Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería en Tecnologías de la Telecomunicación

Trabajo Fin de Grado

Diseño, implementación y evaluación de un demostrador de localización de fuentes de audio en tiempo real

Autor: María Paz González

Tutor: Javier Macías Guarasa

2015

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería en Tecnologías de la Telecomunicación

Trabajo Fin de Grado

Diseño, implementación y evaluación de un demostrador de localización de fuentes de audio en tiempo real

Autor: María Paz González

Director: Javier Macías Guarasa

Tribunal:

Presidente: Marta Marrón Romera

Vocal 1º: Juan Carlos García García

Vocal 2º: Javier Macías Guarasa

Calificación: .....

Fecha: .....

A mi familia, amigos y todos los que lo hicieron posible.

"Sin continuo crecimiento y progreso, tales palabras como mejora, logro y éxito no tienen significado."
Benjamin Franklin

Agradecimientos

Este trabajo es el fruto del esfuerzo de varias personas, gracias a las cuales he conseguido llegar hasta aquí. En primer lugar, mi familia y en concreto mis padres. Ellos fueron la primera generación de mi familia que tuvo la oportunidad de acceder a estudios universitarios y gracias a su esfuerzo y dedicación consiguieron que su hija corriera la misma suerte. Porque tras haberme dicho miles de veces: ¡Estudia María! (ese profesor de primaria que llevan dentro no les abandonará nunca, que le vamos a hacer...), espero que se sientan orgullosos de verme acabar la carrera. Y como no, mi hermano pequeño, que seguirá mis pasos por estos pasillos este nuevo curso y al que espero haber servido y servir de inspiración y motivación, aunque el resto lo tenga que poner de su parte.

Agradecer también a mis amigos y compañeros, tanto a los que dejé lejos cuando decidí iniciar esta nueva etapa, a los de prácticamente toda la vida, como a los que me apoyaron y vivieron conmigo esa angustia del primer examen o esa mezcla de impaciencia e incertidumbre que se siente en el último. Mención especial merece Marcos, por haberme aguantado estos años juntos en cada una de las asignaturas, y por si fuera poco también haciendo el TFG ambos con el grupo GEINTRA. Por su ayuda incondicional y por los momentos, casi siempre acompañados de una buena cerveza, cuando necesitábamos desconectar. Y Marina, que a pesar de que hace poco mas de un año ni siquiera nos conocíamos, se ha convertido en uno de mis grandes apoyos en cualquier ámbito.

Mencionar también la gran aportación de los autores de los trabajos previos que han contribuido al desarrollo de este proyecto y todos aquellos que han participado en él en menor o mayor medida. A Javier Macías Guarrasa, por ser un excelente tutor para mí y para este proyecto. Por su paciencia y ganas de enseñar y sobre todo por compartir conmigo una parte del gran conocimiento que puebla su cabeza. Agradecer también a toda la comunidad de investigadores, por mantener esas ganas e ilusión por ir siempre un paso mas allá, a pesar de ofrecerse cada vez menos oportunidades a este sector tan importante para el desarrollo personal, social y humano.

Resumen

Este proyecto describe el diseño, desarrollo, implementación y evaluación de un demostrador en tiempo real de un sistema de localización y seguimiento de hablantes mediante agrupaciones de micrófonos en un espacio inteligente.

Con este objetivo se han combinado los resultados de trabajos previos de otros Proyectos Fin de Carrera y Fin de Master en los que se desarrollaron librerías de adquisición y reproducción de audio multicanal, algorítmica de detección de actividad de voz en entornos con agrupaciones de micrófonos, módulos de visualización de entornos virtuales y algorítmica de localización de locutores, integrando su funcionalidad y adaptándola a las exigencias del funcionamiento en tiempo real.

Palabras clave: Localización acústica, detección de actividad de voz, demostrador en tiempo real, procesamiento basado en agrupaciones de micrófonos, espacio inteligente.

Abstract

This project describes the design, development, implementation and evaluation of a speakers location and tracking system real time demonstrator, using microphone arrays in a smart space.

To this end it has been combined the results of previous Thesis which developed multichannel acquisition and reproduction libraries, activity voice detection algorithms in environments with microphone clusters, virtual environment visualization modules and speaker location algorithms, integrating their functionalities and adapting them to real time operating requirements.

Keywords: Acoustic localization, voice activity detection, real time demonstration, microphone array processing, smart space .

Resumen extendido

Este proyecto tiene como objetivo el estudio, implementación y evaluación de un sistema demostrador en tiempo real basado en un sistema de localización y seguimiento de hablantes mediante agrupaciones de micrófonos en un espacio inteligente. Para ello se han incluido e integrado las funcionalidades de diferentes proyectos como son "Diseño, implementación de un sistema de adquisición, procesamiento y generación de audio multicanal en aplicaciones para espacios inteligentes" realizado por Jesús Pablo Martinez González [1], "Estudio, implementación y evaluación de un sistema de detección de actividad de voz en espacios inteligentes" desarrollado por Rubén Peral Nuño [2], "Speaker Localization Techniques in Reverberant Acoustic Environments" realizado por Carlos Castro [3] y "Diseño, implementación y evaluación de una interfaz de control multimodal en un espacio inteligente: control gestual" llevado a cabo por David Casillas [4].

Para implementar este demostrados de localización de fuentes de audio en tiempo real se ha utilizado el lenguaje de programación C junto con el entorno de desarrollo Emacs [5]. También se ha hecho uso de programación Shell para la realización de algunos Scripts que nos han facilitado las tareas de experimentación y de generación múltiple de gráficas. Para la generación de gráficas se ha utilizado la herramienta Gnuplot [6]. Aparte de las librerías típicas usadas en programación en C, para este diseño hemos tenido que aprender el uso de dos de ellas:

- La librería `sndfile` con la cual se ha realizado la escritura en ficheros de audio para posteriores experimentos a partir de las grabaciones de las señales de voz.
- La librería `fftw` [7] con la cual realizar transformadas de Fourier para poder hallar el valor de correlación entre dos señales.
- La librería `rtAudio` la cual nos permite llevar a cabo tanto la adquisición como la reproducción de las señales de audio en tiempo real.

Partiendo de los proyectos comentados anteriormente y realizando una adaptación e integración entre ellos y con los requerimientos y restricciones que presenta una ejecución en tiempo real, el sistema final se compondrá de los diferentes módulos cuya funcionalidad y aportación al demostrador se define a continuación:

- **Módulo de adquisición de audio multicanal**

  Este módulo se implementa sobre un conjunto de librerías de soporte para la captura, procesamiento y generación de audio multicanal en tiempo real y se encargará de todo el proceso de adquisición del audio de los diferentes arrays de micrófonos que se incluyan en el sistema, así como de la preparación de los datos y del sincronismo de dichos procesos con los de tratamiento y procesamiento de los datos.

Se hace esencial el uso de arrays de micrófonos como un método para mejorar la calidad del habla capturada y eliminar en la medida de lo posible los errores y ruidos que pueda introducir el entorno, por lo que se ha explotado la disponibilidad de agrupaciones de micrófonos en el espacio inteligente para tal fin.

- **Módulo de cálculo de la correlación de las señales de audio**

  Este módulo está destinado a realizar los cálculos de correlación Cross-power Spectrum Phase (CSP) basados en la diferencia temporal Time Difference Of Arrival (TDOA) entre las señales obtenidas de los diferentes pares de micrófonos y almacena los datos obtenidos para que tanto el Voice Activity Detector (VAD) como el algoritmo Steered Response Power (SRP) puedan disponer posteriormente de ellos.

- **Módulo de detección de actividad de voz**

  Este módulo consta a su vez de los bloques de cálculo del umbral de detección y autómata de decisión, que contribuyen a la funcionalidad de un autómata de detección de la existencia de voz en el entorno en tiempo real basándose en la comparación de los resultados de los cálculos de correlación con un margen fijado para determinar en que estado se encuentra el sistema en cada momento y descartar o no la trama de audio en base a ducha decisión.

- **Módulo de estimación de la posición de la fuente sonora**

  Este módulo se encarga de la localización del locutor, se va a implementar sobre un algoritmo de localización robusto basado en el Método de la respuesta en potencia dirigida (SRP) que evalúa la actividad acústica en localizaciones específicas, orientando el patrón de directividad del array ($beamforming$). A su vez, este esquema se basa en una técnica de localización más básica, capaz de estimar la dirección de llegada (Direction of Arrival (DOA)) del habla haciendo uso de los cálculos de correlación entre pares de micrófonos, aprovechando las diferencias de tiempos de llegada de la voz a distintos micrófonos de un array para completar el cálculo de la ubicación de la fuente.

- **Módulo de visualización 3D**

  Por último, este módulo consta de un espacio inteligente virtual en 3 Dimensiones (3D) que permite la visualización tanto del entorno como de ciertos objetos que que permiten una interacción hombre-máquina a modo de actuadores del espacio inteligente. Gracias a esta sencilla e intuitiva representación se puede tener una vista funcional del espacio inteligente que permite, tanto hacer cambios de vistas y proyecciones como observar cambios en los actuadores del entorno virtual o representar la posición del usuario de dicho entorno.

Para llevar a cabo los experimentos que evaluarán el comportamiento del sistema completo se ha hecho uso de la librería `sndfile` para comprobar los resultados obtenidos con los que ofrecen cada una de las implementaciones offline de cada módulo.

Índice general

Índice de figuras

Índice de tablas

Índice de listados de código fuente

Lista de acrónimos

3D	3 Dimensiones.
ADC	Analog to Digital Converter.
API	Application Programming Interface.
CC	Cross Correlation.
CSP	Cross-power Spectrum Phase.
DOA	Direction of Arrival.
FFT	Fast Fourioer Transform.
GCC	Generalized Cross Correlation.
GEINTRA	Grupo de Ingeniería Electrónica aplicada a Espacios Inteligentes y Transporte.
ML	Maximum Likelihood.
PHAT	Phase Transform.
SNR	Signal to Noise Ratio.
SRP	Steered Response Power.
SRP-PHAT	Steered Response Power with Phase Amplitude Transform.
SSL	Sound Source Localization.
TDOA	Time Difference Of Arrival.
VAD	Voice Activity Detector.

Capítulo 1

Introducción

Los sistemas de reconocimiento, la biometría y el análisis automático de espacios inteligentes mediante el procesamiento de múltiples sensores forman parte de un sector que ha cobrado cada vez mas importancia, siendo un área con cada vez mayor actividad científica y tecnológica que se esta desarrollando en un mercado en vías de expansión. En ese contexto, las tareas de detección, localización y seguimiento de personas son fundamentales para mejorar los procesos de interacción con el entorno, o con otras personas u objetos dentro del mismo.

Este proyecto se centra en la implementación de un demostrador en tiempo real en el contexto de los sistemas de localización y seguimiento de hablantes en un espacio inteligente, uno de los trabajos orientados a la generación de tecnología basada en el procesamiento de las señales captadas por agrupaciones de micrófonos que propone el *Grupo de Ingeniería Electrónica aplicada a Espacios Inteligentes y Transporte (GEINTRA)* del Departamento de Electrónica siguiendo la línea de investigación y desarrollo enfocada a la interacción con los diferentes sensores que se encuentran en el espacio inteligente del cual dispone la Universidad de Alcalá de Henares.

En este Proyecto Fin de Carrera se parte de trabajos previos de otros Proyectos Fin de Carrera y Tesis de Máster, comenzando por el trabajo de Jesús Martínez [1] que desarrolla librerías de adquisición y reproducción de audio multicanal haciendo uso de las agrupaciones de micrófonos disponibles en el espacio inteligente del grupo para reducir los problemas de reverberación, ruido aditivo y baja relación señal-ruido. También un sistema de localización de locutores y técnicas de seguimiento en ambientes acústicos reverberantes implementado por Carlos Castro González [3] y un sistema que determina la presencia o ausencia de una señal de habla desarrollado por Rubén Peral [2]. Por último para la visualización del punto estimado por el demostrador se utiliza el módulo de visualización de entornos virtuales en 3D que desarrolló David Casillas [4].

El objetivo de este trabajo es integrar la funcionalidad de cada uno de estos sistemas y adaptarla a las exigencias de un funcionamiento en tiempo real.

1.1 Motivación y objetivos

La principal motivación para el desarrollo de proyectos que siguen la línea de una tecnología que permite la interacción del humano con diferentes sensores dentro de un espacio inteligente, es el auge que ha experimentado este sector en los últimos años.

Esta línea es la que sigue el grupo GEINTRA y este proyecto en concreto, abordando además el propósito de conseguir esta interacción humano-máquina en tiempo real.

También podemos encontrar una serie de motivaciones y objetivos personales que promueven la realización de este trabajo, como puede ser aprender el uso de diferentes herramientas y recursos, afrontar los problemas que podemos encontrar durante la realización del proyecto o adquirir habilidades relacionadas con la investigación y la búsqueda de soluciones.

El objetivo fundamental de este proyecto es el diseño, implementación y evaluación de un demostrador de un sistema de localización de locutores funcionando en tiempo real. Los objetivos específicos de este proyecto son los siguientes:

- Diseñar la arquitectura del demostrador, que integrará los sistemas de adquisición multicanal, detección de actividad de voz, localización de locutores y visualización.
- Adaptar el sistema de detección de actividad de voz para su funcionamiento en tiempo real, integrado con el de adquisición multicanal.
- Adaptar el sistema de localización de locutores para su funcionamiento en tiempo real, integrado con el de detección de actividad de voz.
- Adaptar el sistema de visualización para su funcionamiento en tiempo (próximo al) real, integrado con los anteriores.
- Utilizar una librería de adquisición de audio multicanal como soporte al procesamiento en tiempo real de señales captadas por agrupaciones de múltiples micrófonos.
- Evaluar las prestaciones y capacidades de procesamiento en tiempo real, y las limitaciones en función de la configuración del sistema.

Los requisitos que debe cumplir el trabajo a desarrollar son:

- Ser flexible en el sentido de permitir modificar con facilidad los parámetros de control de los procesos de adquisición, detección de actividad, localización acústica y visualización.
- Ser flexible en el sentido de permitir la cómoda incorporación y control de nuevos dispositivos de captura y reproducción de audio, así como la incorporación de nueva algorítmica en el mismo entorno de demostración.
- Estar bien documentado para facilitar su utilización en futuros proyectos.
- Disponer de un software eficiente y robusto.

1.2 Organización de la memoria

Esta memoria se ha estructurado en cinco capítulos y un anexo, que se describen a continuación:

1. Introducción: En el capítulo 1 se realiza una presentación general del trabajo realizado, incluyendo el punto de partida. También se incluyen los objetivos iniciales y las motivaciones que han impulsado la realización de este trabajo.
2. Diseño del sistema: El capítulo 2 es el que va a describir el funcionamiento teórico y el diseño tanto de cada uno de los bloques que componen el demostrador como del sistema en tiempo real completo, así como todos los conceptos a tener en cuenta. También se estudiarán los posibles problemas que se pueden encontrar a la hora de realizar una implementación en tiempo real.

3. Desarrollo: En el capítulo 3 se explican los pasos que se han seguido para implementar el sistema demostrador, desde los cambios realizados a cada bloque para conseguir una ejecución en tiempo real, hasta la forma en la que deben integrarse para alcanzar el objetivo final.

4. Resultados y evaluación: En el capítulo 4 se comentan los resultados obtenidos tras la implementación, depuración y ejecución del sistema.

5. Conclusiones y líneas futuras: El capítulo 5 muestra las conclusiones a las que se ha podido llegar tras el desarrollo y la implementación de nuestro sistema demostrador en tiempo real y se propone una línea de posibles trabajos futuros.

6. Apéndices:

   (a) Herramientas y recursos: En el apéndice A se enumeran las herramientas necesarias para la elaboración del proyecto.

Capítulo 2

Diseño del Sistema

2.1 Introducción

En esta sección de estudio, previa al desarrollo del proyecto, se conocerá la arquitectura general del sistema y se explicará brevemente la funcionalidad de cada uno de los bloques que van a formar parte del demostrador en tiempo real completo. Algunos de los bloques que lo forman trabajaban originalmente *offline*, sobre archivos de audio previamente grabados, de modo que estudiaremos los cambios y adaptaciones que deberán producirse para conseguir una ejecución en tiempo real.

Tras haber visto el funcionamiento de cada uno de los integrantes del sistema, se describirá el modo en que se deben ensamblar para llegar a construir la estructura completa que dará forma a nuestro demostrador en tiempo real y los aspectos a tener en cuenta en la implementación. Esta implementación supondrá unos compromisos de sincronismo entre procesos y nos enfrentaremos a la problemática de adaptarlos adecuadamente a una ejecución en tiempo real.

2.2 Sistema en tiempo real

El demostrador que se desea implementar con la realización de este proyecto pretende ser un sistema capaz de determinar la localización en tiempo real de un sujeto hablante situado dentro de un entorno inteligente que dispone de numerosos sensores y transductores, que obtienen información sobre los diferentes eventos que se puedan producir en este espacio (una o varias personas en movimiento, la presencia o no de un sujeto, un robot haciendo un determinado recorrido, un locutor actuando como fuente sonora, etc).

La adquisición de datos es una parte esencial para el desarrollo de este sistema, ya que la calidad de la señal captada jugará un papel crucial en los procesos realizados posteriormente y en los resultados finales. En la implementación de sistemas que se basan en el procesado de señales de audio, es necesario el uso de transductores acústico-eléctricos, ya que el sonido se propaga en forma de ondas y para poder extraer sus características es necesario transformar la presión sonora en un señal eléctrica y así poder proceder a su digitalización.

Todos estos aspectos correspondientes al proceso de adquisición de audio, se abordan en las librerías de adquisición y reproducción de audio multicanal implementadas sobre el soporte hardware que son las agrupaciones de micrófonos, previos y tarjetas *Analog to Digital Converter (ADC)* disponibles en el espacio inteligente en el que se desarrolla el trabajo, en las cuales se profundizará un poco más en apartados posteriores.

Cualquier aplicación que involucre una señal de voz conlleva un procesado de audio, posterior a la adquisición y digitalización del mismo, que permite extraer de éste un conjunto de características propias de la señal que luego serán útiles para los cálculos posteriores que se deben realizar en las siguientes etapas del sistema para conseguir obtener finalmente el objetivo deseado.

Una vez obtenidos los datos de la señal digitalizada, la primera etapa de procesado tiene como objetivo principal diferenciar lo que es ruido de lo que es voz en una señal de audio dada. Se trata de un detector de actividad de voz, del inglés Voice Activity Detector (VAD), que basa su algoritmo en la medida de la correlación, Cross-power Spectrum Phase (CSP), obtenida a partir de las características extraídas de la señal de audio adquirida por un par de micrófonos y extendida para integrar la información procedente de múltiples pares de micrófonos que serán los que forman las agrupaciones, que se evaluará usando un autómata de decisión para determinar la existencia o no de señal de voz.

La siguiente función que realiza el sistema es la de localización del hablante dentro del entorno. Esta funcionalidad la aporta un algoritmo de localización robusto basado en el Método de la respuesta en potencia dirigida, Steered Response Power (SRP), que evalúa la actividad acústica en localizaciones específicas, orientando el patrón de directividad del array (*beamforming*). A su vez, este esquema se basa en una técnica de localización más básica, capaz de estimar la dirección de llegada, Direction of Arrival (DOA), del habla mediante el cálculo de la correlación cruzada generalizada, *Generalized Cross Correlation* (*GCC*), entre pares de micrófonos, aprovechando las diferencias de tiempos de llegada de la voz a distintos micrófonos de un array para completar el calculo de la ubicación de la fuente.

La etapa de detección de actividad de voz, supone una gran optimización del sistema en cuanto a su ejecución en tiempo real se refiere. Según el resultado que obtengamos de este bloque, podemos evitar ejecuciones del algoritmo de localización que vayan a trabajar sobre audio que no corresponda a una señal de voz propiamente dicha. Si no dispusiéramos de un detector que descarte tramas de ruido ambiente, obtendríamos como resultado final la estimación de puntos de localización que realmente solo corresponden a los puntos de máxima potencia de señal dentro del espacio en el que se trabaja, es decir, los puntos en los que se genere un mayor ruido.

Llegados a este punto, ya se habrá obtenido la estimación del punto que se corresponde con la localización espacial del locutor, de modo que el último paso a seguir es el de visualización de dicha estimación. Para ello se dispone de un espacio inteligente virtual en 3D que permite la visualización de ciertos objetos que interactuarán con el usuario a modo de actuadores del espacio inteligente. Este espacio sirve como mecanismo de visualización de dichos actuadores. Como modelo del espacio, se ha utilizado la sala física del Edificio Politécnico de la Universidad de Alcalá en la que se desarrolla este trabajo conocida como $geintra - ispace$. Gracias a esta sencilla e intuitiva representación se puede tener una vista funcional del espacio inteligente que permite, tanto hacer cambios de vistas y proyecciones como observar cambios en los actuadores del entorno virtual o representar la posición del usuario de dicho entorno.

