m. El hecho de simular esta geometría se debía a la poca precisión en la coordenada y obtenida en las pruebas realizadas hasta el momento. Con esta geometría se ha simulado al hablante en diferentes posiciones del eje x de la sala.

- La primera posición del hablante simulada ha sido: x=6.0, y=3.0, z=2.0 y los resultados obtenidos con diferentes mallados son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.3 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	0.0	3.2	1.8	0.0	3.5	0.2
362	0.0	3.4	0.6	6.1	3.5	0.0
543	0.0	3.5	0.0	0.0	3.5	0.2
	0.1	3.5	0.0	0.0	3.5	0.2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.25	1.3 : 0.2 : 3.75	0.0 : 0.2 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	0.2	3.5	0.0	0.0	3.5	0.2
362	0.0	3.3	1.0	0.0	3.5	0.2
543	0.0	3.3	1.0	0.0	3.5	0.2
724	0.0	3.5	0.2	0.0	3.5	0.2
905	6.0	3.3	0.4	0.0	3.5	0.2
1086	6.0	3.3	0.4	0.0	3.5	0.2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.3 : 6.25	1.3 : 0.3 : 3.75	0.0 : 0.3 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	0.9	3.1	0	0.3	3.4	0.0
362	0.0	3.4	0.6	0.3	3.4	0.0
543	0.0	3.4	0.6	0.3	3.4	0.0
724	0.3	3.4	0.0	0.3	3.4	0.0
905	0.0	3.4	0.6	0.3	3.4	0.0
1086	0.0	3.4	0.6	0.3	3.4	0.0

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 6.25	1.3 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	1.5	2.8	0.0	1.5	2.8	0.0
362	1.5	2.8	0.0	1.5	2.8	0.0
543	1.5	2.8	0.0	1.5	2.8	0.0
724	1.5	2.8	0.0	1.5	2.8	0.0
905	1.5	2.8	0.0	1.5	2.8	0.0

Se puede ver como el error relativo de la coordenada 'y' en el mejor de los casos (mallado de 0.5 m) es de 6.66% pero la precisión de localización en coordenadas 'x', 'z' se ha reducido considerablemente, con errores relativos de: 75 % en 'x', 100% en 'z'. No obstante, antes de descartar dicha geometría se han realizado pruebas en diferentes posiciones para comprobar si se produce el efecto que se ha producido con la geometría de array en cruz con 6 micrófonos, con la que se ha comprobado que se obtienen peores resultados de localización si el hablante se encuentra más próximo a las paredes de la sala.

- Posición del hablante simulada: x=3.0, y=3.0, z=2.0. Los resultados obtenidos con diferentes mallados son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.3 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	2,8	3,6	2	3,1	3,6	2
362	3	3	1,9	3,4	3,6	1,9
543	2,9	3,1	2	2,9	3,6	2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.25	1.3 : 0.2 : 3.75	0.0 : 0.2 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3	2,9	1,8	3,2	3,5	2
362	3,2	1,9	2	3,4	3,5	2
543	3	2,3	2	3,2	3,5	1,8
724	3	3,1	1,8	3,4	3,5	2
905	3	2,3	2	3	3,5	1,8
1086	3	3,3	1,8	3,4	3,5	2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.3 : 6.25	1.3 : 0.3 : 3.75	0.0 : 0.3 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3,3	3,1	1,8	3	3,4	2,1
362	3	2,5	2,1	3,3	3,4	1,8
543	3	2,8	2,1	3,3	3,4	2,1
724	3,3	3,1	2,1	3,3	3,4	1,8
905	3	2,8	2,1	3	3,4	1,8
1086	3	2,5	2,1	3,3	3,4	1,8

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 6.25	1.3 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3	2,3	2	3	2,8	2
362	3	2,3	2	3	2,3	2
543	3	2,3	2	3	2,3	2
724	3	2,3	2	3	2,3	2
905	3	2,3	2	3	2,3	2

Se comprueba que los resultados que se obtienen, con esta geometría lineal en el eje 'y', son sustancialmente mejores, cuando el hablante se encuentra más próximo al eje 'y' que define el array de micrófonos. En el mejor de los casos (mallado de 0.2 m para tau más próximo) se obtiene un error relativo de: 0% en 'x', 10% en 'y', 10% en 'z'.

Debido a estos buenos resultados obtenidos, se ha decidido realizar una última simulación con esta geometría, situando al hablante en una posición muy cercana a la pared izquierda de la sala, es decir, el caso opuesto a la primera simulación realizada con esta geometría.

