UNIVERSIDAD DE GRANADA

TRABAJO DE FIN DE MASTER

INGENIERÍA DE TELECOMUNICACIÓN

Diseño e implementación de un sistema de posicionamiento en interiores basado en el tiempo de vuelo de señales acústicas

Autor
Mohammed Boujemaoui Boulagmoudi

Directores
Angel De La Torre Vega
José Carlos Segura Luna

Escuela Tecnica Superior de Ingenierias Informatica y de Telecomunicacion

Granada, Mayo de 2018

Diseño e implementación de un sistema de posicionamiento en interiores basado en el tiempo de vuelo de señales acústicas

Mohammed Boujemaoui Boulaghmoudi

Palabras clave: LPS, TOF, TOA, DOA, GPS...

Resumen

El objetivo principal de este proyecto es el desarrollo e implementación de una herramienta para el control y análisis de un sistema inalámbrico de posicionamiento en interiores (WLPS, Wireless Local Positioning System) basado en señales acústicas bautizado como AcousticLPS. El proceso de identificación y posicionamiento de cada uno de los dispositivos disponibles en el área de visión se realiza a través de la estimación del tiempo de llegada (TOA, *Time of Arrival*) y la dirección de llegada (DOA, *Direction of Arrival*) de las señales acústicas capturadas basándose. Conocidas las condiciones del entorno, se podría calcular la distancia recorrida por cada una de las señales y a través de técnicas de trilateración triangular la posición del dispositivo.
Se han implementado dos herramientas: QAcousticLPS y SensorMaps. QAcousticLPS implementa una interfaz de usuario para configurar y calibrar el sistema durante su instalación. La herramienta permite el control de los usuarios conectados a la red así como la extracción de información relevante sobre el estado del sistema en tiempo real. Por otro lado, SensorMaps es una aplicación para dispositivos móviles que se ejecuta en cada uno de los dispositivos conectados a la red. La herramienta realiza los procesos de estimación y visualización de la posición del dispositivo dentro del recinto. Ambas aplicaciones han sido liberadas bajo la licencia GNU 2.0 para permitir un uso libre y posibles contribuciones futuras.

En este documento se presentan cada una de las pautas realizadas para el diseño, planificación e implementación de las herramientas así como su evaluación en un sistema desarrollado en trabajos precedentes. Finalmente, se presentan algunas pruebas de campo realizadas en un recinto aislado junto a los procesos de calibración, problemas encontrados así como posibles trabajos futuros para mejorar el rendimiento del sistema.

Design and implementation of an indoor positioning system based on the time of flight of acoustic signals

Mohammed Boujemaoui Boulaghmoudi

Keywords: LPS, TOF, TOA, DOA, GPS...

Abstract

The main objective of this project is the development and implementation of a tool for the control and analysis of a Wireless Indoor Positioning System (WLPS) based on acoustic signals baptized as AcousticLPS. The identification and positioning process of each of the available devices in the vision area is done through the estimation of the TOA (Time of Arrival) and the DOA (Direction of Arrival) of the captured acoustic signals. Knowing the conditions of the environment, one could calculate the distance traveled by each of the signals and through triangular trilateration techniques the position of the device.

Two tools have been implemented: QAcousticLPS and SensorMaps. QAcousticLPS implements a user interface to configure and calibrate the system during its installation. The tool allows the control of the users connected to the network as well as the extraction of relevant information about the state of the system in real time. On the other hand, SensorMaps is an application for mobile devices that run on each of the devices connected to the network. The tool performs the processes of estimation and visualization of the position of the device inside the enclosure. Both applications have been released under the GNU 2.0 license to allow free use and possible future contributions.

This document presents each of the guidelines for the design, planning, and implementation of the tools as well as their evaluation of a system developed in previous works. Finally, some field tests performed in an isolated room are presented along with the calibration processes, problems encountered as well as possible future work to improve the performance of the system.

Yo, Mohammed Boujemaoui Boulaghmoudi, alumno de la titulación Máster en Ingeniería de Telecomunicación de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 54609738X, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Máster en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Mohammed Boujamoui Boulaghmoudi

Granada a 5 de Mayo de 2018.

D. Angel de la Torre Vega, Profesor del Área de Teoría de la Señal y Comunicaciones del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada.

D. José Carlos Segura Luna, Profesor del Área de Teoría de la Señal y Comunicaciones del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado **Diseño e implementación de un sistema de posicionamiento en interiores basado en el tiempo de vuelo de señales acústicas**, ha sido realizado bajo su supervisión por **Mohammed Boujemaoui Boulaghmoudi**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 5 de Mayo de 2018.

Los directores:

Angel de la Torre Vega **José Carlos Segura Luna**

Agradecimientos

A mi familia, por ser el apoyo principal a lo largo de mi vida, y en especial, a mis padres, por ser quienes siempre apostaron y creyeron en mí.

Especial agradecimiento a los tutores del proyecto, Ángel de la Torre Vega y José Carlos Segura Luna, por compartir todos sus conocimientos en el área de procesado de señal y ayudarme en todo el proceso de desarrollo del trabajo.

Índice general

Índice de figuras

Nomenclatura

AT	Accuracy Threshold
BC	Beacon Controller
BS	Base Controller
CPU	Central Processing Unit
CV	Computer Vision
DCT	Discrete Cosine Transform
DHT	Discrete Hilberg Transform
DOA	Direction of Arrial
DSP	Digital Signal Processing
ER	Early Reflections
FE	Feature Extraction
FFT	Fast Fourier Transform
GPS	Global Positioning System
GPU	Graphics Processing Unit
HOL	Header Only Library
HTTPS	HyperText Transfer Protocol Secure
JSON	JavaScript Object Notation
LOS	Line of Sight
LPS	Local Positioning System
LR	Late Reflections
LT	Local Tracking

LT Local Tracking

NLOS No Line of Sight

OOP Object Oriented Programming

OS Operative System

QML Qt Modeling Language

REST REpresentational State Transfer

RFID Radio Frequency IDentificacion

SDLC Software Development Life Cycle

SNR Signal-to-Noise Ratio

SO Sistema Operativo

SRTDA Super-Resolution Time-Delay Algorithm

TOA Time-Of-Arrival

TR Time of Reverberation

VR Virtuality Reality

WLPS Wireless Local Positioning System

Capítulo 1

Introducción

En los últimos siglos se han producido desplazamientos masivos de la población hacia zonas más urbanísticas y dénsamente pobladas. Estos movimientos conforman grandes aglomeraciones que han derivado en un necesidad infraestructural cada vez mayor para poder ofrecer diferentes servicios a gran escala. Nace entonces la necesidad de extraer la información necesaria para situarnos en el espacio en un entorno no conocido tanto en recintos exteriores como en interiores.

La localización de dispositivos ha sido fruto de estudio en las ultimas decadas. En entornos exteriorizados, este problema ha sido prácticamente resuelto mediante el comúnmente estandarizado sistema GPS (*Global Positioning System*) o alguno de sus variantes como el sistema GLONASS o Galileo. Estos sistemas trabajan con satélites en órbita que requieren de un lazo directo (espacio abierto) que permita la recepción de las señales transmitidas de forma adecuada. Este requisito limita su uso en interiores donde la diversidad de obstáculos así como el propio recinto dificultan la recepción adecuada de la señal de GPS.

Nace entonces una necesidad, una adaptación para sistemas de posicionamiento en interiores (LPS, Local Positioning System). En las ultimas décadas han surgido diferentes variantes basadas en redes inalámbricas (WLPS, Wireless LPS) tales como sistemas infrarrojos (IR), sistemas de identificación por radio-frecuencia (RFID, Radio Frequency IDentificacion) o sistemas basados en ultrasonidos. La mayoría de estos sistemas presenta una estructura equivalente basada en una red de nodos inalámbricos que interactúan emitiendo diferentes señales que permiten la localización de un dispositivo de forma activa o pasiva en el area de cobertura. Este esquema permite una escalabilidad sencilla y robusta. El inconveniente principal radica, generalmente, en el despliegue y calibración de la infraestructura requerida, especialmente en instalaciones en infraestructuras dedicadas al sector de servicio o de cierta implicación histórica.

Para paliar este problema se presentaron diferentes alternativas basadas en señales acústicas que permiten la tiangulación esférica de un dispositivo a través de un conjunto de focos emisores (altavoces) previamente calibrados y sincronizados [19, 22] [18] Este proyecto se centra en el desarrollo de un conjunto de herramientas que permitan la evaluación de estos sistemas haciendo uso de equipos de sonido existentes en la mayoría de infraestructuras publicas a nivel nacional. El sistema utiliza una arquitectura similar: cada una de las pistas de audio disponibles emitiran un conjunto de códigos ortogonales enmascarados en un pista de audio. Cada uno de los altavoces podrá actuar como nodo independiente y la sincronización entre los altavoces se ajustará a través de la tarjeta de sonido. Un nodo central, generalmente un ordenador, controlara la configuración así como la calibración del sistema, por lo que los costes de despliegue e infraestructura serán ínfimos.

1.1. Motivaciones

En este proyecto presentamos un nuevo conjunto de herramientas que permiten el calibración, despliegue y evaluación de un sistema de posicionamiento en interiores basado en señales acústicas bautizado como AcousticLPS. El sistema es capaz de estimar la posición de un dispositivo móvil con una precisión centimétrica y con buenas prestaciones en cuanto a escalabilidad, consumo y bajo coste. El un principio, las herramientas se desarrollan para inferir en un mejor despliegue y calibración del sistema en entornos con un sector de la población con movilidad reducida con el objetivo de facilitar la estancia y velar por su seguridad.

La idea general es la evaluación de un sistema WLPS que pudiera hacer uso de las instalaciones tecnológicas existentes en la mayoría de arquitecturas modernas del sector de servicio, tales como hospitales, hoteles o centros comerciales. De esta forma los costes de instalación serían ínfimos y su integración trivial y escalable en la mayoría de los casos. Este tipo de infraestructura disponen de un sistema de sonido con diversas fuentes sonoras distribuidos a lo largo de todo el recinto que se utilizan para emitir mensajes de información o reproducir emisoras de radio o música de ambiente. Los equipos suelen estar situados en puntos estratégicos que permitan la percepción del mensaje en todo el recinto por lo que su uso puede ser de gran interés para este proyecto.

Haciendo uso de técnicas avanzadas de tecnologías de telecomunicación así como de ingeniería acústica se podría enmascarar una señal con determinados patrones con el resto de las emitidas en cada uno de las fuentes sonoras. Los dispositivos móviles actuarían de forma pasiva capturando cada una de estas señales a través de una matriz de micrófonos con la finalidad de localizar y estimar su posición en el recinto. De esta forma, el sistema podría integrarse de forma sencilla con estos sistemas de sonido.

La idea es utilizar cada una de las pistas disponibles (5.1, 7.1 o equipos profesionales con muchas salidas) para enmascarar códigos ortogonales en las señales emitidas. Posteriormente, el dispositivo móvil podrá de-multiplexar cada una de las fuentes aplicando diferentes correladores. Como resultado se podría obtener cada una de las diferentes señales emitidas de forma independiente. Conocidas las condiciones del entorno (temperatura, humedad, etc.) se podría estimar la distancia de cada una de las fuentes sonoras a través de la estimación del tiempo de llegada (TOA, Time-Of-Arrival) para posteriormente mediante técnicas de trilateración estimar la posición real del dispositivo con cierta precisión.

1.2. Especificaciones

AcousticLPS pretende ser un sistema de bajo consumo y coste moderado. El sistema debe de ser escalable y flexible para permitir posibles mejoras futuras sin una necesidad de re-estructuración de la instalación previa. Finalmente, el sistema debiera de ser liberado bajo una licencia Open-Source para permitir futuras contribuciones por la comunidad de investigadores. Para alcanzar todos

estos objetivos se han pre-establecido diferentes requisitos:

- *Precisión centimétrica:* el sistema debe de permitir el posicionamiento de un elemento en el entorno por encima de un umbral precisión (*AT, Accuracy Threshold*) en este caso a nivel centimétrico.
- *Escalabidad*: el sistema debe de ser extensible. Debe de permitir la adhesión de nuevos nodos de forma sencilla y robusta con el fin de mejorar el rendimiento en entornos con alta densidad de usuarios.
- *Bajo consumo*: el sistema debe de reducir el consumo en la medida de lo posible especialmente en plataformas móviles con una capacidad energética limitada.
- *Multiplataforma:* las diferentes herramientas para configurar, controlar y conectarse al sistema podrán ejecutarse en diferentes sistemas operativos tanto en plataformas de escritorio como en sistemas empotrados o dispositivos móviles.
- *Software Libre:* el código fuente utilizado para cada uno de los módulos así como los diferentes esquemáticos deberán ser publicados bajo una licencia de software libre. Esto permitirá la contribución de terceras partes y una posible evolución del sistema.
- *Tiempo real:* el sistema debe de funcionar correctamente en tiempo real por lo que los tiempos de computo así como los protocolos de comunicación deberán de ser optimizados para esta finalidad.
- *Configuración & Calibrado:* el sistema debe de ser facilmente configurable y calibrable, inclusive de forma automática.
- *Extracción de resultados:* la herramienta debe de permitir exportar los resultados obtenidos a los formatos más convenientes: doc, pdf, png...

1.3. Planificación

Para un desarrollo adecuado de AcousticLPS se requieren tanto de conocimientos avanzados en acústica y sistemas de posicionamiento como en ingeniería de software. Es por tanto que se hace necesario una definición de las diferentes etapas con la finalidad de optimizar los procesos de desarrollo:

1. Investigación (*Research):* estudio del estado del arte en lo que respecta a sistemas de posicionamiento en interiores. La idea principal de esta fase es el descubrimiento de nuevos patrones y tecnologías en el desarrollo de sistemas LPS basados especialmente en señales acústicas y definir el sistema en el cuál se basara la solución final.

2. Diseño (*Prototyping):* una vez analizadas las posibles implementaciones se procede con el prototipado del sistema. En este caso se podrán hacer pruebas de campo mediante simuladores para ver el rendimiento del sistema planteado previo a la implementación del mismo.

3. Implementación (*Development):* implementación y desarrollo del sistema. En esta etapa se desarrollará el sistema planteado. Se deberá de definir el entorno de desarrollo, los elementos a utilizar así como las pruebas de campo específicas para la validación del mismo.

4. Validación (*Testing):* en esta etapa se procede con la validación del sistema. Se realizarán las diferentes pruebas de campo para estimar el rendimiento del sistema así como la robustez del mismo.

5. Despliegue y mantenimiento (*Release):* una vez desarrollado se procederá con la documentación y explotación del mismo analizando los posibles nichos de mercado así como las diversas aplicaciones.

Parte I

Metodología

Capítulo 2

Propagación del sonido en entornos cerrados

En física clásica, la alteración mecánica de las partículas en un medio elástico pueden producir una sensación auditiva. Esta alteración produce unas vibraciones que son propagadas a través de diferentes medios, generalmente el aire, en forma de *ondas sonoras.* Estas ultimas provocan cambios en la presión atmosférica (diferencia entre la presión instantánea y la presión atmosférica) que se traduce en una excitación de una onda de presión de propagación conocida como *presión acústica o presión sonora.* El estudio de estas ondas es vital para el correcto diseño del sistema ya que conforman el núcleo principal del sistema. En este capítulo se discuten algunos de los efectos del entorno que se tienen que considerar a la hora de diseñar e implementar el sistema.

2.1. Frente de onda directo

El sonido percibido directo lo conforma el frente de ondas que se propaga de forma directa desde el foco emisor hacia el receptor sin verse alterado por los distintos elementos del entorno. El *sonido directo* en espacio abierto sin guía de onda presenta una atenuación de 6dB por cada vez que se dobla la distancia. Este atenuación incrementa en ambientes con un alto factor de absorción.

En la práctica existen diferentes elementos con factores de absorción y características variadas. Esto hace que en la recepción se capten tanto reflexiones cercanas (*ER, Early Reflections*) como lejanas (*LR, Late Reflections).*

Las reflexiones cercanas por regla general no presentan un gran problema en el sistema ya que pueden actuar intensificando la señal recibida. Las reflexiones con grandes retardosson percibidas como ecos que distorsionan la señal original, Figura 2.1.1. La distorsión puede afectar drasticamente al sistema ya que pueden darse situaciones en las que las que la contribución energética de las reflexiones retardadas sea mayor que la de la señal de visión directa.

Esto se traduce en la salida de los distintos correladores utilizados para la estimación de la TOA. En condiciones ideales, el pico de la correlación cruzada entre la señal enmascarada y el codigo ortogonal coincidirá con la TOA. Cuando la influencia de las señales retardadas incrementa pueden darse situaciones en las que el pico principal se corresponda con el de una señal retardada de la señal original introduciendo un error en la estimación de la posición final.

2.2. Reverberacion

La reverberación es un fenómeno acústico producido por las distintas reflexiones que se producen en un recinto cuando los frentes de onda inciden contra los distintos elementos del entorno conformando un *campo reverberante.* En situaciones no controladas produce una *persistencia acústica* que puede llegar a modificar las características originales de la onda sonora. Para medir los niveles de reverberación de un entorno se utiliza el *tiempo de reverberación* (TR_x) que puede considerarse como el tiempo transcurrido desde que la fuente excitadora interrumpe la emisión hasta que desciende x dBs del valor inicial. Por

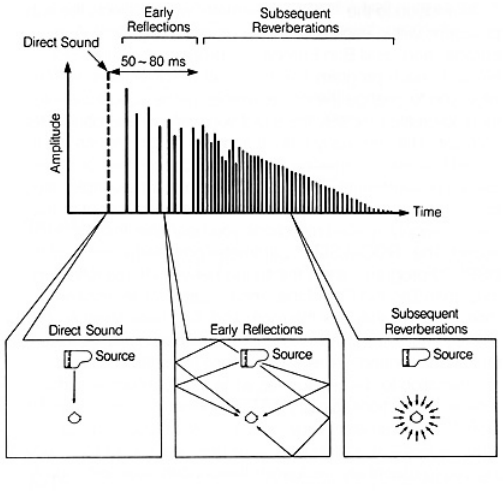

Figura 2.1.1: Respuesta impulsiva de una onda sonora en una habitación

convencion se suele utilizar $x = 60$ [16], Figura 2.2.1.

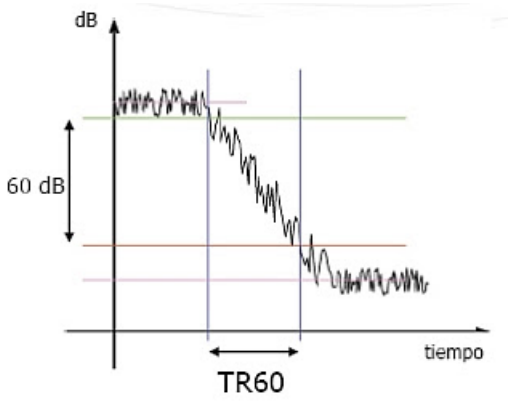

Figura 2.2.1: Estimación del tiempo de reverberación

La estimación de este parámetro es de vital importancia ya que permitirá un mejor ajuste del sistema para paliar los distintos efectos derivados del mismo. En condiciones óptimas donde todas las superficies del entorno absorben y reflejan la misma fracción de sonido, el tiempo de reverberación es proporcional a la relación entre la superficie y el volumen del entorno. En la práctica cada uno de los medios presentan unas propiedades diferentes por lo que la estimación correcta es compleja y requiere de aparatos electrónicos dedicados o técnicas avanzadas de procesamiento de señal.