Una vez visto el funcionamiento general del sistema demostrador en tiempo real, se profundizará un poco más en la funcionalidad y aportación de cada uno de los bloques que lo forman.

2.2.1 Arquitectura del sistema

Tras haberse explicado la funcionalidad del demostrador en tiempo real completo y del papel que juega cada uno de los módulos en su implementación, se puede elaborar un esquema de bloques que represente la estructura o arquitectura de este sistema con el objetivo de exponer las dependencias y relaciones entre ellos. Este esquema puede observarse en la figura 2.1.

La etapa de adquisición de la señal de audio es la que debe actuar en primer lugar, ya que es la

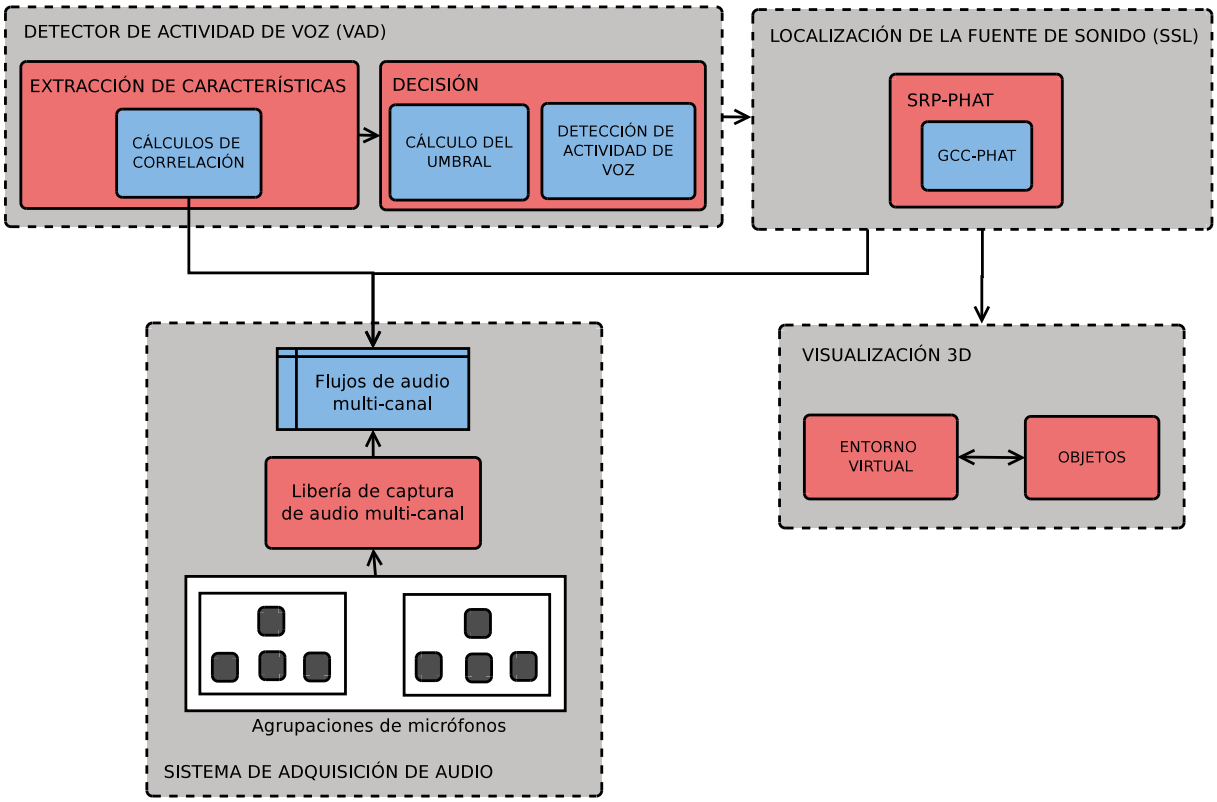


Figura 2.1: Arquitectura del demostrador de localización en tiempo real.

que permite obtener los datos digitales que se van a procesar posteriormente a partir de una magnitud física que en este caso es la señal de audio. Gracias a una serie de transductores acústico-eléctricos que convierten la energía acústica (vibraciones sonoras: oscilaciones en la presión del aire) en energía eléctrica (variaciones de voltaje) y a un conversor analógico-digital, ADC , que la digitaliza y normaliza, que harán posible la transformación de la señal de audio en una serie de muestras que podrán ser procesadas posteriormente.

Tanto el módulo de detección de señal de voz VAD , como el encargado de determinar la posición del locutor (*Sound Source Localization (SSL)* basado en cálculos de respuesta en potencia dirigida con transformación de fase, *Steered Response Power with Phase Amplitude Transform (SRP-PHAT)*) se implementan sobre las librerías de adquisición, ya que para realizar los cálculos pertinentes en cada uno de ellos se precisan los datos correspondientes a las muestras obtenidas de la señal de audio. El bloque de adquisición de audio multi-canal sirve de base y soporte para la implementación de estas otras etapas del sistema.

La etapa encargada de determinar si existe actividad de voz, parte de un algoritmo que se implementa sobre la información que aportan un conjunto de ficheros de audio. De modo que se deberá adaptar su funcionamiento para que pueda tomar la decisión de descartar o no la trama de audio partiendo de la información disponible en cada momento y obtener así una respuesta en tiempo real.

También aparecerá en este bloque una etapa de cálculo del umbral de decisión en tiempo real. Al trabajar con ficheros, se disponía de toda la señal de audio a procesar y un cálculo fiable del umbral de decisión se podía realizar recorriendo todo el archivo y fijando un determinado margen. Sin embargo, en una ejecución en tiempo real no se va a disponer de la señal de audio al completo, sino que se van obteniendo tramas según la velocidad de muestreo y digitalización del sistema. De esta forma se deberá realizar un cálculo del umbral en tiempo real y que permita también una adaptación a posibles cambios

de potencia de la señal o ruidos en el entorno. Este aspecto se explica con más detalle en el apartado 2.2.3, a partir de la página 10.

La etapa de localización de la fuente sonora SSL basada en el algoritmo SRP-PHAT tiene una dependencia directa del bloque de detección de actividad de voz, ya que si éste determina que la señal no corresponde a audio procedente del locutor, si no que contiene ruido generado por el entorno, no debería producirse el intento de estimación de la posición.

Por último, el módulo de visualización 3D tiene a su vez una dependencia directa con el bloque de cálculo SRP y, por lo tanto, también con el bloque VAD . Esto se debe a que la representación del punto estimado, únicamente se producirá cuando se obtenga un resultado que es fruto del cálculo sobre una trama que realmente corresponde a señal de voz.

De este modo, cuando el sistema se encuentre con tramas que solamente contienen ruido ambiente, éstas podrán ser descartadas antes de someterse a los cálculos pertinentes para la localización del individuo. Este hecho supone un ahorro en cuanto a tiempo de procesado de datos, ya que solo continuará avanzando a lo largo del sistema una trama que corresponda a muestras de señal de voz.

2.2.2 Módulo de adquisición multi-canal

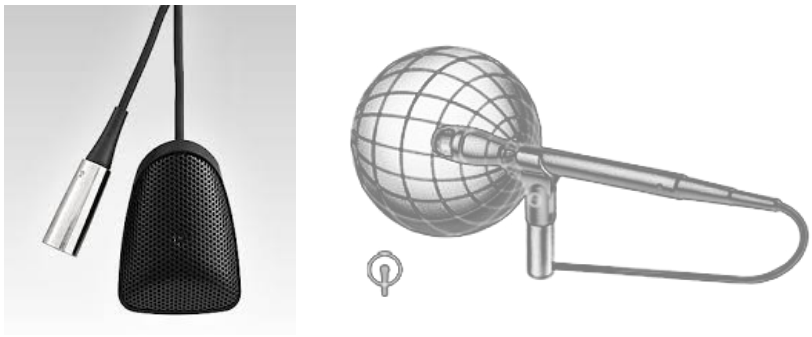
La etapa principal y la base sobre la que se sustenta todo el sistema es la de adquisición de datos. De ella dependerá que la calidad de la señal de audio captada sea la suficiente para obtener un demostrador fiable y eficiente y que proporcione unos resultados finales que se ajusten a la realidad.

Este módulo se implementó con el objetivo de conseguir sistemas que den soporte a las implementaciones prácticas en el área del procesamiento de audio basado en múltiples micrófonos, enfocado a facilitar el desarrollo de sistemas de demostración en tiempo real. Con ese fin, se desarrolló un conjunto de librerías de soporte para la captura, procesamiento y generación de audio multicanal en tiempo real.

Las aplicaciones en las que la captura de la señal de habla se hace usando micrófonos alejados del locutor (habitualmente para distancias superiores a un metro) muestran una fuerte sensibilidad a los problemas de reverberación, ruido aditivo y baja relación señal a ruido. Por este motivo, se hace esencial el uso de arrays de micrófonos como un método para mejorar la calidad del habla capturada y eliminar en la medida de lo posible los errores y ruidos que pueda introducir el entorno. El tipo de micrófono, la frecuencia de muestreo y la cuantización realizada en la captación del audio deberá adecuarse al ancho de banda de la voz y sus características. Hay factores externos al locutor como la elección de los parámetros anteriores, *Signal to Noise Ratio (SNR)* de las muestras adquiridas o la utilización de micrófonos con diferentes curvas de respuesta frecuencial que pueden influir negativamente en el resultado.

Para el desarrollo de este sistema y para la realización de este proyecto se han utilizado micrófonos ominidireccionales de condensador electret Shure series MX391/O cuyo formato se muestra en la figura 2.2, que proporcionan una alta calidad de grabación siendo su tamaño muy pequeño, lo que permite modelar la agrupación de micrófonos con mayor facilidad. El sistema empleado para adquirir la señal eléctrica que proporcionan los micrófonos se basa en un preamplificador RME Octamic D y una tarjeta de adquisición multicanal PCI RME Hammerfall DSP 9652. La etapa de previos Octamic D dispone de ocho canales con una resolución de hasta 192kHz a 24bit con previos de micro o línea de alta calidad. Es un módulo preparado para sistemas de grabación, especialmente pensado para las series de interfaces Fireface y Hammerfall DSP.

Esta tarjeta de adquisición posee un ADC encargado de digitalizar la señal de audio proveniente de los micrófonos pudiendo procesar diferentes fuentes simultáneamente repartiéndolas en canales. Los micrófonos poseen un rango de frecuencias comprendido entre los 50 a los 17000 Hz, por lo que según



(a) Micrófono Shure series MX391/O [8]

(b) Dirección de captación de un micrófono omnidireccional. [9]

Figura 2.2: Modelo de los micrófonos Shure MX391/0 disponibles en el *ispace*.

el teorema de Nyquist deberemos utilizar una frecuencia de muestreo igual o superior a 40000 Hz para recuperar la señal analógica. Las frecuencias de muestreo que soporta la tarjeta de adquisición que se va a utilizar son:

- 32000 Hz
- 44100 Hz
- 48000 Hz
- 88200 Hz
- 96000 Hz

Por lo que la frecuencia de muestreo mínima compatible con esta aplicación sería de 44100 Hz.

Otro elemento que ocupa un lugar importante en los sistemas en tiempo real son las estrategias de gestión de memoria y de sincronismo entre procesos. Este bloque debe tratar con grandes cantidades de datos, y la celeridad con la que se generan estos datos y la premura con la que tienen que ser procesados condiciona que la memoria en la que se almacenen sea rápida.

En este proyecto se ha utilizado un procesador Intel(R) Xeon(R) CPU E5345 @ 2.33GHz con un sistema operativo basado en un kernel de 64 bits. También se debe tener en cuenta, que limitando el acceso a zonas de memoria mas lentas se evitarán grandes latencias en el programa. Se intentan evitar las operaciones de lectura y escritura, accediendo a disco duro, en el programa ya que disminuyen la velocidad del procesado de datos, de forma que los accesos mas rápidos serán aquellos que se realicen en memoria caché y los siguientes los que accedan a memoria RAM. Ha de considerarse que si el tamaño de datos es muy grande, existirá gran cantidad de fallos de página y de caché, lo que retardará la ejecución del manejo de datos. Si se desea profundizar más en este ámbito, se puede encontrar más información en el proyecto de Jesús Martinez [1].

El sincronismo entre procesos permite el diseño de sistemas concurrentes en el que el peso del procesado se paralelice, posibilidad que ahorra tiempo de ejecución.

De esta forma, estas librerías permiten realizar varias funciones, de las cuales solo interesarán en este caso las que hacen referencia a procesos en tiempo real. Estas funciones son las dedicadas a la adquisición en tiempo real, proporcionando al usuario el control de los hilos que las conforman, el tratamiento de los datos presentes en memoria y el sincronismo de los procesos concurrentes presentes en el programa.

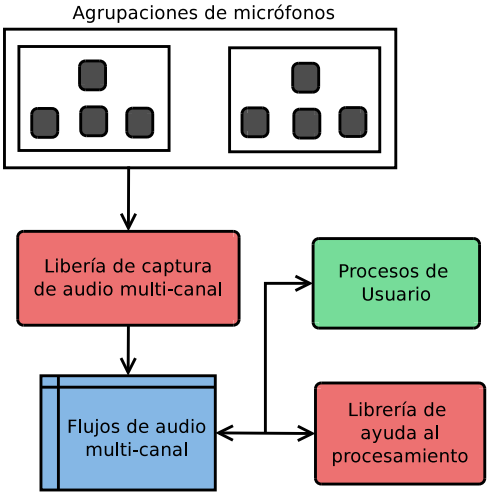


Figura 2.3: Estructura del sistema de adquisición de audio multi-canal.

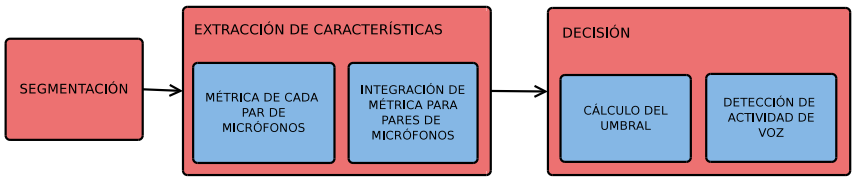


Figura 2.4: Estructura del sistema VAD .

Para el desarrollo de este proyecto se usará del proyecto original únicamente la parte correspondiente a adquisición de audio con procesado de datos en memoria, que sigue la estructura que se muestra en la figura 2.3, a pesar de aportar también las opciones de reproducción y generación de audio y el procesado sobre ficheros de audio, sobre las que se puede obtener mas información en el Proyecto de Fin de Carrera de Jesús Martinez [1].

2.2.3 Módulo de detección de actividad de voz

Una práctica muy común en aplicaciones de procesado de voz es la determinación de periodos de voz o de silencio en una señal dada. Esta tarea no es tan fácil de solucionar como podría parecer en un principio ya que la presencia de ruido en el entorno hace que se cometan numerosos errores en la detección.

La estructura de este VAD , que se muestra en la figura 2.4, se compone de tres grandes módulos que se encargaran de tratar los datos obtenidos del bloque de adquisición de audio multicanal y prepararlos para su procesado, extraer los cálculos necesarios a partir de estos datos y tomar una decisión basándose en la comparación de los resultados de estos cálculos con un margen fijado. Para la implementación del sistema de detección de actividad de voz, se ha partido del Trabajo de Fin de Carrera original de Rubén Peral [2], en el que se puede encontrar información mas detallada.

Segmentación

La etapa de segmentación consiste en procesar la señal de entrada mediante segmentos o ventanas como se muestra en la figura 2.5, en la nomenclatura (frames o tramas) normalmente de una duración de entre 20-40ms. Se elige este tamaño de ventana en base a las propiedades de la señal de voz, para conseguir un adecuado compromiso entre la resolución temporal y espectral del análisis. Las hipótesis de partida más usuales son:

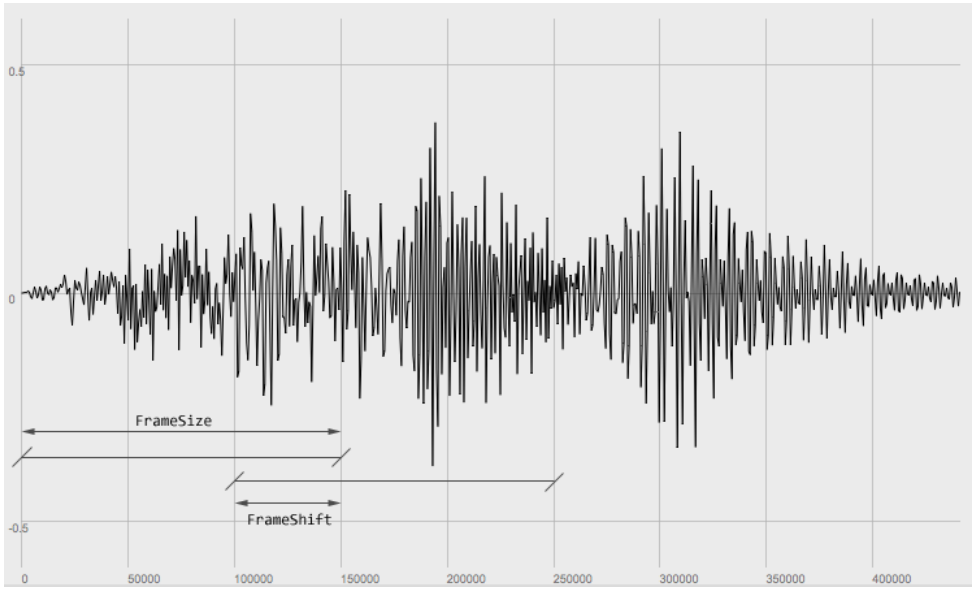


Figura 2.5: Ejemplo de segmentación de una señal sonora

- El ruido ambiente es aditivo a la señal de voz, es decir, la energía en los periodos de voz será la suma de la energía de la señal de ruido más la energía de la señal de voz limpia.
- El segmento de la señal de voz tiene un valor de energía mayor que el segmento de ruido ambiente.
- La voz es estacionaria en periodos cortos de tiempo, por ejemplo T <40 ms.
- El ruido es también estacionario para periodos mucho más largos, por ejemplo T >2seg.
- La voz tiene más componentes periódicas que el ruido.
- El espectro de la voz está más ”organizado” que el del ruido.

Extracción de características

La etapa de extracción de características se encarga de obtener información de la señal segmentada, en este caso, la relacionada con el dominio temporal. Mediante las agrupaciones de micrófonos se puede extraer la diferencia en el tiempo de llegada a cada par de micrófonos. A partir de las dos señales obtenidas de un par de micrófonos obtendremos una señal retrasada en el tiempo y atenuada con respecto a la otra, esta es la idea principal de los algoritmos basados en TDOA . Posteriormente se integran los cálculos obtenidos para cada par de micrófonos en un solo valor global.

- **Módulo de estimación de la métrica para cada par de micrófonos**

  El método mas común para determinar la diferencia en el tiempo de llegada, haciendo uso de las señales procedentes de un par de micrófonos, requiere el cómputo de de la función de correlación cruzada *Cross Correlation (CC)* . Se va a implementar la técnica basada en el análisis del Crosspower-Spectrum Phase, también llamado Generalized Cross-Correlation.

  La correlación cruzada es una buena técnica para la estimación del retardo temporal cuando las señales solo se ven afectadas por fuente de ruido incorrelado. Sin embargo, no es tan eficiente cuando se encuentra en presencia de una fuerte reverberación, ya que la señal mostrará una alta correlación con sus réplicas. Por ello, se ha partido de la *Generalized Cross Correlation (GCC)* [10], que consiste en aplicar un prefiltrado a las señales antes de calcular su correlación con el objetivo de mejorar los

resultados que ofrece la correlación cruzada común. La transformada de Fourier de la función de CC es lo que se conoce como Cross-Power Spectrum CSP .

Esta estimación en el dominio frecuencial, dependerá de una serie de factores incluyendo los niveles de ruido y reverberación, la distancia de separación entre micrófonos y la elección de la función de ponderación. Para conseguir estimaciones mas robustas, las correlaciones deben ser filtradas, para lo que se usa una función de ponderación. En este caso se usará el sistema de ponderación *Phase Transform (PHAT)* que se utiliza para obtener una amplitud unitaria para todas las componentes de frecuencia, preservando al mismo tiempo las fases que contienen la información sobre el retardo temporal. De esta forma resulta el algoritmo GCC -PHAT .