- Posición del hablante simulada: x=1.0, y=3.0, z=2.0. Los resultados obtenidos con diferentes mallados son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.25	1.3 : 0.1 : 3.75	0.0 : 0.1 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	1,5	2,6	1,5	4	2,7	0,4
362	4,1	2,6	0,6	4,5	2,5	1,2
543	4,6	2,4	2	4,6	2,4	2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.25	1.3 : 0.2 : 3.75	0.0 : 0.2 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3,2	2,5	0,4	4	2,7	0,4
362	3	2,5	0,4	4,6	2,5	1,4
543	2,2	2,3	1	4,6	2,5	1,4
724	4,6	2,5	1,4	4,6	2,5	1,4
905	4,6	2,5	1,4	4,6	2,5	1,4
1086	3,2	2,5	0,4	4,6	2,5	1,4

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.3 : 6.25	1.3 : 0.3 : 3.75	0.0 : 0.3 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	3,9	2,5	0,6	1,2	2,5	0,6
362	4,5	2,5	1,2	4,5	2,5	1,2
543	4,5	2,5	1,2	4,5	2,5	1,2
724	4,5	2,5	1,2	4,5	2,5	1,2
905	3,9	2,5	0,6	4,5	2,5	1,2
1086	3,9	2,5	0,6	4,5	2,5	1,2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 6.25	1.3 : 0.5 : 3.75	0.0 : 0.5 : 2.5		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	1,5	2,8	1	5	2,8	1,5
362	5	2,8	1,5	4,5	2,3	2
543	5	2,8	1,5	4,5	2,3	2
724	5	2,8	1,5	4,5	2,3	2
905	4,5	2,3	2	4,5	2,3	2

Los resultados obtenidos vuelven a no ser precisos, como ocurría cuando el hablante se situaba cerca de la pared derecha de la sala. Obteniéndose en el mejor de los casos (mallado de 0.2 m), un error relativo de: 360% en 'x', 16,66% en 'y', 30% en 'z'. En el eje 'x' la detección es nula. Además, en esta simulación se ha detectado que existe una gran dispersión en las posiciones detectadas para cada una de las tramas procesadas. Queda por tanto descartada este tipo de geometría.

- **Array 6 micrófonos esquinas y centro paredes:** Array en el que los micrófonos se disponen en el centro y las esquinas de cada pared de la habitación en la que se quiere localizar al hablante.

Dado que hasta el momento las geometrías probadas no proporcionan una localización precisa en las tres coordenadas cartesianas, se han revisado varias publicaciones científicas, analizando las geometrías de array empleadas en cada una de ellas. Como es el caso de [16] en el que se ha empleado un array de 6 micrófonos. En este caso, la disposición cambia radicalmente respecto a las geometrías probadas hasta

el momento. Los micrófonos se localizan a una altura ‘z’ media de la sala, en cada una de las esquinas y el centro de la pared de la sala a simular. Existiendo una mayor distancia entre los micrófonos del array. En la siguiente figura se muestra el plano con la disposición utilizada para los micrófonos.

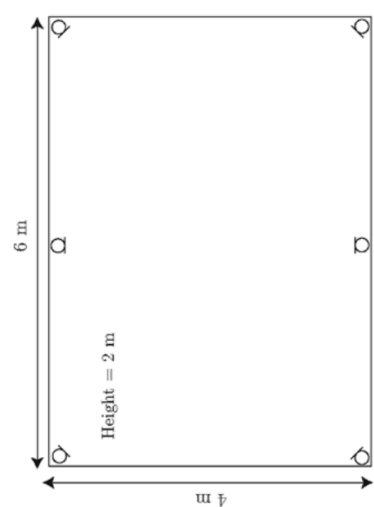


Figura 6.16 Disposición de micrófonos en [16]

- Se ha simulado una sala con las mismas dimensiones de [16]. Localizando al hablante en x=2.2, y=3.2, z=1.6, los resultados que se obtienen para diferentes mallados del área a detectar son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 4.0	0.0 : 0.1 : 6.0	0.0 : 0.1 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
1810	2.2	3.2	0.4	2.2	3.2	0.4
1086	2.2	3.2	0.4	2.2	3.2	0.4
543	2.2	3.2	0.4	2.2	3.2	0.4
181	2.2	3.2	0.4	2.2	3.2	0.4

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 4.0	0.0 : 0.5 : 6.0	0.0 : 0.5 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
1810	2.5	3.0	0.5	2.5	3.0	0.5
1086	2.5	3.0	0.5	2.5	3.0	0.5
543	3.5	3.5	0.0	2.5	3.0	0.5
181	1.0	4.0	1.0	2.5	3.0	0.5

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.05 : 4.0	0.0 : 0.05 : 6.0	0.0 : 0.05 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
1810	2.2	3.2	0.4	2.2	3.2	0.4
1086	2.2	3.2	0.45	2.2	3.2	0.45
543	2.2	3.2	0.35	2.2	3.2	0.4
181	2.2	3.2	0.25	2.2	3.2	0.4

Con este tipo de geometría de array se aprecia claramente que tanto la localización en la coordenada 'x' como 'y' es exacta para los mallados de 0.1 y 0.05 m. Y en el caso del mallado de 0.5 m. el punto 'x', 'y' obtenido es el punto más próximo dentro de los puntos que definen dicho mallado. Debido a estos buenos resultados se ha decidido simular el mismo sistema pero invirtiendo las coordenadas 'x', 'y' del hablante. Así, la posición del hablante en este caso es: x=3.2, y=2.2, z=1.6, de esta manera se comprobará si efectivamente dicha geometría es tan precisa para estas coordenadas. Los resultados obtenidos son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 4.0	0.0 : 0.1 : 6.0	0.0 : 0.1 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
1810	3.2	2.2	0.4	3.2	2.2	0.4
1086	3.2	2.2	0.4	3.2	2.2	0.4
543	3.2	2.2	0.4	3.2	2.2	0.4
181	3.2	2.2	0.5	3.2	2.2	0.4