2.2.1. Coeficiente de absorción sonora (α)

En entornos reales coexisten distintos medios con diferentes características, es por ello que se deben de analizar cada una de las regiones de forma independiente. Cuando una onda sonora interfiere con el área de un obstáculo una fracción de la onda es absorbida por el medio y parte es reflejada por el mismo. La cantidad reflejada o absorbida dependerá de las características del medio así como de la frecuencia de la onda incidente (f) . Podrán coexistir entonces diferentes situaciones:

1. Imagen Especular ($f \ll x$) : se produce una reflexión imagen semejante a la de la onda incidente con un ángulo igual al de la incidencia y dependiente de la rugosidad de la superficie especular.
2. Ningún efecto ($f \gg x$) : en estas condiciones no se produce efecto alguno.
3. Difracción ($f \sim x$) : es una forma particular de interferencia en la que se produce una desviación de la onda original.

Para la parametrización de estos efectos se utiliza el coeficiente de absorción sonora (α) que relaciona la cantidad de potencia absorbida por un material con la reflejada por el mismo; es dependiente de la estructura del propio material así como de la frecuencia de la onda incidente. Analíticamente se puede obtener como:

$$\alpha \quad = \quad 1 - |R|^2 \quad \alpha \in [0,1] \tag{2.2.1}$$

- Si $\alpha = 1$: toda la energía incidente es absorbida por el propio material.
- Si $\alpha = 0$: toda la energía incidente es reflejada.

En la práctica dispondremos de distintas superficies (S_i) cada una de ellas con su coeficiente de absorción asociado (α_i). Es necesario entonces la estimación de una *zona total de absorción* (S_e) que engloba la suma de todas las áreas de absorción independientes de la habitación:

$$S_e = \sum_{i=0}^{N} \alpha_i S_i$$

Por consiguiente, el tiempo de reverberación usando la fórmula de Wallace Sabine se puede calcular como:

$$RT_{60} = 0{,}161 \frac{V}{S_e} \tag{2.2.2}$$

2.2.2. Coeficiente de absorción del aire (m)

La fórmula genérica de Wallace Sabine, ecuación (2.2.2), no presenta buenos resultados en recintos extensos debido a las perdidas de absorción del medio de propagación. El tiempo de reverberación también se ve afectado por el propio medio de propagación ya que la temperatura, la humedad o la frecuencia propagación alteran los valores promedio. Estos efectos son parametrizados empíricamente mediante el uso del coeficiente de absorción del medio (m) **??**. La ecuación (2.2.3) propone una forma más ajustable para la estimación de los tiempos de reverberación considerando todos estos factores.

$$RT_{60} = 0{,}161\frac{V}{S_e + m{\cdot}V} \tag{2.2.3}$$

2.3. Efecto Doppler

El efecto Doppler [2] se identifica como un cambio de frecuencia aparente en una onda determinada debido a un movimiento relativo de la fuente sonora con respecto al observador. La relación entre la frecuencia observada y la frecuencia emitida dependerán de las velocidades tanto del foco emisor (v_x) como del observador (v_s) con respecto al medio, analíticamente se puede obtener como:

$$f = \frac{c + v_x}{c + v_s} f_0$$

Produciéndose entonces un cambio de frecuencia aparente, Δf:

$$\Delta f = \frac{\Delta v}{c} f_0 = \frac{v_x - v_s}{c} f_0$$

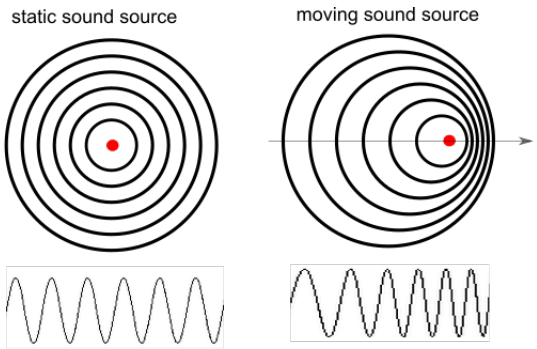

Figura 2.3.1: Efecto Doppler - Fuente estática vs dinámica

2.4. Velocidad de propagación: influencia del viento

La velocidad de propagación del sonido depende del ambiente en cuestión. Factores como la temperatura, humedad o la velocidad de movimiento del aire introducen una deriva que se puede traducir en una desviación en la estimación final. El movimiento del medio de propagación (aire) tiene un doble impacto en la estimación produciendo un incremento de la medida estimada y un cambio en la componente longitudinal dependiente de la dirección del viento [20] [2.4.1]

$$v_{sa} = v_{al} + v_s\sqrt{1 - \left(\frac{v_{at}}{v_s}\right)^2} \tag{2.4.1}$$

donde v_{sa} es la velocidad de propagación del sonido en el aire, v_{al} y v_{at} sus componentes longitudinales y trasversales respectivamente y v_s la velocidad de movimiento del aire. En condiciones normales $v_s \in [0, 10]m/s$, por lo que:

$$v_{sa} \cong v_s + v_{al} - \frac{1}{2}\left(\frac{v_{at}^2}{v_s}\right) \tag{2.4.2}$$

Por consiguiente, la influencia de la componente trasversal de la velocidad del viento puede expresarse como:

$$v_{sa} = v_s + v_{al} = v_s + \vec{v_a}\vec{u}_i \qquad \vec{u}_i = \frac{(\vec{x}_i - \vec{x})}{\|\vec{x_i} - \vec{x}\|} \tag{2.4.3}$$

Donde $\vec{u}_i$ es un vector unitario desde el emisor hacia el receptor (i). De la ecuación (2.4.3) derivamos las componentes del vector de propagación:

$$\vec{v}_a = (v_{ax}, v_{ay}, v_{az})$$

Capítulo 3

Estimación del tiempo de llegada (TOA)

En el capitulo anterior se estudiaron los efectos que pueden alterar la percepción sonora de una onda acústica. Existe una gran dependencia con cada uno de los efectos planteados ya que requerimos una estimación de la señal de de llegada para poder estimar la distancia con respecto al foco emisor. Este capitulo describe las diferentes técnicas para su estimación así como los posibles inconvenientes y efectos adversos que pueden introducir errores en el proceso bajo la existencia de una señal con o sin linea de visión directa (LOS, Line-of-Sight) o (NLOS, No-Line-of-Sight).

3.1. Modelado del canal de propagación

En la actualidad existen una infinidad de algoritmos para la estimación de la TOA basados en la determinación del retardo de la señal que se recibe por el primer camino. A groso modo, estos se pueden agrupar en dos grupos:

- Basados en el dominio del tiempo: analizan las propiedades de la respuesta impulsiva del canal para estimar la TOA.
- Basados en el dominio de la frecuencia: analizan las propiedades espectrales del canal para estimar la TOA. Generalmente son conocidos como algoritmos de estimación de alta resolución (SRTDA, Super-Resolution Time-Delay Algorithm) por su extremada precisión [11].

En cualquiera de los casos se requiere de un conocimiento previo del canal de propagación para una estimación adecuada. De forma genérica un canal inalámbrico con los efectos planteados en las secciones anteriores se puede modelar según la Figura 3.1.1, donde $h(n)$ representa los efectos de los diferentes caminos y $w(n)$ el ruido aditivo que por conveniencia sera independiente a la señal de propagación.

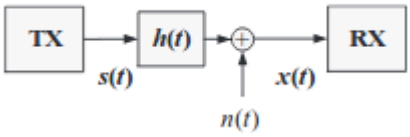

Figura 3.1.1: Canal con un ruido aditivo y efecto multipath (h(n))

$$h(n) = \sum_{m=0}^{M-1} \alpha_m \delta\left(t - \tau_m\right) \tag{3.1.1}$$

En la ecuación (3.1.1) , M representa el numero de caminos así como α_m y τ_m la atenuación y retardo de cada camino. Estos parámetros dependen de las condiciones del entorno así como de la posición del objeto móvil. De esta forma si transmitimos una señal $s(n)$, la señal recibida sera:

$$x(n) = s(n) * h(n) + w(n) = \sum_{m=0}^{M-1} \alpha_m s\left(t - \tau_m\right) + w(n) \tag{3.1.2}$$

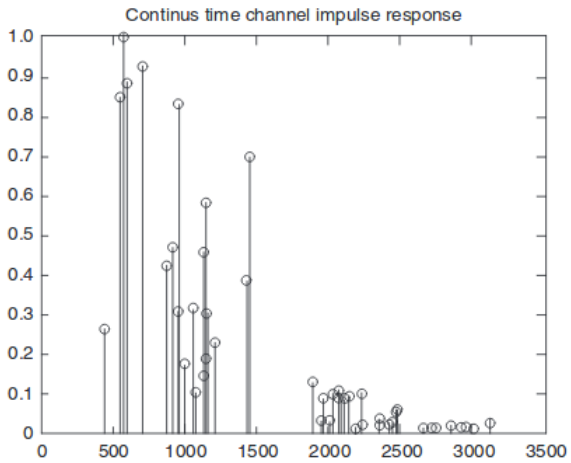

Figure 7.4 Channel impulse response.

Figura 3.1.2: Respuesta impulsiva de un canal inalambrico con ruido aditivo gaussiano y varios caminos.

3.2. Estimación mediante el uso de Cross-Correladores

La TOA puede interpretarse como la función coste que maximiza la correlación cruzada entre la señal recibida y una señal conocida transmitida. Muchos de los algoritmos existentes para estimar la TOA hacen uso de esta propiedad y son conocidos como Cross-Correladores, Figura 3.2.1 y 3.2.2. Esta técnica es altamente sensible a los efectos derivados del efecto multipath ya que podran darse casos donde el camino que maximice la función de coste del Cross-Correlador no sea la señal LOS [24]. Pese a ello estos algoritmos son muy usados debido a su sencillez y bajo coste computacional.

Figura 3.2.1: Diagrama de bloques de un Cross-Correlador gen´erico

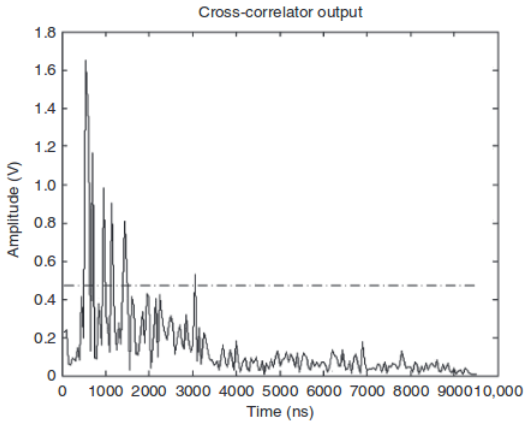

Figura 3.2.2: Ejemplo: Salida de un Cross-Correlador

En la práctica solo dispondremos de un número finito de muestras en un intervalo T por lo que tendremos que trabajar con una versión estimada de la función de correlación, $\tilde{R}$. Este inconveniente se minimiza asumiendo que nuestro sistema es un *proceso ergódico* así que con las suficientes muestras podremos derivar las propiedades estadísticas de la señal. En este caso, la función de autocorrelación se puede estimar según [3.2.1].

$$\tilde{R}_{x_i\tilde{x}_i}(t) = E\left[x_i(t)\,\tilde{x}_i(t-\tau)\right] = \frac{1}{T-\tau}\int_{\tau}^{T} x_i(t)\,\tilde{x}_i(t-\tau)\,dt \tag{3.2.1}$$

Para mejorar la precisión en la estimación del intervalo $\tilde{D}$ es deseable realizar un proceso previo de pre-filtrado de las señales x_i y $\tilde{x}_i$, obteniendo las señales y_i y $\tilde{y}_i$, para posteriormente aplicar el proceso de integración y cuadratura tal y como ilustra la Figura 3.2.3. El retardo introducido en el filtro H_2 que maximice el pico en la salida será una estimación del retardo. Nótese que cuando $H_1(f) = H_2(f) = 1 \quad \nabla f$, la estimación del retardo se corresponde con valor del eje de abscisas en que la función de autocorrelación presenta un máximo.

La función $\tilde{R}$ puede expresarse en función de la correlación de la energía espectral de las señales de salida de los filtros, $G_{y_i\tilde{y}_i}(f)$, ecuación [3.2.2]. En la práctica, solo podremos obtener una versión estimada por lo que la señal de estimación del retardo se puede obtener según [3.2.3]. Se hace necesario entonces un conocimiento previo de las señales y de la naturaleza del entorno para ajustar y optimizar la estimación de $\psi_g(f)$.

$$G_{y_i\tilde{y}_i}(f) = H_1(f)\,H_2^*(f)\,G_{x_i\tilde{x}_i}(f) = \psi_g(f)\,G_{x_i\tilde{x}_i}(f) \tag{3.2.2}$$

$$\tilde{R}_{y_i\tilde{y}_i}(\tau) = \mathcal{F}^{-1}\left[G_{y_i\tilde{y}_i}(f)\right] = \int_{-\infty}^{\infty} \psi_g(f)\,\tilde{G}_{x_i\tilde{x}_i}(f)\,e^{j2\pi f\tau}df \tag{3.2.3}$$

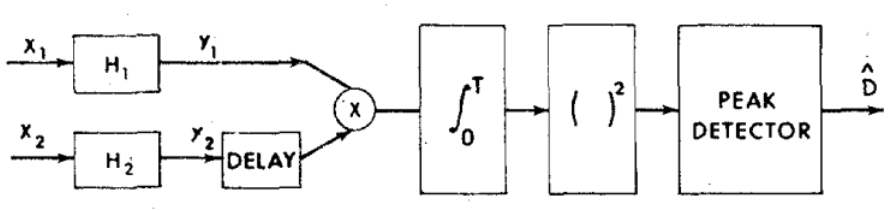

Figura 3.2.3: Señal recibida preprocesada para distintos valores de τ para mejorar la estimación del retardo $\tilde{D}$

A continuación se presentan algunos de los algoritmos más cómunes para la estimación de la TOA.

3.2.1. Algoritmo GCC

Un Cross-Correlador comúnmente usado se basa en un estimador ML para el retardo D que intentara derivar el valor de τ que maximice (3.2.3]) conocidas las propiedasdes espectrales de la señal original y el ruido de ambiente. La correlación cruzada se puede obtener como:

$$R_{x_i\tilde{x}_i}(\tau) = \alpha R_{s_i s_i}(\tau - D) + R_{n_1,n_2}(\tau) \tag{3.2.4}$$

$$G_{x_i\tilde{x}_i}(f) = \alpha G_{s_i s_i}(f)\, e^{-j2\pi f D} + G_{n_1 n_2}(f) \tag{3.2.5}$$

Aplicando la transformada de Fourier, se puede derivar la densidad espectral correlada de ambas señales. Como los ruidos presentes no están correlados, entonces la correlación de la densidad espectral de ambas señales es una versión espectral escalada multiplicada por una exponencial compleja debido al retardo D, ecuación [3.2.5]. Una operación de multiplicación en el dominio de la frecuencia implica una convolución en el dominio temporal, por lo que:

$$R_{x_i\tilde{x}_i}(\tau) = \alpha R_{s_i s_i}(\tau) * \delta(\tau - D) \tag{3.2.6}$$

Aunque una propiedad fundamental de la función de autocorrelación es que $R_{ss}(\tau) \leq R_{ss}(0)$, para el caso que nos incumbe por regla general esta igualdad se cumple cuando $\tau \neq 0$; el valor máximo del CC se encuentra en D. Esto no es un gran problema en caso de darse un único retardo pero en la práctica debido a los efectos de los distintos ecos (multipath) coexisten distintos retardos:

$$R_{x_i\tilde{x}_i}(\tau) = R_{s_i s_i}(\tau) \otimes \sum_n \alpha_n \delta(\tau - D_n) \tag{3.2.7}$$

Por tanto en condiciones ideales en que $\nabla f \quad \tilde{G}_{x_i\tilde{x}_i}(f) \cong G_{x_i\tilde{x}_i}(f)$, el conjunto de filtros $\psi_g(f)$ deben de albergar cada uno de los máximos locales derivados de cada uno de los distintos retardos. El inconveniente radica en que estos máximos son más sensibles al tamaño de muestras especialmente en condiciones

ruidosas por lo que se hace necesario un compromiso entre la resolución y la estabilidad del sistema a la hora de seleccionar los filtros.

3.2.2. Algoritmo de Roth

El método de Roth [6] propone la estimación de la respuesta impulsiva del filtro de Wiener-Hopf óptimo utilizando [3.2.8]. De esta forma, la función de autocorrelación se puede estimar según [3.2.9].

$$\psi_i(f) = \frac{1}{G_{x_i,x_i}} = \frac{1}{G_{s_i,s_i} + G_{n_1,n_1}} \tag{3.2.8}$$

$$\tilde{R}_{y_i\tilde{y}_i}(\tau) = \delta(\tau - D) \otimes \alpha \int_{-\infty}^{\infty} \frac{G_{s_i s_i}(f)}{\{G_{s_i,s_i} + G_{n_1,n_1}\}} e^{j2\pi f\tau} df \tag{3.2.9}$$

En la práctica presenta el inconveniente de que elimina aquellas regiones de frecuencias en las que $G_{n_1,n_1}(f)$ es grande por lo que la estimación de $\tilde{G}_{x_i\tilde{x}_i}(f)$ es propensa a errores debido a que introduce efectos de "spreading" en el espectro.