Teniendo en cuenta este algoritmo como base de los cálculos de correlación, se va a usar la fórmula 2.1 que fue desarrollada en el trabajo de Maurizio Omologo [11]. Comienza calculando el espectro de las señales aplicando la Transformada de Fourier (*Fast Fourioer Transform (FFT)*) a cada ventana, este espectro es lo que se utiliza para calcular la diferencia de fase entre las dos señales multiplicando el espectro de una por el conjugado de la otra. Finalmente se divide por el producto de los módulos y así solo queda la información sobre la diferencia de fases.

$$\phi(t,f) = \frac{S_1(t,f) \cdot S_2^*(t,f)}{|S_1(t,f)| \cdot |S_2(t,f)|} \tag{2.1}$$

Finalmente solo queda hacer la Transformada inversa para obtener la función de correlación.

$$C(t,\tau) = \int_{-\infty}^{+\infty} \phi(t,f) e^{j2\pi f \tau} df \tag{2.2}$$

La correlación es una medida de la similitud entre las dos señales tanto en morfología como en ubicación temporal. La función correlación cruzada representa la evolución de esta similitud según varía τ, y será máxima cuando dos señales sean similares en forma, y estén en fase.

De esta forma, cuando se estén analizando señales que corresponden a voz, tendrán una morfología que presentará una mayor similitud que en el caso de ruido ambiente y se obtendrán valores máximos de correlación para los desplazamientos temporales que produzcan mayores coincidencias entre ambas señales. El valor de los desplazamientos temporales que hagan máximo el valor de correlación corresponderán con el retardo existente entre ambas señales.

- **Módulo de integración de métrica para pares de micrófonos**

  Se encarga de procesar, utilizando diferentes estrategias y para cada trama, el valor de correlación máximo de todos los pares de micrófonos que se han utilizado para el cálculo de las correlaciones individuales. Las posibles estrategias que ofrece, se basan en medidas de la suma, el cálculo del máximo o la suma de logaritmos de la métrica de correlación. En este caso, se ha implementado la estrategia basada en el cálculo del máximo valor de entre todos los datos de CSP disponibles de todas las combinaciones de pares de micrófonos que se puede observar en la ecuación 2.3.

$$M[n] = \max_{\forall par(i,j)} CSPmax[par(i,j)][n] \tag{2.3}$$

  El número total de pares de micrófonos sobre los que se va a realizar el cálculo viene dado por la suma de las posibles combinaciones de dos micrófonos que se puedan hacer dentro de un mismo array, teniendo en cuenta el conjunto de todos los arrays disponibles. Se ha llevado a cabo esta implementación para crear un sistema flexible ante posibles incorporaciones de nuevos arrays de micrófonos.

Decisión

La etapa de decisión se encarga de determinar si la porción de señal analizada es voz o no. Se implementa una máquina de estados o autómata que se encarga de determinar en que estado se encuentra el sistema en cada momento (espera, voz o no voz). Dependiendo del estado en el que se encontrara en momentos anteriores y mediante la comparación del resultado de los cálculos de CSP con un umbral de decisión, se puede determinar si la ventana de audio actual se trata de voz o ruido.

- **Módulo de cálculo del umbral**

  Como se ha comentado en el apartado 2.2.1, en el sistema de detección de actividad de voz original se utilizaban ficheros de audio para poder realizar la estimación del umbral. En una ejecución en tiempo real no es posible disponer de todos los datos de audio para poder obtener a partir de ellos una estimación del margen o umbral que determinará la decisión.

  Por ello, se ha implementado un módulo de cálculo de umbral adaptado a los requerimientos de una aplicación en tiempo real, de modo que el valor de éste se va actualizando periódicamente para hacerlo flexible ante posibles variaciones del entorno en cuanto a potencia de señal o ruido ambiente. Se explicará en profundidad en el apartado 3.2.2, a partir de la página 24.

- **Módulo de detección de actividad de voz**

  Esta basado en el uso de un autómata que, a partir de unos parámetros dados, nos dice si detecta voz o no voz en una ventana determinada. Originalmente este autómata estaba implementado sobre ficheros de audio, de modo que ha sido precisa una remodelación de esta etapa.

  El flujo de este autómata se basa en la existencia de tres posibles estados: espera, posible voz y posible fin. Los parámetros que determinarán en que estado se encuentra el sistema en cada instante son los que corresponden a la duración de la voz y la duración del silencio, estableciendo un cierto margen para posibles pequeños silencios que no supongan una ausencia de voz. El funcionamiento de este módulo se explicará detalladamente en el apartado 3.2.1, a partir de la página 21.

2.2.4 Módulo de cálculo de potencia acústica dirigida (Steered Response Power)

La principal ventaja de los procedimientos de localización basados en TDOA es su simplicidad y bajo coste computacional. Sin embargo, su utilidad en entornos acústicos reales es limitada, ya que su rendimiento decae cuando el sistema se expone a reverberación y niveles altos de ruido. Además, la posición estimada generalmente consiste en una dirección de llegada DOA más que en una localización espacial exacta. Las estrategias basadas en *beamforming* conllevan una mayor carga computacional, pero tienden a ofrecer un sistema de localización más robusto.

La elección de el tipo de micrófonos que forman las agrupaciones ha permitido el diseño e implementación de un sistema de localización basado en patrones de directividad dirigidos mediante la técnica de *beamforming*. Además, cuanto mayor es el número de micrófonos mayor es la calidad de la localización. La posición relativa entre micrófono y micrófono y su posición frente a la fuente sonora puede reducir drásticamente el número de micrófonos para conseguir la misma calidad en la localización.

Las situaciones en las que el hablante se encuentra en un entorno cerrado, lejos de los sensores receptores, están marcadas por una baja relación señal-ruido SNR y efectos de reverberación. Hacer uso de arrays de micrófonos es una forma eficiente de atenuar estos fenómenos. En general, la fuente sonora no suele permanecer estática y sera necesario realizar un seguimiento de su localización alrededor de la

habitación con el fin de explotar las características de los arrays de micrófonos: centrando el patrón de recepción en los alrededores de la localización de la fuente con el objetivo de evitar ruidos indeseados, otras posibles fuentes de audio y reverberaciones.

El patrón de directividad de un array lineal de sensores idénticos y equi-espaciados depende de tres principales factores:

- El número de elementos del array.
- La separación entre dichos elementos.
- La frecuencia.

La máxima ganancia es la que ofrece a señales que se reciben de una dirección perpendicular al plano que forma el array $\phi = 90^o$. *Beamforming* es la técnica que permitirá dirigir el patrón de directividad del array de micrófonos a las diferentes direcciones espaciales. Mediante la modificación de los valores de amplitud, podemos variar la forma del patrón de directividad, mientras que la modificación de las ponderaciones de fase, puede controlar la posición angular del lóbulo principal. Este tipo de técnicas determinan estas ponderaciones de fase con el fin de obtener el direccionamiento deseado de la directividad del array.

Filter-and-sum Beamformer

Esta tecnica parte del resultado que ofrece un *beamformer* del tipo *delay-and-sum*, ya que introduce mejoras en toda la señal capturada debido a que la señal deseada, la que proviene del plano perpendicular con una orientación de 90^o, se suma en fase incrementando su potencia. Mientras que las señales no deseadas y el ruido proveniente de diferentes direcciones se suman desfasadas y decae su potencia.

Se considera que el algoritmo *filter-and-sum beamforming* es una generalización de el delay-and-sum beamformer ya que simplemente consiste en añadir un pre-filtrado a cada uno de los canales. Es importante tener en cuenta que, dependiendo del filtro que se escoja, no solamente puede variar la fase sino también los valores de amplitud en un *filter-and-sum beamformer*.

2.2.4.1 Algoritmo de localización

El algoritmo utilizado para la localización se basa también en la estimación del retardo temporal. Para la obtención del TDOA se hace uso de la técnica de análisis de CSP , también conocida como GCC , la cual se ha explicado previamente en la sección 2.2.3, a partir de la página 11.

Una vez se conoce la diferencia temporal entre las señales de dos micrófonos, se puede hacer uso de ella junto con la información relativa a la posición espacial de los micrófonos con el fin de generar las curvas hiperbólicas que representan los lugares geométricos donde puede encontrarse el locutor de acuerdo con el TDOA obtenido. Éstas curvas hiperbólicas generadas por un único par de micrófonos, se intersectan con las curvas obtenidas del resto de micrófonos dando lugar a una estimación de la localización del hablante.

Estas técnicas hacen uso de *beamforming* que puede ser aplicado ya sea a la captura de la señal de audio o a la localización de la fuente sonora. Si se conoce la posición del locutor, el *beamformer* puede ser enfocado a la misma con el fin de obtener una versión mejorada de la señal. Por el contrario, si la localización de la fuente no es conocida, el *beamformer* puede usarse para escanear, o dirigir el array, sobre un conjunto de localizaciones espaciales en un espacio de búsqueda predefinido. Cuando se usa de esta forma, la salida del *beamformer* se conoce como *Steered Response*. A continuación, se utiliza un estimador *Maximum Likelihood (ML)* busca un máximo en la potencia de salida que debe coincidir con la localización del hablante.

El tipo más simple de *Steered Response* se obtiene a partir de un *delay-and-sum beamformer*. Diferentes desplazamientos temporales, diseñados para coincidir con los retardos de propagación de la fuente de audio, se aplican a las señales del array. Posteriormente, estas señales son alineadas en el tiempo y sumadas en conjunto para conformar una única señal de salida. *Beamformers* mas generales y sofisticados aplican filtros a las señales procedentes de los arrays además de realizar esta alineación temporal. En este caso, se hará uso de filtros PHAT , que han demostrado ser útiles en términos de la estimación de TDOA.

Con todo esto, resulta una técnica robusta que crea una equivalencia entre SRP y la suma de todas las transformadas de fase de las posibles combinaciones de pares. Esta técnica se denomina SRP-PHAT y su robustez radica en el hecho de que explota la redundancia espacial de los micrófonos promediando todos los posibles pares de cruces GCC-PHAT.

El algoritmo SRP-PHAT

La salida de un *filter-and-sum beamformer* de M elementos es equivalente a un *beamformer* de dominio temporal que puede ser usado como medio para la localización de la fuente sonora dirigiendo el array a un conjunto de puntos espaciales de interés específicos y analizando la potencia de la señal de salida en cada uno de ellos. Cuando se enfoca a un punto que corresponde con la localización del locutor, SRP debería alcanzar un máximo global. La expresión para el *Steered Response Power* de una determinada localización espacial puede ser expresada como la correspondiente salida del *filter-and-sum beamformer*.

Sin embargo, esta técnica dependiente de la potencia puede presentar en la práctica picos en un determinado número de localizaciones incorrectas debido a las condiciones de reflexión del entorno o al efecto de la geometría del array, induciendo a error los resultados de la localización. La elección de los filtros adecuados puede ayudar a minimizar estos efectos. Como se ha visto, la estrategia seguida por *Phase Transform* PHAT de ponderación de cada componente frecuencial por igual ha demostrado obtener respuestas correctas en situaciones prácticas.

2.2.5 Sistema de visualización 3D

Como ya se comentaba en el apartado 2.2,a partir de la página 5, para la representación de la localización final obtenida se utiliza un sistema de visualización 3D que simula la vista del espacio inteligente en el que se lleva a cabo este trabajo.

En concreto, se dispone de un espacio inteligente virtual en 3D que permite la visualización tanto del entorno como de ciertos objetos que que permiten una interacción hombre-máquina a modo de actuadores del espacio inteligente. Gracias a esta sencilla e intuitiva representación se puede tener una vista funcional del espacio inteligente que permite, tanto hacer cambios de vistas y proyecciones como observar cambios en los actuadores del entorno virtual o representar la posición del usuario de dicho entorno.

Este sistema de visualización virtual se ha desarrollado sobre la librería de gráficos `OpenGL` que define una *Application Programming Interface (API)* multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos. Para más información sobre `OpenGL` se puede consultar [12].

La representación visual que nos ofrece este sistema consta de una base o entorno sobre el que aparecen una serie de objetos, como son los actuadores y sensores. El entorno del espacio virtual constituye la estructura del mismo y se puede observar en la figura 2.7a. Únicamente incluye las estructuras básicas que permiten modelar el propio espacio. Para ello el programa desarrollado carga un fichero base en el que se encuentra definida la estructura del entorno y que incluye: suelo, techo, las cuatro paredes, marco de la puerta, dos marcos de ventana, una columna central, dos mesas laterales, una balda larga en la

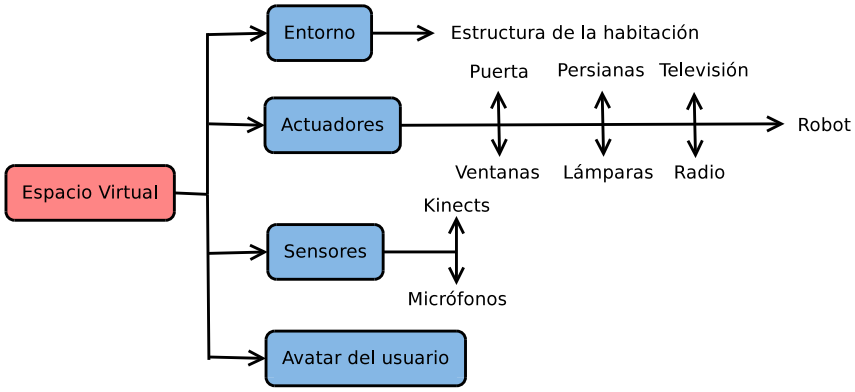


Figura 2.6: Elementos del espacio virtual.

pared derecha, una balda corta en la pared derecha, un armario en la pared izquierda, una pizarra, una superficie de proyección y cuatro marcos donde se sitúan cuatro arrays de micrófonos que se encuentran en la sala.

Esta estructura no es modificable por el usuario y compone el esqueleto sobre el que hemos construido el resto de actuadores, el uso de la librería `OpenGL` nos permite además implementar una serie de transformaciones que permitan adaptar la vista, hacer zoom, realizar rotaciones en los tres ejes y alternar entre vista 2D y 3D.

Sobre el entorno inteligente virtual se incluyen diferentes objetos, como son los actuadores y sensores que podrían encontrarse en el *ispace*. Concretamente, algunos de los actuadores que se representan en este espacio virtual no corresponden con los actuadores reales que existen en la sala, sino que se ha preferido utilizar actuadores virtuales de modo que se pueda tener una idea mas amplia en cuanto a los posibles objetos de ejemplo, de los cuales podría disponerse en el espacio inteligente en un futuro.

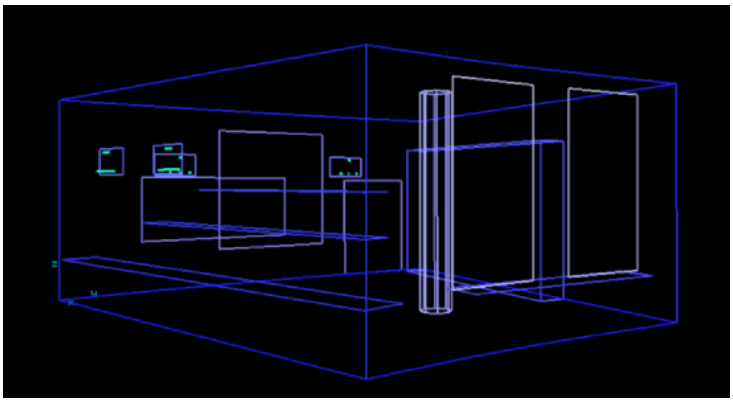
Se han implementado una serie de actuadores que se han añadido a la estructura base del entorno. Estos objetos son controlados por la lógica del sistema del espacio virtual, y pueden realizar acciones en función de dicha lógica. Estas acciones pueden ser más o menos complejas dependiendo del objeto, desde una puerta que puede abrirse o cerrarse, a la luz del techo que puede encenderse y apagarse o aumentar o disminuir la luminosidad de forma progresiva, persianas que se suben y se bajan o hasta el movimiento de un robot. La visualización de los objetos puede activarse o desactivarse, de forma que se permite que la estructura del espacio virtual pueda ser usada en distintas aplicaciones sin los actuadores presentes.

En el espacio virtual se han incluido también los sensores disponibles, con el objetivo de proporcionar una referencia visual rápida al usuario que le permita saber dónde se encuentran situados en el espacio real. Los sensores que se han representado son dos sensores Kinect y los diferentes arrays de micrófonos con los que cuenta la sala. En la figura 2.7b se puede observar el entorno virtual con los diferentes sensores y actuadores.

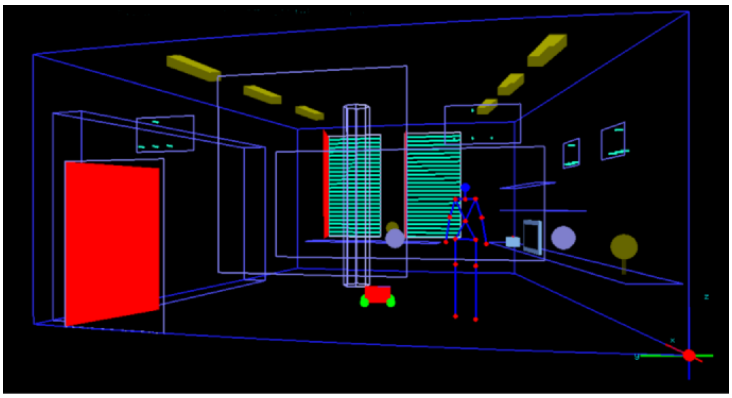
Para el desarrollo de este trabajo se va a usar una de sus funcionalidades mas básicas con un entorno virtual mas simplificado en el que no se muestran detalles de los actuadores y sensores, para obtener una vista precisa y sencilla de la localización del locutor, que el sistema de visualización generará a partir de la información del punto estimado.

2.3 Sincronismo de procesos

En el sistema que se esta tratando en este proyecto, el primer proceso que tiene lugar es el de transformación de la señal sonora analógica en una serie de datos digitales sobre los que se podrá trabajar.



(a) Entorno virtual.



(b) Entorno virtual con objetos.

Figura 2.7: Visualización 3D.

Estos datos deberán estar disponibles en un lapso de tiempo lo mas corto posible, ya que posteriormente tendrán que realizarse los cálculos y evaluaciones pertinentes en el resto de módulos antes de que haya llegado el próximo conjunto de datos de audio. Se puede, por lo tanto, considerar que éste será el proceso más crítico en cuanto a rapidez y sincronismo.

Para solventar esta cuestión, en el módulo de adquisición de la señal, se han utilizado procesos cooperantes para realizar el procesado de los datos haciendo uso de programación concurrente sobre varios hilos, mediante la utilización de la biblioteca POSIX Threads. Estos hilos poseen recursos de memoria compartida por lo que debe existir un sincronismo entre ellos para evitar la inanición o condiciones de carrera.

Por ello, se ha sido necesario el uso de semáforos. De esta forma, el sistema operativo garantiza que al iniciar una operación con un semáforo, ningún otro proceso puede tener acceso al semáforo hasta que la operación termine o se bloquee. Esta atomicidad[1] es absolutamente esencial para asegurar que la sección crítica cumple los siguientes requisitos:

- Exclusión mutua: Si el proceso P_i está ejecutando su sección crítica, ningún otro proceso puede estar ejecutando su sección crítica.

- Progreso: Si ningún proceso está ejecutando su sección crítica y existen algunos que quieren entrar en su sección crítica, sólo los procesos que no estén ejecutando su sección restante pueden participar en

[1] Propiedad que asegura que una operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias. Si una operación atómica consiste en una serie de pasos, debe realizarse en su totalidad o en caso de ser interrumpida poder deshacer sus acciones de modo que fuese como si no se hubiese realizado acción alguna.

la decisión de qué proceso puede ingresar en su sección crítica, y esta selección no puede posponerse indefinidamente.

- Espera limitada: Hay un límite para el número de veces que otros procesos pueden entrar a sus secciones críticas después de que un proceso ha solicitado entrar en su sección crítica y antes de que se le otorgue la autorización para hacerlo.

De esta forma, se dispondrá de dos procesos , un proceso productor y un proceso consumidor. El proceso productor se encarga de generar los elementos y copiarlos en el buffer circular que alojará los datos de audio obtenidos. El proceso consumidor extrae los elementos del buffer circular y los copia a un buffer auxiliar de procesado, en éste proceso estarán contenidos los algoritmos de procesado pertenecientes a los diferentes módulos del sistema. Los elementos que son leídos se marcan como libres, para poder ser sobre-escritos por el proceso productor.