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.5 : 4.0	0.0 : 0.5 : 6.0	0.0 : 0.5 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
1810	3.0	2.0	1.0	3.0	2.0	1.0
1086	3.5	2.0	0.0	3.0	2.0	0
543	3.5	2.0	0.0	3.0	2.0	0.0
181	3.5	2.0	0.0	2.0	2.0	1.0

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.05 : 4.0	0.0 : 0.05 : 6.0	0.0 : 0.05 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
1810	3.2	2.2	0.4	3.2	2.2	0.4
1086	3.2	2.2	0.4	3.2	2.2	0.4
543	3.2	2.2	0.4	3.2	2.2	0.4
181	3.15	2.25	0.2	3.2	2.2	0.45

Se comprueba que la localización en 'x', 'y' es perfecta. El único problema que sigue existiendo es la localización en 'z', de la que podemos ver que los resultados obtenidos no son nada precisos. Además, uno de los posibles fines que puede tener este proyecto es su implementación en un sistema de robot autónomo. Esta geometría es útil

en una sala de videoconferencia, pero no podría ser incorporada a un robot debido a sus dimensiones.

- **Array de 4 micrófonos para robótica:** Se trata de una geometría cuadrada con cuatro micrófonos en un mismo plano 'z' y con una separación entre micrófonos de 0.17 m.

De las pruebas previas se ha comprobado que el aumento de los taus equiespaciados no mejora los resultados de localización, si no que éstos permanecen constantes. Lo que sí implica, es un mayor tiempo de computación, con lo cual las pruebas sucesivas se han realizado con un menor número de taus.

Buscando una geometría de array adaptable a un sistema de robótica y con la que se consigan resultados de localización precisos se ha tomando como referencia la geometría usada en [8], cuyo array propuesto se ha simulado con un hablante localizado en: $x=3.7$, $y=4.0$, $z=1.6$ con estos resultados:

			Mallado x:	Mallado y:	Mallado z:		
			0.0 : 0.1 : 5.0	0.0 : 0.1 : 6.0	0.0 : 0.1 : 2.0		
			POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
			tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z	
543	4.2	4.9	1.9	4.2	4.9	1.9	
181	4.2	4.9	1.4	4.2	4.9	1.9	

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

La localización en 'x', 'y' no es tan precisa coma la obtenida con la geometría de [16], error relativo en el mejor de los casos (mallado de 0.05 m) de: 2,7% en 'x', 3,75% en 'y'. Lo que sí se consigue con esta geometría es que los resultados en 'z' mejoren respecto a [16] (error relativo de 18,75% en 'z').

- Para comprobar el desempeño de dicha geometría, se ha simulado al hablante desde otra posición de la sala: $x=2.0$, $y=5.0$, $z=1.6$. Los resultados obtenidos son:

		Mallado x:		Mallado y:		Mallado z:		
		0.0 : 0.1 : 5.0		0.0 : 0.1 : 6.0		0.0 : 0.1 : 2.0		
		POSICIÓN ESTIMADA						
		tau Próximo			POSICIÓN ESTIMADA			
					tau interpolado			
Nº taus	x	y	z	x	y	z		
543	2.0	5.3	1.7	2.0	5.4	1.7		
181	1.9	5.8	1.9	2.0	5.3	1.7		

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.05 : 5.0	0.0 : 0.05 : 6.0	0.0 : 0.05 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
543	1.95	5.7	1.8	1.95	5.65	1.8
181	1.95	5.75	1.85	1.9	5.95	1.95

En el mejor de los casos (mallado de 0.1 m) se obtiene un error relativo de: 0% en 'x', 6% en 'y', 6.25 % en 'z'. Resultados incluso mejores que para la posición previamente simulada.

- **Array en cruz con 6 micrófonos:** Geometría de array simulada para tratar de conseguir mayor precisión en la coordenada de profundidad 'y', pero con mayor separación entre micrófonos. Esta geometría está basada en el estudio realizado hasta el momento. Consta de 2 micrófonos en el eje 'x' (separación entre micros de 1 m), 4 micrófonos en el eje 'y' (separación entre micros de 0.5 m). Con esta geometría se ha simulado al hablante en diferentes posiciones de la sala.

- Se ha simulado al hablante en la posición: x=2.6, y=2.8, z=1.6. Se obtienen estos resultados:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 6.0	0.0 : 0.1 : 4.0	0.0 : 0.1 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	2.6	2.8	1.5	2.6	2.8	1.5
362	2.6	2.8	1.6	2.6	2.8	1.6
543	2.6	2.8	1.6	2.6	2.8	1.5

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 6.0	1.2 : 0.2 : 4.0	0.0 : 0.2 : 2.0		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	2.6	2.8	1.6	2.6	2.8	1.6
362	2.6	2.8	1.6	2.6	2.8	1.6
543	2.6	2.8	1.6	2.6	2.8	1.6

Podemos ver que la precisión es del 100% en todas las coordenadas. Esta geometría el único inconveniente que tiene es que no es aplicable a un robot autónomo, ya que el array empleado tiene unas dimensiones de 1 m de ancho por 1.5 m. de fondo, lo cual es demasiado para un sistema de este tipo.