3.2.3. Algoritmo SCOT

El inconveniente principal del algoritmo de Roth en la estimación espectral de $\tilde{G}_{x_i\tilde{x}_i}(f)$ se debe a que solo considera los efectos debidos a aquellas regiones del espectro en que las propiedades espectrales de $G_{n_1,n_1}(f)$ se acentúan, despreciando las de $G_{n_2,n_2}(f)$. El algoritmo de SCOT (Smoothed Coherence Transform) [6]intenta solventar este inconveniente introduciendo los efectos de ambos [6]. Aunque presente mejores resultados sigue existiendo un problema de "spreading" que se incrementa conforme mayor es la presencia de ruido.

$$\psi_i(f) = \frac{1}{\sqrt{G_{x_i,x_i} G_{\tilde{x}_i\tilde{x}_i}}} \tag{3.2.10}$$

$$\tilde{R}_{y_i\tilde{y}_i}(\tau) = \int_{-\infty}^{\infty} \frac{\tilde{G}_{x_i,\tilde{x}_i}}{\sqrt{G_{x_i,x_i} G_{\tilde{x}_i\tilde{x}_i}}} e^{j2\pi f\tau} df \tag{3.2.11}$$

3.2.4. Algoritmo PHAT

El algoritmo PHAT (Phase Transform) intenta reducir los efectos del "spreading" realizando una ponderación de los efectos del ruido [6]:

$$\psi_i(f) = \frac{1}{|G_{x_i\tilde{x}_i}|} \tag{3.2.12}$$

$$\tilde{R}_{y_i\tilde{y}_i}(\tau) = \alpha \int_{-\infty}^{\infty} \frac{\tilde{G}_{x_i\tilde{x}_i}(f)}{|G_{x_i\tilde{x}_i}|} e^{j2\pi f\tau} df \tag{3.2.13}$$

en el caso en qué los ruidos no están correlados entonces:

$$\psi_i(f) = \frac{1}{|G_{x_i\tilde{x}_i}|} = \frac{1}{|\alpha G_{s_i s_i}(f) + G_{n_1 n_2}(f)|} = \frac{1}{\alpha G_{s_i s_i}(f)}$$

En condiciones ideales, si la estimación $\tilde{G}_{x_i\tilde{x}_i}(f) = |G_{x_i\tilde{x}_i}|$ entonces se cumple [??]. El resultado presenta una magnitud unitaria que deriva la ecuación [3.2.14].

$$\frac{\tilde{G}_{x_i\tilde{x}_i}(f)}{|G_{x_i\tilde{x}_i}|} = e^{j\theta(f)} = e^{j2\pi f D}$$

$$\tilde{R}_{y_i\tilde{y}_i}(\tau) = \delta(\tau - D) \tag{3.2.14}$$

En la práctica existen dos inconveniente principales:

- La función obtenida es una aproximación de la función $\delta(t)$.
- La ponderación de $\tilde{G}_{x_i\tilde{x}_i}(f)$ como inversa de $G_{s_i s_i}(f)$ hace que los errores en la estimación sean inversamente proporcionales a la energía de la señal. Presentando indeterminaciones cuando la energía de la señal es nula.

3.3. Errores en la estimación de la TOA

El cálculo de la distancia a partir de la estimación del retardo en la propagación de una determinada señal es el foco principal del sistema planteado. La exactitud del sistema depende drásticamente de la estimación correcta de la TOA. La TOA presenta una sensibilidad a los efectos planteados en el capitulo anterior pero también a muchos otros relacionados con el propio diseño del sistema. En esta sección se discuten algunos para paliar los efectos de los mismos durante los procesos posteriores de calibración.

3.3.1. Sensibilidad al ancho de banda

En un caso real los diferentes objetos producirán reflexiones y refracciones de la señal original que se traducirán en diferentes caminos de llegada al receptor. Este efecto es conocido como efecto multi-camino (*multipath*) y es considerado el mayor problema en la estimación de la TOA [5, 14, 18]. Los efectos en la estimación son dependientes de los caminos recibidos:

- Caminos cercanos (*early reflections*): estos caminos llegan justo después de la señal LOS. El inconveniente radica en que estos producen un solapamiento que entorpece la estimación de la TOA a través de la señal de visión directa.
- Caminos lejanos (*late reflections*): debido a efectos adversos la señal de visión directa puede ser fuertemente atenuada llegando a recibir caminos retardados temporalmente con una energía mayor.

En el peor de los casos, este problema se ve agravado cuando la diferencia temporal entre diferentes caminos es inferior a la del pulso de sincronización (*ancho de banda*). Esto implica que para reducir los efectos derivados es necesario utilizar señales de corta duración, lo cual implica un mayor ancho de banda. *En otras palabras, la estimación de la TOA es una función dependiente del ancho de banda.*

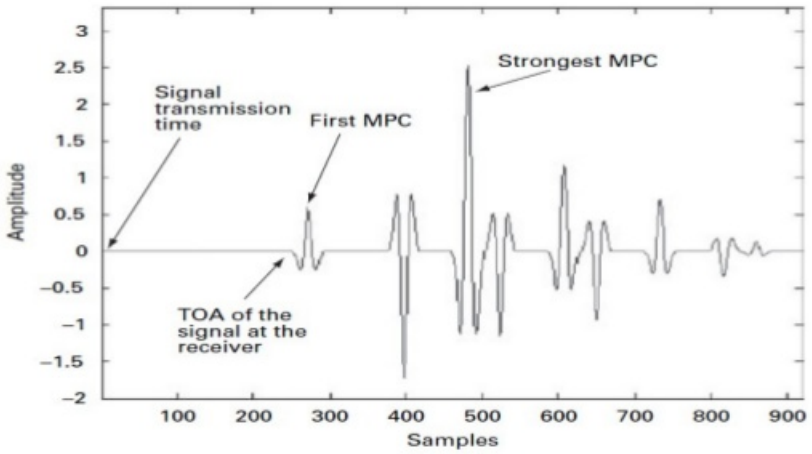

Figura 3.3.1: Problemas en la estimación de la TOA en un canal multipath

Estos efectos distorsionan fuertemente las estimaciones de los correladores. Tras la estimación de las correlaciones cruzadas el detector de picos puede estimar falsos positivos ya que bajo efectos de caminos lejanos puede darse el caso de que la señal de visión directa presenta un pico inferior al te los diferentes señales retardadas, Figura 3.3.1. Este factor distorsiona gravemente las estimaciones por lo que se requiere de técnicas que puedan palizar y reducir los efectos derivados de este tipo de canales.

3.3.2. Sensibilidad al ruido

El ruido aditivo en el canal de propagación puede dificultar la estimación de la TOA incluso en condiciones ideales sin efecto multipath. El nivel de exactitud en la estimación de la TOA del sistema depende del ancho de banda (B) y el nivel de ruido (SNR, *Signal-to-Noise-Ratio*) . Para una señal de duración T_s y con frecuencia central f_c, en [20] se propone un limite inferior en la varianza de la estimación de la TOA:

$$var\,(TOA) = \frac{1}{8\pi^2 B T_s f_c^2 SNR} \qquad f_c \gg B$$

Capítulo 4

Trilateración

Las técnicas de trilateración permiten la localización de un objeto dado un conjunto de medidas tomadas a partir de ciertos puntos de referencia y utilizando diferentes figuras geométricos para la estimación: *círculos, esferas o triángulos* entre otras [7]. A lo largo de las últimas décadas se han desarrollado diferentes técnicas de trilateración optimizadas para diferentes situaciones, tanto en interiores como en exteriores.

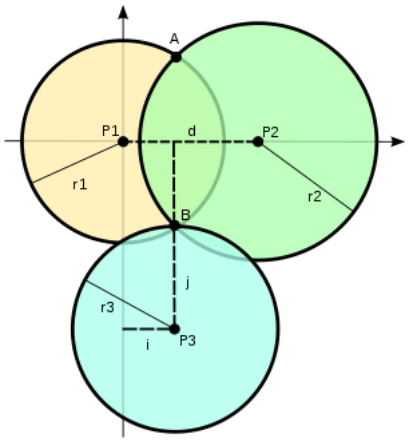

Figura 4.0.1: Principios de la trilateración en un entorno 2D

En términos simples, podemos determinar la posición (x, y, z) de cualquier elemento en un espacio tridimensional si es la intersección de al menos tres esferas de radio y origen conocido. Por consiguiente, si partimos de la ecuación general de un esfera centrada en un punto (x_a, y_a, z_a):

$$r^2 = (x - x_a)^2 + (y - y_a)^2 + (z - z_a)^2 \tag{4.0.1}$$

tendremos que resolver un sistema conformado por las ecuaciones de N esferas, conocidos sus respectivos radios para identificar la posición (x, y, z) del objeto:

$$(x, y, z) = \begin{cases} r_1^2 = (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 \\ r_2^2 = (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 \\ \vdots \\ r_N^2 = (x - x_N)^2 + (y - y_N)^2 + (z - z_N)^2 \end{cases} \qquad N \geq 3 \tag{4.0.2}$$

Esta situación presenta la ventaja de que podemos situar todos los focos emisores a una misma altura, por lo que podemos considerar todos los valores de $z_a = 0$, de esta forma la ecuación (4.0.1) se reduce a la ecuación (4.0.3), simplificando todo el desarrollo.

$$r^2 = (x - x_a)^2 + (y - y_a)^2 + z^2 \tag{4.0.3}$$

Por otro lado, podemos utilizar el método de Fang [23] y situar uno de los focos emisores de referencia en el origen de coordenadas $(0, 0, 0)$ y otro de referencia a una distancia conocida en el eje de abscisas $(x_2, 0, 0)$, de forma que para cualquier tercera esfera situada en un punto $(x_3, y_3, 0)$ tal y como ilustra la Figura 4.0.2.

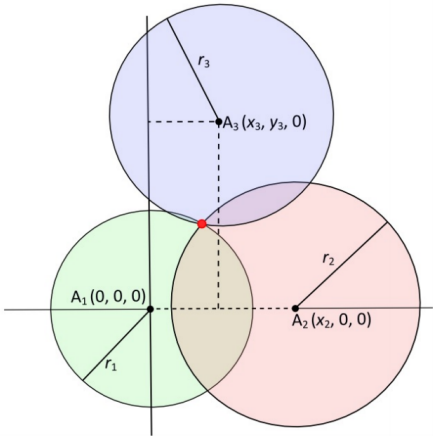

Figura 4.0.2: Método de Fang para entornos 2D

El punto de intersección vendrá dado por el sistema de ecuaciones siguiente:

$$\begin{aligned} r_1^2 &= x^2 + y^2 + z^2 \\ r_2^2 &= (x - x_1)^2 + y^2 + z^2 \\ r_3^2 &= (x - x_3)^2 + (y - y_3)^2 + z^2 \end{aligned}$$

de la cual se obtienen las siguientes soluciones:

$$\begin{aligned} x &= \frac{r_1^2 - r_2^2 + x_2^2}{2x_2} \\ y &= \frac{r_1^2 - r_3^2 + x_3^2 + y_3^2}{2y_3} \\ z &= \sqrt{r_1^2 - x^2 - y^2} \end{aligned} \tag{4.0.4}$$

Esta técnica no es siempre aplicable para condiciones reales ya que la posición de los focos emisores en muchas ocasiones queda fuera de nuestro control. A continuación se presentan diferentes técnicas para resolver las ecuaciones planteadas [25].

4.1. Trilateración mediante linealización

El sistema de ecuaciones planteado en (4.0.2) es un sistema complejo de difícil resolución ya que no es lineal. En esta sección se plantea un proceso para la linealización de la ecuación planteada para obtener una solución adecuada de cómputos reducidos. La ecuación (4.0.1) puede ser reescrita añadiendo un componente (x_j, y_j, z_j) como restricción para el proceso de linealización [21]. Por consiguiente:

$$r_i^2 = (x - x_j + x_j - x_i)^2 + (y - y_j + y_j - y_i)^2 + (z - z_j + z_j - z_i)^2$$

con $(i = 1, 2, ..., j-1, j+1, ..., n)$.
Desarrollando y reagrupando términos:

$$\begin{aligned}
&(x - x_j)(x_i - x_j) + (y - y_j)(y_i - y_j) + (z - z_j)(z_i - z_j) \\
&= \frac{1}{2}\left[(x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2 - r_i^2 + (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2\right] \\
&= \frac{1}{2}\left[r_j^2 - r_i^2 + d_{ij}^2\right] = b_{ij} \qquad (4.1.1)
\end{aligned}$$

donde d_{ij} es la distancia entre el emisor B_i y el emisor B_j:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \qquad (4.1.2)$$

Realmente no importa cual de los distintos emisores tomemos como restricción, por normalización arbitraria tomaremos el valor del primero de ellos $(j = 1)$, de forma que, $i = 2, 3, ..., N$, dando como resultado un sistema de $(N-1)$ ecuaciones con 3 incógnitas (x, y, z):

$$\begin{aligned}
(x - x_1)(x_2 - x_1) + (y - y_1)(y_2 - y_1) + (z - z_1)(z_2 - z_1) &= b_{21} \\
(x - x_1)(x_3 - x_1) + (y - y_1)(y_3 - y_1) + (z - z_1)(z_3 - z_1) &= b_{31} \\
&\vdots \\
(x - x_1)(x_N - x_1) + (y - y_1)(y_N - y_1) + (z - z_1)(z_N - z_1) &= b_{N1}
\end{aligned} \qquad (4.1.3)$$

El sistema puede ser escrito de forma matricial fácilmente:

$$A\overrightarrow{x} = \overrightarrow{b} \qquad (4.1.4)$$

donde:

$$A = \begin{pmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ \vdots & \vdots & \vdots \\ x_N - x_1 & y_N - y_1 & z_N - z_1 \end{pmatrix} \quad \overrightarrow{x} = \begin{pmatrix} x - x_1 \\ y - y_1 \\ z - z_1 \end{pmatrix} \quad \overrightarrow{b} = \begin{pmatrix} b_{21} \\ b_{31} \\ \vdots \\ b_{N1} \end{pmatrix}$$

4.2. Trilateración mediante mínimos cuadrados lineales

El sistema de ecuaciones planteado en (4.1.4) presenta el inconveniente de que presenta pésimos resultados cuando las distancias entre los distintos emisores son muy próximas [21]. Además, partiendo de la idea de que la obtención de los distintos radios del sistema es aproximada, nuestro sistema de ecuaciones quedaría como (4.2.1). En estos casos no se concreta un punto de convergencia sino una región que albergará la posición real del objeto, Figura .4.2.1

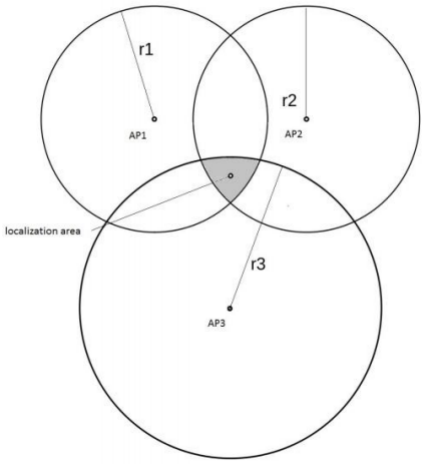

Figura 4.2.1: Área de convergencia de las distintas esferas debido a un error en la estimación de los radios.

$$A\overrightarrow{x} \approx \overrightarrow{b} \tag{4.2.1}$$

Una forma de optimizar los resultados es la reducción del error cuadrático de las desviaciones en las estimaciones de los radios mediante el uso de técnicas como la resolución por mínimos cuadrados. Partiendo del error cuadrático residual:

$$S = \overrightarrow{r}^T\overrightarrow{r} = \left(\overrightarrow{b} - A\overrightarrow{x}\right)^T\left(\overrightarrow{b} - A\overrightarrow{x}\right)$$

derivamos la ecuación normalizada [12]:

$$A^TA\overrightarrow{x} = A^T\overrightarrow{b} \tag{4.2.2}$$

La forma más sencilla de resolver las ecuación para la obtención de $\overrightarrow{x}$ dependerá de las condiciones de A^TA, de forma que:

- Si A^TA no es singular y presenta las condiciones adecuadas entonces:

$$\overrightarrow{x} = \left(A^T A\right)^{-1} A^T \overrightarrow{b}$$

- Si $A^T A$ es singular y presenta unas condiciones pésimas tendremos que realizar una descomposición QR normalizada. En este método consideramos que $A = QR$, donde Q es una matriz ortogonal y R una matriz triangular. De esta forma el sistema se reduce a:

$$R\overrightarrow{x} = Q^T \overrightarrow{b}$$

4.3. Trilateración mediante descomposición en valores singulares

Existen situaciones en las que la operación $A^T A$ da como resultado una matriz con características próximas a la singularidad pese a que A no las presente. Esta operación permite una descomposición mediante la descomposición QR [12, 21]. Utilizando las propiedades de la matriz pseudo-inversa, la solución optima para $\overrightarrow{x_0}$ para el problema de la minimización de $\left\|A\overrightarrow{x} - \overrightarrow{b}\right\|$ viene determinado por la ecuación (4.3.1).

$$\overrightarrow{x_0} = A + \overrightarrow{b} \tag{4.3.1}$$

Si realizamos un proceso de factorización haciendo uso de la descomposición en valores singulares, ecuación (4.3.3), podemos determinar la matriz pseudo inversa A^+como:

$$A^+ = V\Sigma^+ U^H \tag{4.3.2}$$

siendo $U \in \Re^{mxn}$ una matriz con columnas ortogonales, $V \in \Re^{nxn}$ una matriz ortogonal y $\Sigma \in \Re^{mxn}$ una matriz formada por los valores singulares de la matriz A ordenados descendentemente en la diagonal.

$$A = U\Sigma V^T \tag{4.3.3}$$

4.3.1. Valores singulares

Toda matriz A de tamaño mxn presenta una matriz $A^T A$ de tamaño nxn simétrica y por tanto diagonizable ortogonálmente con auto-valores reales y no negativos [12]. Esta propiedad se cumple ya que para cualquier auto-vector u, se tiene que:

$$A^T Au = \lambda u \tag{4.3.4}$$

equivalentemente:

$$u^T A^T Au = \lambda u^T u$$

por consiguiente:

$$\|Au\|^2 = \lambda \|u\|^2 \;\; \lambda \geqslant 0$$

Por la propia simetría, presentará auto-vectores ortogonales de longitud unitaria, por lo que la ecuación (4.3.4) puede expresarse equivalentemente:

$$\|Au_i\|^2 = \lambda_i \quad i \in [1, n]$$

por consiguiente, los valores singulares singulares, σ_i, de la matriz pueden obtenerse segun (4.3.5). Es una convención acomodar los valores singulares de modo que $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n$.

$$\sigma_i = \|Au_i\| = \sqrt{\lambda_i} \tag{4.3.5}$$

4.3.2. Descomposición en valores singulares

Una condición necesaria para resolver la ecuación (4.3.3) es que los valores singulares no sean nulos $\sigma_1 \geq \sigma_r \geq 0$ siendo $r \leq n$. En estas situaciones Σ tendrá una forma en bloques, formada por la matriz de valores singulares D y un conjunto de matrices nulas $\mathcal{O}$:

$$\Sigma = \begin{pmatrix} D_{nxn} & \mathcal{O}_{rx(n-r)} \\ \mathcal{O}_{(n-r)x(r)} & \mathcal{O}_{(n-r)x(n-r)} \end{pmatrix} \tag{4.3.6}$$

$$D = \begin{pmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_r \end{pmatrix} \tag{4.3.7}$$