Este esquema de funcionamiento requiere que el proceso productor siempre se ejecute antes que una operación del proceso consumidor, ya que de no ser así éste podría experimentar inanición al no existir ningún elemento para procesar en el buffer circular. De la misma forma, el proceso productor deberá comprobar si el buffer esta lleno antes de realizar una nueva escritura, o en caso contrario podría existir overflow y sobre-escribirse datos que aún no han sido leídos.

Dentro de este bloque, la división de los datos en ventanas permite también cumplir con las especificaciones de sincronismo y rapidez de procesos que impone la etapa de adquisición. Ha de considerarse que si el tamaño de datos es muy grande, existirá gran cantidad de fallos de página y de caché, lo que retardará la ejecución del manejo de datos. Por ello, la segmentación de los datos permite que su procesado sea mas rápido, se debe encontrar un tamaño de ventana que cumpla también un compromiso entre rapidez y cantidad fiable de datos.

Esta es la única parte del sistema que se puede ”paralelizar” o seguir una ejecución concurrente. El resto de bloques deben ejecutarse de forma secuencial, ya que cada bloque depende del resultado que aporte la etapa anterior. Una vez obtenida la señal de audio, el módulo de estimación de la posición mediante SRP debe esperar la respuesta del vad y a su vez la visualización solo puede mostrar el resultado si se ha producido un punto estimado final válido como posición del locutor en el bloque SSL .

A pesar de regirse por una ejecución secuencial, una forma de optimizar el sincronismo de procesos es evitar cálculos redundantes, ya que tanto el bloque VAD como el bloque SSL basan sus cálculos en la obtención de la correlación entre pares de micrófonos. Se profundizará mas en este aspecto en el apartado 3.3, a partir de la página 27.

El hecho de poder descartar tramas de ruido mejora el tiempo que invierte el programa en procesar datos. La etapa de detección de actividad de voz, supone una gran optimización del sistema en cuanto a su ejecución en tiempo real se refiere. Según el resultado que obtengamos de este bloque, podemos evitar ejecuciones del algoritmo de localización que vayan a trabajar sobre audio que no corresponda a una señal de voz propiamente dicha. Si no dispusiéramos de un detector que descarte tramas de ruido ambiente, obtendríamos como resultado final la estimación de puntos de localización que realmente solo corresponden a los puntos de máxima potencia de señal dentro del espacio en el que se trabaja, es decir, los puntos en los que se genere un mayor ruido.

2.4 Requisitos y posibles conflictos de una implementación en RT

Un sistema en tiempo real posee una serie de características especiales que lo diferencian de los demás tipos de sistemas e introducen en la propia definición del sistema un conjunto de requerimientos no funcionales, que no se refieren directamente a las funciones específicas si no a propiedades emergentes como pueden ser, requisitos de fiabilidad, eficiencia o implementación.

Las funciones que realiza un sistema en tiempo real suponen una interacción constante con el entorno físico que le rodea y debe responder a los estímulos que recibe del mismo dentro de un plazo de tiempo determinado. Para que el funcionamiento del sistema sea correcto no basta con que las acciones sean correctas, también deben ejecutarse dentro de un intervalo de tiempo especificado que puede ser más o menos crítico.

El comportamiento temporal de las tareas que va a realizar se especifica mediante sus atributos temporales:

- Cuándo se ejecutan: esquema de activación
- Qué plazo tienen para ejecutar cada acción

Según el criterio que sigue la activación de cada evento que se encargará de realizar cada una de las funciones, los sistemas de pueden clasificar en dos grupos:

- Activación periódica

  Las tareas se ejecutan con un determinado periodo y a intervalos temporales regulares.

- Activación aperiódica

  Las tareas se ejecutan cada vez que ocurre un suceso determinado. Estos a su vez se pueden dividir en dos conjuntos según sigan una activación aperiódica esporádica, con una separación mínima entre activaciones, o una activación aperiódica estocástica, a rachas e irregular.

En este caso, se está desarrollando un sistema en tiempo real de activación periódica, ya que la activación de las acciones en las que el sistema interactúa con el entorno se rige por el tamaño de la trama que constituirá el intervalo temporal o periodo entre activaciones.

En cuanto a los requisitos temporales de este sistema, se puede definir como de tiempo real flexible (soft real-time). En este tipo de sistemas se puede asumir la pérdida de determinados plazos o datos en algunos momentos sin que sea crítico para su correcto funcionamiento. En instantes de sobrecarga el comportamiento puede degradarse, pero deben asegurar la recuperación ante posibles fallos.

Como se ha explicado en el apartado 2.3, una de las características de estos sistemas es la simultaneidad de las acciones (concurrencia). Los dispositivos físicos controlados funcionan al mismo tiempo y las tareas que los controlan actúan concurrentemente. Están compuestos por dispositivos de entrada y salida especiales y los manejadores de los dispositivos forman parte del software de la aplicación.

Otras cualidades que debe poseer un sistema de este tipo son fiabilidad y seguridad. La fiabilidad supone que el sistema sea capaz de proporcionar el servicio especificado, siendo máximo el tiempo entre averías. La seguridad es también un problema de programación en los sistemas en tiempo real. Es necesario establecer controles de verosimilitud en los programas. Hay que utilizar medios de protección de los ficheros y utilizar los dispositivos de protección de memoria. En un sistema que posea un alto grado de multiprogramación es particularmente difícil conseguir una buena seguridad en los programas.

2.5 Conclusiones

A lo largo de este capítulo se han intentado resumir las principales características que constituyen la base teórica de este proyecto. Se han explicado de forma concisa distintos conceptos necesarios para la comprensión del trabajo desarrollado y la filosofía que deberá aplicar.

De los puntos teóricos son especialmente relevantes los correspondientes al sistema de adquisición de audio, extracción de características de las señales adquiridas, el módulo de decisión del VAD y el módulo de estimación de la posición basado en SRP . También se ha estudiado la aportación de cada uno de los bloques al sistema completo y el sincronismo que deberán cumplir para un correcto funcionamiento. No es objetivo de este proyecto, explicar en profundidad ninguno de dichos apartados, para más información puede consultarse la bibliografía incluida en cada una de las secciones.

Capítulo 3

Desarrollo

3.1 Introducción

En este nuevo capítulo se expone el proceso de desarrollo e implementación del demostrador de localización de una fuente sonora en tiempo real. Se van a describir cada uno de los cambios que se ha precisado hacer en los módulos ya implementados, así como las nuevas aportaciones que han sido necesarias como pueden ser las modificaciones del bloque VAD para conseguir adaptarse a la ejecución en tiempo real, la creación y modificación de las funciones de inicialización de los diferentes bloques, la integración de todas las etapas evitando en la medida de lo posible cálculos redundantes, entre otros.

Por último, se explicará la forma en que se han integrado todos los bloques y se han sincronizado sus procesos. También se comentarán los resultados obtenidos de la implementación y ejecución del sistema completo y se realizará una evaluación del demostrador de localización desarrollado.

3.2 Implementación del bloque VAD

La adaptación de este módulo para introducirlo en el sistema en tiempo real ha supuesto una significante remodelación del mismo. Se parte de la base implementada en el proyecto original, siguiendo la idea global y los cálculos sobre los que se desarrolla. Sin embargo, esta nueva implementación del detector de voz introduce varias modificaciones y añade nuevas funcionalidades que exige una ejecución en tiempo real con respecto al VAD inicial. Estos cambios se detallan en los apartados que se tratarán a continuación.

3.2.1 Módulo de decisión en tiempo real

En éste módulo, para llevar a cabo la decisión de si la última trama adquirida contiene muestras que corresponden a voz o simplemente a ruido ambiente, se ha desarrollado un autómata basado en una máquina de estados. Según este autómata, el sistema podrá encontrarse, en cada momento, en uno de los tres estados posibles: ESPERA, POSIBLE VOZ y POSIBLE FIN.

El autómata se encarga de analizar los datos de correlación obtenidos para la ventana de audio actual y, en base a una serie de parámetros preestablecidos, determinará la existencia o no de voz y el estado en el que se encuentra el sistema en ese instante. Este conjunto de parámetros preestablecidos esta formado por constantes y valores de comparación que se describen a continuación:

- `Minvalue`. Es el valor que se tomará como umbral para determinar si la ventana que se esta analizando es una posible candidata para albergar voz o no (véase la sección del módulo de cálculo del umbral en el apartado 3.2.2, a partir de la página 24).

- `TimeDown`. Representa el periodo de tiempo máximo (en segundos) que el valor de correlación obtenido puede permanecer por debajo del umbral `Minvalue` y seguir siendo candidato a la existencia de voz o seguir afirmando que existe voz. Se ha establecido a 0.3 segundos, ya que es un valor que permite cubrir posibles silencios entre palabras o posibles fluctuaciones en los valores de correlación correspondientes a los datos de audio.

- `TimeVoiceOK`. Representa el periodo de tiempo mínimo (en segundos) que el valor de correlación debe permanecer por encima del nivel del umbral para poder determinar que realmente existe voz. Se ha fijado un valor de 0.5 segundos, ya que nos permite cubrir los casos de posibles ruidos que puedan generar valores altos de correlación.

Estos tres parámetros ya se encontraban en la implementación original del VAD , sin embargo, para el desarrollo del sistema en tiempo real, se ha añadido un nuevo parámetro: `WaitingTimeVoiceOK`. El parámetro `TimeVoiceOK`, supone un retardo en la determinación de la existencia de voz. Si la estimación de la posición del locutor se realizara únicamente cuando tenemos la certeza de que existe voz, estaríamos sufriendo este retardo y perdiendo los datos correspondientes a esas tramas en las que el sistema aún se encontraba en un estado de posible voz. Para solventar este hecho en la medida de lo posible, se ha introducido el parámetro `WaitingTimeVoiceOK` que representa el tiempo mínimo (en segundos) que el dato de correlación obtenido debe permanecer por encima del umbral para que se genere una llamada al algoritmo SRP para la estimación de la localización. Este parámetro debe ser menor que `TimeVoiceOK` para que el sistema pueda realizar los cálculos para obtener la posición antes de que se haya determinado si realmente existe voz. De esta manera, habrá situaciones en las que nos encontremos con tramas de voz donde se ganará tiempo en la localización y otros casos en los que la trama obtenida contenga ruido y los cálculos que se hayan realizado deban ser descartados.

Además de estos parámetros constantes o predefinidos, el autómata hace uso de una serie de variables que se usarán para determinar el estado del sistema como se explicará posteriormente. Estas variables son las siguientes:

- `longVoz`. Representa el periodo de tiempo transcurrido desde que se produjo un evento de posible voz, es decir, desde que se recibió un valor de correlación que superó el nivel del umbral. Esta variable se compara con `TimeVoiceOK` para determinar si el periodo de tiempo transcurrido es suficiente como para asegurar que existe voz.

- `longSilencio`. Representa el periodo de tiempo transcurrido desde que se produjo un evento de posible fin, es decir, desde que se recibió un valor de correlación que se encontraba por debajo del umbral. Esta variable se compara con `TimeDown` para determinar si el periodo de tiempo transcurrido es suficiente como para asegurar que no existe voz.

- `FrameIndex`. Esta variable es un contador que se usa para identificar el número de ventana que acabamos de adquirir. Gracias a ella se obtiene la referencia a partir de la cual se calculan `longVoz` y `longSilencio` junto con el valor de `frameShiftSecs`.

- `call_SRP`. Es una variable de salida, se asigna según el estado del autómata en el que nos encontremos y el valor del resto de variables. Representa el tipo de llamada que se realizará al algoritmo SRP para la localización. Más adelante se profundizará en los posibles valores que puede tomar dicha variable y como se actúa en cada uno de los casos.

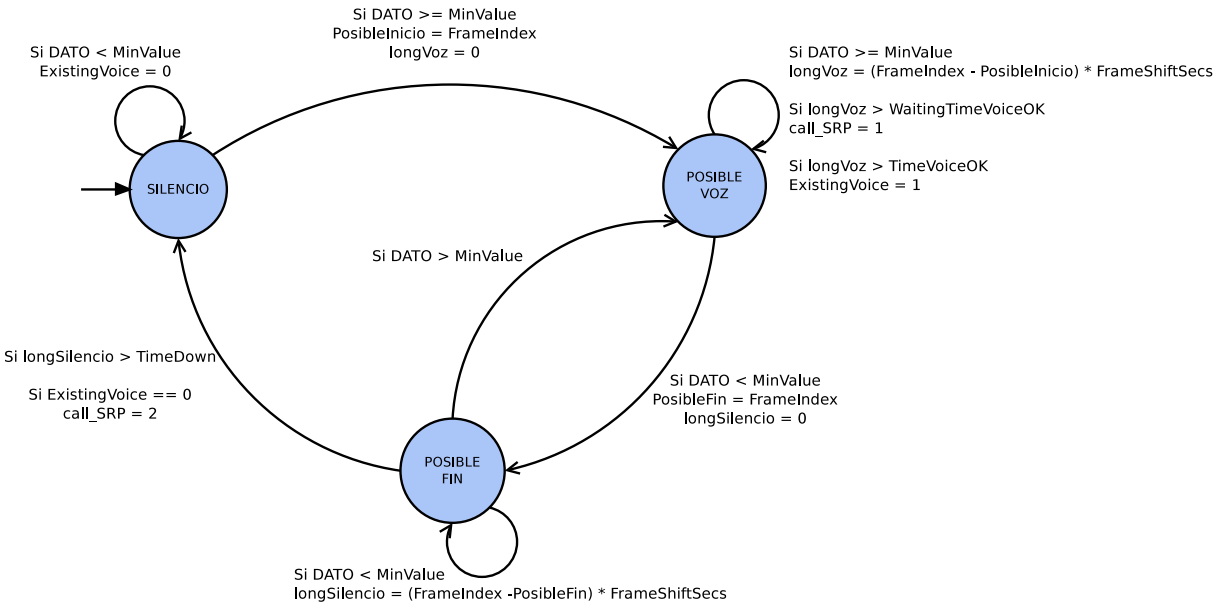


Figura 3.1: Máquina de estados del autómata.

- `ExistingVoice`. Si `longVoz` ha alcanzado o superado el valor de `TimeVoiceOK` y se puede asegurar que realmente existe voz, este hecho se reflejará en el valor de esta variable.

Como ya se ha comentado anteriormente, el autómata se basa en una maquina de estados que consta de tres estados: ESPERA, POSIBLE VOZ y POSIBLE FIN. En la figura 3.1 se puede observar el grafo correspondiente a dicho autómata. Como se indica en la figura, el estado de reposo o estado inicial será el de ESPERA y el sistema permanecerá en dicho estado siempre que el dato de correlación no supere el umbral `MinValue`. Que el sistema se encuentre en el estado de ESPERA supone que estamos ante una situación en la que no existe voz. En el momento que se obtenga un valor de correlación que supere el umbral, se pasará al estado POSIBLE VOZ. En esta transición se resetea el valor de `longVoz` y se almacena el identificador de ventana `FrameIndex` en la variable `PosibleInicio` para posteriormente obtener el periodo de tiempo que transcurrió desde dicha transición como se muestra en la ecuación 3.1.

$$longVoz = (FrameIndex - PosibleInicio) * frameShiftSecs \tag{3.1}$$

Se permanecerá en el estado POSIBLE VOZ siempre que los valores de correlación obtenidos superen el nivel de umbral y se irá aumentando el valor de `longVoz`. En cada nueva ejecución del autómata se comparará este valor con `WaitingTimeVoiceOK` y `TimeVoiceOK`, si se supera el primero, el sistema se encontrará en una situación de posible voz aún por verificar, pero que producirá una llamada al algoritmo de SRP que aún no sabe si trabaja con datos correspondientes a voz o silencio. Por otro lado, si `longVoz` supera el valor de `TimeVoiceOK` se tendrá la certeza de la existencia de voz y la variable `ExistingVoice` tomará el valor '1', en este caso la llamada a SRP se considera válida por trabajar con datos correspondientes a voz real.

La transición del estado POSIBLE VOZ a POSIBLE FIN se produce cuando el dato de correlación obtenido tiene un valor inferior a `MinValue`, en esta transición se resetea el valor de `longSilencio` y se almacena el identificador de ventana `FrameIndex` en la variable `PosibleFin` para posteriormente obtener el periodo de tiempo que transcurrió desde el comienzo de un posible silencio que se calculará de forma análoga a `longVoz`. Se permanecerá en este estado mientras los datos de correlación con los que se trata en cada nueva ejecución del autómata estén por debajo del valor del umbral y se irá aumentando el

valor de `longSilencio`. En este caso tenemos dos posibles cambios de estado, si el valor de correlación obtenido supera el umbral el sistema regresará al estado POSIBLE VOZ y la ejecución continuará su funcionamiento como se ha explicado anteriormente. Por otro lado, si el valor de `longSilencio` alcanza el parámetro de comparación `TimeDown`, se tendrá la certeza de que el sistema se encuentra en un entorno de silencio y se pasará al estado de ESPERA iniciándose así todo el proceso de nuevo.

La llamada al algoritmo de localización SRP , se realiza atendiendo al valor de `call_SRP`. Este parámetro de salida puede tomar tres distintos valores: 0, 1 ó 2. Inicialmente en cada nueva ejecución del autómata, su valor será 0, lo que implica que no se llevará a cabo el proceso de estimación de la posición del posible hablante. Cuando la variable `longVoz` alcance el valor de `WaitingTimeVoiceOK`, el valor de `call_SRP` será 1, lo que implica que el flujo de ejecución del sistema continuará con la estimación de la localización del supuesto locutor. Por último, esta variable de salida tomará el valor 2, si en el momento en el que se tiene certeza de un silencio real `ExistingVoice` está a 0. Esto implica que no se realizará el cálculo de la localización y que además dicho silencio real, parte de una situación de posible voz que no alcanzó el tiempo suficiente como para considerarse voz real, de modo que cualquier posible llamada a SRP que se hubiese realizado en ese periodo sería una llamada no válida.

El bloque encargado de gestionar las llamadas no válidas a SRP será el de visualización 3D. En el apartado 3.4, a partir de la página 27, se explicarán los pasos a seguir según nos encontremos en cada uno de los casos, ya que el descarte de estimaciones cuando nos encontremos ante una llamada a SRP que no corresponde a voz se realizaría en este módulo para evitar la visualización de datos incorrectos.

En este bloque hay un último aspecto a tener en cuenta. La variable `FrameIndex` alberga el valor correspondiente al número de trama con la que el sistema trabaja en cada momento. Esto implica que dicho valor puede aumentar considerablemente cuando el sistema se ejecuta durante un periodo de tiempo elevado y se debe estudiar si sería posible alcanzar una situación de overflow. Esta variable se ha declarado del tipo `long unsigned int`, que tiene un tamaño de 32bits y admite un valor máximo de 4294967295. El valor de la variable se aumenta con cada nueva adquisición de la trama de audio, de modo que entre cada nuevo valor transcurrirá un tiempo equivalente a `FrameShiftSecs` y el tiempo que deberá transcurrir hasta que tenga lugar una situación de overflow será el que se muestra en la ecuación 3.2.

$$tiempo_{overflow} = 4294967295 * \texttt{FrameShiftSecs} \tag{3.2}$$

El valor de `FrameShiftSecs` se debe fijar aproximadamente a la mitad del valor de `FrameSizeSecs`. Atendiendo a los requisitos de memoria, se ha fijado el valor de `FrameSizeSecs` a 0.1 segundos y el de `FrameShiftSecs` a 0.5 segundos. Teniendo en cuenta estos valores, el tiempo que debe transcurrir hasta que se produzca overflow será 858993459 segundos o lo que es lo mismo 6.8096 años. De modo que se puede asegurar que la variable `FrameIndex` no producirá overflow en condiciones normales de funcionamiento.

3.2.2 Módulo de cálculo del umbral

Para obtener el valor del umbral sobre el que se evaluará el estado del autómata y se tomará posteriormente la decisión acerca de la existencia o no de voz en la señal de audio, se ha introducido este módulo previo a la evaluación y toma de decisiones.