- **Array de 4 micrófonos para robótica implementado en este trabajo:**

Basada en la geometría de [8] pero con mayor distancia entre micrófonos. Ya que según las pruebas realizadas con la geometría previa, se ha podido comprobar que la localización mejora a medida que aumentamos la distancia entre micrófonos. Se trata de una geometría con cuatro micrófonos en un mismo plano z y con una separación entre micrófonos de 0.3 m (separación mayor que [8] pero menor que la geometría previa, para poder ser aplicable a un robot móvil).

Se ha simulado dicha geometría con el software Roomsim emulando una sala con las dimensiones del laboratorio de investigación L-135 del Campus Científico Tecnológico de Linares. El motivo de elegir esta sala es porque será donde se van a realizar posteriormente las pruebas en tiempo real.

- El hablante se ha localizado en (x=4.0, y=4.5, z=1.7) los resultados obtenidos son:

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.1 : 5.5	0.0 : 0.1 : 9.0	0.0 : 0.1 : 3.3		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4.0	4.5	0.1	3.9	4.7	0.2
362	3.9	4.7	0.2	3.9	4.7	0.2
543	3.9	4.7	0.2	3.9	4.7	0.2

		Mallado x:	Mallado y:	Mallado z:		
		0.0 : 0.2 : 5.5	1.2 : 0.2 : 9.0	0.0 : 0.2 : 3.3		
		POSICIÓN ESTIMADA			POSICIÓN ESTIMADA	
		tau Próximo			tau interpolado	
Nº taus	x	y	z	x	y	z
181	4.0	4.4	0.0	4.0	4.4	0.0
362	4.0	4.4	0.0	4.0	4.4	0.0
543	4.0	4.4	0.0	4.0	4.4	0.0

Sea aprecia que los resultados tanto en 'x' como en 'y' son muy buenos. Consiguiendo una precisión casi perfecta. El error relativo en el mejor de los casos es de 0% en x, 2.22% en y. La precisión en 'z' es nula pero, del estudio de las diferentes geometrías de investigaciones previas, se ha podido comprobar que los resultados obtenidos en la coordenada 'z', tampoco eran precisos, como ocurre en [16]. Además, esta geometría cuenta con unas medidas que hacen que sea posible su implementación en un robot autónomo. Por todo ello, esta ha sido la geometría elegida para realizar las pruebas en tiempo real y la que será incorporada al sistema SLAV desarrollado (Figura 6.17).



Figura 6.17 Array de micrófonos implementado en este proyecto

6.1.2 Entorno acústico para pruebas de SLAV

Las pruebas en tiempo real del algoritmo SRP-PHAT para localización de hablantes, han sido realizadas en el laboratorio de investigación L-135 del Campus Científico Tecnológico de Linares, un entorno totalmente real. Dicho laboratorio cuenta con unas dimensiones de: 5.5 x 9 x 3.3 m. No es una sala anecoica, como la simulada en las pruebas de Roomsim. Sus paredes están compuestas de azulejos y yeso. Una de las paredes es un ventanal, con lo cual pueden existir reflexiones de las ondas sonoras en dicha superficie. Al tratarse de un laboratorio de investigación, en ella existen ordenadores (ruido de ventiladores), mesas, armarios con puertas de cristal y diferentes objetos en los que las ondas sonoras se pueden reflejar.



Figura 6.18 Entorno de pruebas para SLAV

Este ha sido también el entorno usado para las pruebas de procesamiento de vídeo. Para la detección de rostros por medio de OpenCV la iluminación de fluorescentes y las puertas de cristal de los armarios son factores bastante problemáticos para una correcta detección, que provocan la detección de falsos positivos.

Las pruebas con el sistema fusionado SLAV también se han realizado en él. Los resultados obtenidos se mostrarán más adelante.

6.1.3 On-line(real time)

Una vez realizadas todas las pruebas off-line previamente descritas y habiendo elegido la geometría del array que se va a usar, esta geometría ha sido implementada físicamente y se han realizado una serie de pruebas on-line para comprobar su funcionamiento en un entorno real, no un entorno anecoico como las simulaciones realizadas.

Para las pruebas en tiempo real, el primer paso es obtener la posición real del hablante para cada prueba, pudiendo así posteriormente comparar entre la posición real y la estimada por el sistema. Para ello se ha utilizado un medidor láser. En cada prueba

se han capturado 2 segundos de audio. En las pruebas en tiempo real se ha configurado el número de retardos equiespaciados en 181. Del estudio de las pruebas off-line se ha comprobado que aumentar el número de taus no supone una mejor detección y sin embargo, lo que sí implica es una mayor tiempo computacional. Este sistema está pensado para funcionar en tiempo real por lo que se debe buscar reducir el tiempo de ejecución del sistema de localización lo máximo posible. El mallado configurado cubre todo el espacio del laboratorio, con un salto entre cada punto del mallado de 0.1 m, es decir, un mallado muy fino para poder tener así una muy buena precisión (10 cm.) en la localización del hablante.

El centro del array de micrófonos se ha situado en las coordenadas cartesianas: $x=3.55$, $y=2.9$, $z=1.0$ del laboratorio.