Para la obtención de V necesitamos determinar una base ortogonal $\{v_1, ..., v_n\}$ de vectores de $\Re^n$ compuesta por auto-vectores de la matriz $A^T A$ de nxn, de forma que:

$$V = \{v_1, ..., v_n\} \tag{4.3.8}$$

Seguidamente, procedemos a obtener la matriz U. Retomando el conjunto de vectores:

$$\{Av_1, Av_2, ..., Av_n\}$$

se tiene que para que por ser ortogonales, para todo $i \neq j$:

$$v_j^T A^T A v_i = v_j^T \lambda_i v_i = 0$$

por tanto, Av_i es ortogonal a Av_j si no son nulas ya que v_i lo es a v_j. La definición de los auto-valores, ecuación (4.3.5), nos permite normalizar un conjunto u_i ortogonal para $i \leq r$:

$$u_i = \frac{Av_i}{\sigma_i}$$

Para $i > r$, tenemos que $\|Av_i\|^2 = 0$, luego $Av_i = 0$, luego el conjunto restante de u_i se puede extender para tener n vectores ortogonales en $\Re^n$. La matriz U se puede formar por las columnas ortogonales del conjunto de vectores calculados:

$$U = [u_1, u_2, ..., u_r, |u_{r+1}, ..., u_n] \tag{4.3.9}$$

Para el caso que nos incumbe, se cumple que $m \geq n$, luego podemos expresar la ecuación (4.3.3) como:

$$\begin{aligned}
A &= U\Sigma V^T \\
&= \begin{pmatrix} u_1 & u_2 & \cdots & u_n \end{pmatrix} \begin{pmatrix} \sigma_1 & \dots & 0 & \\ \vdots & \ddots & \vdots & \mathcal{O} \\ 0 & \dots & \sigma_r & \\ & \mathcal{O} & & \mathcal{O} \end{pmatrix} \begin{pmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_n^T \end{pmatrix} \\
&= \begin{pmatrix} u_1 & \cdots & u_r & |u_{r+1} & \cdots & u_n \end{pmatrix} \begin{pmatrix} \sigma_1 & \dots & 0 & \\ \vdots & \ddots & \vdots & \mathcal{O} \\ 0 & \dots & \sigma_r & \\ & \mathcal{O} & & \mathcal{O} \end{pmatrix} \begin{pmatrix} v_1^T \\ \vdots \\ v_r^T \\ v_{r+1}^T \\ \vdots \\ v_n^T \end{pmatrix}
\end{aligned}$$

Obteniéndose:

$$\begin{aligned}
A &= U\Sigma V^T \\
&= \begin{pmatrix} u_1 & u_2 & \cdots & u_r \end{pmatrix} \begin{pmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_r \end{pmatrix} \begin{pmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_r^T \end{pmatrix} + \begin{pmatrix} u_{r+1} & \cdots & u_n \end{pmatrix} (\mathcal{O}) \begin{pmatrix} v_{r+1}^T \\ \vdots \\ v_n^T \end{pmatrix} \\
&= \begin{pmatrix} u_1 & u_2 & \cdots & u_r \end{pmatrix} \begin{pmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_r \end{pmatrix} \begin{pmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_r^T \end{pmatrix} \\
&= \begin{pmatrix} \sigma_1 u_1 & \sigma_2 u_2 & \cdots & \sigma_r u_r \end{pmatrix} \begin{pmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_r^T \end{pmatrix} \\
&= \sigma_1 u_1 v_1^T + \ldots + \sigma_r u_r v_r^T
\end{aligned}$$

Finalmente, obtenemos una expresión sencilla para la descomposición mediante valores singulares:

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T \tag{4.3.10}$$

4.4. Resolución por mínimos cuadrados no lineales

En [21] se han desarrollado diversos estudios sobre las técnicas más optimizadas para la trilateración de estimaciones aproximadas siendo la resolución por mínimos cuadrados la técnica que mejores resultados ha presentado para entornos cerrados. La ventaja yace en que esta técnica no presenta algunas de las restricciones de los algoritmos anteriormente planteados. El uso de la misma no es tan extendido en sistemas de posicionamiento debido a algunas de sus restricciones:

- Esta técnica no debe de utilizarse en entornos no controlados. Es válida siempre y cuando los dispositivos se encuentren dentro del perímetro de cobertura de los distintos nodos. En la práctica se producen errores en las estimaciones conforme se alejan del perímetro de cobertura.
- Esta técnica no debe de utilizarse para situar un objeto en un espacio tridimensional ya que la solución presentará ambigüedad en los signos. El error en las estimaciones crece conforme aumentamos la elevación del punto a estimar.

En cualquier caso, el algoritmo siempre proporcionará una solución pese a que algunas de las restricciones sean violadas. Para el caso que nos incumbe ambas restricciones son asumibles. Entendemos que en entornos de interiores controlados las zonas sin cobertura son reducidas. Además las restricciones con respecto a la elevación son despreciables, pretendemos situar el objeto en el entorno independientemente de la altura en que se encuentre.

El algoritmo minimiza la suma de los errores cuadráticos de cada una de las estimaciones en las distancias, $\hat{r}_i$, de cada uno de los focos emisores, B_i. Por consiguiente, partiendo de la ecuación (4.0.1) se deriva:

$$\hat{r}_i^2 = (x - x_j) + (y - y_j)^2 + (z - z_j)^2$$

La función a minimizar será:

$$F(x, y, z) = \sum_{i=1}^{n} (\hat{r}_i - r_i)^2 = \sum_{i=1}^{n} f_i(x, y, z)^2 \tag{4.4.1}$$

siendo

$$f_i(x, y, z)^2 = \hat{r}_i - r_i = \sqrt{(x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2} - r_i^2$$

Existen diversos algoritmos para la resolución de esta ecuación [3]. En nuestro caso optamos por un proceso iterativo de ágil y sencilla implementación como el de Gauss-Newton, de forma que partiendo de las derivadas parciales de (4.4.1) y considerando que $F_{min} > 0$ y $n > 3$, entonces:

$$\frac{\partial F}{\partial x} = 2\sum_{i=1}^{n} f_i \frac{\partial f_i}{\partial x} \quad \frac{\partial F}{\partial y} = 2\sum_{i=1}^{n} f_i \frac{\partial f_i}{\partial y} \quad \frac{\partial F}{\partial z} = 2\sum_{i=1}^{n} f_i \frac{\partial f_i}{\partial z} \tag{4.4.2}$$

Si aplicamos el operador Jacobiano, J, tenemos que:

$$\overrightarrow{g} = 2J^T \overrightarrow{f}$$

donde

$$\overrightarrow{g} = \begin{pmatrix} \frac{\partial F}{\partial x} \\ \frac{\partial F}{\partial y} \\ \frac{\partial F}{\partial z} \end{pmatrix} \quad \overrightarrow{f} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{pmatrix} \quad J = \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x} & \frac{\partial f_n}{\partial y} & \frac{\partial f_n}{\partial z} \end{pmatrix}$$

El proceso iterativo se reduce entonces a:

$$\overrightarrow{R}_{k+1} = \overrightarrow{R}_k - \left(J_k^T J_k\right)^{-1} J_k^T \overrightarrow{f}_k \tag{4.4.3}$$

En la práctica este tipo de soluciones es realmente rápidas especialmente cuando $J^T J$ se ve incrementada por una matriz diagonal que hace que la dirección de búsqueda converja hacia la pendiente más pronunciada. Este método fue desarrollado por Levenberg y Marquardt [10] y su principal ventaja radica en que a diferencia del resto no hace búsqueda de un mínimo global si no que se limita a la estimación de un mínimo local prediciendo una región muy próxima a la solución. La región es estimada combinando el proceso iterativo de Gauss-Newton con el método de descenso por gradiente.

Parte II

Diseño, Implementación del Sistema

Capítulo 5

Diseño del sistema LPS

En la primera parte de esta tesis se discutieron cada una de los conceptos teóricos así como la metodología relacionada con el desarrollo del proyecto. Este capitulo introduce la descripción del sistema utilizado por cada una de las herramientas. En la sección se introduce los conceptos básicos que conciernen el desarrollo de un sistema de posicionamiento así como las variantes planteadas.

5.1. Diseño de un sistema LPS acústico

El objetivo principal de este proyecto es el desarrollo, implementación y evaluación de una herramienta para manipular un sistema inalámbrico de posicionamiento en interiores basado en señales acústicas bautizado como AcousticLPS. El sistema planteado actua de forma pasiva, similar a otros planteado en [14]. El proceso de identificación y posicionamiento de cada uno de los dispositivos móviles encontrados en el área de visión se realiza a través de la estimación del tiempo de llegada (TOA, *Time of Arrival*) y la dirección de llegada (DOA, *Direction of Arrial*) [7] de cada uno de los nodos de la matriz de fuentes sonoras, Figura 5.1.1.

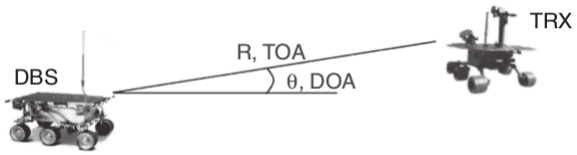

Figura 5.1.1: WLPS equipado con un sistema DBS y TRX

A través de la estimación de la TOA, conocidas las condiciones del entorno, se puede calcular la distancia recorrida por cada una de las señales sonoras. Posteriormente, la posición de un dispositivo es típicamente calculada aplicando técnicas de trilateración a partir de cada una distancias recorridas respecto a las posiciones fijas de los nodos transmisores previamente conocidas. Para hacer uso de una sencilla trilateración esférica cada uno de los nodos es previamente calibrado y sincronizado.

AcousticLPS presenta una topología maestro-esclavo, Figura 5.1.2, compuesta por un nodo controlador (BC, Beacon Controller) y un conjunto de M nodos (B, Beacons) que conformaran una matriz de fuentes sonoras, B_i. El nodo BC es el encargado de la sincronización y configuración del resto de los Beacons. Cada uno de los Beacons transmite un conjunto de códigos ortogonales que permite la identificación de la fuente original transmisora utilizando técnicas basadas en correladores en cada uno de los dispositivos conectados a la red de sensores [18] [19].

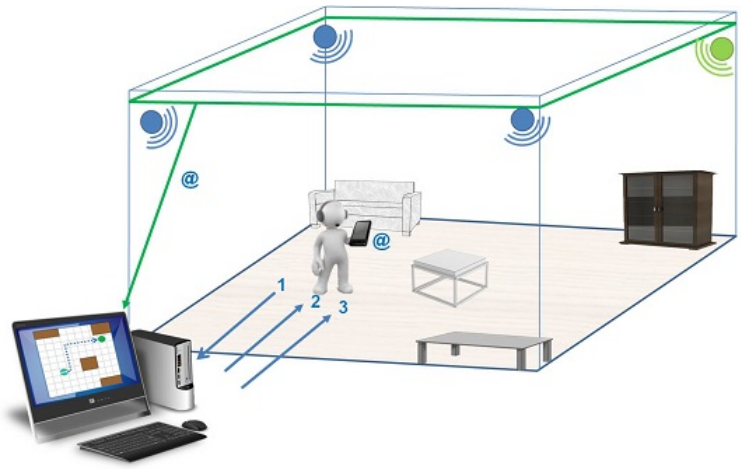

Figura 5.1.2: Sistema WLPS planteado basado en la sincronización a través de ondas acústicas

5.1.1. Beacon Controller

El nodo controlador actuá como nodo coordinador configurando y manejando cada uno de los focos sonoros, B_i. El BC viene definido por un conjunto de subrutinas que permiten la configuración, sincronización y emisión de diferentes señales, S_i, en cada uno de los nodos. Cada una de estas señales enmascara una señal sonora junto a un conjunto de códigos ortogonales similares a los códigos utilizados en el sistema CDMA [15] [14].

El nodo actuá como pasarela con el usuario a través de la aplicación AcousticLPS, Figura 5.1.3, que permite la configuración del entorno así como el rastreo (*Tracking*) en tiempo real de cada uno de los dispositivos móviles conectados en el área. Para ello se plantean dos posibilidades: rastreo remoto o rastreo local.

5.1.1.1. Rastreo remoto, (RT, *Remote Tracking*)

En esta configuración los cómputos relativos a la estimación y ajuste de la posición del dispositivo se ejecutan en remoto, Cloud Service, a través de una red inalámbrica WLAN. Esto permite el uso de algoritmos avanzados así como de ajustes precisos en tiempo real con técnicas de inteligencia artificial y aprendizaje automático. A través de la motorización de la red se puede tener un mayor control sobre los dispositivos conectados así como un ajuste preciso dependiente de las propiedades de estos últimos.

La comunicación entre cliente-servidor se realiza a través de un tunel que implemente el protocolo TCP/IP. Los mensajes son transmitidos en un formato estándar (JSON (*JavaScript Object Notation*)) y encriptados a través de un túnel HTTPS (*HyperText Transfer Protocol Secure*). El servidor implementa un aplicación con una entrada API REST (*REpresentational State Transfer*) que

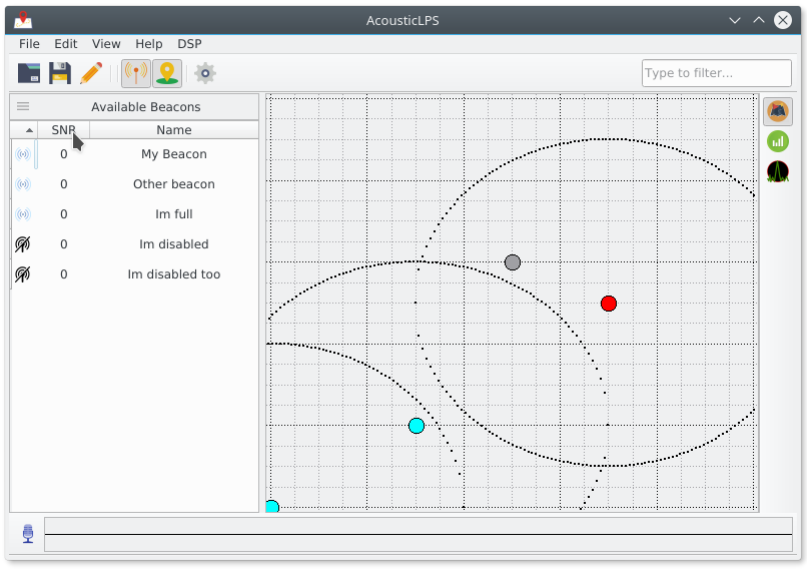

Figura 5.1.3: AcousticLPS: aplicacion Open Source para el tracking en tiempo real de cada uno de los usuarios conectados.

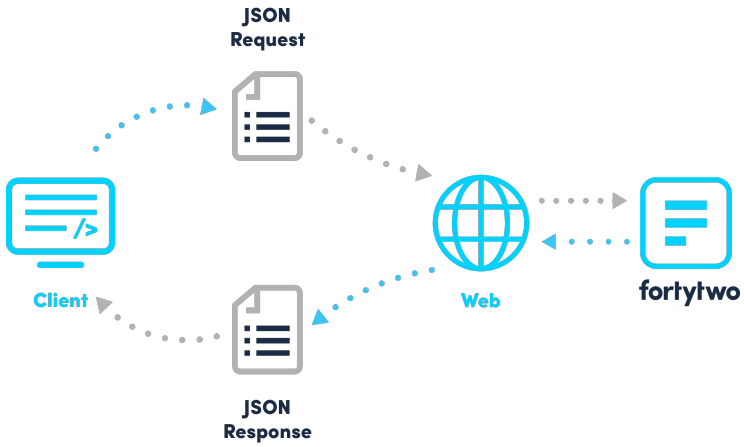

Figura 5.1.4: Intercambio de mensajes en formato JSON a traves de un tunel HTTP con un servidor remoto.

interpreta los datos, los procesa y devuelve una respuesta, Figura 5.1.4.

5.1.1.2. Rastreo local, (LT, *Local Tracking*)

En esta configuración todos los cálculos se realizan localmente en cada uno de los dispositivos móviles. En este sistema todos los dispositivos actúan de forma

pasiva y autónoma por lo que el BC no tiene ninguna información relativa al número de dispositivos conectados o al estado de la red. El cálculo se realiza a través de la aplicación SoundMaps disponible tanto para plataformas Android como dispositivos iOS.

Capítulo 6

Diseño e Implementación de las herramientas

En este capitulo se discuten las pautas discutidas para planificación y diseño de las diferentes herramientas desarrolladas para el despliegue e integración del sistema. Se han desarrollado dos aplicaciones totalmente independientes:

- QAcousticLPS: aplicación desarrollada para controlar la estación base y configurar los diferentes nodos transmisores.
- SoundMaps: aplicación para plataformas móviles (esencialmente teléfonos inteligentes y tabletas digitales) que integra el procesado necesario para la estimación y visualización de la posición del dispositivo en tiempo real.

El capitulo no ha sido escrito para ser usado como manual de usuario, describe la estructura interna de las herramientas para servir de guía para futuros desarrolladores. Se divide en tres partes principalmente, la Sección 6.1 introduce información relativa al entorno de desarrollo, la Sección 6.2 plantea la planificación para el mantenimiento del proyecto para finalmente en la Sección 6.3 introducir la estructura finalmente utilizada así como cada una de las herramientas externas integradas en la posterior implementación.

6.1. Selección del entorno de desarrollo

Para un desarrollo adecuado de un proyecto de software es necesario la planificación y análisis de los requisitos y objetivos predefinidos con el fin de optimizar el proceso de selección del entorno de desarrollo adecuado: plataforma, lenguaje de programación, compilador, test etc.

El proyecto en cuestión engloba tanto conceptos de ciencias de la computación como de ingeniería de telecomunicaciones y procesamiento de señal. Es de vital importancia el procesamiento en tiempo real para reducir los retardos en los cómputos y ofrecer unos resultados acordes a las necesidades del usuario. Se requiere entonces de un diseño que permita una ejecución rápida, eficiente y robusta de cada uno de las componentes que conforman el software final. Es por tanto necesario el uso de un lenguaje de programación que permita tanto un desarrollo ágil de alto nivel como una optimización próxima al hardware especialmente a componentes vitales como la tarjeta de sonido o la cache del procesador.

La elección no es trivial. En el mercado coexisten una amplia variedad de lenguajes de programación diseñados para diferentes aplicaciones. Cuando se trata de sistemas empotrados así como procesamiento de datos en tiempo real el lenguaje por excelencia es el lenguaje C. C es un lenguaje extensamente usado, rápido, eficiente y portable [1]. Por otra parte es la base de la mayoría de sistemas operativos en el mercado por lo que existe un gran abanico de librerías y utilidades totalmente gratuitas. Otro factor importante es que cuenta con la comunidad de desarrolladores mas amplia del mundo por lo que existen una gran cantidad de información en la red para optimizar y mejorar el código desarrollado, Figura 6.1.1.

Un factor importante a considerar es que el lenguaje C es por defecto un lenguaje de programación procedural por lo que no permitirá el uso de patrones

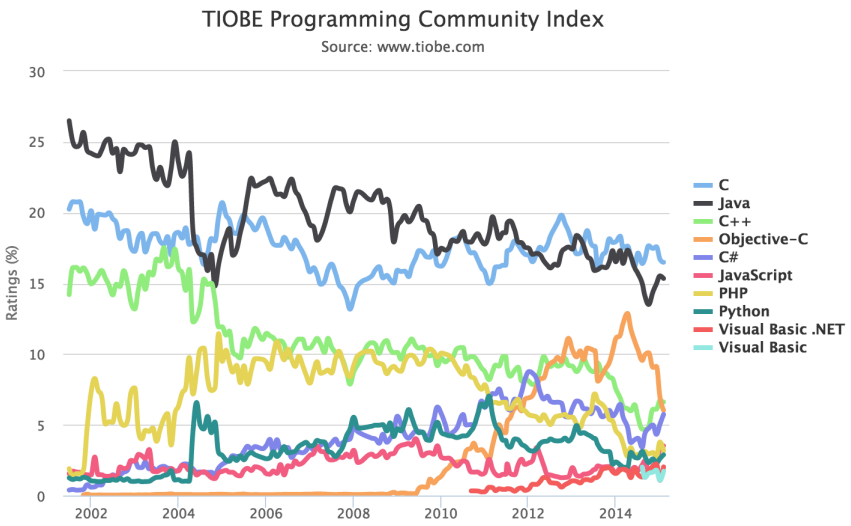

Figura 6.1.1: Evolución de la comunidad de desarrolladores de distintos lenguages de programacion. Fuente: TIOBE

de diseño avanzados diseñados para lenguajes de programación orientados a objectos (OOP, *Object Oriented Programming*), Figura 6.1.2. Estos patrones son el pilar de muchos de los programas que utilizamos a diario especialmente de aquellos que disponen de una interfaz de usuario (UI, *User Interface*). Es necesario entonces buscar una alternativa OOP que soporte las características principales de C.