En el desarrollo del proyecto original [2], se disponía de ficheros de audio que contenían los datos a procesar. De esta forma, se realizaba el cálculo de la correlación por tramas y por canales y se extraía el valor del máximo CSP de cada una de las tramas. De entre todos los valores de CSPmax obtenidos

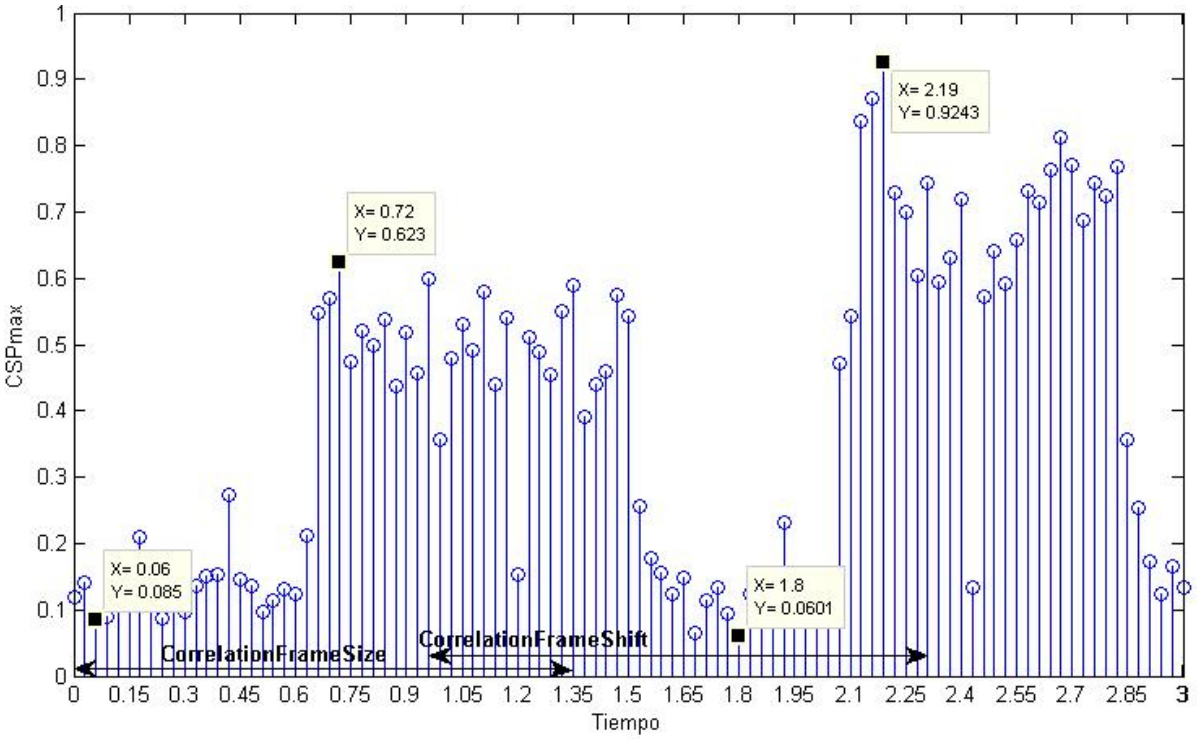


Figura 3.2: Ejemplo de cálculo del umbral de detección.

se delimitaba el máximo y el mínimo. Una vez se disponía de estos datos, se establecía un nivel que delimitaría el umbral con respecto a los límites máximo y mínimo.

Al trabajar con una aplicación que opera en tiempo real, estos cálculos deben realizarse sobre los datos que se van obteniendo en cada instante. No obstante, para que el valor del umbral estimado sea adecuado y se ajuste a las características que ofrece la señal de audio, es preciso disponer de un conjunto suficiente de datos sobre el que realizar la evaluación. Para ello, se debe mantener una historia a cerca de los valores pasados que se irán almacenando junto con los nuevos datos que se vayan adquiriendo en cada momento.

Sin embargo, solo podemos mantener una historia de los datos que se remonte un determinado tiempo atrás, ya que de crecer demasiado el tamaño de los datos la memoria se saturaría y el cálculo supondría una gran carga computacional. Para solventar esta cuestión, se ha optado por una implementación que sigue la idea de segmentación de una señal. Los datos se almacenarán en memoria hasta una longitud determinada y se irá haciendo un desplazamiento de éstos, de forma que se descarten los datos mas antiguos para dar cabida a los actuales. Este método que se ha empleado para la implementación del módulo y que se muestra en la figura 3.2, además permite la actualización del umbral con una determinada frecuencia, lo que le aporta flexibilidad ante posibles cambios en el entorno.

En la implementación de esta etapa dentro de un sistema en tiempo real, no vamos a obtener una estimación válida del umbral hasta que se haya realizado el cálculo sobre un número significativo de datos, de forma que tras arrancar la ejecución se precisará un tiempo de estabilización del sistema hasta que las tramas se puedan evaluar mediante un umbral coherente.

En el momento del desarrollo, se ha sometido a estudio el desplazamiento que se aplicará a los datos almacenados para el cálculo del umbral. Alcanzándose así estas dos posibles variantes:

- Desplazamiento de un único dato.

Este método permite evitar iteraciones sobre el array completo cada vez que se realiza el cálculo de nuevo. Esto solo será necesario en el caso de que la muestra desplazada y descartada corresponda con uno de los dos límites, máximo o mínimo, puesto que se deberá estimar un nuevo límite dentro del array.

En cuanto al desarrollo, el método que se está analizando conlleva una algorítmica que es mucho más fácil de implementar que en el caso de desplazamiento de un conjunto de muestras.

Sin embargo, un inconveniente que presenta es que el cálculo y la estimación de un nuevo valor de umbral se realiza con cada nueva entrada en el autómata.

- Desplazamiento de un conjunto de datos.

  En este método, solo se realiza el cálculo cuando se haya completado el desplazamiento y descarte del número de datos especificado, de modo que en algunas iteraciones del autómata no será necesario estimar el umbral. En contraposición, cada vez que se debe realizar el cálculo, se debe recorrer el array de elementos descartados en busca de coincidencias con los límites máximo y mínimo y en caso de encontrarse, también será necesario recorrer el array de correlaciones actual para determinar los nuevos límites.

  En cuanto al desarrollo, la algorítmica que permite exponer este método es más complicada de implementar, ya que se debe trabajar con dos estructuras de datos para ir descartando valores pasados y añadir los actuales.

En este proyecto se ha empleado un desplazamiento del array de correlaciones de una sola muestra o dato, debido a la carga computacional que libera al evitar iteraciones sobre arrays de datos y a la diferencia en cuanto a la complejidad en el desarrollo de la algorítmica. Según esta implementación, cada nueva trama adquirida generará un único valor de CSPmax, de modo que el umbral se volverá a estimar periódicamente en el momento en el que se disponga de un nuevo bloque de datos de audio, siendo este periodo igual al desplazamiento aplicado en la segmentación de la señal (`FrameShift`).

Sin embargo, a pesar de disponer de una actualización del umbral que se realiza a intervalos temporales de la orden de las decenas de milisegundos, no es deseable que con cada nueva iteración del autómata cambie el valor con el que se debe hacer la comparación para tomar la decisión. Es preferible que este valor se mantenga estable durante cierto tiempo y se actualice cuando haya pasado un intervalo suficientemente grande como para poder contemplar posibles cambios significativos en el entorno que se vean reflejados en el nuevo valor de umbral. De forma que, aunque se disponga de un dato actualizado no se informará de ello al autómata hasta que este intervalo temporal haya cumplido.

Dentro de este módulo existen varios parámetros que son configurables. Esto permite la evaluación de los resultados dependiendo del valor que se le de a cada uno de ellos y como responde el sistema frente a sus variaciones. Estos parámetros son:

- `CorrelationFrameSizeSecs`. Permite establecer el intervalo temporal que ocupará un array de datos de correlación completo, a partir de este dato y teniendo en cuenta el valor de `FrameShiftSecs` se determina el tamaño del array en muestras. De ello depende la cantidad de memoria que se deba reservar para alojarlo y el tiempo que tarda el sistema en estabilizarse, ya que hasta que no se completa este array no se obtendrá una estimación válida del umbral.

- `Threshold_update`. Este parámetro establece la frecuencia con la que se actualizará en el autómata el valor del umbral. Se expresa como un periodo temporal medido en segundos.

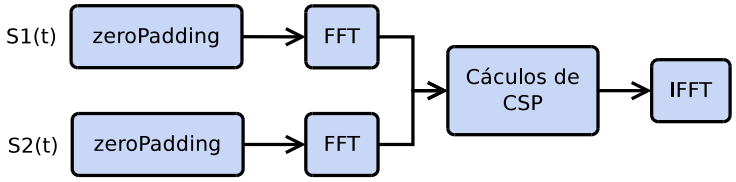


Figura 3.3: Cálculo de la correlación de dos señales.

- `Threshold_percent`. En el cálculo del umbral, el nivel con respecto a los límites máximo y mínimo que permitirá determinar su valor viene dado por este parámetro que se expresa en tanto por uno.

3.3 Integración de cálculos de correlación entre SRP y VAD

Para el desarrollo de este sistema de localización de un locutor en tiempo real se han incluido los bloques de detección de actividad de voz VAD y el módulo de localización basado en la respuesta en potencia dirigida SRP .

Además de las funcionales específicas de cada uno de los bloques, ambos tienen un módulo de extracción de características a partir de los datos de audio adquiridos. Este módulo es común a estos dos bloques y sus cálculos están basados en la diferencia temporal entre las señales adquiridas por los diferentes micrófonos para obtener la información de correlación entre las señales recibidas. Estos cálculos se explican con mayor profundidad en el apartado 2.2.3. En la figura 3.3, se puede ver un esquema a modo de resumen de los cálculos que se realizan para la obtención de los datos de correlación, incluyendo el relleno con ceros (zeroPadding) y las transformadas y transformadas inversas de Fourier (FFT e IFFT) para trabajar en el dominio frecuencial.

Por ello, para realizar la integración de estos dos bloques se han aunado en un solo módulo independiente los cálculos que son comunes a ambos. De este modo, una vez se adquieren las muestras de audio, previo a la detección de actividad de voz, se extraen las señales de audio y se realizan los cálculos pertinentes para la obtención de los datos de correlación que se almacenan hasta que se produzca una nueva adquisición de audio. Posteriormente cada uno de los bloques accederá a estos datos y hará uso de ellos para llevar a cabo su propia función.

3.4 Módulo de visualización 3D

En este proyecto, el módulo de visualización se ha utilizado sin realizar ningún cambio con respecto al original. Simplemente se representa el punto que se obtiene de la estimación de la posición del locutor en el espacio sobre el entorno 3D que reproduce este módulo.

Sin embargo, en este caso, además, recae sobre el módulo de visualización 3D la tarea de gestionar la salida que se obtiene del algoritmo de localización. Según el valor de `call_SRP`, los datos obtenidos serán fruto de una estimación de la localización válida o de un cálculo de la posición que corresponde a una situación en la que no existía voz realmente. De esta forma, el módulo de visualización tiene dos formas diferentes de gestionar o descartar las muestras no válidas:

- Aplicación de un buffer de visualización.

  Este método utiliza un buffer que se encarga de almacenar los últimos datos obtenidos para llevar una determinada historia de momentos anteriores, de forma que si se determina que las últimas

estimaciones de la localización del locutor no fueron válidas, se puedan descartar dichos datos antes de que sean visualizados. De esta manera se puede asegurar que los datos mostrados sobre el espacio virtual 3D siempre van a pertenecer a instantes en los que realmente existía voz. El inconveniente que presenta este método es que la visualización de los datos sufre un determinado retardo, con lo que perdemos en cierto modo la fidelidad con una ejecución en tiempo real.

- Visualización en tiempo real.

  Este otro método, sin embargo, se caracteriza por ser fiel a los requerimientos de una implementación en tiempo real. Los datos se visualizan en el mismo momento en el que son obtenidos, sin existir ningún tipo de historia sobre instantes anteriores. Aunque esta forma de gestionar los datos estimados supone mayor rapidez en la ejecución del sistema, también tiene un inconveniente. La visualización de los datos sin verificar si pertenecen a una llamada a SRP válida o no, supone que existirán determinados instantes en los que aparezcan datos de posición sobre el entorno virtual que no representarán la localización real de un locutor, ya que no existía voz en ese momento.

A la hora de decidir qué método se implementará en este sistema, existe un compromiso entre visualización en tiempo real o descarte de los datos que corresponden a momentos en los que no existe actividad de voz. La solución mas fiel a un comportamiento en tiempo real es mostrar todos los valores calculados en el momento que se obtienen aunque en determinados momentos se visualicen algunas muestras que no correspondan a señales de voz reales y será la que se implemente en este proyecto.

3.5 Callback function

El bloque de adquisición hace uso de varios hilos que trabajan de forma recurrente, como ya se explicó en el apartado 2.3 a partir de la página 16. Existe un hilo productor encargado de la adquisición de frames de audio y administrado por la librería `rtAudio()`, este hilo sigue una ejecución continua sin dejar de adquirir muestras de sonido hasta que el hilo de control se lo comunique. Otro de los hilos que componen la funcionalidad de este módulo, es el hilo consumidor que se encarga de presentar los datos a la función que los procesa. Ambos hilos hacen uso de un buffer circular al que agregará datos el productor y del que recogerá datos el consumidor para nutrir a la función de usuario. En este buffer se guardan un determinado número de items que actuarán a su vez como buffer de almacenamiento de los datos de audio previo a la entrada o salida del buffer circular. Estos items tienen el formato de una estructura de datos `TBufferItemInfo`. Los accesos que realizan cada uno de los hilos a este buffer circular se rigen por un semáforo que regula esta sección de exclusión mutua, evitando así condiciones de carrera.

Listado 3.1: Estructura que contiene los datos obtenidos en la adquisición.

```
typedef struct TBufferItemInfo{
  const char *fileName;
  SF_INFO sfInfo;
  RtAudioFormat rtFormat;
  unsigned int frameSizePerCh;
  unsigned int frameShiftPerCh;
  void *aux;
  TSizeInBytes sizeInBytes;
  void *Item1D;
  void **Item2D;
} TBufferItemInfo;
```

Al iniciarse los procesos de usuario, se inicia el proceso de inicialización del sistema, que configurará cada uno de los bloques para su correcto funcionamiento, y posteriormente se realiza la llamada a la función `__rtAudioIO()`, en ese momento comienza la configuración e inicialización interna del sistema de adquisición, se abre el stream de RtAudio y se inicia la adquisición en el hilo productor. El productor se encarga de introducir el item en el buffer circular, de actualizar los punteros de lectura y escritura y de gestionar posibles casos de overflow. Una vez se dispone de los datos en el buffer circular, el hilo consumidor se encarga de controlar la lectura y el preparado de frames para la ejecución de las funciones de procesamiento. Este hilo consume la mayor parte de los recursos del procesador y está constantemente ejecutándose (ejecuta un bucle infinito) siempre que haya frames en el buffer circular. Se copia la información que existe en el roundBuffer en un item de salida y se ejecuta la función `userCallback()`, a la que le entrega este outputItem para llevar cabo el procesado de la información contenida en él.

Listado 3.2: Función de inicio de la adquisición de audio.

```
extern int __rtAudioIO( const char * fileName,
  unsigned int audioDevID,
  unsigned int numChannels,
  unsigned int offset,
  unsigned int RtAudioBufferFramesCh,
  int frameSizeCh,
  int frameShiftCh,
  userCallback callback,
  unsigned int sampleRate,
  RtAudioFormat format,
  double numberItems,
  char manualInterleaving
);
```

De esta forma, al iniciarse el sistema el bloque encargado de la adquisición y el preparado del audio que reciben los diferentes canales será el que rija la ejecución y se encargue del sincronismo de todos los procesos que se llevarán a cabo en todo momento, debido a su ejecución recursiva y periódica. Una vez se dispone de los datos, y hasta que se vuelva a producir una nueva adquisición, se entra en la función de callback y comienza el tiempo disponible para el procesado de dichos datos. Por lo que será en la función de callback donde se implementarán las funcionalidades de los bloques VAD y localización por SRP ya que precisan de los datos adquiridos para poder realizar sus cálculos y estimaciones.

Un aspecto importante que se ha tenido en cuenta, es tanto el tamaño de trama como el desplazamiento de la misma que se han configurado, ya que de ello dependerá el volumen de datos que se deban procesar y el tiempo del que se dispondrá para hacerlo. El desplazamiento de la trama determina el tiempo que transcurre hasta que se produce una nueva llamada a la función de callback, por lo que desplazamientos grandes darán mas tiempo para que se procesen los datos. Sin embargo, el tamaño de trama debe ser proporcional al desplazamiento y esto supondría volúmenes de datos que precisarían una gran cantidad de memoria para poder procesarse, lo cual podría ser crítico para el sistema.

3.6 Funciones de inicialización y configuración

Con el objetivo de implementar un sistema flexible y abierto a posibles lineas futuras, se ha hecho independiente el proceso de inicialización y configuración de los bloques, del resto de la implementación

del sistema.

Además, cada uno de los módulos tendrá su propio proceso de inicialización individualizado, de modo que puedan agregarse o eliminarse según se precise su funcionalidad en el sistema o no. Esto es útil para la posible implementación de sistemas que hagan uso de alguno de estos módulos pero no de todos en su conjunto, pudiendo así incluir su funcionalidad de forma independiente. Esto es posible gracias a que el desarrollo del sistema también se ha implementado diferenciando los trabajos de cada uno de los bloques de forma individual.

Cuando comienza el proceso de ejecución del sistema y se arranca el módulo de adquisición de audio, ya debe haberse configurado todo el conjunto de parámetros que influirán en el desarrollo de la ejecución. Por lo tanto, este proceso de inicialización será previo al inicio de cualquier módulo que participe en la funcionalidad del sistema, en el momento de arranque de la ejecución.

De esta forma se tendrán las siguientes funciones de inicialización y configuración:

- `InitAudioIOLib`

  Se encarga de inicializar la librería RtAudio e incluir los parámetros necesarios para caracterizar los dispositivos de audio en el sistema.

- `InitRT`

  Su función es la de configurar todos los parámetros globales que se van a utilizar en el sistema completo y que corresponden a parámetros propios de la ejecución en tiempo real.

- `InitSRPVAD`

  Esta función inicializa los parámetros y lleva a cabo las reservas de memoria necesarias para alojar las variables que se van a precisar para realizar el cálculo de la localización del locutor basándose en el algoritmo SRP.

  De este bloque se ha eliminado la configuración de los parámetros pertenecientes al cálculo de los datos de CSP, puesto que al haberse implementado en un nuevo módulo para que tanto el módulo VAD como el de localización basada en SRP puedan disponer de dichos datos, también su inicialización se realizará de forma independiente.

- `InitVADCSP`

  Se encarga de inicializar los parámetros y reserva memoria para las diferentes variables de las que se hará uso para realizar los cálculos correspondientes a la correlación entre las señales de audio de los diferentes canales. También se crearán los planes que se ocuparán de las transformadas y transformadas inversas de Fourier.

- `InitGrafo`

  Tiene la función de inicialización y configuración de los parámetros y constantes que se utilizarán para realizar las comparaciones en el módulo de detección de actividad de voz, tanto en el cálculo del umbral de detección como en el autómata que decide el estado del sistema.

- `initVisualSimulation`

  Es la función encargada de configurar e inicializar el módulo de visualización 3D. Desde el momento en que se llama a esta función se lanza la ventana que muestra el espacio 3D donde tendrá lugar la visualización del punto estimado donde se encuentra el locutor en caso de existir voz.

El primer paso de inicialización y configuración del sistema tiene lugar en el momento del lanzamiento de la ejecución del sistema, cuando asignamos el valor de los argumentos configurables en tiempo de ejecución. De esta forma, además de estas funciones de configuración de cada módulo, se debe incluir en el proceso de inicialización del sistema una fase previa que se encargue del tratamiento de estos argumentos, que deben ser gestionados y asignados a los parámetros internos que posteriormente utilizará el programa para su proceso de ejecución.

3.6.1 Procesamiento de los argumentos

De la tarea de tratamiento de los argumentos se ocupa la función `__ProcesaOpciones`. A esta función se le pasa como parámetro una estructura del tipo `TConfGeneral` en la que se definirán los diferentes módulos de configuración que se incluirán según sea necesario. Cada uno declara una serie de parámetros y se encarga de recoger los valores que se pasan como argumento en tiempo de ejecución y las asigna a estas estructuras y variables que serán accesibles globalmente por cada uno de los bloques que conforman el sistema. Además de la configuración general, se pasan también como parámetros de la función el número de argumentos recogido en el lanzamiento de la ejecución, todos los argumentos y una variable `ultimoArgumento` en la que se almacenará el valor de el primer argumento no reconocido en caso de error.

```
int __ProcesaOpciones (int argc, char **argv, TConfGeneral configuracion,
int *ultimoArgumento)
```

Para el sistema que se ha implementado, se ha creado un modelo de configuración que recoge los parámetros pasados como argumentos que afectan a cada uno de los módulos cuyas funcionalidades se incluyen en este sistema. Este modelo es `__allCfgRTVadSRP`, que incluye los módulos existentes necesarios para la configuración del sistema, así como un nuevo módulo que se ha creado para la gestión de los parámetros correspondientes a la nueva implementación del VAD en tiempo real. El diseño de este modelo se puede ver a continuación.

```
typedef struct TAutomata
{
  double WaitingTimeVoiceOk;
  double Umbral;
  double correlationFrameSizeSecs;
  double thresholdUpdate;

} TAutomata;
```

En esta estructura se definen los parámetros que se deberán configurar correspondientes al bloque VAD en tiempo real. Hacen referencia al tiempo que deberá mantenerse el valor de correlación de las señales por encima del umbral para que se realice una llamada al algoritmo SRP (`WaitingTimeVoiceOk`), al porcentaje que se aplicará sobre la diferencia entre el valor máximo y mínimo de correlación para obtener el valor del umbral (`Umbral`), al tamaño de la trama que se utilizará para almacenar los valores de correlación obtenidos con el fin de calcular el umbral (`correlationFrameSizeSecs`) y al periodo de actualización del umbral ante el autómata del bloque VAD (`thresholdUpdate`). Una explicación más extendida sobre la aplicación de estos parámetros se puede encontrar en el apartado 3.2, a partir de la página 21.