Se han realizado pruebas desde diferentes posiciones del laboratorio para poder probar la precisión del sistema de localización, los resultados que se han obtenido son:

Nº taus			Mallado x:	Mallado y:	Mallado z:			
181			0.0 : 0.1 : 5.5	0.0 : 0.1 : 9.0	0.0 : 0.1 : 3.3			
POSICIÓN REAL DEL HABLANTE			POSICIÓN ESTIMADA tau Próximo			POSICIÓN ESTIMADA tau interpolado		
x	y	z	x	y	z	x	y	z
4.4	5.3	1.55	4.4	5.2	0.5	4.4	5.1	0.5
2.4	5.2	1.55	2.6	4.9	0.6	2.4	4.7	0.5
3.7	1.0	1.55	3.7	2.2	0.5	3.7	0.9	0.5
2.5	1.2	1.55	2.6	1.0	0.6	2.5	1.1	0.5
1.6	4.5	1.55	1.5	4.4	0.7	1.6	4.4	0.6
5.0	3.5	1.55	4.8	3.4	0.6	5.1	3.5	0.6
4.5	1.3	1.55	4.5	1.2	0.6	4.5	1.3	0.6

La coordenada 'z' toma siempre valor 1.55 m debido a que en las pruebas que se han realizado, el hablante está de pie y esta es la altura de su boca respecto del suelo, que sería $z=0$ m.

Se puede apreciar la gran precisión del sistema de localización en las coordenadas 'x', 'y' llegando a ser con el método de tau interpolado una localización perfecta en muchas de las pruebas. Por otro lado, en la coordenada 'z' la localización es casi nula, este efecto ya ocurría en las pruebas off-line.

Ambos métodos son bastante precisos en la localización de 'x', 'y', pero el método del tau interpolado es levemente mejor, por lo que este método ha sido el implementado para el sistema de localización de audio, al ser fusionado con el vídeo.

EL siguiente paso ha sido realizar pruebas para testear el sistema de localización auditivo, antes de ser fusionado, es decir, cuando el sistema de referencia se encuentra en el centro de SLAV. La casi nula precisión obtenida en la coordenada 'z' ha supuesto que los resultados obtenidos en el ángulo de elevación no sean nada precisos. Por ello,

la detección del ángulo de elevación del hablante se implementará mediante el sistema de procesamiento de vídeo.

6.2 Procesado de vídeo

En el siguiente vídeo se muestra una prueba realizada con el sistema de localización visual.

<https://youtu.be/tPkqk-Q0ips>

En ella se puede comprobar como la cara detectada se marca con una elipse de color fucsia. La persona que aparece en el flujo de vídeo de la cámara se mueve a la derecha, izquierda, arriba y abajo. La cámara es capaz de realizar un seguimiento de la posición de la persona en la sala.

6.3 Fusión audio-visual

Una vez testeados ambos sistemas de localización de manera individual, en la tercera fase del proyecto se ha llevado a cabo el testeo del sistema fusionado (SLAV).

Se han realizado diversas pruebas de dicho sistema. Considerando que la mejor manera de mostrar sus resultados es un documento visual en el que se muestre el desempeño de éste.

- **Prueba_1:**

En esta prueba existen tres hablantes que se dirigen hacia SLAV en instantes de tiempo diferentes. Inicialmente, los hablantes están sentados de frente a SLAV, entre ellos existe 1 m. de distancia aproximadamente.

Al principio de la prueba podemos ver como el hablante que se encuentra más a la izquierda habla y el azimut de SLAV apunta hacia él. Posteriormente el hablante que se encuentra en el centro comienza a hablar y se aprecia como el sistema dirige la atención hacia él. Cuando dicho hablante se levanta de la silla se puede detectar como el sistema de procesamiento de vídeo hace un ajuste del ángulo de elevación de la vídeo cámara, dejando así al hablante centrado en su frame de visión. Por último el hablante que se encuentra a la derecha comienza a hablar y el sistema dirige el azimut de la cámara hacia él.

Seguidamente uno de los hablantes se dirige al sistema desde dos de las posiciones más alejadas posibles, dentro del entorno usado para las pruebas. Primero habla desde una posición más centrada y se puede apreciar como SLAV dirige la atención hacia él. A continuación habla desde una de las esquinas de la sala y de nuevo SLAV dirige su atención hacia él. Estas posiciones no pertenecen al área mallada por el

sistema, pero éste lo detecta como si fuera el punto del mallado intermedio, en la diagonal entre SLAV y el punto desde el que está el hablante, ya que el trayecto directo de la señal sonora, se propaga a través de dicha diagonal. En estas posiciones más alejadas el sistema de detección de caras no actúa, ya que ha sido configurado para poder detectar caras de hablantes no más alejados de 2 m.

Por último el hablante se dirige hacia SLAV desde la parte posterior del sistema mostrando que el sistema es capaz de localizar hablantes en 360°.

Se muestra como SLAV consigue una alta precisión en la detección del ángulo azimut desde el que está el hablante, ajustando a cada hablante en el centro de su frame de visión.

Durante la realización de la prueba se puede apreciar como existe ruido de fondo (ventiladores de ordenadores, ruido del teléfono...) que pueden en momentos concretos generar un falso positivo en el sistema de localización y que éste dirija su atención hacia estas posiciones.