El lenguaje de programación C++ es el hijo predilecto del lenguaje C. Este lenguaje fue diseñado para ser tan optimo como C pero añade un soporte nativo para el paradigma OOP. Por otro lado, permite la integración de código y librerías escritas en C de forma nativa así como código en ensamblador (*Assembler*) permitiendo el uso de todo el código previamente escrito en el lenguaje antecesor. Existe una extensa comunidad de desarrolladores que soportan y contribuyen al desarrollo del lenguaje especialmente en campos relaciones con cálculos matemáticos o procesado de audio e imagen así como tecnologías de realidad virtual, inteligencia artificial e aprendizaje automático. Todas estas ventajas hacen de C++ el candidato ideal para el desarrollo del proyecto.

Definido el lenguaje de programación, la siguiente etapa será la elección de un entorno de desarrollo que integre un compilador de C/C++ así como un editor de texto que permita programar de forma sencilla. En este caso se ha optado por un sistema Linux ya que integra todas las facilidades de forma totalmente sencilla y sin coste alguno. La distribución elegida es Ubuntu, un sistema operativo basado en Debian, ampliamente utilizado y con una gran soporte basado en una extensa comunidad de desarrolladores, Figura 6.1.3.

Linux es un sistema operativo que dispone de drivers para la mayoría de tar-

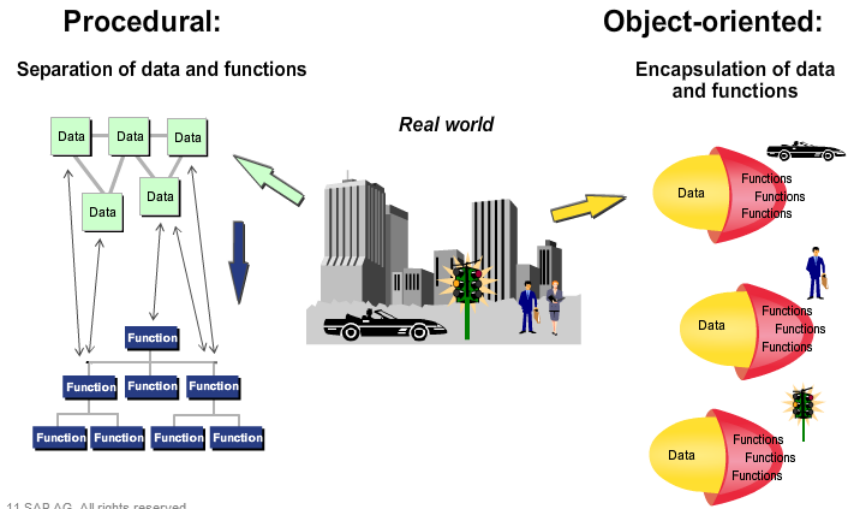

Figura 6.1.2: Programación procedural vs Programacion orientada a objetos

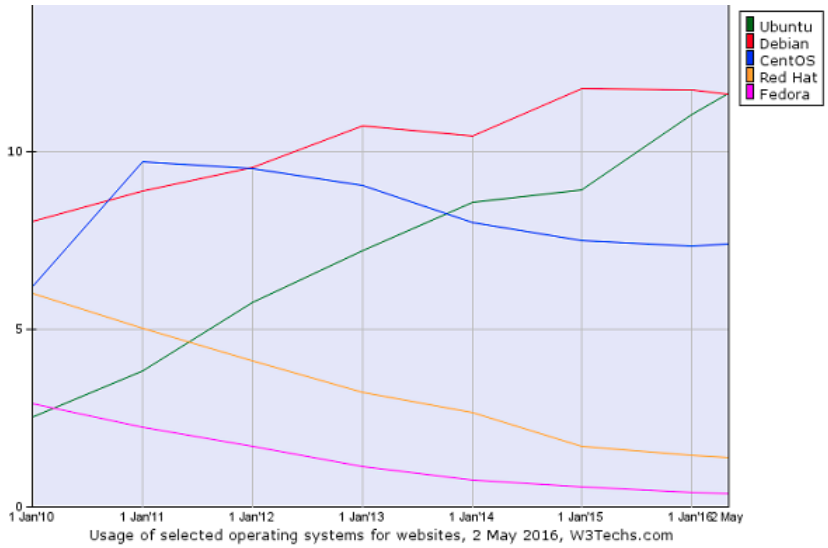

Figura 6.1.3: Distribuciones de Linux mas usadas en el mercado. Fuente: W3Techs

jetas de sonido sin modificación alguna y que incluye gestores de paquetes que facilitan el desarrollo e integración de componentes de terceros. Por otro lado, los tiempos de acceso a la tarjeta de sonido así como el manejo y sincronización de las hebras es mucho más dinámica y robusta que en entornos Windows. Finalmente, y no por ello menos importante, el código estará abierto para posibles

modificaciones y adaptaciones a las necesidades del proyecto.

6.2. Prototipado y diseño estructural

Una vez elegido el entorno de desarrollo, se procede a la definición de cada una de las etapas del proyecto. La Figura 6.2.2 muestra las componentes vitales del proyecto asi como la inter-operabilidad entre las mismas. Las componentes principales a destacar:

- **Captura**: captura de datos a través de un micrófono y el sistema IMS.
- **Extracción y procesamiento**: extracción de propiedades, computo y estimación de la posición del objeto a través de un conjunto de sub-rutinas de álgebra lineal y DSP.
- **Representación**: representación de los resultados finales en una aplicación que integre una interfaz de usuario sencilla.

Cada una de las etapas serán desarrolladas de forma totalmente independiente. Se podrán considerar el uso de pasarelas (Gateways) para adaptar y permitir inter-operabilidad entre cada una de ellas. El desarrollo se realizará a través de una integración continua basada en las pautas de la estructura del ciclo de vida de desarrollo software (SDLC, Software Development Life Cycle), Figura 6.2.1. Nacida en los 70s con una constante evolucion hasta nuestros dias [17], la estructura divide el desarrollo de software en diferentes etapas.

- **Fase de planificación**: detección y análisis de requisitos además de la elección de herramientas de desarrollo. En esta fase se barajaran cada una de las opciones disponibles en el mercado y se elegirá la más óptima para el proyecto.
- **Fase de implementación**: desarrollo y testeo del software. Se construirán cada uno de los bloques que conformarán el programa de forma independiente para posteriormente ser testeados y optimizados.
- **Fase de despliegue y mantenimiento**: liberación de la herramienta y soporte. Una vez terminada la programación y esta sea testeada se dará pie a una serie de pruebas piloto para garantizar el correcto funcionamiento de la misma.

Las etapas podrán variar permitiendo cierta flexibilidad a cambios en los requisitos del proyecto. Para el manteamiento y un mejor control se hace uso de un sistema de control de versiones, por defecto Git. Una vez desarrollada la herramienta se planificará un programa específico para exprimir todas sus características y de esta forma poder realizar una serie de pruebas de campo experimentales.

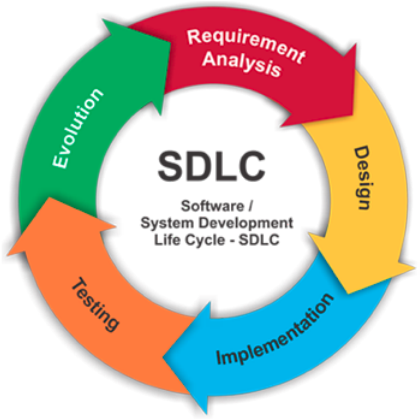

Figura 6.2.1: Estructura SDLC

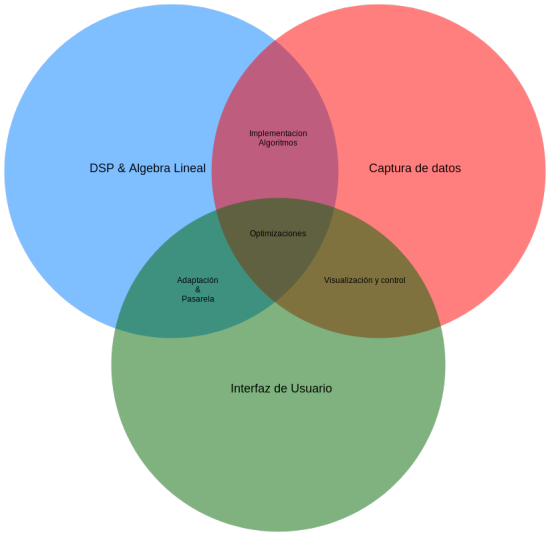

Figura 6.2.2: Planificacion del proyecto desarrollado

6.3. Estructura diseñada

Un esquemático del diseño planteado de cada una de las etapas previamente discutidas se puede encontrar en la Figura 6.3.1. Es un sistema multi-capa (Multi Layer) en el cada una de las capas tiene un papel independiente.

Este sistema es totalmente abstracto al sistema operativo en cuestión así como de la capa de interfaz de usuario permitiendo cambiar algunas de las componentes utilizadas sin alterar el resto de las componentes vitales del proyecto.

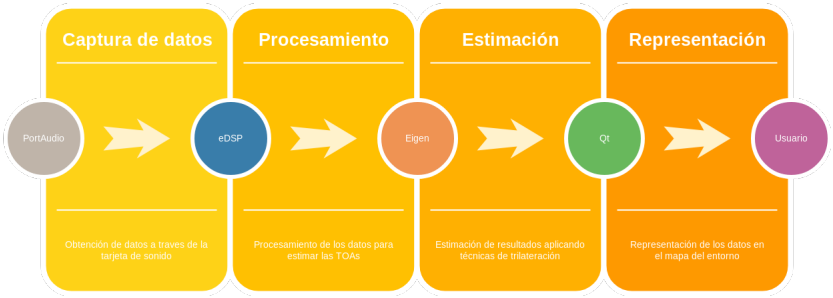

Figura 6.3.1: Implementación de cada una de las etapas definidas

Esto es especialmente útil en la capa de interfaz de usuario ya que permitirá una fácil potabilidad a diferentes plataformas con el fin de obtener una interfaz de usuario nativa. Seguidamente se procederá a la descripción de cada una de las componentes seleccionadas así como las ventajas de las mismas.

6.3.1. Implementación de una librería para captura de datos

El desarrollo de drivers en un tema complejo que no es objeto de estudio en este proyecto. Para la obtención de la señal acústica registrada por un micrófono a través de la tarjeta de sonido se utilizarán proyectos previamente desarrollados y documentados. Existen diferentes implementaciones dependientes de la arquitectura así como del sistema operativo del sistema. Como el objetivo final es el diseño de un software multiplataforma se ha optado por el uso de una librería de código abierto llamada PortAudio [4].

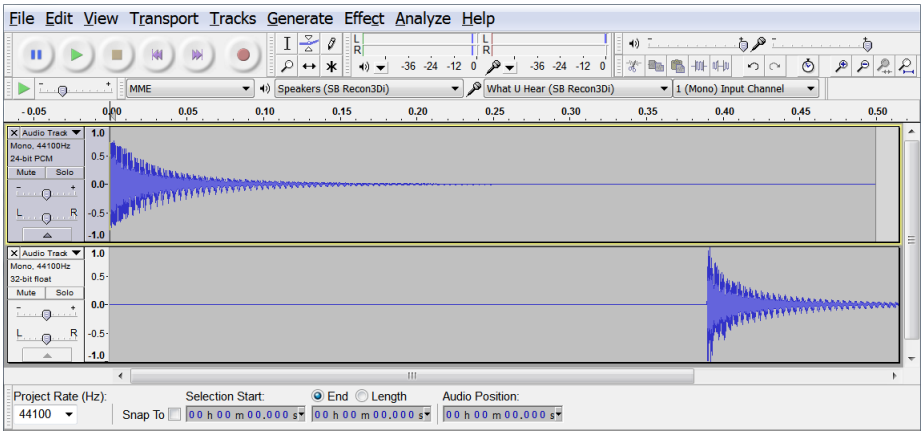

Figura 6.3.2: Audacityes uno de los programas más usados para la edición de audio. Utiliza la libreria PortAudio para interactuar con la tarjeta de sonido.

PortAudio es una librería multiplataforma para grabación y reproducción de audio. Es una librería desarrollada en C que introduce una sencilla API para acceder a componentes de bajo nivel de la tarjeta de sonido y una configuración flexible y dinámica de la misma. Para el procesamiento de los datos la librería integra un conjunto de llamadas a funciones (Callbacks) que irán procesando cada uno de los bloques de sonido. Es una librería ampliamente utilizada y con una gran comunidad de desarrolladores que contribuyen y la actualizan constantemente, Figura 6.3.2.

6.3.2. Implementación de una librería de Algebra Lineal

En la primera parte de esta TFM se desarrollaron las diferentes técnicas para la estimación de la posición del dispositivo a través del uso de técnicas de trilateración basadas en cálculos complejos de álgebra lineal y de cálculo numérico. Estas operaciones son complejas de programar, el problemas se incrementa cuando el computo debe de ser óptimo y eficiente. Esta tarea queda fuera de los objetivos definidos en este proyecto por lo que para agilizar el proceso se ha decidido utilizar librerías desarrolladas por terceras partes.

Figura 6.3.3: Libreria Eigen para cálculos de Álgebra Lineal

La librería en cuestión es conocida como Eigen [13] Figura 6.3.3, una librería extensamente utilizada en campos como la Visión por Computación (CV, *Computer Vision*) o Realidad Virtual (VR, *Virtual Reality*). Esta librería ha sido desarrollada en C++ y no presenta ninguna dependencia externa por lo que su uso se reduce a la adhesión de sus cabeceras al proyecto en cuestión (*HOL, Header Only Library*).

6.3.3. Implementación de una librería de DSP

El lenguaje de programación elegido para este proyecto es uno de los más complejos dado que permite interactuar y combinar con cierta facilidad una programación de bajo/alto nivel con diferentes niveles de abstracción El lenguaje C++ es un lenguaje longevo, robusto y optimizado para procesamiento de datos. Existen un inmenso número de librerías que agilizan el desarrollo de un proyecto pero el conflicto entre licencias debe ser estudiado cuidadosamente.

En este proyecto se requieren de algunos algoritmos para procesamiento de señales acústicas, especialmente aquellos que permitan extraer ciertas características (*FE, Feature Extraction*) que permitan la clasificación y organización de los datos a procesar. Para agruparlos todos de forma independiente y genérica se ha desarrollado una librería de procesamiento de señales digitales (DSP, *Digital Signal Processing*) bautizada como **eDSP (easy DSP)**.

Figura 6.3.4: Logo identificado para la comunidad de software Open Source con licencia MIT

eDSP nace como un proyecto de software libre (*Open Source Project*) liberado bajo la licencia *GPL 2.0,* Figura 6.3.4, una licencia permisiva con unas restricciones limitadas para usos comerciales. eDSP integra un conjunto de algoritmos comúnmente usados por la comunidad de desarrolladores de DSP bajo una interfaz sencilla y portable (*Cross-Platform Library*) optimizada tanto para plataformas de escritorio como embebidas. La librería hace uso de técnicas avanzadas de meta-programación por «templates» (*Templates Metaprogramming*) asi como de las nuevas características del estándar de C++14/17 para permitir la integración de cada una de sus componentes con no solo con los contenedores estándar si no que también con componentes de terceras partes. El proyecto se puede descargar desde el siguiente enlace.

Filtros FIR/IRR

La librería integra una implementación completa de muchos de los filtros analógicos existentes a través de adaptaciones y versiones digitales basadas en filtros FIR. Estos filtros presentan el inconveniente de introducir cierto retardo en el procesamiento debido a su gran número de etapas por lo que se requieren de técnicas avanzadas y costosas para reducir el efecto. Es por ello que en el dominio del procesado de señal acústico los filtros IRR de segundo orden también conocidos como *Biquad* son el principal objeto de estudio. La librería implementa un «framework» completo para la manipulación de este tipo de filtros presentando diferentes estructuras e implementaciones que permiten su adaptación al sistema.

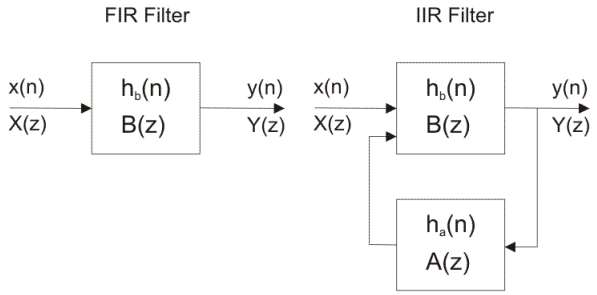

Figura 6.3.5: FIR vs IRR - Diagrama de bloques para la implementación

Efectos acústicos

La librería integra un conjunto de bloques para construir distintos «Audio-Paths» con la finalidad de introducir diferentes efectos en la señal acústica. Entre los mas destacados podemos encontrar un conversor de sonido multi-canal a sonido mono o elementos como el detector de envolvente o efectos de fundido (*Fade In/Out*) 6.3.6. En el repositorio se puede encontrar cada uno de los bloques desarrollados y algunos planteados para futuras implementaciones.

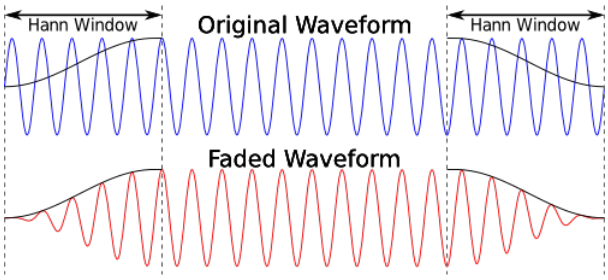

Figura 6.3.6: Ejemplo del efecto Fade In - Fade Out en una señal acustica aplicando un ventana de Hann

Análisis espectral

La librería utiliza el patrón de diseño conocido como *Façade* [9] para implementar una interfaz sencilla para el computo de las transformadas más comunes (FFT, DCT o DHT) haciendo uso de la librería FFTW [8]. FFTW es una librería de software libre que permite de forma optima el cálculo de la transformada discreta de Fourier desarrollada por Matteo Frigo y Steven G. Johnson del Massachusetts Institute of Technology. Esta librería, desarrollada como conjunto de subrutinas en C, es considerada como la librería open-source mas rápida del mercado, Figura 6.3.7, capaz de realizar sus cálculos con una complejidad $O(n \log n)$ con el mínimo coste de operaciones y consumo de memoria.