Listado 3.3: Configuración de los argumentos de inicialización.

```
TConfModulo *__allCfgModules[] =
{
  &__help,
  &__automata,
  &__frontEndCfgModule,
  &__dirWavesInCfgModule,
  &__dirInputFilesCfgModule,
  &__signalProcessingCfgModule,
  &__srpExtensionsCfgModule,
  &__simulationConfigFileCfgModule,
  &__soundspeedCfgModule,
  &__channelsCfgModule,
  &__dirRootDistribCfgModule,
  &__audioIOLibCfgModule,
  &__printPowLocCfgModule,
  &__generateSpeakerIDCfgModule,
  &__generateARFFCfgModule,
  &__initialTimeStampCfgModule,
  &__useSpecificInitialTimeStampCfgModule,
};

TConfGeneral __allCfgRTVadSRP =
{
  __allCfgModules,
  sizeof(__allCfgModules) / sizeof(TConfModulo *),
  DESCRIPTION,
  0
};
```

El módulo `__automata` es el encargado de asignar a las variables definidas en la estructura de tipo `TAutomata` el valor que se pasa como argumento en el momento de la ejecución. Además se incluyen una serie de módulos destinados a la caracterización del entorno que se cargará a partir de diferentes archivos que contienen la información necesaria. Estos ficheros contendrán datos sobre la posición de los micrófonos, útil para los cálculos basados en SRP, o sobre las dimensiones y elementos del espacio inteligente que se mostrará en la visualización 3D. También se han añadido módulos de configuración de la librería para tratamiento del audio `audioIOLib`, los canales de adquisición y velocidad del sonido. Para la configuración del algoritmo SRP se incluyen módulos de directividad de ondas, potencia y procesamiento de señales.

3.7 Integración de los bloques e implementación del demostrador completo

Una vez explicada en profundidad la funcionalidad de cada uno de los bloques o módulos que conforman el sistema completo en tiempo real, solo queda exponer como se integran para componer el funcionamiento deseado para el demostrador en tiempo real que se quiere implementar.

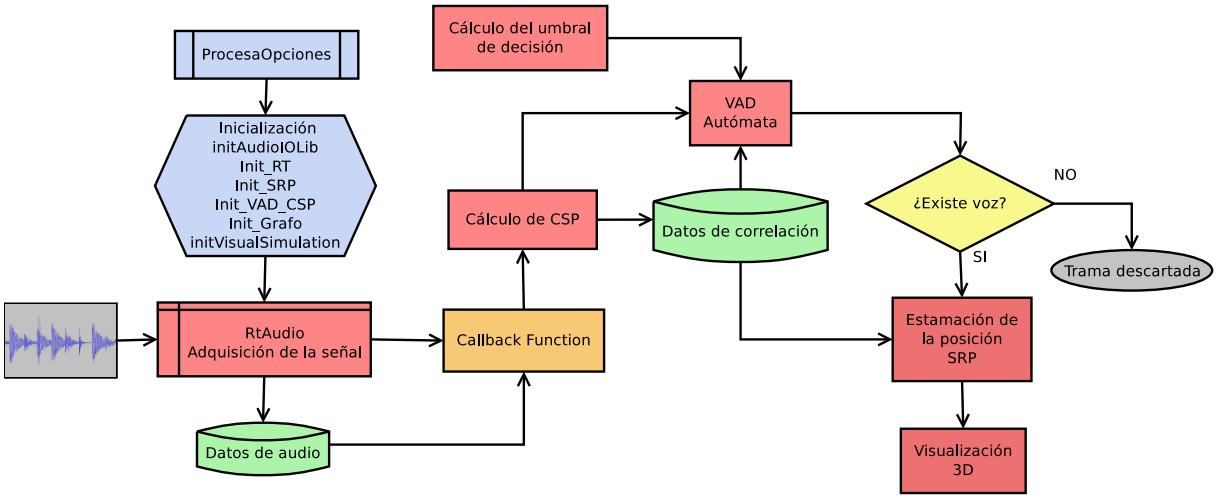


Figura 3.4: Esquema del flujo de ejecución del sistema.

Un esquema simplificado del sincronismo y el flujo de ejecución del sistema, se puede ver en la figura 3.4. En este esquema se puede observar qué bloques generan los datos y cuáles deben ejecutarse de forma secuencial para poder disponer de ellos, así como el camino que seguirá el sistema en caso de encontrarse en un estado o en otro. Se ha simplificado su estructura, ya que como se ha visto en apartados anteriores no siempre nos encontraremos ante tramas de audio en las que se pueda determinar con certeza si hay voz o no, sino que el sistema puede encontrarse en un estado intermedio de posible voz o posible silencio. Además, también se ha explicado que las funcionalidades de los bloques VAD y SRP en realidad se integran dentro de la función de callback, pero en el esquema se han incluido independientemente para poder obtener una visión mas intuitiva de que bloque se ejecuta en cada momento y cuando se producen las llamadas a cada uno.

3.8 Conclusiones

A lo largo de este capítulo se ha intentado exponer de forma clara todo el trabajo desarrollado a lo largo del proyecto. Se han expuesto las diferentes opciones de implementación para los módulos que ofrecían esta posibilidad y se ha incluido información sobre las principales decisiones adoptadas en cada uno de los módulos, explicando sus principales funcionalidades.

Se ha explicado paso a paso el funcionamiento del demostrador y el de cada uno de los bloques que se han implementado para conseguir su ejecución en tiempo real. También se ha aportado información sobre la integración de todos ellos para desarrollar el sistema completo, así como del sincronismo entre sus procesos y del flujo de ejecución del sistema.

Capítulo 4

Evaluación y resultados

4.1 Introducción

En el presente apartado se explicarán las diferentes pruebas prácticas que se han realizado sobre el sistema de localización de un locutor en tiempo real. Estos experimentos pretenden demostrar el correcto funcionamiento del sistema comparando escenarios en tiempo real con los resultados que se obtendrían en una ejecución offline. También se mostrará su flexibilidad antes diferentes configuraciones y su eficiencia. Finalmente se expondrán los resultados obtenidos y se determinará la efectividad del diseño.

4.2 Cálculos de correlación de la señal

El primer módulo que se ha sometido a prueba es el de obtención de los datos de correlación a partir de las señales de audio obtenidas de todos los canales configurados para la adquisición, ya que el resto de los bloques requieren de ellos para poder realizar los cálculos pertinentes. De la fiabilidad de estos datos, dependerá el correcto funcionamiento y eficiencia del sistema completo.

Para comprobar el correcto funcionamiento de este módulo en tiempo real se ha ejecutado el sistema guardando los datos obtenidos en cada nueva llamada a la función de callback. De esta forma, se van a generar tres tipos de ficheros: los correspondientes a las señales de audio de cada uno de los canales que han intervenido en la adquisición en el momento de la ejecución del sistema, otro en el que se almacenan los valores de correlación obtenidos para cada par de micrófonos realizando todas las combinaciones posibles dentro de los micrófonos del mismo array y, por último, un fichero en el que se almacenan los datos correspondientes al valor máximo de correlación de entre los valores ofrecidos por cada uno de los pares en cada momento.

Una vez generados estos ficheros de resultados, se han comparado con los que nos ofrece la implementación del VAD offline realizada por Ruben Peral [2] y se han generado varias gráficas en las que se puede observar tanto la señal de audio sobre la que se ha realizado el experimento, los valores de correlación obtenidos para un determinado par de micrófonos y una representación 3D que relaciona el retardo entre las señales, la frecuencia y la intensidad.

En las gráficas que se muestran en la figura 4.1 se puede encontrar una comparativa de los valores máximos de correlación, extraídos de entre todos los valores de correlación correspondientes a todas las posibles combinaciones de pares de micrófonos, entre la ejecución sobre ficheros de audio grabado que realiza el proyecto original y ejecución que adquiere las señales de audio en tiempo real que realiza este

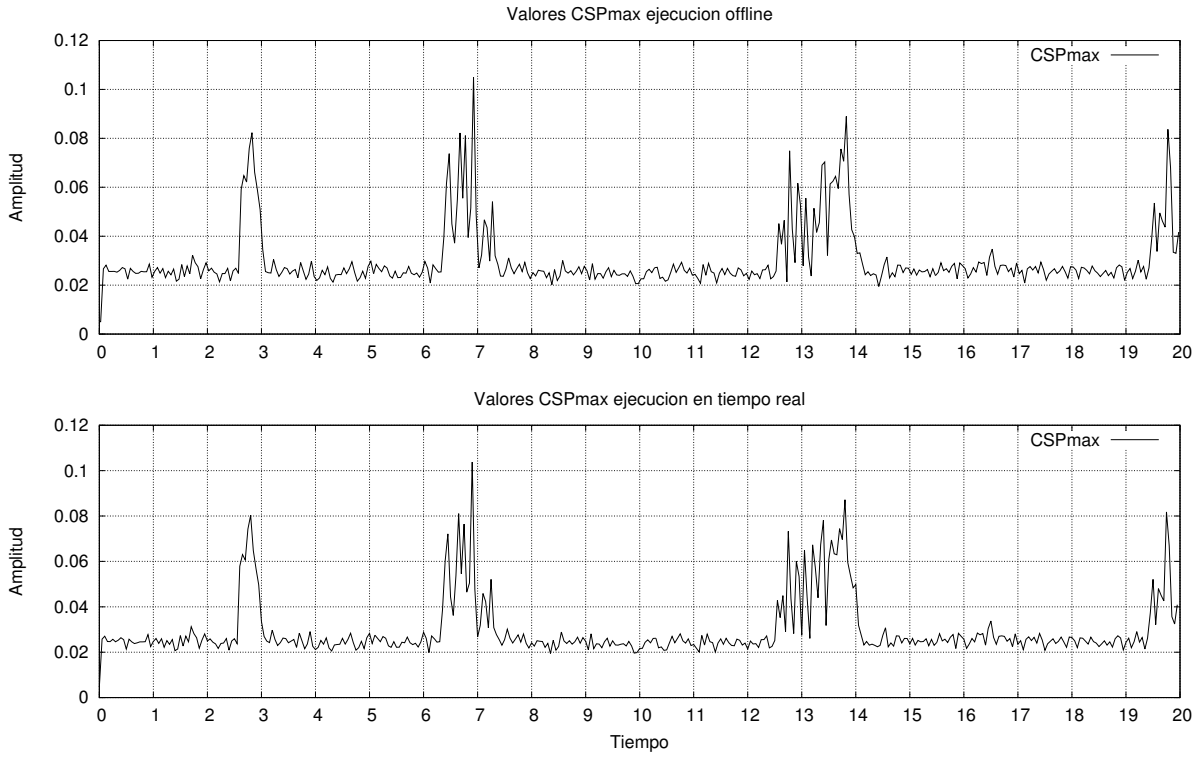


Figura 4.1: Comparación de los valores de CSPmax offline y en tiempo real.

proyecto. Se puede observar que existe una gran similitud entre los valores de las diferentes gráficas que se están comparando.

Dos parámetros que se pueden configurar y que debemos tener en cuenta a la hora de tratar las señales de audio y calcular la correlación cruzada entre ellas, son `frameSize` y `frameShift`. En las pruebas realizadas a este sistema se han obtenido resultados idénticos entre la implementación offline y en tiempo real para un `frameSize` = `frameShift`, por lo que será la premisa que se tendrá en cuenta en el resto de experimentos que se van a realizar.

En las figuras 4.2 y 4.3 se observan las gráficas comentadas anteriormente obtenidas en dos escenarios diferentes y comparando los resultados con dos tamaños de trama diferentes. La figura 4.3 corresponde a un escenario en el que el entorno presenta un nivel mayor de ruido ambiente, se pueden observar unas mayores variaciones de correlación en momentos de silencio que las que se presentan en el caso de la figura 4.2 que se ha obtenido en un entorno en el que se ha intentado disminuir el ruido ambiente.

4.3 Método de cálculo del umbral de decisión

Como se ha explicado en el apartado 3.2.2, se ha implementado un módulo de cálculo del umbral de decisión que pretende realizar una actualización periódica de dicho valor de forma que sea robusto ante posibles cambios en el entorno. Sin embargo, las pruebas que se han realizado sobre el sistema han determinado que este diseño no ofrece buenos resultados en la ejecución en tiempo real del sistema. Teóricamente, se presenta un diseño factible, en cambio, esta actualización periódica puede exponer al sistema a cambios muy bruscos en el valor del umbral de detección. Esto puede ocurrir en dos escenarios diferentes:

- Si el sistema se expone a largos periodos de silencio. En este caso, el umbral irá disminuyendo

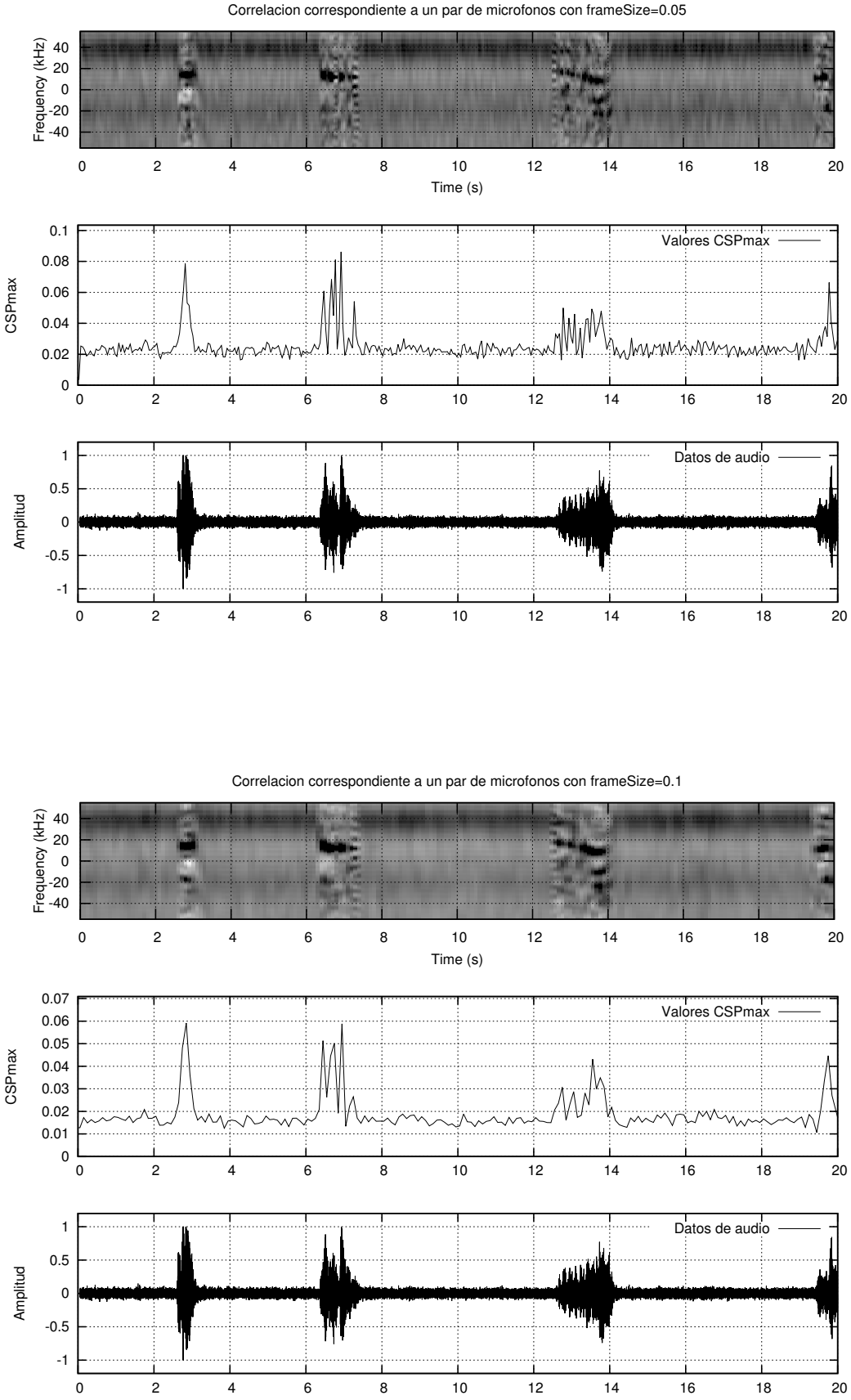


Figura 4.2: Cálculo de la correlación de dos señales.

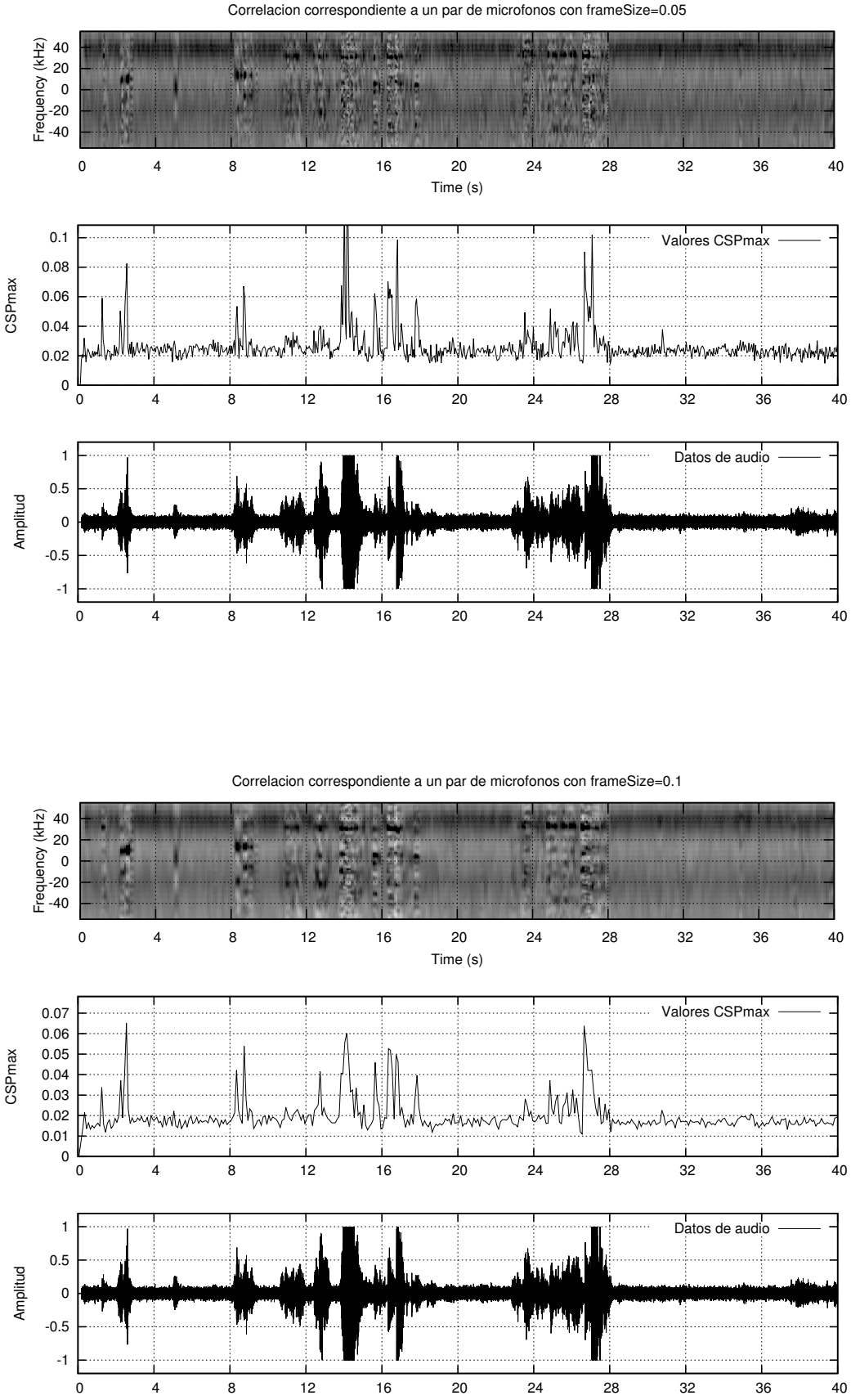


Figura 4.3: Cálculo de la correlación de dos señales.

hasta alcanzar valores tan bajos como los valores de correlación que presente el entorno en silencio. Esto supondrá que cualquier mínima variación en estos valores superará el umbral haciendo que el sistema pase a un estado de posible voz cuando en realidad no debería.