También es posible apreciar que la localización del ángulo azimut de los hablantes, está sujeta a un leve retardo del sistema. Esto es debido a que el sistema de audio realiza una detección cada 2.5 segundos (2 segundos de captura y 0.5 segundos de procesado). Por tanto, debe transcurrir dicho tiempo para que el sistema pueda realizar una localización del azimut del hablante. No obstante se trata de un delay asumible para el tipo de aplicaciones en las que se plantea usar el presente sistema y, en el peor de los casos, el sistema se estabiliza cada vez que hay una persona hablando durante varios segundos, situación esperable en este tipo de aplicaciones.

La prueba realizada se muestra en dos vídeos.

- Un vídeo grabado con una cámara móvil que muestra el escenario de detección y el flujo de vídeo emitido por la cámara y recibido en el PC:

<https://youtu.be/hRwchuPrXBk>

- Un vídeo grabado con una cámara desde una posición fija en la que se muestran los movimientos realizados por la cámara:

<https://youtu.be/XZ-y6ciEz7k>

- **Prueba_2:**

Por medio de esta prueba, en la que existen dos hablantes de frente a SLAV con menos de 1 m. de distancia entre ellos, se pretende mostrar la precisión del sistema en la detección del ángulo azimut.

El vídeo que muestra la prueba realizada es el siguiente:

<https://youtu.be/NfQBUXlt49A>

- **Prueba_3:**

En esta prueba existen de nuevo tres hablantes. Es una prueba parecida a la realizada con “Prueba_1” pero la posición inicial de los hablantes en la sala varía. Dos de los hablantes se encuentran en la parte frontal del sistema de localización y un tercer hablante se encuentra en la parte posterior. En este caso todos los hablantes están de pie.

Al inicio de la prueba se puede apreciar como la cámara dirige la atención hacia los hablantes que se encuentra en su parte frontal. En el momento que el tercer hablante, situado en la parte posterior empieza a hablar, ésta dirige la atención hacia él.

En la segunda parte de la prueba se ha querido mostrar la limitación que existe en el hardware respecto a la rotación del ángulo azimuth. La vídeo cámara empleado puede rotar su posición desde -170° hasta 170° , por lo que, si un hablante se encuentra exactamente en 180° la cámara no podrá alinearse totalmente con su azimuth. Cuando dos hablantes hablan desde ángulos azimuth 160° y -160° aproximadamente, la cámara para localizarlos deberá rotar totalmente su posición.

En la última parte de la prueba, se vuelve a mostrar la precisa localización de hablantes del sistema desde posiciones más alejadas en el entorno utilizado para las pruebas.

El vídeo que muestra la prueba realizada es el siguiente:

https://youtu.be/t6_IYjuavkY

En las diferentes pruebas realizadas se detecta que existe un leve retardo en el sistema, para modificar su ángulo azimuth pero, una vez el sistema dirige la atención hacia un determinado hablante, mientras que éste no termine de hablar el sistema apunta hacia esta dirección.

También se ha podido comprobar que una vez que se lanza la ejecución del programa fusionado, existe un tiempo de ajuste del sistema hasta que el flujo de vídeo se emite sin retardos. Este ajuste de vídeo es solo necesario al comienzo de cada sesión, y una vez realizado no existen retardos adicionales en el procesado de vídeo más allá del tiempo requerido para la detección de caras.

7 CONCLUSIONES

El sistema de detección finalmente implementado funciona para su propósito. De las diferentes pruebas realizadas con él, se ha mostrado que la detección del ángulo azimut es muy precisa en un amplio rango del espacio de la habitación. Además, por medio del procesado visual el sistema es capaz de modificar el ángulo de elevación de la cámara. Este ajuste del ángulo de elevación es necesario, ya que como se ha expuesto previamente, ésta es la principal limitación de la localización auditiva, cuando se emplean este tipo de arrays de micrófonos. En definitiva, ambos sistemas de localización son necesarios para obtener la posición final del hablante.

El sistema es incluso capaz de detectar el ángulo azimut de aquellos hablantes que se encuentran más alejados respecto a la posición de SLAV. En este caso el procesado visual no actuaría pero se ha comprobado como la localización del azimut es correcta.

Asimismo, la localización del ángulo azimut de un hablante está sujeta a un leve retardo. Reducir dicho retardo implementando algoritmos de detección diferentes o con computadores de mayor capacidad de procesado, puede ser una de las líneas de trabajo futuras de este proyecto.

Otra posible línea de trabajo futura sería implementar un módulo maestro de decisión, siendo éste el responsable de determinar los movimientos producidos por el sistema. Asimismo, se podría hacer un estudio entre los diferentes algoritmos de decisión para dicho enfoque.

Una parte importante de este proyecto ha sido el estudio, realizado mediante la simulación, del entorno acústico, con el que se ha comprobado el desempeño del algoritmo SRP-PHAT con diferentes geometrías de micrófonos, determinando finalmente el array de micrófonos que se ha usado en este proyecto.

La aplicación del presente proyecto en una sala de videoconferencia puede ser útil, ya que los diferentes hablantes no tienen que estar sentados, pueden moverse por la habitación y el sistema será capaz de seguirlos.

Por otro lado, este sistema puede ser implementado en un robot para mejorar su interacción con las personas, el robot será capaz de mover su cabeza para dirigir la atención a un hablante.