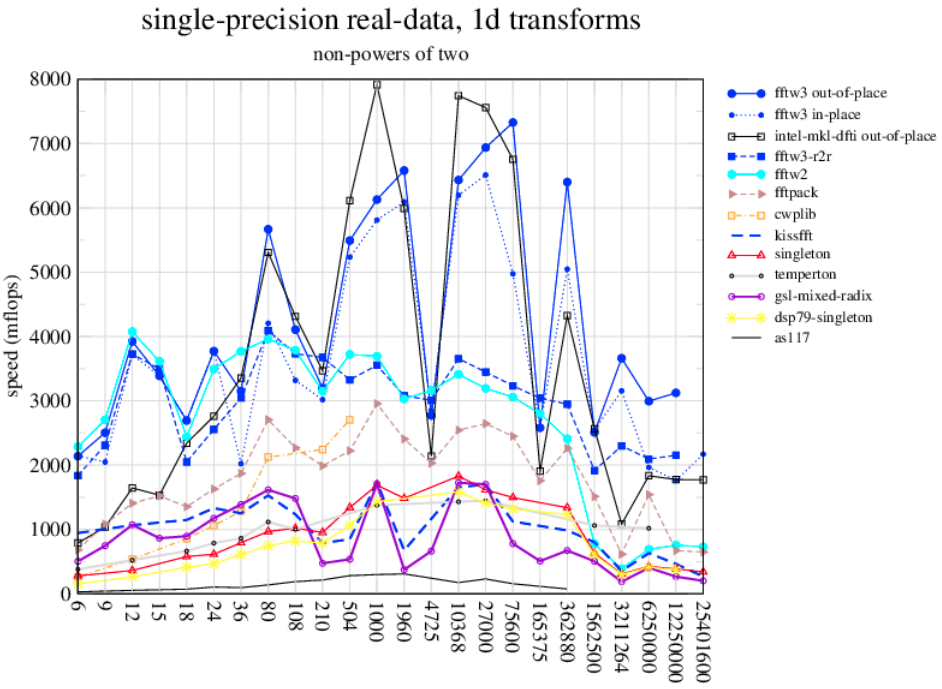

Figura 6.3.7: Benchmark: los resultados de una comparativa de las diferentes librerías para el computo de la FFT

6.3.4. Implementación de la interfaz de usuario

La biblioteca gráfica por excelencia para el desarrollo en C/C++ es la librería Qt por lo que fue la primera elección como entorno desarrollo. La aplicación de escritorio así como la aplicación móvil han sido desarrolladas en su totalidad con el framework Qt, Figura 6.3.8, utilizando el lengauge C/C++, Javascript y QML entre otros. Entre las ventajas principales del uso de este framework podemos encontrar:

Figura 6.3.8: Logo - Qt Framework desarrollado por Digia

- Multiplataforma: Qt es una librería multiplataforma que permite su integración nativa con la mayoría de los sistemas operativos, incluidos sistemas embebidos y móviles.
- Multi-lenguaje: Qt integra una librería para el soporte de diferentes idiomas, Qt Linguist.
- Software Libre: Qt dispone de distintas distribuciones tanto para uso comercial como libre.
- Recursos Reducidos: Qt hace uso de las librerías nativas de cada sistema operativo para reconstruir su entorno gráfico por lo que el consumo de recursos es mínimo.
- Exportación de resultados: Qt incluye librerías nativas para exportar información en los formatos más comunes (XML y JSON).

Junto a Qt, se han utilizado librerías de terceros para la representación de grafos auxiliares como QCustomPlot o OpenGL.

Capítulo 7

Configuración y Calibración del Sistema

El sistema descrito en las secciones anteriores requiere de una calibración y una configuracion previa que permita optimizar y adaptar el sistema para el entorno en el que será instalado. AcousticLPS permite la configuración de la estación base (BS, *Base Station*) de una forma sencilla y trivial a través de la aplicación QAcousticLPS.

Este capitulo pretende ser un manual de usuario inicial así como una guia para procesos futuros. En la Sección 7.1 se describe cada una de las componentes de la aplicación así como la finalidad para la cual fueron desarrolladas. Posteriormente en la Sección 7.2 se describen las pautas seguidas para la configuración y calibración del sistema en un caso real aplicado a un recinto cerrado de varios metros cuadrados. Finalmente, en la Sección **??** se discuten cada uno de los inconvenientes encontrados así como posibles trabajos futuros para mejorar los resultados obtenidos.

7.1. QAcousticLPS

En esta sección se describe la herramienta desarrollada para la configuración así como calibración del sistema AcousticLPS. La herramienta ha sido bautizada como QAcousticLPS en referencia a una nomenclatura común para las aplicaciones Open-Source desarrolladas con el framework Qt. La herramienta ha sido íntegramente desarrollada en el lenguaje de programación C++, usando algunos de los software de terceros previamente descritos para algunas de las componentes.

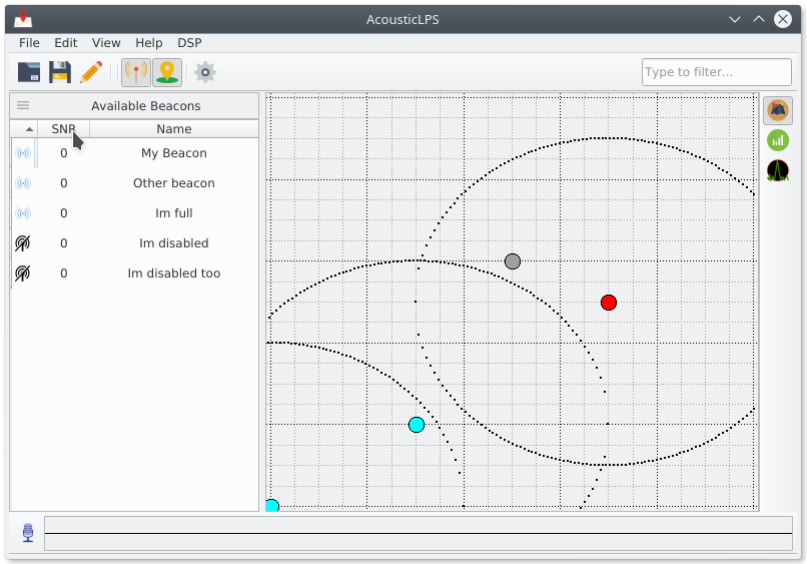

Figura 7.1.1: QAcousticLPS, ventana principal

QAcousticLPS fue diseñada con una idea en mente, una herramienta que pretendía facilitar la contribución de terceros así como su uso y extensión en proyectos futuros por lo que se definieron una serie de objetivos y requisitos tales como:

- **Estabilidad**: la herramienta debiera de funcionar de forma robusta y estable permitiendo una ejecución en tiempo-real sin una excesiva carga del sistema.
- **Flexibilidad**: la herramienta debiera soporta la extensión e integración de funcionalidades y componentes a través de extensiones (plugins) o contribuciones de terceros.
- **Patrones de diseño**: la herramienta debiera ser desarrollada siguiendo una de serie de patrones de diseño modernos y escalables que permitan un mantenimiento y desarrollo sencillo de la aplicación en contribuciones futuras.
- **Multiplataforma**: la herramienta debiera poder ejecutarse en las plataformas más comunes de la actualidad, no solo en plataformas de escritorio sino además en plataformas embebidas como sistemas empotrados o teléfonos móviles (Android, iOS y Windows Phone).
- **Usabilidad**: la herramienta debiera de integrar una interfaz de usuario sencilla y moderna con la finalidad de integrarse fácilmente con otros entornos de trabajo de los usuarios finales.
- **GPL-Licence**: la herramienta debiera de ser liberada bajo las condiciones de la licencia GPL 2.0. Cualquier componente externo utilizado debiera de ser compatible con la misma con el fin de evitar problemas con licencias comerciales o de uso limitado.

La herramienta ha sido liberada y almacenada en un repositorio de GitHub, puede descargarse desde este enlace. Seguidamente se describen cada una de las componentes principales de la interfaz de usuario principal de QAcousticLPS, Figura 7.1.1:

1. **Barra principal** (*Main Toolbar*): como muchas otras aplicaciones, QAcousticLPS cuenta con una barra principal que permite el control total de la aplicación en el despliegue de cada uno de los sub-menus. La Figura 7.1.2 muestra los iconos principales de la aplicación. Entre las opciones disponibles se destacan algunas como:
   a) Abrir archivos de configuración de escenarios: la aplicación permite la carga de configuraciones previamente utilizadas.
   b) Guardar archivos con la configuración del escenario utilizado.
   c) Creación de nuevos escenarios: la aplicación permite la configuración y calibración de un nuevo escenario mediante un dialogo modal.

d) Configuración de la ventana de trilateración que permite la visualización de la posición de cada uno de los nodos conectados al sistema en tiempo real.

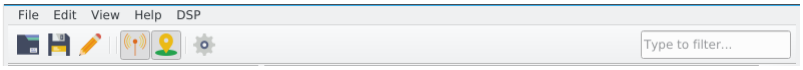

Figura 7.1.2: QAcousticLPS, barra principal (*Main Toolbar*)

2. **Panel de control de nodos** (*Beacon's List Widget*): el panel de la Figura 7.1.3 permite la visualización de cada uno de los nodos que están emitiendo en la red asi como el estado actual. Entre las tareas que se pueden ejecutar encontramos:

   a) Organización de los nodos:

      1) SNR (*Signal-to-Noise-Ratio*): El nivel de SNR es el nivel promedio recibido por todos los usuarios conectados a la red. Esto nos permite analizar el nivel de carga del sistema así como los nodos con una posición estratégica más eficiente.
      2) Nombre: un ordenamiento simple basada en el alias asignado a cada uno de los nodos.
      3) Estado: organización basada en el estado de los nodos, habilitados o deshabilitados.

   b) Deshabilitar/Habilitar la emisión de cada uno de los nodos transmisores.

   c) Selección de un nodo para desplegar una información más detallada del mismo.

3. **Panel de trilateración** (*Trilaterion Widget*): el panel de la Figura 7.1.4 presenta diferentes grafos para el control en tiempo real del sistema. Podemos destacar:

   a) Trayectoria de los usuarios conectados: el sistema presenta en tiempo real la posición de los nodos así como la intersección de las diferentes esferas que conforman la posición estimada del nodo. Esto se realiza a través de la información emitida por cada uno de los usuarios conectados a la red mediante un intercambio de mensajes TCP/IP.

   b) Información espectral: permite establecer una visualización en tiempo real de la respuesta impulsiva del sistema así como de la señal de audio capturada. Esto nos permite analizar la percepción acústica del sistema así como las diferentes componentes que coexisten en el mismo. Su principal utilidad es la identificación de posibles interferencias así como el nivel de ruido en el entorno, Figura 7.1.6.

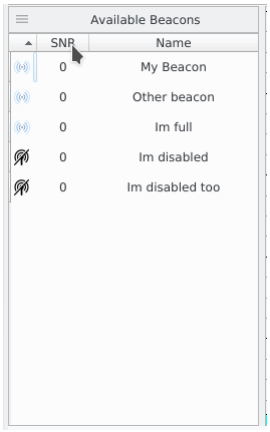

Figura 7.1.3: QAcousticLPS, panel de control de nodos conectados (*Beacon's List Widget*)

c) Información de los correladores: el algoritmo de estimación hace uso de diferentes correladores para estimar la posición del dispositivo. Estos grafos permiten obtener información de la contribución de cada uno de los nodos basándose en la información recibida por cada uno de los usuarios conectados al sistema.

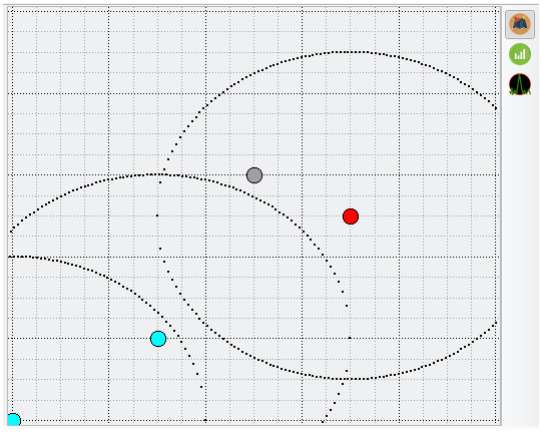

Figura 7.1.4: QAcousticLPS, panel de trilateración (*Trilateration Widget*)

4. **Barra de control inferior** (*Bottom Toolbar*): en la Figura 7.1.5 podemos observar el botón inferior que inicializa el proceso de grabación de las señales sonoras a través de la tarjeta de sonido del dispositivo. El panel visualiza en tiempo real la señal acústica grabada por lo que permite un análisis simple de los niveles de saturación del sistema.

Figura 7.1.5: QAcousticLPS, barra de control inferior (*Bottom Toolbar*)

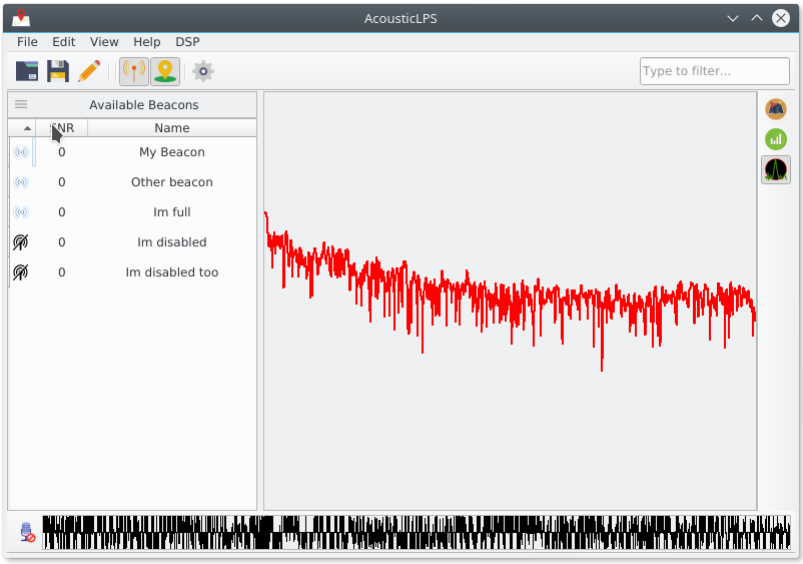

Figura 7.1.6: QAcousticLPS, ejemplo de análisis de la respuesta impulsiva del entorno

7.2. Instalación del sistema

Una de las utilidades más importantes de QAcousticLPS es la flexibilidad para integrarse en nuevos entornos. Esto se consigue a través de una interfaz de configuración de entorno desplegada en un dialogo modal, Figura 7.2.1. El dialogo presenta una interfaz similar al panel principal de control de nodos, Figura 7.1.3 con la diferencia de que este permite añadir nuevos nodos al sistema así como una configuración detallada de las dimensiones del entorno.

El sistema permite un ajuste y calibración hasta niveles centimétricos. A continuación se explican cada uno de los pasos seguidos para la configuración y calibración del entorno:

1. **Medida del entorno**: como primer paso se requiere la obtención precisa de las dimensiones del entorno. QAcousticLPS no permite superficies complejas por lo que supone que el recinto presenta una forma cúbica. La medición se realizo mediante el uso de un medidor de distancia láser (modelo Suaoki D60) con un alcance de hasta 60m y una precisión milimétrica, Figura 7.2.2.

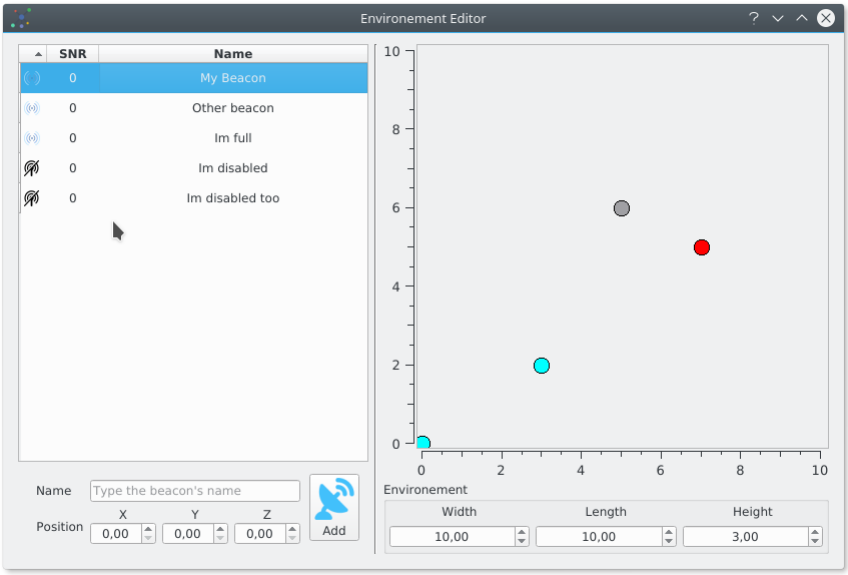

Figura 7.2.1: QAcousticLPS, dialogo modal para la configuración del entorno

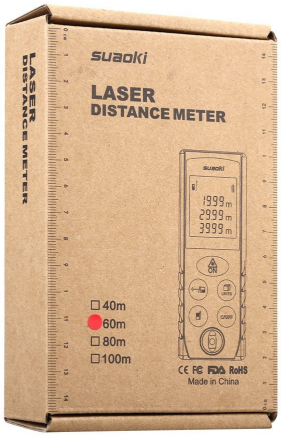

Figura 7.2.2: Medidor de distancia laser Suaoki D60

Para un medida precisa se requiere de una visión directa entre los dos puntos a medir por lo que la presencia de obstáculos puede dificultar el proceso. Se ha optado por el uso de una nave industrial situada en la zona del poniente Almeriense generalmente utilizada para cultivo y almacenaje

de material agrícola, Figura 7.2.3.

En un principio el recinto estaba fuera de servicio por lo que el numero de obstáculos era reducido facilitando esta etapa del proceso. El inconveniente principal del recinto es que este no presentaba un aislamiento sonoro óptimo. Su situación próxima a una carril agrícola activamente transitado hacia de este inconveniente un problema mayor. Como solución preliminar las pruebas de campo se hicieron en horarios de poca actividad laboral para reducir el número de perturbaciones externas aunque finalmente se utilizo un acristalamiento típicamente reservado para épocas frías del año para aislar el recinto.

Figura 7.2.3: Almacén agrícola utilizado para las pruebas de campo

2. **Despliegue de los nodos**: esta etapa es la más importante. Los nodos transmisores deben de colocarse en zonas estratégicas que permitan reducir los efectos no deseados descritos en secciones anteriores. Generalmente el sistema presentará mejores prestaciones cuanto mayor sea el numero de nodos disponibles. En el caso en cuestión se ha utilizado un sistema de altavoces 5.1 que permite el despliegue de 4 nodos simultáneamente. El sistema utilizado es un modelo low-cost, concretamente el modelo Inspire T6300, Figura 7.2.4, con una potencia eléctrica promedio de 57W.

   Los cables Jacks suministrados por defecto presenta una longitud promedio de 1.5m. Las dimensiones del recinto eran superiores por lo que se requiere del uso de cables más extensos. Los utilizados presentaban una longitud de 6.3m del fabricante Regai, Figura 7.2.5.