- Si el sistema se expone a largos periodos de voz. En este caso, el umbral tomará un valor cada vez mayor, ya que se va guardando una historia de los últimos valores de correlación sobre los que se aplicará el porcentaje de umbral configurado. A consecuencia de ello, solo los valores mas altos de correlación superarán el nivel del umbral presentando una posibilidad de existencia de voz y posibles escenarios de voz que presenten unos valores de correlación relativamente mas bajos no llegarán a superar el umbral y serán considerados silencio. Esto aumenta el número de falsos negativos descartando tramas que realmente si que contenían voz.

Para solventar este problema, se han tomado los ficheros generados por el sistema para la comprobación de los valores de correlación y se ha realizado una lectura de dichos datos para obtener un nivel de umbral de detección que sea constante y coherente con los valores que nos ofrece el entorno en cada momento.

Aunque, de esta forma, se evitan la mayoría de los problemas a los que estaba expuesto el sistema haciendo uso del módulo de actualización del umbral, el valor de este si que sigue viéndose afectado por posibles cambios en el entorno como pueden ser mayores niveles de ruido ambiente. En la figura 4.4 se puede observar la diferencia en el valor que toma el umbral en dos escenarios con diferentes niveles de ruido en instantes de silencio, tomando un valor de 0.02958 en el caso de un entorno mas silencioso y de 0.038426 cuando esta expuesto a un mayor nivel de ruido ambiente.

Otro parámetro configurable que se ha sometido a estudio es el porcentaje de umbral que se aplica sobre la diferencia entre los valores máximo y mínimo de correlación entre las señales para obtener el valor final del umbral. Según la experiencia de Rubén Peral en su proyecto se exponía que un porcentaje del 8 % generaba unos resultados coherentes para las bases de datos que se utilizaron en su evaluación. Por ello, se ha partido de este dato y se han evaluado diferentes valores cercanos al 8 % para estudiar sus resultados. Así se han obtenido las gráficas mostradas en las figuras 4.4, 4.5 y 4.6 en las que podemos ver la evolución del sistema para unos valores de umbral del 12 %, 10 % y 8 %, respectivamente.

En dichas figuras se puede observar que un porcentaje de umbral del 8 % es demasiado bajo y se obtienen intervalos considerados como voz que realmente solo son variaciones en los datos de correlación correspondientes a instantes de silencio. Los valores de 10 % y 12 % no presentan diferencias considerables, generando resultados bastante fiables.

A consecuencia de los resultados obtenidos se ha adoptado un valor de umbral del 12 % para el resto de experimentos que se realizarán para evaluar el sistema.

4.4 Autómata para la detección de voz

En el módulo VAD se ha evaluado la configuración previa del sistema, los parámetros correspondientes a los tiempos que deben transcurrir para determinar la existencia de voz o de silencio. También se someterá a estudio la transición entre los diferentes estados en los que se puede encontrar el sistema en cada momento.

4.4.1 Comparación del detector de actividad de voz offline y en tiempo real

En los experimentos que se realizaron para evaluar el funcionamiento del VAD desarrollado en el proyecto original, se obtuvieron como resultado los datos correspondientes a cada uno de los parámetros y variables

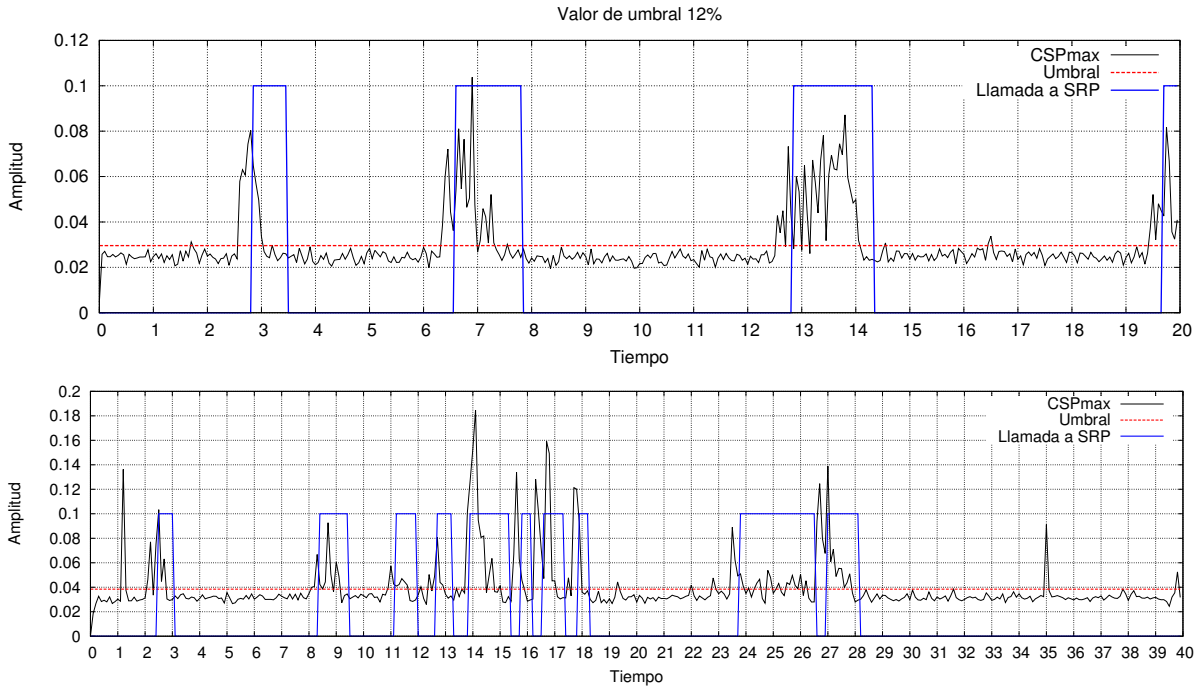


Figura 4.4: Resultado del VAD con un umbral del 12 %.

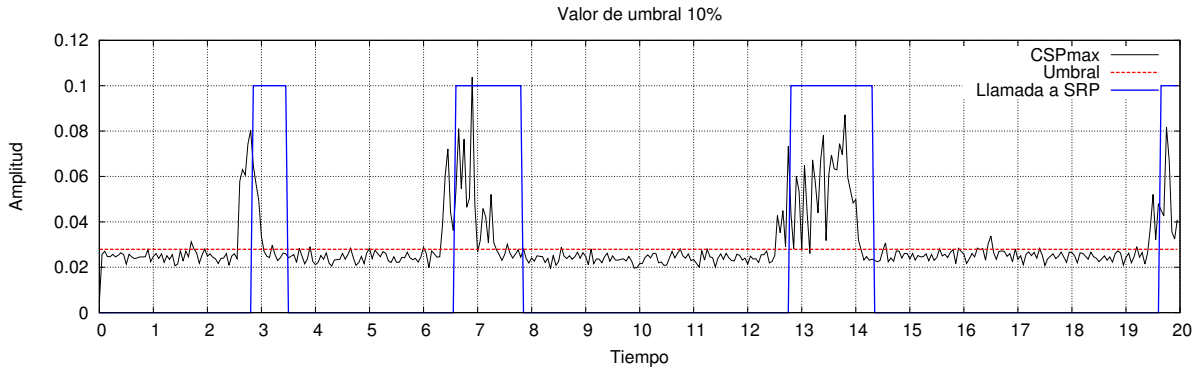


Figura 4.5: Resultado del VAD con un umbral del 10 %.

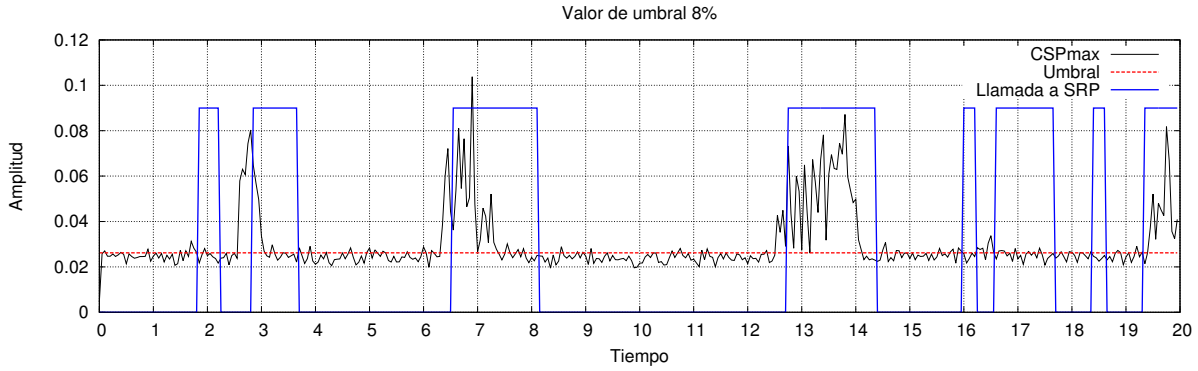


Figura 4.6: Resultado del VAD con un umbral del 8 %.

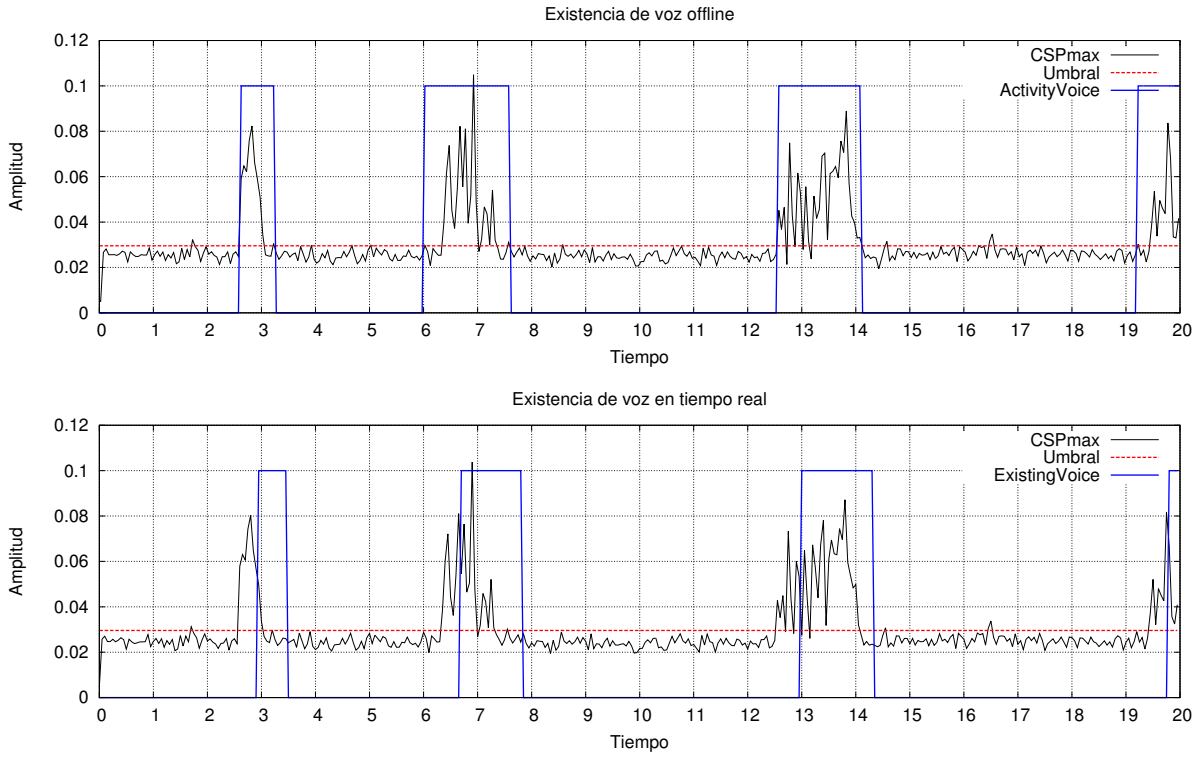


Figura 4.7: Comparación de los sistemas VAD offline y en tiempo real.

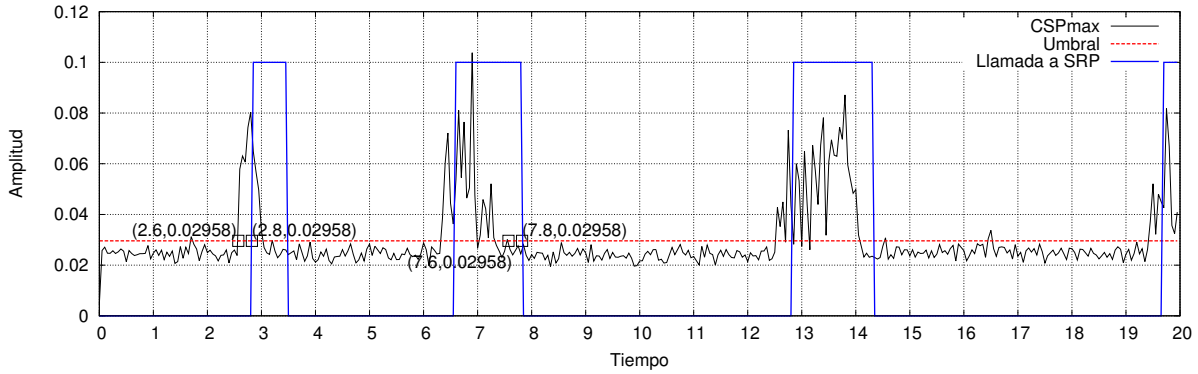
que se utilizan para la implementación del autómata de decisión. En ellos se hace un barrido que evalúa los resultados para diferentes niveles del valor del umbral. De esta forma, para realizar una comparación precisa de los resultados obtenidos en el sistema en tiempo real desarrollado y los resultados obtenidos en el sistema offline implementado en el proyecto original, se han extraído los datos correspondientes al nivel de umbral que coincide con el que se ha configurado en este sistema.

En las gráficas que se muestran en la figura 4.7, se pueden observar los resultados obtenidos para cada una de estas ejecuciones. En el caso en el que el sistema trabaja sobre ficheros de audio, se dispone de los datos de correlación obtenidos al completo en todo momento, esto facilita que la determinación de la existencia de voz o no se ajuste perfectamente a los instantes de voz. El sistema también requiere de los tiempos de espera necesarios para llevar a cabo la decisión `TimeVoiceOk` y `TimeDown`, sin embargo, una vez se superan estos tiempos y se obtiene una decisión con suficiente certeza, se realiza una rectificación de los datos tratados con anterioridad para ajustarlos al escenario real.

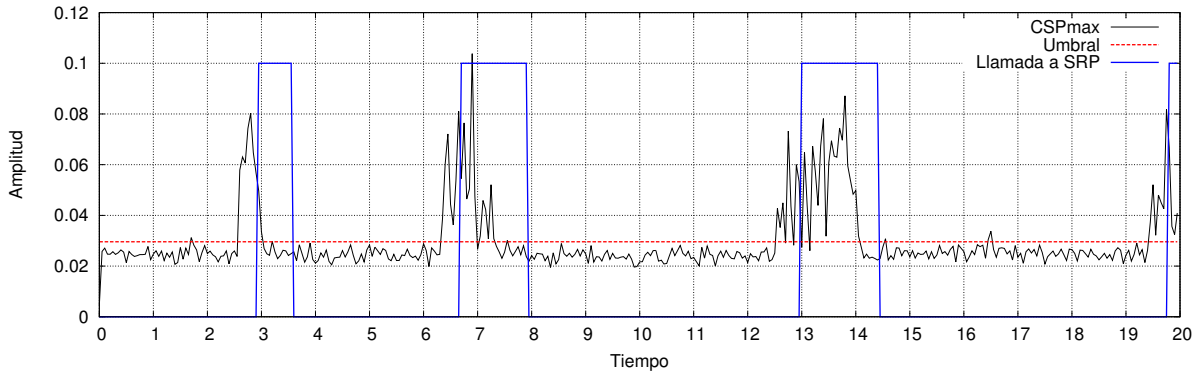
En el caso de la ejecución en tiempo real, no es posible rectificar la decisión tomada para datos obtenidos en instantes anteriores, de forma que el sistema sufrirá pequeños retardos que corresponden a los valores fijados para `TimeVoiceOk` y `TimeDown`, que supondrán un retardo en la detección de voz y un retardo en la detección de silencio, respectivamente.

4.4.2 Configuración de los parámetros

En el proceso de inicialización y configuración del autómata para la detección de voz, se definen varios parámetros que determinarán el momento exacto en el que el sistema experimentará un cambio de estado. Estos parámetros son `WaitingTimeVoiceOk`, `TimeVoiceOk` y `TimeDown` y se puede ver una descripción más detallada de su aplicación y función en el sistema en el apartado 3.2.1. Los valores de estos parámetros serán determinantes en la ejecución del sistema, puesto que introducen un retardo en



(a) WaitingTimeVoiceOk = 0.2 y TimeDown = 0.2.



(b) WaitingTimeVoiceOk = 0.3 y TimeDown = 0.3.

Figura 4.8: Análisis de los tiempos establecidos para WaitingTimeVoiceOk y TimeDown.

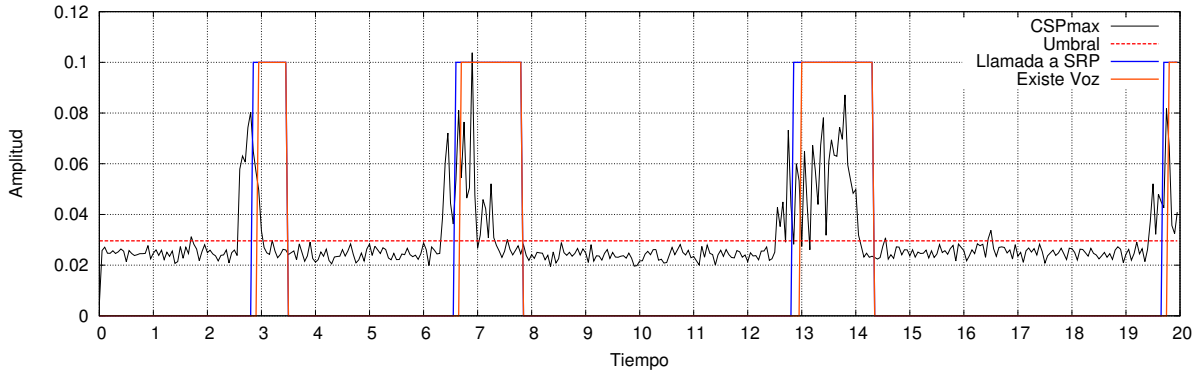
la detección de voz y silencio para poder asegurar su existencia.

El parámetro `WaitingTimeVoiceOk` supone un retardo entre el momento en que el valor de correlación de las señales obtenidas supera el umbral y el momento en el que se realiza la llamada al algoritmo SRP. Se ha creado para que el sistema pueda anticiparse a posibles momentos de voz que se verificarán cuando haya transcurrido un tiempo igual o superior a `TimeVoiceOk` y así ceñirse en la medida de lo posible a una ejecución en tiempo real. A su vez, el parámetro `TimeDown` introducirá un cierto retardo desde el momento en que los valores de correlación quedan por debajo del umbral de detección hasta que se puede asegurar que cesó la existencia de voz. A pesar de ello, en un sistema que debe responder en tiempo real, será mas crítica la pérdida de tramas que contengan voz y no sean detectadas como tal, ante la estimación y la visualización de la localización de ciertas tramas que en realidad correspondan a instantes de silencio.