8 REFERENCIAS BIBLIOGRÁFICAS

- [1] PERRIS, E. E. & CLIFTON, R. K. Reaching in the dark toward sound as a measure of auditory localization in infants. *Infant Behavior and Development*, 1988, 4(11), 473 – 491.
- [2] KAGA M. Development of sound localization. *Acta Paediatr Jpn*, 1992, 34, 134–138.
- [3] BLAUERT J. Spatial hearing: The psychophysics of human sound localization. Cambridge: MA: MIT Press., 1983.
- [4] YOST W. Fundamentals of hearing: an introduction. San Diego: Academic Press., 1994.
- [5] GUEROLA, A. M. Multichannel Audio Processing for Speaker Localization, Separation and Enhancement. thesis. 2013.
- [6] VALIN J.-M., et al. Robust sound source localization using a microphone array on a mobile robot. *Intelligent Robots and Systems (IROS 2003)*, 2003, 2, 1228–1233.
- [7] ETTINGER E. & FREUND Y. Coordinate-free calibration of an acoustically driven camera pointing system. *Second ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 2008)*, 2008.
- [8] CHO Y., et al. Sound Source Localization for Robot Auditory Systems. *IEEE Transactions on Consumer Electronics*, 2009, 3(55).
- [9] AARABI P. The fusion of distributed microphone arrays for sound localization. *EURASIP Journal on Applied Signal Processing*, 2003, 2003, 338–347.
- [10] MAKOUS J. C. & MIDDLEBROOKS J. C. Two-dimensional sound localization by human listeners. *The Journal of the Acoustical Society of America*, 1990, 5(87), 2188–2200.
- [11] SCHAU H. & ROBINSON A. Passive source localization employing intersecting spherical surfaces from time-of-arrival differences. *Acoustics, Speech and Signal Processing*, 1987, 8 (35), 1223–1225.
- [12] MADHU N. & MARTIN R. Advances in Digital Speech Transmission. ch. Acoustic source localization with microphone arrays, 135–166, Wiley, 2008.
- [13] CHEN J., BENESTY J. & HUANG Y. Time delay estimation in room acoustic environments: an overview. *EURASIP Journal on Applied Signal Processing*, 2006, 2006, 1–19.
- [14] DO H., SILVERMAN H. F., & YU Y. A real-time SRP-PHAT source location implementation using stochastic region contraction (SRC) on a large-aperture microphone array. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, 2007.

- [15] DO H. & SILVERMAN H. F. A fast microphone array SRP-PHAT source location implementation using coarse-to-fine region contraction (CFRC). Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA 2007), 2007.
- [16] COBOS M., MARTÍ A. & LÓPEZ J. J. A Modified SRP-PHAT Functional for Robust Real-Time Sound Source Localization With Scalable Spatial Sampling. IEEE signal processing letters, 2011, 1(18).
- [17] TERVO S. Direction estimation based on sound intensity vectors. 17th European Signal Processing Conference (EUSIPCO 2009), 2009.
- [18] <http://noticiasdelaciencia.com/not/839/como-el-cerebro-humano-procesa-realmente-la-informacion-visual/>
- [19] DICARLO J. J., ZOCCOLAN D. & RUST N. C. How Does the Brain Solve Visual Object Recognition?.
- [20] SUNG K.-K & POGGIO T. Example-Based Learning for View-Based Human Face Detection. IEEE Transactions Analysis and machine intelligence, 1998, 1(20).
- [21] ROWLEY H. A., BALUJA S. & KANADE T. Neuronal Network-Based Face Detection. Computer Vision and Pattern Recognition. 1996.
- [22] VIOLA P. & JONES M. J. Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE CVPR, 2001.
- [23] LIENHART R. & MAYDT J. An Extended Set of Haar-like Features for Rapid Object Detection.
- [24] VIOLA P. & JONES M. J. Robust Real-Time Face Detection. International Journal of Computer Vision, 2004, 57(2), 137–154.
- [25] FREUND Y. & SCHAPIRE R.E. A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence, 1999, 14(5), 771-780.
- [26] SCHNEIDERMAN H. & KANADE T. A Statistical Method for 3D Object Detection Applied to Faces and Cars, 2000.
- [27] YANG M.-H., ROTH D. & AHUJA N. A SNoW-Based Face Detector. 2000.
- [28] LI Z. et al. Multiple Active Speaker Localization based on Audio-visual Fusion in two Stages.
- [29] VICIANA-ABAD R., et al. Audio-Visual Perception System for a Humanoid Robotic Head. 2014.
- [30] CECH J. et al. Active-Speaker Detection and Localization with Microphones and Cameras Embedded into a Robotic Head. Humanoids 2013 - IEEE-RAS International Conference on Humanoid Robots, 2013, 203-210.