   La selección de la posición de cada uno de los nodos debe de optimizar el área de cobertura evitando zonas nulas que compliquen el rastreo de

Figura 7.2.4: Sistema de altavoces 5.1, Inspire T6300

Figura 7.2.5: Cable Jack del fabricante Regai utilizado para distribuir los diferentes nodos

los dispositivos. Para el uso de un equipo de altavoces manejando cada una de las pistas de forma independiente se requiere de una tarjeta de sonido multicanal. Los equipos portátiles por regla general solo disponen de una salida «jack stereo» por lo que se requiere de una tarjeta de sonido externa.

La opción seleccionada presenta unas buenas condiciones en relación calidad-precio. Se ha optado por una tarjeta de sonido 7.1 con conector USB 2.0 del fabricante CSL. La ventaja de esta tarjeta es que presenta una interfaz Plug & Play que evita la necesidad de instalar drivers externos que podría complicar el uso de la misma con plataformas no estandarizadas en la industria musical. Por otro lado, la tarjeta de sonido permite unas buenas cualidades que evitarán la necesidad de sincronizar los diferentes altavoces permitiendo una triangulación esférica que simplifica el proceso de localización de los dispositivos.

3. **Selección de pistas**: Una vez instalado el sistema se necesita seleccionar los códigos que serán enmascarados con las pistas de audio utilizadas. Las

Figura 7.2.6: Tarjeta de sonido 7.1 del fabricante CSL

diferentes técnicas existentes así como la implementación de las mismas queda pendiente para futuras versiones de QAcousticLPS. Actualmente se han utilizado herramientas de terceros para mezclas las pistas así como para reproducirlas. Por ser de uso libre y multiplataforma se ha optado por el uso de Audacity para la reproducción de las diferentes pistas en cada una de las salidas independientes. Para la mezcla se han utilizado scripts en lenguaje M, ejecutables tanto en MATLAB como en la versión equivalente Open-Source, GNU Octave.

7.2.1. SoundMaps

El sistema pretende estimar la posición de un dispositivo en la red de sensores en tiempo real. Para una visualización y computo de la posición del mismo era necesario el desarrollo de una herramienta que pudiera ejecutarse en plataformas móviles. Esta herramienta ha sido bautizada como SoundMaps, Figura 7.2.7, en referencia a otras aplicaciones similares como GoogleMaps o Apple Maps. La herramienta ha sido desarrollada en C++ utilizando el framework Qt, análogamente a su hermana QAcousticLPS.

El framework Qt presenta dos alternativas para el diseño de interfaces:

- Interfaces desarrolladas en C++/XML: estas interfaces son las clásicas desarrolladas en Qt, esencialmente utilizadas en plataformas de escritorio.
- Interfaces desarrolladas en QML/JavaScript: los dispositivos móviles generalmente disponen de pantallas táctiles que hacen que el diseño de las aplicaciones sea muy diferente a las de una aplicación de escritorio. Qt integra estas funcionalidades en un lenguaje propia bautizado como QML (*Qt Modeling Language*).

Como el desarrollo de esta aplicación tiene como finalidad su ejecución en plataformas móviles se ha optado por esta última opción para el diseño de la aplicación. Paralelamente, se han utilizado diferentes cross-compiladores para compilar el código en las diferentes plataformas. Esto evita la necesidad de duplicación de código utilizando diferentes librerias nativas del propio SO en cuestión.

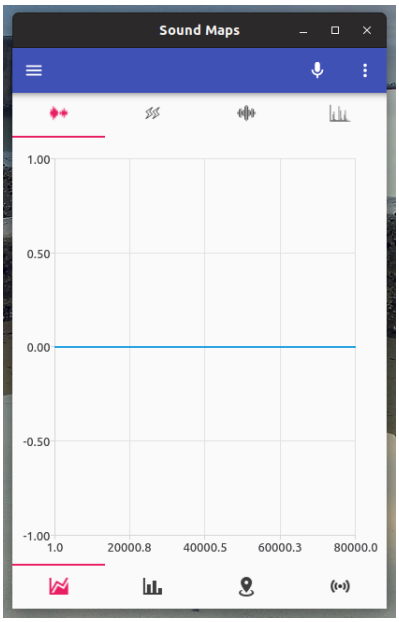

Figura 7.2.7: Ventana principal de la aplicación SoundMaps

Configuración de los dispositivos conectados

La configuración del sistema es una labor exhaustiva que requiere de unos ajustes precisos de algunos aspectos como el ancho de banda dedicado así como los códigos utilizados [3].

La enmascaración de estos códigos se realiza a través de cualquier herramienta de audio. En este caso, para una facil configuración se ha utilizado la herramienta MATLAB con la extensión Audio Toolbox dedicada expresamente al procesado de señales acústicas.

La herramienta QAcousticLPS permitirá definir con mayor precisión las dimensiones del entorno, las condiciones así como la posición de cada uno de los nodos fijos. Esta herramienta generara un mapa con toda la información que debera de ser cargado en cada una de los dispositivos conectados a la red.

En la Figura 7.2.7 podemos encontrar la ventana principal de la aplicación SoundMaps que procesara este mapa. La aplicación presenta por defecto una configuración básica del proceso de captura de sonido pero en muchas ocasiones por las limitaciones del teléfono móvil así como las condiciones del sistema es

necesario una configuración previa. En la Figura 7.2.8 podemos encontrar las diferentes opciones para la configuración del sistema:

1. Selección de la tarjeta de sonido: algunos dispositivos presentan diferentes tarjetas de sonido cada una con diferentes cualidades. La aplicación permite seleccionar la tarjeta de sonido que se pretende utilizar para la captura de datos. Esto permite el uso de tarjetas de sonido externas en pruebas experimentales durante el diseño del sistema.

2. Selección de la frecuencia de muestreo: en muchas ocasiones no se requiere una frecuencia de muestreo elevada para la captura de datos ya que la mayoría de información la encontramos en bajas frecuencias. La aplicación permite ajustar la frecuencia de muestreo permitiendo entre muchas otras cosas, la reducción de los costes de computo a frecuencias inferiores.

3. Selección de la frecuencia de visualización: las plataformas móviles presentan limitaciones en la tarjeta gráfica. En muchas ocasiones, esta no esta disponible por lo que los cómputos gráficos se realizan en la CPU. Este hecho ralentiza el sistema por lo que la aplicación permite reducir la frecuencia de refresco de las diferentes visualizaciones para paliar los problemas derivados.

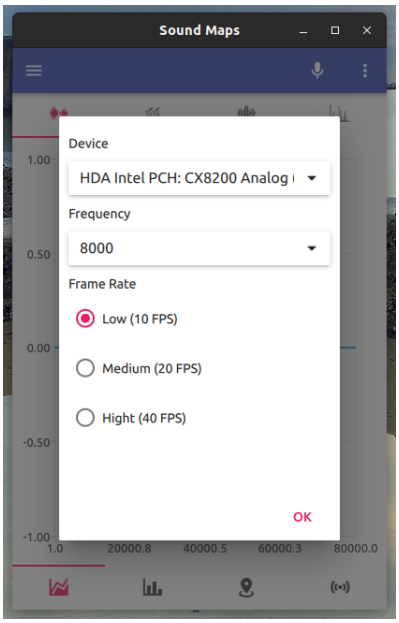

Figura 7.2.8: Ventana de configuración de SoundMaps

Procesamiento de datos

Tal y como ofrece QAcousticLPS, SoundMaps permite una visualización en tiempo real de algunos de los cómputos realizados internamente para la estimación de la posición del dispositivo:

- Señal temporal: la aplicación visualiza dinámicamente la señal capturada con el dispositivo. Esto permite analizar los niveles de saturación o ruido en el sistema.
- Energía: la representación de la evolución energética de la señal capturada es de vital importancia para evaluar el sistema.
- Correlación: representa la señal de correlación con los diferentes correladores usados internamente en la aplicación. Esto permite evaluar cuales de los nodos emisores contribuye en mayor cantidad a la estimación de la posición así como detectar posibles zonas nulas fuera de cobertura.
- Espectro: la representación espectral de la señal permite analizar la respuesta en frecuencia del sistema así como detectar posibles interferencias,

Este característica viene integrada para optimizar las pruebas de campo; sirve como apoyo para la evaluación de la carga del sistema, optimización de recurros y análisis de las estimaciones. En la Figura 7.2.9, podemos ver la visualización de la señal de audio capturada en tiempo real así como el computo de la respuesta en frecuencia.

7.2.1.1. Visualización de resultados

SoundMaps se encuentra aún bajo desarrollo por lo que no todas las funcionalidad han sido desarrolladas en su totalidad. A día de hoy, entre las cualidades que presenta la aplicación por norma general podemos encontrar:

1. Visualización de cómputos: en la sección anterior se mostraban algunos de los resultados de los cómputos realizados internamente para evaluar el sistema.
2. Visualización de los nodos disponibles: la aplicación conoce la localización así como los códigos asignados a cada uno de los nodos conectados a la red. Con esta información se puede estimar cuales de los nodos que están emitiendo en la red están contribuyendo a la estimación de la posición. Basándonos en los valores extraídos a través de los correladores se puede estimar información es de vital importancia para diferentes tareas:
   a) Detección de zonas nulas sin cobertura.
   b) Redistribución de los nodos para mejorar el rendimiento del sistema.
   c) Evaluación de diferentes códigos durante el proceso de enmascaración

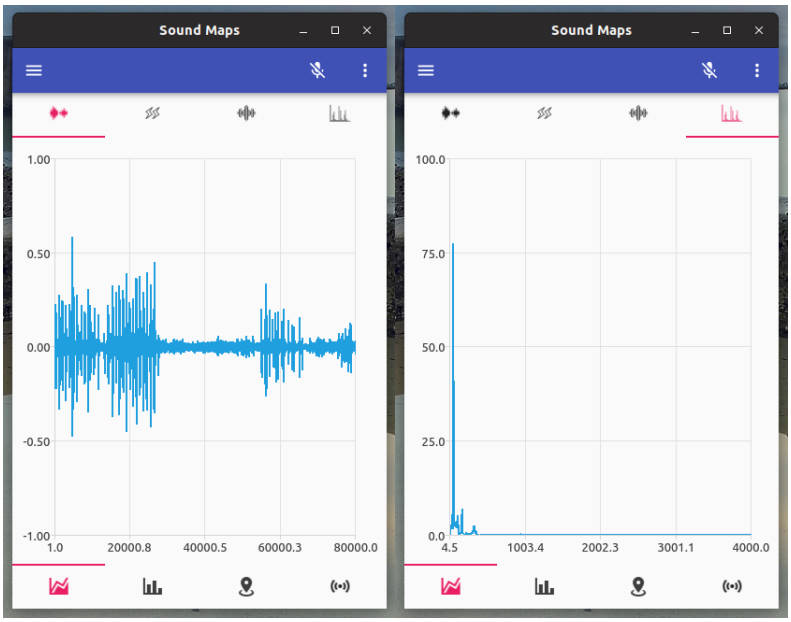

Figura 7.2.9: Computos realizados por la aplicación SoundMaps

3. Visualización de la señal de GPS: mediante el uso de GPS se visualiza la posición global del sistema, Figura 7.2.10. Eso es útil al conectarse al servidor central para descargar diferentes mapas y recintos previamente calibrados y configurados en el entorno. Como se definió en secciones anteriores, la idea es que AcousticLPS sea un sistema escalable y extensible por lo que una configuración remota puede ser integrada a través de este modulo.

4. Visualización de estadísticas de la red: esta sección quedará presentada para trabajos futuros. El servidor central podrá recolectar información de todos los nodos conectados a la red y devolver diferentes estadísticas de interes a los usuarios conectados. Esto puede ser interesante para evitar aglomeraciones en ciertos recintos o simplemente para localizar a otros individuos conectados a la red.

7.3. Rendimiento del sistema

En la sección anterior se introduce la aplicación desarrollada para estimar la posición del cliente en tiempo real. Esta aplicación aun se encuentra en desarrollo por lo que para las pruebas de campo se ha optado por estimar la posición, exportar los resultados en un archivo de texto y realizar una posterior visualización de los resultados. Para la evaluación se han trazado diferentes rutas en

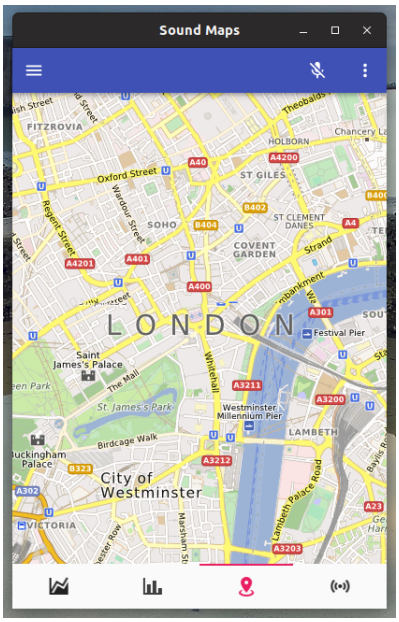

Figura 7.2.10: Visualización de la posición del GPS en SoundMaps

el recinto que posteriormente se han seguido por un simple robot seguidor de línea, Figura 7.3.1.

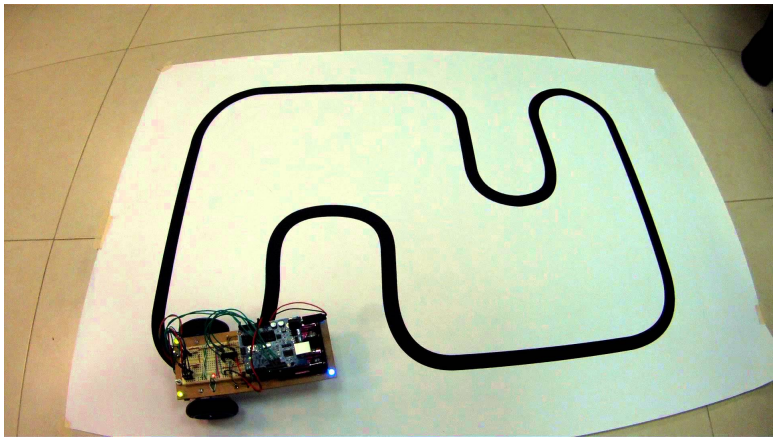

Figura 7.3.1: Robot seguidor de linea

El robot utilizado es capaz de realizar unos movimientos muy precisos en el trazado por lo que es perfecto para la evaluación del sistema. El robot portará un pequeño terminal Android que correrá el sistema la aplicación que estimará

las posiciones con un cierto periodo de estimación por minuto, T_e. Se han hecho varios trazos del mismo recorrido para realizar una ponderación estimada del error. En la Tabla 7.4 podemos ver los resultados obtenidos así como la precisión del sistema.

T_e (puntos/minuto)	Precisión
500	62,75 %
1000	77,85 %
3000	78,75 %
5000	83,75 %

Cuadro 7.1: Resultados obtenidos para una evaluación de diferentes puntos en el sistema

Como podemos ver existe una relación entre el frecuencia de estimación (número de puntos a considerar) y el trazado final estimado. A mayor número de puntos, mayor precisión. El inconveniente radica en las limitaciones de procesamiento de los dispositivos móviles por lo que se tendrá que buscar un compromiso entre la frecuencia de estimación y la precisión deseada. En la sección sucesora se introducen algunas técnicas para mejorar el rendimiento del sistema incluso a bajas frecuencias de estimación.

7.4. Mejoras en la estimación - Postprocesamiento

En la Figura 7.4.1 podemos ver una de las rutas estimadas por el sistema en una de las pruebas experimentales. Si se analiza la imagen con detalle, se pueden distinguir diferentes zonas erráticas que requieren de una corrección y tratamiento para una mejora en la estimación de la ruta final. Se determina entonces que es necesario implementar un algoritmo que sea capaz de detectar las zonas con mayor erraticismo y aplicar diferentes técnicas para paliar los efectos en los resultados finales.

Si partimos de la hipótesis de que la aplicación será usada en un sistema de «tracking» de peatones podemos adaptar los resultados aplicando diferentes técnicas de post-procesamiento previamente utilizadas en otros proyectos similares. La idea principal radica en que el ser humano es vago por naturaleza; por regla general siempre intenta buscar la ruta más rapida y corta de desplazarse entre dos puntos por regla general a una velocidad constante. Por consiguiente, los *movimientos aleatorios no forman parte de la ruta común dentro de las trayectorias genéricas de un peatón.*

Teniendo en cuenta este factor podemos detectar las zonas erráticas como aquellas cuyos puntos definan una trayecto con cambios en la velocidad así como en la dirección del movimiento que no estén correlados con sus predecesores. Si aplicamos un filtro MA (*Moving Average Filter*) causal que analize la validez de cada estimación en función de las anteriores, Figura 7.4.2, podremos estimar

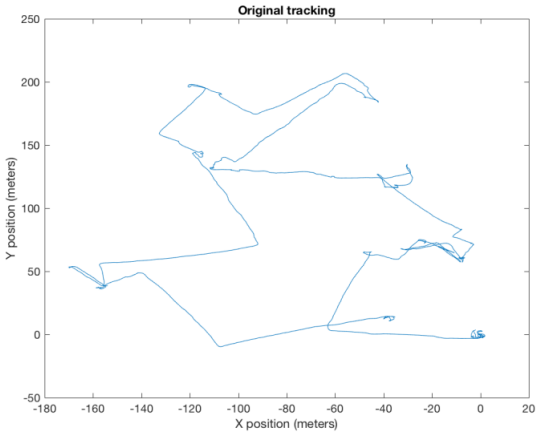

Figura 7.4.1: Prueba de campo: ruta estimada por el sistema el sistema

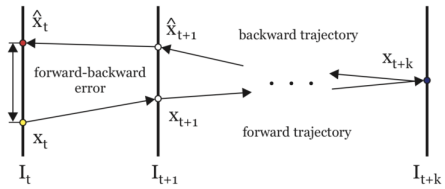

Figura 7.4.2: Filtro aplicado para la detección de zonas erráticas

aquellas zonas cuyo erraticismo es mayor considerando la premisa de que velocidad de desplazamiento debiera ser constante. Podremos entonces comparar la velocidad instantánea, v_i, con la media ponderada estimada de los puntos predecesores, $\hat{v}_i$, para estimar el erraticismo ϵ_i según la ecuación (7.4.1) usando la función SMAPE (**??**).

$$\epsilon_i = \frac{\sum_{i=0}^{N} |v_i - \hat{v}_i|}{\sum_{i=0}^{N} (v_i + \hat{v}_i)} \tag{7.4.1}$$

Si además considerando el error de BIAS así como posibles derivados de la instrumentación utilizada en el sistema, podremos estimar las zonas de erraticismo como aquellas que superen un umbral de tolerancia $\Delta\epsilon$, Figura 7.4.3.

$$\epsilon_{BIAS} = \frac{1}{N} \sum_{i=0}^{N} \epsilon_i \tag{7.4.2}$$

Aquellas zonas con alto nivel de erraticismo podran ser descartadas mientras que aquellas que presenten niveles moderados podrán ser ajustadas mediante un filtro de «smoothing». En la Figura 7.4.4 y 7.4.5 podemos observar las diferentes

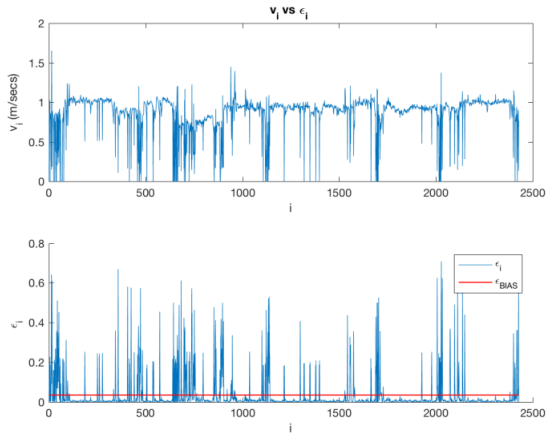

Figura 7.4.3: Erraticismo estimado: las zonas de erráticas podemos observar como se producen cambios drásticos en la velocidad del desplazamiento.

zonas erráticas detectadas aplicando diferentes umbrales de erraticismo. Vemos como un nivel de tolerancia muy bajo puede dar lugar a falsos positivos y a una excesiva corrección de la ruta. Es por tanto necesario adaptar el umbral a las condiciones del entorno para intentar reducir en la medida de lo posible todas las erratas que podrían perturbar el rendimiento final del sistema.