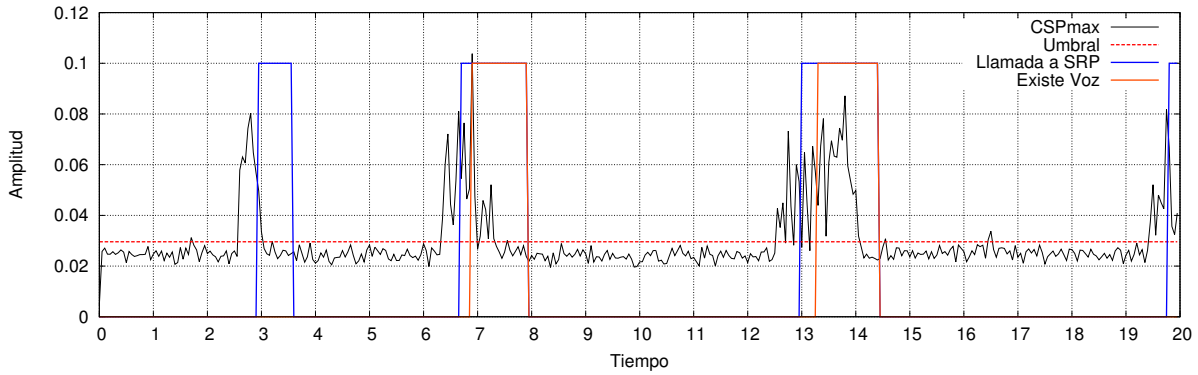
Inicialmente se configura el sistema con los valores 0.5s para `TimeVoiceOk` y 0.3s para `WaitingTimeVoiceOk` y `TimeDown`, ya que fueron los que se propusieron en el proyecto de Ruben Peral [2] por generar buenos resultados. Para evaluar su eficiencia se han asignado también los valores 0.3s y 0.2s, respectivamente, para comparar los resultados obtenidos si se ajustan estos parámetros.

De esta forma, se han obtenido las gráficas mostradas en la figura 4.8 que comparan los diferentes valores establecidos para `WaitingTimeVoiceOk` y `TimeDown`. Se puede observar que ajustando estos valores se puede obtener una respuesta mas rápida del sistema que se adapte en mayor medida a la situación real que se da en el entorno.

Por otro lado, se han obtenido las diferentes respuestas del sistema ante las distintas configuraciones de `WaitingTimeVoiceOk` y `TimeVoiceOk`. La diferencia entre estos dos valores supondrá el tiempo



(a) WaitingTimeVoiceOk = 0.2 y TimeVoiceOk = 0.3.



(b) WaitingTimeVoiceOk = 0.3 y TimeVoiceOk = 0.5.

Figura 4.9: Análisis de los tiempos WaitingTimeVoiceOk y TimeVoiceOk.

transcurrido desde una llamada a SRP anticipada y el momento en el que el sistema determina que realmente se encuentra en un entorno que contiene voz. En este caso, también se han extraído mejores resultados cuando los valores son relativamente menores, ya que, como se observa en la figura 4.9, en intervalos de voz de poca duración puede ser que el sistema pase a un estado de posible fin de voz antes de que se haya considerado que realmente existía voz.

4.4.3 Flujo de estados del sistema

En este apartado se van a evaluar el correcto funcionamiento del autómata de detección de voz, comprobando que el sistema experimente las transiciones entre los posibles estados de forma coherente y ajustándose tanto a los parámetros y valores generados por el sistema, como a la situación real a la que esta sometido el sistema.

Para el estudio de los resultados obtenidos en estos experimentos, se ha inicializado y configurado el bloque VAD con los valores que mejor respuesta han ofrecido según los datos obtenidos en pruebas anteriores. Atendiendo a esto, los valores que se han fijado para estos parámetros serán 0.3s para `TimeVoiceOk` y 0.2s para `WaitingTimeVoiceOk` y `TimeDown`.

En la figura 4.10 se muestra la evolución del estado del sistema en cada instante. La línea que representa el estado del sistema tomará un valor 0 si se encuentra en estado de silencio o espera a la aparición de voz, el valor 0.6 determina que se ha producido una transición al estado de posible voz y el valor 0.8 supone una transición del sistema al estado de posible fin. Se puede observar que las transiciones se producen en los momentos en los que el valor de correlaciones de las señales obtenidas experimenta un cambio en sentido ascendente o descendente con respecto al valor del umbral. En el apartado 3.2.1, a partir de la

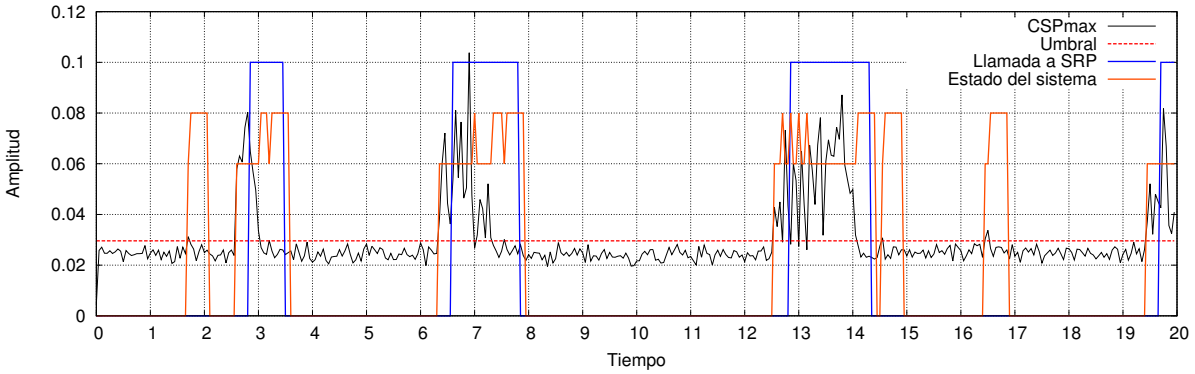


Figura 4.10: Análisis del estado del sistema en cada instante.

página 21, se expone una descripción detallada de qué supone la situación del sistema en cada uno de los estados.

Además de obtener los resultados sobre la gráfica para ofrecer una vía mas visual e intuitiva de reconocimiento de los cambios de estado del sistema, también se han extraído los valores de cada una de las variables de las que dispone el autómata para realizar las comparaciones y determinar las transiciones entre estados. Estos resultados se han evaluado sobre uno de los periodos de voz que aparecen en la secuencia de audio con la que se esta atacando al sistema para realizar su evaluación.

Le evolución del valor de estas variables y parámetros se pueden encontrar en la tabla 4.1, en la que se han marcado las transiciones mas importantes y los momentos en los que se alcanzan los valores temporales que determinan una llamada a SRP y la existencia de voz o silencio.

4.5 Módulo de localización basado en SRP

La funcionalidad del módulo de estimación de la posición del locutor basado en el algoritmo SRP se ha incluido en este proyecto. Jesús Martinez ya incluyó en su proyecto de implementación de un sistema de adquisición de audio multicanal [1], la funcionalidad del sistema de localización en tiempo real que desarrolló en primera instancia Carlos Castro en su proyecto de estudio de las técnicas de localización de hablantes en entornos reverberantes [3]. Partiendo de sus implementaciones se ha añadido en este sistema el algoritmo de localización integrándolo con el resto de bloques del sistema, de modo que no haya redundancias en cuanto a los cálculos de correlación de las señales. Sin embargo, es preciso someterlo a pruebas exhaustivas y realizar una evaluación en profundidad para conseguir obtener unos resultados que en su conjunto sean fieles a la eficiencia y robustez que se busca para este proyecto.

4.6 Conclusiones

En este capítulo se han expuesto los diferentes experimentos realizados para evaluar el correcto funcionamiento del demostrador de localización de fuentes de audio en tiempo real que se ha implementado.

Para ello, se han comparado los resultados obtenidos con los que se exponen en los proyectos originales de los que se ha partido para realizar el desarrollo de este sistema. También se ha estudiado el comportamiento del sistema, buscando una correspondencia del estado que presenta en cada momento con el que se cabría esperar atendiendo a la situación que se este dando en el entorno real del que se obtienen las características.

Siguiente estado	longVoz(s)	longSilencio(s)	Valor CSPmax	Umbral	callSRP	ExistingVoice
0	0.000000	0.000000	0.020882	0.029580	0	0
0	0.000000	0.000000	0.024924	0.029580	0	0
0	0.000000	0.00000	0.025955	0.029580	0	0
0	0.000000	0.000000	0.023780	0.029580	0	0
0	0.000000	0.000000	0.057875	0.029580	0	0
1	0.050000	0.000000	0.063036	0.029580	0	0
1	0.100000	0.000000	0.060641	0.029580	0	0
1	0.150000	0.000000	0.074287	0.029580	0	0
1	0.200000	0.000000	0.080317	0.029580	0	0
1	0.250000	0.000000	0.064790	0.029580	1	0
1	0.300000	0.000000	0.056702	0.029580	1	0
1	0.350000	0.000000	0.049693	0.029580	1	1
1	0.400000	0.000000	0.033547	0.029580	1	1
1	0.450000	0.000000	0.026935	0.029580	1	1
2	0.450000	0.050000	0.024848	0.029580	1	1
2	0.450000	0.100000	0.024250	0.029580	1	1
2	0.450000	0.150000	0.029722	0.029580	1	1
1	0.500000	0.000000	0.025593	0.029580	1	1
2	0.500000	0.050000	0.022936	0.029580	1	1
2	0.500000	0.100000	0.024233	0.029580	1	1
2	0.500000	0.150000	0.026228	0.029580	1	1
2	0.500000	0.200000	0.025981	0.029580	1	1
2	0.000000	0.000000	0.024170	0.029580	0	0
0	0.000000	0.000000	0.024904	0.029580	0	0
0	0.000000	0.000000	0.025621	0.029580	0	0
0	0.000000	0.000000	0.022127	0.029580	0	0

Tabla 4.1: Evolución de los estados del sistema.

El cálculo de los datos correspondientes a la correlación cruzada entre cada una de las señales de audio adquiridas de los arrays de micrófonos, es una de las tareas más importantes que realiza el sistema. De la fiabilidad y validez de estos datos dependerá el correcto funcionamiento y eficiencia del sistema completo, ya que tanto el bloque VAD como el de localización basada en el algoritmo SRP requieren de estos datos para poder ejecutar su algorítmica y realizar los cálculos pertinentes en cada uno de ellos. En los resultados obtenidos se ha observado que los datos de correlación generados son muy similares a los que se obtienen a partir de la ejecución sobre ficheros de audio y además se corresponden con los valores que se cabría esperar, presentando máximos en los fragmentos de señal que presentan mayor similitud y en los desplazamientos temporales que corresponden con el retardo entre las señales de un par de micrófonos.

En cuanto al cálculo del umbral que utilizará el autómata para tomar sus decisiones, tras haber realizado diferentes pruebas, se ha llegado a la determinación de que no es factible el método de actualización periódica que se diseñó para el desarrollo de este módulo. Además se ha adaptado su nivel entre el valor máximo y mínimo de correlación a un 12 % en lugar del 8 % propuesto en el proyecto original, ya que genera unos resultados más fiables. Esto puede deberse a las variaciones de los valores de correlación que se presentan a causa de los diferentes niveles de ruido ambiente que se encuentran tanto en el escenario en tiempo real del que se han extraído las señales de audio para este proyecto, como en las diferentes bases de datos utilizadas en los experimentos realizados para la evaluación del VAD implementado en el proyecto original.

Para la evaluación del correcto funcionamiento del autómata de detección de actividad de voz, se

ha sometido a estudio el estado en el que se encuentra el sistema en cada momento y las transiciones entre dichos estados. Además se han comparado los resultados obtenidos con los generados por el VAD implementado sobre ficheros de audio grabado, llegando a la conclusión de que éstos presentan unas características que son las esperadas debido a la propia implementación del sistema, como pueden ser los retardos introducidos en la detección de voz.

En cuanto al bloque de estimación de la posición de la fuente sonora en el entorno, sin embargo, debe someterse aún a pruebas exhaustivas para poder determinar que cumple con los objetivos de flexibilidad, fiabilidad y eficiencia marcados para este sistema.

Capítulo 5

Conclusiones y líneas futuras

5.1 Introducción

A continuación se exponen las conclusiones más relevantes que se pueden extraer de la realización del presente proyecto y de los resultados obtenidos en la evaluación del sistema.

Además se mostrarán líneas de trabajo futuras que han surgido en la consecución de las fases del proyecto y dan valor a aquellas tareas de interés que se consideran importantes utilizando el presente trabajo.

5.2 Conclusiones

Este trabajo se ha centrado en el diseño e implementación de un sistema demostrados de localización de fuentes de audio en tiempo real. Para ello se ha estudiado el funcionamiento de cada uno de los módulos que han formado parte de él, con el objetivo de integrarlos y adaptarlos en su conjunto a un sistema en tiempo real que fuera robusto y eficiente. Se ha realizado una optimización del código de programación de forma que en la integración no existan redundancias o repeticiones innecesarias.

En la implementación de este sistema se han incluido las funcionalidades propias de la librería de adquisición y reproducción de audio multicanal que se encarga de todo el proceso de adquisición del audio de los diferentes arrays de micrófonos que se incluyan en el sistema, así como de la preparación de los datos y del sincronismo de dichos procesos con los de tratamiento y procesamiento de los datos. Se ha creado un módulo de cálculo de correlación CSP entre los diferentes pares de micrófonos que almacena los datos obtenidos para que tanto el VAD como el algoritmo SRP puedan disponer posteriormente de ellos. Para la detección de actividad de voz se ha diseñado e implementado un autómata de decisión en tiempo real que es el encargado de descartar o no una determinada trama según se decida la presencia de voz o no.

Estos bloques han generado unos resultados fiables y eficientes respetando las restricciones que impone una ejecución en tiempo real. También se ha incluido el bloque de estimación de la posición de la fuente sonora en el entorno, sin embargo, debe someterse aún a pruebas exhaustivas para poder determinar que cumple con los objetivos de flexibilidad, fiabilidad y eficiencia marcados para este sistema. Se ha añadido también un módulo de visualización 3D que muestra el entorno del Ispace y marca la posición del locutor en caso de existir voz.

Tras la realización de este proyecto se han obtenido las siguientes conclusiones:

- El sistema implementado muestra una gran flexibilidad en cuanto al hardware de audio que se desee emplear para su implementación. Existe la posibilidad de añadir o eliminar canales o incluir nuevos arrays de micrófonos, siempre configurando correctamente el sistema proporcionandole todos los datos necesarios sobre el número final de micrófonos, arrays y la posición de los mismos. En el proceso de evaluación se han realizado pruebas para 4 y 8 canales, incluyendo un array de micrófonos o dos y obteniendo resultados igualmente precisos.
- Se ha respetado un sincronismo que se ajusta a una ejecución en tiempo real, reduciendo los posibles retardos con respecto a los eventos reales del entorno de alrededor de los 500ms.
- En el sistema se han implementado las diferentes funcionalidades en bloques independientes para hacerlo flexible a posibles reinterpretaciones o necesidades para sistemas futuros.

5.3 Líneas futuras

En este apartado proponemos algunas de las lineas futuras de investigación para mejorar el diseño expuesto en este proyecto. Las futuras mejoras están enfocadas tanto en la implementación como la evaluación del sistema.

- Realizar la implementación de un módulo de actualización del umbral de decisión en el sistema de detección de actividad de voz y lo haga flexible y robusto ante posibles cambios en el entorno acústico, pero evitando que sea vulnerable y propenso a cambios bruscos en el nivel de umbral que se aplicará en el autómata.
- Estudiar y optimizar el uso que el sistema hace de los recursos de memoria, evaluando y probando el correcto funcionamiento ante diferentes estrategias de optimización de procesamiento: elección del tamaño del buffer de salida, tamaño de muestra, longitud y desplazamiento de trama, etc.
- Evaluar y someter a experimentos exhaustivos el algoritmo SRP incluido en el proyecto con el fin de obtener una respuesta fiel al comportamiento del entorno en tiempo real.
- Realizar un estudio e implementación de diferentes algoritmos alternativos de localización basados en audio explotando las amplias posibilidades que ofrecen las técnicas de Beamforming y Steered Response Power.
- Proponer y evaluar algoritmos de detección de voz alternativos a la máquina de estados implementada por el autómata del módulo VAD y realizar una comparativa para obtener el rendimiento de cada uno de ellos.

Capítulo 6

Presupuesto

6.1 Costes de equipamiento

- Equipamento Hardware utilizado:

Concepto	Cantidad	Coste Unitario	Subtotal(€)
PC Dual Xeon CPU E5345	1	500€	500€
Micrófonos Shure series MX391/O	8	155€	1240€
Previos RME Octamic II	3	1280€	3840€
PCI RME Hammerfall DSP 9652	1	440€	440€
Coste total HW			6020€

Tabla 6.1: Coste equipamiento Hardware utilizado

- Recursos Software utilizados:

Concepto	Cantidad	Coste Unitario	Subtotal(€)
Ubuntu 14.04.1 LTS	1	0€	0€
Libreria rtAudio	1	0€	0€
Software LaTeX	1	0€	0€
Coste total SW			0€

Tabla 6.2: Coste equipamiento Software utilizado

6.2 Costes de mano de obra

Concepto	Cantidad	Coste Unitario	Subtotal(€)
Desarrollo SW	200	65€/hora	13000€
Mecanografiado del documento	100	15€/hora	1500€
Coste total mano de obra			14500€

Tabla 6.3: Coste debido a mano de obra

6.3 Costes total del presupuesto

Concepto	Subtotal(€)
Equipamiento Hardware	6020€
Recursos Software	0€
Mano de obra	14500€
Coste total presupuesto	20520€

Tabla 6.4: Coste total del presupuesto

El importe total del presupuesto asciende a la cantidad de: VEINTEMIL QUINIENTOS VEINTE

En Alcalá de Henares a ___ de _____ de 20___

María Paz González.

Bibliografía

[1] J. P. M. González, "Diseño, implementación de un sistema de adquisición, procesamiento y generación de audio multicanal en aplicaciones para espacios inteligentes," Master's thesis, Escuela Politécnica Superior. Universidad de Alcalá. Spain, 2014.

[2] R. P. Nuño, "Estudio, implementación y evaluación de un sistema de detección de actividad de voz en espacios inteligentes," Master's thesis, Escuela Politécnica Superior. Universidad de Alcalá. Spain, 2014.

[3] C. Castro, "Speaker localization techniques in reverberant acoustic environments," Master's thesis, School of Electrical Engineering. Royal Institute of Technology (KTH). Sweden, 2007.

[4] D. C. Pérez, "Diseño, implementación y evaluación de una interfaz de control multimodal en un espacio inteligente: control gestual," Master's thesis, Escuela Politécnica Superior. Universidad de Alcalá. Spain, 2013.

[5] "Página de la aplicación emacs," http://savannah.gnu.org/projects/emacs/ [Último acceso 6/febrero/2015].

[6] "Página de la aplicación gnuplot," http://www.gnuplot.info/ [Último acceso Julio 2013].

[7] "Página sobre fft," http://www.fftw.org [Último acceso 6/febrero/2015].

[8] "Micrófono shure mx391," http://www.shure.es/productos/microflex/mx391_mx392_mx393 [Último acceso 6/septiembre/2015].

[9] "Directividad micrófonos shure," http://www.shure.es/asistencia_descargas/contenido-educativo/microfonos/microphone_polar_patterns [Último acceso 6/septiembre/2015].

[10] C. H. Knapp and G. C. Carter, "The generalized correlation me- thod for estimation of time delay." *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-24, no. 4, pp. 320–327, August 1976.

[11] L. Armani, M. Matassoni, M. Omologo, and P. Svaizer, "Use of a csp-based voice activity detector for distant-talking asr," in *European Conference on Speech Communication and Technology*, 2003, pp. 501–504.

[12] "Página sobre opengl," https://es.wikipedia.org/wiki/OpenGL [Último acceso 5/Septiembre/2015].

[13] "Información sobre gnu/linux en wikipedia," http://es.wikipedia.org/wiki/GNU/Linux [Último acceso 6/febrero/2015].

[14] L. Lamport, *LaTeX: A Document Preparation System, 2nd edition.* Addison Wesley Professional, 1994.

[15] "Página de la aplicación octave," http://www.octave.org [Último acceso 6/febrero/2015].

[16] "Página de la aplicación cvs," http://savannah.nongnu.org/projects/cvs/ [Último acceso 6/febrero/2015].

[17] "Página de la aplicación gcc," http://savannah.gnu.org/projects/gcc/ [Último acceso 6/febrero/2015].

[18] "Página de la aplicación make," http://savannah.gnu.org/projects/make/ [Último acceso 6/febrero/2015].

Apéndice A

Herramientas y recursos

Las herramientas necesarias para la elaboración del proyecto han sido:

- PC compatible
- Sistema operativo GNU/Linux [13]
- Entorno de desarrollo Emacs [5]
- Procesador de textos LaTeX[14]
- Lenguaje de procesamiento matemático Octave [15]
- Control de versiones CVS [16]
- Compilador C/C++ gcc [17]
- Gestor de compilaciones make [18]

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR

Universidad
de Alcalá