- [31] KAPRALOS B., JENKIN M. R. M. & MILIOS E. Audio-visual localization of multiple speakers in a video teleconferencing setting.
- [32] <https://ccrma.stanford.edu/software/stk/>
- [33] OPPENHEIM, A.V. & SCHAFER R.W. Tratamiento de señales en tiempo discreto. Madrid: Prentice Hall Iberia, 2000.
- [34] <http://www.fftw.org/>
- [35] COOLEY, James W. and TUKEY, John W. An Algorithm for the Machine Calculation of Complex Fourier Series. Mathematics of Computation. 1965. Vol. 19, no. 90p. 297–301.
- [36] FOWLER M. L. & HU X. Signal models for tdoa/fdoa estimation. IEEE Transactions on Aerospace and Electronic Systems, 2008, 4(44), 1543–1550.
- [37] TELLAKULA A. K. Acoustic source localization using time delay estimation. Ph.D. dissertation, Indian Institute of Science, 2007.
- [38] STOICA P. & LI J. Source localization from range-difference measurements. IEEE Signal Processing Magazine, 2006, 63–69.
- [39] SVAIZER P., MATASSONI M. & OMOLOGO M. Acoustic source location in a three-dimensional space cross-power spectrum phase. IEEE Int. Conf. Acoust., Speech Signal Processing, 1997, ICASSP-97, 231– 234.
- [40] KNAPP C. H. & CARTER G. C. The generalized correlation method for estimation of time delay. Transactions on Acoustics, Speech and Signal Processing, 1976, ASSP-24, 320–327.
- [41] MUNGAMURU B. & AARABI P. Enhanced sound localization, IEEE Trans Syst, Man, Cybernet Part B: Cybernet 2004, 34(3).
- [42] CHA ZHANG F. & ZHENGYOU Z. Why does PHAT work well in low noise, reverberant environments. Las Vegas, NV: ICASSP, 2008 4(2008), 2565–8.
- [43] JOHNSON D. H. & DUDGEON D. E. Array Signal Processing: Concepts and Techniques., N. J. . Englewood Cliffs, Ed. P T R Prentice Hall, 1993.
- [44] DIBIASE J. H. A high accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays. Ph.D. dissertation, Brown University, Providence, RI, 2000.
- [45] TERVO S. & LOKKI T. Interpolation methods for the SRP-PHAT algorithm. International Workshop for Acoustic Echo and Noise Control, Seattle, WA, USA, 2008, 14-17.
- [46] HABETS E.A.P. & SOMMEN P.C.W. Optimal microphone placement for source localization using Time Delay Estimation. Proc. Workshop Circuits, Systems and Signal Processing (ProRISC), Eindhoven, 2002.

- [47] CAMPBELL D.R., PALOMAKI K.J. & BROWN G. J. A MATLAB Simulation of “Shoebox” Room Acoustics for use in Research and Teaching. Computing and Information Systems Journal, 9, 2005.
- [48] SCHIMMEL S.M., MULLER M.F. & DILLIER N. A FAST AND ACCURATE “SHOEBOX” ROOM ACOUSTICS SIMULATOR. Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference, 2009.
- [49] MARQUÉS I. Face Recognition Algorithms. Proyecto fin de carrera Universidad del País Vasco, 2010.
- [50] YANG M.-H., KRIEGMAND. & AHUJA N. Detecting faces in images: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(1), 34–58.
- [51] ZHAO W., et al. Face recognition: A literature survey. ACM Computing Surveys, 2003, 399–458.
- [52] LAM K.-M. & YAN H. Fast algorithm for locating head boundaries. Journal of Electronic Imaging, 1994, 03(04), 351–359.
- [53] LOUBAN R. Image Processing of Edge and Surface Defects Theoretical Basis of Adaptive Algorithms with Numerous Practical Applications, 2009, 123, chapter Edge Detection, 9–29.
- [54] SINGH S. K., et al. A robust skin color based face detection algorithm. Tamkang Journal of Science and Engineering, 2003, 6(4), 227–234.
- [55] NECHYBA M. C., BRANDY L. & SCHNEIDERMAN H. Lecture Notes in Computer Science, 2009, 4625/2009, chapter PittPatt Face Detection and Tracking for the CLEAR 2007 Evaluation, 126–137.
- [56] DIVJAK M. Divjak & BISCHOF H. Real-time video-based eye blink analysis for detection of low blink-rate during computer use. In 1st International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS), 2008, 99–107.
- [57] KAWATO S. & TETSUTAN N. Detection and tracking of eyes for gaze-camera control. In Proc. 15th Int. Conf. on Vision Interface, 2002, 348–353.
- [58] HAN C.-C., et al. Lecture Notes in Computer Science, 1997, 1311, chapter Fast face detection via morphology-based pre-processing, 469–476.
- [59] BAEK K., et al. AI 2005, chapter Multiple Face Tracking Using Kalman Estimator Based Color SSD Algorithm, 2005, 1229-1232.
- [60] <http://opencv.org/>
- [61] http://docs.opencv.org/trunk/d7/d8b/tutorial_py_face_detection.html
- [62] <http://acodigo.blogspot.com.es/2013/06/deteccion-de-rostros.html>
- [63] <https://curl.haxx.se/libcurl/>

- [64] ZOTKIN D., et al. An audio-video front-end for multimedia applications. in Systems, Man, and Cybernetics, 2000 IEEE International Conference on, 2000, 2(2000), 786–791.
- [65] RÄTY T. Survey on contemporary remote surveillance systems for public safety. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 2010, 5(40), 493–515.
- [66] NAKANO A. Y., NAKAGAWA S. & YAMAMOTO K. Automatic estimation of position and orientation of an acoustic source by a microphone array network. J. Acoust. Soc., 2009, 126, 3084–3094.
- [67] PÉREZ-LORENZO J. M., et al. Evaluation of general cross-correlation methods for direction of arrival estimation using two microphones in real environments. Applied Acoustics, 2012, 73(8), 698 – 712.