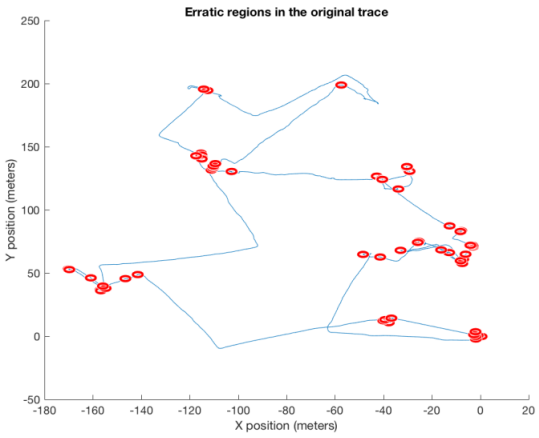

Figura 7.4.4: Zonas erráticas en el trazo original usando un umbral de erraticismo de un 30 %

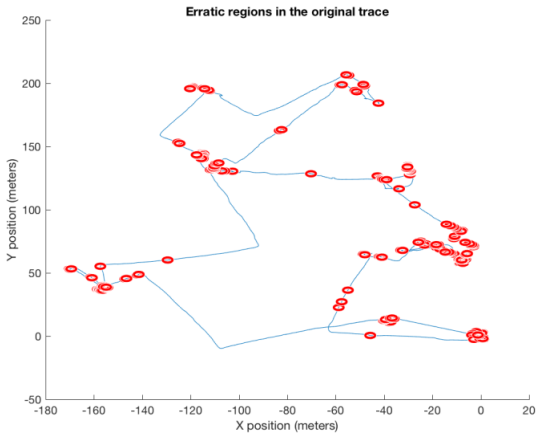

Figura 7.4.5: Zonas erráticas en el trazo original usando un umbral de erraticismo de un 10 %

Capítulo 8

Evaluación de resultados

La herramienta ha presentado un comportamiento correcto en las distintas plataformas en las que se testeó durante cada una de las pruebas de campo, tanto en dispositivos fijos como sistemas móviles y empotrados. Algunas de las plataformas testeadas junto al rendimiento obtenido se pueden encontrar en la Tabla 8.1.

Sistema Operativo	S. QAcousticLPS	S. SoundMaps	Procesador
Windows 7	SI	SI	Intel
Windows 10	SI	SI	Intel
Ubuntu 16.10 LTS	SI	SI	Intel
Mac Os X	SI	SI	Intel
iOS (iPhone)	-	SI	A6
iOS (iPad)	-	SI	A6
Android 6.0	SI	SI	ARMv7
Android Car	SI	SI	ARMv8
Raspberri Pi III	-	SI	ARMv7

Cuadro 8.1: Plataformas testeadas: SI (testeado y ejecución satisfactoria), NO (Testeado pero ejecución fallida), - (No testeado)

En términos de rendimiento, los cómputos necesarios para la estimación de un objeto así como para su representación en un espacio bidimensional se ejecutan de forma robusta, eficiente y con un mínimo de retardos incluso en aquellos dispositivos con unas prestaciones más reducidas. Esto permite la monitorización y análisis del sistema en tiempo real. Por otro lado, las interfaces de usuario, sencillas y modernas a la par, permiten una configuración y calibración del sistema trivial y eficaz. El sistema fue diseñado desde un principio para ser integrado de forma sencilla en infraestructuras audio-visuales previamente desplegadas por lo que este puede ser considerado uno de los puntos fuertes del mismo.

En lo que respecta a evaluación del rendimiento del sistema en términos de precisión en las estimaciones, la realización de cada una de las pruebas de campo, aunque limitadas, desglosan en unos resultados experimentales satisfactorios abriendo las puertas para futuras investigaciones en el campo. El análisis temporal y espectral de las señales acústicas ha permitido la optimización y análisis del entorno permitiendo un ajuste y adaptación del sistema más preciso.

Cabe destacar la gran capacidad de procesamiento de SoundMaps que al ser desarrollada en C++ elimina alguna de las barreras de los lenguajes nativos de plataformas como Android, donde el lenguaje de programación JAVA presenta ciertas limitaciones para el procesamiento de datos en tiempo real. El hecho de que ambas plataformas sean liberadas como proyectos de software libre abre las puertas a un futuro venidero donde se esperan posibles contribuciones de terceras partes y de otros investigadores.

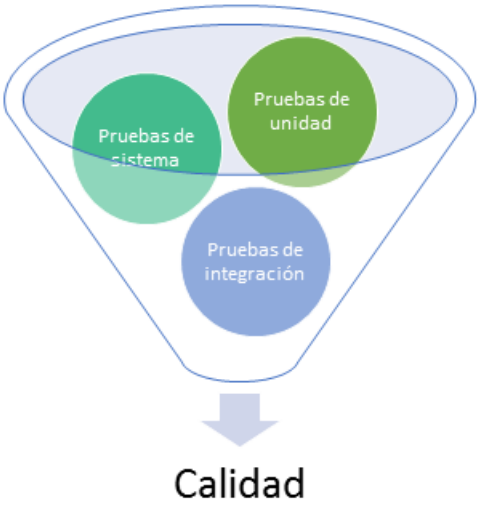

Figura 8.0.1: Control de calidad de la herramienta

8.1. Conclusiones

El proyecto desarrollado presenta una perspectiva más cercana al desarrollo y la innovación que al de investigación en lo que respecta a I+D+i. Se basa fundamentalmente en la uso de conocimientos científicos sobre procesado de señal acústica para ofertar una solución de ingeniería para un determinado problema. Es por ello que las conclusiones que se pueden extraer se aferran más al éxito de la consecución de los objetivos así como en la experiencia adquirida al abordar el problema para posibles proyectos futuros.

El proyecto se centra en buscar una solución a las necesidades que presentan ciertos sectores de la población para la orientación y movilidad en espacios interiores. Se desarrolló bajo la premisa de construir un sistema de posicionamiento en interiores que no requiriera de un despliegue tecnológico costoso y que pudiera hacer uso de instalaciones existentes. La idea esencial es hacer un uso de conceptos técnicos sobre procesado de señal acústica, ingeniería de software y tecnologías de telecomunicación para diseñar e implementar el sistema para posteriormente mediante ciertas pruebas de campo analizar la viabilidad del mismo.

Las pruebas de campo demuestran un futuro venidero para el campo de estudio en cuestión. El sistema ofreció rendimientos excelentes en tiempo real, siendo este aun un simple embrión de lo que pudiera llegar a ser. Algunos de los usos destacables de la aplicación:

- Uso como sistema de posicionamiento en entornos industriales: el sistema puede utilizarse para localizar la maquinaria industrial, empleados y entes participantes en el proceso de fabricación con el fin de mejorar, asegurar y monitorizar el mismo.

- Uso como sistema de posicionamiento y localización de personas en determinados servicios a la población: con la premisa de una necesidad de rastreo y monitorización de cierto sector de la población, especialmente aquellos con ciertas movilidad reducidas, se podría instalar el sistema para controlar y asistir en caso de necesidad a cada uno de los usuarios conectados al sistema.
- Uso como sistema de posicionamiento en grandes almacenes: los grandes almacenes cada vez son más extensos ofreciendo un amplio abanico de posibilidades ne todo el recinto. Se podría utilizar el sistema para monitorizar a todos los usuarios permitiendo tanto el uso de la aplicación como guía dentro del recinto como para aplicaciones de Big-Data donde se podrían ofrecer ofertas y servicios adaptados a cada cliente.
- Uso como sistema de posicionamiento para personas invidentes que con asistencia acústica podría hacer de guía en determinados recintos.
- Uso como herramienta de apoyo para estudios relacionados con sistemas de posicionamiento en interiores.

Como se puede ver, existen un gran número de aplicaciones. Determinamos entonces que el desarrollo del proyecto ha alcanzado los objetivos planteados a priori. El sistema funciona, no requiere de un gran coste infraestructural para su instalación, ha sido liberado bajo una licencia Open-Source y presenta todas las cualidades para una investigación más profunda en un futuro cercano. Finalmente, y no por ello menos importante, las aplicaciones desarrolladas han recibido una fuerte acogida tanto por los usuarios como por la comunidad de desarrolladores que está dispuesta a contribuir en la comunidad de GitHub especialmente las funcionalidades relacionadas con procesamiento digital de señales.

8.2. Trabajos futuros

El sistema planteado fue diseñado para ser escalable y extensible ya que desde un principio se consideró una posible creciente subida del número de usuarios conectados a la red de sensores así como de las funcionalidades que este podría integrar. Este factor es de vital para comprender el diseño planteado del sistema así como algunas de las futuras mejoras y extensiones del sistema:

- Escalabilidad: desde un principio se barajó la posibilidad de un escalado exponencial del número de usuarios conectados a la red. Esto hace que la cantidad de «hosts» sea escalable. El servidor planteado que monitoriza el sistema puede extenderse usando tecnologías mas modernas como sistemas distribuidos o sistemas multicore.
- Extensibilidad: los frameworks diseñados son robustos, eficaces y extensible permitiendo la adhesión de extensiones de terceros. Esta propiedad permite añadir funcionalidades al sistema sin una necesidad de reestructuración y rediseño.

- Flexibilidad: los sistemas han sido diseñados utilizando las características propias de los estándares de C++. El framework principal puede compilarse y ejecutarse en prácticamente cualquier plataforma que soporte un compilador de C++ moderno, véase la Tabla 8.1 para los diferentes casos testeados. Esto permite la integración del sistema en nuevos dispositivos tales como relojes inteligentes o gafas inteligentes.

- Rendimiento: el sistema se ha diseñado utilizando algunas funcionalidades que mejoran el rendimiento de los diferentes algoritmos a muy bajo nivel. Esto se consigue utilizando las comúnmente llamadas funciones intrínsecas del propio procesador (*CPU, Central Processing Unit*). En trabajos futuros esto se podría mejorar adaptando estas mismas a cada una de las plataformas en las que el sistema sera ejecutado. Por otro lado existe la posibilidad de ejecutar algunos de los cómputos más costosos en la tarjeta gráfica (*GPU, Graphics Processing Unit*) del sistema que integra algunas mejoras para cálculos algebraicos.

- Investigación: los algoritmo utilizado para estimar la TOA no es el más robusto del mercado. Existen alternativas que pueden paliar algunos de los efectos que introducen errores en el sistema. Una posible mejora del sistema consideraría paliar algunos de los efectos derivados de la reverberación o efecto multi-canal del recinto así como errores derivados de consideraciones tomadas a priori. La integración de un sensor de temperatura y humedad ayudaría a obtener una medida más precisa de la velocidad de propagación del aire. Por otro lado, un sensor de presión así como de velocidad del viento podría mejorar las estimaciones finales obteniendo una mejora en la precisión final del sistema.

- Integración: el sistema pudiera ser integrado junto a otros sistemas existentes para mejorar el rendimiento final. En la sección anterior se discutieron algunas de las posibles utilidades del sistema. Estas eran unas pocas entre muchas tantas, por lo que se podrían hacer pruebas de campo para ver la robustez del sistema en otras aplicaciones.

Como podemos ver existe un amplio abanico en las posibles aplicaciones del sistema así como en posibles mejoras. AcousticLPS ha demostrado ser un sistema viable con un gran número de aplicaciones a un coste de desarrollo e instalación relativamente bajo para los rendimientos obtenidos.

Como nota final se invita a la comunidad de investigadores a contribuir ya sea mediante la publicación de extensiones del sistema como con posibles sugerencias para la mejora del mismo.

Capítulo 9

Presupuesto

En este apartado se presentan las justificaciones de todos los costes globales que han supuesto la investigación, desarrollo y evaluación del presente proyecto. Incluye los costes imputables tanto a personal humano como a material informático requerido. Se toma como referencia el salario personal de un Ingeniero Junior en España de 1894.39€ para 131.25 horas mensuales. Análogamente siendo:

- A: dedicación en número de meses o fracción.
- B: periodo de depreciación en los que se amortiza el equipo
- C: Coste del software sin IVA.

El coste imputable para el material informático se puede obtener como:

$$P_i(€) = \frac{A \cdot C}{B}$$

Fase	Dedicación (h)	Coste I. Junior (Sueldo/mes)	C. I. (€)
Investigación	120	1894,39	1690,51
Desarrollo	400	1894,39	5635,04
Evaluación	130	1894,39	1831,88
Redacción de informe	50	1894,39	704,38
Horas totales	700	1894,39	9861.13

Cuadro 9.1: Fases de elaboración de proyecto y horas invertidas

	C. Licencia (€)	Dedicación (h)	Depreciación (Meses)	C. I. (€)
M. Office 2013	94,00	10	24	0,05
MVS 2013	387,00	50	60	0,44
Matlab R2017a	2000,00	75	60	3,47
QtCreator	0,00	350	60	0,00
TexMaker	0,00	40	24	0,00
Tarjeta de Sonido	25,00	15	24	0,02
Cables Jack	71,60	15	36	0,06
Altavoces 5.1	59,95	15	36	0,05
Metro Laser	25,50	2	24	0,02
Coste Total	2481,00	525	-	4,11

Cuadro 9.2: Material informático utilizado

Concepto	Coste Imputable
Desarrollo	9861,13 €
Software	4,11 €
Manutención	700,00 €
Local de investigación	1840,00 €
Total	12767,13 €

Cuadro 9.3: Costes totales para el desarrollo de LogoSpeech Studio

Bibliografía

[1] Tiobe index. `https://www.tiobe.com/tiobe-index/`. Accessed: 2017-06-03.

[2] D. F. Albuquerque, J. M. N. Vieira, S. I. Lopes, T. Aguilera, and F. J. Álvarez. Doppler resilient modulation in a cdma-based acoustic local positioning system. In *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, Oct 2016.

[3] M. C. Bartholomew-Biggs and A. B. Forbes. A two-dimensional search used with a nonlinear least-squares solver. *Journal of Optimization Theory and Applications*, 104(1):215–234, Jan 2000.

[4] Ross Bencina and Phil Burk. Portaudio - an open source cross platform audio api. In *ICMC*, 2001.

[5] O. Bunting and D. Chesmore. Time frequency source separation and direction of arrival estimation in a 3d soundscape environment. *Applied Acoustics*, 74(2):264 – 268, 2013. Applied Soundscapes: Recent Advances in Soundscape Research.

[6] Lin Chen, Yongchun Liu, Fancheng Kong, and Na He. Acoustic source localization based on generalized cross-correlation time-delay estimation. *Procedia Engineering*, 15:4912 – 4919, 2011. CEIS 2011.

[7] E. Doukhnitch, M. Salamah, and E. Ozen. An efficient approach for trilateration in 3d positioning. *Computer Communications*, 31(17):4124 – 4129, 2008.

[8] M. Frigo and S. G. Johnson. The design and implementation of fftw3. *Proceedings of the IEEE*, 93(2):216–231, Feb 2005.

[9] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.

[10] ©Henri P. Gavin. The levenberg-marquardt method for nonlinear least squares curve-fitting problems. 2013.

[11] F. X. Ge, D. Shen, Y. Peng, and V. O. K. Li. Super-resolution time delay estimation in multipath environments. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 54(9):1977–1986, Sept 2007.

[12] M.J.C. Gover. Comparative review of applied linear algebra, 3rd ed., by b. noble and j.w. daniel and linear algebra and its applications, 3rd ed., by g. strang. *Linear Algebra and its Applications*, 118:159 – 161, 1989.

[13] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[14] Mike Hazas and Andy Ward. A novel broadband ultrasonic location system. In *Proceedings of the 4th International Conference on Ubiquitous Computing*, UbiComp '02, pages 264–280, London, UK, UK, 2002. Springer-Verlag.

[15] Jack K. Holmes. *Spread Spectrum Systems for GNSS and Wireless Communications*. Artech House, Inc., Norwood, MA, USA, 1st edition, 2007.

[16] Hyperphysics. http://hyperphysics.phy-astr.gsu.edu/hbase/acoustic/revtim.html. 2005.

[17] Parag C. Pendharkar, James A. Rodger, and Girish H. Subramanian. An empirical study of the cobb douglas production function properties of software development effort. *Information and Software Technology*, 50(12):1181 – 1188, 2008.

[18] M. C. Perez, J. Urena, A. Hernandez, C. de Marziani, A. Jimenez, J. M. Villadangos, and F. Alvarez. Ultrasonic beacon-based local positioning system using loosely synchronous codes. In *2007 IEEE International Symposium on Intelligent Signal Processing*, pages 1–6, Oct 2007.

[19] J. C. Prieto, A. R. Jimenez, J. Guevara, J. L. Ealo, F. Seco, J. O. Roa, and F. Ramos. Performance evaluation of 3d-locus advanced acoustic lps. *IEEE Transactions on Instrumentation and Measurement*, 58(8):2385–2395, Aug 2009.

[20] E Richard Robinson and Azizul H. Quazi. Effect of sound-speed profile on differential time-delay estimation. 77:1086–1090, 03 1985.

[21] William S. Murphy and Willy Hereman. Determination of a position in three dimensions using trilateration and approximate distances. 01 2000.

[22] B. Schaffer, G. Kalverkamp, and E. Biebl. A 2.4 ghz high precision local positioning system based on cooperative roundtrip time of flight ranging. In *GeMiC 2014; German Microwave Conference*, pages 1–4, March 2014.

[23] Bertrand T. Fang. Trilateration and extension to global positioning system navigation. 9:715–717, 11 1986.

[24] G. Turin. An introduction to matched filters. *IRE Transactions on Information Theory*, 6(3):311–329, June 1960.

[25] M. Yassin and E. Rachid. A survey of positioning techniques and location based services in wireless networks. In *2015 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, pages 1–5, Feb 2015.